ADOBE® COLDFUSION® 10
CFML リファレンス

法律上の注意

コンテンツ

第 5 章：AJAX JavaScript 関数

第 8 章：Application.cfc リファレンス

第 9 章：ColdFusion イベントゲートウェイリファレンス

第 12 章：WDDX JavaScript オブジェクト

第 13 章：ColdFusion ActionScript 関数

第 1 章：はじめに

『CFML リファレンス』は、CFML (ColdFusion Markup Language) の主要リファレンスガイドです。このマニュアルでは、CFML タグと関数、ColdFusion 式、および Adobe® ColdFusion® 9 における WDDX の JavaScript オブジェクトの使い方について説明しています。Java™ および C++ CFX インターフェイスに関する詳しいリファレンスも用意しています。

Adobe ColdFusion 10 マニュアルについて

ColdFusion のマニュアルは、あらゆるユーザーを総合的にサポートできるように作成されています。

マニュアルセット

ColdFusion のマニュアルセットには、次のマニュアルが含まれています。

マニュアル	説明
Adobe® ColdFusion® 10 インストール	Windows®、Macintosh®、Solaris™、Linux®、および AIX® 環境でのシステムインストールおよび基本設定について説明します。
Adobe® ColdFusion® 10 設定と管理	ColdFusion の管理作業 (サーバー設定の管理、データソースの設定、セキュリティの管理、ColdFusion アプリケーションのデプロイ、キャッシュ、CFX タグのセットアップ、ColdFusion サーバーモニタを使用したサーバーアクティビティの監視、Web サーバーの設定など) について説明します。
Adobe® ColdFusion® 10 アプリケーションの開発	ダイナミック Web アプリケーションの開発方法について説明します。 このガイドには、CFML プログラミング言語と ColdFusion の機能 (ColdFusion Web サービス、ColdFusion ポートレット、ColdFusion ORM、AJAX サポート、Flex および AIR 統合など) の使用方法と、他社の製品およびテクノロジー (Microsoft Office、OpenOffice、SharePoint など) との統合に関する詳細情報が含まれています。
Adobe® ColdFusion® 10 CFML リファレンス	すべての ColdFusion タグ、関数、変数に関する説明、シンタックス、使用方法、コード例が含まれています。

オンラインマニュアルの参照

ColdFusion のオンラインマニュアルはすべて HTML および Adobe PDF ファイルで提供されます。オンラインマニュアルを表示するには、ColdFusion ヘルプ＆サポートページ（www.adobe.com/go/learn_cfu_support_jp）にアクセスしてください。これらのオンラインマニュアルでは、新しいコメントを追加したり、既存のコメントを参照することもできます。

第 2 章：予約語と変数

Adobe ColdFusion 言語には、予約語とスコープ変数があります。

予約語

ColdFusion の変数、ユーザー定義関数名、およびカスタムタグ名に使用できない単語を次に示します。これらの語句には、特定の状況では使用できるものもありますが、エラーを回避するためにすべて使用しないことをお勧めします。

- cf で始まる任意の名前。ただし、CFML カスタムタグを直接呼び出す場合は、カスタムタグページ名の前に cf_ を付けます。
- Now、Hash などのビルトイン関数名
- Form、Session などのスコープ名
- NE、IS などの演算子
- Error、File などのビルトインデータ構造体の名前
- RecordCount、CGI 変数名などのビルトイン変数の名前
- 次の CFScript 言語要素名
  - for
  - default
  - continue
  - import
  - finally
  - local (関数宣言の内部)
  - interface
  - pageencoding

  ColdFusion では大文字と小文字が区別されません。たとえば、IS、Is、iS、および is はすべて予約語です。

  注意：新しく追加されたステートメントのキーワード（abort、rethrow、param など）は予約されていません。

フォームの予約語

次の語句のいずれかで終わるフォームフィールド名も作成しないでください。ただし、hidden フォームフィールド名を使用してフォームフィールドの検証ルールを指定する場合は例外です。

- _integer
- _float
- _range
- _date
- _time
- _eurodate

クエリーの予約語

次の表は、ColdFusion のクエリーオブクエリーで予約されている SQL キーワードの一覧です。この一覧には、SQL 標準の予約語がすべて含まれています。クエリー内の変数では、これらの予約語を使用しないでください。どのクエリー内でも、これらのキーワードを変数名には使用しないでください。

注意：多くのデータベース管理システムは、そのデータベースに対するクエリー内で変数名として使用できない予約語を別に定めています。具体的な一覧については、DBMS のドキュメントを参照してください。

ABSOLUTE	ACTION	ADD	ALL	ALLOCATE
ALTER	AND	ANY	ARE	AS
ASC	ASSERTION	AT	AUTHORIZATION	AVG
BEGIN	BETWEEN	BIT	BIT_LENGTH	BOTH
BY	CASCADE	CASCADED	CASE	CAST
CATALOG	CHAR	CHARACTER	CHARACTER_LENGTH	CHAR_LENGTH
CHECK	CLOSE	COALESCE	COLLATE	COLLATION
COLUMN	COMMIT	CONNECT	CONNECTION	CONSTRAINT
CONSTRAINTS	CONTINUE	CONVERT	CORRESPONDING	COUNT
CREATE	CROSS	CURRENT	CURRENT_DATE	CURRENT_TIME
CURRENT_TIMESTAMP	CURRENT_USER	CURSOR	DATE	DAY
DEALLOCATE	DEC	DECIMAL	DECLARE	DEFAULT
DEFERRABLE	DEFERRED	DELETE	DESC	DESCRIBE
DESCRIPTOR	DIAGNOSTICS	DISCONNECT	DISTINCT	DOMAIN
DOUBLE	DROP	ELSE	END	END-EXEC
ESCAPE	EXCEPT	EXCEPTION	EXEC	EXECUTE
EXISTS	EXTERNAL	EXTRACT	FALSE	FETCH
FIRST	FLOAT	FOR	FOREIGN	FOUND
FROM	FULL	GET	GLOBAL	GO
GOTO	GRANT	GROUP	HAVING	HOUR
IDENTITY	IMMEDIATE	IN	INDICATOR	INITIALLY
INNER	INPUT	INSENSITIVE	INSERT	INT
INTEGER	INTERSECT	INTERVAL	INTO	IS
ISOLATION	JOIN	KEY	LANGUAGE	LAST
LEADING	LEFT	LEVEL	LIKE	LOCAL
LOWER	MATCH	MAX	MIN	MINUTE
MODULE	MONTH	NAMES	NATIONAL	NATURAL
NCHAR	NEXT	NO	NOT	NULL
NULLIF	NUMERIC	OCTET_LENGTH	OF	ON

ONLY	OPEN	OPTION	OR	ORDER
OUTER	OUTPUT	OVERLAPS	PAD	PARTIAL
POSITION	PRECISION	PREPARE	PRESERVE	PRIMARY
PRIOR	PRIVILEGES	PROCEDURE	PUBLIC	READ
REAL	REFERENCES	RELATIVE	RESTRICT	REVOKE
RIGHT	ROLLBACK	ROWS	SCHEMA	SCROLL
SECOND	SECTION	SELECT	SESSION	SESSION_USER
SET	SIZE	SMALLINT	SOME	SPACE
SQL	SQLCODE	SQLERROR	SQLSTATE	SUBSTRING
SUM	SYSTEM_USER	TABLE	TEMPORARY	THEN
TIME	TIMESTAMP	TIMEZONE_HOUR	TIMEZONE_MINUTE	TO
TRAILING	TRANSACTION	TRANSLATE	TRANSLATION	TRIM
TRUE	UNION	UNIQUE	UNKNOWN	UPDATE
UPPER	USAGE	USER	USING	VALUE
VALUES	VARCHAR	VARYING	VIEW	WHEN
WHENEVER	WHERE	WITH	WORK	WRITE
YEAR	ZONE			

スコープ固有のビルトイン変数

ColdFusion では、cfdirectory や cfftp などのオペレーションにおいて、さまざまな変数が返されます。通常、変数は、そのタイプに応じてスコープを設定することで参照できます。つまり、変数が使用されるコードコンテキストに従って、Session.varname や Application.varname のように名前が付けられます。ColdFusion スコープの詳細については、『ColdFusion アプリケーションの開発』の Using ColdFusion Variables を参照してください。

cflock タグを使用すると、共有のデータ構造体、ファイル、および CFX を修正するための CFML 構文のスコープを制限することにより、修正を順番に行うことができます。詳細については、406 ページの「cflock」、および『ColdFusion アプリケーションの開発』の Using Persistent Data and Locking を参照してください。

変数スコープ

ColdFusion では、Variables スコープがサポートされます。cfset タグを使用して作成された変数でスコープが設定されていないものには、デフォルトで Variables スコープが設定されます。たとえば、ステートメント <CFSET linguist = Chomsky> によって作成された変数は、#Variables.linguist# として参照されます。

Caller スコープ

履歴

ColdFusion MX: Caller スコープは、構造体としてアクセス可能です。従来の ColdFusion リリースでは、このようなアクセスは行えませんでした。

CGI 変数

11 ページの「CGI 環境 (CGI スコープ) 変数」を参照。

クライアント変数

次のクライアント変数が予約されています。

```
Client.CFID
Client.CFToken
Client.HitCount
Client.LastVisit
Client.TimeCreated
Client.URLToken
```

サーバー変数

サーバー変数を参照するには、次のように Server 接頭辞を使用します。

```
Server.ColdFusion.ProductName
Server.ColdFusion.ProductVersion
Server.ColdFusion.ProductLevel
Server.ColdFusion.SerialNumber
Server.ColdFusion.SupportedLocales
Server.ColdFusion.AppServer
Server.ColdFusion.Expiration
Server.ColdFusion.RootDir
Server.OS.Name
Server.OS.AdditionalInformation
Server.OS.Version
Server.OS.BuildNumber
```

アプリケーション変数とセッション変数

アプリケーション変数とセッション変数を有効にするには、cfapplication タグまたは Application.cfc を使用します。次のように参照します。

```
Application.myvariable
Session.myvariable
```

共有データの修正が意図した順序で正しく行われるようにするには、cflock タグを使用します。詳細については、406 ページの「cflock」を参照してください。

ColdFusion には、あらかじめ定義されている次のアプリケーション変数およびセッション変数があります。

```
Application.ApplicationName
Session.CFID
Session.CFToken
Session.URLToken
```

カスタムタグ変数

ColdFusion カスタムタグは次の変数を返します。

```
ThisTag.ExecutionMode
ThisTag.HasEndTag
ThisTag.GeneratedContent
ThisTag.AssocAttribs[index]
```

カスタムタグに Caller 変数を設定し、呼び出し側に情報を提供することができます。Caller 変数は次のように設定します。

```
<cfset Caller.variable_name = "value">
```

呼び出し側のページは、cfoutput タグを次のように使用して、この変数にアクセスできます。

```
<cfoutput>#variable_name#</cfoutput>
```

リクエスト変数

リクエスト変数には、1 つのページのリクエスト処理に関係するデータが保管されます。リクエスト変数を使うと、カスタムタグなどのネストされたタグや処理済みタグに渡される構造体にデータを保管することができます。

ネストされたタグに情報を指定するには、リクエスト変数を次のように設定します。

```
<CFSET Request.field_name1 = "value">
<CFSET Request.field_name2 = "value">
<CFSET Request.field_name3 = "value">
...
```

ネストされたタグは、cfoutput タグを次のように使用することでリクエスト変数にアクセスできます。

```
<CFOUTPUT>#Request.field_name1#</CFOUTPUT>
```

フォーム変数

ColdFusion では、フォーム変数 FieldNames がサポートされます。FieldNames はフォームのフィールドの名前を返す変数です。この変数は、フォームに割り当てられたアクションページで次のように使用します。

```
Form.FieldNames
```

ColdFusion タグ固有の変数

ColdFusion タグには、データを変数として返すものがあります。たとえば、cffile タグではファイルサイズ情報が FileSize 変数で返され、CFFILE.FileSize として参照されます。

次のタグで返されるデータは、変数として参照できます。

```
cfcatch
cfdirectory
cferror
cffile
cfftp
cfhttp
cfindex
cfldap
cfpop
cfquery
cfregistry
cfsearch
cfstoredproc
```

ColdFusion クエリー変数

クエリーオブジェクトを返す ColdFusion タグでは、次の変数がサポートされます。ここで、**queryname** は name 属性の値です。

```
queryname.CurrentRow
queryname.RecordCount
queryname.ColumnList
```

CFCATCH 変数

cfcatch ブロック内では、アクティブな例外のプロパティに次の変数でアクセスできます。

```
CFCATCH.Type
CFCATCH.Message
CFCATCH.Detail
CFCATCH.ErrNumber
CFCATCH.NativeErrorCode
CFCATCH.SQLState
CFCATCH.LockName
CFCATCH.LockOperation
CFCATCH.MissingFileName
CFCATCH.TagContext
CFCATCH.ErrorCode
CFCATCH.ExtendedInfo
```

cfcatch ブロック内では、データベースの例外のプロパティに次の変数でアクセスできます。

```
CFCATCH.QueryError
CFCATCH.SQL
CFCATCH.Where
CFCATCH.Datasource
```

cfcatch ブロック内では、未定義の変数による例外のプロパティに次の変数でアクセスできます。

```
CFCATCH.Name
```

cfcatch ブロック内では、シンタックスと解析の例外のプロパティに次の変数でアクセスできます。

```
CFCATCH.TokenText
CFCATCH.Snippet
CFCATCH.Column
CFCATCH.KnownColumn
CFCATCH.Line
CFCATCH.KnownLine
```

CFDIRECTORY 変数

cfdirectory タグを、action=list とともに使用すると、次のようなクエリーオブジェクトが返されます。ここで、**queryname** は name 属性の値です。

```
queryname.Name
queryname.Size
queryname.Type
queryname.DateLastModified
queryname.Attributes
queryname.Mode
```

CFERROR 変数

cferror によるエラーページの生成時、type="request" または "exception" が指定されている場合は、次のエラー変数を使用できます。

```
Error.Diagnostics
Error.MailTo
Error.DateTime
Error.Browser
Error.GeneratedContent
Error.RemoteAddress
Error.HTTPReferer
Error.Template
Error.QueryString
```

type="validation" の場合は、次のエラー変数を使用できます。

```
Error.ValidationHeader
Error.InvalidFields
Error.ValidationFooter
```

例外タイプに当てはまる cfcatch 変数には、Error スコープ内で次のようにアクセスできます。

```
Error.Type
Error.Message
Error.Detail
Error.ErrNumber
Error.NativeErrorCode
Error.SQLState
Error.LockName
Error.LockOperation
Error.MissingFileName
Error.TagContext
Error.ErrorCode
Error.ExtendedInfo
```

注意：type = "Exception" の場合は、CFERROR.Diagnostics、CFERROR.Mailto、CFERROR.DateTime のように、Error の代わりに CFERROR を接頭辞として使用することができます。

CFFILE ACTION=Upload 変数

ファイル変数は読み取り専用です。ファイル変数を参照するには、CFFILE.ClientDirectory などのように CFFILE 接頭辞を使用します。CFFILE 接頭辞を優先するために、File 接頭辞は非推奨になりました。

```
CFFILE.AttemptedServerFile
CFFILE.ClientDirectory
CFFILE.ClientFile
CFFILE.ClientFileExt
CFFILE.ClientFileName
CFFILE.ContentSubType
CFFILE.ContentType
CFFILE.DateLastAccessed
CFFILE.FileExisted
CFFILE.FileSize
CFFILE.FileWasAppended
CFFILE.FileWasOverwritten
CFFILE.FileWasRenamed
CFFILE.FileWasSaved
CFFILE.OldFileSize
CFFILE.ServerDirectory
CFFILE.ServerFile
CFFILE.ServerFileExt
CFFILE.ServerFileName
CFFILE.TimeCreated
CFFILE.TimeLastModified
```

CFFTP エラー変数

cfftp stoponerror 属性を使用すると、次の変数に値が挿入されます。

```
CFFTP.Succeeded
CFFTP.ErrorCode
CFFTP.ErrorText
```

CFFTP.ReturnValue 変数

一部の cfftp ファイルおよびディレクトリオペレーションでは、変数 CFFTP.ReturnValue に戻り値が返される場合があります。この値は、action 属性の結果によって決まります。次のいずれかのアクションを指定した場合は、cfftp により値が返されます。

```
GetCurrentDir
GetCurrentURL
ExistsDir
ExistsFile
Exists
```

CFFTP クエリーオブジェクト列

cfftp タグを listdir アクションとともに使用すると、cfftp によりクエリーオブジェクトが返されます。ここで、**queryname** は name 属性の値で、***row*** は各ファイルまたはディレクトリエントリの行番号です。

```
queryname.Name[row]
queryname.Path[row]
queryname.URL[row]
queryname.Length[row]
queryname.LastModified[row]
queryname.Attributes
queryname.IsDirectory
queryname.Mode
```

CFHTTP 変数

cfhttp get オペレーションでは、テキストおよびバイナリファイルが返されます。ファイルがダウンロードされると、MIME タイプに応じて、内容が変数またはファイル内に次のように保管されます。

```
CFHTTP.FileContent
CFHTTP.MimeType
CFHTTP.Header
CFHTTP.ResponseHeader[http_hd_key]
CFHTTP.StatusCode
```

CFLDAP 変数

cfldapaction=query タグでは、LDAP クエリーについての情報が次のように返されます。

```
queryname.CurrentRow
queryname.RecordCount
queryname.ColumnList
```

CFPOP 変数

cfpop タグでは action 属性の値や、attachmentPath などの属性の使用に応じて、次の結果列が返されます。ここで、**queryname** は name 属性の値です。

```
queryname.Date
queryname.From
queryname.Body
queryname.Header
queryname.MessageNumber
queryname.ReplyTo
queryname.Subject
queryname.CC
queryname.To
queryname.CurrentRow
queryname.RecordCount
queryname.ColumnList
queryname.Attachments
queryname.AttachmentFiles
```

CFQUERY 変数と CFSTOREDPROC 変数

cfquery タグでは、クエリーについての情報が次の変数内で返されます。

```
CFQUERY.ExecutionTime
```

cfquery タグではクエリー名を使用して、クエリーについての次のデータのスコープを設定します。

```
queryname.CurrentRow
queryname.RecordCount
queryname.ColumnList
```

cfstoredproc タグでは、次の変数が返されます。

```
CFSTOREDPROC.ExecutionTime
CFSTOREDPROC.StatusCode
```

CFREGISTRY 変数

cfregistry タグでは、クエリーのレコードセットが次のように返され、GetAll アクションの実行後に参照できます。ここで、**queryname** は name 属性の値です。

```
queryname.Entry
queryname.Type
queryname.Value
```

CFSEARCH 変数

cfsearch オペレーションでは次の変数が返されます。ここで、**searchname** は name 属性の値です。

```
searchname.URL
searchname.Key
searchname.Title
searchname.Score
searchname.Custom1 and Custom2
searchname.Summary
searchname.RecordCount
searchname.CurrentRow
searchname.RecordsSearched
searchname.ColumnList
```

CGI 環境 (CGI スコープ) 変数

ブラウザからサーバーへのリクエスト送信時、Web サーバーおよびブラウザによって環境変数が作成されます。ColdFusion では、これらの変数は **CGI** 環境変数と呼ばれます。CGI 環境変数には、IP アドレス、ブラウザタイプ、認証済みのユーザー名など、ブラウザとサーバーの間のトランザクションについてのデータが含まれます。使用可能な CGI 環境変数は、ブラウザおよびサーバーソフトウェアによって異なります。

CGI 変数は、CGI スコープ内の ColdFusion ページに使用できます。これらの変数には、サーバーが API または CGI のいずれを使用して ColdFusion サーバーと通信するかにかかわらず、CGI 接頭辞が使用されます。CGI 環境変数は、特定のページリクエストに対して、ページ内の任意の場所で参照できます。CGI 変数は読み取り専用です。

デフォルトでは、cfdump タグを使用して CGI スコープを表示するか、CGI スコープのデバッグ出力をリクエストすると、CGI 環境変数の標準の固定リストが表示されます。使用可能な変数は、サーバー、ブラウザ、および双方の対話のタイプに依存するので、一部の変数は正常に使用できない場合があります。そのような変数は、デバッグ出力で空の文字列として表現されます。ダンプまたデバッグ出力で表示される変数リストに存在しない変数も含めて、アプリケーションコードの任意の CGI 変数をリクエストできます。

ColdFusion では、cfdump タグおよびデバッグ出力について、次の変数をチェックします。

```
AUTH_PASSWORD
AUTH_TYPE
AUTH_USER
CERT_COOKIE
CERT_FLAGS
CERT_ISSUER
CERT_KEYSIZE
CERT_SECRETKEYSIZE
CERT_SERIALNUMBER
CERT_SERVER_ISSUER
CERT_SERVER_SUBJECT
CERT_SUBJECT
CF_TEMPLATE_PATH
CONTENT_LENGTH
CONTENT_TYPE
CONTEXT_PATH
GATEWAY_INTERFACE
HTTPS
HTTPS_KEYSIZE
HTTPS_SECRETKEYSIZE
HTTPS_SERVER_ISSUER
HTTPS_SERVER_SUBJECT
```

```
HTTP_ACCEPT
HTTP_ACCEPT_ENCODING
HTTP_ACCEPT_LANGUAGE
HTTP_CONNECTION
HTTP_COOKIE
HTTP_HOST
HTTP_REFERER
HTTP_USER_AGENT
QUERY_STRING
REMOTE_ADDR
REMOTE_HOST
REMOTE_USER
REQUEST_METHOD
SCRIPT_NAME
SERVER_NAME
SERVER_PORT
SERVER_PORT_SECURE
SERVER_PROTOCOL
SERVER_SOFTWARE
WEB_SERVER_API (This value is always blank; retained for compatibility.)
```

次のセクションでは、CGI 環境変数のテスト方法について説明し、一般的に使用されるいくつかの CGI 環境変数に関する情報を提供します。

CGI 変数のテスト

ブラウザによってはいくつかの CGI 変数をサポートしないものもあるため、ColdFusion では CGI 変数が存在するかどうかをテストすると、ブラウザがサポートしているかいないかにかかわらず、常に true が返されます。CGI 変数が使用可能かどうかを調べるには、次の例のように、空の文字列についてテストします。

```
<cfif CGI.varname IS NOT "">
    CGI variable exists
<cfelse>
    CGI variable does not exist
</cfif>
```

CGI サーバー変数

次の表に、サーバーにより作成される一般的な CGI 環境変数を示します。これらの変数の中には、サーバーによっては利用できないものもあります。

CGI サーバー変数	説明
SERVER_SOFTWARE	リクエストに応答している (また、ゲートウェイを実行している) 情報サーバーソフトウェアの名前およびバージョンです。形式は name/version です。
SERVER_NAME	サーバーのホスト名、DNS エイリアス、または自己参照 URL で示される IP アドレスです。
GATEWAY_INTERFACE	そのサーバーが対応する CGI 仕様のリビジョンです。形式は CGI/revision です。
SERVER_PROTOCOL	そのリクエストが送信された情報プロトコルの名前とリビジョンです。形式は protocol/revision です。
SERVER_PORT	リクエストの送信先のポート番号です。
REQUEST_METHOD	リクエストの生成に使用されたメソッドです。HTTP の場合は、Get、Head、Post などです。
PATH_INFO	クライアントにより指定された追加パス情報です。スクリプトへは、仮想パス名の後に追加情報を加えることによってアクセスできます。追加情報は、PATH_INFO として送信されます。
PATH_TRANSLATED	仮想パスから物理パスへマッピングされた、PATH_INFO の変換後のバージョンです。

CGI サーバー変数	説明
SCRIPT_NAME	実行中のスクリプトへの仮想パスです。自己参照 URL で使用されます。
QUERY_STRING	そのスクリプトを参照した URL 内で、文字「?」の後に続くクエリー情報です。
REMOTE_HOST	リクエストを送信しているホスト名です。サーバーにこの情報がない場合は、REMOTE_ADDR が設定され、REMOTE_HOST は設定されません。
REMOTE_ADDR	リクエストを送信しているリモートホストの IP アドレスです。
AUTH_TYPE	サーバーがユーザー認証をサポートし、スクリプトが保護されている場合には、プロトコル固有の認証メソッドを使用してユーザーの検証が行われます。
REMOTE_USER AUTH_USER	サーバーがユーザー認証をサポートし、スクリプトが保護されている場合には、ユーザーが認証されたときのユーザー名が使用されます (AUTH_USER としても使用可能)。
REMOTE_IDENT	HTTP サーバーが RFC 931 による識別をサポートしている場合、この変数は、サーバーから取得したリモートユーザー名に設定されます。この変数は、ロギングにのみ使用します。
CONTENT_TYPE	HTTP POST や PUT などの情報が添付されたクエリーの場合は、これがデータのコンテンツタイプになります。
CONTENT_LENGTH	クライアントから指定されたコンテンツの長さです。

CGI クライアント変数

次の表では、ブラウザによって作成され、リクエストヘッダーに渡される一般的な CGI 環境変数について説明します。

CGI クライアント変数	説明
HTTP_REFERER	リンクあるいは、送信されたフォームデータの参照元ドキュメントです。
HTTP_USER_AGENT	クライアントがリクエストを送信するために現在使用しているブラウザです。形式は software/version library/version です。
HTTP_IF_MODIFIED_SINCE	ページが最後に修正された日時です。この変数を設定するかどうかはブラウザにより決定されます。これは通常、LAST_MODIFIED HTTP ヘッダを送信したサーバーへのレスポンスとして行われます。この変数は、ブラウザ側のキャッシュ機能を利用するときに使用できます。

CGI クライアント証明書変数

ColdFusion では、次のクライアント証明書データが使用できます。これらの変数は、SSL 環境で Microsoft IIS 4.0 または Netscape Enterprise を実行するときに、ご使用の Web サーバーがクライアント証明書を受け入れるように設定されている場合に使用できます。

CGI クライアント証明書変数	説明
CERT_SUBJECT	Web サーバーにより提供されるクライアント固有の情報です。このデータには通常、クライアント名、電子メールアドレスなどが含まれます。次に例を示します。 O = "VeriSign, Inc.", OU = VeriSign Trust Network, OU = "www.verisign.com/repository/RPA Incorp. by Ref.,LIAB.LTD(c)98", OU = Persona Not Validated, OU = Digital ID Class 1 - Microsoft, CN = Matthew Lund, E = mlund@.com
CERT_ISSUER	クライアント証明書を提供した証明機関についての情報です。次に例を示します。 O = "VeriSign, Inc.", OU = VeriSign Trust Network, OU = "www.verisign.com/repository/RPA Incorp. By Ref.,LIAB.LTD(c)98", CN = VeriSign Class 1 CA Individual Subscriber-Persona Not Validated

第 3 章：ColdFusion のタグ

CFML (ColdFusion Markup Language) には、ColdFusion のページ内でデータソースとの対話、データの操作、および出力の表示を行うために使用する一連のタグが用意されています。CFML タグのシンタックスは、HTML 要素のシンタックスと類似しています。

ここでは、CFML タグをカテゴリ別のリストとアルファベット順のリストで示します。その後で、個々のタグについて詳しく説明します。

ColdFusion 10 の新規のタグ

次の表では、ColdFusion 10 で追加された CFML タグについて簡単に説明します。

CFML タグ	カテゴリ	説明
cfexchangeconversation	通信タグ	Microsoft Exchange アカウントからの会話の整理および管理を支援します。
cfexchangefolder	通信タグ	メールフォルダーに対して様々なアクションを実行できます（フォルダー情報の取得、フォルダーの検索、フォルダーの作成、コピー、変更、移動、削除、フォルダーの中身のクリアなど）。
cfwebsocket	Web Socket タグ	CFM テンプレートで WebSocket オブジェクトを作成できます。このタグは、WebSocket の JavaScript オブジェクトへの参照をクライアントサイドに作成します。

タグの一覧

次の表で、CFML タグの簡単な説明を示します。

CFML タグ	カテゴリ	説明
cfabort	フロー制御タグ	このタグの位置で ColdFusion ページの処理を停止します。
cfajaximport	インターネットプロトコルタグ	ColdFusion AJAX ベースの機能で使用する JavaScript ファイルのインポートを制御します。
cfajaxproxy	インターネットプロトコルタグ	ColdFusion コンポーネントのクライアントページに AJAX プロキシクラスを生成します。
cfapplet	フォームタグ	cfform タグ内に Java アプレットを埋め込みます。
cfapplication	アプリケーションフレームワークタグ	アプリケーション名の定義、クライアント変数のアクティブ化、クライアント変数のストアメカニズムの指定を行います。
cfargument	拡張タグ	コンポーネント定義内にパラメータ定義を作成します。関数の引数を定義します。
cfassociate	アプリケーションフレームワークタグ	サブタグデータをベースタグとともに保存できるようにします。
cfbreak	フロー制御タグ	CFML のループ処理を中断します。
cfcache	ページ処理タグ	ColdFusion ページをキャッシュします。
cfcalendar	フォームタグ	日付を選択する対象のカレンダーを提供します。

CFML タグ	カテゴリ	説明
cfcase	フロー制御タグ	cfswitch タグおよび cfdefaultcase タグとともに使用します。
cfcatch	例外処理タグ、 フロー制御タグ	ColdFusion ページ内で例外を検出します。
cfchart	データ出力タグ	チャートの生成と表示を行います。
cfchartdata	データ出力タグ	チャートのデータポイントを定義します。
cfchartseries	データ出力タグ	チャートデータの表示スタイルを定義します。
cfcol	データ出力タグ	テーブルの列ヘッダ、プロパティを定義します。
cfcollection	拡張タグ	Solr コレクションを管理します。
cfcomponent	拡張タグ	コンポーネントオブジェクトの作成と定義を行います。
cfcontent	データ出力タグ、 ページ処理タグ	現在のページでダウンロードするファイルのコンテンツタイプとファイル名を定義します。
cfcontinue	フロー制御タグ	ループの上に処理を返します。cfloop タグ内で使用します。
cfcookie	変数操作タグ	有効期限、セキュリティオプションなどの Cookie 変数を定義し、設定します。
cfdbinfo	データベース操作タグ	データソースに関する情報を取得します。
cfdefaultcase	フロー制御タグ	一致する cfcase タグ値がない場合に、制御権を受け取ります。
cfdirectory	ファイル管理タグ	ColdFusion アプリケーション内から一般的なディレクトリ処理タスクを行います。
cfdiv	表示管理タグ	バインド式によってデータが挿入される HTML タグを作成します。
cfdocument	データ出力タグ	CFML および HTML を含むテキストブロックから PDF または Adobe® FlashPaper® 形式の出力を作成します。
cfdocumentitem	データ出力タグ	PDF または FlashPaper 形式のドキュメントについて、ヘッダ、フッタ、ページ区切りなどのアクション項目を指定します。
cfdocumentsection	データ出力タグ	PDF または FlashPaper 形式のドキュメントをセクションに分割します。
cfdump	デバッグタグ、 変数操作タグ	デバッグ目的で変数を出力します。
cfelse	フロー制御タグ	IF-THEN-ELSE 構文を作成します。
cfelseif	フロー制御タグ	IF-THEN-ELSE 構文を作成します。
cferror	例外処理タグ、 アプリケーションフレームワークタグ	エラーの発生時に、カスタムの HTML エラーページを表示します。
cfexchangecalendar	通信タグ	Microsoft Exchange のカレンダーイベントを取得、作成、削除、修正します (またはイベントに応答します)。
cfexchangeconnection	通信タグ	Exchange サーバーとの継続的接続を開きます (または閉じます)。
cfexchangecontact	通信タグ	Exchange の連絡先を取得、作成、削除、または修正します。
cfexchangeconversation	通信タグ	Microsoft Exchange アカウントからの会話の整理および管理を支援します。
cfexchangefolder	通信タグ	メールフォルダーに対して様々なアクションを実行できます（フォルダー情報の取得、フォルダーの検索、フォルダーの作成、コピー、変更、移動、削除、フォルダーの中身のクリアなど）。

CFML タグ	カテゴリ	説明
cfexchangefilter	通信タグ	Exchange タグの get オペレーションで使用するフィルタ条件を設定します。
cfexchangemail	通信タグ	Exchange メールメッセージの取得と削除を行い、メッセージプロパティを設定します。
cfexchangetask	通信タグ	Exchange のユーザータスクを取得、作成、削除、または修正します。
cfexecute	フロー制御タグ、拡張タグ	サーバーコンピュータ上で開発者が指定した処理を実行します。
cfexit	フロー制御タグ	実行中の CFML タグの処理を中断します。
cffeed	通信タグ、インターネットプロトコルタグ	Atom および RSS フィードの読み込み、作成、変換を行います。
cffile	ファイル管理タグ	ColdFusion アプリケーション内から一般的なファイル処理タスクを実行します。
cffileupload	ファイル管理タグ、フォームタグ	ユーザーのシステムから複数のファイルをアップロードするためのダイアログボックスを表示します。
cffinally	例外処理タグ	cftry タグの内部で使用します。
cfflush	データ出力タグ、ページ処理タグ	現在使用可能なデータをクライアントにフラッシュします。
cfform	フォームタグ	入力フォームの構築、クライアントサイドの入力検証を行います。
cfformgroup	フォームタグ	フォームコントロールを、コントロールを含むオブジェクトにグループ化します。
cfformitem	フォームタグ	テキストおよび分割ルールを Adobe® Flash® フォームに追加します。
cfftp	フォームタグ、拡張タグ、インターネットプロトコルタグ	FTP ファイルオペレーションを可能にします。
cffunction	拡張タグ	CFML で構築する関数を定義します。
cfgrid	フォームタグ	cfform タグ内で、テーブル形式のグリッドコントロールを表示します。
cfgridcolumn	フォームタグ	cfform 内で使用します。cfgrid 内の列を定義します。
cfgridrow	フォームタグ	グリッドの行を定義します。cfgrid と併用します。
cfgridupdate	フォームタグ	編集したグリッドデータから ODBC データソースを直接更新します。
cfheader	データ出力タグ、ページ処理タグ	HTTP ヘッダを生成します。
cfhtmlhead	ページ処理タグ	テキストや HTML をページの HEAD セクションに書き込みます。
cfhttp	インターネットプロトコルタグ	GET および POST を実行して、ファイルをアップロードしたり、フォーム、Cookie、クエリー、または CGI 変数をサーバーに直接アップロードしたりします。
cfhttpparam	インターネットプロトコルタグ	cfhttp の POST オペレーションに必要なパラメータを指定します。cfhttp とともに使用します。
cfif	フロー制御タグ	IF-THEN-ELSE 構文を作成します。
cfimage	その他のタグ	cfimage (image 関数で操作できる ColdFusion のデータ型) を作成します。
cfimap	通信タグ、インターネットプロトコルタグ	IMAP サーバーの電子メールおよびフォルダを取得し、管理します。

CFML タグ	カテゴリ	説明
cfimport	アプリケーションフレームワークタグ	CFML ページに JSP タグライブラリをインポートします。
cfinclude	フロー制御タグ	ColdFusion ページにリファレンスを埋め込みます。
cfindex	拡張タグ	Solr 検索インデックスを作成します。
cfinput	フォームタグ	ラジオボタン、チェックボックス、テキスト入力ボックスなどの入力要素を作成します。cfform 内で使用します。
cfinsert	データベース操作タグ	データソースにレコードを挿入します。
cfinterface	アプリケーションフレームワークタグ、拡張タグ	ColdFusion コンポーネントで実装できるインターフェイスを定義します。
cfinvoke	拡張タグ	コンポーネントメソッドを ColdFusion ページまたはコンポーネントから呼び出します。
cfinvokeargument	拡張タグ	パラメータをコンポーネントメソッドまたは Web サービスに渡します。
cflayout	表示管理タグ	特定のレイアウト動作を指定してコンテナの領域を作成します。
cflayoutarea	表示管理タグ	cflayout タグ本文内の表示領域を定義します。
cfldap	インターネットプロトコルタグ	LDAP ディレクトリサーバーにアクセスします。
cflocation	フロー制御タグ	ページの実行を制御します。
cflock	アプリケーションフレームワークタグ	データの整合性を維持しながら、CFML コードの実行を同期化します。
cflog	データ出力タグ、 その他のタグ	ログファイルにメッセージを書き込みます。
cflogin	セキュリティタグ	ユーザーログインおよび認証コード用のコンテナを定義します。
cfloginuser	セキュリティタグ	ColdFusion に対して認証済みのユーザーを識別します。
cflogout	セキュリティタグ	現在のユーザーをログアウトさせます。
cfloop	フロー制御タグ	条件に基づいて一連の命令を繰り返します。
cfmail	通信タグ、インターネットプロトコルタグ	電子メールメッセージを組み立てて送信します。
cfmailparam	通信タグ、インターネットプロトコルタグ	電子メールメッセージにファイルを添付したり、ヘッダを追加したりします。
cfmailpart	通信タグ、インターネットプロトコルタグ	マルチパートメールメッセージの一部を含めます。
cfmap	その他のタグ	ColdFusion Web ページ内に Google マップを埋め込みます。
cfmapitem	その他のタグ	マップ上にマーカーを作成します。cfmap タグの子タグです。
cfmediaplayer	その他のタグ	FLV ファイルを再生できる組み込みメディアプレーヤーを作成します。
cfmenu	表示管理タグ	トップレベルのメニューまたはツールバーを作成します。
cfmenuitem	表示管理タグ	サブメニューの頭になるアイテムを含め、メニュー内の個々のエントリを定義します。
cfmessagebox	アプリケーションフレームワークタグ	ポップアップメッセージを表示するコントロールを定義します。
cfNTauthenticate	セキュリティタグ	NT ドメインについてユーザー情報を認証します。

CFML タグ	カテゴリ	説明
cfobject	拡張タグ	COM オブジェクト、コンポーネントオブジェクト、CORBA オブジェクト、Java オブジェクト、および Web サービスオブジェクトを作成します。
cfobjectcache	データベース操作タグ	クエリーキャッシュをフラッシュします。
cfoutput	データ出力タグ	データベースクエリーや他のオペレーションの出力を表示します。
cfparam	変数操作タグ	パラメータとそのデフォルト値を定義します。
cfpdf	フォームタグ	既存の PDF ドキュメントを操作します。
cfpdfform	フォームタグ	PDF フォームの作成と操作を行います。
cfpdfformparam	フォームタグ	PDF フォーム上にインタラクティブフィールドを作成します。
cfpdfparam	フォームタグ	cfpdf タグの子タグです。複数のページまたは PDF ドキュメントを 1 つのファイルにマージするマージアクションのみで使用します。
cfpdfsubform	フォームタグ	PDF フォーム内にサブフォームを作成します。
cfpod	表示管理タグ	ブラウザの領域またはレイアウト領域を作成します。オプションでタイトルバーと本文も指定できます。
cfpop	通信タグ、インターネットプロトコルタグ	POP メールサーバーからメッセージの取得と削除を行います。
cfpresentation	データ出力タグ	HTML ページまたは SWF ファイルからダイナミックにプレゼンテーションを作成します。
cfpresentationslide	データ出力タグ	HTML ページまたは SWF ソースファイル (cfpresentation タグの子タグ) からダイナミックにスライドを作成します。
cfpresenter	データ出力タグ	スライドプレゼンテーション内でプレゼンタの紹介を行います。
cfprint	データ出力タグ	PDF ドキュメントを印刷します。印刷ジョブを自動化する場合に使用します。
cfprocessingdirective	データ出力タグ	空白などの出力を抑制します。
cfprocparam	データベース操作タグ	ストアドプロシージャのパラメータ情報を保持します。
cfprocresult	データベース操作タグ	ストアドプロシージャの結果セットにアクセスするために ColdFusion タグが使用する結果セット名です。
cfprogressbar	その他のタグ	アクティビティの進行状況を示すプログレスバーを定義します。
cfproperty	拡張タグ	コンポーネントを定義します。
cfquery	データベース操作タグ	SQL ステートメントをデータベースに渡します。
cfqueryparam	データベース操作タグ	クエリーパラメータのデータ型をチェックします。
cfregistry	その他のタグ、 変数操作タグ	Windows システムレジストリ内のキーと値の読み取り、書き込み、および削除を行います。
cfreport	例外処理タグ	ColdFusion Report Builder レポートまたは Crystal Reports レポートを埋め込みます。
cfreportparam	例外処理タグ	入力パラメータを ColdFusion Report Builder レポートに渡します。
cfrethrow	例外処理タグ	現在アクティブな例外を再び返します。
cfreturn	拡張タグ	コンポーネントメソッドから結果を返します。
cfsavecontent	変数操作タグ	タグ本文内部で生成された内容を変数に保存します。
cfschedule	変数操作タグ	ページの実行をスケジューリングし、オプションでスタティックページを作成します。

CFML タグ	カテゴリ	説明
cfscript	アプリケーションフレームワークタグ	一連の cfscript ステートメントを囲みます。
cfsearch	拡張タグ	cfindex を使用して、Solr コレクション内のインデックス付きデータに対して検索を実行します。
cfselect	フォームタグ	ドロップダウンリストボックスフォーム要素を作成します。cfform タグ内で使用します。
cfset	変数操作タグ	変数を定義します。
cfsetting	その他のタグ、 変数操作タグ	ColdFusion 設定を定義し、制御します。
cfsharepoint	拡張タグ	ColdFusion から SharePoint アクションを呼び出します。
cfsilent	データ出力タグ、 ページ処理タグ	タグのスコープ内での CFML 出力を抑制します。
cfslider	フォームタグ	スライダコントロールを作成します。cfform 内で使用します。
cfspreadsheet	拡張タグ	Excel スプレッドシートファイルを管理します。
cfsprydataset	インターネットプロトコルタグ	spry データセットを作成します。
cfstoredproc	データベース操作タグ	データベース接続情報を保持し、実行するストアドプロシージャを識別します。
cfswitch	フロー制御タグ	渡された式を評価し、マッチする cfcase タグに制御権を渡します。
cftable	データ出力タグ	ColdFusion ページ内にテーブルを構築します。
cftextarea	フォームタグ	複数行で構成されたテキストボックスをフォームに挿入します。
cfthread	アプリケーションフレームワークタグ	ColdFusion スレッド (独立した実行ストリーム) の作成と管理を行います。
cfthrow	例外処理タグ、 フロー制御タグ	開発者指定の例外を返します。
cftimer	デバッグタグ	コードのブロックの実行時間を表示します。
cftooltip	表示管理タグ	マウスポインタをタグの本文要素の上に置いたときに表示されるテキストを指定します。
cftrace	デバッグタグ	アプリケーションデバッグデータの表示およびロギングを行います。
cftransaction	データベース操作タグ	cfquery オペレーションを 1 つのトランザクションにグループ化し、ロールバック処理を行います。
cftree	フォームタグ	ツリーコントロール要素を作成します。cfform 内で使用します。
cftreeitem	フォームタグ	フォーム内にツリーコントロール要素を挿入します。cftree とともに使用します。
cftry	例外処理タグ、 フロー制御タグ	ColdFusion ページ内で例外を検出します。
cfupdate	データベース操作タグ	データベースのデータソース内の行を更新します。
cfwebsocket	Web Socket タグ	CFM テンプレートで WebSocket オブジェクトを作成できます。このタグは、WebSocket の JavaScript オブジェクトへの参照をクライアントサイドに作成します。
cfwddx	拡張タグ	CFML データ構造体の、XML ベースの WDDX 形式へのシリアル化およびシリアル化解除を行います。

CFML タグ	カテゴリ	説明
cfwindow	表示管理タグ	ブラウザ内にポップアップウィンドウを作成します。
cfxml	拡張タグ	XML ドキュメントオブジェクトを作成します。
cfzip	ファイル管理タグ	ZIP ファイルと JAR ファイルを操作します。
cfzipparam	ファイル管理タグ	ZIP ファイルと JAR ファイルを操作します。

機能別のタグ一覧

次に、機能または目的別のタグ一覧表を示します。

アプリケーションフレームワークタグ

cfapplication
cfassociate
cferror
cfimport
cfinterface
cflock
cfscript
cfthread

通信タグ

cfexchangecalendar
cfexchangeconnection
cfexchangecontact
cfexchangefilter
cfexchangemail
cfexchangetask
cffeed
cfmail
cfmailparam
cfmailpart
cfpop
cfimap

データベース操作タグ

cfdbinfo
cfinsert
cfobjectcache
cfprocparam
cfprocresult
cfquery
cfqueryparam
cfstoredproc
cftransaction
cfupdate

データ出力タグ

cfchart
cfchartdata
cfchartseries
cfcol
cfcontent
cfdocument
cfdocumentitem
cfdocumentsection
cfflush
cfheader
cflog
cfoutput
cfpresentation
cfpresentationslide
cfpresenter
cfprocessingdirective
cfprint
cfreport
cfreportparam
cfsilent
cftable

デバッグタグ

cfdump
cftimer
cftrace

表示管理タグ

cfdiv
cflayout
cflayoutarea
cfmap
cfmapitem
cfmediaplayer
cfmenu
cfmenuitem
cfmessagebox
cfpod
cfprogressbar
cftooltip
cfwindow

例外処理タグ

cfcatch
cferror
cffinally
cfrethrow
cfthrow
cftry

拡張タグ

cfchart
cfchartdata
cfchartseries
cfcollection
cfcomponent
cfexecute
cfftp
cffunction
cfindex
cfinterface
cfinvoke
cfinvokeargument
cfobject
cfproperty
cfreport
cfreportparam
cfreturn
cfsearch
cfsharepoint
cfspreadsheet
cfwddx
cfxml

ファイル管理タグ

cfdirectory
cffile
cffileupload
cfftp
cfzip
cfzipparam

フロー制御タグ

cfabort
cfbreak
cfcase
cfcontinue
cfdefaultcase
cfelse
cfelseif
cfexecute
cfexit
cfif
cfinclude
cflocation
cfloop
cfrethrow
cfswitch
cfthrow
cftry

フォームタグ

cfapplet
cfcalendar
cffileupload
cfform
cfformgroup
cfformitem
cfgrid
cfgridcolumn
cfgridrow
cfgridupdate
cfinput
cfpdf
cfpdfform
cfpdfformparam
cfpdfparam
cfpdfsubform
cfselect
cfslider
cftextarea
cftree
cftreeitem

インターネットプロトコルタグ

cfajaximport
cfajaxproxy
cfftp
cffeed
cfimap
cfhttp
cfhttpparam
cfldap
cfmail
cfmailparam
cfmailpart
cfpop
cfsprydataset

ページ処理タグ

cfcache
cfcontent
cfflush
cfheader
cfhtmlhead
cfinclude
cfprocessingdirective
cfsetting
cfsilent

セキュリティタグ

cflogin
cfloginuser
cflogout
cfNTauthenticate

変数操作タグ

cfcookie
cfdump
cfparam
cfregistry
cfsavecontent
cfschedule
cfset
cfsetting

Web Socket タグ

cfwebsocket

その他のタグ

cfimage
cflog
cfregistry

ColdFusion 5 以降に変更されたタグ

次の表では、ColdFusion 5 以降に変更されたタグ、属性、値と、それらがどのリリースで変更されたかを示します。

新規のタグ、属性、および値

次の表では、ColdFusion MX リリース以降に追加されたタグ、属性、属性値を示します。

タグ	属性または値	追加された ColdFusion リリース
複数のタグ	attributeCollection	ColdFusion 8
cfajaximport	すべて	ColdFusion 8
cfajaxproxy	すべて	ColdFusion 8

タグ	属性または値	追加された ColdFusion リリース
cfapplication	scriptProtect	ColdFusion MX 7
	loginStorage	ColdFusion MX 6.1
cfargument	type 属性の component 値	ColdFusion 8
	type 属性の xml 値	ColdFusion MX 7
	すべて	ColdFusion MX
cfcache	cachedirectory 属性、timespan 属性	ColdFusion MX
cfcalendar	onBlur 属性および onFocus 属性	ColdFusion MX 7.01
	すべて	ColdFusion MX 7
cfchart	style 属性、title 属性	ColdFusion MX 7
	xAxisType 属性、yAxisType 属性	ColdFusion MX 6.1
	すべて	ColdFusion MX
cfchartdata	すべて	ColdFusion MX
cfchartseries	datalabelstyle 属性	ColdFusion MX 7
	type 属性の horizontalbar 値	
	すべて	ColdFusion MX
cfcollection	categories 属性	ColdFusion MX 7
	language 属性の新しい値	
	action 属性の list 値および categoryList 値	
	name 属性	ColdFusion MX
cfcomponent	implements 属性、serviceaddress 属性	ColdFusion 8
	extends 属性の component 値	
	style、namespace、serviceportname、porttypename、wsdlfile、bindingname、および output の各属性	ColdFusion MX 7
	document-literal スタイル Web サービスの発行時における、hint 属性と displayname 属性の拡張機能	
	すべて	ColdFusion MX
cfcontent	variable 属性	ColdFusion MX 7
cfcontinue	すべて	ColdFusion 9
cfdbinfo	すべて	ColdFusion 8
cfdirectory	listinfo 属性および type 属性	ColdFusion 8
	list アクションおよび delete アクションの recurse 属性	ColdFusion MX 7
cfdiv	すべて	ColdFusion 8

タグ	属性または値	追加された ColdFusion リリース
cfdocument	bookmark、authPassword、authUser、localUrl、proxyHost、proxyPassword、proxyPort、proxyUser、saveAsName、および userAgent の各属性 totalsectionpagecount スコープ変数および currentsectionpagenumber スコープ変数	ColdFusion 8
	src、srcfile、および mimetype の各属性	ColdFusion MX 7.01
	すべて	ColdFusion MX 7
cfdocumentitem	すべて	ColdFusion MX 7
cfdocumentsection	name、authPassword、authUser、および userAgent の各属性	ColdFusion 8
	すべて	ColdFusion MX 7
cfdump	show、format、hide、keys、metainfo、output、および showUDFs の各属性	ColdFusion 8
cfexchangecalendar	すべて	ColdFusion 8
cfexchangeconnection	すべて	ColdFusion 8
cfexchangecontact	すべて	ColdFusion 8
cfexchangefilter	すべて	ColdFusion 8
cfexchangemail	すべて	ColdFusion 8
cfexchangetask	すべて	ColdFusion 8
cfexecute	variable 属性	ColdFusion MX 6.1
cffeed	すべて	ColdFusion 8
cffile	action="upload" アクションの result 属性	ColdFusion MX 7
	action="append" アクションおよび action="write" アクションの fixnewline 属性	
cffileupload	すべて	ColdFusion 9
cffinally	すべて	ColdFusion 9
cfform	onSuccess 属性の追加、AJAX コントロールでの onError 属性のサポート	ColdFusion 8
	name 属性および action 属性はオプション	ColdFusion MX 7
	accessible、format、height、width、method、onError、onReset、preloader、scriptsrc、skin、style、timeout、および wMode の各属性	
cfformgroup	すべて	ColdFusion MX 7
cfformitem	type 属性の script 値	ColdFusion MX 7.01
	すべて	ColdFusion MX 7
cfftp	fingerprint、key、paraphrase、および secure の各属性 action 属性の quote、site、allo、および acct の各値	ColdFusion 8
	result 属性	ColdFusion MX 7

タグ	属性または値	追加された ColdFusion リリース
cffunction	description 属性。returntype 属性の XML 値	ColdFusion MX 7
	すべて	ColdFusion MX
cfgrid	bind、bindOnLoad、pageSize、preservePageOnSort、stripeRows、stripeRowColor の各属性。format 属性の HTML 値。	ColdFusion 8
	onBlur 属性および onFocus 属性	ColdFusion MX 7.01
	format 属性、および Flash 形式と XML 形式の出力のサポート enabled、onChange、style、tooltip、および visible の各属性 (Flash 形式のみ)	ColdFusion MX 7
cfgridcolumn	mask 属性	ColdFusion MX 7
	type 属性の currency 値	ColdFusion MX 7
cfhttp	clientCert および clientCertPassword 属性	ColdFusion 8
	GetAsBinary 属性の never 値	ColdFusion MX 7.01
	result 属性	ColdFusion MX 7
	method 属性の HEAD 値、PUT 値、DELETE 値、OPTIONS 値、および TRACE 値	ColdFusion MX 6.1
	multipart、getasbinary、proxyUser、および proxyPassword の各属性	
	charset 属性、firstrowasheaders 属性	ColdFusion MX
cfhttpparam	type 属性の header および body 値	ColdFusion MX 6.1
	encoded 属性、mimeType 属性	
cfimage	すべて	ColdFusion 8
cfimport	すべて	ColdFusion MX
cfimap	すべて	ColdFusion 9
cfindex	prefix 属性	ColdFusion MX 7.01
	update アクションおよび refresh アクションの custom3 属性、custom4 属性、category 属性、および categorytree 属性	ColdFusion MX 7
	update、refresh、delete、および purge の各アクションの status 属性	
	language 属性の新しい値	

タグ	属性または値	追加された ColdFusion リリース
cfinput	autosuggest、autosuggestBindDelay、autosuggestMinLength、delimiter、maxResultsDisplayed、showAutosuggestLoadingIcon、sourceForTooltip、および typeahead の各属性。	ColdFusion 8
	HTML フォームでの bind 属性のサポート。bindAttribute、bindOnload、および onBindError の各属性。	
	HTML フォームの type 属性の datefield 値	
	height 属性および width 属性 (checkbox および radiobutton を除くすべて)	ColdFusion MX 7
	bind 属性 (text および password)	
	label 属性 (button、image、reset、および submit を除くすべて)	
	mask 属性 (text のみ)	
	validateAt 属性 (button、image、reset、および submit を除くすべて)	
	type 属性の datefield、button、file、hidden、image、reset、および submit の各値	
	daynames 属性および monthnames 属性 (type="datefield" のみ)	
	validate 属性の boolean、email、guid、maxlength、noblanks、range、submitOnce、URL、USdate、および uuid の各値	
	tooltip 属性、visible 属性、enabled 属性 (Flash 形式のみ)	
	src 属性 (image のみ)	
cfinterface	すべて	ColdFusion 8
cfinvoke	refreshWSDL 引数、wsdl2java 引数	ColdFusion 8
	Web サービスの servicePort 属性	ColdFusion MX 7
	すべて	ColdFusion MX
cfinvokeargument	omit 属性	ColdFusion MX 7
	すべて	ColdFusion MX
cflayout	すべて	ColdFusion 8
cflayoutarea	すべて	ColdFusion 8
cfldap	returnAsBinary 属性	ColdFusion MX 7
cflocation	statusCode 属性	ColdFusion 8
cflock	scope 属性の request 値	ColdFusion 8
cflogin	すべて	ColdFusion MX
cfloginuser	すべて	ColdFusion MX
cflogout	すべて	ColdFusion MX

タグ	属性または値	追加された ColdFusion リリース
cfloop	characters、file、および array の各属性	ColdFusion 8
cfmail	priority、useSSL、および useTLS	ColdFusion 8
	cfmail タグの本文に MIME エンコードされたメッセージ全体を埋め込んでマルチパートメールを送信することができなくなりました。代わりに cfmailpart タグを使用してください。	ColdFusion MX 7
	charset、failto、replyTo、userName、password、および wrapText の各属性	ColdFusion MX 6.1
	spoolEnable 属性	ColdFusion MX
cfmailparam	contentID 属性、disposition 属性	ColdFusion MX 7
	type 属性	ColdFusion MX 6.1
cfmailpart	すべて	ColdFusion MX 6.1
cfmap	すべて	ColdFusion 9
cfmapitem	すべて	ColdFusion 9
cfmediaplayer	すべて	ColdFusion 9
cfmenu	すべて	ColdFusion 8
cfmenuitem	すべて	ColdFusion 8
cfmessagebox	すべて	ColdFusion 9
cfNTauthenticate	すべて	ColdFusion MX 7
cfobject	type 属性の .net 値。関連する assembly、port、protocol、および secure の各属性	ColdFusion 8
	Web サービスの password、proxyPassword、proxyPort、proxyServer、proxyUser、refreshWSDL、userName、wsdl2JavaArgs、および wsportname の各属性	
	component および webservice 属性	ColdFusion MX
cfobjectcache	すべて	ColdFusion MX
cfparam	min 属性、max 属性、pattern 属性	ColdFusion MX 7
	type 属性の creditcard、email、eurodate、float、integer、range、regex、regular_expression、ssn、social_security_number、time、URL、USdate、XML、および zipcode の各値	

タグ	属性または値	追加された ColdFusion リリース

cfpdf	action = "thumbnail" には、次の新規属性があります。
	• hires
	• compresstiffs
	• maxlength
	• maxbreadth
	• maxscale
	action = "optimize" (新規アクションおよびすべて新しい属性)
	action = "addheader" (新規アクション)
	action = "addfooter" (新規アクション)
	action = "removeheaderfooter" (新規アクション)
	action = "extracttext"
	action = "extractimage"
	action = "write" package = "true"
	action = "merge" encodeall="true"
	action = "write" name= #PDF variable#
	action = "transform" (新規アクション)
		ColdFusion 9

cfpdfform	すべて	ColdFusion 8
cfpdfformparam	すべて	ColdFusion 8
cfpdfparam	すべて	ColdFusion 8
cfpdfsubform	すべて	ColdFusion 8
cfpod	すべて	ColdFusion 8
cfpop	cids クエリー変数	ColdFusion MX 7.01
cfpresentation	すべて	ColdFusion 8
cfpresentationslide	すべて	ColdFusion 8
cfpresenter	すべて	ColdFusion 8
cfprint	すべて	ColdFusion 8
cfprocessingdirective	pageEncoding 属性	ColdFusion MX
cfprocparam	すべて	ColdFusion 9
cfprogressbar	すべて	ColdFusion 9
cfproperty	すべて	ColdFusion MX
cfquery	result 属性	ColdFusion MX 7
cfreturn	すべて	ColdFusion MX

タグ	属性または値	追加された ColdFusion リリース
cfreport	format 属性の HTML 値および XML 値、resourceTimespan 属性、style 属性	ColdFusion 8
	format 属性の RTF 値	ColdFusion MX 7.01
	template、format、name、filename、query、および overwrite の各属性	ColdFusion MX 7
cfreportparam	chart、query、series、style、subreport の各属性	ColdFusion 8
	name 属性、value 属性	ColdFusion MX 7
cfsearch	category、categoryTree、status、suggestions、contextPassages、contextBytes、contextHighlightBegin、contextHighlightEnd、および previousCriteria の各属性	ColdFusion MX 7
	type 属性の natural、internet、および internet_basic の各値	
cfselect	HTML フォームでの bind 属性のサポート。bindAttribute、bindOnload、および onBindError の各属性。	ColdFusion 8
	HTML フォームでのツールヒントのサポート (sourceForTooltip 属性など)	
	selected 属性をリストで使用	ColdFusion MX 7
	enabled、group、height、label、onKeyUp、onKeyDown、onMouseUp、onMouseDown、onChange、onClick、queryPosition、tooltip、visible、および width の各属性	
cfsetting	requestTimeOut 属性	ColdFusion MX
cfsharepoint	すべて	ColdFusion 9
cfspreadsheet	すべて	ColdFusion 9
cfsprydataset	すべて	ColdFusion 8
cfstoredproc	result 属性	ColdFusion MX 7
cftextarea	リッチテキストエディタでの richtext、basepath、fontFormats、fontNames、fontSizes、skin、stylesXML、templatesXML、toolbar、toolbarOnFocus などの属性のサポート (HTML 形式のみ)。HTML 形式での height および width 属性のサポート。	ColdFusion 8
	bind 属性のサポート。HTML 形式での bindAttribute、bindOnLoad、および onBindError 属性のサポート。	
	HTML 形式でのツールヒントのサポート (tooltip および sourceForToolTip 属性など)	
	html 属性	ColdFusion MX 7.01
	すべて	ColdFusion MX 7
cfthread	すべて	ColdFusion 8
cfthrow	object 属性	ColdFusion MX
cftimer	すべて	ColdFusion MX 7

タグ	属性または値	追加された ColdFusion リリース
cftooltip	すべて	ColdFusion 8
cftree	onBlur 属性および onFocus 属性	ColdFusion MX 7.01
	format、onChange、style の各属性	ColdFusion MX 7
cftrace	すべて	ColdFusion MX
cfwindow	すべて	ColdFusion 8
cfxml	すべて	ColdFusion MX
cfzip	すべて	ColdFusion 8
cfzipparam	すべて	ColdFusion 8

非推奨のタグ、属性、および値

次のタグ、属性、および属性値は非推奨となっています。これらを ColdFusion のアプリケーションで使用しないでください。ColdFusion MX よりも新しいリリースでは、これらは動作せずエラーが発生する可能性があります。

タグ	属性または値	非推奨となった ColdFusion リリース
cfcache	cachedirectory 属性、timeout 属性	ColdFusion MX
cfcollection	action 属性の map オプションおよび repair オプション	ColdFusion MX 7
cferror	exception 属性の monitor オプション	ColdFusion MX
cffile	attributes 属性の system 値	ColdFusion MX
	attributes 属性の temporary 値	ColdFusion MX
cfform	passthrough 属性	ColdFusion MX 7
	enableCAB 属性	ColdFusion MX
cfftp	agentname 属性	ColdFusion MX
cfgraph	すべて	ColdFusion MX
cfgraphdata	すべて	ColdFusion MX
cfgridupdate	connectString、dbName、dbServer、dbType、provider、および providerDSN の各属性	ColdFusion MX
cfinput	passthrough 属性	ColdFusion MX 7
cfinsert	connectString、dbName、dbServer、dbType、provider、および providerDSN の各属性	ColdFusion MX
cfldap	filterFile 属性	ColdFusion MX
cflog	date、thread、および time の各属性	ColdFusion MX
cfquery	connectString、dbName、dbServer、provider、providerDSN、および sql の各属性	ColdFusion MX
	次の dbType 属性値 dynamic、ODBC、Oracle73、Oracle80、Sybase11、OLEDB、DB2	ColdFusion MX (値 query は有効)
cfregistry	すべて (UNIX 上でのみ)	ColdFusion MX

タグ	属性または値	非推奨となった ColdFusion リリース
cfsearch	external 属性、language 属性	ColdFusion MX
cfselect	passthrough 属性	ColdFusion MX 7
cfservlet	すべて	ColdFusion MX
cfservletparam	すべて	ColdFusion MX
cfslider	img、imgStyle、grooveColor、refreshLabel、tickmarkimages、tickmarklabels、tickmarkmajor、および tickmarkminor の各属性	ColdFusion MX
cfstoredproc	connectString、dbName、dbServer、dbType、provider、および providerDSN の各属性	ColdFusion MX
cftextinput	すべて	ColdFusion MX 7
cfupdate	connectString、dbName、dbServer、dbType、provider、および providerDSN の各属性	ColdFusion MX

廃止されたタグ、属性、および値

次のタグ、属性、および属性値は廃止されました。これらを ColdFusion のアプリケーションで使用しないでください。ColdFusion 5 以降のリリースでは、これらは機能せず、エラーを引き起こす可能性があります。

タグ	属性または値	これらが廃止となった ColdFusion リリース
cfauthenticate	すべて	ColdFusion MX
cfchart	rotated 属性	ColdFusion MX 7
cffile	attributes 属性の値 archive	ColdFusion MX
cfimpersonate	すべて	ColdFusion MX
cfindex	action 属性の値 optimize	ColdFusion MX
	external 属性	
cfinternaladminsecurity	すべて	ColdFusion MX このタグは『CFML リファレンス』にはありませんでした。
cfldap	filterConfig 属性および filterFile 属性	ColdFusion MX
cfnewinternaladminsecurity	すべて	ColdFusion MX このタグは『CFML リファレンス』にはありませんでした。
cfsetting	catchExceptionsByPattern 属性	ColdFusion MX

タグ a ～ b

cfabort

説明

このタグの位置で ColdFusion ページの処理を停止します。ColdFusion によって、このタグの前に処理されたすべての内容が返されます。このタグはしばしば条件論理式とともに使用され、ある条件が発生した場合にページの処理を停止します。

カテゴリ

フロー制御タグ

シンタックス

```
<cfabort
   showError = "error message">
```

注意：このタグの属性は attributeCollection で指定でき、その値は構造体になります。attributeCollection で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfbreak、cfexecute、cfexit、cfif、cflocation、cfloop、cfswitch、cfthrow、cftry、『ColdFusion アプリケーションの開発』の cfabort and cfexit

属性

属性	必須 / オプション	デフォルト	説明
showError	オプション		タグの実行時に、標準の ColdFusion エラーページに表示するエラーです。

使用方法

cfabort タグと cferror タグをともに使用すると、cfabort タグによりすぐに処理が停止され、cferror タグにより指定されたページに出力がリダイレクトされます。

このタグに showError 属性値が含まれない場合、タグに到達すると処理はすぐに停止され、ColdFusion により cfabort タグを含む行までのページコンテンツが返されます。

このタグを showError 属性とともに使用する一方で、cferror タグを使用してエラーページを定義しなかった場合は、cfabort タグに到達したときにページ処理が停止されます。また、showError 内のメッセージがクライアントに表示されます。

このタグを showError 属性とともに使用し、同時に cferror を使用してエラーページを定義すると、cferror タグで指定されたエラーページに出力がリダイレクトされます。

注意：cfabort、cflocation または cfcontent タグを使用した場合は、OnRequestEnd ではなく OnAbort メソッドが呼び出されます。

例

この例は、処理を停止するための cfabort の使用方法を示します。cfabort が使用されている 2 番目の例では、結果が表示されません。

```
 <h3>Example A: Let the instruction complete itself</h3>
<!--- first, set a variable --->
<cfset myVariable = 3>
<!--- now, perform a loop that increments this value --->
<cfloop from = "1" to = "4" index = "Counter">
    <cfset myVariable = myVariable + 1>
</cfloop>

<cfoutput>
<p>The value of myVariable after incrementing through the loop #Counter# times is:
    #myVariable#</p>
</cfoutput>

<h3>Example B: Use cfabort to halt the instructions with showmessage attribute and
    cferror</h3>
<!--- Reset the variable and show the use of cfabort. --->
<cfset myVariable = 3>
<!--- Now, perform a loop that increments this value. --->
<cfloop from = "1" to = "4" index = "Counter">
<!--- On the second time through the loop, cfabort. --->
    <cfif Counter is 2>
    <!--- Take out the cferror line to see cfabort error processed by CF error page. --->
        <cferror type="request" template="request_err.cfm">
        <cfabort showerror="CFABORT has been called for no good reason">
<!--- Processing is stopped, --->
<!--- and subsequent operations are not carried out.--->
    <cfelse>
        <cfset myVariable = myVariable + 1>
    </cfif>
</cfloop>

<cfoutput>
<p> The value of myVariable after incrementing through the loop#counter# times is: #myVariable#</p>
</cfoutput>
```

cfajaximport

説明

ColdFusion の AJAX タグや AJAX 機能を使用するページで使用するためにインポートされる JavaScript ファイルを制御します。

カテゴリ

インターネットプロトコルタグ

シンタックス

```
 <cfajaximport
    cssSrc = "local URL path"
    params = "parameters"
    scriptSrc = "local URL path"
    tags = "comma-delimited list">
```

注意：このタグの属性は attributeCollection で指定でき、その値は構造体になります。attributeCollection で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfform、cfgrid、cfinput、cflayout、cfmenu、cfpod、cfsprydataset、cftextarea、cftooltip、cftree、cfwindow、『ColdFusion アプリケーションの開発』の Specifying client-side support files

履歴

ColdFusion 8: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
cssSrc	オプション	**scriptsrc**/ajax	ColdFusion AJAX 機能 (リッチテキストエディタを除く) で使用される CSS ファイルが存在するディレクトリの URL (Web ルートからのパス) を指定します。このディレクトリは、<Web のルートディレクトリ >/CFIDE/scripts/ajax/resources ディレクトリと同じディレクトリ構造であり、同じ CSS ファイルとその CSS ファイルに必要なイメージファイルが含まれている必要があります。 この属性を使用すると、さまざまなアプリケーションで ColdFusion AJAX コントロールのさまざまなカスタムスタイルを作成できます。
params	オプション		この属性では、CFM ページのパラメータを指定できます。 現在、指定できるパラメータは googlemapkey のみです。次のように、googlemapkey を指定できます。 <cfajaximport params=#{googlemapkey="Map API Key"}#>
scriptSrc	オプション	Administrator での scriptsrc の設定 デフォルトパスは /CFIDE/scripts/	ColdFusion で使用されるクライアントサイドのスクリプトファイルが存在するディレクトリの URL (Web ルートからのパス) を指定します。このディレクトリには、すべての AJAX 機能で使用される JavaScript ファイル、および CSS ファイルのデフォルトの場所が含まれています。 この属性を使用する場合、cfajaximport タグは、ページ上の他のすべての ColdFusion AJAX タグ (スクリプトに依存するすべてのタグ) よりも前に配置する必要があります。 cfajaximport タグまたは cfform タグの場合、1 ページに使用できる scriptsrc 属性は 1 つのみです。cfajaximport タグで scriptsrc 属性を使用すると、ページ内のすべてのフォームにその値を適用できます。
tags	オプション		このページで JavaScript ファイルをインポートする必要があるタグまたはタグ属性の組み合わせを示すカンマ区切りリスト。 この属性を使用する場合は、このページおよびタグの source 属性に指定されているページで使用する ColdFusion AJAX タグをすべて指定する必要があります。 有効な属性値およびそれらの目的については、「使用方法」を参照してください。

使用方法

scriptsrc 属性および cssSrc 属性の使用

scriptsrc 属性は、JavaScript ファイルがデフォルトの場所にない場合に役立ちます。この属性は、/CFIDE ディレクトリへのアクセスをブロックする一部のホスティング環境および構成で必要です。

デフォルトの scriptsrc の値は、ColdFusion Administrator の [サーバーの設定]-[設定] ページの [デフォルト CFFORM ScriptSrc ディレクトリ] 設定によって決まります。cfform タグでは、この属性よりも cfform タグの scriptsrc 属性が優先されます。

この属性を使用できるのは、トップレベルのページに cfajaximport タグがある場合のみです。トップレベルのページとは、クライアントが直接リクエストするページのことです。たとえば、cfwindow タグの source 属性で指定されるページでは使用できません。

cfajaximport タグで scriptsrc 属性を使用する場合、指定したディレクトリは、/CFIDE/scripts ディレクトリと同じ構造である必要があります。たとえば、scriptsrc="/resources/myScripts" を指定した場合は、AJAX で使用される JavaScript ファイルが /resources/myScripts/ajax ディレクトリに存在している必要があります。

この属性は、現在のページにある後続のすべてのタグ (AJAX ベースのタグのみでなく) に対する ColdFusion のクライアントサイドファイルが存在するフォルダを指定します。したがって、ディレクトリツリーには、これらのタグで使用される ColdFusion のクライアントサイドファイルがすべて含まれている必要があります。たとえば、ページ内の cfform タグの形式が Flash またはアプレットの場合は、scriptsrc 属性で指定されているディレクトリ内に CF_RunActiveContent.js ファイルを含めます。

ColdFusion AJAX 機能に必要な CSS ファイルの場所を指定するには、cssSrc 属性を使用します。この属性は、現在のページの **scriptsrc**/ajax/resources ディレクトリよりも優先されます。したがって、カスタムの scriptsrc ディレクトリを使用するすべてのページで、カスタムの cssSrc ディレクトリも使用する場合は、scriptsrc ディレクトリツリー内に ColdFusion AJAX CSS ファイルを含める必要はありません。

tags 属性の使用または属性の使用なし
AJAX UI 機能を持つ ColdFusion タグが含まれるページでは、cfajaximport タグが使用されていなくても、ほとんどの場合は、必要な JavaScript ファイルが ColdFusion によって正しくインポートされます。このタグは、次の場合に JavaScript ファイルを明示的にインポートするために使用します。

- 他の方法では AJAX JavaScript 関数がインポートされないページで ColdFusion.navigate、ColdFusion.Ajax.submitForm、ColdFusion.Log.info などの ColdFusion AJAX JavaScript 関数を使用する場合。この場合は、属性を指定せずに cfajaximport タグを使用して、ベース JavaScript 関数のみをインポートします。たとえば、ColdFusion AJAX ベースタグが含まれていないページでこのタグを使用します。
- 次の条件に該当する場合
  - cflayoutarea、cfpod、または cfwindow タグで source 属性を使用しているか、cfdiv タグで bind 属性を使用している場合
  - source 属性または bind 属性で指定されているファイルに、次の表にリストされているタグが含まれている場合
  - リストされているタグをトップレベルのページで使用していない場合

  これらの条件に該当する場合は、トップレベルのページで cfajaximport タグを使用し、他のページでのみ使用されるタグを指定する tags 属性を指定する必要があります。そうしないと、どのようなタグが使用されるかを ColdFusion が判断できないので、必要な JavaScript ファイルがインポートされません。

次のいずれかまたはすべてのタグ属性値を指定できます。

属性値	用途
cfdiv	cfdiv タグ
cfform	cfpod、cfwindow、または cflayoutarea タグ本文にあるフォーム
cfgrid	AJAX 形式の cfgrid タグ
cfinput-autosuggest	autosuggest 属性を使用する cfinput タグ
cfinput-datefield	datefield 属性を使用する HTML 形式の cfinput タグ
cflayout-border	type 属性の値が border に設定された cflayout タグ
cflayout-tab	type 属性の値が tab に設定された cflayout タグ
cfmenu	cfmenu タグ
cfpod	cfpod タグ
cfsprydataset-JSON	Spry JSON データセットを生成する cfsprydataset タグ
cfsprydataset-XML	Spry XML データセットを生成する cfsprydataset タグ
cftextarea	HTML 形式の cftextarea タグ

属性値	用途
cftooltip	cftooltip タグ
cftree	HTML 形式の cftree タグ
cfwindow	cfwindow タグ

例

次の cfajaximport タグの例では、AJAX 機能に使用されるスクリプトおよび AJAX CSS ファイル用に別々のカスタムの場所が指定されています。また、cftree と cftooltip で使用されるすべての JavaScript ファイルがインポートされます。

```
<cfajaximport cssSrc="/collegeApp/application/cssFiles"
   scriptsrc="/collegeApp/ajaxScripts"
   tags="cftooltip, cfwindow">
```

cfajaxproxy

説明

AJAX クライアントで使用する ColdFusion コンポーネントの JavaScript プロキシを作成します。または、コントロール属性値にバインドされる CFC メソッド、JavaScript 関数、または URL のプロキシを作成します。

カテゴリ

インターネットプロトコルタグ

シンタックス

```
<cfajaxproxy
   cfc = "CFC name"
   jsclassname = "JavaScript proxy class name">
OR

<cfajaxproxy
   bind = "bind expression"
   onError = "JavaScript function name"
   onSuccess = "JavaScript function name">
```

注意：このタグの属性は attributeCollection で指定でき、その値は構造体になります。attributeCollection で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

DeserializeJSON、IsJSON、SerializeJSON、『ColdFusion アプリケーションの開発』の Using Ajax Data and Development Features

履歴

ColdFusion 8: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
bind	bind または cfc 属性が必須		呼び出す CFC メソッド、JavaScript 関数、または URL を指定するバインド式です。バインド式の指定に関する詳細については、『ColdFusion アプリケーションの開発』の Binding data to form fields を参照してください。 この属性は cfc 属性と一緒には使用できません。
cfc	bind または cfc 属性が必須		プロキシを作成する CFC です。CFC へのパスをドット区切り形式で指定します。パスは、絶対ファイルパスで指定しても、CFML ページの場所を基準とした相対ファイルパスで指定してもかまいません。たとえば、ColdFusion ページの cfcs サブディレクトリに myCFC という CFC がある場合は、cfcs.myCFC と指定します。 CFC によって別の CFC が拡張される場合は、拡張された CFC メソッドも JavaScript プロキシに対して使用可能になります。 UNIX ベースのシステムでは、まず指定された属性値に対応する名前またはパス (ただしすべて小文字) のファイルが検索されます。該当するファイルが見つからない場合は、大文字と小文字の区別も含めて属性値と完全に一致するファイル名またはパスが検索されます。 この属性は bind 属性と一緒には使用できません。
jsclassname	オプション	cfc 属性の値	CFC を表す JavaScript プロキシクラスに使用する名前です。 この属性は bind 属性と一緒には使用できません。
onError	オプション		bind 属性で指定したバインドが失敗したときに呼び出す JavaScript 関数の名前です。この関数は 2 つの引数 (エラーコードとエラーメッセージ) を取る必要があります。 この属性は cfc 属性と一緒には使用できません。
onSuccess	オプション		bind 属性で指定したバインドが成功したときに呼び出す JavaScript 関数の名前です。この関数は 1 つの引数 (bind 関数の戻り値) を取る必要があります。bind 関数が CFC 関数の場合は、戻り値が JavaScript 変数に自動的に変換されてから onSuccess 関数に渡されます。 この属性は cfc 属性と一緒には使用できません。

使用方法

ColdFusion Administrator の [サーバーの設定]-[設定] ページで [HTTP ステータスコードの有効化] オプションが選択されていることを確認してください。このオプションが選択されていないと、CFC 関数の呼び出しでエラーが発生したかどうかをプロキシが判断できないので、エラーハンドラを呼び出せません。

bind 属性が設定された cfajaxproxy タグでは、バインド式に指定されていないコントロールは更新されません。

bind 属性に URL を指定する場合は、onSuccess の引数に対応するオブジェクト、配列、または単純値の JSON 表記が HTTP 応答として返される必要があります。

CFC プロキシの作成

cfc 属性が設定された cfajaxproxy タグを使用すると、Web クライアント上の CFC を表す JavaScript プロキシが生成されます。このタグとプロキシには次の特徴があります。

- プロキシは、各 CFC リモート関数に対応する関数を 1 つ提供します。クライアントサイドの JavaScript コードでこれらの関数を呼び出すと、サーバー上の CFC 関数がリモートで呼び出されます。
- プロキシは、クライアントとサーバーのやり取りを設定するための関数を提供します。これらの関数によって、プロキシがサーバーとやり取りするときに使用する XMLHttpRequest 呼び出しの HTTP メソッドと同期モードが設定されます。また、非同期呼び出しの場合の JavaScript コールバックハンドラとエラーハンドラも指定できます。
- JavaScript では大文字と小文字が区別されるので、クライアントに送信する ColdFusion の構造体やスコープに指定されているキーの大文字と小文字が一致していることを確認します。ColdFusion の変数名や構造要素名は、デフォルトで

は大文字で記述されます (myStruct["myElement"]="value" のように連想配列表記法で名前を指定すると、構造体の要素名を小文字で作成できます)。たとえば、クエリーを表すために ColdFusion の serializeJSON タグで生成される JSON オブジェクト内の 2 つの配列のキーは、columns と data ではなく、COLUMNS と DATA になります。

AJAX CFC プロキシの使用方法の詳細については、『ColdFusion アプリケーションの開発』の Using AJAX Data and Development Features の Using ColdFusion AJAX CFC proxies を参照してください。

注意：プロキシは、呼び出した CFC 関数に _CF_NODEBUG というブール型の引数を渡します。この値が true の場合、ColdFusion はデバッグ情報を追加せずに応答を送信します (通常は、応答の最後にデバッグ情報が追加されます)。この動作により、JSON 以外のテキスト (つまりデバッグ情報) が AJAX リクエストに対する JSON 応答に含まれなくなります。

CFC プロキシユーティリティ関数

cfc オプションを使用する場合は、サーバーとのやり取りを制御するために、次の関数が JavaScript プロキシオブジェクトによって提供されます。

関数	説明
setAsyncMode()	呼び出しモードを非同期に設定します。プロキシ関数を呼び出したときに呼び出し元のスレッド (ページ処理を行っているクライアントシステムの Java スレッド) がブロックされないので、サーバーからの応答を待っている間もページ処理を続行できます。 プロキシは、サーバーからの応答を使用して、setCallbackHandler 関数で指定されている関数を呼び出します。エラーが発生した場合、プロキシは setErrorHandler 関数で指定されているエラーハンドラを呼び出します。
setCallbackHandler(< 関数 >)	非同期呼び出しの場合のコールバックハンドラを指定します。この < 関数 > パラメータは、引数として呼び出す JavaScript 関数です。 このコールバック関数は 1 つのパラメータ (プロキシによって JSON 表記から JavaScript 表記にシリアル化解除された CFC からの戻り値) を取る必要があります。 このメソッドは、自動的に呼び出しモードを非同期に設定します。
setErrorHandler(< 関数 >)	非同期呼び出しでエラーが発生した場合にプロキシが呼び出すエラーハンドラを設定します。この < 関数 > パラメータは、呼び出す JavaScript 関数です。 このエラーハンドラ関数は次の 2 つのパラメータを取る必要があります。 • HTTP エラーコード • ステータスメッセージ このメソッドは、自動的に呼び出しモードを非同期に設定します。
setForm(*ID*)	この関数の直後に呼び出されたプロキシ関数から渡された引数に、ID 属性で指定されているフォームのフィールドの名前と値を追加します。 詳細については、『ColdFusion アプリケーションの開発』の Submitting data to a CFC を参照してください。
setHTTPMethod("< メソッド >")	呼び出しに使用する HTTP メソッドを設定します。この "< メソッド >" パラメータでは、次のいずれかの値を指定する必要があります (大文字と小文字は区別されません)。 • GET (デフォルトのメソッド) • POST

関数	説明
setQueryFormat(< 形式 >)	ColdFusion クエリーデータを返す JSON 形式を指定します。パラメータの有効な値は次のとおりです。 • row: (デフォルト) 列名と行配列の配列の 2 つのエントリを含む JSON オブジェクトとしてデータを送信します。 • column: WDDX クエリー形式を表す JSON オブジェクトとしてデータを送信します。このオブジェクトには、3 つのエントリ (行数、列名を含む配列、キーが列名で値が列データを含む配列であるオブジェクト) が含まれています。 クエリー形式の詳細については、SerializeJSON を参照してください。
setReturnFormat(< 形式 >)	CFC 関数から返される結果の形式を指定します。ColdFusion では、関数の戻り値が指定した形式に変換されてからクライアントに返されます。 パラメータの有効な値は次のとおりです。 • json (この関数を使用しない場合のデフォルトの形式) • plain • wddx plain を指定し、サーバーからのレスポンスの "content-type" ヘッダを text/xml に設定した場合は、プロキシから呼び出し元またはコールバック関数に XML オブジェクトが返されます。コンテンツタイプが text/xml に設定されていない場合、サーバーからの戻り値はそのまま返されます。 この関数は、ブラウザに XML またはプレーン文字列を返す場合に役立ちます。
setSyncMode()	呼び出しモードを同期 (デフォルトの同期モード) に設定します。呼び出し元のスレッドは、応答が返されるまでブロックされたままになります。エラーが発生した場合は、プロキシから例外が返されます。同期モードでは、CFC プロキシ内のメソッドから呼び出し元に直接 CFC メソッドの結果が返されます。

例

次の例では、リモート CFC メソッドを使用して従業員ドロップダウンリストを設定しています。リストから名前を選択すると、CFC メソッドが呼び出されて従業員に関する情報が取得され、結果が表示されます。

アプリケーションページには次の行が含まれています。

```
 <!--- The cfajaxproxy tag creates a client-side proxy for the emp CFC.
        View the generated page source to see the resulting JavaScript.
        The emp CFC is in the components subdirectory of the directory that
        contains this page. --->
<cfajaxproxy cfc="components.emp" jsclassname="emp">

<html>
    <head>
        <script type="text/javascript">

            // Function to find the index in an array of the first entry
            // with a specific value.
            // It is used to get the index of a column in the column list.
            Array.prototype.findIdx = function(value){
                for (var i=0; i < this.length; i++) {
                    if (this[i] == value) {
                        return i;
                    }
                }
            }

            // Use an asynchronous call to get the employees for the
            // drop-down employee list from the ColdFusion server.
```

```
var getEmployees = function(){
    // create an instance of the proxy.
    var e = new emp();
    // Setting a callback handler for the proxy automatically makes
    // the proxy's calls asynchronous.
    e.setCallbackHandler(populateEmployees);
    e.setErrorHandler(myErrorHandler);
// The proxy getEmployees function represents the CFC
// getEmployees function.
    e.getEmployees();
}

// Callback function to handle the results returned by the
// getEmployees function and populate the drop-down list.
var populateEmployees = function(res)
{
    with(document.simpleAJAX){
        var option = new Option();
        option.text='Select Employee';
        option.value='0';
        employee.options[0] = option;
        for(i=0;i<res.DATA.length;i++){
            var option = new Option();
            option.text=res.DATA[i][res.COLUMNS.findIdx('FIRSTNAME')]
                + ' ' + res.DATA[i][[res.COLUMNS.findIdx('LASTNAME')]];
            option.value=res.DATA[i][res.COLUMNS.findIdx('EMP_ID')];
            employee.options[i+1] = option;
        }
    }
}

// Use an asynchronous call to get the employee details.
// The function is called when the user selects an employee.
var getEmployeeDetails = function(id){
    var e = new emp();
    e.setCallbackHandler(populateEmployeeDetails);
    e.setErrorHandler(myErrorHandler);
// This time, pass the employee name to the getEmployees CFC
// function.
    e.getEmployees(id);
}
// Callback function to display the results of the getEmployeeDetails
// function.
var populateEmployeeDetails = function(employee)
{
    var eId = employee.DATA[0][0];
    var efname = employee.DATA[0][1];
    var elname = employee.DATA[0][2];
    var eemail = employee.DATA[0][3];
    var ephone = employee.DATA[0][4];
    var edepartment = employee.DATA[0][5];

    with(document.simpleAJAX){
        empData.innerHTML =
        '<span style="width:100px">Employee Id:</span>'
        + '<font color="green"><span align="left">'
        + eId + '</font></span><br>'
        + '<span style="width:100px">First Name:</span>'
        + '<font color="green"><span align="left">'
        + efname + '</font></span><br>'
        + '<span style="width:100px">Last Name:</span>'
        + '<font color="green"><span align="left">'
```

```
                    + elname + '</font></span><br>'
                    + '<span style="width:100px">Email:</span>'
                    + '<font color="green"><span align="left">'
                    + eemail + '</span></font><br>'
                    + '<span style="width:100px">Phone:</span>'
                    + '<font color="green"><span align="left">'
                    + ephone + '</font></span><br>'
                    + '<span style="width:100px">Department:</span>'
                    + '<font color="green"><span align="left">'
                    + edepartment + '</font></span>';
                }
            }

            // Error handler for the asynchronous functions.
            var myErrorHandler = function(statusCode, statusMsg)
            {
                alert('Status: ' + statusCode + ', ' + statusMsg);
            }

        </script>
    </head>

    <body>
        <!--- The form to display the employee drop-down list and
        employee data. --->
        <form name="simpleAJAX" method="get">
        List of Employees:   
        <select name="employee" onChange="getEmployeeDetails(this.value)">
            <script language="javascript">
                getEmployees();
            </script>
        </select>
        <br><br>
        <span id="empData"></span>
        </form>
    </body>
</html>
```

データソースからデータを取得する次のコンポーネントは、"application" ディレクトリの "components" サブディレクトリの "emp.cfc" というファイルに含める必要があります。この CFC は、ColdFusion に付属する cfdocexamples データソースを使用します。

```
 <cfcomponent>
    <cfset this.dsn = "cfdocexamples">
    <cffunction name="getEmployees" access="remote" returnFormat="json"
            output="false">
        <cfargument name="empid" required="no" type="string" default="0">
        <cfquery name="qryEmp" datasource="#this.dsn#">
            select * from Employees
            <cfif empid neq 0>
            where Emp_ID = #empid#
            </cfif>
        </Cfquery>
        <cfreturn qryEmp>
    </cffunction>
</cfcomponent>
```

cfapplet

説明
このタグからは、登録されているカスタム Java アプレットが参照されます。Java アプレットを登録するには、ColdFusion Administrator で、[拡張機能]-[Java アプレット] を選択します。

cfform タグ内でのこのタグの使用はオプションです。cfform 内で cfapplet タグを使用し、Administrator で method 属性が定義されている場合は、戻り値がフォームに取り込まれます。

カテゴリ
フォームタグ

シンタックス

```
<cfapplet
   appletSource = "applet name"
   name = "form variable name"
   align = "alignment option"
   height = "height in pixels"
   hSpace = "space on each side in pixels"
   notSupported = "message to display for non-Java browser"
   param_1 = "applet parameter name"
   param_2 = "applet parameter name"
   param_n = "applet parameter name"
   vSpace = "space above and below in pixels"
   width = "width in pixels">
```

注意：このタグの属性は attributeCollection で指定でき、その値は構造体になります。attributeCollection で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目
cfform、cfformgroup、cfformitem、cfgrid、cfinput、cfobject、cfselect、cfservlet、cfslider、cftextarea、cftree

履歴
ColdFusion MX:

- このタグを cfform タグ内で使用するという必要条件はなくなりました。
- このタグを cfform タグ内で使用した場合の動作が変わりました。具体的には、Administrator 内で method 属性が定義されていると、アプレットのメソッドの戻り値がフォームに取り込まれます。

属性

属性	必須 / オプション	デフォルト	説明
appletSource	必須		登録されているアプレットの名前です。
name	必須		アプレットのフォーム変数名です。
align	オプション		配置を表します。 • Left • Right • Bottom • Top • TextTop • Middle • AbsMiddle • Baseline • AbsBottom
height	オプション		アプレットの高さです (単位 : ピクセル)。
hSpace	オプション		アプレットの左右の余白です (単位 : ピクセル)。
notSupported	オプション	「説明」を参照	Java アプレットベースの cfform コントロールが含まれているページを、Java がサポートされていない (または Java のサポートが無効になっている) ブラウザで開いた場合に表示するテキストです。例 : notSupported = "ColdFusion の Java アプレットを表示するには、ブラウザが Java に対応していなければなりません。" デフォルト値 : ColdFusion Java アプレットを表示するには、 ブラウザが Java をサポートしている必要があります。
param_n	オプション		アプレットの登録済みパラメータです。ColdFusion Administrator 内のアプレットの値を上書きする場合にのみ指定します。
vSpace	オプション		アプレットの垂直方向の余白です (単位 : ピクセル)。
width	オプション		アプレットの幅です (単位 : ピクセル)。

使用方法

アプレットの method 属性は、Administrator の [Java アプレット] ビューでのみ指定できます。他の属性については、[Administrator] ビューのデフォルト値をそのまま使用するか、またはこのタグ内で値を指定してデフォルト値を上書きすることができます。

Java アプレットコンポーネントが JAR ファイルに保管されている場合は、[J2EE アーカイブ]-[ColdFusion Administrator] に情報を入力します。詳細については、『ColdFusion アプリケーションの開発』の Embedding Java applets を参照してください。

例

```
 <p>cfapplet lets you reference custom Java applets that have been
    registered using the ColdFusion Administrator.
<p>To register a Java applet, open the ColdFusion Administrator and
    click "Applets" link under "extensions" section.
<p>This example applet copies text that you type into a form. Type
    some text, and then click "copy" to see the copied text.

<cfform action = "index.cfm">
    <cfapplet appletsource = "copytext" name = "copytext">
</cfform>
```

cfapplication

説明

ColdFusion アプリケーションのスコープを定義します。クライアント変数のストレージを有効または無効にし、クライアント変数のストレージメカニズムを指定します。また、セッション変数を有効にし、アプリケーション変数のタイムアウトを設定します。

カテゴリ

アプリケーションフレームワークタグ

シンタックス

```
 <cfapplication
    datasource="data_source_name"
    name = "application name"
    applicationTimeout = #CreateTimeSpan(days, hours, minutes, seconds)#
    clientManagement = "yes|no"
    clientStorage = "data_source_name|Registry|Cookie"
    loginStorage = "cookie|session"
    googleMapKey = "map key"
    scriptProtect = "none|all|list"
    serverSideFormValidation = "yes|no"
    sessionManagement = "yes|no"
    sessionTimeout = #CreateTimeSpan(days, hours, minutes, seconds)#
    setClientCookies = "yes|no"
    setDomainCookies = "yes|no">
```

注意：このタグの属性は attributeCollection で指定でき、その値は構造体になります。attributeCollection で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfassociate、cferror、cflock、cfmessagebox、Application.cfc リファレンス、『ColdFusion アプリケーションの開発』の Designing and Optimizing a ColdFusion Application および Integrating J2EE and Java Elements in CFML Applications

履歴

ColdFusion 9: datasource、googleMapKey、serverSideFormValidation 属性が追加されました。

ColdFusion 8: secureJSON 属性と SecureJSONPrefix 属性が追加されました。

ColdFusion MX 7: scriptProtect 属性が追加されました。

ColdFusion MX 6.1: loginStorage 属性が追加されました。

ColdFusion MX:

- パーシスタントスコープの使用方法が変更されました。Server、Session、および Application スコープ変数が、メモリに構造体として保管されます。以前のリリースでは、Session スコープおよび Application スコープの変数だけがこの方法で保管されていました。また、UDF 関数のスコープには、構造体としてはアクセスできません。
- CFTOKEN 変数値を設定するアルゴリズムが変更されました。レジストリキー UUIDToken が 0 でない場合、ColdFusion では UUID から構築された数に乱数を加えた数が使用されます。UUIDToken が 0 の場合は、正の乱数 (整数) を使用して CFTOKEN 変数のデフォルト値が設定されます。以前のリリースでは、常に UUID から構築された数に乱数を加えた数が使用されていました。

属性

属性	必須 / オプション	デフォルト	説明
authCookie	オプション		ColdFusion の認証 Cookie 関連のプロパティが含まれる構造体です。
datasource	オプション		クエリーによってデータを取得するデータソースの名前です。
name	「説明」を参照		アプリケーション名です。長さは 64 文字までです。 アプリケーション変数およびセッション変数 : 必須。 クライアント変数 : オプション
applicationTimeout	オプション	ColdFusion Administrator の [変数] ページ内に指定された値	アプリケーション変数の有効期限です。CreateTimeSpan 関数と日付、時刻、分、および秒の値をカンマで区切って指定します。
clientManagement	オプション	no	• yes: クライアント変数を有効にします。 • no
clientStorage	オプション	registry	クライアント変数の保管方法を示します。 • datasource_name: ODBC またはネイティブのデータソースに保管します。Administrator でストレージレポジトリを作成する必要があります。 • registry: システムレジストリに保管します。 • cookie: クライアントコンピュータ上の Cookie 内に保管します。拡張可能です。クライアントがブラウザで Cookie を無効にしている場合、クライアント変数は機能しません。
exchangeServerVersion	オプション	2007	Microsoft Exchange Server のバージョンを指定します。 有効な値は次のとおりです。 • 2003 • 2007 • 2010 詳細を指定しない場合は、デフォルトで 2007 が使用されます。
googleMapKey	オプション		Web ページに Google マップを埋め込むために必要な Google Map API キーです。
loginStorage	オプション	cookie	• cookie: ログイン情報を Cookie スコープに保管します。 • session: ログイン情報を Session スコープに保管します。

属性	必須 / オプション	デフォルト	説明
scriptProtect	オプション	ColdFusion Administrator の [グローバルなスクリプト保護] の設定に基づいて決定されます。	変数をクロスサイトスクリプティング攻撃から保護するかどうかを指定します。 • none: 変数を保護しません。 • all: フォーム、URL、CGI、および Cookie の変数を保護します。 • ColdFusion スコープのカンマ区切りリスト : 指定したスコープ内の変数を保護します。 詳細については、「使用方法」を参照してください。
secureJSON	オプション	Administrator の値	リモート呼び出しへの応答として ColdFusion 関数が JSON 形式で返す値の前に、セキュリティの接頭辞を追加するかどうかを指定するブール値です。 デフォルト値は、Administrator の [サーバーの設定]-[設定] ページの [シリアル化 JSON への接頭辞付加] 設定の値 (デフォルトは false) です。cffunction タグ内でこの変数値を上書きすることができます。 詳細については、『ColdFusion アプリケーションの開発』の Improving security を参照してください。
serverSideFormValidation	オプション	yes	no の場合、フォームの送信時に cfform フィールドでの検証を無効にします。
secureJSONPrefix	オプション	Administrator の値	secureJSON の設定が true の場合に、リモート呼び出しへの応答として ColdFusion 関数が JSON 形式で返す値の前に挿入するセキュリティの接頭辞です。 デフォルト値は、Administrator の [サーバーの設定]-[設定] ページの [シリアル化 JSON への接頭辞付加] 設定の値 (デフォルトは JavaScript のコメント文字である //) です。 詳細については、『ColdFusion アプリケーションの開発』の Improving security を参照してください。
sessionCookie	オプション		ColdFusion のセッション Cookie 関連のプロパティが含まれる構造体です。
sessionManagement	オプション	no	• yes: セッション変数を有効にします。 • no
sessionTimeout	オプション	ColdFusion Administrator の [変数] ページ内に指定された値	セッション変数の有効期限です。CreateTimeSpan 関数と日付、時刻、分、および秒の値をカンマで区切って指定します。
setClientCookies	オプション	yes	• yes: クライアント Cookie を有効にします。 • no: CFID Cookie および CFTOKEN Cookie は、クライアントのブラウザに自動的には送信されません。セッション変数またはクライアント変数が使用されるすべてのページの URL 上で CFID および CFTOKEN を手作業でコーディングする必要があります。
setDomainCookies	オプション	no	• yes: クライアント変数保管のために Cookie を使用するときに、CFID Cookie と CFTOKEN Cookie およびすべてのクライアント変数に対してドメイン Cookie を使用します。クラスタで実行されるアプリケーションの場合は必須です。 • no: CFID Cookie、CFTOKEN Cookie、およびすべてのクライアント変数 Cookie に対してホスト固有の Cookie を使用します。

使用方法

このタグは、通常 "Application.cfm" ファイル内で、ColdFusion アプリケーションのデフォルト値を設定するために使用されます。

注意："Application.cfc" ファイルでアプリケーションのデフォルト値を設定することもできます。詳細については、1523 ページの「アプリケーション変数」を参照してください。

このタグが ColdFusion Administrator 内で無効になっていないかぎり、このタグを使用してアプリケーション変数を有効にすることができます。また、Administrator 設定は sessionManagement 属性よりも優先されます。詳細については、『ColdFusion 設定と管理』を参照してください。

クラスタ上で ColdFusion が稼動している場合は、clientStorage = "cookie" またはデータソース名を指定します ("registry" は指定できません)。

アプリケーション名が 64 文字を超えるときは、エラーが生成されます。

CFTOKEN 変数の長さは 8 バイトで、範囲は 10000000 ～ 99999999 です。

注意：ClientStorage=cookie を指定すると、cfflush タグに続いて設定された Client スコープ変数はクライアントブラウザに保管されません。

クロスサイトスクリプティング攻撃からの変数の保護

ScriptProtect 属性を使用すると、変数スコープをクロスサイトスクリプティング攻撃から保護できます。この攻撃では、クライアントがユーザーのアプリケーションを利用して悪意のあるコードをユーザーのブラウザに送信しようとします。この攻撃では、フォームフィールドや URL 変数のユーザー入力により、ユーザー出力用の CF 変数が設定されます。JavaScript、アプレット、オブジェクト参照などの悪意のあるコードを含むデータが送信され、ユーザーのシステム上で実行されます。

注意：デフォルトのスクリプト保護設定は、ColdFusion Administrator の [設定] ページにある [グローバルなスクリプト保護] オプションによって決定されます。scriptProtect 属性を使用すると、Administrator での設定よりも優先されます。Application.cfc 初期化コードを使用して保護に関する値を設定することもできます。

ColdFusion のクロスサイトスクリプティング保護処理は、ColdFusion がリクエストの最初にアプリケーション設定を処理するときに実行されます。したがって、ユーザーのリクエスト内の URL、Cookie、CGI、およびフォーム変数を処理することができます。デフォルトでは、object、embed、script、applet、および meta の各 HTML タグ名をテキスト **InvalidTag** に置き換えます。これらの名前をプレーンテキストで示すことが可能になり、タグ名として使われた場合はその名前を置換します。

ColdFusion のスコープの一部または全部を保護対象に指定することができます。ただし、未知のソースからの変数を持つことがあるのはフォーム、URL、CGI、および Cookie のスコープだけです。スコープを保護するには、追加の処理も必要になります。このため、all 属性の値はこれら 4 つのスコープだけに保護を適用します。

スクリプト保護のメカニズムでは、サーバー設定の <ColdFusion のルートディレクトリ >/lib/neo-security.xml ファイル、または J2EE 設定の <ColdFusion のルートディレクトリ >/WEB-INF/cfusion/lib/neo-security.xml ファイルで定義される正規表現を変数値に適用します。**CrossSiteScriptPatterns** 変数の正規表現を修正することにより、ColdFusion の置換パターンをカスタマイズすることができます。

サーバー変数、アプリケーション変数、およびセッション変数のロック

Server、Application、および Session スコープの変数を設定または更新する場合は、scope 属性を次の値に設定した状態で cflock タグを使用します。

- サーバー変数の場合は、server を指定します。
- アプリケーション変数の場合は、application を指定します。
- セッション変数の場合は、session を指定します。

これらのスコープの変数を読み取るコードをロックする必要がある場合もあります。スコープのロック方法については、406 ページの「cflock」を参照してください。

例

```
 <!--- This example shows how to use cflock to prevent race conditions during data updates to variables in
Application, Server, and Session scopes. --->
<h3>cfapplication Example</h3>
<p>cfapplication defines scoping for a ColdFusion application and enables or disables application and/or
session variable storage. This tag is placed in a special file called Application.cfm that automatically
runs before any other CF page in a directory (or subdirectory) where the Application.cfm file appears.</p>

<cfapplication name = "ETurtle"
sessionTimeout = #CreateTimeSpan(0, 0, 0, 60)#
sessionManagement = "Yes">

<!--- Initialize session and application variables used by E-Turtleneck. --->
<cfparam name="application.number" default="1">
<cfparam name="session.color" default= "">
<cfparam name="session.size" default="">

<cfif IsDefined("session.numPurchased") AND IsNumeric(trim(session.cartTotal))>
<!--- Use the application scope for the application variable to prevent race condition. This variable keeps
track of total number of turtlenecks sold. --->
    <cflock scope = "Application" timeout = "30" type = "Exclusive">
    <cfset application.number = application.number + session.numPurchased>
    </cflock>
</cfif>

<cfoutput>
E-Turtleneck is proud to say that we have sold #application.number# turtlenecks to date.
</cfoutput>
<!--- End of Application.cfm --->
```

cfargument

説明

コンポーネント定義内にパラメータ定義を作成します。また、関数の引数を定義します。cffunction タグ内で使用します。

カテゴリ

拡張タグ

シンタックス

```
 <cfargument
    name="string"
    default="default value"
    displayname="descriptive name"
    hint="extended description"
    required="yes|no"
    type="data type">
```

関連項目

cfcomponent、cffunction、cfinterface、cfinvoke、cfinvokeargument、cfobject、cfproperty、cfreturn

履歴

ColdFusion 10：restArgSource、restArgName の REST 属性が追加されました。

ColdFusion 8: component が ReturnType 属性の有効な値として追加されました。

ColdFusion MX 7: type 属性の xml 値が追加されました。

ColdFusion MX: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
name	必須		文字列です。引数名を指定します。
default	オプション		引数が渡されない場合に、デフォルトの引数値を指定します。
displayname	オプション	name 属性の値	CFC メソッドのパラメータの場合にのみ意味があります。イントロスペクションを使用して CFC についての情報を示すときに表示する値です。
hint	オプション		CFC メソッドのパラメータの場合にのみ意味があります。イントロスペクションを使用して CFC についての情報を示すときに表示するテキストです。hint 属性の値は、パラメータ説明行の中で displayname 属性値の後に表示されます。この属性を使用して、パラメータの目的を説明します。
required	オプション	no	**メモ :** Web サービスとして呼び出される場合は、定義方法に関係なく、すべての引数が必要です。 コンポーネントメソッドを実行するためにパラメータが必要かどうかを指定します。default 属性を指定した場合、パラメータは必須ではありません。 • yes • no

属性	必須 / オプション	デフォルト	説明
restArgSource	オプション		次のいずれかです。 • path：リソース URL からパラメーターを抽出します。URI パスのパラメーターはリクエスト URI から抽出され、パラメーター名は、cffunction で指定された restPath で指定された URI パスのテンプレート変数名に対応します。path パラメーターでは、デフォルト値は使用できません。 • query：URL クエリパラメーターからパラメーターを抽出します。ほとんどの場合、HTTP の GET メソッドで使用されます。GET メソッドを使用してデータを送信した場合、パラメーターと値は URL にエンコードされます。これらの値を抽出して適切な変数に割り当てるには、QueryParam を使用します。 • form：フォームの送信からパラメーターを抽出します。これは、HTTP の POST メソッドで使用します。 • cookie：Cookie から値を抽出します。 • header：HTTP ヘッダーからパラメーターを抽出します。 • matrix：マトリックス URI. からパラメーターを抽出します。詳しくは、http://www.w3.org/DesignIssues/MatrixURIs.html を参照してください。 値を指定しない場合、パラメーターは、リクエストの本文から取得されます。 cfhttp を使用して、引数を form または body で送信できます。form パラメーターの場合、restArgSource の値に form を指定すると、引数が使用されます。body パラメーターの場合、restArgSource を指定していないと、引数が使用されます。 サービスの呼び出し時に引数を body で渡すと、場合によっては、form として受け取られることがあります。例えば、<cfhttpparam type="body" value="arg=somevalue"> の場合です。これは、フォームの名前と値のペアを cfhttp で渡したり、本文を "name=value" で渡した場合は、本文のコンテンツが同一であるので、REST サービスは渡されたものを検出できないからです。 サービスで、body 引数を想定しているにもかかわらず cfhttp でサービスにアクセスした場合、引数は body で渡されませんが、それでも body 引数は空の文字列で受け取ります。 この属性を使用しても type 属性を指定していない場合は、デフォルトで string が type として見なされます。
restArgName	オプション		引数名にマッピング可能な名前です。 関数の呼び出し時には、着信リクエストから引数が抽出されます。restArgName を指定した場合、リクエストの restArgSource スコープでその名前が検索され、引数が読み込まれます。指定しない場合は、引数名が検索されます。 指定した場合、引数名は何の影響も与えません。正しい引数名を指定できるので、引数名を適切なパラメーターに対応させることができます。 REST サービスに渡される引数名が有効な Java 識別子でない場合、例えば名前にハイフンが含まれている場合（例：arg-name）などに、この属性を使用して属性名をマッピングできます。

属性	必須 / オプション	デフォルト	説明
type	オプション	any	タイプ名の文字列です。引数のデータ型を指定します。 • any • array • binary • boolean • component: 引数には ColdFusion コンポーネントを指定する必要があります。 • date • guid: この引数は **xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx** という形式の UUID または GUID でなければなりません。**x** は 16 進数の 1 文字 (0 ～ 9、A ～ F) を表します。 • numeric • query • string • struct • uuid: この引数は **xxxxxxxx-xxxx-xxxx-xxxxxxxxxxxxxxxx** という形式の ColdFusion UUID でなければなりません。**x** は 16 進数の 1 文字を表します (0 ～ 9、A ～ F)。 • variableName: ColdFusion 変数のネーミング規則に従った形式の文字列です。 • xml: XML オブジェクトと XML 文字列です。 • コンポーネント名 : type 属性の値が上記のどれにも当てはまらない場合、ColdFusion はそれを ColdFusion コンポーネントの名前として扱います。関数を実行したときに、渡された引数が指定の名前を持つ CFC でない場合はエラーになります。

使用方法

このタグは cffunction タグ内に指定する必要があり、cffunction タグ本文の中で他のどのタグよりも前に置く必要があります。

メソッドの呼び出し時に渡される引数は、メソッド本文から次のようにアクセスできます。

- 省略型のシンタックスの場合 : #myargument#

  (この例では、引数 myargument にアクセスします。)
- 引数スコープを配列として使用する場合 : #arguments[1]#

  (この例では、cffunction 内で最初に定義されている引数にアクセスします。)
- 引数スコープを構造体として使用する場合 : #arguments.myargument#

  (この例では、配列内の引数 myargument にアクセスします。)

例

```
 <!--- This example defines a function that takes a course number parameter and returns the course
description. --->
<cffunction name="getDescription">
    <!--- Identify argument. --->
    <cfargument name="Course_Number" type="numeric" required="true">
    <!--- Use the argument to get a course description from the database. --->
    <cfquery name="Description" datasource="cfdocexamples">
        SELECT Descript
        FROM Courses
        WHERE Number = '#Course_Number#'
    </cfquery>
    <!--- Specify the variable that the function returns. --->
    <cfreturn Description.Descript>
</cffunction>
```

cfassociate

説明

サブタグデータをベースタグとともに保存できるようにします。この機能は、カスタムタグに対してのみ有効です。

カテゴリ

アプリケーションフレームワークタグ

シンタックス

```
 <cfassociate
    baseTag = "base tag name"
    dataCollection = "collection name">
```

注意：このタグの属性は attributeCollection で指定でき、その値は構造体になります。attributeCollection で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfapplication、cferror、cflock、cfmessagebox、『ColdFusion アプリケーションの開発』の High-level data exchange

属性

属性	必須 / オプション	デフォルト	説明
baseTag	必須		ベースタグ名です。
dataCollection	オプション	AssocAttribs	ベースタグ内にサブタグデータを保管するための構造体です。

使用方法

ベースタグ内にサブタグデータを保存するには、このタグをサブタグ内から呼び出します。

ColdFusion では、サブタグ属性がベースタグに渡されると、それらはデフォルト名が AssocAttribs の構造体に保存されます。複数のサブタグを指定できるベースタグ内にサブタグ属性を区別して保存するには、dataCollection 属性に構造体名を指定します。この構造体は、**thistag**.collectionName という配列に追加されます。

カスタムタグコード内では、cfmodule タグの attributeCollection 属性を使用してタグに渡された属性は、それぞれ独立した値として保存され、それらの属性がカスタムタグの呼び出し側によって構造体にグループ化されたことは示されません。したがって、呼び出されたタグでは、特定の属性に値を割り当てた場合、サブタグの呼び出し時に使用した attributeCollection 属性内に渡された値が置き換えられます。

例

```
 <!--- Find the context. --->
<cfif thisTag.executionMode is "start">
    <!--- Associate attributes. --->
    <cfassociate baseTag = "CF_TAGBASE">

    <!--- Define defaults for attributes. --->
    <cfparam name = "attributes.happy" default = "yes">
    <cfparam name = "attributes.sad" default = "no">
...
```

cfauthenticate

説明

このタグは廃止になりました。新しいセキュリティツールを使用してください。詳細については、711 ページの「変換関数」および『ColdFusion アプリケーションの開発』の Securing Applications を参照してください。

履歴

ColdFusion MX: このタグは廃止されました。ColdFusion MX およびこれ以降のリリースでは機能しません。

cfbreak

説明

cfloop タグ内で使用します。ループを中断します。

カテゴリ

フロー制御タグ

シンタックス

```
 <cfbreak>
```

関連項目

cfabort、cfcontinue、cfexecute、cfif、cflocation、cfloop、cfthrow、cftry、『ColdFusion アプリケーションの開発』の cfloop and cfbreak

例

```
 <!--- This shows the use of cfbreak to return processing to the top of a loop.--->
<!--- Select courses; use cfloop to find a condition; then break the loop. --->
<!--- Check that number is numeric. --->
<cfif IsDefined("form.course_number")>
    <cfif Not IsNumeric(form.course_number)>
        <cfabort>
    </cfif>
</cfif>
<cfquery name="GetCourses" datasource="cfdocexamples">
    SELECT *
    FROM COURSES
    ORDER by NUMBER
</cfquery>

<p> This example uses CFLOOP to cycle through a query to find a value.
(In our example, a list of values corresponding to courses in the cfdocexamples
datasource). When the conditions of the query are met, CFBREAK stops the loop. </p>
<p> Please enter a Course Number, and hit the "submit" button: </p>
<cfform>
    <cfselect name="courseNum">
        <cfoutput query="GetCourses">
            <option value="#NUMBER#">#NUMBER#
        </cfoutput>
    </cfselect>
    <cfinput type="Submit" name="" value="Search on my Number">
</cfform>
<!--- If the courseNum variable is not defined, don't loop through the query.--->
<cfif IsDefined ("form.courseNum") IS "True">
<!--- Loop through query until value found, then use CFBREAK to exit query.--->
    <cfloop query="GetCourses">
        <cfif GetCourses.NUMBER IS form.courseNum>
            <cfoutput>
                <h4>Your Desired Course was found:</h4>
                <pre>#NUMBER# #DESCRIPT#</pre>
            </cfoutput>
            <cfbreak>
        <cfelse>
            <br> Searching...
        </cfif>
    </cfloop>
</cfif>
```

タグ c

cfcache

説明

ページのコピーをサーバーコンピュータまたはクライアントコンピュータに保管して、ページレンダリングのパフォーマンスを向上させます。これを行うために、このタグでは、ColdFusion ページにより返されたスタティック HTML が含まれるテンポラリファイルが作成されます。

このタグは、ユーザーがページにアクセスするたびにダイナミックコンテンツを取り込む必要がない場合に使用します。

単純な URL や、URL パラメータを含む URL に対して、このタグを使用できます。

カテゴリ

ページ処理タグ

シンタックス

```
<cfcache
    action = "action"
    dependsOn = "variable name list"
    directory = "directory path"
    expireURL = "wildcarded URL reference"
    id = "object identifier"
    idleTime = "decimal number of days"
    metadata = "variable name"
    name = "variable name"
    password = "password"
    port = "port number"
    protocol = "http://|https://"
    region = "region_name"
    stripWhiteSpace = "false|true"
    throwOnError = "false|true"
    timespan = "decimal number of days">
    useCache = "true|false"
    usequerystring = "false|true"
    username = "username"
    value = "value">

    The page fragment to be cached, if any.

</cfcache>
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfflush、cfheader、cfhtmlhead、cfsetting、cfsilent、キャッシュ関数

履歴

ColdFusion 10：region 属性が追加されました。

ColdFusion 9:

- 次の機能がサポートされるようになりました。
  - メモリ内でのキャッシュ。メモリがキャッシュのデフォルトの場所になりました。
  - ページフラグメントのキャッシュ。
  - キャッシュオブジェクトの追加、取得、消去機能を含む、特定のオブジェクトのキャッシュ。
  - キャッシュ依存関係の設定。
  - アイドルタイムアウトの設定。
  - キャッシュされたオブジェクトについてのメタデータの取得。
  - キャッシュされたページフラグメントからのスペースの削除機能。
  - キャッシュされたオブジェクトを消去するときにエラーが発生した場合に例外を返す機能。
- action 属性の get 値および put 値が追加されました。これらの値によって、オブジェクトのキャッシュがサポートされます。

- dependsOn、id、idleTime、key、metadata、name、stripWhiteSpace、throwOnError、useCache、usequerystring、value の各属性が追加されました。

ColdFusion MX:

- cachedirectory 属性および timeout 属性は非推奨になりました。これらの属性は今後のリリースでは機能せず、エラーが発生する可能性があります。
- timespan 属性が追加されました。
- ページのキャッシュ方法が変更されました。action 属性のデフォルト値である cache を使用すると、サーバー上とクライアント上でページがキャッシュされます (以前のリリースでは、このオプションを使用した場合、サーバー上のみでページがキャッシュされていました)。
- protocol と port の値のソースが変更されました。protocol および port のデフォルト値は、現在のページの URL から取得されます (以前のリリースでは、これらの値はそれぞれ、"http" と "80" でした)。
- ページをキャッシュするときのセッションステートの扱い方が変更されました。このタグでは、ColdFusion のログインにより安全が確保されているページなど、セッションステートに依存したページをキャッシュすることができます (以前のリリースでは、ページがキャッシュされるとセッションステートがクリアされていたため、認証が失われる原因となっていました)。
- ファイルのキャッシュ方法が変更されました。このタグでは、ファイルをキャッシュするファイル名に URL の hash() が使用されます (以前のリリースでは、"cfcache.map" ファイルが使用されていました)。

属性

属性	アクション	必須 / オプション	デフォルト	説明
action	すべて	オプション	serverCache	• cache: サーバーサイドおよびクライアントサイドのページをキャッシュします。 • clientcache: ブラウザサイドのページのみをキャッシュします。個人用ページをキャッシュする場合に、このタグを使用します。 • flush: 現在キャッシュされているページ、フラグメント、またはオブジェクトをキャッシュから削除します。キャッシュは、ユーザーが次回そのアイテムにアクセスしたときに更新されます。詳細については、「説明」を参照してください。 • get: キャッシュからオブジェクトを取得します。 • optimal: cache と同じです。 • put: オブジェクトをキャッシュに追加します。 • serverCache: サーバーサイドのみをキャッシュします。
dependsOn	cache、serverCache、optimal	オプション		変数のカンマ区切りのリストです。いずれかの変数の値が変更されると、キャッシュが更新されます。この属性には、変数のリストを返す式を指定できます。
directory	cache、serverCache、clientCache、optimal、flush、put	オプション	メモリ内でのキャッシュ	キャッシュディレクトリの絶対パスです。
expireURL	flush	オプション	キャッシュされているページをすべて消去	URL への参照です。"*/view.cfm?id=*" のようにワイルドカードを指定できます。指定された URL またはパターンに一致するページがキャッシュから消去されます。

属性	アクション	必須 / オプション	デフォルト	説明
id	flush、get、put	「説明」を参照		キャッシュされたオブジェクトの識別子です。この属性には、識別子のカンマ区切りリストを指定することも可能です。 この属性は、オブジェクトに対するすべての操作で必須です。したがって、この属性は get アクション、put アクション、およびオブジェクトの消去時に必要です。ページの消去時には必要ありません。
idleTime	cache、serverCache、clientCache、optimal、flush、put	オプション	アイドルタイムアウトなし	キャッシュされたアイテムが一定期間アクセスされない場合に、そのアイテムを消去します。 • 小数で表された日数です。例 : 4 分の 1 日 (6 時間) の場合は「.25」、1 日の場合は「1」、1 日半の場合は「1.5」 • CreateTimeSpan 関数の戻り値です。例 : "#CreateTimeSpan(0,6,0,0)#"
metadata	get	オプション		オブジェクトのメタデータを格納する構造体変数の名前です。get オペレーションでは、次のデータが返されます。 • timespan: キャッシュされたアイテムの有効期間です。キャッシュされたアイテムの timespan 属性の値が返されます。 • createdtime: キャッシュが作成された時刻です。 • lasthit: キャッシュされたアイテムが最後に使用された時刻です。 • hitnumber: キャッシュされたアイテムが使用された回数です。 • missnumber: キャッシュミスの回数です。
name	get	必須		取得したオブジェクトを格納する変数の名前です。
password	cache、serverCache、clientCache、optimal、flush	オプション		パスワードです。ページに Web サーバーレベルの認証が必要な場合に、この属性を指定します。
port	cache、serverCache、clientCache、optimal、flush	オプション	現在表示しているページのポート	キャッシュされたページの URL をリクエストしている Web サーバーのポート番号です。ColdFusion では、cfcache から cfhttp への内部呼び出しにおいて、ページ内の各 URL 変数が変換されます。これにより、ページ内のリンクの有効性を保つことができます。
protocol	cache、serverCache、clientCache、optimal、flush	オプション	現在表示しているページのプロトコル	キャッシュから URL を作成するときに使用するプロトコル • http:// • https://
region		オプション		キャッシュ領域に割り当てる名前です。
stripWhiteSpace	cache、serverCache、optimal	オプション	false	キャッシュされたページフラグメントから不要な空白文字を削除するかどうかを指定します。キャッシュされたページやオブジェクトに対する処理は行われません。
throwOnError	id 属性が指定された flush	オプション	false	flush アクションでエラーが発生した場合にエラーを返すかどうかを指定するブール値です。指定しない場合は、このアクションが失敗した場合でもエラーは返されません。この属性が true であれば、たとえば指定された id 値が無効である場合に、cfcatch ブロックでエラーを処理できます。

属性	アクション	必須 / オプション	デフォルト	説明
timespan	cache、serverCache、clientCache、optimal、flush、put	オプション	「説明」を参照	キャッシュからアイテムが消去されるまでの間隔です。 • 小数で表された日数です。例 : 4 分の 1 日 (6 時間) の場合は「.25」、1 日の場合は「1」、1 日半の場合は「1.5」 • CreateTimeSpan 関数の戻り値です。例 : "#CreateTimeSpan(0,6,0,0)#" デフォルトのアクションでは、アイテムがアイドル状態になって idleTime 属性に指定された時間が経過した場合、または cfcache action="flush" が実行された場合にアイテムが消去されます。
useCache	cache、serverCache、optimal	オプション	true	ページに対してキャッシュを使用するかどうかを指定します。この属性は開発中に役に立つことがあります。例えば、アプリケーションの状態に基づいてキャッシュを使用するタイミングを予測する関数を使用できます。
usequerystring		オプション	false	true の場合、クエリー文字列を含むテンプレートキャッシュ ID が生成されます。つまり、クエリ文字列が変化するたびに新しいテンプレートキャッシュが作成されます。 true に設定すると、テンプレートキャッシュを生成する際に、クエリ文字列で定義されている URL パラメーターも dependson 属性で考慮されます。 「使用方法」も参照してください。
username	cache、serverCache、clientCache、optimal、flush	オプション		ユーザー名です。キャッシュまたは消去するページで Web サーバーレベルの認証が必要な場合に、この属性を指定します。
value	put	必須		キャッシュするオブジェクトです。

使用方法

ページフラグメント : ページフラグメントをキャッシュするには、フラグメントを開始タグと終了タグの間にあるタグ本文内に配置します。タグ本文を使用してページやオブジェクト全体をキャッシュしないでください。

flush:flush アクションには 2 つの形式があります。1 つは ExpireURL 属性を使用して消去するページを指定する形式、もう 1 つは id 属性を使用して消去するオブジェクトを指定する形式です。オブジェクトを消去する場合、デフォルトではエラーは無視されます。throwOnError 属性に true 値を指定すると、アクションからエラーが返されるようになるため、catch ブロックを使用してエラーを処理できます。この機能は、無効なキャッシュ ID 値を使用しているかどうかを判断する場合に役立ちます。

ユーザー定義キャッシュ : ユーザー定義キャッシュを作成するには、次の手順を使用します。

1 次のコードを (<ColdFusion のルートディレクト >¥lib¥ にある) ehcache.xml に追加します。

```
<cache name="cf9"
maxElementsInMemory="10000"
eternal="false"
timeToIdleSeconds="86400"
timeToLiveSeconds="86400"
overflowToDisk="true"
diskSpoolBufferSizeMB="30"
maxElementsOnDisk="10000000"
diskPersistent="true"
diskExpiryThreadIntervalSeconds="3600"
memoryStoreEvictionPolicy="LRU"/>
```

2 ユーザー定義キャッシュを参照するには、次のように key 属性を使用します。

```
<cfcache key="cf9" timespan=#createtimespan(0,0,1,0)# >
      <cfoutput>#now()#</cfoutput>
<cfcache>
```

デフォルトでは、キャッシュはディスク上にではなくメモリ内に作成されます。デフォルトでは、アプリケーションごとに 10000 のオブジェクトキャッシュおよび 10000 のテンプレートキャッシュが作成されます。キャッシュに格納できるオブジェクトとテンプレートの数には制限があることに注意する必要があります。

デフォルトでは、キャッシュからディスクへのオーバーフローは false に設定されています。ディスクキャッシュを有効にするには、ehcache.xml で overflowTodisk を true に設定します。キャッシュされているデータをサーバーの再起動後も使用できるようにするには、diskPersistent を true に設定します。

ehcache.xml のプロパティの詳細については、次の URL にあるドキュメントを参照してください。

http://ehcache.org/

ColdFusion 8 以前のリリース

このタグに関する次の情報は、以前のリリースに対しても当てはまります。

このタグは、コンテンツが頻繁に更新されないページで使用します。このアクションを実行すると、ユーザーが使用するアプリケーションのパフォーマンスが大幅に向上します。

キャッシュされたページの出力は、クライアントのブラウザまたは ColdFusion サーバー上のファイルに保管されます。ColdFusion では、ページの出力がリクエストされるたびにそれを再生成してダウンロードするのではなく、キャッシュされた出力を使用します。ColdFusion では、timespan 属性の指定、または cfcache action=flush の呼び出しによってキャッシュが消去される場合にのみ、ページが再生成され、ダウンロードされます。

単純な形式でのキャッシュ機能を有効にするには、timespan 属性を指定した cfcache タグをページの先頭に挿入します。ColdFusion により、指定時間が経過するごとに、キャッシュからページのコピーが消去 (削除) され、新しいコピーがキャッシュされて、ユーザーからのアクセスに備えられます。

action 属性を使用すると、クライアントサイドでのキャッシュを指定することも、クライアントサイドとサーバーサイド両方でのキャッシュ (デフォルト) を指定することもできます。クライアントサイドでのキャッシュの利点は、ColdFusion のリソースが必要ないという点です。ここではブラウザ自身のキャッシュにページが保管され、パフォーマンスが高められます。クライアントサイドとサーバーサイド両方でのキャッシュの利点は、サーバーのパフォーマンスを最適化できるという点です。たとえば、ブラウザにページのキャッシュがない場合、サーバーは自身のキャッシュからデータを取得することができます。クライアントサイドとサーバーサイド両方でのキャッシュをお勧めします。また、サーバーサイドのみのキャッシュは使用しないでください。

個人用のコンテンツが含まれているページの場合、action = "clientcache" オプションを使用すると、個人用のページのコピーが他のユーザー用にキャッシュされるおそれがなくなります。

アプリケーションページでデバッグが有効にされていないかぎり、デバッグを設定しても cfcache では無効になります。キャッシュファイルの生成時、cfcache では cfsetting showDebugOutput = "no" が使用されます。

cfcache タグでは、URL パラメータを含む固有の URL それぞれが、キャッシュを行うための独立したページとして評価されます。たとえば、http://server/view.cfm?id=1 の出力と http://server/view.cfm?id=2 の出力は別々にキャッシュされます。

cfcache タグでは、cfhttp タグを使用してキャッシュするページのコンテンツが取得されます。ページへのアクセス中に HTTP エラーが発生した場合、コンテンツはキャッシュされません。ColdFusion エラーが発生すると、そのエラーがキャッシュされます。

詳細については、『ColdFusion アプリケーションの開発』の Caching ColdFusion pages that change infrequently を参照してください。

動作の変更

ColdFusion 9 までは、リクエストのクエリ文字列が自動的にページ識別子の一部として使用されていたので、異なるクエリ文字列（URL パラメーター）を持つページは互いに独立した形でキャッシュされていました。ただし、現在は、以前の動作を行うようにするには、新しいオプションの usequerystring 属性を指定して true の値を設定する必要があります。

例

```
<!--- This example produces as many cached files as there are URL parameter permutations.
You can see that the page is cached when the timestamp doesn't change.--->

<cfcache
    timespan="#createTimeSpan(0,0,10,0)#">
<body>

<h3>This is a test of some simple output</h3>

<cfoutput>
    This page was generated at #now()#<br>
</cfoutput>

<cfparam name = "URL.x" default = "no URL parm passed">
<cfoutput>The value of URL.x = # URL.x #</cfoutput>
```

cfcalendar

説明

Flash 形式のインタラクティブなカレンダーを HTML フォームまたは Flash フォームに配置します。XML 形式フォームではサポートされていません。ユーザーは、このカレンダーで日付を選択して、フォーム変数として送信することができます。

カテゴリ

フォームタグ

シンタックス

```
<cfcalendar
    name = "name of calendar"
    dayNames = "days of the week labels"
    disabled = "yes|no|no attribute value"
    enabled = "yes|no"
    endRange = "last disabled date"
    height = "height"
    mask = "character pattern"
    monthNames = "month labels"
    onBlur = "ActionScript to invoke"
    onChange = "ActionScript to invoke"
    onFocus = "ActionScript to invoke"
    selectedDate = "date"
    startRange = "first disabled date"
    style="Flash ActionScript style"
    tooltip = "text"
    visible = "yes|no"
    width = "width">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfform、cfgrid、cfinput、cfselect、cfslider、cftextarea、cftree、『ColdFusion アプリケーションの開発』の About Flash form styles

履歴

ColdFusion MX 7.01: onBlur イベントと onFocus イベントがサポートされるようになりました。

ColdFusion MX 7: タグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
name	必須		カレンダーの名前です。
dayNames	オプション	S、M、T、W、Th、F、S	カレンダーに表示される曜日名を設定するカンマ区切りのリストです。日曜日 (Sunday) が先頭で、残りの曜日が通常どおりにその後に続きます。
disabled	オプション	no	すべてのユーザー入力を無効にし、コントロールを読み取り専用にします。入力を無効にするには、属性を省略して disabled を指定するか、disabled="yes" (または ColdFusion で使用する true などの真を表すブール値) を指定します。入力を有効にするには、属性を省略するか、disabled="no" (または ColdFusion で使用する false などの偽を表すブール値) を指定します。
enabled	オプション	yes	Flash の場合のみ : コントロールを有効にするかどうかを指定するブール値です。無効なコントロールはライトグレーで表示されます。disabled 属性の反対です。
endRange	オプション		無効な日付の範囲の終了日です。ユーザーは、startRange 属性で指定する日付からこの日付までの中から日付を選択することはできません。
firstDayOfWeek	オプション	0	カレンダーでその週の最初の曜日を指定する 0 ～ 6 の範囲の整数です。0 は日曜日、6 は土曜日を示します。
height	オプション	Flash により決定	カレンダーの垂直方向の寸法はピクセル単位で指定します。
mask	オプション	MM/DD/YYYY	送信される日付の形式を指定するパターンです。マスク文字は次のとおりです。 • D は日です。0 ～ 2 個のマスク文字を使用できます。 • M は月です。0 ～ 4 個のマスク文字を使用できます。 • Y は年です。0、2、または 4 個の文字を使用できます。 • E は曜日です。0 ～ 4 個の文字を使用できます。 • その他の文字は、指定の場所に文字を挿入します。 マスクの詳細については、cfinput リファレンスページの「入力データのマスキング」を参照してください。
monthNames	オプション	January、February、March、April、May、June、July、August、September、October、November、December	カレンダーの最上部に表示される月名のカンマ区切りのリストです。
onBlur	オプション		カレンダーがフォーカスを失ったときに実行される ActionScript です。
onChange	オプション		ユーザーが日付を選択したときに実行される ActionScript です。
onFocus	オプション		カレンダーがフォーカスを取得したときに実行される ActionScript です。

属性	必須 / オプション	デフォルト	説明
selectedDate	オプション	なし (Flash は現在の月を表示)	最初に選択した日付です。フォームスキンで指定された色で強調表示されます。現在のロケールに応じた mm/dd/yyyy または dd/mm/yyyy の形式でなければなりません。(必要に応じて、setlocale 関数を使用してロケールを設定します。)
startRange	オプション		無効な日付の範囲の開始日です。ユーザーは、この日付から endRange 属性で指定する日付までの中から日付を選択することはできません。
style	オプション		Flash ActionScript のスタイルまたはカレンダーに適用するスタイルです。詳細については、『ColdFusion アプリケーションの開発』の Setting styles and skins in Flash forms を参照してください。
tooltip	オプション		Flash の場合のみ : マウスポインタをコントロールの上に置いたときに表示されるテキストです。
visible	オプション	yes	Flash の場合のみ : コントロールを表示するかどうかを指定するブール値です。表示されないコントロールが使用するスペースは空白です。
width	オプション	Flash により決定	カレンダーの水平方向の寸法はピクセル単位で指定します。

使用方法

cfcalendar タグは、カレンダーの月を表示します。そこには、月、年、その月の日を示すグリッド、および曜日を示す見出しが含まれています。カレンダーには次に進む矢印ボタンと前に戻る矢印ボタンがあるので、これらのボタンを使用して表示される月と年を変更することができます。

selectedDate 属性の値を使用する場合、その日付は緑で強調表示され、初期表示される月と年が決まります。表示される月と年を変更しても、その選択した日付は変わりません。ユーザーは、カレンダー上の別の日付をクリックすることにより、選択した日付を変更できます。onChange 属性では、ユーザーが日付を選択したときに実行される ActionScript イベントハンドラ関数を指定することができます。

現在の日付は反転表示で示されます。つまり、数字は白、背景は黒で表示されます。ただし、選択した日付が別の月や年のものである場合、次に進む矢印ボタンや前に戻る矢印ボタンをクリックしてその月や年に進まない限り、現在の日付は表示されません。

mask 属性では、アプリケーションに返される選択した日付の形式を指定することができます。

キーボードを使用して、cfcalendar コントロールにアクセスし、コントロールから日付を選択することができます。

- 上下左右の矢印キーを使用すると、現在選択されている日付を変更できます。
- Home キーと End キーを使用すると、月の開始日と終了日にそれぞれ進むことができます。
- Page Up キーと Page Down キーを使用すると、前月と翌月にそれぞれ進むことができます。

注意：cfcalendar タグは、XML 形式のフォームではサポートされていません。

例

この例では、Flash haloBlue スキンによる 200 ピクセル ×150 ピクセルのカレンダーを作成します。省略された月名および 2 文字の曜日名が表示されます。selectedDate 属性での指定に基づき、今日の日付が初期表示されます。[保存] ボタンをクリックすると、フォームは現在のページに送信され、送信された情報が表示されます。

この例には、3 つの dateField コントロールもあります。これらのコントロールを使用すると、カレンダーに表示される初期選択日付およびブロックアウトされる日付範囲を変更することができます。ブロックアウトされる日付範囲の初期値は、今日の日付の直前 4 日間です。

注意：この例は、mm/dd/yyyy の日付形式を使用しないロケールで機能するように修正する必要があります。そのためには、DateFormat 関数の代わりに LSDateFormat 関数を使用し、ロケールに適したマスク (dd/mm/yyyy など) を使用します。

```
<!--- Set initial selected and blocked-out dates.--->
<cfparam name="Form.startdate" default="#dateformat(now()-5, 'mm/dd/yyyy')#">
<cfparam name="Form.enddate" default="#dateformat(now()-1, 'mm/dd/yyyy')#">
<cfparam name="Form.selectdate" default="#dateformat(now(), 'mm/dd/yyyy')#">

<!--- If the form has been submitted, display the selected date. --->
<cfif isDefined("Form.submitit")>
    <cfoutput><b>You selected #Form.selectedDate#</b><br><br></cfoutput>
</cfif>

<b>Please select a date on the calendar and click Save.</b><br>
<br>
<cfform name="form1" format="Flash" skin="haloBlue" width="375" height="350" >
    <cfcalendar name="selectedDate"
        selectedDate="#Form.selectdate#"
        startRange="#Form.startdate#"
        endRange="#Form.enddate#"
        mask="mmm dd, yyyy"
        dayNames="SU,MO,TU,WE,TH,FR,SA"
        firstDayOfWeek="first day of the week in integer"
        monthNames="JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP, OCT, NOV, DEC"
        style="rollOverColor:##FF0000"
        width="200" height="150">
    <cfinput type="dateField" name="startdate" label="Block out starts"
        width="100" value="#Form.startdate#">
    <cfinput type="dateField" name="enddate" label="Block out ends" width="100"
        value="#Form.enddate#">
    <cfinput type="dateField" name="selectdate" label="Initial date" width="100"
        value="#Form.selectdate#" >
    <cfinput type="Submit" name="submitit" value="Save" width="100">
</cfform>
```

cfcase

説明

cfswitch タグ本文でのみ使用されます。cfswitch タグ内に指定された式が特定の値を取るときに実行されるコードを含みます。

カテゴリ

フロー制御タグ

シンタックス

```
<cfcase
    value = "value|delimited set of values"
    delimiters = "delimiter characters">
```

関連項目

cfdefaultcase、cfswitch、『ColdFusion アプリケーションの開発』の cfswitch, cfcase, and cfdefaultcase

履歴

ColdFusion 8: ColdFusion で cfcase の値を解析する方法が変更されました。以前は、数値の日付が設定された cfcase タグからは期待どおりの結果が返されませんでした。たとえば、<cfcase value="00"> と <cfcase value="0A"> は両方とも 0 と評価されていました。"0A" という値は、日付として処理されて、12/30/1899 から 0 日に変換されていました。"00" という値も 0

という値に評価されていました。このため、「CFCASE タグ用のコンテキスト検証エラーです。CFSWITCH には、値 "0.0" 用の重複した CFCASE があります。」という例外が発生していました。現在、cfswitch タグは期待どおりの結果を返します。

属性

属性	必須 / オプション	デフォルト	説明
value	必須		cfswitch タグの expression 属性と一致させる必要のある値を指定します。一致させる値を複数指定するときは、個々の値を delimiter 属性の文字で区切ります。指定する値は、単純な定数か、定数式でなければなりません。変数は指定できません。
delimiters	オプション	, (カンマ)	一致させる複数の値を区切るための区切り文字です。複数の区切り文字を指定した場合は、値を区切るときにどの文字を使ってもかまいません。

使用方法

cfcase タグ本文のコンテンツが実行されるのは、cfswitch タグの expression 属性の評価結果がこのタグの value 属性に指定した値と一致するときだけです。cfcase タグ本文のコンテンツには、HTML およびテキストと、CFML のタグ、関数、変数、および式を含めることができます。一部の言語とは異なり、cfcase タグを明示的に終了させる必要はありません。

1 つの cfcase タグを複数の expression 値に一致させることもできます。そのためには、一致させる値をデフォルトの区切り文字で区切ります。たとえば、次の行は "red"、"blue"、または "green" に一致させます。

```
<cfcase value="red,blue,green">
```

delimiter 属性を使用すれば、カンマの代わりに使う区切り文字を指定することができます。たとえば、次の行は "cargo, live"、"cargo, liquid"、"cargo, solid" に一致させます。

```
<cfcase value="cargo, live;cargo, liquid-cargo, solid" delimiters=";-">
```

例

この例では、1 ～ 10 の得点に応じて等級を表示します。一部の cfcase タグは複数の得点に一致します。わかりやすくするため、ここでは得点を 7 に設定しています。

```
<cfset score="7">
<cfswitch expression="#score#">
    <cfcase value="10">
        <cfset grade="A">
    </cfcase>
    <cfcase value="9;8" delimiters=";">
        <cfset grade="B">
    </cfcase>
    <cfcase value="7;6" delimiters=";">
        <cfset grade="C">
    </cfcase>
    <cfcase value="5;4;" delimiters=";">
        <cfset grade="D">
    </cfcase>
    <cfdefaultcase>
        <cfset grade="F">
    </cfdefaultcase>
</cfswitch>
<cfoutput>
    Your grade is #grade#
</cfoutput>
```

cfcatch

説明

cftry タグの内部で使用します。これらを併用すると、ColdFusion ページで発生する例外の検出や処理を実行できます。例外とは、データベースオペレーションの失敗、インクルードファイルの欠如、開発者によって指定されたイベントなど、ColdFusion ページ内での正常な命令の流れを妨げるイベントです。

カテゴリ

例外処理タグ

シンタックス

```
<cfcatch type = "exception type">
    Exception processing code here</cfcatch>
```

関連項目

cftry、cferror、cffinally、cfrethrow、cfthrow、onError、『ColdFusion アプリケーションの開発』の Handling Errors

履歴

ColdFusion 10：sessionCookie 属性と authCookie 属性が追加されました。

ColdFusion MX:

- SQLSTATE 値の動作が変更されました。cfcatch タグ内の SQLSTATE 戻り値は、データベースドライバのタイプによって異なります。
  - Type 1 (JDBC-ODBC ブリッジ) の場合、値は ColdFusion 5 での値と同じになります。
  - Type 4 (100% Java、ネイティブメソッドなし) の場合、値は異なる可能性があります。

  アプリケーションのフロー制御が SQLSTATE 値に依存している場合は、ColdFusion MX でアプリケーションを使用すると予期せぬ動作が発生することがあります。
- type="any" のときのこのタグの動作が変更されました。type="any" を指定した cfcatch タグを含める場合、ブロック内の最後の cfcatch タグに type="any" を指定して他のすべてのテストがその前に実行されていることを確認する必要はありません。ColdFusion により、一致度が最も高い cfcatch ブロックが検出されます。
- cfscript タグの動作が変更されました。このタグは、cftry タグおよび cfcatch タグと同じ役割を果たす try ステートメントと catch ステートメントを含みます。
- オブジェクトの修正について変更がありました。cfcatch によって返されたオブジェクトを修正することはできません。
- 返す例外が変更されました。cfcollection、cfindex、および cfsearch タグは、SEARCHENGINE 例外を返すことができます。以前のリリースでは、これらのタグの処理時にエラーが起きたときは UNKNOWN 例外のみが返されていました。

属性

属性	必須 / オプション	デフォルト	説明
name	オプション		cfcatch 式の変数名です。
type	オプション	any	• application: アプリケーション例外を検出します。 • database: データベース例外を検出します。 • template: ColdFusion ページ例外を検出します。 • security: セキュリティの例外を検出します。 • object: オブジェクトの例外を検出します。 • missingInclude: インクルードファイル欠如の例外を検出します。 • expression: 式の例外を検出します。 • lock: ロックの例外を検出します。 • *custom_type*: cfthrow タグで定義された特定のカスタム例外タイプを検出します。 • searchengine：Solr 検索エンジンの例外を検出します。 • any: すべての例外タイプを検出します。

使用方法

cftry ブロック内に少なくとも 1 つの cfcatch タグを記述する必要があります。cfcatch タグは cftry ブロックの最後に配置します。複数の cfcatch タグがある場合、ColdFusion はそれらを出現順にテストします。このタグには終了タグが必要です。

type="any" が指定されている場合、ColdFusion は CFML タグ、データソース、または外部オブジェクトからの例外を検出します。例外タイプを取得するには、次のようなコードを使用します。

```
#cfcatch.type#
```

アプリケーションで cfthrow タグを使用して、開発者定義の例外を返すことができます。これらの例外は、次の type オプションのいずれかによって検出できます。

- "**custom_type**"
- "Application"
- "Any"

custom_type タイプは、cfthrow タグで指定される開発者定義のタイプです。カスタムタイプをピリオドで連結された一連の文字列として定義した場合 ("MyApp.BusinessRuleException.InvalidAccount" など)、ColdFusion はその文字パターンに基づいてカスタムタイプを検出できます。ColdFusion は、cftry ブロック内で、一致する例外タイプの cfcatch タグを探します。このとき、最も条件の厳しいもの (全文字列が一致するもの) から条件の緩いものへという順序で検索が行われます。

たとえば、次のようなタイプを定義したとします。

```
<cfthrow type = "MyApp.BusinessRuleException.InvalidAccount">
```

このとき、次のような cfcatch タグがある場合は、このタグが例外を処理します。

```
<cfcatch type = "MyApp.BusinessRuleException.InvalidAccount">
```

このタグがない場合、次のような cfcatch タグがあれば、このタグが例外を処理します。

```
<cfcatch type = "MyApp.BusinessRuleException">
```

このタグもない場合、次のような cfcatch タグがあれば、このタグが例外を処理します。

```
<cfcatch type = "MyApp">
```

カスタム例外タイプを検出するために、cfcatch タグを任意の順序でコーディングすることができます。

type = "Application" を指定した cfcatch タグは、カスタム例外を定義する cfthrow タグ内で Application タイプが指定されているカスタム例外のみを検出します。

cfinclude タグ、cfmodule タグ、および cferror タグは、type = "template" の例外を返します。

cfcatch ブロック内で発生した例外を、その cfcatch タグを直接囲んでいる cftry ブロックで処理することはできません。ただし、cfrethrow タグを使用して現在アクティブな例外を再び返すことができます。

cfcatch タグの変数では、次の例外情報が提供されます。

cfcatch 変数	**内容**
cfcatch.type	タイプ : cfcatch で指定した例外タイプです。
cfcatch.message	メッセージ : 例外の診断メッセージが提供されている場合はそのメッセージ、提供されていない場合は空の文字列が cfcatch.message 変数に設定されます。
cfcatch.detail	CFML インタープリタからの詳細メッセージ、または cfthrow タグ内で指定されたメッセージです。cfthrow ではなく ColdFusion によって例外が生成されたときは、メッセージに HTML 形式を含めることができます。このメッセージは、どのタグが例外を返したのかを判別するのに役立ちます。
cfcatch.tagcontext	タグコンテキスト構造の配列です。それぞれは、例外発生時のアクティブなタグコンテキストのレベルを表します。
cfcatch.NativeErrorCode	type="database" に適用されます。例外に割り当てられるネイティブエラーコードです。通常、データベースドライバから、データベースオペレーションの失敗を診断するためのエラーコードが与えられます。デフォルト値は -1 です。
cfcatch.SQLState	type="database" に適用されます。例外に割り当てられる SQLState 値です。通常、データベースドライバから、データベースオペレーションの失敗を診断するためのエラーコードが与えられます。デフォルト値は -1 です。
cfcatch.Sql	type="database" に適用されます。データソースに送信された SQL ステートメント。
cfcatch.queryError	type= "database" に適用されます。データベースドライバによってレポートされたエラーメッセージ。
cfcatch.where	type="database" に適用されます。クエリーで cfqueryparam タグを使用する場合は、クエリーパラメータの名前 / 値のペアです。
cfcatch.ErrNumber	type="expression" に適用されます。内部式のエラー番号です。
cfcatch.MissingFileName	type="missingInclude" に適用されます。インクルードできなかったファイルの名前です。
cfcatch.LockName	type="lock" に適用されます。影響を受けたロックの名前です。ロックに名前が付けられていない場合、値は "anonymous" になります。
cfcatch.LockOperation	type="lock" に適用されます。エラーとなったオペレーションです (Timeout、Create Mutex、または Unknown)。
cfcatch.ErrorCode	type="custom" に適用されます。文字列エラーコードです。
cfcatch.ExtendedInfo	type="application" および "custom" に適用されます。カスタムエラーメッセージです。デフォルトの例外ハンドラでは表示できない情報です。

例

```
<!--- The cfcatch example that uses TagContext to display the tag stack. --->
<h3>cftry Example</h3>
<!--- Open a cftry block. --->
<cftry>
    <!--- Notice misspelled tablename "employees" as "employeeas". --->
    <cfquery name = "TestQuery" dataSource = "cfdocexamples">
        SELECT *
        FROM employees
    </cfquery>
    <!--- Other processing goes here. --->
    <!--- Specify the type of error for which we search. --->
    <cfcatch type = "Database">
        <!--- The message to display. --->
        <h3>You've Thrown a Database <b>Error</b></h3>
        <cfoutput>
            <!--- The diagnostic message from ColdFusion. --->
            <p>#cfcatch.message#</p>
            <p>Caught an exception, type = #CFCATCH.TYPE#</p>
            <p>The contents of the tag stack are:</p>
            <cfdump var="#cfcatch.tagcontext#">
        </cfoutput>
    </cfcatch>
</cftry>
```

cfchart

説明

チャートを生成し、表示します。

カテゴリ

データ出力タグ、拡張タグ

シンタックス

```
<!--- This syntax uses a JSON file to specify the chart style. --->
<cfchart
    format="html"
    style = "JSON filename">
</cfchart>

OR

<!--- This syntax uses the attributes of the cfchart tag to specify the chart style. --->
<cfchart
    alpha = "value between 0 and 1"
    arrows = "JSON string representation"
    aspect3D = "JSON string representation"
    background = "JSON string representation"
    bevel = "JSON string representation"
    border = "JSON string representation"
    backgroundColor = "hexadecimal value|web color"
    chartHeight = "integer number of pixels"
    chartWidth = "integer number of pixels"
    crosshair = "JSON string representation"
    dataBackgroundColor = "hexadecimal value|web color"
    fill = "JSON string representation"
    font = "font name"
    fontBold = "yes|no"
```

```
    fontItalic = "yes|no"
    fontSize = "font size"
    foregroundColor = "hexadecimal value|web color"
    format = "flash|jpg|png|html"
    gridlines = "integer number of lines"
    height = "height in pixels"
    ID = "chart identifier"
    labels = "JSON string representation"
    legend = "JSON string representation"
    labelFormat = "number|currency|percent|date"
    marker = "JSON string representation"
    markerSize = "integer number of pixels"
    name = "string"
    pieSliceStyle = "solid|sliced"
    plot = "JSON string representation"
    plotarea = "JSON string representation"
    preview = "JSON string representation"
    refresh = "canvas|flash|svg|vml"
    renderer = "canvas|flash|svg|vml"
    scales = "comma-seperated list of axes"
    scaleFrom = "integer minimum value"
    scaleTo = "integer maximum value"
    seriesPlacement = "default|cluster| stacked|percent"
    show3D = "yes|no"
    showBorder = "yes|no"
    showLegend = "yes|no"
    showMarkers = "yes|no"
    showXGridlines = "yes|no"
    showYGridlines = "yes|no"
    sortXAxis = "yes|no"
    tipBGColor = "hexadecimal value|web color"
    tipStyle = "MouseDown|MouseOver|none"
    title = "title of chart"
    tooltip = "JSON string representation"
    url = "onClick destination page"
    xAxis = "JSON string representation"
    xAxis2 = "JSON string representation"
    xAxisTitle = "title text"
    xAxisType = "scale|category"
    xAxisValues = "Array of values"
    xOffset = "number between -1 and 1"
    yAxis = "JSON string representation"
    yAxis2 = "JSON string representation"
    yAxisTitle = "title text"
    yAxisType = "scale|category"
    yAxisValues = "Array of values"
    yOffset = "number between -1 and 1">
    zoom = "JSON string representation"
</cfchart>
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfchartdata、cfchartseries、cfdocument、『ColdFusion アプリケーションの開発』の Controlling chart appearance

履歴

ColdFusion 10：新しく alpha、arrows、aspect3D、background、bevel、border、crosshair、fill、format、height、ID、labels、legend、plot、plotarea、preview、refresh、renderer、scales、type、tooltip、width、xaxis、axis2、xaxisvalues、yaxis、yaxis2、yaxisvalues、zoom の各属性が追加されました。

ColdFusion 8:

- チャートスタイルファイル (charting¥styles ディレクトリにある XML ファイル) に showLegend という新しい属性が追加されました。この属性は各ポイントのエントリを表示します。この属性はデータ系列を 1 つだけ含むチャートに対してのみ使用できます。デフォルトでは、showLegend の値は true に設定されます。この機能をオフにするには、すべてのチャートスタイルファイルで設定を変更するか、カスタムスタイルファイルを使用します。

ColdFusion MX 7.01：マニュアルが修正され、fontSize 属性で整数以外の数値を使用可能であるという記述に変更されました。

ColdFusion MX 7:

- style 属性と title 属性が追加されました。
- 8 桁の 16 進数字を使用して RGB カラーおよび透明度を指定できるようになりました。
- rotated 属性が削除されました。

ColdFusion MX 6.1:

- xAxisType 属性と yAxisType 属性が追加されました。
- 補間の動作が変更されました。タグは、複数の系列を持つ折れ線グラフのデータポイントを補間するようになりました。

ColdFusion MX: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
alpha	オプション		背景のアルファ（透明度）レベルです。有効な値は、0（透明）～ 1（不透明）です。
arrows	オプション		データやその他のチャート項目を指すための矢印を作成します。 to、from、size、label などの値が含まれる構造体の配列の JSON 文字列表現です。
aspect3D	オプション		3D の外観の角度を定義する構造体の JSON 文字列表現です。有効な構造体キーは、angle および depth です。
background	オプション		background に関連する次のキーの構造体です。 • color：背景色を設定します。 • color-1：矢印の最初の背景色を設定します。 • color-2：矢印の 2 番目の背景色（グラデーションで使用）を設定します。 • transparent：下層の色やチャートが表示されるように、背景イメージの透明度を設定します。 • fit：背景の領域に合わせた幅と高さを定義します。 • repeat：イメージの繰り返しタイプを定義します。 • image：背景イメージのパスを定義します。 • position：背景イメージの位置を定義します。
backgroundColor	オプション		ラベルの周囲および凡例の周囲の、データの背景とチャートボーダーとの間の領域の色です。 16 進数の値またはサポートされる色の名前を指定します。「使用方法」にある名前のリストを参照してください。16 進数の値を入力するには、"##xxxxxx" または "##xxxxxxxx" という形式を使用します。ここで、x は 0 ～ 9 または A ～ F です。シャープ記号 (#) は 2 つ使用するか、または使用しないでください。

属性	必須 / オプション	デフォルト	説明
bevel	オプション		bevel に関連する次のキーの構造体です。 • color：ベベルの色を定義します。 • blur-x：x 方向のベベルのエッジの鋭さ／滑らかさを定義します。 • blur-y：y 方向のベベルのエッジの鋭さ／滑らかさを定義します。 • angle：ベベルの角度を定義します。 • distance：# | #px 単位の距離で、ベベルが表示されるオブジェクトからの距離を示します。
border	オプション		border に関連する次のキーの構造体です。 • color：ボーダーの色を設定します。 • radius：丸みがある角の半径を定義します。 • width：ボーダーの幅を定義します。
chartHeight	オプション	240	チャートの縦幅です。ピクセル値 (整数) で指定します。
chartWidth	オプション	320	チャートの横幅です。ピクセル値 (整数) で指定します。
crosshair	オプション		crosshair に関連する次のキーの構造体です。 • line-color：十字線の色を設定します。 • alpha：線のアルファ透明度レベルを定義します。 • line-style：線のスタイルを定義します。
dataBackgroundColor	オプション	white	チャートデータの周囲の領域の色です。 16 進数の値またはサポートされる色の名前を指定します。「使用方法」にある名前のリストを参照してください。 16 進数の値を入力するには、"##xxxxxx" または "##xxxxxxxx" という形式を使用します。ここで、x は 0 ～ 9 または A ～ F です。シャープ記号 (#) は 2 つ使用するか、または使用しないでください。
fill	オプション		fill に関連する次のキーの構造体です。 • angle：直線状の塗りつぶしが表示されるときの角度を設定します。塗りつぶしの角度が 0 の場合、上部（background-color-1）から下部（background-color-2）まで垂直のグラデーションが表示されます。 • offset-x：背景のグラデーションの x 軸のオフセットを設定します。 • offset-y：背景のグラデーションの y 軸のオフセットを設定します。
format	必須	flash	レンダリングされるチャートの形式です。サポートされている形式は、html、flash、jpg および png です。

属性	必須 / オプション	デフォルト	説明
font	オプション	arial	テキストフォント名です。 • arial • times • courier • arialunicodeMS。このオプションは、UNIX 上で 2 バイト文字セットを使用する場合、または Windows 上で Flash のファイルタイプで 2 バイト文字セットを使用する場合には必須です。
fontBold	オプション	no	テキストをボールドにするかどうかを指定します。 • yes • no
fontItalic	オプション	no	テキストをイタリックにするかどうかを指定します。 • yes • no
fontSize	オプション	11	フォントサイズ。整数以外の数値を指定した場合は、直近の整数に繰り上げられます。
foregroundColor	オプション	black	テキスト、グリッド線、およびラベルの色です。 16 進数の値またはサポートされる色の名前を指定します。「使用方法」にある名前のリストを参照してください。 16 進数の値を入力するには、"##xxxxxx" または "##xxxxxxxx" という形式を使用します。ここで、x は 0 ～ 9 または A ～ F です。シャープ記号 (#) は 2 つ使用するか、または使用しないでください。
format	オプション	flash	グラフを保存するときのファイル形式です。 • flash • jpg • png
gridlines	オプション	10 (上端と下端も含む)	軸を含め、値軸に表示するグリッド線の数です。正の整数で指定します。
height	オプション		チャートの縦幅です。整数のピクセル値で指定します。
ID	オプション		チャートの ID です。基盤となるチャートオブジェクトを取得するために使用します。
labels	オプション		チャートにカスタムテキストやイメージ（例えば、作成者やチャートの情報）を表示するために使用する構造体の配列です。
labelFormat	オプション	number	Y 軸のラベルの形式です。 • number • currency • percent • date

属性	必須 / オプション	デフォルト	説明
legend	オプション		凡例の属性（background-color や margin-top など）を定義するために使用する構造体です。
markerSize	オプション	(自動)	データポイントマーカのサイズです (ピクセル単位)。整数で指定します。
name	オプション		ページ変数名です。文字列で指定します。グラフをバイナリデータとして生成し、指定した変数に割り当てます。チャートは表示されません。cffile タグ内でこの name 値を使用すれば、チャートをファイルに書き出すことができます。
pieSliceStyle	オプション	sliced	cfchartseries の type 属性の値が pie であるときに適用されます。 • solid: 円グラフを項目ごとに分割せずに表示します。 • sliced: 円グラフを項目ごとに分割して表示します。
plot	オプション		animation、aspect、margin、marker など、チャートのスタイル設定に使用するキーの構造体です。
plotarea	オプション		position、margin など、チャート領域のスタイル設定に使用するキーの構造体です。
preview	オプション		visible、margin など、チャートのプレビューを制御するためのキーの構造体です。
refresh	オプション		type、url、interval など、ダイナミックチャートを作成するためのキーの構造体です。
renderer	オプション	format が html の場合は canvas	レンダリング方法を指定します。format が html の場合にのみ適用されます。サポートされている値は、canvas、flash、svg および vml です。
scaleFrom	オプション	データにより決定	Y 軸の最小値です。整数で指定します。
scales	オプション		チャートをプロットする座標軸のカンマ区切りリストです（例：x,y2）。
scaleTo	オプション	データにより決定	Y 軸の最大値です。整数で指定します。
seriesPlacement	オプション	default	複数のデータ系列があるチャート内の系列の相対位置を指定します。 • default: ColdFusion では、グラフのタイプに基づいて相対位置が決定されます。 • cluster • stacked • percent
show3D	オプション	yes	チャートに 3 次元効果を付けて表示するかどうかを指定します。 • yes • no
showBorder	オプション	no	チャートの周囲にボーダーを表示するかどうかを指定します。 • yes • no
showLegend	オプション	yes	チャートに複数のデータ系列がある場合に、凡例を表示するかどうかを指定します。 • yes • no

属性	必須 / オプション	デフォルト	説明
showMarkers	オプション	yes	折れ線グラフ、曲線グラフ、および散布グラフのデータポイントにマーカーを表示するかどうかを指定します。 • yes • no
showXGridlines	オプション	no	X 軸のグリッド線を表示するかどうかを指定します。 • yes • no
showYGridlines	オプション	yes	Y 軸のグリッド線を表示するかどうかを指定します。 • yes • no
sortXAxis	オプション	no	列ラベルを X 軸に沿ってアルファベット順に表示するかどうかを指定します。 • yes • no xAxisType 属性が scale の場合は無視されます。
style	オプション		チャートのスタイルを指定する XML ファイルまたは文字列です。
title	オプション		チャートのタイトルです。
tipbgcolor	オプション	white	ヒントの背景色です。Flash 形式のグラフファイルのみに適用されます。 16 進数の値またはサポートされる色の名前を指定します。「使用方法」にある名前のリストを参照してください。 16 進数の値を入力するには、"##xxxxxx" という形式を使用します。ここで、x は 0 ～ 9 または A ～ F です。シャープ記号 (#) は 2 つ使用するか、または使用しないでください。
tipStyle	オプション	mouseOver	現在のチャート要素に関する情報を表示するポップアップウィンドウを開くアクションを決定します。 • mouseDown: ユーザーがカーソルを要素の位置に置いてマウスをクリックすると、表示されます。Flash 形式のグラフファイルのみに適用されます。その他の形式では、このオプションは、mouseOver と同様に機能します。 • mouseOver: ユーザーがカーソルを要素の位置に置くと表示されます。 • none: 表示されません。
tooltip	オプション		background、font、border など、ツールヒントのスタイル設定に使用するキーの構造体です。
type	オプション		チャートのタイプです。

属性	必須 / オプション	デフォルト	説明
url	オプション		ユーザーがデータ系列の項目をクリックしたときに開く URL を指定します。onClick での移動先ページです。 URL の文字列内には変数を指定できます。ColdFusion により、変数の現在の値が渡されます。 • $VALUE$: 選択した行の値です。選択しない場合、値は空の文字列になります。 • $ITEMLABEL$: 選択した項目のラベルです。選択しない場合、値は空の文字列になります。 • $SERIESLABEL$: 選択した系列のラベルです。選択しない場合、値は空の文字列になります。例 : "somepage.cfm?item=$ITEMLABEL$&series=$SERIESLABEL$&value=$VALUE$ • "javascript:....": クライアントサイドのスクリプトを実行します。
width	オプション	320	チャートの幅です（単位：ピクセル）。
xAxis	オプション		format、guide、item、label など、x 軸のスタイル設定に使用するキーの構造体です。
xAxis2	オプション		format、guide、item、label など、第 2 の x 軸のスタイル設定に使用するキーの構造体です。第 2 の x 軸は、チャートの上部に表示されます。
xAxisTitle	オプション		X 軸に表示されるタイトルです。テキストで指定します。
xAxisType	オプション	category	X 軸がデータを表すか、または数値を表すかを指定します。 • category: X 軸はデータカテゴリを表します。データは sortXAxis 属性に従って並べ替えられます。 • scale: X 軸は数値を表します。cfchartdata タグの item 属性の値はすべて数値でなければなりません。X 軸は自動的に数値順に並べ替えられます。
xAxisvalues	オプション		x 軸に表示する値の配列です。
xOffset	オプション	0.1	チャートを水平方向に傾斜させるように表示するときの単位数を指定します。show3D="yes" を指定する場合に適用されます。-1 ～ 1 の範囲の数字を使用できます。ここで、"-1" は左方向に 90 度、"1" は右方向に 90 度傾けることを意味します。
yaxis	オプション		format、guide、item、label など、y 軸のスタイル設定に使用するキーの構造体です。
yaxis2	オプション		format、guide、item、label など、第 2 の y 軸のスタイル設定に使用するキーの構造体です。第 2 の y 軸は、チャートの上部に表示されます。
yAxisTitle	オプション		Y 軸に表示されるタイトルです。テキストで指定します。
yAxisType	オプション	category	Y 軸は常にデータ値に使用されるので、現時点では効果がありません。
yaxisvalues	オプション		y 軸に表示する値の配列です。
yOffset	オプション	0.1	チャートを垂直方向に傾斜させるように表示するときの単位数を指定します。show3D="yes" を指定する場合に適用されます。-1 ～ 1 の範囲の数字を使用できます。ここで、"-1" は左方向に 90 度、"1" は右方向に 90 度傾けることを意味します。
zoom	オプション		alpha、background、bevel など、チャートをズームしたときに適用されるキーの構造体です。

使用方法

cfchart タグでは、グラフを表示するコンテナを定義します。ここでは、高さ、幅、背景色、ラベルなどを定義します。cfchartseries タグでは、棒グラフ、折れ線グラフ、円グラフなど、データを表示するチャートのスタイルを定義します。cfchartdata タグでは、データポイントを定義します。

データは、次の形で cfchartseries タグに渡されます。

- クエリーとして渡されます。
- cfchartdata タグで定義されたデータポイントとして渡されます。

font 属性の値 ArialUnicodeMS には、次のルールが適用されます。

- Windows の場合、Flash 形式のチャート (type = "flash") で 2 バイト文字セットをレンダリングできるようにするためには、この値を選択します。
- UNIX の場合、すべての type 値に対して、2 バイト文字セットをレンダリングできるようにするためには、この値を選択します。
- この値を選択した場合、fontBold 属性と fontItalic 属性は無効になります。

次の表に、color 属性で使用できる W3C HTML 4 カラー名または 16 進値を示します。

カラー名	**RGB 値**
Aqua	##00FFFF
Black	#000000
Blue	##0000FF
Fuchsia	##FF00FF
Gray	##808080
Green	##008000
Lime	##00FF00
Maroon	##800000
Navy	##000080
Olive	##808000
Purple	##800080
Red	##FF0000
Silver	##C0C0C0
Teal	##008080
White	##FFFFFF
Yellow	##FFFF00

その他の色を指定する場合は、16 進数値を入力します。RGB 値を指定する 6 桁の値、または RGB 値と透明度を指定する 8 桁の値を指定できます。8 桁の 16 進値の先頭の 2 桁は透明度を示します。FF は不透明を、00 は透明を表しています。00 ～ FF の範囲の値を使用できます。

一般的なブラウザでサポートされるその他のカラー名については、www.w3.org/TR/css3-color を参照してください。

また、チャートをメモリにキャッシュするかどうか、キャッシュするチャートの数、および ColdFusion で同時に処理できるチャートリクエストの数などを指定することができます。ColdFusion Administrator でこれらのオプションを設定するには、[サーバーの設定]-[チャート] を選択します。

クライアントサイドのチャート作成では、次の属性はサポートされません。format、labelformat、seriesplacement の percent の値、sort、xaxis、tipsstlye、url、xAxisTYpe、xoffset、yaxistype および yoffset。

例

```
<!---The following example analyzes the salary data in the cfdocexamples database and
    generates a bar chart showing average salary by department. The body of the
    cfchartseries tag includes one cfchartdata tag to include data that is not available
    from the query. --->

<!--- Get the raw data from the database. --->
<cfquery name="GetSalaries" datasource="cfdocexamples">
    SELECT Departmt.Dept_Name,
        Employee.Dept_ID,
        Employee.Salary
    FROM Departmt, Employee
    WHERE Departmt.Dept_ID = Employee.Dept_ID
</cfquery>

<!--- Use a query of queries to generate a new query with --->
<!--- statistical data for each department. --->
<!--- AVG and SUM calculate statistics. --->
<!--- GROUP BY generates results for each department. --->
<cfquery dbtype = "query" name = "DataTable">
    SELECT Dept_Name,
    AVG(Salary) AS avgSal,
    SUM(Salary) AS sumSal
    FROM GetSalaries
    GROUP BY Dept_Name
</cfquery>

<!--- Reformat the generated numbers to show only thousands. --->
<cfloop index = "i" from = "1" to = "#DataTable.RecordCount#">
    <cfset DataTable.sumSal[i] = Round(DataTable.sumSal[i]/1000)*1000>
    <cfset DataTable.avgSal[i] = Round(DataTable.avgSal[i]/1000)*1000>
</cfloop>

<h1>Employee Salary Analysis</h1>
<!--- Bar graph, from Query of Queries --->
<cfchart format="flash"
    xaxistitle="Department"
    yaxistitle="Salary Average">

<cfchartseries type="bar"
    query="DataTable"
    itemcolumn="Dept_Name"
    valuecolumn="avgSal">

<cfchartdata item="Facilities" value="35000">

</cfchartseries>
</cfchart>
```

cfchartdata

説明

cfchart タグおよび cfchartseries タグとともに使用します。このタグでは、チャートのデータポイントを定義します。データは、cfchartseries タグに送信されます。

カテゴリ

データ出力タグ、拡張タグ

シンタックス

```
<cfchartdata
    item = "text"
    value = "number">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfchart、cfchartseries、『ColdFusion アプリケーションの開発』の Creating Charts and Graphs

ColdFusion MX: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
item	必須		データポイントの名前です。文字列で指定します。
value	必須		データポイントの値です。数値または式で指定します。

例

```
<!--- The following example analyzes the salary data in the cfdocexamples
database and generates a bar chart showing average salary by department. The
body of the cfchartseries tag loops over a cfchartdata tag to include data
available from the query. --->

<!--- Get the raw data from the database. --->
<cfquery name="GetSalaries" datasource="cfdocexamples">
    SELECT Departmt.Dept_Name,
        Employee.Dept_ID,
        Employee.Salary
    FROM Departmt, Employee
    WHERE Departmt.Dept_ID = Employee.Dept_ID
</cfquery>

<!--- Use a query of queries to generate a new query with --->
<!--- statistical data for each department. --->
<!--- AVG and SUM calculate statistics. --->
<!--- GROUP BY generates results for each department. --->
<cfquery dbtype = "query" name = "DataTable">
    SELECT Dept_Name,
    AVG(Salary) AS avgSal,
    SUM(Salary) AS sumSal
    FROM GetSalaries
    GROUP BY Dept_Name
</cfquery>
```

```
<!--- Reformat the generated numbers to show only thousands. --->
<cfloop index = "i" from = "1" to = "#DataTable.RecordCount#">
<cfset DataTable.sumSal[i] = Round(DataTable.sumSal[i]/1000)*1000>
<cfset DataTable.avgSal[i] = Round(DataTable.avgSal[i]/1000)*1000>
</cfloop>

<h1>Employee Salary Analysis</h1>
<!--- Bar graph, from Query of Queries. --->
<cfchart format="flash"
xaxistitle="Department"
yaxistitle="Salary Average">

<cfchartseries type="bar"
itemcolumn="Dept_Name"
valuecolumn="avgSal">

<cfloop query="DataTable">
<cfchartdata item="#DataTable.Dept_Name#" value="#DataTable.avgSal#">
</cfloop>

</cfchartseries>
</cfchart>
```

cfchartseries

説明

cfchart タグとともに使用します。このタグでは、棒グラフ、折れ線グラフ、円グラフなど、データを表示するチャートのスタイルを定義します。

カテゴリ

データ出力タグ、拡張タグ

シンタックス

```
<cfchartseries
    alpha = "Integer between 0 and 1"
    animate = "JSON string representation"
    aspect = "text"
    background = "JSON string representation"
    bevel = "JSON string representation"
    border = "JSON string representation"
    color = "text"
    data = "JSON string representation"
    hovermarker = "JSON string representation"
    marker = "JSON string representation"
    type="type"
    itemColumn ="query column"
    label = "text"
    valueColumn="query column"
    colorlist = "list"
    dataLabelStyle="style"
    markerStyle="style"
    paintStyle="plain|raise|shade|light"
    query="query name"
    scales = "comma-seperated list of axes"
    shadow = "JSON string representation"
    seriesColor="hexadecimal value|web color"
    seriesLabel="label text">
    toolTip = "JSON string representation"
    zColumn = "query column"
    type="type"
    itemColumn="query column"
    valueColumn="query column"
    colorlist = "list"
    dataLabelStyle="style"
    markerStyle="style"
    paintStyle="plain|raise|shade|light"
    query="query name"
    seriesColor="hexadecimal value|web color"
    seriesLabel="label text">
</cfchartseries>
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfchart、cfchartdata、『ColdFusion アプリケーションの開発』の Creating Charts and Graphs

履歴

ColdFusion 10：alpha、animate、aspect、background、bevel、border、marker、color、label、hoverMarker、data、scales、shadow、tooltip、zcolumn の各属性が追加されました。

ColdFusion MX 7:

- dataLabelStyle 属性が追加されました。
- type 属性の horizontalbar 値が追加されました。

ColdFusion MX 6.1: 補間の動作が変更されました。タグは、複数の系列を持つ折れ線グラフのデータポイントを補間するようになりました。

ColdFusion MX: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
alpha	オプション		背景のアルファ（透明度）レベルです。 有効な値は、0（透明）～ 1（不透明）です。
animate	オプション		effect、speed など、アニメーションを定義するためのキーの構造体です。 空の構造体の場合、デフォルトのアニメーションとして appear 効果が設定されます。
aspect	オプション		チャートタイプの変化を定義します（例えば、line、area、レーダーチャートの dots など）。
background	オプション		background に関連する次のキーの構造体です。 • color：背景色を設定します。 • color-1：矢印の最初の背景色を設定します。 • color-2：矢印の 2 番目の背景色（グラデーションで使用）を設定します。 • transparent：下層の色やチャートが表示されるように、背景イメージの透明度を設定します。 • fit：背景の領域に合わせた幅と高さを定義します。 • repeat：イメージの繰り返しタイプを定義します。 • image：背景イメージのパスを定義します。 • position：背景イメージの位置を定義します。
bevel	オプション		bevel に関連する次のキーの構造体です。 • color：ベベルの色を定義します。 • blur-x：x 方向のベベルのエッジの鋭さ／滑らかさを定義します。 • blur-y：y 方向のベベルのエッジの鋭さ／滑らかさを定義します。 • angle：ベベルの角度を定義します。 • distance：# | #px 単位の距離で、ベベルが表示されるオブジェクトからの距離を示します。
border	オプション		border に関連する次のキーの構造体です。 • color：ボーダーの色を設定します。 • radius：丸みがある角の半径を定義します。 • width：ボーダーの幅を定義します。
marker	オプション		size、border、background、bevel など、マーカーのスタイル設定に使用するキーの構造体です。
color	オプション		チャートの主要な要素 (棒グラフの棒など) の色です。 円グラフの場合、これは最初の項目の色です。 16 進数値またはサポートされるカラー名です。名前のリストおよび 6 桁または 8 桁の 16 進数値については、cfchart タグの「使用方法」を参照してください。 16 進数の値を入力するには、"##xxxxxx" または "##xxxxxxxx" という形式を使用します。ここで、x は 0 ～ 9 または A ～ F です。シャープ記号（#）は 2 つ使用するか、または使用しないでください。

属性	必須 / オプション	デフォルト	説明
label	オプション		データ系列のラベルのテキストです。
hoverMarker	オプション		size、border、background、bevel など、マーカーの上にマウスが置かれたときのマーカーのスタイル設定に使用するキーの構造体です。
data	オプション		チャートのデータです。これは、配列の配列です。 次のようにデータを指定します。 2,3],[4,5]] または [[3,4,5],[6,7,3]]（バブルチャートなど、z パラメーターを持つチャートの場合）、または [[3],[4]]（円グラフの場合）
scales	オプション		系列をプロットする座標軸のカンマ区切りリストです（例：x,y2）。
shadow	オプション	false	シャドウを有効または無効にするために使用します。 値には、yes|no、または alpha のキーを持つ構造体を指定できます。
tooltip	オプション		background、font、border など、ツールヒントのスタイル設定に使用するキーの構造体です。
zcolumn	オプション		次元の値です。 チャートが、x と y に加えて 3 つ目の次元を持つ場合に適用できます。
type	必須		チャートの表示スタイルを設定します。 • bar • line • pyramid • area • horizontalbar • cone • curve • cylinder • step • scatter • pie
itemColumn	query 属性を指定している場合は必須		query 属性に指定されたクエリー内の列の名前です。グラフで表すデータポイントの項目ラベルを指定します。
valueColumn	query 属性を指定している場合は必須		query 属性に指定されたクエリー内の列の名前です。グラフで表すデータ値を指定します。
colorlist	オプション		各データポイントの色を設定します。cfchartseries の type 属性が pie、pyramid、area、horizontalbar、cone、cylinder、または step の場合に適用されます。 16 進数値またはサポートされる Web カラー名のカンマ区切りリストです。名前のリストおよび 6 桁または 8 桁の 16 進数値については、cfchart の「使用方法」を参照してください。 16 進数の値を入力するには、"##xxxxxx" または "##xxxxxxxx" という形式を使用します。ここで、x は 0 ～ 9 または A ～ F です。シャープ記号 (#) は 2 つ使用するか、または使用しないでください。

属性	必須 / オプション	デフォルト	説明
dataLabelStyle	オプション	none	系列の項目に色を適用する方法を指定します。 • none: 何も出力しません。 • value: データポイントの値です。 • rowLabel: 行のラベルです。 • columnLabel: 列のラベルです。 • pattern: トータルグラフに対するパーセンテージを示す columnLabel の値など、列ラベル、値、および集計情報の組み合わせです。たとえば、Sales 55,000 20% of 277,000 などです。
markerStyle	オプション	rectangle	2 次元の折れ線グラフ、曲線グラフ、および散布グラフのデータポイントにマーカーとして表示するアイコンを設定します。 • rectangle • triangle • diamond • circle • letter • mcross • snow • rcross
paintStyle	オプション	plain	データ系列のペイント表示スタイルを設定します。 • plain: 1 色の塗りつぶしです。 • raise: ボタン状の外観です。 • shade: 端部が濃くなるグラデーション塗りつぶしです。 • light: 淡い色のグラデーション塗りつぶしです。
query	オプション		グラフ化するデータを取得するための ColdFusion クエリーの名前です。
seriesColor	オプション		チャートの主要な要素 (棒グラフの棒など) の色です。円グラフの場合は、最初の項目の色を示します。 16 進数値またはサポートされるカラー名です。名前のリストおよび 6 桁または 8 桁の 16 進数値については、cfchart タグの「使用方法」を参照してください。 16 進数の値を入力するには、"##xxxxxx" または "##xxxxxxxx" という形式を使用します。ここで、x は 0 ～ 9 または A ～ F です。シャープ記号 (#) は 2 つ使用するか、または使用しないでください。
seriesLabel	オプション		データ系列のラベルのテキストです。

使用方法

クライアントサイドのチャート作成では、次の属性はサポートされません。

paintStyle、および markerstyle の値の letterx、mcross、snow および rcross。

cfchartdata に新しい zvalue 属性が追加されました。チャートが、x と y に加えて 3 つ目の次元を持つ場合に適用できます。

円グラフの場合、円の項目の色は次のように設定されます。

- seriesColor 属性を省略すると、自動的に各項目の色が設定されます。
- seriesColor 属性を指定すると、最初の項目に指定された色が付けられ、順にその後の各項目に自動的に色が付けられます。

制約

クライアントサイドのチャート作成では、次のサーバーサイドのチャート作成機能は使用できません。

- チャートのリンク先 URL の設定
- 変数へのチャートの書き込み

例 1

```
 <!--- The following example analyzes the salary data in the cfdocexamples
database and generates a bar chart showing average salary by department. --->

<!--- Get the raw data from the database. --->
<cfquery name="GetSalaries" datasource="cfdocexamples">
    SELECT Departmt.Dept_Name,
        Employee.Dept_ID,
        Employee.Salary
    FROM Departmt, Employee
    WHERE Departmt.Dept_ID = Employee.Dept_ID
</cfquery>

<!--- Use a query of queries to generate a new query with --->
<!--- statistical data for each department. --->
<!--- AVG and SUM calculate statistics. --->
<!--- GROUP BY generates results for each department. --->
<cfquery dbtype = "query" name = "DataTable">
SELECT
Dept_Name,
AVG(Salary) AS avgSal,
SUM(Salary) AS sumSal
FROM GetSalaries
GROUP BY Dept_Name
</cfquery>

<!--- Reformat the generated numbers to show only thousands. --->
<cfloop index = "i" from = "1" to = "#DataTable.RecordCount#">
<cfset DataTable.sumSal[i] = Round(DataTable.sumSal[i]/1000)*1000>
<cfset DataTable.avgSal[i] = Round(DataTable.avgSal[i]/1000)*1000>
</cfloop>

<h1>Employee Salary Analysis</h1>
<!--- Bar graph, from Query of Queries --->
<cfchart format="flash"
xaxistitle="Department"
yaxistitle="Salary Average">

<cfchartseries type="bar"
query="DataTable"
itemcolumn="Dept_Name"
valuecolumn="avgSal" />
</cfchart>
```

例 2

次に、クライアントサイドのチャート作成の基本的な使用例を示します。

```
<cfchart format ="html" type="pie">
    <cfchartseries>
        <cfchartdata item="New car sales" value="50000">
        <cfchartdata item="Used car sales" value="25000">
        <cfchartdata item="Leasing" value="30000">
        <cfchartdata item="Service" value="40000">
    </cfchartseries>
</cfchart>
```

例 3

この例は、zcolumn を指定して単純なバブルチャートを作成する方法を示しています。

```
<cfchart format ="html" type="bubble" query="ChartQuery" showlegend="false">
<cfchartseries query="ProdQuery" type="bubble" itemcolumn="title" valuecolumn="total_days"
zcolumn="rem_days" label="Total_Days">
</cfchart>
```

例 4

これは、labels を構造体で指定する例です。

```
<cfchart format ="html" type="bubble" que       ry = " ChartQ ue ry " showlegend="false"
labels=#[{"text"="Hello Label","hook":"node:plot=0,index=2,offset-x=-10,offset-y=-90"}]#>
<cfchartseries type="bubble" label="Total_Days">
<cfchartdata item=1 value=10 zvalue=40>
<cfchartdata item=2 value=20 zvalue=30>
<cfchartdata item=3 value=30 zvalue=20>
<cfchartdata item=4 value=20 zvalue=35>
<cfchartdata item=5 value=40 zvalue=10>
</cfchartseries>
</cfchart>
```

cfcol

説明

テーブルの列ヘッダ、幅、配置、およびテキストを定義します。cftable タグ内で使用します。

カテゴリ

データ出力タグ

シンタックス

```
<cfcol
    header = "column header text"
    text = "column text"
    align = "left|right|center"
    width = "number that indicates width of column">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfcontent、cfoutput、cftable、『ColdFusion アプリケーションの開発』の Performing file operations with cfftp

履歴

ColdFusion MX: ダイナミックな cfcol ステートメントを構築する機能が追加されました。

属性

属性	必須 / オプション	デフォルト	説明
header	必須		列ヘッダのテキストです。この属性を使用する場合は、cftable タグの colHeaders 属性も使用する必要があります。
text	必須		二重引用符 (") で区切ったテキストです。表示する内容を決定します。ルールは、cfoutput セクションの場合と同じです。ハイパーリンク、イメージへの参照、入力コントロールなどを埋め込むことができます。
align	オプション	left	列の配置です。 • left • right • center
width	オプション	20	列の幅です。表示されるデータの長さがこの値を超える場合、データは列内に収まるように切り捨てられます。これを避けるには、HTML の table タグを使用します。 前後の cftable タグに htmltable 属性がある場合は、width でテーブル幅のパーセントを指定すると、テキストが切り捨てられません。あるいは、width で文字数を指定します。

使用方法

cftable タグ内には、少なくとも 1 つの cfcol タグを指定する必要があります。また、cfcol タグおよび cftable タグはページ内で隣接させて配置する必要があります。cftable タグ内にネストできるのは、cfcol タグのみです。cftable タグはネストできません。

cfcol の header で指定したテキストを表示するには、cfcol の header 属性と cftable の colHeader 属性を指定します。いずれかの属性を一方だけ指定しても、ヘッダは表示されません。ここではエラーは発生しません。

例

```
<!--- This example shows the use of cfcol and cftable to align
    information returned from a query. --->
<!--- Query selects information from cfdocexamples data source. --->
<cfquery name = "GetEmployees" dataSource = "cfdocexamples">
    SELECT Emp_ID, FirstName, LastName, EMail, Phone, Department
    FROM Employees
</cfquery>
<html>
<body>
<h3>cfcol Example</h3>
<!--- Uses the HTMLTable attribute to display cftable as an HTML
    table, rather than PRE formatted information --->
<cftable
    query = "GetEmployees"
    startRow = "1" colSpacing = "3"
    HTMLTable colheaders>
<!--- Each cfcol tag sets the width of a column in the table,
    the header information, and the text/CFML for the cell. --->
    <cfcol header = "<b>ID</b>"
        align = "Left"
        width = 2
        text= "#Emp_ID#">
    <cfcol header = "<b>Name/Email</b>"
        align = "Left"
        width = 15
        text= "<a href = 'mailto:#Email#'>#FirstName# #LastName#</A>">
    <cfcol header = "<b>Phone Number</b>"
        align = "Center"
        width = 15
        text= "#Phone#">
</cftable>
```

cfcollection

説明

Solr 検索エンジンのコレクションを作成および管理します。

カテゴリ

拡張タグ

シンタックス

```
cfcollection supports script style syntax:

new collection().CREATE(collection="<collection_name>", engine="solr", path="<path to the solr
directory>");


<cfcollection
    action = "action"
    categories = "yes|no"
    collection = "collection name"
    engine = "solr
    language = "language"
    name = "query name"
    path = "c">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfexecute、cfindex、cfobject、cfreport、cfsearch、cfwddx

履歴

ColdFusion 9: engine 属性が追加されました (Solr のサポートのため)。

ColdFusion MX 7:

- ColdFusion MX 7 以降では、cfcollection タグを使用して既存のコレクションのエイリアス名を作成できません。Verity はすべてのコレクション情報を保持するので、2 つの名前が同じコレクションをポイントすることはできません。
- 外部コレクションへの参照が削除されました。
- action 属性の map オプションと repair オプションは非推奨になりました。これらの属性は今後のリリースでは機能せず、エラーが発生する可能性があります。
- categories 属性と categorylist アクションが追加されました。
- CATEGORIES、SIZE、DOCCOUNT、および LASTMODIFIED が、list アクションによって返される変数のリストに追加されました。
- list アクションによって返される MAPPED、ONLINE、および REGISTERED 変数は廃止になりました。

ColdFusion MX:

- action 属性の必要条件が変更され、指定が必須になりました。
- action 属性の list 値が追加されました。この値がデフォルト値です。
- action 属性の値 map の必要条件が変更され、action 属性の値 map を指定する必要がなくなりました。ColdFusion により、コレクションが検出され、必要に応じてコレクションがマッピングされます。
- コレクションのネーミング規則が変更され、スペースを含むコレクション名が使用できるようになりました。
- Verity オペレーションの動作が変更されました。ColdFusion で、Acrobat PDF ファイルに対する Verity オペレーションがサポートされるようになりました。
- 返す例外が変更されました。このタグでは SEARCHENGINE 例外を返すことができます。

属性

属性	必須 / オプション	デフォルト	説明
action	必須。「使用方法」を参照	list	• categorylist：コレクションからカテゴリを取得し、各カテゴリ内のドキュメント数を示します。構造体の構造体を返します。そこでは、各サブ構造体を表すカテゴリが一連のドキュメントに関連付けられています。カテゴリツリー内のカテゴリの場合、ドキュメントの数はそのツリー内の当該レベル以下のドキュメントの数です。 • create: ColdFusion でコレクションを登録します。コレクションが存在する場合は、コレクションのマッピングを作成します。コレクションが存在しない場合は、コレクションを作成します。 • delete: コレクションを登録解除して、コレクションのディレクトリを削除します。 • list: ColdFusion によって登録されているコレクション群の属性のクエリー結果セットを返します (この結果セットには、name 属性値の名前が付けられます)。Solr コレクションを使用しており、engine 属性を省略した場合は、両方のタイプのコレクションの情報が一覧表示されます。 • map: コレクションのマッピングを作成します。アクションが create で、コレクションが既に存在している場合、ColdFusion ではコレクションのマッピングも作成されます。 • optimize: 検索のためにコレクションの構造体と内容を最適化し、それによってスペースを確保します。コレクションがオフラインになり、検索とインデックス作成ができないようになります。 • repair: 非推奨になりました。何も設定されません。
categories	「使用方法」を参照	no	コレクション作成のためにのみ使用できます。 • yes: このコレクションはカテゴリをサポートします。 • no: このコレクションはカテゴリをサポートしません。
collection	「使用方法」を参照		• コレクション名です。名前にはスペースを含めることができます。
engine	オプション	Solr	コレクションの検索エンジンです。 • solr: Apache Lucene オープンソース検索エンジンです。 create アクションおよび map アクションでのデフォルトは Solr です。list アクションでは、デフォルトで Solr のすべてのコレクションが一覧表示されます。他のすべてのアクションでは、ColdFusion によってコレクションタイプが決定されます。
language	「使用方法」を参照	English	オプションのリストについては、「使用方法」を参照してください。
name	「使用方法」を参照		list アクションおよび categorylist アクションによって返されるクエリー結果の名前です。
path	「使用方法」を参照		コレクションの絶対パスです。 既存のコレクションをマッピングするには、コレクションの完全修飾パスを指定します (コレクション名は含めない)。たとえば、"C:¥MyCollections¥" と指定します。

使用方法

このタグを使用すると、次の操作を実行できます。

- Solr コレクションの作成
- このタグまたは ColdFusion Administrator によって作成された Solr コレクションの管理

次の表に、このタグの属性値の依存関係を示します。

各属性の指定要件 (必須、オプション、または不要 (空白))	action 属性の値						
	list	**create**	**map**	**optimize**	**repair**	**delete**	**category list**
collection		必須	必須	必須	必須	必須	必須
path		必須	必須				
language		オプション	オプション				
name	必須						必須
categories							

次の例では、categorylist アクションによって返される構造体を示します。

CATEGORIES	
blue	10
green	3
magenta	3
purple	2
CATEGORYTREES	
a/	10
a/b	10
a/b/c	10
a/b/c/subdir	3

list アクションは、1 つのコレクションにつき 1 行を含む結果セットを返します。この情報を次に示します。

列	内容
CATEGORIES	• yes: このコレクションではカテゴリのサポートが有効です。 • no: このコレクションではカテゴリのサポートが無効です。
CHARSET	コレクションの文字セットです。
CREATED	コレクションが作成された日付と時刻です。
DOCCOUNT	このコレクション内のドキュメントの数です。
EXTERNAL	• yes: コレクションは外部です。 • no: コレクションは外部ではありません。 • not found: コレクションは登録されていますが、定義されたパスで取得できません。
LANGUAGE	コレクションのロケール設定です。 この情報は K2 サーバーコレクションには使用できません。
LASTMODIFIED	コレクションが最終変更された日付と時刻です。
MAPPED	廃止

列	内容
NAME	コレクションの名前
ONLINE	廃止
PATH	コレクションの絶対パスです。
REGISTERED	廃止
SIZE	コレクションのサイズです。単位はキロバイトです。

uni を指定して複数の言語のサポートを有効にすることもできます。

コレクションが存在するかどうか判断するには、次のようなコードを使用して、クエリーオブクエリーを行います。

```
 <cfcollection action="list" name="myCollections" >
<cfquery name="qoq" dbtype="query">
    SELECT * from myCollections
    WHERE myCollections.name = 'myCollectionName'
    </cfquery>
<cfif qoq.recordcount GT 0>
    <!--- Collection exists --->
    <cfdump var = #qoq#>
</cfif>
```

Solr コレクションが存在するかどうかを確認するには、engine 属性追加し、その値として solr を指定する必要があります。次に例を示します。

<cfcollection action="list" name="myCollections" engine="solr">

検索サーバーで登録されているすべてのコレクションの値とともに結果セットを取得するには、次のようなコードを使用します。

```
 <cfcollection action="list" name="myCollections">
<cfoutput query="myCollections">
    #name#<br>
</cfoutput>
```

コレクションに内容を追加するには、cfindex を使用します。コレクションを検索するには、cfsearch を使用します。

この変更を有効にするには、ColdFusion 検索サービスを再起動する必要があります。

Solr コレクションの場合、このタグの language 属性で次のオプションがサポートされています。

Brazilian	cjk (中国語、日本語、韓国語)	French	Russian
Czech	Dutch	German	Thai
Chinese	English	Greek	

cfcomponent

説明

コンポーネントオブジェクトを作成および定義します。また、CFML で構築し、cffunction タグ内に囲んだ機能を囲みます。このタグ内には、メソッドを定義している cffunction タグを指定できます。このタグの本文内に指定された cffunction タグ以外のコードは、コンポーネントのインスタンスを作成するときに実行されます。

コンポーネントファイルには CFC という拡張子が付けられ、アプリケーションの任意のディレクトリに保管されます。

コンポーネントメソッドは、次の方法で呼び出されます。

- ColdFusion ページの cfinvoke タグ内で
- CFC ファイルを呼び出す URL 内でメソッド名を URL パラメータとして渡す
- cfscript タグ内で
- Web サービスとして
- Flash コードから

カテゴリ

拡張タグ

シンタックス

```
<cfcomponent
    accessors = "yes|no"
    autoIndex = "yes|no"
    alias = "ActionScript 3 type alias"
    bindingname = "binding element name"
    displayname = "string"
    extends = "component name"
    hint = "string"
    implements = "ColdFusion interface"
    indexable = "yes|no"
    indexLanguage = "language"
    mappedSuperClass = "yes|no"
    namespace = "default service namespace"
    output = "no value|no|yes"
    porttypename = "port type element name"
    Serializable = "yes|no"
    serviceaddress = "service URL"
    serviceportname = "port element name"
    style = "rpc|document|wrapped"
    wsversion = "1|2"
    wsdlfile = "path">
    variable declarations
    <cffunction ...>
        ...
    </cffunction>

    <cffunction ...>
        ...
    </cffunction>
</cfcomponent>
```

関連項目

cfargument、cffunction、cfinterface、cfinvoke、cfinvokeargument、cfobject、cfproperty、cfreturn、IsInstanceOf、
『ColdFusion アプリケーションの開発』の Building and Using ColdFusion Components

履歴

ColdFusion 10：rest、restPath、httpMethod、produces、consumes、indexable、indexLanguage、autoIndex、wsVersion

ColdFusion 9.0.1：mappedSuperClass 属性が追加されました。

ColdFusion 9: serializable 属性と accessors 属性が追加されました。

ColdFusion 8:

- implements 属性と serviceaddress 属性が追加されました。

- onMissingMethod 関数がサポートされるようになりました。

ColdFusion MX 7:

- document-literal スタイルの Web サービスを発行できるようになりました。
- style、namespace、serviceportname、porttypename、wsdlfile、bindingname、output の各属性が追加されました。
- document-literal スタイルの Web サービス公開時の hint 属性と displayname 属性の機能が拡張されました。

ColdFusion MX: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
accessors	オプション	yes (永続 CFC の場合) no (その他の場合)	yes に設定すると、暗黙の getter および setter を呼び出すことができます。 永続 CFC の場合、accessors は常に有効です。
alias	オプション		オブジェクトが CFML から ActionScript 3 に変換されるときに設定される type ラベルを指定します。これは AS3 タイプの alias 属性に一致します。この属性は、Flash Remoting と LiveCycle Data Services の値オブジェクトにのみ適用され、ColdFusion と Flash の両方でタイプ付きオブジェクトを操作できるようにします。
autoIndex	オプション	yes	no の場合、CFC の自動インデックス作成は実行されません。つまり、インデックスの作成はオフラインモードでのみ行います。
bindingname	オプション		WSDL の port 要素の binding 属性を指定します。この属性を指定しない場合、ColdFusion はこの値を CFC クラス名から取得します。
consumes	オプション	*/*	受け入れ可能な MIME タイプのカンマ区切りリストです（例：consumes="text/plain,text/html"）。 HTTP リクエストの Content-Type ヘッダーを検索します。例えば、値が HTML や XML である場合に、ColdFusion によって、リクエストの処理とメソッドの呼び出しが可能な適切なメソッドが検索されます。 リソースが受け入れ可能（つまり使用可能）なクライアントからの MIME メディアタイプの表現を指定するために使用します。 この属性をコンポーネントレベルで使用した場合（かつ関数レベルで使用されていない場合）、デフォルトでは、指定された MIME タイプはすべての応答メソッドで受け入れられます。 クライアントリクエストの MIME タイプを使用可能な関数が CFC に存在しない場合、「415 Unsupported Media Type」のエラーが発生します。 値を指定しない場合、デフォルトで「*/*」が設定されます（これは、すべての MIME タイプが使用されることを意味します）。
displayname	オプション		イントロスペクションを使用して CFC についての情報を示すときに表示する文字列です。この情報は、コンポーネント名に続けて見出しの上に表示されます。
extends	オプション	WEB-INF.cftags.component	メソッドとプロパティを継承する親コンポーネントの名前です。キーワード component を使用して、デフォルト値を指定できます。
hint	オプション		イントロスペクションを使用して CFC についての情報を示すときに表示するテキストです。hint 属性の値は、コンポーネント名の見出しの下に表示されます。この属性を使用して、パラメータの目的を説明します。

属性	必須 / オプション	デフォルト	説明
httpMethod	オプション		使用する HTTP メソッドです。次のいずれかを指定する必要があります。 • GET：サーバーに情報をリクエストします。リクエストされた情報をサーバーが識別するために必要なデータはすべて、URL 内か cfhttpparam type="URL" タグ内に含まれている必要があります。 • POST：処理する情報をサーバーに送信します。cfhttpparam タグが必要です。フォームに似たデータを送信するときによく使われます。 • PUT：指定の URL にメッセージ本文を保管するようサーバーにリクエストします。このメソッドは、サーバーにファイルを送信するときに使われます。 • DELETE：指定の URL を削除するようサーバーにリクエストします。 • HEAD：GET メソッドと同じですが、サーバーはレスポンス内にメッセージ本文を送信しません。このメソッドは、ハイパーテキストリンクの有効性やアクセシビリティのテスト、ドキュメントのタイプや修正時刻の確認、またはサーバーのタイプの判別に使用します。HEAD を指定していない場合、デフォルトで GET メソッドが呼び出されます。ただし、レスポンス内にメッセージ本文は送信されません。 • OPTIONS：サーバーまたは指定の URL で使用できる通信オプションについての情報をリクエストします。このメソッドを使用すると、ColdFusion アプリケーションは、追加のサーバー活動を要求しなくても、URL に関連付けられているオプションや要件、あるいはサーバーの機能を判別することができます。OPTIONS を指定していない場合は、ColdFusion によってレスポンスが送信されます。 リクエストメソッドに基づいて、対応する関数が呼び出されます。例えば、サーバーに GET リクエストが送信された場合、httpMethod が GET の関数が呼び出されます。 関数レベルでこの値が指定されていない場合は、ここで定義した値が使用されます。そのため、httpMethod が指定されていないすべての関数に、この値が割り当てられます。
implements	オプション		このコンポーネントが実装する ColdFusion インターフェイスの名前です。コンポーネントでインターフェイスを実装する場合は、インターフェイスのすべての関数を定義する必要があり、関数定義はインターフェイスで指定されている定義に準拠している必要があります。詳細については、cfinterface を参照してください。 コンポーネントでは、任意の数のインターフェイスを実装できます。複数のインターフェイスを指定するには、interface1,interface2 という形式のカンマ区切りリストを使用します。
indexable	オプション	no	yes の場合、コンポーネントでインデックスの作成が可能になります。
indexLanguage	オプション	english	インデックスおよび検索に使用する言語を指定します。 設定した値は、Application.cfc で定義された値よりも優先されます。
mappedSuperClass	オプション	no	永続的でない CFC で yes に設定すると、子の CFC はプロパティを継承できます。たとえば、基本 CFC を一般的なプロパティの ID、version、createdOn などで定義しておき、その他のすべての永続 CFC でこれらのプロパティを展開していけば、共通する 1 つの動作を取るようにできます。 永続 CFC では、mappedSuperClass を yes に設定できません。

属性	必須 / オプション	デフォルト	説明
namespace	オプション	クラス名	Web サービスとして呼び出される CFC 用の WSDL で使われる名前空間を指定します。この属性を指定しない場合、ColdFusion はこの値を CFC クラス名から取得します。
output	オプション	標準 CFML として処理されるコンポーネント本体の表示可能テキスト	コンポーネント内のコンストラクタコードが HTML 出力を生成できるかどうかを指定します。そのコンポーネント内にある cffunction タグ本文の中の出力には影響しません。 • yes: コンストラクタコードは cfoutput タグ内にある場合と同様に処理されます。シャープ記号 (#) で囲まれた変数名は、自動的に実際の値に置き換えられます。 • no: コンストラクタコードは cfsilent タグ内にある場合と同様に処理されます。 • この属性を指定しない場合、コンストラクタコードは標準 CFML として処理されます。すべての変数は cfoutput タグ内に置く必要があります。
porttypename	オプション		WSDL の porttype 要素の name 属性を指定します。この属性を指定しない場合、ColdFusion はこの値を CFC クラス名から取得します。
produces	オプション	*/*	受け入れ可能な MIME タイプのカンマ区切りリストです（例：produces="text/plain,text/html"）。 HTTP リクエストの Accept ヘッダーまたは URL の拡張子（有効な拡張子は .xml および .json）を検索します。例えば、値が HTML や JSON、XML である場合に、ColdFusion によって、リクエストの処理とメソッドの呼び出しが可能な適切なメソッドが検索されます。 CFC が生成してクライアントに返信できる MIME メディアタイプまたは表現を指定するために使用します。 コンポーネントレベルで指定した場合（かつ関数レベルで指定されていない場合）、デフォルトでは、指定された MIME タイプは CFC のすべての関数が生成できます。 クライアントリクエストの MIME タイプを生成可能なメソッドが CFC に存在しない場合、「406 Not Acceptable」のエラーが発生します。 値を指定しない場合、デフォルトで「*/*」が設定されます（これは、すべての MIME タイプが生成されることを意味します）。
rest	オプション	RestPath が指定された場合は true	true の場合、CFC は REST 対応です。

属性	必須 / オプション	デフォルト	説明
restPath	オプション		REST パスまたは CFC パスを指定して、REST リソースにアクセスすることができます。そのため、CFC パス以外のパスを使用するかどうかを指定します。rest="true" で、この属性を指定しない場合は、CFC へのパスが使用されます。 CFC へのパスは、REST サービスとして登録されているディレクトリから始まるものである必要があります。また、パス内に CFC 名を含める必要があります。 例えば、college というフォルダーが RestTest として登録されており、パブリッシュする student.cfc が（college フォルダー内の）department サブフォルダーに存在する場合、student.cfc へのアクセスに使用する URL は次のようになります。 http://localhost:8500/rest/RestTest/department/student この場合の restPath は、department/student です。 パスの大文字と小文字は区別されます。また、パス内には特殊文字を使用しないことが望ましいです。 CFC レベルでは、次のようにパスを指定します。 restPath="restService" または restPath="test/restService" restPath の値としてテンプレートを指定できます。例えば、restPath="{name}" のようにします。この場合、< ベース *URL*>/rest/<service_mapping>/< スラッシュなしの文字列 > の形式のすべての URL が、この restPath に一致します。例えば、http://localhost:8500/rest/service1/abc などです。 テンプレートの代わりに値にアクセスするには、restargsource の値 path を使用します。詳しくは、cfargument の属性の詳細を参照してください。 この属性では、複数要素の表現を値として指定できます。例えば、restpath="customers/{firstname}-{lastname}" と指定できます。これは、"/customers/paul-bensen" などの URL に一致しますが、"/customers/paulbensen" には一致しません。 また、restPath には正規表現を含めることもできます。例えば、restpath="/customers/{id : \d+}" と指定できます。この場合、id は整数で、"/customers/123111" に一致しますが、"/customers/asd" や "/customers/123/122" には一致しません。 異なるパス属性値を持つ 2 つのメソッドがあり、これらのパスが明確でない場合は、一致を検索するための優先ルールがあります。例えば、restpath="/customers/{id : .+}" の場合を考えます。この表現は、/customer の後に続くあらゆる文字に一致します。これは、"/customers/123"、"/customers/asd" および "/customers/123/12/asdfa" に一致しますが、"/customers" には一致しません。 また、restPath="/customers/{id : .+}/address" などの表現も使用できます。"/customers/123/asd/address" という URL は、両方の URL に一致します。このようなケースでは、一致の検索に優先ルールが使用されます。
serializable	オプション	true	このコンポーネントがシリアル化可能かどうかを指定します。この値を false に設定した場合、コンポーネント、およびコンポーネントの This スコープと Variables スコープ内のデータはシリアル化できないため、これらはセッションレプリケーションでは保持されず、コンポーネントはデフォルトの状態に戻ります。

属性	必須 / オプション	デフォルト	説明
serviceaddress	オプション	CFC の URL	Web サービスの SOAP URL を指定します。この属性を指定しない場合、ColdFusion は WSDL のサービス記述にある CFC の URL を使用します。この属性を使用してプロトコルを指定するには、URL の前に https:// などを付けます。 この属性は、Web サービスのみに適用されます。
serviceportname	オプション		WSDL の port 要素の name 属性を指定します。この属性を指定しない場合、ColdFusion はこの値を CFC クラス名から取得します。
style	オプション	rpc	Web サービスに使われる CFC で RPC-encoded スタイルまたは document-literal スタイルを使用するかどうかを指定します。 • rpc: RPC-encoded スタイルを使用します。 • document: document-literal スタイルを使用します。 • wrapped：wsVersion を 2 に設定した場合、デフォルト値は wrapped、wsVersion が 1 の場合、デフォルト値は rpc です。
wsdlfile	オプション		ColdFusion で生成された WSDL の代わりに使用する、正しい形式の WSDL ファイルです。
wsVersion	オプション		2 を指定した場合、CFC のデプロイには Axis 2 エンジンが使用されます。指定した値は、アプリケーションレベルまたはサーバーレベルで指定した値よりも優先されます。

使用方法

extends 属性を指定した場合は、親コンポーネントのデータとメソッドを、現在のコンポーネントの一部であるかのように CFC メソッドに使用できます。たとえば managerCFC コンポーネントが employeeCFC コンポーネントを拡張していて、employeeCFC コンポーネントが getEmployeeName メソッドを備えている場合は、次のようにして managerCFC からこのメソッドを呼び出すことができます。

```
<cfinvoke component="managerCFC" method="getEmployeeName" returnVariable="managerName"
    EmployeeID=#EmpID#>
```

このタグには終了タグが必要です。

style="document" を指定した場合、ColdFusion は CFC を document-literal スタイルの Web サービスとして公開します。詳細については、『ColdFusion アプリケーションの開発』の Publishing document-literal style web services を参照してください。

CFC は onMissingMethod 関数をサポートしています。CFC で実装されていないメソッドの呼び出しを処理するには、CFC の cfcomponent タグの本文で onMissingMethod 関数を定義します。アプリケーションが CFC で定義されていない関数を呼び出した場合、ColdFusion は onMissingMethod 関数を呼び出して、リクエストされたメソッドの名前と引数を渡します。onMissingMethod 関数を定義していない場合、CFC で定義されていないメソッドが呼び出されると、ColdFusion はエラーを返します。このエラーは呼び出し元のコードで処理する必要があります。

onMissingMethod 関数は次のような目的に使用できます。

- コンポーネントを呼び出すコードの各インスタンスでエラーを処理する代わりに、コンポーネント内でエラーを直接処理する。
- ダイナミックプロキシを作成する (ダイナミックプロキシとは、任意の呼び出しを受け取って正しいアクションをダイナミックに決定するオブジェクトです)。

onMissingMethod 関数は次の形式で指定する必要があります。

```
<cffunction name="onMissingMethod">
    <cfargument name="missingMethodName" type="string">
    <cfargument name="missingMethodArguments" type="struct">
        code to handle call to nonexistent method
</cffunction>
```

注意： onMissingMethod の引数名は変更しないでください。

例 1

```
<cfcomponent>
    <cffunction name="getEmp">
         <cfquery name="empQuery" datasource="cfdocexamples" >
            SELECT FIRSTNAME, LASTNAME, EMAIL
            FROM tblEmployees
         </cfquery>
         <cfreturn empQuery>
    </cffunction>

    <cffunction name="getDept">
        <cfquery name="deptQuery" datasource="cfdocexamples" >
            SELECT *
            FROM tblDepartments
         </cfquery>
         <cfreturn deptQuery>
    </cffunction>
</cfcomponent>
```

cfcontent

説明

次のいずれか、または両方を行います。

- 現在のページの MIME コンテンツエンコードヘッダを設定します。エンコード情報に文字エンコードが含まれている場合は、生成された出力の文字エンコードを設定します。
- ファイルのコンテンツまたはバイナリデータを含む変数のコンテンツを、ページ出力として送信します。

このタグの使用を制限するには、[ColdFusion Administrator]-[セキュリティ]-[サンドボックスセキュリティ] の設定を使用します。詳細については、Administrator のオンラインヘルプを参照してください。

カテゴリ

データ出力タグ

シンタックス

```
<cfcontent
    deleteFile = "yes|no"
    file = "filename"
    reset = "yes|no"
    type = "file type"
    variable = "variable name">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfcol、cfheader、cfhttp、cfoutput、cftable

履歴

ColdFusion 8: type 属性が指定されずに file 属性が指定されている場合のタグの動作が変更されました。以前は、デフォルトのファイルタイプ text/html が使用されていました。現在は、ファイルからコンテンツタイプを取得するように試みます。

ColdFusion MX 7: variable 属性が追加されました。

属性

属性	必須 / オプション	デフォルト	説明
deleteFile	オプション	no	file 属性でファイルを指定する場合にのみ適用されます。 • yes: ファイルのコンテンツをクライアントに送信した後で、サーバー上のファイルを削除します。 • no: サーバー上にファイルを残します。
file	オプション		ページ出力を提供するコンテンツを含むディスク上またはメモリ内のファイルの名前です。ファイル名は、ドライブ文字とコロン、またはスラッシュや円記号で始まらなければなりません。 ColdFusion を分散構成で使用する場合、file 属性には、Web サーバーが稼動しているシステムのパスを指定する必要があります。この属性を使用すると、現在の CFML ページのその他の出力は無視されます。指定したファイルのコンテンツのみがクライアントに送信されます。

属性	必須 / オプション	デフォルト	説明
reset	オプション	yes	file 属性または variable 属性を指定した場合、この属性は無効になります。指定しない場合は次のいずれかになります。 • yes: cfcontent を呼び出す前の出力が破棄されます。 • no: cfcontent を呼び出す前の出力が保存されます。この場合は、すべての出力が指定のコンテンツタイプで送信されます。

type	オプション		ページの MIME コンテンツタイプです。その後にセミコロンと文字エンコードを続けることもあります。デフォルトでは、ページは UTF-8 文字エンコードの text/html コンテンツタイプとして送信されます。ただし、file 属性が指定されている場合は、ファイルからコンテンツタイプを取得するように試みます。
			コンテンツタイプにより、ブラウザまたはクライアントがそのページコンテンツをどのように解釈するかが決まります。
			使用できるコンテンツタイプの値の一部を次に示します。
			• text/html
			• text/plain
			• application/x-shockwave-flash
			• application/msword
			• image/jpeg
			一般的に使用される文字エンコードの値を次に示します。
			• utf-8
			• iso-8859-1
			• windows-1252
			• us-ascii
			• shift_jis
			• iso-2022-jp
			• euc-jp
			• euc-kr
			• big5
			• euc-cn
			• utf-16
			たとえば、次のようになります。
			`type = "text/html"`
			`type = "text/html; charset=ISO-8859-1"`

variable	オプション		cfchart タグで生成されたチャートのコンテンツ、cffile action="readBinary" タグで取得された PDF ファイルや Excel ファイルなど、ブラウザで表示可能なコンテンツを持つ ColdFusion バイナリ変数の名前です。この属性を使用すると、現在の CFML ページのその他の出力は無視されます。指定したファイルのコンテンツのみがクライアントに送信されます。

使用方法

HTML ページなど、生成された出力の文字エンコード (文字セット) を設定するには、次のようなコードを使用します。

```
<cfcontent type="text/html; charset=ISO-8859-1">
```

ColdFusion では、HTTP リクエストの処理時に、HTTP レスポンスで返されるデータに対して使用する文字エンコードが決定されます。デフォルトでは、ページ内の HTML の meta タグの値に関係なく、Unicode UTF-8 形式を使用して文字データが返されます。cfcontent タグを使用すると、レスポンスのデフォルトの文字エンコードよりも優先することができます。たとえば、ColdFusion で Japanese EUC 文字エンコードを使用してページを返すには、次のように type 属性を使用します。

```
<cfcontent type="text/html; charset=EUC-JP">
```

cfcontent タグをカスタムタグから呼び出す場合で、現在のページが別のアプリケーションまたはカスタムタグから呼び出されたときに、ページがこのタグによって破棄されないようにするには、reset = "no" に設定します。

ファイル削除オペレーションが正常に終了しない場合は、エラーが発生します。

このタグをページ内の cfflush タグの後に使用しないでください。そのように使用しても効果はなく、ColdFusion でエラーが発生します。

次のタグを使用すると、大部分のブラウザでは、cfcontent タグで指定されたファイルのコンテンツを、filename 属性で指定されたファイル名で保存するかどうかをユーザーに尋ねるダイアログボックスが表示されます。ファイルを開くことを選択すると、大部分のブラウザは、ブラウザウィンドウではなく関連付けられたアプリケーションでファイルを開きます。

```
<cfheader name="Content-Disposition" value="attachment; filename=filename.ext">
```

PDF ドキュメントなどの一部のファイルタイプの場合は、実行可能コードを使用せず、大部分のブラウザに直接表示することができます。ブラウザにファイルを直接表示するようにするには、次のように cfheader タグを使用します。

```
<cfheader name="Content-Disposition" value="inline; filename=name.ext">
```

filename 属性の **filename** の部分には任意の値を使用できますが、**ext** の部分はそのファイルタイプに関する Windows の標準の拡張子でなければなりません。

Microsoft Excel ドキュメントなど、実行可能コードが含まれている可能性があるファイルタイプの場合、大部分のブラウザでは、ドキュメントを開く前に確認メッセージが表示されます。これらのファイルタイプの場合、ユーザーがファイルを開くことを選択すると、インラインコンテンツ処理の仕様により、ブラウザに対してファイルを直接表示するようにリクエストされます。

文字エンコードの詳細については、次の Web ページを参照してください。

- www.w3.org/International/O-charset.html のページには、文字エンコードと Web についての全般的な情報、および役に立つリンクがいくつか掲載されています。
- www.iana.org/assignments/character-sets のページには、インターネットで使用され、Internet Assigned Numbers Authority によって管理されている文字セット名の完全なリストが掲載されています。
- ColdFusion では、エンコードのサポートに Java JCE が使用されています。http://java.sun.com/javase/6/docs/technotes/guides/intl/encoding.doc.html のページには、JCE 6 (したがって ColdFusion) が解釈できる文字エンコードのリストが掲載されています。このリストでは、SetEncoding charset パラメータやその他の ColdFusion 属性やパラメータで使われる IANA の文字エンコード名ではなく、Java の内部名が使用されています。

インターネット上で使用されるメディアタイプの完全なリストについては、www.iana.org/assignments/media-types/ を参照してください。

注意： cfabort、cflocation または cfcontent タグを使用した場合は、OnRequestEnd ではなく OnAbort メソッドが呼び出されます。

例

```
 <!--- CFCONTENT Example 1
This example shows the use of cfcontent to return the contents of the CF
Documentation page dynamically to the browser. You might need to change the
path and/or drive letter depending on how ColdFusion is installed on your
system. Notice that the graphics do not display and the hyperlinks do not work,
because the html page uses relative filename references.
The root of the reference is the ColdFusion page, not the location of the
html page. --->

<cfcontent type = "text/html"
    file = "C:\ColdFusion9\wwwroot\cfdocs\dochome.htm"
    deleteFile = "no">

<!--- CFCONTENT Example 2
This example shows how the Reset attribute changes text output. Notice how the
first text section ("This example shows how the Reset attribute changes output
for text reset = "Yes":123) does NOT print out to the screen. --->

<p>This example shows how the Reset attribute changes output for text.</p>
<p>reset = "Yes": 123 <BR> <cfcontent type = "text/html" reset = "Yes">456</p>
<p>This example shows how the Reset attribute changes output for text.</p>
<p>reset = "No": 123 <BR> <cfcontent type = "text/html" reset = "No">456</p>
<!--- CFCONTENT Example 3
This example triggers a download of an Excel file. The user is prompted with an option to save the file or
open it in the browser. --->

<cfheader name="Content-Disposition" value="inline; filename=acmesales03.xls">
    <cfcontent type="application/vnd.ms-excel" file="c:\temp\acmesales03.xls">

<!--- CFCONTENT Example 4
This example triggers a download of a Word document then deletes the original from the "temp" directory.
The user is prompted with an option to save the file or open it in the browser. --->

<cfheader name="Content-Disposition" value="inline; filename=temp.doc">
<cfcontent type="application/msword" file="c:\temp\Cable.doc" deletefile="yes">

<!--- CFCONTENT Example 5
This example causes the browser to treat the HTML table as Excel data.
Excel interprets the table format.
Because Excel can include executable code, the browser prompts the user whether
to save the file or open it in a browser. --->

<cfheader name="Content-Disposition" value="inline; filename=acmesalesQ1.xls">
<cfcontent type="application/vnd.msexcel">

<table border="2">
<tr><td>Month</td><td>Quantity</td><td>$ Sales</td></tr>
<tr><td>January</td><td>80</td><td >$245</td></tr>
<tr><td>February</td><td>100</td><td>$699</td></tr>
<tr><td>March</td><td>230</td><td >$2036</td></tr>
<tr><td>Total</td><td>=Sum(B2..B4)</td><td>=Sum(C2..C4)</td></tr>
</table>
```

cfcontinue

説明

cfloop タグ内で使用します。処理がループの先頭に戻ります。

カテゴリ

フロー制御タグ

シンタックス

```
<cfcontinue>
```

関連項目

cfabort、cfbreak、cfexecute、cfif、cflocation、cfloop、cfthrow、cftry、『ColdFusion アプリケーションの開発』の cfloop and cfbreak

履歴

ColdFusion 9: タグが追加されました。

例

```
<!--- This shows the use of to return processing to the top of a loop when a condition is met.--->
<!--- Select courses; use cfloop to find a condition; then break the loop. --->
<!--- Check that number is numeric. --->
<cfif IsDefined("form.course_number")>
    <cfif Not IsNumeric(form.course_number)>
        <cfabort>
    </cfif>
</cfif>
<cfquery name="GetCourses" datasource="cfdocexamples">
    SELECT *
    FROM Courses
    ORDER by course_number
</cfquery>

<p> This example uses CFLOOP to cycle through a query to find a value.
(In our example, a list of values corresponding to courses in the cfdocexamples
datasource). When the conditions of the query are met, CFBREAK stops the loop. </p>
<p> Please enter a Course Number, and hit the "submit" button: </p>
<form action="cfbreak.cfm" method="POST">
    <select name="courseNum">
        <cfoutput query="GetCourses">
            <option value="#course_number#">#course_number#
        </cfoutput>
    </select>
    <input type="Submit" name="" value="Search on my Number">
</form>
<!--- If the courseNum variable is not defined, don't loop through the query.--->
<cfif IsDefined ("form.courseNum") IS "True">
<!--- Loop through query until value found, then use CFBREAK to exit query.--->
    <cfloop query="GetCourses">
        <cfif GetCourses.course_number IS form.courseNum>
            <cfoutput>
                <h4>Your Desired Course was found:</h4>
                <pre>#course_number# #descript#</pre>
            </cfoutput>
            <cfbreak>
        <cfelse>
            <br> Searching...
        </cfif>
    </cfloop>
</cfif>
```

cfcookie

説明
有効期限、セキュリティオプションなどの Web ブラウザの Cookie 変数を定義します。

カテゴリ
フォームタグ、変数操作タグ

シンタックス
```
<cfcookie
    name = "cookie name"
    domain = ".domain"
    expires = "period"
    httponly = "yes|no"
    path = "URL"
    secure = "yes|no"
    value = "text">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目
cfdump、cfparam、cfregistry、cfsavecontent、cfschedule、cfset

履歴
ColdFusion 10：preserveCase 属性と encodeValue 属性が追加されました。

ColdFusion MX 6.1:

- expires 属性が変更され、日付 / 時刻オブジェクトを受け入れるようになりました。
- Cookie 名には、カンマ、セミコロン、空白文字を除くすべての ASCII 文字を含めることができます。

ColdFusion 9:

- httponly 属性が追加されました。

属性

属性	必須 / オプション	デフォルト	説明
name	必須		Cookie 変数の名前です。ColdFusion は Cookie 名をすべて大文字に変換します。このタグを使って設定する Cookie 名には、カンマ、セミコロン、空白文字を除くすべての印刷可能な ASCII 文字を含めることができます。
domain	path 属性を指定している場合は必須、それ以外の場合はオプション		Cookie が有効で、ユーザーのシステムから Cookie の内容を送信できるドメインを指定します。デフォルトでは、Cookie はそれを設定したサーバーでのみ使用できます。この属性を指定すると、Cookie を別のサーバーで使用できるようになります。 ピリオド (.) で始める必要があります。値がサブドメインの場合は、その文字列で終わるすべてのドメイン名が有効なドメインになります。この属性は、サイト上で Cookie を使用できる利用可能なサブドメインを設定します。 国コードで終わる domain 値の場合は、".mongo.state.us" のように、3 つ以上のピリオドが含まれている必要があります。トップレベルドメインの場合は、".mgm.com" のように、ピリオドは 2 つだけ必要です。 IP アドレスをドメインとして使用することはできません。
encodevalue	オプション		Cookie の値をエンコードするかどうかを指定します。
expires	オプション	セッションのみ	Cookie 変数の有効期間です。 • Cookie は、ユーザーがブラウザを閉じると無効になります。つまり、Cookie はセッションのみで有効です。 • 日付、または日付 / 時刻オブジェクト (例 : 10/09/97) • 日数 (例 : 10、100) • now: クライアントの "cookie.txt" ファイルから Cookie を削除します (ただし、アクティブページの Cookie スコープ内の対応する変数は削除しません)。 • never: Cookie は作成時刻から 30 年後に時間切れになります (Web 上の時間でいえば、実質的に無期限ということになります)。
httponly	オプション		yes の場合、Cookie が httponly に設定され、JavaScript を使用してアクセスできなくなります。ただし、ブラウザが httponly に対応している必要があります。
path	オプション		この Cookie を適用するドメイン内の URL です。通常はディレクトリです。指定のパス内のページのみがこの Cookie を使用できます。デフォルトでは、Cookie を設定したサーバー上のすべてのページがその Cookie にアクセスできます。 `path = "/services/login"` 複数の URL を指定するには、複数の cfcookie タグを使用します。 path を指定する場合は、domain 属性も指定します。
preserveCase	オプション	False	Cookie 名の大文字と小文字を区別するかどうかを指定します。
secure	オプション		ブラウザで SSL (Secure Socket Layer) がサポートされていない場合、Cookie は送信されません。Cookie を使用するには、https プロトコルを使ってページにアクセスする必要があります。 • yes: 変数を安全に転送する必要があります。 • no
value	オプション		Cookie 変数に割り当てる値です。文字列、または文字列として保管できる変数を指定する必要があります。

使用方法

このタグで、現在のブラウザセッションが終わっても Cookie が保存されるように指定した場合、クライアントブラウザによって Cookie がローカル Cookie ファイルに書き込まれるか更新されます。Cookie は、ブラウザが閉じられるまでブラウザメモリ内に残ります。expires 属性を指定しない場合、Cookie はブラウザの Cookie ファイルに書き込まれません。

ページ内で cfflush タグの後にこのタグを使用した場合、この Cookie はブラウザに送信されません。ただし、そのブラウザセッション中は、設定した値を Cookie スコープ内で使用することができます。

注意: 現在のブラウザセッションが無効になったときに無効になる Cookie を作成するには、cfset タグまたは CFScript の代入ステートメントを使用して Cookie スコープ内に変数を設定するという方法もあります (例 : <cfset Cookie.mycookie="sugar">)。Cookie の値を取得するには、Cookie スコープ内でその Cookie の名前を参照します (例 : <cfif Cookie.mycookie is "oatmeal">)。

次の例のように、Cookie 名にピリオド (.) を使用することができます。

```
<cfcookie name="person.name" value="wilson, john">
<cfset cookie.person.lastname="Santiago">
```

自分が設定した Cookie でも、クライアントから送信された Cookie でも、Cookie にアクセスするには Cookie スコープを使用します。たとえば、先ほどの例で設定した person.name という Cookie の値を表示するには、次のようにします。

```
<cfoutput>#cookie.person.name#</cfoutput>
```

例

```
<!--- This example shows how to set/delete a cfcookie variable. --->
<!--- Select users who have entered comments into a sample database. --->
<cfquery name = "GetAolUser" dataSource = "cfdocexamples">
    SELECT EMail, FromUser, Subject, Posted
    FROM Comments
</cfquery>
<html>
<body>
<h3>cfcookie Example</h3>
<!--- If the URL variable delcookie exists, set cookie expiration date
    to NOW --->
<cfif IsDefined("url.delcookie") is True>
    <cfcookie name = "TimeVisited"
    value = "#Now()#"
    expires = "NOW">
<cfelse>
<!--- Otherwise, loop through list of visitors; stop when you match
    the string aol.com in a visitor's e-mail address. --->
<cfloop query = "GetAolUser">
    <cfif FindNoCase("aol.com", Email, 1) is not 0>
        <cfcookie name = "LastAOLVisitor"
        value = "#Email#"
```

```
        expires = "NOW" >
    </cfif>
</cfloop>
<!--- If the timeVisited cookie is not set, set a value. --->
    <cfif IsDefined("Cookie.TimeVisited") is False>
        <cfcookie name = "TimeVisited"
        value = "#Now()#"
        expires = "10">
    </cfif>
</cfif>
<!--- Show the most recent cookie set. --->
<cfif IsDefined("Cookie.LastAOLVisitor") is "True">
    <p>The last AOL visitor to view this site was
    <cfoutput>#Cookie.LastAOLVisitor#</cfoutput>, on
    <cfoutput>#DateFormat(COOKIE.TimeVisited)#</cfoutput>
<!--- Use this link to reset the cookies. --->
<p><a href = "cfcookie.cfm?delcookie = yes">Hide my tracks</A>
<cfelse>
    <p>No AOL Visitors have viewed the site lately.
</cfif>
```

タグ d ～ e

cfdbinfo

説明
データソースに関する情報を取得します。この情報には、データベース、テーブル、クエリー、プロシージャ、外部キー、インデックスなどの詳細や、データベース、ドライバ、JDBC のバージョン情報などが含まれます。

カテゴリ
データベース操作タグ

シンタックス

```
<cfdbinfo
    datasource="data source name"
    name="result name"
    type="dbnames|tables|columns|version|procedures|foreignkeys|index"
    dbname="database name"
    password="password"
    pattern="filter pattern"
    table="table name"
    username="username">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目
cfinsert、cfprocparam、cfprocresult、cfqueryparam、cfstoredproc、cftransaction、cfupdate、『ColdFusion アプリケーションの開発』の Optimizing database use

履歴
ColdFusion 8: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
datasource	オプション		データベースへの接続に使用するデータソースです。
name	必須		結果の参照に使用する名前です。
type	必須		取得する情報のタイプです。 • dbnames: データベースの名前とタイプ • tables: 名前、タイプ、備考 • columns: 名前、SQL データ型、サイズ、小数部桁数、デフォルト値、文字データ型または整数データ型の列の最大長 (バイト数)、null が許可されているかどうか、順序位置、備考、列がプライマリキーかどうか、列が外部キーかどうか、外部キーが参照するテーブル、外部キーが参照するキー名 • version: データベースの製品名とバージョン、ドライバの名前とバージョン、JDBC のメジャーバージョンとマイナーバージョン • procedures: 名前、タイプ、備考 • foreignkeys: 外部キーの名前とテーブル、プライマリキーの名前、削除と更新のルール • index: 名前、インデックスが適用される列、順序位置、基数、行がテーブルの統計またはインデックスであるかどうか、テーブルまたはインデックスが使用するページ数、インデックス値が一意かどうか • ClientInfo：指定されたデータソースのクライアント情報のメタデータです。
dbname	オプション		データベースの名前です。action = "This overrides the one mentioned as a part of datasource definition." の場合にのみ使用されます。
password	オプション		データベースに接続するためのパスワードです。
pattern	オプション		type = "tables"、type = "columns"、または type = "procedures" の場合にのみ使用されます。特定のテーブル、列、またはストアドプロシージャに関する情報を取得するためのフィルタを指定します。1 個のワイルドカード文字を表すには下線 (_) を使用し、0 個以上の文字のワイルドカードを表すにはパーセント記号 (%) を使用します。
table	type = "columns" または type = "foreignkeys" または type = "index" の場合は必須		情報を取得するテーブルの名前です。
username	オプション	no	データベースに接続するために使用するユーザー名です。

使用方法

cfdbinfo タグを使用すると、データベースに関する情報を含むクエリーオブジェクトが返されます。クエリーオブジェクトの内容は、type 属性で指定した値によって異なります。次の表に、各タイプに対応するクエリーオブジェクトの内容を示します。

タイプ	列名	説明
dbnames	DATABASE_NAME	データベースの名前です。
	TYPE	データベースのタイプです (スキーマまたはカタログ)。
tables	TABLE_NAME	テーブルの名前です。
	TABLE_TYPE	テーブルのタイプです (ビュー、テーブル、シノニム、システムテーブルなど)。
	REMARKS	テーブルの備考です。
columns	COLUMN_NAME	列の名前です。
	TYPE_NAME	列の SQL データ型です。
	IS_NULLABLE	列で null が許可されているかどうかを示します。
	IS_PRIMARYKEY	列がプライマリキーかどうかを示します。
	IS_FOREIGNKEY	列が外部キーかどうかを示します。
	REFERENCED_PRIMARYKEY	列が外部キーの場合は、参照先のテーブルの名前です。
	REFERENCED_PRIMARYKEY_TABLE	列が外部キーの場合は、参照先のキーの名前です。
	COLUMN_SIZE	列のサイズです。
	DECIMAL_DIGITS	小数部の桁数です。
	COLUMN_DEFAULT_VALUE	列のデフォルト値です。
	CHAR_OCTET_LENGTH	文字データ型または整数データ型の列の最大長です (単位 : バイト)。
	ORDINAL_POSITION	列の順序位置です。
	REMARKS	列の備考です。
version	DATABASE_VERSION	データベース管理システムのバージョンです。
	DATABASE_PRODUCTNAME	データベース管理システムの名前です。
	DRIVER_VERSION	データベースドライバのバージョンです。
	DRIVER_NAME	データベースドライバの名前です。
	JDBC_MAJOR_VERSION	ドライバのメジャーバージョン番号です。
	JDBC_MINOR_VERSION	ドライバのマイナーバージョン番号です。
procedures	PROCEDURE_NAME	ストアドプロシージャの名前です。
	REMARKS	ストアドプロシージャの備考です。
	PROCEDURE_TYPE	プロシージャのタイプです。プロシージャが結果を返すかどうかを示します。
foreignkeys	FKCOLUMN_NAME	外部キーの名前です。
	FKTABLE_NAME	外部キーのテーブル名です。
	PKCOLUMN_NAME	プライマリキーの名前です。
	DELETE_RULE	従属レコードを持つレコードを削除したときに実行されるアクションを示します。
	UPDATE_RULE	従属レコードを持つレコードを更新したときに実行されるアクションを示します。

タイプ	列名	説明
index	INDEX_NAME	インデックスの名前です。タイプがテーブル統計の場合は空になります。
	COLUMN_NAME	インデックスが適用される列の名前です。タイプがテーブル統計の場合は空になります。
	ORDINAL_POSITION	順序位置です。
	CARDINALITY	タイプがインデックスの場合は一意の値の数、タイプが統計の場合は行の数です。
	TYPE	行がテーブル統計であるかインデックスであるかを示します。インデックスのタイプは、クラスタ、ハッシュ、またはその他です。
	PAGES	タイプがテーブル統計の場合はテーブルが使用するページ数、インデックスの場合はインデックスが使用するページ数です。
	NON_UNIQUE	インデックス値が一意であるかどうかを示します。

例

```
 <cfset datasrc = "oratest">

<cfdbinfo
    type="dbnames"
    datasource="#datasrc#"
    name="dbdata">

<cfoutput>
The #datasrc# data source has the following databases:<br />
</cfoutput>
<table border="1">
<tr>
    <th valign="top" align="left">Database name</th><th>Type</th>
</tr>
    <cfoutput query="dbdata">
    <tr>
        <td>#dbdata.DATABASE_NAME#</td><td>#dbdata.TYPE#</td>
    </tr>
    </cfoutput>
</table>
```

cfdefaultcase

説明

cfswitch タグ本文でのみ使用されます。cfswitch タグで指定された式が cfcase タグで指定された値と一致しないときに実行されるコードを含みます。

カテゴリ

フロー制御タグ

シンタックス

```
 <cfdefaultcase>
```

関連項目

cfcase、cfswitch、『ColdFusion アプリケーションの開発』の cfswitch, cfcase, and cfdefaultcase

履歴
ColdFusion MX: 配置の必要条件が変更されました。このタグを、cfswitch タグ本文の中にあるすべての cfcase タグよりも後に記述する必要がなくなりました。

使用方法
cfdefaultcase タグ本文の内容が実行されるのは、cfswitch タグの expression 属性が、cfswitch タグ本文の中にある cfcase タグで指定されたどの値にも一致しない場合だけです。cfdefaultcase タグ本文の内容には、HTML およびテキストと、CFML のタグ、関数、変数、および式を含めることができます。

cfdefaultcase タグは、cfswitch タグ内に 1 つしか指定できません。cfdefaultcase タグは cfswitch ステートメント内の任意の場所に配置できます。最後の項目にする必要はありませんが、最後に配置するのが慣例になっています。

例
```
 <!--- The following example displays a grade based on a 1-10 score.
        Several of the cfcase tags match more than one score.
        For simplicity, the example sets the score to 7. --->
<cfset score="7">
<cfswitch expression="#score#">
    <cfcase value="10">
        <cfset grade="A">
    </cfcase>
    <cfcase value="9;8" delimiters=";">
        <cfset grade="B">
    </cfcase>
    <cfcase value="7;6" delimiters=";">
        <cfset grade="C">
    </cfcase>
    <cfcase value="5;4;" delimiters=";">
        <cfset grade="D">
    </cfcase>
    <cfdefaultcase>
        <cfset grade="F">
    </cfdefaultcase>
</cfswitch>
<cfoutput>
    Your grade is #grade#
</cfoutput>
```

cfdirectory

説明
ディレクトリに関する操作を管理します。

カテゴリ
ファイル管理タグ

シンタックス

```
<cfdirectory
    directory = "directory name"
    action = "list|copy|create|delete|rename"
    destination = "full pathname"
    filter = "list filter"
    listInfo = "name|all"
    mode = "permission"
    name = "query name"
    newDirectory = "new directory name"
    recurse = "yes|no"
    sort = "sort specification"
    storeACL = "S3_premissions"
    storeLocation = "location"
    type = "file|dir|all">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cffile

履歴

ColdFusion 10：copy アクションと destination 属性が追加されました。

ColdFusion 9.0.1：storeACL 属性と storeLocation 属性が追加されました。

ColdFusion 8: listinfo 属性と type 属性が追加されました。

ColdFusion MX 7: recurse 属性と directory 結果セット列が追加されました。

ColdFusion MX:

❖ action = "list" の動作が変更されました。

- Windows の場合、cfdirectory タグで action = "list" を指定したときに、現在のディレクトリとその親ディレクトリを表す "." (ドット) または ".." (二重ドット) のディレクトリエントリは返されなくなりました。
- Windows の場合、cfdirectory タグで action = "list" を指定したときに、Archive 属性と System 属性の値が返されなくなりました。
- UNIX および Linux の場合、cfdirectory タグで action = "list" を指定したときに、mode 列の情報は返されなくなりました。

属性

属性	必須 / オプション	デフォルト	説明
directory	必須		アクションを実行する対象ディレクトリの絶対パス名です。 次の例のように IP アドレスを使用できます。 `<cfdirectory directory="//12.3.123.123/c_drive/" name="dirQuery" action="LIST">`
action	オプション	list	• list: 指定されたディレクトリにあるファイルのクエリーレコードセットを返します。現在のディレクトリと親ディレクトリを表す "."(ドット) と ".."(二重ドット) のディレクトリエントリは返されません。 • create • delete • rename • copy
destination	action = "copy" の場合は必須		宛先のディレクトリのパスです。絶対パスを指定しない場合は、ソースディレクトリを基準とした相対パスを指定します。
filter	action = "list" の場合はオプション		返される名前に適用するファイル拡張子のフィルタです。たとえば、*.cfm などです。適用できるフィルタは 1 つだけです。
listinfo	オプション	all	• all: 結果セットの全情報を含めます。 • name: 結果セットのファイル名のみを含めます。
mode	オプション		action = "create" とともに使用し、許可を表します。UNIX および Linux だけに適用されます。chmod コマンドの 8 進数値を使用し、所有者、グループ、および他の利用者それぞれに割り当てられます。例 : • 644: 所有者に読み取り / 書き込み許可、グループおよび他の利用者に読み取り許可を割り当てます。 • 777: 利用者全員に読み取り / 書き込み / 実行許可を割り当てます。
name	action = "list" の場合は必須		出力レコードセットの名前です。
newDirectory	action = "rename" の場合は必須		ディレクトリの新規名です。
recurse	オプション	no	ColdFusion でサブディレクトリ上のアクションを実行するかどうかを指定します。 • yes • no action="list" および action="delete" の場合に有効です。
sort	オプション (action = "list" の場合に使用)	ASC	ディレクトリリストのソートに使用するクエリー列です。クエリー出力に含まれる列をカンマ区切りのリストで指定します。 列に条件を付ける場合は、次のいずれかの値を使用します。 • asc: 昇順 (a ～ z) のソートです。 • desc: 降順 (z ～ a) のソートです。 たとえば、次のようになります。 `sort = "directory ASC, size DESC, datelastmodified"`

属性	必須 / オプション	デフォルト	説明
storeACL	オプション（action = "create" の場合に使用）		構造体の配列で、各構造体は Permission または Grant を表します。 詳しくは、Using Amazon S3 storage を参照してください。
StoreLocation	オプション（action = "create" の場合に使用）	US	作成したバケットの場所を変更する場合に使用します。場所は、EU、US または US-WEST を指定できます。 詳しくは、Using Amazon S3 storage を参照してください。
type	オプション	all	• file: ファイル名のみを含めます。 • dir: ディレクトリ名のみを含めます。 • all: ファイル名とディレクトリ名の両方を含めます。
storeLocation	オプション		作成したバケットの場所を変更する場合に使用します。場所は、EU または US のいずれかです。デフォルトの場所は US です。
storeACL	オプション		構造体の配列で、各構造体は Permission または Grant を表します。

使用方法

複数のカスタマが使用するサーバーに ColdFusion アプリケーションを置く場合は、権限のないユーザーがこのタグを使用してファイルやディレクトリのアップロードや操作を行えないようにセキュリティを考慮する必要があります。ColdFusion タグを保護する方法については、『ColdFusion 設定と管理』を参照してください。

action = "list" の場合、cfdirectory によって次の結果列が返されます。これらの結果列は cfoutput タグ内で参照できます。

- name: ディレクトリエントリ名です。"." と ".." のエントリは返されません。
- directory: エントリを含むディレクトリです。
- size: ディレクトリエントリのサイズです。
- type: ファイルタイプです。ファイルの場合は file、ディレクトリの場合は dir です。
- dateLastModified: エントリの最終修正日です。
- attributes: ファイル属性です (該当する場合のみ)。
- mode: 空の列です。UNIX の ColdFusion 5 アプリケーションとの下位互換性を保つために残されています。

次の結果列は、標準 CFML 式で使用することができます。結果列名の先頭には、クエリー名が付けられます。

```
 #mydirectory.name#
#mydirectory.directory#
#mydirectory.size#
#mydirectory.type#
#mydirectory.dateLastModified#
#mydirectory.attributes#
#mydirectory.mode#
```

注意: cfdirectory タグが機能していないように見える場合 (たとえば list オペレーションで空の結果セットが返される場合) は、そのディレクトリにアクセスする正しい権限を持っているかどうかを確認してください。たとえば、ColdFusion を Windows 上のサービスとして実行している場合、ColdFusion はデフォルトで System として機能し、リモートシステムやマップされたドライブ上のディレクトリにはアクセスできません。この問題を解決するには、ローカルシステムアカウントでは ColdFusion を実行しないでください。

filter 属性には、1 つまたは複数の文字のパターンを指定します。このパターンに一致する名前はすべて結果リストに含まれます。Windows システムではパターンマッチングの際に大文字と小文字は無視され、UNIX および Linux では大文字と小文字は区別されます。

次の 2 つの文字はパターン内で特別な意味を持ち、メタ文字と呼ばれます。

- アスタリスク (*) は、0 個以上の任意の文字に一致します。
- 疑問符 (?) は、任意の 1 文字に一致します。

次の表に、パターンとそれに一致するファイル名の例を示します。

パターン	一致するファイル
foo.*	任意の拡張子を持つ、foo という名前のファイル (例 : foo.html、foo.cfm、foo.xml)
*.html	.html という拡張子を持つすべてのファイル (.htm という拡張子のファイルは含まれない)
??	2 文字の名前を持つすべてのファイル

例

```
 <!--- EXAMPLE 1: Creating and Renaming
Check that the directory exists to avoid getting a ColdFusion error message. --->
<cfset newDirectory = "otherNewDir">
<cfset currentDirectory = GetDirectoryFromPath(GetTemplatePath()) & "newDir">
<!--- Check whether the directory exists. --->
<cfif DirectoryExists(currentDirectory)>
<!--- If yes, rename the directory. --->
    <cfdirectory action = "rename" directory = "#currentDirectory#"
        newDirectory = "#newDirectory#" >
    <cfoutput>
    <p>The directory existed and the name has been changed to: #newDirectory#</p>
    </cfoutput>
<cfelse>
    <!--- If no, create the directory. --->
    <cfdirectory action = "create" directory = "#currentDirectory#" >
    <cfoutput><p>Your directory has been created.</p></cfoutput>
</cfif>

<!--- EXAMPLE 2: Deleting a directory
Check that the directory exists and that files are not in the directory to avoid getting ColdFusion error
messages. --->

<cfset currentDirectory = GetDirectoryFromPath(GetTemplatePath()) & "otherNewDir">
<!--- Check whether the directory exists. --->
<cfif DirectoryExists(currentDirectory)>
    <!--- If yes, check whether there are files in the directory before deleting. --->
    <cfdirectory action="list" directory="#currentDirectory#"
    name="myDirectory">
    <cfif myDirectory.recordcount gt 0>
    <!--- If yes, delete the files from the directory. --->
        <cfoutput>
        <p>Files exist in this directory. Either delete the files or code
            something to do so.</P>
        </cfoutput>
    <cfelse>
    <!--- Directory is empty - just delete the directory. --->
        <cfdirectory action = "delete" directory = "#currentDirectory#">
        <cfoutput>
        <p>The directory existed and has been deleted.</P>
        </cfoutput>
    </cfif>
<cfelse>
    <!--- If no, post message or do some other function. --->
    <cfoutput><p>The directory did NOT exist.</p></cfoutput>
</cfif>
```

```
<!---EXAMPLE 3: List directories
The following example creates both an array of directory names and a query that contains entries for the
directories only. --->

<cfdirectory directory="C:/temp" name="dirQuery" action="LIST">

<!--- Get an array of directory names. --->
<cfset dirsArray=arraynew(1)>
<cfset i=1>
<cfloop query="dirQuery">
<cfif dirQuery.type IS "dir">
    <cfset dirsArray[i]=dirQuery.name>
    <cfset i = i + 1>
</cfif>
</cfloop>
<cfdump var="#dirsArray#">
<br>
<!--- Get all directory information in a query of queries.--->
<cfquery dbtype="query" name="dirsOnly">
SELECT * FROM dirQuery
WHERE TYPE='Dir'
</cfquery>
<cfdump var="#dirsOnly#">
```

cfdiv

説明

HTML の div タグなどのコンテナタグを作成し、フォームの非同期送信やタグの内容をダイナミックに制御するバインド式を使用できるようにします。

カテゴリ

表示管理タグ

シンタックス

```
 <cfdiv
    bind = "bind expression"
    bindOnLoad = "true|false"
    ID = "HTML tag ID"
    onBindError = "JavaScript function name"
    tagName = "HTML tag name"
/>

OR

<cfdiv
    ID = "HTML tag ID"
    tagName = "HTML tag name">
        tag body contents
</cfdiv>
```

本文と終了タグを省略する場合は、タグの最後を /> で閉じます。

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfajaximport、cflayout、cfpod、cfwindow

履歴
ColdFusion 8: このタグが追加されました。

属性
次の表は、ColdFusion が直接使用する属性のリストです。その他の属性を指定した場合は、タグ属性として HTML タグに直接渡されます。

属性	必須 / オプション	デフォルト	説明
bind	オプション		コンテナの内容を返すバインド式です。この属性を指定した場合、cfdiv タグに本文は指定できません。 **メモ :** この属性で指定した CFML ページに AJAX 機能を使用するタグ (cfform、cfgrid、cfwindow など) が含まれている場合は、そのページで cfdiv タグとともに cfajaximport タグを使用する必要があります。詳細については、cfajaximport を参照してください。
bindOnLoad	オプション	true	• *true*: 最初にタグをロードしたときに bind 属性式を実行します。 • false: 最初のバインドイベントが発生するまで bind 属性式を実行しません。 この属性を使用する場合は、bind 属性も指定します。 詳細については、『ColdFusion アプリケーションの開発』の Using Ajax User Interface Components and Features の Using the bindOnLoad attribute を参照してください。
ID	オプション		生成されるコンテナタグに割り当てる HTML ID 属性値です。
onBindError	オプション	「説明」を参照	バインド式の評価でエラーが発生した場合に実行する JavaScript 関数の名前です。この関数は 2 つの属性 (HTTP ステータスコードとメッセージ) を取る必要があります。 この属性を省略し、(ColdFusion.setGlobalErrorHandler 関数を使用して) グローバルエラーハンドラを指定すると、エラーメッセージが表示されます。それ以外の場合は、デフォルトのエラーポップアップウィンドウが表示されます。 この属性を使用する場合は、bind 属性も指定します。
tagName	オプション	DIV	作成する HTML コンテナタグです。

使用方法
デフォルトでは、cfdiv タグは HTML 要素 div を作成します。HTML と CSS の標準的なテクニックを使用して、要素とその内容の位置および外観を制御できます。

tagName 属性を使用すると、span や b などの HTML コンテンツ要素を作成して、要素の内容を挿入できます。cfdiv タグは、本文に HTML マークアップコンテンツを直接配置できるタグ (span、i、b、p など) を作成する場合に使用します。本文に HTML マークアップコンテンツを直接配置できないタグ (input、option、frameset など) の作成には使用できません。

cfdiv タグ内にあるフォーム (バインド式から返される HTML に含まれる) を送信すると、フォームは非同期で送信され、フォーム送信からのレスポンスが cfdiv 領域に挿入されます。

bind 属性を指定すると、バインド式を使用して要素がダイナミックに挿入されます。バインド式では、CFC 関数、JavaScript 関数、URL、または bind パラメータを含む文字列を指定できます。内容の取得中は、アニメーションのアイコンと「ロード中 ...」というテキストが表示されます。バインド属性とバインド式の使用方法の詳細については、『ColdFusion アプリケーションの開発』の Using AJAX Data and Development Features を参照してください。

例
次の簡単な例で、cfdiv タグの使用方法を示します。この例では、バインドを使用して、テキスト入力フィールドの内容を HTML DIV 領域に表示します。

メインのアプリケーションファイルである cfdivtag.cfm ファイルの内容は次のとおりです。

```
 <html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>cfdiv Example</title>
</head>

<body>
<cfform>
    <cfinput name="tinput1" type="text">
</cfform>

<h3> using a div</h3>
<cfdiv bind="url:divsource.cfm?InputText={tinput1}" ID="theDiv"
        style="background-color:##CCffFF; color:red; height:350"/>
</body>
</html>
```

div 領域の内容を定義する "divsource.cfm" ファイルのコードは、次のとおりです。

```
 <h3>Echoing main page input:</h3>
<cfoutput>
    <cfif isdefined("url.InputText") AND url.InputText NEQ "">
        #url.InputText#
    <cfelse>
        No input
    </cfif>
</cfoutput>
```

このコードをテストするには、cfdivtag.cfm ページを実行し、任意のテキストを入力した後、Tab キーを押してテキストボックスからフォーカスを移動するか、テキストボックスの外をクリックします。div 領域の背景はライトブルーで表示され、テキストは赤で表示されます。テキストボックスの外にフォーカスを移動すると、入力したテキストが表示されます。

cfdocument

説明

CFML および HTML を含むテキストブロックから PDF または FlashPaper 形式の出力を作成します。

カテゴリ

データ出力タグ

シンタックス

```
<cfdocument
    format = "PDF|FlashPaper"
    authPassword = "authentication password"
    authUser = "authentication user name"
    backgroundVisible = "yes|no"
    bookmark = "yes|no"
    encryption = "128-bit|40-bit|none"
    filename = "filename"
    fontEmbed = "yes|no"
    formfields = "yes|no"
    formsType = "FDF|PDF|HTML|XML"
    localUrl = "yes|no"
    marginBottom = "number"
    marginLeft = "number"
    marginRight = "number"
    marginTop = "number"
    mimeType = "text/plain|application/xml|image/jpeg|image/png|image/bmp|image/gif"
    name = "output variable name"
    openpassword = "password to open protected documents"
    orientation = "portrait|landscape"
    overwrite = "yes|no"
    ownerPassword = "password"
    pageHeight = "page height in inches"
    pageType = "page type"
    pageWidth = "page width in inches"
    pdfa = "yes|no"
    permissions = "permission list"
    permissionspassword = "password to access restricted permissions"
    proxyHost = "IP address or server name for proxy host"
    proxyPassword = "password for the proxy host"
    proxyPort = "port of the proxy host"
    proxyUser = "user name for the proxy host"
    saveAsName = "PDF filename"
    scale = "percentage less than 100"
    src = "URL|pathname relative to web root"
    srcfile = "absolute pathname to a file"
    tagged = "yes|no"
    unit = "in|cm"
    userAgent = "HTTP user agent identifier"
    userPassword = "password">
        HTML and CFML code
</cfdocument>
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfdocumentitem、cfdocumentsection、cfform、cfpdf、cfpdfform、cfpresentation、cfprint、cfreport

履歴

ColdFusion 9: srcFile 属性で ppt がサポートされるようになりました。

OpenOffice ライブラリを使用した Word ドキュメントから PDF または HTML への変換をサポートするために、次の属性が追加されました。

- formfields 属性
- formsType 属性
- openpassword 属性

- permissionspassword 属性
- pdfa 属性
- tagged 属性

ColdFusion 8: 次の属性および変数が追加されました。

- bookmark 属性
- localUrl 属性
- cfdocument タグ内で cfpdfform タグを使用して既存の PDF フォームを埋め込む機能。
- mimeType 属性が指定されていない場合に、ソースファイル名を基準にしてソースファイルの MIME タイプを判別する機能。
- cfdocument タグで作成された PDF 変数を cfpdf タグのソースとして渡す機能。
- authPassword、authUser、proxyHost、proxyPassword、proxyPort、proxyUser、および userAgent 属性
- saveAsName 属性
- totalsectionpagecount スコープ変数および currentsectionpagenumber スコープ変数

ColdFusion MX 7.01: src、srcfile、および mimetype の各属性が追加されました。

ColdFusion MX 7: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
authPassword	オプション		基本認証のターゲット URL に送信するパスワードです。username と組み合わせて、認証ヘッダで渡される base64 エンコード文字列を形成します。統合 Windows 認証、NTLM 認証、または Kerebos 認証には対応しません。
authUser	オプション		基本認証のターゲット URL に送信するユーザー名です。password と組み合わせて、認証ヘッダで渡される base64 エンコード文字列を形成します。統合 Windows 認証、NTLM 認証、または Kerebos 認証には対応しません。
backgroundVisible	オプション	no	ドキュメントの出力時に背景を出力するかどうかを指定します。 • yes: 出力時に背景を含めます。 • no: 出力時に背景を含めません。
bookmark	オプション	no	ドキュメント内にブックマークを作成するかどうかを指定します。 • yes: ブックマークを作成します。 • no: ブックマークを作成しません。
encryption	オプション	none	(format="PDF" の場合のみ) 出力を暗号化するかどうかを指定します。 • 128-bit • 40-bit • none
filename	オプション		PDF または FlashPaper 形式の出力を保存するファイルの名前です。 filename 属性を省略すると、出力はブラウザに表示されます。

属性	必須 / オプション	デフォルト	説明
fontEmbed	オプション	yes	ColdFusion が出力にフォントを埋め込むかどうかを指定します。 • yes: フォントを埋め込みます。 • no: フォントを埋め込みません。 • selective: Java フォントおよびコアフォントを除くすべてのフォントを埋め込みます。
format	必須		レポートの形式です。 • PDF • FlashPaper
formfields	オプション	yes	この属性は、ColdFusion に OpenOffice が統合されている場合にのみ使用可能です。 フォームフィールドをウィジェットとしてエクスポートするか、またはその固定出力表現のみをエクスポートするかを指定するブール値です。
formstype	オプション	FDF	この属性は、ColdFusion に OpenOffice が統合されている場合にのみ使用可能です。 PDF フォームの送信されたフォーマットを指定します。次のいずれかの値を指定できます。 • FDF • PDF • HTML • XML
localUrl	オプション	no	イメージファイルをローカルドライブから直接取得するかどうかを指定します。 • yes: HTTP、HTTPS、またはプロキシを使用せずに、イメージファイルをローカルドライブから直接取得します。 • no: イメージファイルがローカルに保管されている場合でも、HTTP、HTTPS、またはプロキシを使用してイメージファイルを取得します。 詳細については、「イメージファイルの URL の使用」を参照してください。
marginBottom	オプション		ページの下マージンをインチ (デフォルト) またはセンチメートル単位で指定します。下マージンをセンチメートル単位で指定するには、unit=cm 属性を使用します。
marginLeft	オプション		ページの左マージンをインチ (デフォルト) またはセンチメートル単位で指定します。左マージンをセンチメートル単位で指定するには、unit=cm 属性を使用します。
marginRight	オプション		ページの右マージンをインチ (デフォルト) またはセンチメートル単位で指定します。右マージンをセンチメートル単位で指定するには、unit=cm 属性を使用します。
marginTop	オプション		ページの上マージンをインチ (デフォルト) またはセンチメートル単位で指定します。上マージンをセンチメートル単位で指定するには、unit=cm 属性を使用します。

属性	必須 / オプション	デフォルト	説明
mimeType	オプション	text/html	ソースドキュメントの MIME タイプです。サポートされる MIME タイプは次のとおりです。 • text/html • text/plain • application/xml • image/bmp • image/jpeg • image/png • image/gif この属性を明示的に指定しない場合は、ファイル名に基づいて MIME タイプが判別されます。
name	オプション		PDF または FlashPaper 形式の出力を格納する既存の変数の名前です。
openpassword	オプション		この属性は、ColdFusion に OpenOffice が統合されている場合にのみ使用可能です。 パスワードで保護されたドキュメントを開くために必要なパスワードです。
orientation	オプション	portrait	ページの向き : • portrait • landscape
overwrite	オプション	no	既存のファイルを上書きするかどうかを指定します。filename 属性とともに使用します。
ownerPassword	オプション		(format="PDF" の場合のみ) 所有者のパスワードを指定します。 userPassword と同じものにはできません。
pageHeight	オプション		ページの高さをインチ (デフォルト) またはセンチメートル単位で指定します。この属性は、pagetype=custom の場合にのみ有効です。ページの高さをセンチメートル単位で指定するには、unit=cm 属性を使用します。

属性	必須 / オプション	デフォルト	説明
pageType	オプション	letter	ColdFusion で生成するレポートのページタイプです。 • legal: 8.5 インチ × 14 インチ • letter: 8.5 インチ × 11 インチ • A4: 8.27 インチ × 11.69 インチ • A5: 5.81 インチ × 8.25 インチ • B4: 9.88 インチ × 13.88 インチ • B5: 7 インチ × 9.88 インチ • B4-JIS: 10.13 インチ × 14.31 インチ • B5-JIS: 7.19 インチ × 10.13 インチ • custom: カスタムの高さと幅です。custom を指定する場合は、pageHeight 属性と pageWidth 属性も指定します。必要に応じて、margin に関する属性の指定や、単位をインチまたはセンチメートルのいずれにするかの指定もできます。
pageWidth	オプション		ページの幅をインチ (デフォルト) またはセンチメートル単位で指定します。この属性は、pageType=custom の場合にのみ有効です。ページの幅をセンチメートル単位で指定するには、unit=cm 属性を使用します。
pdfa	オプション	no	この属性は、ColdFusion に OpenOffice が統合されている場合にのみ使用可能です。 タイプ PDF/A-1 (ISO 19005-1:2005) の PDF を作成する必要があるかどうかを指定するブール値です。
permissionpasswrd	オプション		この属性は、ColdFusion に OpenOffice が統合されている場合にのみ使用可能です。 制限されたアクセス許可にアクセスするために必要なパスワードです。制限されたアクセス許可は、permissions 属性を使用して指定されます。
permissions	オプション		(format="PDF" の場合のみ) 次のいずれかのアクセス許可を設定します。 • AllowPrinting • AllowModifyContents • AllowCopy • AllowModifyAnnotations • AllowFillIn • AllowScreenReaders • AllowAssembly • AllowDegradedPrinting 複数のアクセス許可を指定する場合は、カンマで区切ります。
proxyHost	オプション		リクエストの送信先となるプロキシサーバーのホスト名または IP アドレスです。
proxyPassword	オプション		プロキシサーバーで要求されるパスワードです。
proxyPort	オプション	80	プロキシサーバー上で接続するポートです。

属性	必須 / オプション	デフォルト	説明
proxyUser	オプション		プロキシサーバーに提供するユーザー名です。
scale	オプション	ColdFusion が計算する値	スケール係数をパーセントで指定します。このオプションを使用して HTML 出力のサイズを縮小し、用紙のサイズに合うようにします。100 未満の数値を指定します。
saveAsName	オプション		(format="PDF" の場合のみ) ブラウザに出力された PDF ファイルをユーザーが保存するときに [名前を付けて保存] ダイアログボックスに表示されるファイル名です。
src	オプション		URL、または Web ルートからの相対パスです。src 属性と srcfile 属性の両方を指定することはできません。HTML、HTM、BMP、PNG など、ブラウザに出力可能な形式のファイルを指定する必要があります。
srcfile	オプション		サーバー上のファイルの絶対パスです。src 属性と srcfile 属性の両方を指定することはできません。PPT ファイル、Word ファイル、または HTML、HTM、BMP、PNG などブラウザに出力可能な形式のファイルを指定する必要があります。
tagged	オプション	no	この属性は、ColdFusion に OpenOffice が統合されている場合にのみ使用可能です。 PDF が Tagged PDF タグを使用して作成されるかどうかを決定するブール値です。
unit	オプション	in	pageHeight、pageWidth、および margin の各属性で使用するデフォルトの単位です。 • in: インチ • cm: センチメートル
userAgent	オプション	ColdFusion	HTTP User-Agent リクエストヘッダフィールドに配置されるテキストです。リクエストクライアントソフトウェアを識別するために使われます。
userPassword	オプション		(format="PDF" の場合のみ) ユーザーのパスワードを指定します。 ownerPassword と同じものにはできません。

使用方法

cfdocument タグを使用して、HTML および CFML の出力を PDF または FlashPaper 形式でレンダリングします。<cfdocument> タグと </cfdocument> タグのペアの外側にある HTML や CFML は返されません。

cfdocument タグでは、次の標準をサポートする HTML をレンダリングできます。

- HTML 4.01
- XML 1.0
- DOM Level 1 および 2
- CSS1 および CSS2 (詳細については、「サポートされている CSS スタイル」を参照してください)

cfdocument タグは、Microsoft Word で生成された Internet Explorer 固有の HTML をサポートしません。

メモリ内のファイルを指定するには、filename 属性に次のシンタックスを使用します。この場合、ディスクには書き込まれません。メモリ内のファイルを使用すると、一時的データの処理が速くなります。

```
ram:///filepath
```

ファイルパスには、ram:///petStore/tracking/ordersummary.pdf のようなディレクトリを含めることができます。ファイルを指定する前に、パスに含まれるディレクトリを作成しておく必要があります。メモリ内のファイルの使用方法について詳しくは、『ColdFusion アプリケーションの開発』の Working with in-memory files を参照してください。

指定したファイルまたは URL から PDF または FlashPaper 形式の出力を作成するには、src、srcfile、および mimeType の各属性を使用できます。cfdocument タグで結果を表示するには、cfhttp タグの代わりに src 属性および srcfile 属性を使用します。src または srcfile 属性を指定する場合は、cfdocument タグ内に他の内容を含めないでください。ColdFusion では、追加の内容は無視されます。

cfdocument タグで返された PDF または FlashPaper 形式のドキュメントにより、入力ストリーム内の既存の HTML は上書きされ、</cfdocument> タグの後の HTML は無視されます。

cfreport タグを cfdocument タグに埋め込むことはできません。

注意：cfdocument タグの出力でヘッダテキストの一部が表示されない場合は、marginTop 属性の値を増やしてください。

サポートされている CSS スタイル

cfdocument タグでは次の CSS スタイルがサポートされています。

background	background-attachment	background-color	background-image
background-position	background-repeat	border	border-bottom
border-bottom-color	border-bottom-style (solid ボーダーのみ)	border-bottom-width	border-color
border-left	border-left-color	border-left-style (solid ボーダーのみ)	border-left-width
border-right	border-right-color	border-right-style (solid ボーダーのみ)	border-right-width
border-spacing	border-style (solid ボーダーのみ)	border-top	border-top-color
border-top-style (solid ボーダーのみ)	border-top-width	border-width	bottom
clear	clip	color	content (string、counter のみ)
counter-increment	counter-reset	cursor	display
float	font	font-family	font-size
font-style	font-weight	height	left
letter-spacing	line-height	list-style-type	margin
margin-bottom	margin-left	margin-right	margin-top
outline	outline-color	outline-style (solid、dotted、dashed のみ)	outline-width
padding	padding-bottom	padding-left	padding-right
padding-top	page-break-after	page-break-before	page-break-inside
position	right	text-align (left、right、および center)	text-decoration
text-indent	top	unicode-bidi	vertical-align
visibility	white space (normal、nowrap のみ)	width	z-index

イメージファイルの URL の使用

最適なパフォーマンスと信頼性を確保するには、サーバーに保管されているイメージのローカルファイルの URL を指定することをお勧めします。次の例の cfdocument タグは、イメージファイルがローカルに保管されている場合でも、HTTP を使用してイメージをサーバーにリクエストします。

```
<cfdocument format="PDF">
    <table>
        <tr>
            <td>bird</td>
            <td><image src="images/bird.jpg"></td>
        </tr>
        <tr>
            <td>fruit</td>
            <td><image src="images/fruit.jpg"></td>
        </tr>
        <tr>
            <td>rose</td>
            <td><image src="images/rose.jpg"></td>
        </tr>
    </table>
</cfdocument>
```

また、一部のアプリケーションでは、ブラウザにイメージではなく赤い X というイメージエラーが表示される場合があります。パフォーマンスを向上させ、赤い X というイメージエラーを回避するには、localUrl 属性を yes に設定してください。

```
<cfdocument localUrl="yes" format="PDF">
    <table>
        <tr>
            <td>bird</td>
            <td><image src="images/bird.jpg"></td>
        </tr>
        <tr>
            <td>fruit</td>
            <td><image src="images/fruit.jpg"></td>
        </tr>
        <tr>
            <td>rose</td>
            <td><image src="images/rose.jpg"></td>
        </tr>
    </table>
</cfdocument>
```

スコープ変数

cfdocument タグを使用すると、ColdFusion によって cfdocument という名前のスコープが作成されます。このスコープには、次の変数が含まれます。

- currentpagenumber
- totalpagecount
- totalsectionpagecount
- currentsectionpagenumber

ColdFusion では、cfdocumentitem タグ内の式でスコープ変数を使用できます。

次の例は、currentpagenumber 変数を使用して、偶数ページのヘッダにセクション名を表示し、奇数ページのヘッダに章名を表示する方法を示しています。

```
<cfdocument format="flashpaper">
    <cfdocumentitem type="header" evalAtPrint="true">
        <cfif (cfdocument.currentpagenumber mod 2) is 0>
                <cfoutput>#cfdocument.totalpagecount#</cfoutput>
        <cfelse>
                <cfoutput>#cfdocument.currentpagenumber#</cfoutput>
        </cfif>
    </cfdocumentitem>
        ...
</cfdocument>
```

cfdocument タグ内で cfdocumentsection タグを定義した場合は、totalsectionpagecount 変数を次のように指定します。

```
<cfdocument format="pdf">
    <cfdocumentitem type="header" evalatprint="true" >
    <cfif (cfdocument.currentpagenumber mod 2) is 0>
    <cfoutput>#cfdocument.totalpagecount#</cfoutput>
        <cfelse>
            <cfoutput>#cfdocument.currentpagenumber#</cfoutput>
    </cfif>
<cfoutput>cfdocument.currentpagenumber :#cfdocument.currentpagenumber#</cfoutput>
<cfoutput>cfdocument.totalpagecount :#cfdocument.totalpagecount#</cfoutput>
<cfoutput>cfdocument.totalsectionpagecount :#cfdocument.totalsectionpagecount#</cfoutput>
<cfoutput>cfdocument.currentsectionpagenumber :#cfdocument.currentsectionpagenumber#</cfoutput>
</cfdocumentitem>

<cfdocumentitem type="footer" evalatprint="true" >
    <cfif ! (cfdocument.currentpagenumber mod 2) is 0>
    <cfoutput>if#cfdocument.totalpagecount#</cfoutput>
        <cfelse>
    <cfoutput>else#cfdocument.currentpagenumber#</cfoutput>
    </cfif>
</cfdocumentitem>
<cfdocumentsection >Example Text
</cfdocumentsection>
</cfdocument>
```

ブックマーク

ColdFusion 9 はブックマークをサポートしています。cfdocument タグで bookmark 属性を yes に設定します。次に、cfdocumentsection タグごとにブックマーク名を指定します。

次の例は、ドキュメントのセクションごとにブックマークを指定する方法を示しています。

```
 <!--- This example creates two bookmarks named "Section 1" and "Section 2" in a PDF file. --->
<cfdocument format="pdf" bookmark="yes">
    <cfdocumentsection name="Section 1">
<!--- Insert HTML content here.--->
    </cfdocumentsection>
    <cfdocumentsection name="Section 2">
<!--- Insert HTML content here. --->
    </cfdocumentsection>
</cfdocument>
```

例

例 1

```
 <!--- This example creates generates a FlashPaper document. --->
<cfdocument format="flashpaper">
<p>This is a document rendered by the cfdocument tag.</p>

<table width="50%" border="2" cellspacing="2" cellpadding="2">
<tr>
<td><strong>Name</strong></td>
<td><strong>Role</strong></td>
</tr>
<tr>
<td>Bill</td>
<td>Lead</td>
</tr>
<tr>
<td>Susan</td>
<td>Principal Writer</td>
</tr>
<tr>
<td>Adelaide</td>
<td>Part Time Senior Writer</td>
</tr>
<tr>
<td>Thomas</td>
<td>Full Time for 6 months</td>
</tr>
<tr>
<td>Michael</td>
<td>Full Time for 4 months</td>
</tr>
</table>
</cfdocument>
```

例 2

```
 <!--- The following example shows how to use the cfdocument scope variables to generate section numbers and
page numbers. --->

<cfdocument format="pdf">
<cfdocumentitem type="header" evalatprint="true">
    <table width="100%" border="0" cellpadding="0" cellspacing="0">
        <tr><td align="right"><cfoutput>#cfdocument.currentsectionpagenumber# of
            #cfdocument.totalsectionpagecount#</cfoutput></td></tr>
    </table>
</cfdocumentitem>

<cfdocumentitem type="footer" evalatprint="true">
    <table width="100%" border="0" cellpadding="0" cellspacing="0">
        <tr><td align="center"><cfoutput>#cfdocument.currentpagenumber# of
            #cfdocument.totalpagecount#</cfoutput></td></tr>
    </table>
</cfdocumentitem>

<cfdocumentsection>
    <h1>Section 1</h1>
    <cfloop from=1 to=50 index="i">
        Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore
et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip
ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu
fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt
mollit anim id est laborum.<p>
    </cfloop>
</cfdocumentsection>
```

```
<cfdocumentsection>
    <h1>Section 2</h1>
    <cfloop from=1 to=50 index="i">
        Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore
et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip
ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu
fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt
mollit anim id est laborum.<p>
    </cfloop>
</cfdocumentsection>

<cfdocumentsection>
<h1>Section 3</h1>
    <cfloop from=1 to=50 index="i">
        Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore
et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip
ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu
fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt
mollit anim id est laborum.<p>
    </cfloop>
</cfdocumentsection>
</cfdocument>
```

cfdocumentitem

説明

cfdocument タグによって作成された PDF または FlashPaper ドキュメントに対するアクション項目を指定します。次のアクション項目があります。

- header
- footer
- pagebreak

カテゴリ

データ出力タグ

シンタックス

```
<cfdocument ...>
    <cfdocumentitem
    type = "pagebreak|header|footer"
    evalAtPrint = "true"
    header/footer text</cfdocumentitem>
</cfdocument>
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfreport、cfdocument、cfdocumentsection

履歴

ColdFusion 9：evalAtPrint 属性が追加されました。

ColdFusion 8: cfdocument.currentpagenumber、cfdocument.totalpagecount cfdocument.totalsectionpagecount、および cfdocument.currentsectionpagenumber スコープ変数を使用できるようになりました。

ColdFusion MX 7.01: src、srcfile、および mimetype の各属性が追加されました。

ColdFusion MX 7: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
type	必須		アクションを指定します。 • pagebreak: タグの位置で新規ページを開始します。 • header: <cfdocumentitem> タグと </cfdocumentitem> タグの間のテキストをランニングヘッダとして使用します。 • footer: <cfdocumentitem> タグと </cfdocumentitem> タグの間のテキストをランニングフッタとして使用します。
evalAtPrint	オプション	false	ドキュメントの出力時に cfdocumentitem タグ本文の内容を評価するかどうかを決定するブール値です。 • true: cfdocumentitem タグ本文の内容をドキュメントの出力時にのみ評価します。 • false: cfdocumentitem タグ本文の内容をすぐに評価します。

使用方法

evalAtPrint タグは、出力前にドキュメントの内容を評価し、追加の属性も受け入れる場合に使用します。

cfdocumentitem タグを使用して、PDF または FlashPaper レポートの形式設定を制御します。このタグは、<cfdocument> と </cfdocument> のペアで囲む必要があります。

改ページ、ランニングヘッダ、またはランニングフッタごとに 1 つの cfdocumentitem タグをコーディングします。

ColdFusion では、cfdocumentitem タグ内で cfdocument のスコープ変数を使用できるようになりました。cfdocument のスコープ変数 cfdocument.currentpagenumber を使用して、現在のページ番号をヘッダまたはフッタに表示することができます。cfdocument.totalpagecoun を使用して、ページの総数を表示することもできます。次に例を示します。

```
 ...
<cfdocumentitem type= "footer>
      #cfdocument.currentpagenumber# of #cfdocument.totalpagecount#
</cfdocumentitem>
```

cfdocument.totalsectionpagecount および cfdocument.currentsectionpagenumber スコープ変数を使用する例については、cfdocument を参照してください。

cfdocumentsection タグを cfdocumentsection タグと組み合わせて使用する場合と、組み合わせずに使用する場合の結果は次のようになります。

cfdocumentsection が設定されていない場合： cfdocumentitem 属性は、次に示すようにドキュメント全体に適用されます。

- タグがドキュメントの先頭にある場合は、ドキュメント全体に適用されます。
- タグがドキュメントの中央にある場合は、ドキュメントの残りの部分に適用されます。
- タグがドキュメントの終わりにある場合は、何も効果を持ちません。

cfdocumentsection タグが設定されている場合： cfdocumentitem 属性はそのセクションのみに適用され、以前に指定したヘッダおよびフッタは無効になります。

例

```
 <cfquery datasource="cfdocexamples" name="parksQuery">
    SELECT parkname, suptmgr from parks
</cfquery>

<cfdocument format="PDF">
<cfdocumentitem type="header">National Parks Report</cfdocumentitem>
<!--- Use a footer with current page of totalpages format. --->
<cfdocumentitem type="footer">
<cfoutput>Page #cfdocument.currentpagenumber# of #cfdocument.totalpagecount#</cfoutput>
    </cfdocumentitem>

<h1>Park list</h1>
<table width="95%" border="2" cellspacing="2" cellpadding="2" >
<tr>
<th>Park</th>
<th>Manager</th>
</tr>
    <cfoutput query="parksQuery">
    <tr>
<td><font size="-1">#parkname#</font></td>
<td><font size="-1">#suptmgr#</font></td>
    </tr>
    </cfoutput>
</table>
</cfdocument>
```

cfdocumentsection

説明

PDF または FlashPaper 形式のドキュメントをセクションに分割します。このタグを cfdocumentitem タグとともに使用すると、各セクションに固有のヘッダ、フッタ、ページ番号を設定することができます。

カテゴリ

データ出力タグ

シンタックス

```
 <cfdocument ...>
    <cfdocumentsection
        authPassword = "authentication password"
        authUser = "authentication user name"
        marginBottom = "number"
        marginLeft = "number"
        marginRight = "number"
        marginTop = "number"
        mimeType = "text/plain|application/xmlimage/jpeg|image/png|image/bmp|image/gif"
        name = "bookmark for the section"
        src = "URL|path relative to web root"
        srcfile = "absolute path of file"
        userAgent = "HTTP user agent identifier">
        HTML, CFML, and cfdocumentitem tags
    </cfdocumentsection>
</cfdocument>
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfreport、cfdocument、cfdocumentitem

履歴

ColdFusion 8: name、authPassword、authUser、および userAgent の各属性が追加されました。

ColdFusion MX 7.01: src、srcfile、および mimetype の各属性が追加されました。

ColdFusion MX 7: このタグおよび margintop、marginbottom、marginleft、marginright の各属性が追加されました。

属性

属性	必須 / オプション	デフォルト	説明
authPassword	オプション		基本認証のターゲット URL に送信するパスワードです。username と組み合わせて、認証ヘッダで渡される base64 エンコード文字列を形成します。統合 Windows 認証、NTLM 認証、または Kerebos 認証には対応しません。
authUser	オプション		基本認証のターゲット URL に送信するユーザー名です。password と組み合わせて、認証ヘッダで渡される base64 エンコード文字列を形成します。統合 Windows 認証、NTLM 認証、または Kerebos 認証には対応しません。
marginBottom	オプション		ページの下マージンをインチ (デフォルト) またはセンチメートル単位で指定します。下マージンをセンチメートル単位で指定するには、unit="cm" 属性を cfdocument 親タグに含めます。
marginLeft	オプション		ページの左マージンをインチ (デフォルト) またはセンチメートル単位で指定します。左マージンをセンチメートル単位で指定するには、unit="cm" 属性を cfdocument 親タグに含めます。
marginRight	オプション		ページの右マージンをインチ (デフォルト) またはセンチメートル単位で指定します。右マージンをセンチメートル単位で指定するには、unit="cm" 属性を cfdocument 親タグに含めます。
marginTop	オプション		ページの上マージンをインチ (デフォルト) またはセンチメートル単位で指定します。上マージンをセンチメートル単位で指定するには、unit="cm" 属性を cfdocument 親タグに含めます。
mimeType	オプション	text/html	ソースドキュメントの MIME タイプです。サポートされる MIME タイプは次のとおりです。 • text/html • text/plain • application/xml • image/jpeg • image/png • image/gif この属性を明示的に指定しない場合は、ファイル名に基づいて MIME タイプが判別されます。
name	オプション		セクションのブックマーク名です。

属性	必須 / オプション	デフォルト	説明
src	オプション		URL、または Web ルートからの相対パスです。src 属性と srcfile 属性の両方を指定することはできません。
srcfile	オプション		サーバーにあるディスク上またはメモリ内のファイルの絶対パスです。src 属性と srcfile 属性の両方を指定することはできません。
userAgent	オプション	ColdFusion	HTTP User-Agent リクエストヘッダフィールドに配置されるテキストです。リクエストクライアントソフトウェアを識別するために使われます。

使用方法

cfdocumentsection タグを使用して、レポートをセクションに分割します。それぞれの cfdocumentsection タグ内で cfdocumentitem タグを使用して、各セクションに固有のヘッダおよびフッタを指定することができます。

cfdocumentsection を使用する場合、ColdFusion は cfdocumentsection タグの外にある HTML と CFML を無視します。

各セクションのマージン属性は、前のセクションまたは親の cfdocument タグで指定されたマージンよりも優先されます。マージンに関する属性を指定する場合、その単位は cfdocument 親タグの unit 属性で制御されます。unit 属性のデフォルトの単位はインチです。cfdocumentsection タグは、各セクションが新しいページから始まるように改ページを挿入します。

ColdFusion では、ブックマークをサポートするために name 属性が追加されました。documentsection タグのレベルで定義されたブックマークは、cfdocument ルートの子になります。

例

例 1

```
 <cfquery datasource="cfdocexamples" name="empSalary">
SELECT Emp_ID, firstname, lastname, e.dept_id, salary, d.dept_name
FROM employee e, departmt d
WHERE e.dept_id = d.dept_id
ORDER BY d.dept_name
</cfquery>

<cfdocument format="PDF">
<cfoutput query="empSalary" group="dept_id">
    <cfdocumentsection>
    <cfdocumentitem type="header">
        <font size="-3"><i>Salary Report</i></font>
    </cfdocumentitem>
    <cfdocumentitem type="footer">
        <font size="-3">Page #cfdocument.currentpagenumber#</font>
    </cfdocumentitem>
    <h2>#dept_name#</h2>
     <table width="95%" border="2" cellspacing="2" cellpadding="2" >
    <tr>
         <th>Employee</th>
         <th>Salary</th>
    </tr>
    <cfset deptTotal = 0 >
    <!--- inner cfoutput --->
```

```
    <cfoutput>
    <tr>
<td><font size="-1">
        #empSalary.lastname#, #empSalary.firstname#</font>
        </td>
        <td align="right"><font size="-1">
        #DollarFormat(empSalary.salary)#</font>
        </td>
    </tr>
         <cfset deptTotal = deptTotal + empSalary.salary>
    </cfoutput>
        <tr>
<td align="right"><font size="-1">Total</font></td>
        <td align="right"><font size="-1">#DollarFormat(deptTotal)#</font></td>
</tr>
        <cfset deptTotal = 0>
     </table>
    </cfdocumentsection>
    </cfoutput>
</cfdocument>
```

例 2: ブックマーク

```
 <!--- This example uses the name attribute to define bookmarks in a PDF document at the
    section level. --->
<cfdocument format="pdf" bookmark="yes">
    <cfdocumentsection name="section 1">
        <!--- Insert some HTML content here. --->
    </cfdocumentsection>
    <cfdocumentsection name="section 2">
        <!--- Insert some HTML content here. --->
    </cfdocumentsection>
</cfdocument>
```

cfdump

説明

cfdump タグを使用して、ほとんどすべての種類の ColdFusion オブジェクトの要素、変数、値を取得します。このタグはデバッグの際に役立ちます。単純変数、複合変数、オブジェクト、コンポーネント、ユーザー定義関数、その他の要素の内容を表示することができます。cfdump では、CFC をダンプするときに cfproperty で定義されたコンポーネントプロパティが表示されます。PROPERTIES と呼ばれる新しいキーがコンポーネントダンプに追加されました。これはデフォルトで展開された状態になります。また、cfdump のテキスト形式でもこの情報が提供されます。

カテゴリ

デバッグタグ、変数操作タグ

シンタックス

```
<cfdump
    var = "#variable#"
    output = "browser|console|file"
    format = "text|html"
    abort = "true|false">
    label = "text"
    metainfo = "yes|no"
    top = "number of rows|number of levels"
    show = "columns|keys"
    hide = "columns|keys"
    keys = "number of keys to display for structures"
    expand = "yes|no"
    showUDFs = "yes|no">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfcookie、cfparam、cfsavecontent、cfschedule、cfset、cftimer、cfwddx

履歴

- ColdFusion 9：abort 属性が追加されました。
- ColdFusion 8: show、format、hide、keys、metainfo、output、および showUDFs の各属性が追加されました。
- ColdFusion MX 7: top 属性が追加されました。
- ColdFusion MX 6.1: COM オブジェクトをダンプする機能が追加されました。これにより、オブジェクトのメソッドと Get プロパティおよび Put プロパティについての TypeInfo 情報が表示されます。

属性

属性	必須 / オプション	デフォルト	説明
var	必須		表示する変数です。変数名をシャープ記号 (#) で囲みます。 次の種類の変数を使用すると、有意義な cfdump 出力を生成できます。 • array • CFC • COM オブジェクト • ファイルオブジェクト • Java オブジェクト • simple • query • structure • UDF • wddx • xml
expand	オプション	yes	• yes: Internet Explorer および Mozilla で、ビューを展開します。 • no: 展開したビューを縮小します。
format	オプション	text	cfdump の結果をテキストでファイルに保存するか、HTML 形式でファイルに保存するかを指定するために output 属性とともに使用します。
hide	オプション	all	クエリーの場合は、単一の列名または列名のカンマ区切りリストです。構造体の場合は、単一のキーまたはキーのカンマ区切りリストです。 存在しない構造体要素を指定した場合は無視され、エラーにはなりません。
keys	オプション	9999	(構造体の場合のみ) 表示するキーの数です。
label	オプション		文字列です。ダンプ出力用のヘッダを指定します。var 属性の値が単純型の場合は無視されます。
metainfo	オプション	クエリーの場合は yes 永続 CFC の場合は no	クエリーおよび永続 CFC の場合に使用します。クエリーに関する情報 (クエリーがキャッシュされたかどうか、実行時間、SQL など) が cfdump 結果に含まれます。クエリー結果からこの情報を除外するには、metainfo="no" を指定します。永続 CFC の場合、metainfo="yes" を指定すると、getter や setter などのプロパティ属性が返されます。
output	オプション	browser	cfdump の結果の送信先です。使用できる値は次のとおりです。 • browser • console • filename ファイル名を指定する場合は、絶対パス名を使用する必要があります。ファイル名の指定には、絶対パス、または ColdFusion のテンポラリディレクトリを基準とした相対パスを使用できます。ColdFusion のテンポラリディレクトリを調べるには、GetTempDirectory() 関数を使用します。
show	オプション	all	クエリーの場合は、単一の列名または列名のカンマ区切りリストです。構造体の場合は、単一のキーまたはキーのカンマ区切りリストです。

属性	必須 / オプション	デフォルト	説明
showUDFs	オプション	yes	• yes: UDF を含めます (メソッドは折り畳まれた状態です)。 • no: UDF を除外します。
top	オプション	9999	表示する行の数です。構造体の場合、この値は表示するネストレベルの数です。
abort	オプション	false	この属性が "true" に設定されている場合、タグの位置で現在のページの処理を停止します。

使用方法

展開 / 縮小機能は、XML ドキュメントのオブジェクト、構造体、配列などの大きな構造体を操作するときに役に立ちます。

構造体を表示するには、次のようなコードを使用します。ここで、**myDoc** は XmlDocument 型の変数です。

```
<cfif IsXmlDoc(mydoc) is "yes">
    <cfdump var="#mydoc#">
</cfif>
```

このタグの出力は、データ型に応じて色分けされます。

テーブルのセルが空の場合は、このタグにより "[empty string]" が表示されます。

例

```
<!--- This example shows how to use this tag to display the CGI scope as a structure: --->

<cfdump var="#cgi#"> <!--- This displays information about file objects. --->
<cfscript>
myfile = FileOpen("c:\temp\test1.txt", "read");
</cfscript>
myfile refers to:
<cfdump var="#myfile.filepath#">
```

cfelse

説明

cfif タグブロック内の最後の制御ブロックとして使用し、cfif タグまたは cfelseif タグで識別されなかった残りのケースを扱います。

カテゴリ

フロー制御タグ

シンタックス

```
<cfif expression>
        HTML and CFML tags<cfelseif expression>
        HTML and CFML tags
    <cfelse>
        HTML and CFML tags
</cfif>
```

関連項目

cfif、cfelseif、cfabort、cfbreak、cfexecute、cfexit、cflocation、cfloop、cfswitch、cfthrow、cftry

使用方法

cfif タグとすべての cfelseif タグの **expressions** の値が no の場合は、このタグと cfif 終了タグとの間にあるコードが処理されます。このタグは、cfif タグブロックの内部で使用する必要があります。終了タグは必要ありません。

詳細と例については、cfif を参照してください。

cfelseif

説明

cfif タグブロック内の制御ブロックとして使用し、cfif タグまたは cfelseif タグで識別されなかったケースを扱います。

カテゴリ

フロー制御タグ

シンタックス

```
<cfif expression>
        HTML and CFML tags<cfelseif expression>
        HTML and CFML tags <cfelse>
        HTML and CFML tags</cfif>
```

関連項目

cfif、cfelse、cfabort、cfbreak、cfexecute、cfexit、cflocation、cfloop、cfswitch、cfthrow、cftry

使用方法

このタグ内の **expression** の値が yes で、このタグを含んでいる cfif タグ、および先行する cfelseif タグ内の **expressions** の値が no である場合には、このタグと後続の cfelseif または cfelse タグ、あるいは cfif 終了タグとの間にあるコードが処理されます。その後、cfif 終了タグの後ろにあるコードに進みます。そうでない場合は、コードがスキップされます。

このタグは、cfif タグブロックの内部で使用する必要があります。終了タグは必要ありません。

詳細と例については、330 ページの「cfif」を参照してください。

cferror

説明

エラーの発生時にカスタム HTML ページを表示します。この機能を使用すると、アプリケーションの機能ページとエラーページで外観を統一できます。

カテゴリ

例外処理タグ、拡張タグ、アプリケーションフレームワークタグ

シンタックス

```
<cferror
    template = "template path"
    type = "exception|validation|request"
    exception = "exception type"
    mailTo = "e-mail address">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfrethrow、cfthrow、cftry、『ColdFusion アプリケーションの開発』の Handling Errors

履歴

ColdFusion MX: exception 属性の monitor オプションが非推奨になりました。今後のリリースではこのタグは機能せずエラーを引き起こす可能性があります。

属性

属性	必須 / オプション	デフォルト	説明
template	必須		カスタムエラーページへの相対パスです。ColdFusion ページは、以前はテンプレートと呼ばれていました。
type	必須		カスタムエラーページが処理するエラータイプです。このタイプにより、ColdFusion がエラーページをどのように処理するかが決まります。詳細については、『ColdFusion アプリケーションの開発』の Specifying a custom error page を参照してください。 • exception: exception 属性で指定した例外タイプです。 • validation: サーバーサイドのタイプ検証で認識されたエラーです。 • request: 遭遇したエラーです。
exception	オプション	any	このタグが扱う例外のタイプです。 • application: アプリケーションの例外 • database: データベースの例外 • template: ColdFusion ページの例外 • security: セキュリティの例外 • object: オブジェクトの例外 • missingInclude: インクルードファイル欠如の例外 • expression: 式の例外 • lock: ロックの例外 • *custom_type*: cfthrow タグで定義される、開発者によって定義された例外 • any: すべての例外タイプ 例外タイプの詳細については、cftry を参照してください。
mailTo	オプション		電子メールアドレスです。この属性は、エラーページ上で変数 error.mailto として使用できます。ColdFusion がこのアドレスに何かを自動的に送信することはありません。

使用方法

このタグは、アプリケーション内のページにカスタムエラーメッセージを提供するために使われます。これにより、エラーが発生したときでもアプリケーションの外観を一貫させることができます。

通常は、このタグを "Application.cfc" または "Application.cfm" ファイルに埋め込んで、アプリケーション全体のエラー処理責任を指定します。type="validation" を指定した場合は、これらのファイルのいずれかにこのタグを埋め込む**必要があります**。ColdFusion は、他のページではこのタグを無視します。

cftry タグおよび cfcatch タグを使用すると、cferror タグの場合に比べてより対話的に ColdFusion ページ内の ColdFusion エラーを処理できます。ただし、一般的なエラーに対しては cferror タグで十分対処できます。

エラーページを正しく表示するには、cferror タグを含むページのエンコードに cfencode ユーティリティを使用しないようにしてください。

ページタイプ

次の表に、指定できるエラーのタイプと、そのエラータイプを扱うページ上で使用できるコードを示します。

ページタイプ	説明	用途
Exception	未処理の例外が検出されたときに、CFML 言語プロセッサによってダイナミックに呼び出されます。 すべての CFML タグを使用します。エラー変数は cfoutput タグ内に置く必要があります。	特定の例外タイプを扱うか、例外についての全般的な情報を表示することができます。
Request	「エラー変数」セクションで説明するエラー変数が含まれます。 CFML タグを含めることはできませんが、エラー変数をシャープ記号 (#) で囲むと、エラー変数の値を表示することができます (例 : #error.MailTo#)。	例外タイプなどの他のエラー処理メソッドに対するバックアップエラーハンドラとして使用します。
Validation	hidden フォームフィールドの検証または onSubmit 検証を行うフォームの送信時に発生したデータ入力検証エラーを処理します。 CFML タグを含めることはできませんが、エラー変数をシャープ記号 (#) で囲むと、エラー変数の値を表示することができます (例 : #Error.InvalidFields#)。 "Application.cfc" または "Application.cfm" ファイルで検証エラーハンドラを指定します。	hidden フォームフィールドまたは onSubmit の形式検証エラーだけを処理します。

エラー変数

cferror タグの template 属性で指定される例外処理ページには、エラー変数が含まれます。エラーを表示するときに、エラー変数の値が置換されます。

次の表に、エラー変数を示します。

ページタイプ	エラー変数	説明
Validation のみ	error.validationHeader	検証メッセージのヘッダのテキストです。
	error.invalidFields	検証エラーの順不同のリストです。
	error.validationFooter	検証メッセージのフッタのテキストです。
Request と Exception	error.diagnostics	ColdFusion からの詳細なエラー診断です。
	error.mailTo	電子メールアドレスです。cferror.MailTo の値と同じです。
	error.dateTime	エラーが発生した日付と時刻です。
	error.browser	エラーの発生時に動作していたブラウザです。
	error.remoteAddress	リモートクライアントの IP アドレスです。
	error.HTTPReferer	エラーが発生したページへのリンクに、クライアントがアクセスしたページです。
	error.template	エラーの発生時に実行していたページです。
	error.generatedContent	エラー発生時点までにページが生成していたコンテンツです。
	error.queryString	クライアントリクエストの URL クエリー文字列です。

ページタイプ	エラー変数	説明
Exception のみ	error.message	例外に関連付けられているエラーメッセージです。
	error.rootCause	例外の根本原因です。この構造体には、cfcatch タグによって返された情報が含まれます。たとえば、データベース例外の場合、エラーを発生させた SQL ステートメントは error.RootCause.Sql 変数内にあります。Java 例外の場合、この変数には、例外の " 根本原因 " の原因として JVM によってレポートされる Java サーブレットの例外が含まれています。
	error.tagContext	タグスタック内の各タグの情報を含んでいる構造体の配列です。このタグスタックは、現在開かれているタグから成ります。
	error.type	例外のタイプです。

注意：type = "exception" の場合には、Error の代わりに接頭辞 cferror を使用することができます。たとえば、cferror.diagnostics、cferror.mailTo、cferror.dateTime などです。

例

```
 <h3>cferror Example</h3>

<!--- Example of cferror call within a page.
         NOTE: If you use cferror type="VALIDATION" you MUST put it in
         Application.cfc or Application.cfm --->
<cferror type = "REQUEST"
template = "request_err.cfm"
mailTo = "admin@mywebsite.com">
<!--- This query calls a non-existent datasource, triggering an error to be handled. --->
<cfquery name="testQuery" datasource="doesNotExist">
select * from nothing
</cfquery>

<!--- Example of the page (request_err.cfm) to handle this error. --->
<html>
<head>
<title>We're sorry -- An Error Occurred</title>
</head>
<body>
<h2>We're sorry -- An Error Occurred</h2>
<p>
If you continue to have this problem, please contact #error.mailTo#
with the following information:</p>
<p>
<ul>
<li><b>Your Location:</b> #error.remoteAddress#
<li><b>Your Browser:</b> #error.browser#
<li><b>Date and Time the Error Occurred:</b> #error.dateTime#
<li><b>Page You Came From:</b> #error.HTTPReferer#
<li><b>Message Content</b>:
<p>#error.diagnostics#</p>
</ul>
```

cfexchangecalendar

説明

Microsoft Exchange のカレンダーイベントの作成、削除、変更、取得、およびカレンダーイベントへの応答を行います。また、カレンダーイベントの添付ファイルを取得します。

履歴

ColdFusion 10：getUserAvailability、getRooms、getRoomsList

- serverVersion 属性が追加されました。

ColdFusion 8: このタグが追加されました。

カテゴリ

通信タグ

シンタックス

```
 create
<cfexchangecalendar
    required
    action = "create"
    event = "#event information structure#"
    optional
    connection = "connection ID"
    result = "variable for event UID">

delete
<cfexchangecalendar
    required
    action = "delete"
    uid = "event UID,event UID, ..."
    optional
    connection = "connection ID"
    message = "string"
    notify = "yes|no">

deleteAttachments
<cfexchangecalendar
    required
    action = "deleteAttachments"
    uid = "event UID"
    optional
    connection = "connection ID">

get
<cfexchangecalendar
    required
    action = "get"
    name = "query identifier"
    optional
    connection = "connection ID">

getAttachments
<cfexchangecalendar
    required
    action = "getAttachments"
    name = "query identifier"
    uid = "event UID"
    optional
    attachmentPath = "directory path"
    connection = "connection ID">
    generateUniqueFilenames = "no|yes"
getRooms
<cfexchangecalendar
    action = "getRooms"
    emailAddress = "e-mail_ address"
    name = "name"
```

```
    connection = "connection_ID"/>
getRoomsList
<cfexchangecalendar
    action = "getRoomList"
    name = "name"
    connection = "connection_ID"/>
getUserAvailability
<cfexchangecalendar
    action = "getUserAvailability"
    attendees = "attendee_list"
    connection = "connection_ID"
    startDate = "date"
    endDate = "date"
    dataRequestType = "freeBusy|suggestions|freeBusyandSuggestions"
    name = "name" />
modify
<cfexchangecalendar
    required
    action = "modify"
    event = "#event information structure#"
    uid = "event UID"
    optional
    connection = "connection ID">

respond
<cfexchangecalendar
    required
    action = "respond"
    responseType = "accept|decline|tentative"
    uid = "event UID"
    optional
    connection = "connection ID"
    message = "string">
    notify = "yes|no">
```

注意：いずれのアクションに関しても、connection 属性を指定しない場合に使用するその他の属性については、cfexchangeconnection を参照してください。connection 属性を省略する場合は、cfexchangecalendar タグで cfexchangeconnection タグの属性を指定して一時的接続を作成します。この場合、ColdFusion はタグが完了すると接続を閉じます。詳細については、cfexchangeconnection タグの open アクションを参照してください。

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfexchangeconnection、cfexchangecontact、cfexchangefilter、cfexchangemail、cfexchangetask、『ColdFusion アプリケーションの開発』の Working with meetings and appointments

属性

属性	アクション	必須 / オプション	デフォルト	説明
action	N/A	必須		実行するアクションです。有効な値は次のとおりです。 • create • delete • deleteAttachments • get • getAttachments • getRooms • getRoomsList • getUserAvailability • modify • respond
attachmentPath	getAttachments	オプション		添付ファイルを保存する、ディスク上またはメモリ内のディレクトリのファイルパスです。指定したディスク上のディレクトリが存在しない場合は、自動的に作成されます。 **メモ :** この属性を省略すると、添付ファイルは保存されません。相対パスを指定する場合は、GetTempDirectory 関数によって返される ColdFusion テンポラリディレクトリがパスのルートになります。
attendees	getUserAvailability	必須		すべての出席者のカンマ区切りリストです。
connection	すべて	オプション		cfexchangeconnection タグで指定された Exchange サーバーへの接続の名前です。 この属性を省略する場合は、cfexchangecalendar タグで cfexchangeconnection タグの接続属性を指定して一時的接続を作成する必要があります。
dataRequestType	getUserAvailability	必須		• freeBusy：可用性の詳細の配列を返します。 • Suggestions：候補の詳細が含まれる構造体の配列を返します。 • freeBusyandSuggestions：suggestions の配列と attendeeavailability の配列の両方を返します。 詳しくは、suggestion の構造体および attendeeavailability の構造体の節を参照してください。
emailAddress	getRooms	オプション		メールボックスユーザーの SMTP（Simple Mail Transfer Protocol）アドレスを定義します。
endDate	getUserAvailability	必須		ColdFusion が日付時刻値として解釈できる文字列です。
event	create modify	必須		設定または変更するイベントプロパティとその値を含む構造体への参照です。この属性はシャープ記号 (#) で囲んで指定する必要があります。event 属性でもカテゴリキーがサポートされます。 イベント構造体の詳細については、「使用方法」を参照してください。

属性	アクション	必須 / オプション	デフォルト	説明
getOccurrence	True False	オプション		True の場合、指定された startDate 値から EndDate 値までの繰り返しイベントのすべての発生と、さらに単独のイベントも取得します。getOccurrence を True に指定した場合、cfExchangeFilter タグは使用できません。
generateUniqueFilenames	getAttachments	オプション	no	同じ名前の添付ファイルが複数ある場合に、一意のファイル名を生成するかどうかを指定するブール値です。同じ名前を持つ添付ファイルが複数存在し、このオプションが yes の場合、ColdFusion は、競合するファイル名の後ろ (拡張子の前) に番号を付加します。たとえば、myfile.txt という名前の添付ファイルが 3 つある場合は、myfile.txt、myfile1.txt、myfile2.txt という名前で保存されます。
message	delete respond	オプション		応答通知または削除通知で送信するオプションメッセージのテキストです。
name	getAttachments getUserAvailability getRoomsList getRooms	必須		取得したイベント、または取得した添付ファイルに関して取得した情報を格納する ColdFusion クエリー変数の名前です。返されるデータの詳細については、「使用方法」を参照してください。
notify	delete respond	オプション	true	イベントが修正されたときに他のユーザーに通知するかどうかを指定するブール値です。
responseType	respond	必須		有効な値は次のとおりです。 • accept • decline • tentative
result	create	オプション		作成するイベントの UID を格納する変数の名前です。create 以外のアクションでは、uid 属性で UID 値を指定してアクションの対象となるイベントを識別します。
serverVersion		オプション	2007	Microsoft Exchange Server のバージョンを指定します。 有効な値は次のとおりです。 • 2003 • 2007 • 2010 詳細を指定しない場合は、デフォルトで 2007 が使用されます。 指定した値は、アプリケーションレベルで指定した値よりも優先されます。
startDate	getUserAvailability	必須		ColdFusion が日付時刻値として解釈できる文字列です。
uid	delete getAttachments modify respond	必須		アクションの対象となるイベントを一意に識別する Exchange UID 値です。大文字と小文字は区別されます。 delete アクションの場合は、この属性で UID 値のカンマ区切りリストを指定できます。 deleteAttachments、getAttachments、modify、および respond アクションの場合は、1 つの UID 値のみを指定できます。

使用方法

cfexchangecalendar タグは、Exchange サーバー上のカレンダーイベントを管理します。cfexchangecalendar を使用すると、次の操作を実行できます。

- 予定イベントまたは会議イベントを作成する。終日のイベントを作成できます。
- イベントを削除する。
- 件名、送信者 ID、受信者 ID、受信時刻などのフィルタ条件に一致するイベントを取得する。
- 特定のイベントの添付ファイルを取得する。
- 既存のイベントを修正する。
- イベントに応答する。

このタグを使用するには、Exchange サーバーに接続する必要があります。複数の連絡先レコードを作成する場合のように、Exchange サーバーと対話するタグを複数使用する場合は、cfexchangeconnection タグを使用して継続的接続を作成します。その場合は、個々の cfexchangecalendar タグで接続識別子を指定します (タスク、連絡先、メールなどにもアクセスする場合はその他の ColdFusion Exchange タグでも接続識別子を指定します)。これにより、個々のタグに対して接続を作成して閉じる必要がなくなるので、システムの負荷が減少します。

また、ColdFusion が 1 個の cfexchangecalendar タグを処理する間だけ存続する一時的接続を作成することもできます。その場合は、cfexchangecontact タグで接続属性を直接指定します。接続属性の詳細については、cfexchangeconnection タグを参照してください。

注意：Exchange カレンダーの予定を作成するには、カレンダーイベントを作成し、必須またはオプションのいずれの参加者も指定しません。

create アクション

create アクションを指定する場合は、イベントに関する情報を含む構造体を event 属性で指定する必要があります。この構造体には次のエントリを格納できます。

要素	デフォルト	説明
AllDayEvent	no	終日のイベントかどうかを示すブール値です。
Attachments		添付ファイルとして送信するファイルのパスです。複数のファイルパスを指定する場合、Windows ではセミコロン (;)、UNIX および Linux ではコロン (:) を使用して各パスを区切ります。添付ファイルは絶対パスで指定する必要があります。 modify アクションで添付ファイルを指定した場合は、指定した添付ファイルが既存の添付ファイルに追加されます。既存の添付ファイルが削除されることはありません。
Categories		カテゴリのカンマ区切りリストです。リスト内のすべてのカテゴリと一致するイベントが検索されます。
Duration		イベントの継続時間です (単位 : 分)。
EndTime		ColdFusion で有効な日付時刻形式で示されるイベントの終了時刻です。
Importance	normal	次のいずれかの値になります。 • high • normal • low
IsRecurring		そのイベントが反復されるかどうかを示すブール値です。yes の場合は、RecurrenceType 要素と、反復の詳細を示す要素を指定します。反復フィールドの詳細については、次の表を参照してください。
Location		イベントの場所を示す文字列です。

要素	デフォルト	説明
Message		イベントに関するメッセージを含む文字列です。この文字列では HTML の形式設定タグも使用できます。
OptionalAttendees		メール ID のカンマ区切りリストです。
Organizer		会議の開催者の名前を示す文字列です。
Reminder		イベントの何分前にアラームメッセージを表示するかを指定します。
RequiredAttendees		メール ID のカンマ区切りリストです。
Resources		Exchange スケジュールリソース (会議室や機材など) のメール ID のカンマ区切りリストです。
Sensitivity		有効な値は、normal、company-confidential、personal、および private です。
StartTime		ColdFusion で有効な日付時刻形式で示されるイベントの開始時刻です。 この属性で日時を指定し、RecurrenceType で YEARLY を指定して、その他の反復属性を指定しなかった場合、そのイベントは 1 年に 1 回、この属性で指定した日時に繰り返されます。
Subject		イベントの件名を示す文字列です。

IsRecurring フィールドの値を yes に設定した場合にイベントの反復を指定するために使用する要素を次の表に示します。イベントの反復を指定する方法の詳細については、『ColdFusion アプリケーションの開発』の Specifying Calendar recurrence を参照してください。

要素	タイプ	デフォルト	説明
RecurrenceType	すべて	DAILY	構造体の IsRecurring 要素が yes の場合にのみ使用されます。有効な値は次のとおりです。 • DAILY • WEEKLY • MONTHLY • YEARLY
RecurrenceNoEndDate	すべて	yes	ブール値です。この要素が yes の場合、イベントは削除または変更されるまで反復されます。RecurrenceCount または RecurrenceEndDate とともに使用することはできません。
RecurrenceCount	すべて		イベントを反復する回数です。RecurrenceEndDate または RecurrenceNoEndDate とともに使用することはできません。
RecurrenceEndDate	すべて		最後に反復する日付です。RecurrenceCount または RecurrenceNoEndDate とともに使用することはできません。
RecurrenceFrequency	DAILY、WEEKLY、MONTHLY	1	反復する頻度です (日数、週数、または月数)。たとえば、反復のタイプが DAILY の場合、RecurrenceFrequency を 3 に設定すると 3 日おきにイベントがスケジュールされます。
RecurEveryWeekDay	DAILY		日曜日と土曜日を除く平日にイベントを反復します。RecurrenceFrequency とともに使用することはできません。
RecurrenceDays	WEEKLY		イベントが発生する曜日を 1 つまたは複数指定します。次の値をカンマ区切りリストで指定する必要があります。 MON、TUE、WED、THU、FRI、SAT、SUN 反復のタイプが WEEKLY の場合、このフィールドを省略すると、指定された開始日に対応する曜日にイベントが反復されます。

要素	タイプ	デフォルト	説明
RecurrenceDay	MONTHLY、YEARLY		何曜日にイベントを発生させるかを示します。有効な値は次のとおりです。 • MON • TUE • WED • THU • FRI • SAT • SUN
RecurrenceWeek	MONTHLY、YEARLY		月または年の第何週にイベントを繰り返すかを指定します。 使用できる値は次のとおりです。 • first • second • third • fourth • last
RecurrenceMonth	YEARLY		イベントを繰り返す月を指定します。有効な値は、JAN、FEB、MAR、APR、MAY、JUN、JUL、AUG、SEP、OCT、NOV、および DEC です。

delete アクション

delete アクションを指定する場合は、削除するイベントを識別する Exchange UID のカンマ区切りリストを uid 属性で指定します。UID 値を確認するには、適切なフィルタ式を指定した get アクションを使用します。

cfexchangecalendar タグで指定した UID がすべて無効な場合はエラーが発生します。1 つでも有効な UID が含まれる場合は、無効な UID が無視され、有効な UID に対応する項目が削除されます。

get アクション

get アクションを指定する場合は、子の cfexchangefilter タグを使用して取得するメッセージを指定します。フィルタの詳細については、cfexchangefilter を参照してください。

タグの処理が完了すると、name 属性で指定したクエリーオブジェクトに、取得したメッセージごとのレコードが格納されます。各レコードには次の列があります。

AllDayEvent	Duration	EndTime	From
HasAttachment	HtmlMessage	Importance	IsRecurring
Location	Message	OptionalAttendees	Organizer
Reminder	RequiredAttendees	Resources	Sensitivity
StartTime	Subject	UID	Categories

次の表で、From、HtmlMessage、Message、および UID フィールドについて説明します。他のフィールドの詳細については、create アクションの説明の表を参照してください。

列	説明
From	イベントを作成したユーザーの Exchange ID です。
HtmlMessage	イベントに関するメッセージの HTML バージョンです。
Message	イベントに関するメッセージのプレーンテキストバージョンです。
UID	メールイベントに割り当てられた一意の Exchange 識別子です。delete、getAttachments、および modify アクションでは、この値を使用してイベントを識別します。

getAttachments アクション

getAttachments アクションを使用する場合は、1 つの UID と name 属性を指定します。cfexchangecalendar タグにより、指定した名前を持つクエリーオブジェクトにデータが挿入されます。各レコードには、UID で指定したイベントの添付ファイルに関する次の情報が含まれます。

列	説明
attachmentFileName	添付ファイルのファイル名です。
attachmentFilePath	サーバー上の添付ファイルの絶対パスです。attachmentPath 属性を省略すると、この列には空の文字列が入ります。
CID	添付ファイルの content-ID です。通常は、メッセージにイメージを埋め込むために HTML の img タグ内で使用します。
mimeType	添付ファイルの MIME タイプです (text/html など)。
isMessage	添付ファイルがメッセージかどうかを指定するブール値です。
size	添付ファイルのサイズです (単位 : バイト)。

添付ファイルは attachmentPath 属性で指定したディレクトリに保存されます。attachmentPath 属性を省略すると、添付ファイル自体は取得されず、添付ファイルに関する情報が取得されます。この方法を使用すると、添付ファイルの取得に伴うオーバーヘッドを発生させることなく、イベントの添付ファイルを調べることができます。

次のシンタックスを使用して、メモリ内の attachmentPath ディレクトリを指定します。メモリ内のファイルはディスクに書き込まれないため、一時的データの処理が早くなります。

```
attachmentpath = "ram:///path"
```

パスには、ram:///petStore/orders/messageAttachments のように複数のディレクトリを含めることができます。ファイルを指定する前に、パスに含まれるすべてのディレクトリを作成しておく必要があります。メモリ内のファイルの使用方法について詳しくは、『ColdFusion アプリケーションの開発』の Working with in-memory files を参照してください。

getAttachments アクションは、Exchange のサーバー設定で EWS (Exchange Web サービス) の認証が basic に設定されている場合にのみ有効です。IWA (Integrated Windows Authentication) はサポートされていません。

modify アクション

modify アクションを指定する場合は、uid 属性でイベント UID を 1 つだけ指定して、変更するイベントを選択します。複数の UID は指定できません。event 構造体では、変更するフィールドのみを指定してください。フィールドの詳細と有効な値については、create アクションの表を参照してください。

イベントに添付ファイルがある場合、イベントを変更するときに添付ファイルを指定すると、新しい添付ファイルが既存の添付ファイルに追加されます。添付ファイルが置き換えられることはありません。添付ファイルを削除するには、deleteAttachments アクションを使用します。

respond アクション

cfexchangemail タグによって送信された会議通知に応答するには、respond アクションを使用します。メールメッセージに応答して要求を完全に承認するか暫定的に承認するまで、その会議はカレンダーに表示されず、cfexchangecalendar タグを使用して会議にアクセスすることはできません。

respond アクションを指定する場合は、通知メールメッセージに含まれるイベント UID を指定します。また、応答のタイプ (イベントの承認、拒否、または暫定的承認) も指定します。必要に応じて、応答に含めるメッセージを指定することもできます。また、イベントの作成者に応答を通知するかどうかを指定するフラグも設定できます。

respond アクションの使用方法の詳細については、『ColdFusion アプリケーションの開発』の Working with meeting notices and requests を参照してください。

suggestions 構造体の値

構造体の値	説明
date	会議の候補日です。
quality	候補日の品質で、Excellent、Good、Fair または Poor のいずれかです。
TimeSuggestion	次の値が含まれる構造体の配列です。 • MeetingDate：会議の候補時間です。 • Quality：時間の品質です。これは、Excellent、Good、Fair または Poor です。 • 重複の配列：候補時間の重複です。これは、次の値が含まれる構造体です。ConflictType（重複のタイプで、individualAttendeeConflict（出席者の重複）、GroupConflict（グループの 1 人以上のメンバーの重複）、GroupTooBigConflict（グループの 1 人以上のメンバーの重複であるが、詳細情報を返すにはグループが大きすぎた場合）、および UnknownAttendeeConflict（解決不可能な出席者、あるいはユーザー、グループまたは連絡先でない出席者の重複）のいずれかです）。 • FreeBusyStatus：重複している出席者の空き時間情報ステータスを取得します。ConflictType が IndividualAttendee に等しい場合にのみ意味があります。値は、Free、Tentative、Busy、OOF（予定に関連付けられている時間帯は「不在」と表示されている）、または NoData（予定に空き時間情報ステータスが関連付けられていない）です。 • NoOfMembers：重複しているグループのユーザー、リソースおよび会議室の数を取得します。ConflictType が ConflictType の GroupConflict に等しい場合にのみ意味があります。 • NoOfMembersAvailable：重複しているグループ内の出席可能なメンバー（ステータスが Free のメンバー）の数を取得します。ConflictType が ConflictType の GroupConflict に等しい場合にのみ意味があります。 • NoOfMembersWithConflict：重複しているグループ内の重複状態のメンバー（ステータスが Busy、OOF または Tentative のメンバー）の数を取得します。ConflictType が ConflictType の GroupConflict に等しい場合にのみ意味があります。 • NoOfMembersWithNoData：重複しているグループ内の空き時間情報データを公開していないメンバーの数を取得します。ConflictType が ConflictType の GroupConflict に等しい場合にのみ意味があります。 • isWorkTime：候補の会議が勤務時間内に実施されるかどうかです。

attendeeavailability 構造体の値

構造体の値	説明
CalendarEvent	次の値が含まれる構造体です。 • startTime：イベントの開始日時です。 • endTime：イベントの終了日時です。 • freeBusyStatus：イベントに関連付けられている空き時間情報ステータスです。値は、Free、tentative、busy、OOF（不在）、または NoData（予定に空き時間情報ステータスが関連付けられていない）のいずれかです。 • details：予定表イベントの詳細です。詳細を要求したユーザーが適切な権限を持っていない場合、details は null です。details は、次の値が含まれる構造体です。location（予定表の場所）、eventstoreId（予定表イベントのストア ID）、および Subject（isException（予定表イベントが、定期的なアイテムの例外であるかどうかを示すブール値）、isMeeting（予定表イベントが会議であるかどうかを示すブール値）、isPrivate（予定表イベントがプライベートであるかどうかを示すブール値）、isRecurring（予定表イベントが定期的なものであるかどうかを示すブール値）、および isRemainderSet（予定表イベントにアラームが設定されているかどうかを示すブール値）です）。

構造体の値	説明
mergedFreeBusyStatus	ステータスが含まれる構造体の配列です。ステータスは次のいずれかです。 • Free：予定に関連付けられている時間帯は「空き時間」と表示されています。 • Tentative：予定に関連付けられている時間帯は「仮承諾」と表示されています。 • Busy：予定に関連付けられている時間帯は「予定あり」と表示されています。 • OOF：予定に関連付けられている時間帯は「不在」と表示されています。 • NoData：予定に空き時間情報ステータスが関連付けられていません。 • result：応答に関連付けられた結果です。これは、success、warning または error です。
viewType	取得される、出席者の空き時間情報のビュータイプです。次の値を指定できます。 • None：ビューは返されません。この値は、GetUserAvailability の呼び出しでは指定できません。 • MergedOnly：集計された空き時間情報ストリームです。あるフォレストのターゲットユーザーに可用性サービスが設定されていない場合、リクエスターの可用性サービスでは、ターゲットユーザーの空き時間情報は空き時間情報のパブリックフォルダーから取得されます。パブリックフォルダーには結合後の形式の空き時間情報のみが保管されているので、MergedOnly のみが使用可能な情報です。 • FreeBusy：従来のステータス情報です（free、busy、tentative および OOF）。これには、予定の開始時刻および終了時刻も含まれます。このビューでは、集計された空き時間情報ストリームではなく、個々の会議の開始時刻と終了時刻が提供されるので、従来の空き時間情報ビューよりも包括的です。 • FreeBusyMerged：FreeBusy のすべてのプロパティと、結合後の空き時間情報のストリームです。 • Detailed：従来のステータス情報です（free、busy、tentative および OOF、予定の開始時刻と終了時刻、および予定の各種プロパティ（subject、location、importance など））。この要求のビューでは、要求を実行したユーザーが持つ権限に応じた最大限の情報が返されます。結合後の空き時間情報のみが使用可能な場合は、Microsoft Exchange Server 2003 フォレストのユーザーが情報を要求したときと同様に、MergedOnly が返されます。そうでない場合は、FreeBusy または Detailed が返されます。 • DetailedMerged：Detailed のすべてのプロパティと、結合後の空き時間情報のストリームを示します。結合後の空き時間情報のみを使用可能な場合（例えば、メールボックスが存在するコンピューターで Exchange 2003 が稼働している場合）、MergedOnly が返されます。そうでない場合は、FreeBusyMerged または DetailedMerged が返されます。
workingHours	次の詳細が含まれる構造体です。 • startTime：イベントの開始日時です。 • endTime：イベントの終了日時です。 • daysOfTheWeek：構造体の配列です。出席者の出勤日です。Sunday、Monday、Tuesday、Wednesday、Thursday、Friday、Saturday、Weekday および WeekEndDayDay の値が含まれることがあります。 • Timezone：id（タイムゾーン定義の ID）と name（そのタイムゾーン定義の名前）のフィールドが含まれる構造体です。

Exchange UID 値

すべての cfexchangecalendar アクションにおいて uid 属性の値は次のとおりです。

- exchangeServerVersion が 2003 または 2007 に設定されている場合：uid は、開催者のメールボックス内の予定の ID を示します。
- exchangeServerVersion が 2010 に設定されている場合：uid は、出席者のメールボックス内の受信した予定の ID を示します。

Microsoft Exchange Server 2003 または 2007 との対話では、予定が作成されるとすぐに、出席者はすべての操作（応答、添付の削除や取得など）に開催者の UID を使用できます。Microsoft Exchange Server 2010 の場合、この動作は異なります。出席者が予定関連のアクションを実行する必要がある場合は、最初に自分のメールボックス内の予定を検索し、それからその予定の UID を使用する必要があります。

例 1

次の例では、カレンダーイベントを作成してから変更します。最初にフォームを送信すると、カレンダーイベントが作成され、入力したデータとともにフォームが再表示されます。フォームを変更して再送信するには、イベントを承認します。2 回目にフォームを送信するときには、変更された情報が送信されます。詳細については、『ColdFusion アプリケーションの開発』の Working with meetings and appointments を参照してください。

この例ではコードを短くするためにすべてのイベントデータが再送信されますが、変更したデータだけを送信するように記述することもできます。

```
 <!--- Create a structure to hold the event information. --->
<!--- A self-submitting form for the event information --->
<!--- This example omits recurrence to keep the code relatively simple --->
<cfparam name="form.eventID" default="0">

<!--- If the form was submitted, populate the event structure from it. --->
<cfif isDefined("Form.Submit")>
        <cfscript>
        sEvent.AllDayEvent="no";
        sEvent=StructNew();
        sEvent.Subject=Form.subject;
        if (IsDefined("Form.allDay")) {
            sEvent.AllDayEvent="yes";
            sEvent.StartTime=createDateTime(Year(Form.date), Month(Form.date),
                Day(Form.date), 8, 0, 0);
        }
        else {
            sEvent.StartTime=createDateTime(Year(Form.date), Month(Form.date),
                Day(Form.date), Hour(Form.startTime), Minute(Form.startTime), 0);
            sEvent.EndTime=createDateTime(Year(Form.date), Month(Form.date),
                Day(Form.date), Hour(Form.endTime), Minute(Form.endTime), 0);
        }
        sEvent.Location=Form.location;
        sEvent.RequiredAttendees=Form.requiredAttendees;
        sEvent.OptionalAttendees=Form.optionalAttendees;
        //sEvent.Resources=Form.resources;
        if (Form.reminder NEQ "") {
            sEvent.Reminder=Form.reminder;
        }
        else {
            sEvent.Reminder=0;
        }
        sEvent.Importance=Form.importance;
        sEvent.Sensitivity=Form.sensitivity;
        sEvent.message=Form.Message;
    </cfscript>

    <!--- If this is the first time the form is being submitted
            Create a new event. --->
    <cfif form.eventID EQ 0>
    <!--- Create the event in Exchange --->
        <cfexchangecalendar action="create"
            username ="#user1#"
            password="#password1#"
            server="#exchangeServerIP#"
            event="#sEvent#"
```

```
            result="theUID">
        <!--- Output the UID of the new event. --->
        <cfif isDefined("theUID")>
            <cfoutput>Event Added. UID is#theUID#</cfoutput>
            <cfset Form.eventID = theUID >
        </cfif>
    <cfelse>
    <!--- The form is being resubmitted with new data, so update the event. --->
        <cfexchangecalendar action="modify"
            username ="#user1#"
            password="#password1#"
            server="#exchangeServerIP#"
            event="#sEvent#"
            uid="#Form.eventID#">
        <cfoutput>Event ID #Form.eventID# Updated.</cfoutput>

    </cfif>
</cfif>

<cfform format="xml" preservedata="yes" style="width:500" height="600">
    <cfinput type="text" label="Subject" name="subject" style="width:435"><br />
    <cfinput type="checkbox" label="All Day Event" name="allDay">
    <cfinput type="datefield" label="Date" name="date" validate="date" style="width:100">
    <cfinput type="text" label="Start Time" name="startTime" validate="time"
        style="width:100">
    <cfinput type="text" label="End Time" name="endTime" validate="time"
        style="width:100"><br />
    <cfinput type="text" label="Location" name="location" style="width:435"><br />
    <cfinput type="text" label="Required Attendees" name="requiredAttendees"
        style="width:435"><br />
    <cfinput type="text" label="Optional Attendees" name="optionalAttendees"
        style="width:435"><br />
    <cfinput type="text" label="Resources" name="resources" style="width:435"><br />
    <cfinput type="text" label="Reminder (minutes)" validate="integer" name="reminder"
        style="width:200">
    <cfselect name="importance" label="Importance" style="width:100">
        <option value="normal">Normal</option>
        <option value="high">High</option>
        <option value="low">Low</option>
    </cfselect>
    <cfselect name="sensitivity" label="Sensitivity" style="width:100">
        <option value="normal">Normal</option>
        <option value="company-confidential">Confidential</option>
        <option value="personal">Personal</option>
        <option value="private">Private</option>
    </cfselect>
    <cfinput type="textarea" label="Message" name="message" style="width:435;
        height:100">
    <cfinput type="hidden" name="eventID" value="#Form.EventID#">
    <cfinput type="Submit" name="submit" value="Submit">
</cfform>
```

例 2

次の例は、getUserAvailability アクションを実行する方法を示しています。

Application.cfm

```
<!--- Setting the present Date --->
<cfset todayDate = #Now()#>
<cfset fromDate = #DateFormat(todayDate, "mm/dd/yyyy")# & ' ' & #TimeFormat(todayDate, "HH:MM:SS")#>
<!--- Adding one day to present date--->
<cfset toDate1 = DateAdd("d", "1", "#fromDate#")>
<!---     <cfset toDate = #DateFormat(toDate, "mm/dd/yyyy")#&' '&#TimeFormat(toDate,
"HH:MM:SS")#>--->
<cfset toDate1 = #DateFormat(toDate1, "mm/dd/yyyy")# & ' ' & #TimeFormat(toDate1, "HH:MM:SS")#>
<!---Creating a calendar event --->
<cffunction name="constructCalendarStruct" returntype="struct">
    <cfargument name="AllDayEvent" type="string"/>
    <cfargument name="Duartion" type="string"/>
    <cfargument name="EndTime" type="string"/>
    <cfargument name="From" type="string"/>
    <cfargument name="HasAttachment" type="string"/>
    <cfargument name="HtmlMessage" type="string"/>
    <cfargument name="Importance" type="string"/>
    <cfargument name="IsRecurring" type="string"/>
    <cfargument name="Location" type="string"/>
    <cfargument name="Message" type="string"/>
    <cfargument name="OptionalAttendees" type="string"/>
    <cfargument name="Organizer" type="string"/>
    <cfargument name="Reminder" type="string"/>
    <cfargument name="RequiredAttendees" type="string"/>
    <cfargument name="Resources" type="string"/>
    <cfargument name="Sensitivity" type="string"/>
    <cfargument name="StartTime" type="string"/>
    <cfargument name="Subject" type="string"/>
    <cfargument name="UID" type="string"/>
    <cfscript>
        var eventInfo = structNew();
        if(isdefined("AllDayEvent") EQ 1)
            eventInfo.AllDayEvent = AllDayEvent;
        if(isdefined("Duration") EQ 1)
            eventInfo.Duration = Duration;
        if(isdefined("EndTime") EQ 1)
            eventInfo.EndTime = EndTime;
        if(isdefined("From") EQ 1)
            eventInfo.From = From;
        if(isdefined("HasAttachment") EQ 1)
            eventInfo.HasAttachment = HasAttachment;
        if(isdefined("HtmlMessage") EQ 1)
            eventInfo.HtmlMessage = HtmlMessage;
        if(isdefined("Importance") EQ 1)
            eventInfo.Importance = Importance;
        if(isdefined("IsRecurring") EQ 1)
            eventInfo.IsRecurring = IsRecurring;
        if(isdefined("Message") EQ 1)
            eventInfo.Message = Message;
        if(isdefined("OptionalAttendees") EQ 1)
            eventInfo.OptionalAttendees = OptionalAttendees;
        if(isdefined("Organizer") EQ 1)
            eventInfo.Organizer = Organizer;
        if(isdefined("Reminder") EQ 1)
            eventInfo.Reminder = Reminder;
        if(isdefined("RequiredAttendees") EQ 1)
            eventInfo.RequiredAttendees = RequiredAttendees;
        if(isdefined("Resources") EQ 1)
            eventInfo.Resources = Resources;
        if(isdefined("Sensitivity") EQ 1)
            eventInfo.Sensitivity = Sensitivity;
```

```
        if(isdefined("StartTime") EQ 1)
            eventInfo.StartTime = StartTime;
        if(isdefined("Subject") EQ 1)
            eventInfo.Subject = Subject;
        if(isdefined("UID") EQ 1)
            eventInfo.UID = UID;
        if(isdefined("Location") EQ 1)
            eventInfo.Location = Location;
    </cfscript>
    <cfreturn eventInfo>
</cffunction>
<!--- Function to delete calendar events --->
<cffunction name="deleteAllEvents">
    <cfargument name="convariable" type="any"/>
    <cfexchangecalendar action="get" name="deleteQuery" connection="convariable">
    </cfexchangecalendar>
    <cfloop query="deleteQuery">
        <cfexchangecalendar action="delete" connection=convariable uid=#deleteQuery.uid#>
    </cfloop>
</cffunction>
```

Availability.cfm

```
<cfexchangeConnection
    action="open"
    username = "user1"
    password="Password"
    server="IP_Address"
    serverversion="2010"
    connection="conn1">
<cfexchangeConnection
    action="open"
    username = "user2"
    password="Password"
    server="IP_Address"
    serverversion="2010"
    connection="conn2">
<cfset deleteAllEvents(conn1)>
<cfset deleteAllEvents(conn2)>
<!--- Creating All day event for user1 --->
<cfset eventInfo =
constructCalendarStruct(AllDayEvent="yes",Importance="High",Subject="Testplan1",StartTime="#fromDate#")>
<cfset eventInfo1 =
constructCalendarStruct(AllDayEvent="yes",Importance="High",Subject="Testplan",StartTime="#fromDate#")>
<cfexchangeCalendar
action="create"
connection="conn1"
```

```
event="#eventInfo#"
result="UID">
<cfscript>sleep(15000);</cfscript>
<cfexchangeCalendar
action="create"
connection="conn2"
event="#eventInfo1#"
result="UID1">
<cfscript>sleep(15000);</cfscript>
    <cfexchangeCalendar
    attendees="a@cfadobe.com"
    action="getuseravailability"
    connection="conn1"
    datarequesttype="freebusyandsuggestions"
    startDate="#fromDate#"
    endDate="#toDate1#"
    name="result">
    <cfdump var="#result#">
<cfset deleteAllEvents(conn1)>
<cfset deleteAllEvents(conn2)>
```

次の例は、getRooms および getRoomList アクションを使用する方法を示しています。

```
<cfexchangeConnection
    action="open"
    username = "sample"
    password="Password"
    server="IP_Address"
    serverversion="2010"
    connection="conn1">
<cfexchangecalendar action="getroomlist" name="roomList" connection="conn1">
<cfdump var="#roomList#">
<cfexchangecalendar action="getrooms" emailaddress="groundfloor1@cfadobe.com" name="rooms"
connection="conn1">
<cfdump var="#rooms#">
```

cfexchangeconnection

説明

Microsoft Exchange サーバーへの継続的接続を開きます (または閉じます)。また、メールボックスのサブフォルダに関する情報を取得する場合にも使用します。cfexchangecalendar タグ、cfexchangecontact タグ、cfexchangemail タグ、および cfexchangetask タグを使用するには、継続的接続または一時的接続を開く必要があります。

履歴

ColdFusion 10：serverVersion 属性が追加されました。

ColdFusion 8: このタグが追加されました。

カテゴリ

通信タグ

シンタックス

```
 open
<cfexchangeconnection
    required
    action = "open"
    connection = "connection ID">
    server = "Exchange server ID"
    username = "Exchange user ID">
    optional
    ExchangeApplicationName = "Application name"
    ExchangeServerLanguage = "Language name"
    formBasedAuthentication = "no|yes">
    formBasedAuthenticationURL = "URL">
    mailboxName = "Exchange mailbox">
    password = "user password"
    port = "IP port"
    protocol = "http|https"
    proxyHost = "proxy host URL"
    proxyPort = "proxy IP port"

getSubfolders
<cfexchangeconnection
    required
    action = "getSubfolders"
    connection = "connection ID">
    name = "query name"
    optional
    folder = "Exchange folder path">
    recurse = "no|yes">
OR
<cfexchangeconnection
    required
    action = "getSubfolders"
    name = "query name"
    server = "Exchange server ID"
    username = "Exchange user ID">
    optional
    ExchangeApplicationName = "Application name"
    ExchangeServerLanguage = "Language name"
    folder = "Exchange folder path">
    formBasedAuthentication = "no|yes">
    formBasedAuthenticationURL = "URL">
    mailboxName = "Exchange mailbox">
    password = "user password"
    port = "IP port"
    protocol = "http|https"
    proxyHost = "proxy host URL"
    proxyPort = "proxy IP port"
    recurse = "no|yes">

close
<cfexchangeconnection
    required
    action = "close"
    connection = "connection ID">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfexchangecalendar、cfexchangecontact、cfexchangefilter、cfexchangemail、cfexchangetask、『ColdFusion アプリケーションの開発』の Managing connections to the Exchange server

属性

属性	アクション	必須 / オプション	デフォルト	説明
action	すべて	必須		実行するアクションです。有効な値は次のとおりです。 • open: 指定した新しい継続的接続を開きます。 • close: 指定した接続を閉じます。 • getSubfolders: 特定のフォルダのサブフォルダに関する情報を取得します。
connection	すべて	open アクションと close アクションの場合は必須		接続の名前です。この ID は接続を必要とする任意のタグで指定できます。
ExchangeApplicationName	open getSubfolders	オプション	exchange	サーバーにアクセスする URL で使用する Exchange アプリケーションの名前。この属性は、IIS サーバーで Exchange アプリケーションにデフォルト名を使用していない場合に指定します。
ExchangeServerLanguage	open getSubfolders	オプション	english	Exchange サーバーの言語です。言語がわからない場合は、空の文字列を指定できます。english 以外のすべての値 (空の文字列を含む) の場合、このタグはクライアントのローカル言語でサーバーからフォルダ名を取得しようとします。サーバーに大量のデータがある場合などは、ローカル言語で Exchange サーバーからフォルダ名を取得するために、かなり時間がかかることがあります。
folder	getSubfolders	オプション	メールボックスのルート	メールボックスのルートから、サブフォルダを取得する親フォルダまでの、スラッシュ (/) 区切りのパスです。 フォルダ名にスラッシュが含まれる場合は、_xF8FF_ エスケープシーケンスを使用して名前を指定します。
formBasedAuthentication	open getSubfolders	オプション	no	接続時にログインフォームを表示して、フォームベースの認証を使用するかどうかを指定するブール値です。属性値が no (デフォルト) で、ColdFusion による接続試行時に Exchange サーバーが 440 エラーステータスを返す場合は、ColdFusion によってログインフォームが表示され、フォームベースの認証の使用が試みられます。したがって、サーバーでフォームベースの認証を必要とするかどうかわからない場合は、この属性を省略できます。
formBasedAuthenticationURL	open getSubfolders	オプション		Exchange サーバーでフォームベースの認証を使用している場合に、ユーザー ID とパスワードを送信する宛先の URL。この属性は、Exchange サーバーでフォームベースの認証用にデフォルトの URL を使用していない場合にのみ使用します。デフォルト URL の形式は、https://<Exchange サーバー >/exchweb/bin/auth/owaauth.dll です (例 : https://exchange.mycompany.com/exchweb/bin/auth/owaauth.dll)。

属性	アクション	必須 / オプション	デフォルト	説明
mailboxName	open getSubfolders	オプション		使用する Exchange メールボックスの ID です。username 属性で指定したアカウントにアクセス権を委任した所有者のメールボックスにアクセスするには、この属性を指定します。
name	getSubfolders	必須		サブフォルダに関する情報を格納する ColdFusion クエリー変数の名前です。
password	open getSubfolders	オプション		Exchange サーバーへのアクセスに使用するユーザーのパスワードです。
port	open getSubfolders	オプション	80	サーバーがリスンするポートです。通常はポート 80 を使用します。
protocol	open getSubfolders	オプション	http	接続に使用するプロトコルです。有効な値は、http および https です。
proxyHost	open getSubfolders	オプション		プロキシホストの URL または IP アドレスです (ネットワークへのアクセスに必要な場合にのみ指定します)。
proxyPort	open getSubfolders	オプション		接続するプロキシサーバーのポートです。通常はポート 80 を使用します。
recurse	getSubfolders	オプション	false	ブール値です。 • true: 指定したフォルダの直下にあるサブフォルダの情報のみを取得します。 • false: 指定したフォルダ以下にある全サブフォルダの情報を取得します。
server	open getSubfolders	必須		Exchange へのアクセスに使用するサーバーの IP アドレスまたは URL です。
serverVersion		オプション	2007	Microsoft Exchange Server のバージョンを指定します。 有効な値は次のとおりです。 • 2003 • 2007 • 2010 詳細を指定しない場合は、デフォルトで 2007 が使用されます。 指定した値は、アプリケーションレベルで指定した値よりも優先されます。
username	open getSubfolders	必須		Exchange のユーザー ID です。

注意：getSubfolders アクションを指定する場合は、connection 属性を指定しない場合に限り、open アクションと getSubfolders アクションの両方で機能する属性を指定できます。

使用方法

cfexchangeconnection タグを使用すると、Exchange サーバーとの継続的接続を開いたり閉じたりすることができます。cfexchangeconnection を使用して接続を開いた後に cfexchangecalendar、cfexchangecontact、cfexchangemail、または cfexchangetask タグを使用すると、1 つのタグごとに接続を作成する必要がなくなり、複数のタグを使用して Exchange サーバーと対話できます。

注意：接続を確立するには、Exchange サーバーによって Outlook Web Access が許可される必要があります。このアクセスを有効にする方法については、『ColdFusion アプリケーションの開発』の Enabling access to the Exchange server を参照してください。また、特別な認証手順を必要とする場合は、Exchange サーバーへの接続を確立できません。たとえば、VPN PIN が必要である場合や、ファイアウォールの外側にあるサーバーでバイオメトリック認証を実行し、認証サーバーでファイアウォールの内側にある Exchange サーバーにメッセージをルーティングするような場合です。

Exchange サーバーへのアクセスが終了したときに継続的接続を閉じるには、cfexchangeconnection タグを使用します。この接続は自動的にタイムアウトしないため、接続を閉じないと開いたままの状態になります。

1 つのタグを処理する間だけ存続する一時的接続を作成するには、cfexchangecalendar、cfexchangecontact、cfexchangemail、または cfexchangetask タグで (connection 属性ではなく) open アクションの接続属性を指定します。cfexchangeconnection タグを使用して接続を作成する必要はありません。この場合は、タグの処理が完了すると接続が自動的に閉じられます。

getSubfolders アクションを使用すると、指定したフォルダ (またはメールボックスのトップレベル) の直下にあるサブフォルダ、またはすべてのサブフォルダに関する情報を取得できます。サブフォルダを取得するには、継続的接続を使用する必要があります。

getSubfolders アクションによって返されるクエリーには、次の列が含まれます。

列	内容
FOLDERNAME	サブフォルダの名前です (例 : ColdFusion)。
FOLDERPATH	メールボックスのルートからサブフォルダまでのスラッシュ (/) 区切りのパスです。フォルダ名を含みます (例 : Inbox/Marketing/ColdFusion)。
FOLDERSIZE	フォルダのサイズです (単位 : バイト)。

注意：cfexchangeconnection などの ColdFusion exchange タグは、WebDAV を使用して Exchange サーバーに接続します。これらのタグを使用するには、Exchange サーバーで HTTP アクセスが有効になっている必要があります。

例

次の例では、接続を開いた後、spamsource.com から送信されたメールをすべて取得して、Exchange サーバーからそれらのメッセージを削除します。

```
<cfexchangeConnection
    action="open"
    username="#user1#"
    password="#password1#"
    server="#exchangeServerIP#"
    connection="testconn1">

<cfexchangemail action="get" name="spamMail" connection="testconn1">
    <cfexchangefilter name="fromID" value="spamsource.com">
</cfexchangemail>

<cfloop query="spamMail">
    <cfexchangeMail action="delete" connection="testconn1" uid="#spamMail.uid#">
</cfloop>

<cfexchangeConnection
    action="close"
    connection="testconn1">
```

cfexchangecontact

説明

Microsoft Exchange 連絡先レコードの作成、削除、変更、および取得を行い、連絡先レコードの添付ファイルを取得します。

履歴

ColdFusion 10：serverVersion 属性が追加されました。

ColdFusion 8: このタグが追加されました。

カテゴリ

通信タグ

シンタックス

```
 create
<cfexchangecontact
    required
    action = "create"
    contact = "#contact information structure#"
    optional
    connection = "connection ID"
    result = "variable for contact UID">

delete
<cfexchangecontact
    required
    action = "delete"
    uid = "contact UID,contact UID, ..."
    optional
    connection = "connection ID">

deleteAttachments
<cfexchangecontact
    required
    action = "deleteAttachments"
    uid = "contact UID"
    optional
    connection = "connection ID">

get
<cfexchangecontact
    required
    action = "get"
    name = "query identifier"
    optional
    connection = "connection ID">
```

```
getAttachments
<cfexchangecontact
    required
    action = "getAttachments"
    name = "query identifier"
    uid = "contact UID"
    optional
    attachmentPath = "directory path"
    connection = "connection ID"
    generateUniqueFilenames = "no|yes">

modify
<cfexchangecontact
    required
    action = "modify"
    contact = "#contact information structure#"
    uid = "contact UID"
    optional
    connection = "connection ID>"
```

注意：connection 属性を省略する場合は、cfexchangecontact タグで cfexchangeconnection タグの属性を指定して一時的接続を作成します。この場合、ColdFusion はタグが完了すると接続を閉じます。詳細については、cfexchangeconnection タグの open アクションを参照してください。

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfexchangecalendar、cfexchangeconnection、cfexchangefilter、cfexchangemail、cfexchangetask、『ColdFusion アプリケーションの開発』の Interacting with Microsoft Exchange Servers

属性

属性	アクション	必須 / オプション	デフォルト	説明
action	N/A	必須		実行するアクションです。有効な値は次のとおりです。 • create • delete • deleteAttachments • get • getAttachments • modify
attachmentPath	getAttachments	オプション		添付ファイルを保存するディレクトリの絶対ファイルパスです。指定したディレクトリが存在しない場合は、自動的に作成されます。 **メモ :** この属性を省略すると、添付ファイルは保存されません。
connection	すべて	オプション		cfexchangeconnection タグで指定された Exchange サーバーへの接続の名前です。 この属性を省略する場合は、cfexchangecontact タグで cfexchangeconnection タグの接続アクション属性 open を指定して一時的接続を作成します。
contact	create modify	必須		設定または変更する contact プロパティとその値を含む構造体への参照です。この属性はシャープ記号 (#) で囲んで指定する必要があります。 イベント構造体の詳細については、「使用方法」を参照してください。
generateUniqueFilenames	getAttachments	オプション	no	同じ名前の添付ファイルが複数ある場合に、一意のファイル名を生成するかどうかを指定するブール値です。同じ名前を持つ添付ファイルが複数存在し、このオプションが yes の場合、ColdFusion は、競合するファイル名の後ろ (拡張子の前) に番号を付加します。たとえば、myfile.txt という名前の添付ファイルが 3 つある場合は、myfile.txt、myfile1.txt、myfile2.txt という名前で保存されます。
name	get getAttachments	必須		返された連絡先レコード、または取得した添付ファイルに関する情報を格納する ColdFusion クエリー変数の名前です。返されるデータの詳細については、「使用方法」を参照してください。

属性	アクション	必須 / オプション	デフォルト	説明
result	create	オプション		作成する連絡先の UID を格納する変数の名前です。他のアクションでは、uid 属性でこの値を指定してアクションの対象となる連絡先を識別します。
serverVersion		オプション	2007	Microsoft Exchange Server のバージョンを指定します。 有効な値は次のとおりです。 • 2003 • 2007 • 2010 詳細を指定しない場合は、デフォルトで 2007 が使用されます。 指定した値は、アプリケーションレベルで指定した値よりも優先されます。
uid	getAttachments delete modify	必須		アクションの対象となる連絡先を一意に識別する Exchange UID 値です。大文字と小文字は区別されます。delete アクションの場合は、この属性で UID 値のカンマ区切りリストを指定できます。deleteAttachments、getAttachments、および modify アクションの場合は、1 つの UID 値のみを指定できます。

create または modify アクションを指定する場合は、連絡先に関する情報を含む構造体を contact 属性で指定する必要があります。この構造体には次の要素を格納できます。設定または変更する要素のみを含めます。

Assistant	Attachments	BusinessAddress	BusinessFax
BusinessPhoneNumber	Categories	Company	Department
説明	DisplayAs	Email1	Email2
Email3	FirstName	HomeAddress	HomePhoneNumber
JobTitle	LastName	MailingAddressType	Manager
MiddleName	MobilePhoneNumber	NickName	Office
OtherAddress	OtherPhoneNumber	Pager	Profession
SpouseName	WebPage		

BusinessAddress、HomeAddress、および OtherAddress 以外のフィールドにはテキストが入ります。これら 3 つの住所フィールドには、次のフィールドを含む構造体が入ります。

- Street
- City
- State
- Zip
- Country

Attachments フィールドでは、連絡先に含める添付ファイルのパス名を指定する必要があります。複数のファイルを指定する場合、Windows ではセミコロン (;)、UNIX および Linux ではコロン (:) を使用して各項目を区切ります。絶対パスを使用する必要があります。

modify アクションで添付ファイルを指定した場合は、指定した添付ファイルが既存の添付ファイルに追加されます。既存の添付ファイルが削除されることはありません。

Categories フィールドでは、連絡先のカテゴリのカンマ区切りリストを指定できます。

DisplayAs フィールドを指定しない場合、表示名は FirstName, LastName に設定されます。

使用方法

cfexchangecontact タグは、Exchange サーバー上の連絡先レコードを管理します。次のアクションを実行するには、cfexchangecontact タグを使用します。

- 連絡先を作成する。
- 連絡先を削除する。
- 姓、役職、自宅電話番号などのフィルタ条件に一致する連絡先レコードを取得する。
- 特定の連絡先レコードの添付ファイルを取得する。
- 既存の連絡先を修正する。

このタグを使用するには、Exchange サーバーに接続する必要があります。複数の連絡先レコードを作成する場合のように、Exchange サーバーと対話するタグを複数使用する場合は、cfexchangeconnection タグを使用して継続的接続を作成します。その場合は、個々の cfexchangecontact タグで接続識別子を指定します (タスク、連絡先、メールなどにもアクセスする場合はその他の ColdFusion Exchange タグでも接続識別子を指定します)。これにより、個々のタグに対して接続を作成して閉じる必要がなくなるので、システムの負荷が減少します。

また、ColdFusion が 1 個の cfexchangecontact タグを処理する間だけ存続する一時的接続を作成することもできます。その場合は、cfexchangecontact タグで接続属性を直接指定します。接続属性の詳細については、cfexchangeconnection タグの open アクションを参照してください。

attachmentPath 属性

次のシンタックスを使用して、メモリ内の attachmentPath ディレクトリを指定します。メモリ内のファイルはディスクに書き込まれないため、一時的データの処理が早くなります。

```
attachmentpath = "ram:///filepath"
```

パスには、ram:///petStore/orders/messageAttachments のように複数のディレクトリを含めることができます。ファイルを指定する前に、パスに含まれるすべてのディレクトリを作成しておく必要があります。メモリ内のファイルの使用方法について詳しくは、『ColdFusion アプリケーションの開発』の Working with in-memory files を参照してください。

delete アクション

delete アクションを指定する場合は、削除する連絡先を識別する Exchange UID のカンマ区切りリストを uid 属性で指定する必要があります。UID 値を確認するには、適切なフィルタ式を指定した get アクションを使用します。

cfexchangecontact タグで指定した UID がすべて無効な場合はエラーが発生します。1 つでも有効な UID が含まれる場合は、無効な UID が無視され、有効な UID に対応する項目が削除されます。

get アクション

get アクションを指定すると、name 属性で指定したクエリーオブジェクトに、取得した連絡先ごとのレコードが格納されます。クエリーオブジェクトには contact 属性の構造体と同じ名前およびデータ形式を持つフィールドが含まれます。ただし、次の点が異なります。

- クエリーオブジェクトにはブール型の HasAttachment 列が含まれ、Attachments 列は含まれません。HasAttachment フィールドが yes の場合は、getAttachments アクションを使用して添付ファイルを取得します。
- クエリーオブジェクトには Exchange サーバー内の連絡先レコードの UID を含む UID 列が追加されます。getAttachments、delete、および modify アクションでは、この uid 属性の値を使用してレコードを識別します。

- クエリーオブジェクトには HtmlDescription 列が追加されます。Description 列にはプレーンテキスト形式の説明が格納され、HtmlDescription 列には HTML 形式の説明が格納されます。

取得するメッセージを指定するには、子の cfexchangefilter タグを使用します。詳細については、cfexchangefilter を参照してください。

getAttachments アクション

getAttachments アクションを使用する場合は、1 つの UID と name 属性を指定します。cfexchangecontact タグにより、指定した名前を持つクエリーオブジェクトにデータが挿入されます。各レコードには、UID で指定した連絡先の添付ファイルに関する次の情報が含まれます。

列名	説明
attachmentFileName	添付ファイルのファイル名です。
attachmentFilePath	サーバー上の添付ファイルの絶対パスです。attachmentPath 属性を省略すると、この列には空の文字列が入ります。
CID	添付ファイルの content-ID です。通常は、メッセージにイメージを埋め込むために HTML の img タグ内で使用します。
mimeType	添付ファイルの MIME タイプです (text/html など)。
isMessage	添付ファイルがメッセージかどうかを指定するブール値です。
size	添付ファイルのサイズです (単位 : バイト)。

添付ファイルは attachmentPath 属性で指定したディレクトリに保存されます。attachmentPath 属性を省略すると、添付ファイル自体は取得されず、添付ファイルに関する情報が取得されます。この方法を使用すると、添付ファイルの取得に伴うオーバーヘッドを発生させることなく、添付ファイルを調べることができます。

次のシンタックスを使用して、メモリ内の attachmentPath ディレクトリを指定します。メモリ内のファイルはディスクに書き込まれないため、一時的データの処理が早くなります。

```
attachmentpath = "ram:///path"
```

パスには、ram:///petStore/orders/messageAttachments のように複数のディレクトリを含めることができます。ファイルを指定する前に、パスに含まれるすべてのディレクトリを作成しておく必要があります。メモリ内のファイルの使用方法について詳しくは、『ColdFusion アプリケーションの開発』の Working with in-memory files を参照してください。

getAttachments アクションは、Exchange のサーバー設定で EWS (Exchange Web サービス) の認証が basic に設定されている場合にのみ有効です。IWA (Integrated Windows Authentication) はサポートされていません。

modify アクション

modify アクションを指定する場合は、uid 属性で 1 つの Exchange UID を指定する必要があります。contact 構造体では、変更するフィールドのみを指定する必要があります。指定しないフィールドは変更されません。

連絡先に添付ファイルがある場合、連絡先を修正するときに添付ファイルを指定すると、新しい添付ファイルが既存の添付ファイルに追加されます。添付ファイルが置き換えられることはありません。添付ファイルを削除するには、deleteAttachments アクションを使用します。

例

この例では、ユーザーがフォームに情報を入力し、その情報を使用して Exchange サーバーに連絡先を作成します。

```
 <!--- Create a structure to hold the contact information. --->
<cfset sContact="#StructNew()#">

<!--- A self-submitting form for the contact information --->
<cfform format="flash" width="550" height="460">
    <cfformitem type="html"><b>Name</b></cfformitem>
    <cfformgroup type="horizontal" label="">
        <cfinput type="text" label="First" name="firstName" width="200">
        <cfinput type="text" label="Last" name="lastName" width="200">
    </cfformgroup>
    <cfformgroup type="VBox">
        <cfformitem type="html"><b>Address</b></cfformitem>
        <cfinput type="text" label="Company" name="Company" width="435">
        <cfinput type="text" label="Street" name="street" width="435">
        <cfinput type="text" label="City" name="city" width="200">
        <cfselect name="state" label="State" width="100">
            <option value="CA">CA</option>
            <option value="MA">MA</option>
            <option value="WA">WA</option>
        </cfselect>
        <cfinput type="text" label="Country" name="Country" width="200" Value="U.S.A.">
        <cfformitem type="html"><b>Phone</b></cfformitem>
        <cfinput type="text" validate="telephone" label="Business" name="businessPhone"
            width="200">
        <cfinput type="text" validate="telephone" label="Mobile" name="cellPhone"
            width="200">
        <cfinput type="text" validate="telephone" label="Fax" name="fax" width="200">
        <cfformitem type="html"><b>Email</b></cfformitem>
        <cfinput type="text" validate="email" name="email" width="200">
    </cfformgroup>

    <cfinput type="Submit" name="submit" value="Submit" >
</cfform>


<!--- If a form was submitted, populate the contact structure from it. --->
<cfif isDefined("Form.Submit")>
    <cfscript>
        sContact.FirstName=Form.firstName;
        sContact.Company=Form.company;
        sContact.LastName=Form.lastName;
        sContact.BusinessAddress.Street=Form.street;
        sContact.BusinessAddress.City=Form.city;
        sContact.BusinessAddress.State=Form.state;
```

```
        sContact.BusinessAddress.Country=Form.country;
        sContact.BusinessPhoneNumber=Form.businessPhone;
        sContact.MobilePhoneNumber=Form.cellPhone;
        sContact.BusinessFax=Form.fax;
        sContact.Email1=Form.email;
    </cfscript>

    <!--- Create the contact in Exchange --->
    <cfexchangecontact action="create"
        username ="#user1#"
        password="#password1#"
        server="#exchangeServerIP#"
        contact="#sContact#"
        result="theUID">

    <!--- Display a confirmation that the contact was added. --->
    <cfif isDefined("theUID")>
        <cfoutput>Contact Added. UID is#theUID#</cfoutput>
    </cfif>
</cfif>
```

cfexchangeconversation

説明

Microsoft Exchange アカウントからの会話の整理および管理を支援します。

次のアクションがサポートされます。

- フィルターに基づいた、フォルダーおよびサブフォルダー内の必要な会話の検索
- 会話のステータス（既読であるかどうか）
- 会話のコピー、移動または削除

履歴

ColdFusion 10: このタグが追加されました。

カテゴリ

通信タグ

シンタックス

```
get
<cfexchangeconversation
    action = "get"
    connection = "connection_ID"
    folderID = "Exchange folder UID"
    name = "query name"
</cfexchangeconversation>
setReadState
<cfexchangeconversation
    action = "setReadState"
    connection = "connection_ID"
    folderID = "Folder_UID"
    UID = "conversation_UID"
    isRead = true|false
</cfexchangeconversation>
copy
<cfexchangeconversation
    action = "copy"
    connection = "connection_ID"
    FolderID = "conversation_folder_UID"
    UID = "conversation_UID"
    destinationFolderID = "destination_folder_UID"
</cfexchangeconversation>
move
<cfexchangeconversation
    action = "move"
    connection = "connection_ID"
    folderID = "conversation_folder_UID"
    UID = "conversation_UID"
    destinationFolderID = "destination_folder_UID"
</cfexchangeconversation>
delete
<cfexchangeconversation
    action = "delete"
    connection = "connection_ID"
    folderID = "conversation_folder_UID"
    UID = "conversation_UID
    deleteType = hardelete|softdelete|movetodeleteditems
</cfexchangeconversation>
```

関連項目

cfexchangecalendar、cfexchangeconnection、cfexchangefilter、cfexchangemail、cfexchangetask、『ColdFusion アプリケーションの開発』の Interacting with Microsoft Exchange Servers

属性

属性	アクション	必須 / オプション	デフォルト	説明
action	N/A	必須		実行するアクションです。 有効な値は次のとおりです。 • get • setReadState • copy • move • delete
connection	すべてのアクション	必須		cfexchangeconnection タグで指定された Exchange サーバーへの接続の名前です。 この属性を省略する場合は、cfexchangecontact タグで cfexchangeconnection タグの接続アクション属性 open を指定して一時的接続を作成します。
name	get	必須		返された会話情報を格納する ColdFusion クエリ変数の名前です。
folderID	すべてのアクション	必須		フォルダーを一意に識別する Exchange UID 値です。大文字と小文字は区別されます。
UID	すべて	必須		yes の場合、会話を既読としてマークします。
isRead	setReadState	必須		会話のステータス（既読であるかどうか）を示します。
destinationFolderID	move copy	必須		宛先のフォルダーを一意に識別する Exchange UID 値です。大文字と小文字は区別されます。
deleteType	delete	必須	moveToDeletedItems	• hardDelete：ストアからフォルダーを完全に削除します。 • softDelete：フォルダーを削除して収集に移動します（収集を使用可能な場合）。 • moveToDeletedItems：フォルダーを削除済みアイテムフォルダーに移動します。

cfexchangeconversation action="get" 用のフィルターパラメーター

パラメータ	説明
maxRows	返される会話の最大数を指定します。デフォルトは 100 で、-1 が指定された場合はすべての会話が返されます。
catagories	（現在のフォルダーのみの）会話内のメッセージにスタンプされたカテゴリのカンマ区切りリストです。
flagStaus	会話のフラグステータスです。現在のフォルダーの個々のメッセージのフラグステータスを集計して計算されます。次の値を指定できます。 • NotFlagged • Flagged • Complete
GlobalCategories	メールボックスのすべてのフォルダーの会話内のメッセージにスタンプされたカテゴリを集約したカンマ区切りリストです。

パラメータ	説明
GlobalFlagStatus	会話のフラグステータスです。メールボックスのすべてのフォルダーの個々のメッセージのフラグステータスを集計して計算されます。 次の値が含まれることがあります。 • NotFlagged • Flagged • Complete
GlobalHasAttachments	メールボックスのすべてのフォルダーの会話内に、添付ファイルがあるメッセージが 1 件以上存在するかどうかを示す値です。
GlobalImportance	その会話の重要度です。メールボックスのすべてのフォルダーの個々のメッセージの重要度を集計して計算されます。次の値が含まれることがあります。 • Low • Normal • High
GlobalItemClasses	メールボックスのすべてのフォルダーの会話内のアイテムのクラスを集約したカンマ区切りリストです。
GlobalItemIds	メールボックスのすべてのフォルダーの会話内のメッセージの ID のカンマ区切りリストです。
GlobalLastDeliveryTime	メールボックスのすべてのフォルダーの会話内で最後に受信したメッセージの配信時刻です。
GlobalMessageCount	メールボックスのすべてのフォルダーの会話内のメッセージの総数です。
GlobalSize	会話のサイズです。メールボックスのすべてのフォルダーの会話内のすべてのメッセージのサイズを加算して計算されます。
GlobalUniqueRecipients	メールボックスのすべてのフォルダーの会話内の受信者のカンマ区切りリストです。
GlobalUniqueSenders	メールボックスのすべてのフォルダーの会話内の送信者のカンマ区切りリストです。
GlobalUniqueUnreadSenders	メールボックスのすべてのフォルダーの会話内の現在未読のメッセージの送信者のカンマ区切りリストです。
GlobalUnreadCount	メールボックスのすべてのフォルダーの会話内の未読メッセージの総数です。
HasAttachments	現在のフォルダーのみの会話内に、添付ファイルがあるメッセージが 1 件以上存在するかどうかを示すブール値です。
UId	会話の UID です。
Importance	会話の重要度です。（現在のフォルダーのみの）個々のメッセージの重要度を集計して計算されます。 次の値が含まれることがあります。 • Low • Normal • High
ItemClasses	（現在のフォルダーのみの）会話内のアイテムのクラスのカンマ区切りリストです。
ItemIds	（現在のフォルダーのみの）会話内のメッセージの ID のカンマ区切りリストです。これは、ItemId オブジェクトの配列です。
LastDeliveryTime	（現在のフォルダーのみの）会話内で最後に受信したメッセージの配信時刻です。
MessageCount	（現在のフォルダーのみの）会話内のメッセージの総数です。

パラメータ	説明
size	会話のサイズです。（現在のフォルダーのみの）会話内のすべてのメッセージのサイズを加算して計算されます。
topic	会話のトピックです。
uniqueRecipients	（現在のフォルダーのみの）会話内のメッセージ受信者のカンマ区切りリストです。
uniqueSenders	（現在のフォルダーのみの）会話内の送信者のカンマ区切りリストです。
uniqueUnreadSenders	（現在のフォルダーのみの）会話内の現在未読のメッセージの送信者のカンマ区切りリストです。
unreadCount	（現在のフォルダーのみの）会話内の未読メッセージの総数です。

例

次の例は、会話に対して、get、setReadState、copy、move および delete のアクションを実行する方法を示しています。

```
<cfexchangeconnection action="open" username="conv" password="Password" server="IP_Address"
                      serverversion="2010" connection="conn1">
<!--- Finding information about Inbox --->
<cfexchangefolder action="getextendedinfo" connection="conn1" name="result" folderpath="Inbox">
<cfexchangefolder action="getextendedinfo" connection="conn1" name="result1" folderpath="Drafts">
<cfexchangeconversation action="get" folderid="#result.uid#" name="conversations" connection="conn1">
    <cfexchangefilter name="topic" value="testcfexchnage3">
    <cfexchangefilter name="categories" value="Yellow Category">
</cfexchangeconversation>
<cfdump var="#conversations#">
<cfset myArray = ArrayNew(1)>
<cfloop query="conversations">
    <cfset temp = ArrayAppend(myArray, "#UID#")>
</cfloop>
<!--- Copy the conversation to Drafts --->
<cfexchangeconversation action="copy" UID="#myArray[1]#" folderid="#result.uid#"
                        destinationfolderid="#result1.uid#" connection="conn1">
<!--- Getting the detail about the conversation --->
<cfexchangeconversation action="get" folderid="#result1.uid#" name="conversations1"
                        connection="conn1">
    <cfexchangefilter name="topic" value="testcfexchnage3">
    <cfexchangefilter name="categories" value="Yellow Category">
</cfexchangeconversation>
<cfdump var="#conversations1#">
<!--- Marking the item as unread --->
<cfexchangeconversation action="setReadState" UID="#conversations1.uid#"
                        folderid="#result1.uid#" isread="false" connection="conn1">
<!--- Deleting the conversations --->
<cfexchangeconversation action="delete" UID="#conversations1.uid#"
                        folderid="#result1.uid#" deletetype="harddelete" connection="conn1">
<!--- Moving the conversations--->
<cfexchangeconversation action="move" UID="#myArray[1]#" folderid="#result.uid#"
                        destinationfolderid="#result1.uid#" connection="conn1">
<!--- Getting the detail about the conversation --->
<cfexchangeconversation action="get" folderid="#result1.uid#" name="conversations2"
                        connection="conn1">
    <cfexchangefilter name="topic" value="testcfexchnage3">
    <cfexchangefilter name="categories" value="Yellow Category">
</cfexchangeconversation>
<cfdump var="#conversations2#">
<!---Moving the conversation back to the initial location--->
<cfexchangeconversation action="move" UID="#conversations2.uid#" folderid="#result1.uid#"
                        destinationfolderid="#result.uid#" connection="conn1">
```

cfexchangefilter

説明

cfexchangemail、cfexchangecalendar、cfexchangetask、および cfexchangecontact の get オペレーションのアクションを制御するフィルタパラメータを指定します。

履歴

ColdFusion 8: このタグが追加されました。

カテゴリ

通信タグ

シンタックス

```
<cfexchangefilter
    name = "filter type"
    value = "filter value">

OR

<cfexchangefilter
    name = "filter type"
    from = "date/time"
    to = "date/time">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfexchangecalendar、cfexchangeconnection、cfexchangecontact、cfexchangemail、cfexchangetask、『ColdFusion アプリケーションの開発』の Getting Exchange items and attachments

属性

属性	必須 / オプション	デフォルト	説明
name	必須		使用するフィルタのタイプです。
from	オプション		フィルタ処理に使用する範囲の開始日または開始日時です。value 属性とともに使用することはできません。from 属性を指定して to 属性を指定しなかった場合は、指定した日付または時刻以降のすべてのエントリが選択されます。 ColdFusion が認識する任意の日付時刻形式を使用できますが、フィルタのタイプに適した値に対応している必要があります。
to	オプション		フィルタ処理に使用する範囲の終了日または終了日時です。value 属性とともに使用することはできません。to 属性を指定して from 属性を指定しなかった場合は、指定した日付または時刻以前のすべてのエントリが選択されます。 ColdFusion が認識する任意の日付時刻形式を使用できますが、フィルタのタイプに適した値に対応している必要があります。
value	オプション		日付や時刻の範囲を指定しないフィルタで使用する値です。from 属性および to 属性とともに使用することはできません。 内容を空にしてこの属性を指定すると、エラーになります。したがって、空の文字列を使用して空の値を検索することはできません。 この属性を空の文字列 ("") に設定すると、指定したフィールドが空であるエントリが検索されます。6.07

cfexchangeCalendar タグのフィルタでは、次の name 属性と value 属性、または to 属性および from 属性を指定することで、特定のアクションのフィルタパラメータを指定できます。

name 属性	指定属性	指定属性の有効な値
maxRows	value	返す行の最大数を指定する正の整数です。デフォルトの最大行数は 100 です。 -1 を指定すると、一致する行がすべて返されます。
allDayEvent	value	ブール値です。
duration	value	分を示す整数値です。
endTime	from to	ColdFusion が日付時刻値として解釈できる文字列です。
fromID	value	Exchange ユーザー ID です。
hasAttachment	value	ブール値です。
importance	value	次のいずれかの値になります。 • high • normal • low
isRecurring	value	ブール値です。
location	value	文字列です。
message	value	文字列です。
optionalAttendees	value	Exchange ユーザー ID のカンマ区切りリストです。
organizer	value	開催者を示す文字列です。この値は Exchange ID または電子メールアドレスである必要はありません。
requiredAttendees	value	Exchange ユーザー ID のカンマ区切りリストです。
sensitivity	value	次のいずれかの値になります。 • normal • personal • private • confidential
startTime	from to	ColdFusion が日付時刻値として解釈できる文字列です。
subject	value	文字列です。
UID	value	各カレンダーエントリを一意に識別する Exchange メッセージ UID です。大文字と小文字は区別されます。

cfexchangecontact タグのフィルタでは、次の name 属性と value 属性を使用できます。他のタグとは異なり、from 属性および to 属性は使用できません。

name 属性	value 属性
maxRows	返す行の最大数を指定する正の整数です。デフォルトの最大行数は 100 です。 -1 を指定すると、一致する行がすべて返されます。
assistant	文字列です。
businessAddress	Street、City、State、Zip、Country の各フィールドを含む構造体です。
businessFax	文字列です。
businessPhoneNumber	文字列です。
categories	カテゴリのカンマ区切りリストです。リスト内のすべてのカテゴリと一致する連絡先が検索されます。
company	文字列です。
description	文字列です。
displayAs	文字列です。
email1	文字列です。
email2	文字列です。
email3	文字列です。
firstName	文字列です。
hasAttachment	ブール値です。
homeAddress	Street、City、State、Zip、Country の各フィールドを含む構造体です。
homePhoneNumber	文字列です。
jobTitle	文字列です。
lastName	文字列です。
mailingAddressType	次のいずれかの値になります。Home、Business、Other。
manager	文字列です。
middleName	文字列です。
mobilePhoneNumber	文字列です。
nickName	文字列です。
office	文字列です。
otherAddress	Street、City、State、Zip、Country の各フィールドを含む構造体です。
otherPhoneNumber	文字列です。
pager	文字列です。
profession	文字列です。
spouseName	文字列です。
webPage	文字列です。

cfexchangemail タグのフィルタでは、次の name 属性と value 属性、または to 属性および from 属性を指定することで、特定のアクションのフィルタパラメータを指定できます。

name 属性	指定属性	指定属性の値
maxRows	value	返す行の最大数を指定する正の整数です。デフォルトの最大行数は 100 です。 -1 を指定すると、一致する行がすべて返されます。
bcc	value	Exchange 電子メールアドレスまたは Web 電子メールアドレスのカンマ区切りリストです。
cc	value	Exchange 電子メールアドレスまたは Web 電子メールアドレスのカンマ区切りリストです。
folder	value	Exchange メールボックスのルートから、検索するフォルダまでの、スラッシュ (/) 区切りのパスです。デフォルトでは、受信トレイの最上位レベルを検索します。cfexchangemail タグは、指定したフォルダのみを検索し、サブフォルダは検索しません。 フォルダ名にスラッシュが含まれる場合は、_xF8FF_ エスケープシーケンスを使用して名前を指定します。 get アクションと move アクションの場合は、このフィールドの代わりに cfexchangemail タグの folder 属性を使用できます。ただし、このフィールドは、folder 属性で指定された値よりも優先されます。
fromID	value	Exchange 電子メールアドレスまたは Web 電子メールアドレスです。
hasAttachment	value	ブール値です。
importance	value	次のいずれかの値になります。 • high • normal • low
isRead	value	ブール値です。
message	value	文字列です。
MessageType	value	次のいずれかの値になります。Mail、Meeting、Meeting_Cancel、Meeting_Request、Meeting_Response、All。 この属性を省略した場合は、すべてのタイプのメッセージが取得されます。 Meeting 属性は、Meeting_Cancel、Meeting_Request、および Meeting_Response タイプのメッセージを取得します。
MeetingUID	value	Exchange カレンダーイベントの UID です。大文字と小文字は区別されます。会議 UID は、Meeting_request または Meeting_response メッセージタイプでのみ使用されます。MessageType フィールドの値として Mail を指定する場合、このフィールドは指定しないでください。
sensitivity	value	次のいずれかの値になります。 • normal • personal • private • confidential
subject	value	文字列です。
timeReceived	from to	ColdFusion が日付時刻値として解釈できる文字列です。

name 属性	指定属性	指定属性の値
timeSent	from to	ColdFusion が日付時刻値として解釈できる文字列です。
toID	value	Exchange 電子メールアドレスまたは Web 電子メールアドレスのカンマ区切りリストです。
uid	value	Exchange メッセージの UID です。大文字と小文字は区別されます。

cfexchangetask タグのフィルタでは、次の name 属性と value 属性、または to 属性および from 属性を指定することで、特定のアクションのフィルタパラメータを指定できます。

name 属性	指定属性	指定属性の値
maxRows	value	返す行の最大数を指定する正の整数です。デフォルトの最大行数は 100 です。 -1 を指定すると、一致する行がすべて返されます。
actualWork	value	時間数を表す数値です。分を指定するには小数を使用します。
billingInfo	value	文字列です。
companies	value	文字列です。
dateCompleted	value	ColdFusion が日付時刻値として解釈できる文字列です。
dueDate	from to	ColdFusion が日付時刻値として解釈できる文字列です。
mail_ID	value	Exchange メール ID のカンマ区切りリストです。このフィルタ値は、接続ユーザーが複数のユーザーにアクセス権を委任していて、その中から特定のユーザーのタスクを選択する場合に便利です。
message	value	文字列です。
mileage	value	文字列です。
percentCompleted	value	0 ～ 100 の数値です。
priority	value	次のいずれかの値になります。 • high • normal • low
reminderDate	value	ColdFusion が日付時刻値として解釈できる文字列です。
startDate	from to	ColdFusion が日付時刻値として解釈できる文字列です。
status	value	有効な値は次のとおりです。 • NOT_STARTED • IN_PROGRESS • COMPLETED • WAITING • DEFERRED

name 属性	指定属性	指定属性の値
subject	value	文字列です。
totalWork	value	時間数を表す数値です。分を指定するには小数を使用します。
UID	value	Exchange の UID です。大文字と小文字は区別されます。

使用方法

cfexchangefilter タグでは、メールメッセージ、カレンダーエントリ、タスク、または連絡先を取得するときに使用する条件を指定します。指定したフィルタ条件に一致するエントリのみが、親タグの name 属性で指定した構造体に返されます。Message や Subject などのテキスト文字列を取るフィールドをフィルタで指定すると、value 属性で指定した値と完全に一致する語句を含む項目が返されます。

cfexchangefilter タグは、action 属性の値が get に設定された cfexchangecalendar、cfexchangecontact、cfexchangemail、または cfexchangetask タグの子タグである必要があります。

cfexchangemail などの ColdFusion exchange タグの本文で複数の cfexchangefilter タグを指定した場合は、指定したフィルタが組み合わされ、すべての cfexchangefilter タグの条件に一致するレコードが選択されます。name 属性の値が同じである cfexchangefilter タグを複数指定した場合は、その属性を持つ最後のタグのフィルタ条件が使用されます。

例

次の例では、adobe.com を含む電子メールアドレスから特定のユーザーに先週送信されたメールメッセージを取得します。この例の目的はデータを表示することではなくメッセージを取得することであるため、cfdump タグを使用して結果を示します。

```
 <cfset endTime = Now()>
<cfset startTime = DateAdd("d","-7", endTime)>
<cfexchangemail action="get" name="weeksMail" server="#exchangeServerIP#"
        username ="#user1#" password="#password1#">
    <cfexchangefilter name="FromID" value="adobe.com">
    <cfexchangefilter name="TimeSent" from="#startTime#" to="#endTime#">
</cfexchangemail>

<cfdump var="#weeksMail#">
```

cfexchangefolder

説明

メールフォルダーに対して様々なアクションを実行できます（フォルダー情報の取得、フォルダーの検索、フォルダーの作成、コピー、変更、移動、削除、フォルダーの中身のクリアなど）。

履歴

ColdFusion 10: このタグが追加されました。

カテゴリ

通信タグ

シンタックス

```
getExtendedlnfo
<cfexchangefolder
    action = "getExtendedlnfo"
    folderID = "Exchange folder UID"
    connection = "connection_ID"
    name = "query_name"/>
OR

getExtendedlnfo
<cfexchangefolder
    action = "getExtendedlnfo"
    folderPath = "Exchange_folder_Path"
    connection = "connection_ID"
    name = "query_name"
    pathDelimitter = "delimiter_characters"/>
getlnfo
<cfexchangefolder
    action = "getInfo"
    connection = "connection_ID"
    folderID = "Exchange folder UID"
    name = "query_name"/>
OR
getlnfo
<cfexchangefolder
    action = "getInfo"
    folderPath = "Exchange_folder_Name"
    connection = "connection_ID"
    name = "query_name"
    pathDelimitter = "delimiter_characters"/>
findSubFolders
<cfexchangefolder
    action = "findSubFolders"
    folderID = "Exchange folder UID"
    connection = "connection_ID"
    name = "query_name"/>
create
<cfexchangefolder
    action = "create"
    folder = "struct"
    parentFolderID = "folder_UID"
    connection = "connection_ID"
    result = "variable for contact UID"/>
copy
<cfexchangefolder
    action = "copy"
    destinationFolderID = "Folder_UID"
    sourceFolderID = "folder_UID">
    connection = "connection_ID"
    result = "variable for contact UID"/>
delete
<cfexchangefolder
    action = "delete"
    deleteType = "hardDelete|softDelete|moveToDeletedItems"
    uid = "folder UID">
    connection = "connection_ID"/>
```

```
move
<cfexchangefolder
    action = "move"
    destinationFolderID = "Folder_UID"
    sourceFolderID = "folder_UID">
    connection = "connection_ID"
    result = "variable for contact UID"/>
modify
<cfexchangefolder
    action = "modify"
    uid = "folder UID"
    folder = "strut"
    connection = "connection_ID"/>
empty
<cfexchangefolder
    action = "empty"
    uid = "folder UID"
    deleteType = "hardDelete|softDelete|moveToDeletedItems">
    deleteSubFolder = "true|false"
    connection = "connection_ID"/>
```

関連項目

cfexchangecalendar、cfexchangeconnection、cfexchangefilter、cfexchangemail、cfexchangetask、『ColdFusion アプリケーションの開発』の Interacting with Microsoft Exchange Servers

属性

属性	アクション	必須 / オプション	デフォルト	説明
action	N/A	必須		実行するアクションです。 有効な値は次のとおりです。 • getInfo • getExtendedInfo • findSubFolders • create • copy • delete • move • modify • empty
uid	getExtendedInfo getInfo	getExtendedInfo および getInfo でフォルダーパスが指定されている場合は不要		アクションを実行するフォルダーの特定に使用する UID です。
name	getExtendedInfo getInfo findSubFolders	必須		返されたフォルダーの情報を格納する ColdFusion クエリ変数の名前です。

属性	アクション	必須 / オプション	デフォルト	説明
folderID	getExtendedInfo getInfo findSubFolders delete modify empty	getExtendedInfo および getInfo でフォルダーパスが指定されている場合は不要		アクションを実行するフォルダーの特定に使用する UID です。
folderPath	getExtendedInfo getInfo	folderID が指定されている場合は不要		アクションを実行するフォルダーのフルパスです。 パスの区切り文字を指定しない場合は、デフォルトで / が使用されます。
pathDelimiter	getExtendedInfo getInfo	オプション	/	フォルダーの区切りに使用する区切り文字を指定できます。
parentFolderId	create	必須		サブフォルダーを作成するフォルダーの特定に使用する UID です。
connection	すべてのアクション	必須		cfexchangeconnection タグで指定された Exchange サーバーへの接続の名前です。 この属性を省略する場合は、cfexchangecontact タグで cfexchangeconnection タグの接続アクション属性 open を指定して一時的接続を作成します。
result	create copy move	必須		いずれかのアクションを実行したときに Exchange Server から返された結果を格納するクエリ変数です。
destinationFolderID	copy move	必須		宛先のフォルダーを一意に識別する Exchange UID 値です。大文字と小文字は区別されます。
sourceFolderID	copy move	必須		移動先のフォルダーにコピーまたは移動するフォルダーが存在する移動元のフォルダーの特定に使用する UID です。
deleteType	delete empty	オプション	moveToDeletedItems	• hardDelete：Exchange Server からフォルダーを完全に削除します。 • softDelete：Exchange Server の収集にフォルダーを移動します（収集を使用可能な場合）。 • moveToDeletedItems：フォルダーを削除済みアイテムフォルダーに移動します。
deleteSubFolders	empty	オプション	False	true の場合、サブフォルダーを削除します。
folder	create modify	必須		作成または変更するフォルダーの必要な情報（表示名やフォルダーのクラスなど）が含まれる構造体です。

cfexchangefolder action = "getExtendedInfo" の結果の構造体の値

getExtendedInfo アクションでは、result という構造体に次のフィールドが含まれます。

フィールド	説明
ChildFolderCount	フォルダーに存在する子フォルダーの数です。
displayName	フォルダーの表示名です。
folderClass	フォルダーのクラスです。
FolderPath	Exchange Server フォルダーのパスです。
ManagedFolder	次のプロパティが含まれる構造体です。 • canDelete：ユーザーがフォルダーのオブジェクトを削除できるかかどうかを示すブール値です。 • canRenameOrMove：ユーザーがフォルダーのオブジェクトの名前を変更したり、オブジェクトを移動できるかどうかを示すブール値です。 • mustDisplayComment：クライアントアプリケーションがユーザーに comment プロパティを表示する必要があるかどうかを示すブール値です。 • HasQouta：フォルダーにクォータが設定されているかどうかを示すブール値です。 • IsManagedFoldersRoot：フォルダーが管理フォルダー階層のルートであるかどうかを示すブール値です。 • ManagedFolderId：管理フォルダーの ID です。 • comment：フォルダーに関連付けられているコメントです。 • StorageQuota：フォルダーのストレージクォータです。 • FolderSize：フォルダーのサイズです。 • HomePage：フォルダーに関連付けられているホームページです。
mustDisplayComment	クライアントアプリケーションがユーザーに comment プロパティを表示する必要があるかどうかを示すブール値です。
ParentFolderId	フォルダーの親フォルダーの ID です。

フィールド	説明
権限	フォルダーに対する権限のリストです。これは、権限の構造体の配列で、次の値が含まれます。 • canCreateItems：ユーザーが新しいアイテムを作成できるかどうかを示すブール値です。 • canCreateSubFolders：ユーザーがサブフォルダーを作成できるかどうかを示すブール値です。 • deleteItems：ユーザーが既存のアイテムを削除できるかどうかと、その詳細を示します。値は、None（ユーザーは関連する権限を持っていない）、Owned（ユーザーは自分が所有するアイテムに対する関連権限を持っている）、および All（ユーザーはすべてのアイテムに対する関連権限を持っている）です。 • displayPermissionLevel：Outlook で表示される、このフォルダー権限の権限レベルです。次の値が含まれることがあります。None、Owner、PublishingEditor、Editor、PublishingAuthor、Freebusytimeandsubjectandlocation、FreebusytimeOnly、Author、NonEditingAuthor、Reveiwer、Contributor および Custom。 • EditItems：ユーザーが編集の権限を持っているフォルダー内のアイテムを示します。値は none、owned または all です。 • isFolderContact：ユーザーがフォルダーの連絡先であるかどうかを示すブール値です。 • isFolderOwner：ユーザーがフォルダーの所有者であるかどうかを示すブール値です。 • isFolderVisible：ユーザーにフォルダーが表示されるかどうかを示すブール値です。 • PermissionLevel：フォルダーに対してユーザーが持っている権限の組み合わせを表します。値は、displayPermissionLevel の値と同じですが、Freebusytimeandsubjectandlocation と FreebusytimeOnly は除きます。 • readItems：アイテムの読み取りアクセス権限です。値は次のとおりです。None（ユーザーは、フォルダー内のアイテムに対する読み取りアクセス権を持っていません）、TimeOnly（ユーザーは、予定の開始日時および終了日時を読み取ることができます。これは予定表フォルダーにのみ適用されます）、TimeAndSubjectAndLocation（ユーザーは、予定の開始日時、終了日時、件名および場所を読み取ることができます。これは予定表フォルダーにのみ適用されます）、および FullDetails（ユーザーは、アイテムのすべての詳細へのアクセス権を持っています）。 • UserIDDisplayName：ユーザーの表示名です。 • UserIDprimarySMTPAddress：ユーザーのプライマリ SMTP アドレスです。 • userIDSID：ユーザーの SID です。 • UserIdstandardUser：次のいずれかの標準ユーザーを示します。default（デフォルトの委任ユーザー。デフォルトの委任権限を定義するために使用します）、および Anonymous（匿名の委任ユーザー。認証されていないユーザーに対する委任権限を定義するために使用します）。
TotalCount	フォルダーに含まれるアイテムの総数です。
UnreadCount	フォルダー内の未読アイテムの数です。

cfexchangefolder action = "getInfo" の結果の構造体の値

getInfo アクションでは、result という構造体に次のフィールドが含まれます。

フィールド	説明
ChildFolderCount	このフォルダーに存在する子フォルダーの数
displayName	フォルダーの表示名
folderClass	フォルダーのクラス
folderPath	Exchange Server フォルダーのパスです。

フィールド	説明
UID	フォルダーの ID
ParentFolderId	現在のフォルダーの親フォルダーの ID
TotalCount	フォルダーに含まれるアイテムの総数
UnreadCount	フォルダー内の未読アイテムの数

cfexchangefolder action = "findSubFolders" 用のフィルターパラメーター

findSubFolders アクションでは、result というクエリにサブフォルダーの詳細が含まれます。値は、cfexchangefolder action = "getInfo" の値と同じです。

例

次のコードは、getExtendedInfo、findSubFolders、getInfo、copy、delete、modify、move および create のアクションを実行する方法を示しています。

```
<cfexchangeconnection action="open" username="folder" password="Password" server="IP_Address"
                      serverversion="2010" connection="conn1">
<cfexchangefolder action="getextendedinfo" connection="conn1" name="result" folderpath="Drafts">
<cfexchangefolder action="getextendedinfo" connection="conn1" name="result3" folderpath="Inbox">
<!--- Checking if the folder is already present.. If it is present then delete it --->
<cfexchangefolder action="findsubfolders" connection="conn1" name="final" uid="#result3.uid#">
    <cfexchangefilter name="displayname" value="folder1">
</cfexchangefolder>
<cfexchangefolder action="findsubfolders" connection="conn1" name="final1" uid="#result.uid#">
    <cfexchangefilter name="displayname" value="folder1">
</cfexchangefolder>
<cfif (#final.displayname# EQ "folder1")>
    <cfexchangefolder action="delete" connection="conn1" uid="#final.uid#" deletetype="harddelete">
    <cfscript>
        sleep(1000);
    </cfscript>
<cfelseif (#final1.displayname# EQ "folder1")>
    <cfexchangefolder action="delete" connection="conn1" uid="#final1.uid#" deletetype="harddelete">
    <cfscript>
        sleep(1000);
    </cfscript>
</cfif>
<cfset newfolder = structnew()>
<cfset newfolder.displayname = "folder1">
<cfset newfolder.folderclass = "IPF.Note">
<cfexchangefolder action="create" connection="conn1" parentfolderid="#result.uid#"
                  folder="#newfolder#" result="result1">
</cfexchangefolder>
<!--- Source folder --->
<cfexchangefolder action="findsubfolders" connection="conn1" name="result2" uid="#result.uid#">
    <cfexchangefilter name="displayname" value="folder1">
</cfexchangefolder>
<!---modifying the folder--->
<cfset modfolder = structnew()>
<cfset modfolder.displayname = "exchange">
<cfset modfolder.folderclass = "IPF.Calendar">
<cfexchangefolder action="modify" connection="conn1" uid="#result2.uid#" folder="#modfolder#">
</cfexchangefolder>
<cfexchangefolder action="findsubfolders" connection="conn1" name="result5" uid="#result.uid#">
    <cfexchangefilter name="displayname" value="exchange">
</cfexchangefolder>
<cfdump var="#result5#">
<!--- Destination folder --->
```

```
<cfexchangefolder action="getInfo" connection="conn1" name="result3" folderpath="Inbox">
<cfexchangefolder action="copy" connection="conn1" result="copiedfolderid"
                  sourcefolderid="#result2.uid#" destinationfolderid="#result3.uid#">
<cfexchangefolder action="findsubfolders" connection="conn1" name="result2" uid="#result.uid#">
    <cfexchangefilter name="displayname" value="exchange">
</cfexchangefolder>
<cfdump var="#result2#">
<cfexchangefolder action="findsubfolders" connection="conn1" name="result4" uid="#result3.uid#">
    <cfexchangefilter name="displayname" value="exchange">
</cfexchangefolder>
<cfdump var="#result4#">
<!---Deleting the folder from source --->
<cfexchangefolder action="delete" connection="conn1" uid="#result2.uid#" deletetype="harddelete">
<cfexchangefolder action="move" connection="conn1" result="movedfolderid"
                  sourcefolderid="#result4.uid#" destinationfolderid="#result.uid#">
<cfexchangefolder action="findsubfolders" connection="conn1" name="result6" uid="#result.uid#">
    <cfexchangefilter name="displayname" value="exchange">
</cfexchangefolder>
<cfdump var="#result6#">
<cfexchangefolder action="delete" connection="conn1" uid="#result6.uid#" deletetype="harddelete">
```

cfexchangemail

説明

メールメッセージと添付ファイルの取得、メッセージの削除、Microsoft Exchange サーバー上のメッセージのプロパティ設定を行います。

履歴

ColdFusion 10：serverVersion 属性と folderID 属性が追加されました。

ColdFusion 8: このタグが追加されました。

カテゴリ

通信タグ

シンタックス

```
 delete
<cfexchangemail
    required
    action = "delete"
    uid = "message UID,message UID,..."
    optional
    connection = "connection ID"
    folder = "Exchange folder path">

deleteAttachments
<cfexchangemail
    required
    action = "deleteAttachments"
    uid = "message UID"
    optional
    connection = "connection ID">
    folder = "Exchange folder path">

get
<cfexchangemail
    required
```

```
    action = "get"
    name = "query identifier"
    optional
    connection = "connection ID"
    folder = "Exchange folder path">
    folderID = "Exchange folder UID"

    <cfexchangefilter name = "filter type" value = "filter value">
    <cfexchangefilter name = "filter type" value = "filter value">
    ...
</cfexchangemail>

getAttachments
<cfexchangemail
    required
    action = "getAttachments"
    name = "query identifier"
    uid = "message UID"
    optional
    attachmentPath = "directory path"
    connection = "connection ID"
    folder = "Exchange folder path"
    generateUniqueFilenames = "no|yes">

getMeetingInfo
<cfexchangemail
    required
    action = "getMeetingInfo"
    meetingUID = "meeting UID"
    name = "query identifier"
    optional
    connection = "connection ID"
    mailUID = "message UID">

move
<cfexchangemail
    required
    action = "move"
    destinationFolder = "Exchange folder path"
    optional
    connection = "connection ID"
    folder = "Exchange folder path">
    folderID = "Exchange folder UID"

    <cfexchangefilter name = "filter type" value = "filter value">
    <cfexchangefilter name = "filter type" value = "filter value">
    ...
</cfexchangemail>

set
<cfexchangemail
    required
    action = "set"
    message = "#structure with values to set#">
    uid = "message UID">
    optional
    connection = "connection ID"
    folder = "Exchange folder path">
    folderID = "Exchange folder UID"
```

注意：connection 属性を省略する場合は、cfexchangemail タグで cfexchangeconnection タグの属性を指定して一時的接続を作成します。この場合、ColdFusion はタグが完了すると接続を閉じます。詳細については、cfexchangeconnection タグの open アクションを参照してください。

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfexchangecalendar、cfexchangeconnection、cfexchangecontact、cfexchangefilter、cfexchangetask、『ColdFusion アプリケーションの開発』の Interacting with Microsoft Exchange Servers

属性

注意：folder や destinationFolder などの属性でフォルダパスを使用する際に、フォルダ名にスラッシュ (/) が含まれている場合は、変換で文字がパスの区切り文字として解釈されないように、_xF8FF_ エスケープ文字を使用してフォルダ名を指定します。

属性	アクション	必須 / オプション	デフォルト	説明
action	すべて	必須		実行するアクションです。有効な値は次のとおりです。 • delete • deleteAttachments • get • getAttachments • getMeetingInfo • move • set
attachmentPath	getAttachments	オプション		添付ファイルを保存するディレクトリのファイルパスです。指定したディレクトリが存在しない場合は、自動的に作成されます。 **メモ：**この属性を省略すると、添付ファイルは保存されません。相対パスを指定する場合は、GetTempDirectory 関数によって返される ColdFusion テンポラリディレクトリがパスのルートになります。
connection	すべて	オプション		cfexchangeconnection タグで指定された Exchange サーバーへの接続の名前です。 この属性を省略する場合は、cfexchangecalendar タグで cfexchangeconnection タグの接続アクション属性 open を指定して一時的接続を作成します。
destinationFolder	move	必須		メールボックスのルートから、メッセージの移動先フォルダまでの、スラッシュ (/) 区切りのパスです。

属性	アクション	必須 / オプション	デフォルト	説明
folder	getMeetingInfo を除くすべて	オプション		メールボックスのルートから、メッセージが存在するフォルダまでの、スラッシュ (/) 区切りのパスです。cfexchangemail タグは、指定したフォルダのみを検索し、サブフォルダは検索しません。 get および move アクションの場合、cfexchangefilter 子タグで name="folder" 属性を指定すると、この属性を設定したのと同じことになり、この属性の値よりも優先されます。 この属性を省略し (または get および move アクションを指定し)、対応する cfexchangefilter 設定を使用しなかった場合は、受信トレイの最上位レベルが検索されます。
folderID	get、move、set	オプション		フォルダーを一意に識別する Exchange UID 値です。大文字と小文字は区別されます。 指定しない場合は、folder が使用されます。folder と folderID のどちらも指定しない場合は、受信トレイがデフォルトのフォルダーとして使用され、操作が実行されます。
generateUniqueFilenames	getAttachments	オプション	no	同じ名前の添付ファイルが複数ある場合に、一意のファイル名を生成するかどうかを指定するブール値です。同じ名前を持つ添付ファイルが複数存在し、このオプションが yes の場合、ColdFusion は、競合するファイル名の後ろ (拡張子の前) に番号を付加します。たとえば、myfile.txt という名前の添付ファイルが 3 つある場合は、myfile.txt、myfile1.txt、myfile2.txt という名前で保存されます。
mailUID	getMeetingInfo	オプション		会議出席依頼、応答、または中止の通知を含むメールメッセージの UID です。大文字と小文字は区別されます。この属性は、1 つの会議に関連するメッセージが複数ある場合に使用します。
meetingUID	getMeetingInfo	必須		通知を受け取った会議の UID です。大文字と小文字は区別されます。
message	set	必須		設定するプロパティとその値を含む構造体への参照です。この属性はシャープ記号 (#) で囲んで指定する必要があります。 メッセージ構造体の詳細については、「使用方法」を参照してください。

属性	アクション	必須 / オプション	デフォルト	説明
name	get getAttachments getMeetingInfo	必須		返されたメールメッセージ、または添付ファイルや会議に関して取得した情報を格納する ColdFusion クエリー変数の名前です。返されるデータの詳細については、「使用方法」を参照してください。
serverVersion		オプション	2007	Microsoft Exchange Server のバージョンを指定します。 有効な値は次のとおりです。 • 2003 • 2007 • 2010 詳細を指定しない場合は、デフォルトで 2007 が使用されます。 指定した値は、アプリケーションレベルで指定した値よりも優先されます。
uid	delete getAttachments set	必須		アクションの対象となるメッセージの UID です。大文字と小文字は区別されます。 delete アクションの場合は、この属性で UID 値のカンマ区切りリストを指定できます。deleteAttachments、getAttachments、および set アクションの場合は、1 つの UID 値のみを指定できます。

使用方法

cfexchangemail タグを使用すると、cfmail タグでは実行できない Exchange サーバー上のメールに対するアクションを実行できます (メールメッセージの送信、転送、応答を行うには、cfmail タグを使用する必要があります)。次のアクションを実行するには、cfexchangemail タグを使用します。

- サーバーからメールメッセージを完全に削除する。
- 特定のメッセージの添付ファイルを取得する。
- 件名、送信者 ID、受信者 ID、受信時刻などのフィルタ条件に一致するメッセージを取得する。
- 特定のメッセージの添付ファイルを取得する。
- 会議出席依頼や中止通知などの通知を受け取った会議に関する詳細情報を取得する。
- フォルダ間でメッセージを移動する (削除済みアイテムフォルダを含む)。
- 特定のメールメッセージのプロパティを設定する。

このタグを使用するには、Exchange サーバーに接続する必要があります。複数の連絡先レコードを作成する場合のように、Exchange サーバーと対話するタグを複数使用する場合は、cfexchangeconnection タグを使用して継続的接続を作成します。その場合は、個々の cfexchangemail タグで接続識別子を指定します (タスク、連絡先、接続などにもアクセスする場合はその他の ColdFusion Exchange タグでも接続識別子を指定します)。これにより、個々のタグに対して接続を作成して閉じる必要がなくなるので、システムの負荷が減少します。

また、ColdFusion が 1 個の cfexchangemail タグを処理する間だけ存続する一時的接続を作成することもできます。その場合は、cfexchangemail タグで接続属性を直接指定します。接続属性の詳細については、cfexchangeconnection タグを参照してください。

delete アクション

delete アクションはサーバーからメッセージを完全に削除します。Outlook で Shift + Delete キーを押した場合のアクションに相当します。削除済みアイテムフォルダにメッセージを移動するには、move アクションを使用します。これは、Outlook で Delete キーを押した場合のアクションに相当します。

delete アクションを指定する場合は、削除するタスクを識別する Exchange UID のカンマ区切りリストを uid 属性で指定する必要があります。UID 値を確認するには、適切なフィルタ式を指定した get アクションを使用します。

cfexchangemail タグで指定した UID がすべて無効な場合はエラーが発生します。1 つでも有効な UID が含まれる場合は、無効な UID が無視され、有効な UID に対応する項目が削除されます。

get アクション

get アクションを指定する場合は、子の cfexchangefilter タグを使用して取得するメッセージを指定します。詳細については、cfexchangefilter を参照してください。タグの処理が完了すると、name 属性で指定したクエリーオブジェクトに、見つかったメッセージごとのレコードが格納されます。各レコードには次の列があります。

列	説明
BCC	Exchange ユーザー ID または Web 電子メールアドレスのカンマ区切りリストです。
CC	Exchange ユーザー ID または Web 電子メールアドレスのカンマ区切りリストです。
Folder	Exchange メールボックスのルートから、メッセージが存在するメールフォルダまでの、スラッシュ (/) 区切りのパスです。
FromID	Exchange ユーザー ID または Web 電子メールアドレスです。
HasAttachment	メッセージに少なくとも 1 つの添付ファイルがあるかどうかを示すブール値です。
HTMLMessage	HTML 形式のメッセージを含む文字列です。
IsRead	ブール値です。
Message	プレーンテキスト形式のメッセージ内容を含む文字列です。
MessageType	次のいずれかの文字列です。 • Mail • Meeting_Cancel • Meeting_Request • Meeting_Response
MeetingResponse	メッセージタイプが Meeting_response の場合、この列には応答コードとして Accept、Decline、Tentative のいずれかの文字列が入ります。他のメッセージタイプでは、このフィールドは使用されません。
MeetingUID	メッセージタイプが Meeting_Cancel、Meeting_request、または Meeting_response の場合、この列にはこのメッセージに関連するカレンダーイベントの UID が入ります。要求に応答するには、cfexchangecalendar タグでこの値を使用します。メッセージタイプが Mail の場合、このフィールドは使用されません。
Sensitivity	次のいずれかの文字列です。 • public • private • normal • company-confidential
Subject	文字列です。

列	説明
TimeReceived	ColdFusion の日付時刻オブジェクトです。
TimeSent	Coldfusion の日付時刻オブジェクトです。
Told	Exchange ユーザー ID または Web メール ID のカンマ区切りリストです。
UID	メッセージの Exchange UID です。

注意：出席依頼送信者が会議出席依頼メッセージを受信できるのは、その送信者が出席者リストに記載されている場合のみです。

getAttachments アクション

getAttachments アクションを使用する場合は、1 つの UID と name 属性を指定します。cfexchangecontact タグは、name 属性で指定したクエリーオブジェクトに、添付ファイルごとのレコードを格納します。各レコードには、UID で指定した添付ファイルに関する次の情報が含まれます。

列名	説明
attachmentFileName	添付ファイルのファイル名です。
attachmentFilePath	サーバー上の添付ファイルの絶対パスです。attachmentPath 属性を省略すると、この列には空の文字列が入ります。
CID	添付ファイルの content-ID です。メッセージにイメージを埋め込むために HTML の img タグ内で使用します。
mimeType	添付ファイルの MIME タイプです (text/html など)。
isMessage	添付ファイルがメッセージかどうかを指定するブール値です。
size	添付ファイルのサイズです (単位 : バイト)。

添付ファイルは attachmentPath 属性で指定したディレクトリに保存されます。attachmentPath 属性を省略すると、添付ファイル自体は取得されず、添付ファイルに関する情報が取得されます。この方法を使用すると、添付ファイルの取得に伴うオーバーヘッドを発生させることなく、添付ファイルを調べることができます。

1 つのメッセージに同じ名前の添付ファイルが複数ある場合は、generateUniqueFilenames="yes" を指定した場合でも、添付ファイル情報の構造体には重複する元の名前でそれらの添付ファイルがリストされます。generateUniqueFilenames 属性は、ディスク上のファイルの名前に対してのみ適用されます。

次のシンタックスを使用して、メモリ内の attachmentPath ディレクトリを指定します。メモリ内のファイルはディスクに書き込まれないため、一時的データの処理が早くなります。

```
attachmentpath = "ram:///path"
```

パスには、ram:///petStore/orders/messageAttachments のように複数のディレクトリを含めることができます。ファイルを指定する前に、パスに含まれるすべてのディレクトリを作成しておく必要があります。メモリ内のファイルの使用方法について詳しくは、『ColdFusion アプリケーションの開発』の Working with in-memory files を参照してください。

getAttachments アクションは、Exchange のサーバー設定で EWS (Exchange Web サービス) の認証が basic に設定されている場合にのみ有効です。IWA (Integrated Windows Authentication) はサポートされていません。

getMeetingInfo アクション

出席依頼や中止通知などの通知メッセージを受け取った会議に関する情報 (会議の開始時刻、終了時刻、場所など) を取得するには、getMeetingInfo アクションを使用します。この情報は、get アクションで返される通知メッセージのクエリーオブジェクトには直接格納されません。

注意：本マニュアルのこの後の情報では、getMeetingInfo アクションの動作が完全には反映されていません。最新の情報については、Adobe Web サイトで HTML 形式で提供されているオンラインの ColdFusion マニュアルで、cfexchangemail を参照してください。

getMeetingInfo アクションを指定する場合は、会議の UID を meetingUID 属性で指定します。この UID 値は、get アクションで返されるクエリーレコードから取得できます。また、通知を含む特定のメッセージの UID を messageUID 属性で指定することもできます。1 つの会議に関して複数のメッセージを受け取った場合は、この属性を使用して 1 つの通知メッセージを選択できます。

タグの処理が完了すると、name 属性で指定したクエリーオブジェクトに、見つかったメッセージごとのレコードが格納されます。各レコードには次の列があります。

フィールド	**説明**
AllDayEvent	終日のイベントかどうかを示すブール値です。
Duration	イベントの継続時間です (単位 : 分)。
EndTime	ColdFusion の ODBC 日付時刻形式で示されるイベントの終了時刻です。
From	会議通知を送信したユーザーのメール ID です。
HasAttachment	そのイベントに添付ファイルがあるかどうかを示すブール値です。
Importance	次のいずれかの値になります。 • high • normal • low
IsRecurring	そのイベントが反復されるかどうかを示すブール値です。
Location	イベントの場所を示す文字列です。
MeetingUID	カレンダー内のイベントの UID です。
Message	イベントに関するメッセージを含む文字列です。
OptionalAttendees	メール ID のカンマ区切りリストです。
Organizer	文字列です。この値は Exchange ID または電子メールアドレスであるとはかぎりません。
Reminder	イベントの何分前にアラームメッセージを表示するかを指定します。
RequiredAttendees	メール ID のカンマ区切りリストです。
Resources	Exchange スケジュールリソース (会議室や機材など) のメール ID のカンマ区切りリストです。
Sensitivity	次のいずれかの値になります。 • normal • company-confidential • personal • private
StartTime	ODBC 日付時刻形式で示されるイベントの開始時刻です。

フィールド	説明
Subject	イベントの件名を示す文字列です。
TimeReceived	ODBC 日付時刻形式で示されるメッセージの受信時刻です。
UID	イベント通知を含むメッセージの UID です。

move アクション

フォルダ間でメッセージを移動するには、move アクションを使用します。このアクションを使用すると、削除済みアイテムフォルダにメッセージを移動できます。これは、Outlook で Delete キーを押した場合のアクションに相当します。

move アクションを指定する場合は、移動先のフォルダを指定し、必要に応じてメッセージの移動元フォルダを指定します (デフォルトの移動元フォルダは受信トレイです)。取得するメッセージを指定するには、子の cfexchangefilter タグを使用します。詳細については、cfexchangefilter を参照してください。

set アクション

set アクションを指定する場合は、message 属性で指定した構造体に、設定するメッセージプロパティを指定するキーと値のペアを格納します。次の表に、キーの名前と有効な値を示します。

キー名	有効な値
IsRead	yes、no
Importance	high、normal、low
Sensitivity	normal、company-confidential、personal、private

例

次の例では、前週に docuser2 から受信した受信トレイ内のメールメッセージに含まれる添付ファイルを取得します。各メッセージの添付ファイルを、固有の名前のディレクトリに格納します。UID をファイル名として使用することはできません。これは、添付ファイルを含むメッセージごとに、UID には、メッセージの UID、ディレクトリパス、件名、日付、および送信元のアプリケーションレポートが含まれ、その後にメッセージの添付ファイルを示す表が含まれている可能性があるためです。この表には、添付ファイルの名前、サイズ、MIME タイプが表示されます。

```
 <!--- Index for message attachement directory --->
<cfset i=1>
<!--- Dates for date range --->
<cfset rightNow = Now()>
<cfset lastWeek = DateAdd("d","-7", rightNow)>

<cfexchangeconnection
    action="open"
    username ="#user1#"
    password="#password1#"
    server="#exchangeServerIP#"
    connection="testconn1">

<cfexchangemail action="get" folder="Inbox " name="weeksMail" connection="testconn1">
    <cfexchangefilter name="FromID" value="docuser2">
    <cfexchangefilter name="TimeSent" from="#lastWeek#" to="#rightNow#">
</cfexchangemail>

<cfloop query="weeksMail">
    <cfif weeksmail.HasAttachment>
        <cfexchangemail action="getAttachments"
            connection="testconn1"
            folder="Inbox/MailTest"
```

```
            uid="#weeksmail.uid#"
            name="attachData"
            attachmentPath="C:\temp\cf_files\attachments\msg_#i#"
            generateUniqueFilenames="yes">
        <cfoutput>
            Message ID #weeksmail.uid# attachments are in the directory
                C:\temp\cf_files\attachments\Msg_#i#<br />
            <br />
                Message information:<br />
                Subject: #weeksmail.Subject#<br />
                Sent: #dateFormat(weeksmail.TimeSent)#<br />
                From:  #weeksmail.FromID#<br />
            <br />
            Attachments<br />
            <cftable query="attachData" colheaders="yes">
                <cfcol header="File Name" text="#attachmentFilename#">
                <cfcol header="Size" text="#size#">
                <cfcol header="MIME type" text="#mimeType#">
            </cftable>
        </cfoutput>
        <cfset i++>
    </cfif>
</cfloop>
<cfexchangeconnection action="close" connection="testconn1">
```

cfexchangetask

説明

Microsoft Exchange タスクの作成、削除、変更、および取得を行い、タスクの添付ファイルを取得します。

注意：いずれのアクションに関しても、connection 属性を指定しない場合に使用するその他の属性については、cfexchangeconnection を参照してください。

履歴

ColdFusion 10：serverVersion 属性が追加されました。

ColdFusion 8: このタグが追加されました。

カテゴリ

通信タグ

シンタックス

```
 create
<cfexchangetask
    required
    action = "create"
    task = "#task information structure#"
    optional
    connection = "connection ID"
    result = "variable for event UID">

delete
<cfexchangetask
    required
    action = "delete"
    uid = "task UID,task UID, ..."
    optional
    connection = "connection ID">

deleteAttachments
<cfexchangetask
    required
    action = "deleteAttachments"
    uid = "task UID"
    optional
    connection = "connection ID">

get
<cfexchangetask
    required
    action = "get"
    name = "query identifier"
    optional
    connection = "connection ID">

getAttachments
<cfexchangetask
    required
    action = "getAttachments"
    name = "query identifier"
    uid = "task UID"
    optional
    attachmentPath = "directory path"
    connection = "connection ID"
    generateUniqueFilenames = "no|yes">

modify
<cfexchangetask
    required
    action = "modify""
    task = "#task information structure#"
    uid = "task UID">
    optional
    connection = "connection ID">
```

注意：connection 属性を省略する場合は、cfexchangetask タグで cfexchangeconnection タグの属性を指定して一時的接続を作成します。この場合、ColdFusion はタグが完了すると接続を閉じます。詳細については、cfexchangeconnection タグの open アクションを参照してください。

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfexchangecalendar、cfexchangeconnection、cfexchangecontact、cfexchangefilter、cfexchangemail、『ColdFusion アプリケーションの開発』の Interacting with Microsoft Exchange Servers

属性

次の表に、各属性の詳細な情報を示します。この表には、属性の名前、その属性が適用されるアクション (action 属性の値)、その属性がアクションに対して必須かオプションか、デフォルト値 (ある場合)、属性と有効な値についての詳細な説明が示されています。

属性	アクション	必須 / オプション	デフォルト	説明
action	すべて	必須		実行するアクションです。有効な値は次のとおりです。 • create • delete • deleteAttachments • get • getAttachments • modify
attachmentPath	getAttachments	オプション		添付ファイルを保存するディレクトリのファイルパスです。指定したディレクトリが存在しない場合は、自動的に作成されます。 **メモ :** この属性を省略すると、添付ファイルは保存されません。相対パスを指定する場合は、GetTempDirectory 関数によって返される ColdFusion テンポラリディレクトリがパスのルートになります。
connection	すべて	オプション		cfexchangeconnection タグで指定された Exchange サーバーへの接続の名前です。 この属性を省略する場合は、cfexchangetask タグで cfexchangeconnection タグの接続属性を指定して一時的接続を作成します。
generateUniqueFilenames	getAttachments	オプション	no	同じ名前の添付ファイルが複数ある場合に、一意のファイル名を生成するかどうかを指定するブール値です。同じ名前を持つ添付ファイルが複数存在し、このオプションが yes の場合、ColdFusion は、競合するファイル名の後ろ (拡張子の前) に番号を付加します。たとえば、myfile.txt という名前の添付ファイルが 3 つある場合は、myfile.txt、myfile1.txt、myfile2.txt という名前で保存されます。
name	get getAttachments	必須		返されたタスクレコード、または取得した添付ファイルに関する情報を格納する ColdFusion クエリー変数の名前です。返されるデータの詳細については、「使用方法」を参照してください。
result	create	オプション		作成するタスクの UID を格納する変数の名前です。他のアクションでは、uid 属性でこの値を使用してアクションの対象となるタスクを識別します。

属性	アクション	必須 / オプション	デフォルト	説明
serverVersion		オプション	2007	Microsoft Exchange Server のバージョンを指定します。 有効な値は次のとおりです。 • 2003 • 2007 • 2010 詳細を指定しない場合は、デフォルトで 2007 が使用されます。 指定した値は、アプリケーションレベルで指定した値よりも優先されます。
task	create modify	必須		設定または変更するタスクのプロパティとその値を含む構造体への参照です。この属性はシャープ記号 (#) で囲んで指定する必要があります。 イベント構造体の詳細については、「使用方法」を参照してください。
uid	delete getAttachments modify	必須		アクションの対象となるタスクを一意に識別する Exchange UID 値です。大文字と小文字は区別されます。delete アクションの場合は、この属性で UID 値のカンマ区切りリストを指定できます。deleteAttachments、getAttachments、および modify アクションの場合は、1 つの UID 値のみを指定できます。

create または modify アクションを指定する場合は、タスクに関する情報を含む構造体を task 属性で指定する必要があります。この構造体には次のフィールドを格納できます。設定または変更するフィールドのみを含めます。

列	説明
ActualWork	分単位の時間です。ゼロ未満にすることはできません。
Attachments	タスクに含める添付ファイルのパス名です。複数のファイルを指定する場合、Windows ではセミコロン (;)、UNIX および Linux ではコロン (:) を使用して各項目を区切ります。絶対パスを使用する必要があります。 modify アクションで添付ファイルを指定した場合は、指定した添付ファイルが既存の添付ファイルに追加されます。既存の添付ファイルが削除されることはありません。
Categories	カテゴリのカンマ区切りリストです。リスト内のすべてのカテゴリと一致するタスクが検索されます。
BillingInfo	文字列です。
Companies	文字列です。
DateCompleted	ColdFusion で有効な日付形式の文字列です。 このフィールドを省略した場合、Status フィールドを Completed に設定するか、PercentCompleted フィールドを 100 に設定すると、この値は現在の日付に設定されます。 この日付を設定すると、Status フィールドは Completed に設定され、PercentCompleted フィールドは 100 に設定されます。
DueDate	ColdFusion で有効な日付形式の文字列です。
Message	タスクの説明を含む文字列です。
Mileage	文字列です。

列	説明
PercentCompleted	0 ～ 100 の範囲の数値です。 このフィールドを 100 に設定すると、次の値が設定されます。 • Status の値は Completed に設定されます。 • DateCompleted の値が設定されていない場合は、現在の日付に設定されます。 このフィールドを 100 未満の値に設定すると、次の値が設定されます。 • Status フィールドが Completed に設定されている場合は、In_Progress に変更されます。 • DateCompleted の値はクリアされます。
Priority	次のいずれかの値になります。 • low • normal • high
ReminderDate	ColdFusion で有効な日付形式の文字列です。
StartDate	ColdFusion で有効な日付形式の文字列です。タスクを作成すると、デフォルト値 defaults は現在の日付になります。
Status	使用できる値は次のとおりです。Not_Started、In_Progress、Completed、Waiting、または Deferred。 このフィールドを省略し、PercentCompleted の値を 100 未満に設定すると、Status の値は In_Progress に設定されます。 このフィールドを Completed に設定すると、次の値も設定されます。 • PercentCompleted の値は 100 に設定されます。 • DateCompleted の値が設定されていない場合は、現在の日付に設定されます。 このフィールドを Completed 以外の値に設定すると、次の値も設定されます。 • PercentCompleted フィールドが 100 の場合は、PercentCompleted の値が 0 にリセットされます。 • DateCompleted の値は 0 に設定されます。
Subject	文字列です。
TotalWork	分単位の時間です。ゼロ未満にすることはできません。

使用方法

cfexchangetask タグは、Exchange サーバー上のタスクレコードを管理します。次のアクションを実行するには、cfexchangetask タグを使用します。

- タスクを作成する。
- タスクを削除する。
- 姓、役職、自宅電話番号などのフィルタ条件に一致するタスクレコードを取得する。
- 特定のタスクレコードの添付ファイルを取得する。
- 既存のタスクを修正する。

このタグを使用するには、Exchange サーバーに接続する必要があります。複数のタスクレコードを作成する場合のように、Exchange サーバーと対話するタグを複数使用する場合は、cfexchangeconnection タグを使用して継続的接続を作成します。その場合は、個々の cfexchangetask タグで接続識別子を指定します (カレンダーエントリ、連絡先、メールなどにもアクセスする場合はその他の ColdFusion Exchange タグでも接続識別子を指定します)。これにより、個々のタグに対して接続を作成して閉じる必要がなくなるので、システムの負荷が減少します。

また、ColdFusion が 1 個の cfexchangetask タグを処理する間だけ存続する一時的接続を作成することもできます。その場合は、cfexchangetask タグで接続属性を直接指定します。接続属性の詳細については、cfexchangeconnection タグを参照してください。

delete アクション

delete アクションを指定する場合は、削除するタスクを識別する Exchange UID のカンマ区切りリストを uid 属性で指定します。UID 値を確認するには、適切なフィルタ式を指定した get アクションを使用します。

cfexchangetask タグで指定した UID がすべて無効な場合はエラーが発生します。1 つでも有効な UID が含まれる場合は、無効な UID が無視され、有効な UID に対応する項目が削除されます。

get アクション

get アクションを指定すると、name 属性で指定したクエリーオブジェクトに、取得されたタスクごとのレコードが格納されます。クエリーオブジェクトには task 属性の構造体と同じ名前およびデータ形式を持つフィールドが含まれます。ただし、次の点が異なります。

- クエリーオブジェクトにはブール型の HasAttachment 列が含まれ、Attachments 列は含まれません。HasAttachment フィールドの値が yes の場合は、getAttachments アクションを使用して添付ファイルを取得します。
- クエリーオブジェクトには Exchange サーバー内のタスクの UID を含む UID 列が追加されます。getAttachments、delete、および modify アクションでは、この uid 属性の値を使用してタスクを識別します。
- クエリーオブジェクトには HtmlMessage 列が追加されます。Message 列にはタスクに関するプレーンテキスト形式の説明が格納され、HtmlMessage 列には HTML 形式の説明が格納されます。

取得するメッセージを指定するには、子の cfexchangefilter タグを使用します。詳細については、cfexchangefilter を参照してください。

getAttachments アクション

getAttachments アクションを使用する場合は、1 つの UID と name 属性を指定します。cfexchangetask タグは、name 属性で指定したクエリーオブジェクトに情報を格納します。各レコードには、指定したタスクの添付ファイルに関する次の情報が含まれます。

列名	説明
attachmentFileName	添付ファイルのファイル名です。
attachmentFilePath	サーバー上の添付ファイルの絶対パスです。attachmentPath 属性を省略すると、この列には空の文字列が入ります。
CID	添付ファイルの content-ID です。通常は、メッセージにイメージを埋め込むために HTML の img タグ内で使用します。
mimeType	添付ファイルの MIME タイプです (text/html など)。
isMessage	添付ファイルがメッセージかどうかを指定するブール値です。
size	添付ファイルのサイズです (単位 : バイト)。

添付ファイルは attachmentPath 属性で指定したディレクトリに保存されます。attachmentPath 属性を省略すると、添付ファイル自体は取得されず、添付ファイルに関する情報が取得されます。この方法を使用すると、添付ファイルの取得に伴うオーバーヘッドを発生させることなく、添付ファイルを調べることができます。

次のシンタックスを使用して、メモリ内の attachmentPath ディレクトリを指定します。メモリ内のファイルはディスクに書き込まれないため、一時的データの処理が早くなります。

```
attachmentpath = "ram:///path"
```

パスには、ram:///petStore/orders/messageAttachments のように複数のディレクトリを含めることができます。ファイルを指定する前に、パスに含まれるすべてのディレクトリを作成しておく必要があります。メモリ内のファイルの使用方法について詳しくは、『ColdFusion アプリケーションの開発』の Working with in-memory files を参照してください。

getAttachments アクションは、Exchange のサーバー設定で EWS (Exchange Web サービス) の認証が basic に設定されている場合にのみ有効です。IWA (Integrated Windows Authentication) はサポートされていません。

modify アクション

modify アクションを指定する場合は、uid 属性で 1 つの Exchange UID を指定する必要があります。task 構造体では、変更するフィールドのみを指定する必要があります。指定しないフィールドは変更されません。タスク構造体の内容については、「属性」を参照してください。

タスクに添付ファイルがある場合、タスクを修正するときに添付ファイルを指定すると、新しい添付ファイルが既存の添付ファイルに追加されます。添付ファイルが置き換えられることはありません。添付ファイルを削除するには、deleteAttachments アクションを使用します。

例

次の例では、一時的な接続を使用して 1 つのタスクを作成しています。

```
 <!--- Create a structure with the task fields --->
<cfscript>
    stask=StructNew();
    stask.Priority="high";
    stask.Status="Not_Started";
    stask.DueDate="3:00 PM 09/14/2007";
    stask.Subject="My New Task";
    stask.PercentCompleted=0;
    Message="Do this NOW!";
</cfscript>

<!--- Create the task using a transient connection. --->
<cfexchangetask action="create"
    username ="#user1#"
    password="#password1#"
    server="#exchangeServerIP#"
    task="#stask#"
    result="theUID">

<!--- display the UID to confirm that the action completed. --->
<cfdump var="#theUID#">
```

cfexecute

説明

サーバーコンピュータ上で ColdFusion 開発者が指定したすべての処理を実行します。

カテゴリ

拡張タグ、フロー制御タグ

シンタックス

```
<cfexecute
    name = "application name"
    arguments = "command line arguments"
    outputFile = "output filename"
    timeout = "timeout interval"
    variable = "variable name">
    ...
</cfexecute>
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfcollection、cfindex、cfobject、cfreport、cfsearch、cfwddx

履歴

ColdFusion MX 6.1:

- variable 属性が追加されました。
- outputFile 属性のファイルパスの動作が変更されました。outputFile 属性で絶対ファイルパスを指定しなかった場合は、ColdFusion のテンポラリディレクトリを基準とした相対パスと見なされます。

属性

属性	必須 / オプション	デフォルト	説明
name	必須		実行するアプリケーションの絶対パスです。 Windows では、C:¥myapp.exe のように拡張子を指定します。
arguments	オプション		アプリケーションに渡されるコマンドライン変数です。文字列として指定された場合は、次のように処理されます。 • Windows の場合 : プロセスコントロールサブシステムに渡されて、解析されます。 • UNIX の場合 : 引数の配列に置き換えられます。デフォルトのトークンセパレータはスペースです。スペースが埋め込まれた引数を二重引用符 (") で区切ることもできます。 配列として渡された場合は、次のように処理されます。 • Windows の場合 : 要素はトークンの文字列に連結され、スペースで区切られます。プロセスコントロールサブシステムに渡されて、解析されます。 • UNIX の場合 : 要素は exec() 引数の配列にコピーされます。

属性	必須 / オプション	デフォルト	説明
outputFile	オプション		プログラム出力の送信先となるファイルです。outputFile 属性または variable 属性を指定しなかった場合は、出力は呼び出し元のページに表示されます。 絶対パス (ドライブ文字とコロン、あるいはスラッシュまたは円記号から始まるパス) を指定しない場合は、GetTempDirectory 関数から返される ColdFusion テンポラリディレクトリを基準とする相対パスを指定します。
timeout	オプション	0	生成されたプログラムからの出力を ColdFusion が待機する時間を秒単位で指定します。 • 0: 非ブロックモードに相当します。 • 非常に大きい値 : ブロックモードに相当します。 値が 0 の場合、次の処理が行われます。 • 処理が開始され、すぐに返されます。プログラムの出力が表示される前に、呼び出しページに制御権が返される場合があります。プログラムの出力を確実に表示するには、値を 2 以上に設定してください。 • outputFile 属性を指定しない場合、プログラムの出力は破棄されます。
variable	オプション		プログラム出力を保管するための変数です。outputFile 属性または variable 属性を指定しなかった場合、出力は呼び出し元のページに表示されます。

使用方法
cfexecute の開始タグと終了タグの間には、他の ColdFusion のタグや関数を指定しないでください。cfexecute タグはネストできません。

例外
次の例外が発生します。

- アプリケーション名が見つからない場合 : java.io.IOException
- ColdFusion の実行スレッドを認可されたユーザーが、プロセスの実行権限を持っていない場合 : セキュリティ例外

タイムアウト値は、0 から、オペレーティングシステムでサポートされる最大タイムアウト値までの値でなければなりません。

例
```
 <h3>cfexecute</h3>
<p>This example executes the Windows NT version of the netstat network monitoring program, and places its output in a file.

<cfexecute name = "C:\WinNT\System32\netstat.exe"
    arguments = "-e"
    outputFile = "C:\Temp\output.txt"
    timeout = "1">
</cfexecute>
```

cfexit

説明
現在実行中の CFML カスタムタグの処理を中止します。また、現在実行中の CFML カスタムタグ内のページを終了するか、現在実行中の CFML カスタムタグ内のコードセクションを再実行します。

カテゴリ

デバッグタグ、フロー制御タグ

シンタックス

```
<cfexit
    method = "method">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfabort、cfbreak、cfexecute、cfif、cflocation、cfloop、cfswitch、cfthrow、cftry、『ColdFusion アプリケーションの開発』の cfabort and cfexit

属性

属性	必須 / オプション	デフォルト	説明
method	オプション	exitTag	• exitTag: 現在実行中のタグの処理を中止します。 • exitTemplate: 現在実行中のタグのページを終了します。 • loop: 現在実行中のタグの本文を再実行します。

使用方法

ベースページ内やインクルードページ内など、カスタムタグのコンテキスト外で cfexit タグが検出されると、cfabort と同様の処理が行われます。cfexit タグを使用すると、カスタムタグ内のエラーチェックと検証のための論理式を簡略化することができます。

cfexit タグの機能は、その位置と実行モードによって異なります。

method 属性の値	cfexit 呼び出しの場所	動作
exitTag	ベースページ	処理を終了します。
	Execution mode = Start	終了タグの後から処理を継続します。
	Execution mode = End	終了タグの後から処理を継続します。
exitTemplate	ベースページ	処理を終了します。
	Execution mode = Start	本文内の最初のサブタグから処理を継続します。
	Execution mode = End	終了タグの後から処理を継続します。
loop	ベースページ	エラー
	Execution mode = Start	エラー
	Execution mode = End	本文内の最初のサブタグから処理を継続します。

例

```
 <h3>cfexit Example</h3>
<p>cfexit can be used to abort the processing of the currently executing CFML custom tag. Execution resumes
following the invocation of the custom tag in the page that called the tag.
<h3>Usage of cfexit</h3>
<p>cfexit is used primarily to perform a conditional stop of processing inside a custom tag. cfexit returns
control to the page that called that custom tag, or in the case of a tag called by another tag, to the
calling tag.</p>

<!--- cfexit can be used within a CFML custom tag, as follows: --->
<!--- Place this code (uncomment the appropriate sections) within the customtags directory. --->

<!--- MyCustomTag.cfm --->
<!--- This simple custom tag checks for the existence of myValue1 and myValue2. If they are both defined,
the tag adds them and returns the result to the calling page in the variable "result". If either or both of
the expected attribute variables is not present, an error message is generated, and cfexit returns control
to the calling page. --->

<!--- <cfif NOT IsDefined("attributes.myValue2")>
            <cfset caller.result = "Value2 is not defined">
            <cfexit method = "exitTag">
     <cfelseif NOT IsDefined("attributes.myValue1")>
            <cfset caller.result = "Value1 is not defined">
            <cfexit method = "exitTag">
     <cfelse>
            <cfset value1 = attributes.myValue1>
            <cfset value2 = attributes.myValue2>
            <cfset caller.result = value1 + value2>
     </cfif> --->
<!--- End MyCustomTag.cfm --->

<!--- Place this code within your page --->

<!--- <p>The call to the custom tag, and then the result: </p>
<CF_myCustomTag
        myvalue2 = 4>
<cfoutput>#result#</cfoutput> --->
<p>If cfexit is used outside a custom tag, it functions like a cfabort. For example, the text after this
message is not processed:</p>
<cfexit>
<p>This text is not executed because of the cfexit tag above it.</p>
```

タグ f

cffeed

説明

RSS または Atom フィードの読み込みまたは作成を行います。このタグでは、RSS バージョン 0.90、0.91、0.92、0.93、0.94、1.0、2.0、および Atom 0.3 または 1.0 を読み込むことができ、RSS 2.0 または Atom 1.0 のフィードを作成できます。

カテゴリ

通信タグ、インターネットプロトコルタグ

シンタックス

```
create
    required
    <cffeed
    action = "create"
    name = "#structure#"
        One or both of the following:outputFile = "path"
    xmlVar = "variable name"
    optional
    overwrite = "no|yes">
    escapeChars = "true|false">

OR

    required
    <cffeed
    action = "create"
    properties = "#metadata structure#"
    query = "#items/entries query name#"
        One or both of the following:outputFile = "path"
    xmlVar = "variable name"
    optional
    columnMap = "mapping structure"
    overwrite = "no|yes">

read
    required
    <cffeed
    source = "feed source"
        One or more of the following:name = "structure"
    properties = "metadata structure"
    query = "items/entries query"
    outputFile = "path"
    xmlVar = "variable name"
    optional
    action = "read"
    enclosureDir = "path"
    ignoreEnclosureError = "no|yes"
    overwrite = "no|yes"
    overwriteEnclosure = "no|yes"
    proxyServer = "IP address or server name for proxy host"
    proxyPassword = "password for the proxy host"
    proxyPort = "port of the proxy host"
    proxyUser = "user name for the proxy host"
    timeout = "request time-out in seconds"
    userAgent = "HTTP user agent identifier">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

履歴

ColdFusion 8: このタグが追加されました。

ColdFusion 9：escapeChars 属性が新しく追加されました。

属性

属性	必須 / オプション	デフォルト	説明
action	オプション	read	実行するアクションです。次のいずれかの値を指定します。 • create: RSS 2.0 または Atom 1.0 フィードの XML ドキュメントを作成して、変数に保存するかファイルに書き出します (両方を行うこともできます)。 • read: URL または XML ファイルから RSS または Atom フィードを読み込んで解析し、構造体またはクエリーに保存します。別の構造体にあるフィードメタデータを取得することもできます。
columnMap	オプション		query 属性が指定された create アクションにのみ適用されます。 query 属性で指定されたオブジェクト内の列の名前と ColdFusion フィード形式の列の間のマッピングを指定する構造体です (「クエリーオブジェクトのルール」を参照)。 各フィールドのキーは列の名前にする必要があります (「クエリーオブジェクトのルール」の表を参照)。フィールドの値は、create アクションへの入力として使用されるクエリーオブジェクト内で対応する列の名前にする必要があります。
enclosureDir	オプション		read アクションにのみ適用されます。 読み込み中のフィードで使用可能なエンクロージャを保存するディレクトリのパスです。このパスは、絶対パスで指定しても、CFML ファイルからの相対パスで指定してもかまいません。 指定したディレクトリが存在しない場合は、エラーになります。この属性を省略するとエンクロージャは保存されません。現在のページが存在するディレクトリを指定するには、この属性をピリオド (.) に設定します。
escapeChars	オプション	false	create アクションにのみ適用されます。 この属性が true の場合、W3C 仕様によりすべての無効な文字をエスケープまたは置換します。 **メモ :** UTF-8 エンコードの一部ではない日本語は置換されます。UTF-8 ではない日本語はフィード内にそのまま残ります。 この属性が false の場合は無効な文字をエスケープせず、フィードを生成します。JDOM がこれらの無効な文字でファイルの書き込みに失敗した場合、エラーメッセージ "Invalid Character in Input" が表示されます。
ignoreEnclosureError	オプション	no	この属性が yes の場合、ColdFusion はすべてのエンクロージャの保存を試みます。いずれかのエンクロージャをダウンロードするときにエラーが発生しても、他のエンクロージャのダウンロードが続行され、サーバーログにエラー情報が書き込まれます。 この属性が no の場合は、エンクロージャのダウンロード中にエラーが発生すると、すべてのエンクロージャのダウンロードが停止されてエラーになります。 **メモ :** 指定されたタイプのエンクロージャをダウンロードすることが Web サーバーで許可されていない場合は、エンクロージャエラーが発生する可能性があります。
name	「メモ」を参照		完全なフィードデータを含む構造体です。次のデータを含みます。 • read アクションの出力。 • 作成するフィードの入力定義。 create アクションの name 属性を指定するときは、シャープ記号 (#) で囲みます。 詳細については、「name 構造体と properties 構造体のルール」を参照してください。
outputFile	「メモ」を参照		XML テキストとしてフィードを書き込むファイルのパスです。 このパスは、絶対パスで指定しても、CFML ファイルからの相対パスで指定してもかまいません。

属性	必須 / オプション	デフォルト	説明
overwrite	オプション	no	XML フィードファイルが存在する場合に上書きするかどうかを指定します。この属性が yes に設定されていない場合、cffeed タグが既存のファイルにフィードを書き込もうとすると、エラーが発生します。
overwriteEnclosure	オプション	no	read アクションにのみ適用されます。 エンクロージャディレクトリに既存のファイルがある場合に、ファイルを上書きするかどうかを指定します。この属性が yes に設定されていない場合、cffeed タグが既存のファイルにフィードを書き込もうとすると、エラーが発生します。
properties	「メモ」を参照		フィードメタデータ (フィード全体に関する情報) を含む構造体です。次のいずれかを含みます。 • read アクションの出力。 • create アクションへの入力。 properties 属性と query 属性を組み合わせると、完全なフィード情報を提供できます。 create アクションの properties 属性を指定するときは、シャープ記号 (#) で囲みます。 詳細については、「name 構造体と properties 構造体のルール」を参照してください。
proxyPassword	オプション		プロキシサーバーで要求されるパスワードです。
proxyPort	オプション	80	プロキシサーバー上で接続するポートです。
proxyServer	オプション		リクエストの送信先となるプロキシサーバーのホスト名または IP アドレスです。
proxyUser	オプション		プロキシサーバーに提供するユーザー名です。
query	「メモ」を参照		フィード内の Atom エントリまたは RSS 項目を含むクエリーオブジェクトです。次のいずれかを含みます。 • read アクションの出力。 • create アクションへの入力。 properties 属性と query 属性を組み合わせると、完全なフィード情報を提供できます。 create アクションの query 属性を指定するときは、シャープ記号 (#) で囲みます。 詳細については、「クエリーオブジェクトのルール」を参照してください。
source	必須		read アクションにのみ適用されます。 フィードの URL またはフィードコンテンツを含む XML ファイルのパスです。パスは、絶対パスで指定しても、CFML ファイルからの相対パスで指定してもかまいません。
timeout	オプション	リクエストタイムアウト	フィードソースからのレスポンスを待機する秒数です。値が 0 の場合は、リクエストがタイムアウトしないことを示します。 ColdFusion は、ColdFusion Administrator の [サーバーの設定]-[設定] ページのリクエストタイムアウトの設定をデフォルトで使用します。
userAgent	オプション	Cold Fusion	HTTP User-Agent リクエストヘッダフィールドに配置されるテキストです。リクエストクライアントソフトウェアを識別するために使われます。
xmlVar	「メモ」を参照		読み込んだフィードまたは作成したフィードを XML テキストとして保存する変数です。

使用方法

ファイル属性とディレクトリ属性の指定
次のシンタックスを使用して、メモリ内のファイルまたはディレクトリを指定します。メモリ内のファイルはディスクに書き込まれないため、一時的データの処理が早くなります。

```
ram:///filepath
```

ファイルパスには、複数のディレクトリを含めることができます (例 : ram:///petStore/images/poodle.jpg)。ファイルを指定する前に、パスに含まれるディレクトリを作成しておく必要があります。メモリ内のファイルの使用方法について詳しくは、『ColdFusion アプリケーションの開発』の Working with in-memory files を参照してください。

フィード情報の設定と取得
cffeed タグを使用すると、さまざまな方法でフィードデータを指定し、保存できます。

使用方法
メモリ内のファイルを指定するには、次のシンタックスを使用します。この場合、ディスクには書き込まれません。メモリ内のファイルを使用すると、一時的データの処理が速くなります。

```
ram:///filepath
```

ファイルパスには、ram:///petStore/images/poodle.jpg のようにディレクトリを含めることができます。ファイルを指定する前に、パスに含まれるディレクトリを作成しておく必要があります。メモリ内のファイルの使用方法について詳しくは、『ColdFusion アプリケーションの開発』の Working with in-memory files を参照してください。

フィードを作成する場合

- 次のいずれかの方法でフィードデータを指定します。
  - name 属性で指定した単一の構造体にすべてのメタデータとエントリまたは項目データを配置する方法。
  - properties 構造体で指定した構造体にメタデータを配置し、query 属性で指定したクエリーオブジェクトにエントリまたは項目を行として配置する方法。
- 次のいずれか、または両方に結果のフィード XML を保存します。
  - OutputFile 属性で指定したファイル。cffeed タグでは、データは UTF-8 エンコードで保存されます。
  - xmlVar 属性で指定した変数。

フィードを読み込む場合

次の形式を組み合わせてフィードデータを保存できます。

- name 属性で指定した単一の構造体にすべてのエントリまたは項目データとメタデータを保存する方法。
- query 属性で指定したクエリーオブジェクトにエントリまたは項目を行として保存する方法。
- properties 構造体で指定した構造体にメタデータを保存する方法。
- OutputFile 属性で指定したファイルにフィード XML を書き込む方法。cffeed タグでは、データは UTF-8 エンコードで保存されます。
- xmlVar 属性で指定した ColdFusion XML 変数にフィード XML を保存する方法。

フィードデータを保存するとき、メタデータとエントリまたは項目データの両方を保存する必要はありません。properties 属性のみを指定することも、query 属性のみを指定することもできます。

name 構造体と properties 構造体のルール

name 構造体と properties 構造体は、次のルールに準拠している必要があります。特定のメタデータエントリの必要条件の詳細については、「フィードメタデータの表現」を参照してください。

- 構造体のキー名はすべて、対応するフィード要素名と一致している必要があります。ただし、version フィールドと encoding フィールドは例外です。また、Dublin Core および Apple® iTunes の拡張要素のキー名はそれぞれ、DC_ および ITUNES_ で始まります。
- properties 構造体のフィールドは、name 構造体のメタデータフィールドと同じです。
- フィードの読み込み時には、フィード内に存在する要素と属性の値のみが構造体に含まれます。create アクションの必要条件については、「フィードの作成」を参照してください。
- フィード内に同じタイプの要素 (entry、item、link など) が複数存在する可能性がある場合、name または property の構造体には、すべての要素のデータが含まれる単一のエントリが存在します。構造体のエントリの形式は次のとおりです。
  - キーは要素名 (例 : item) です。
  - 値は構造体の配列です。
  - 配列に含まれる各構造体は 1 つの要素を表します。

  ColdFusion では、要素が 1 つしか存在しない場合でも配列が使用されます。たとえば、Atom フィードに link 要素が 1 つだけ含まれている場合は、name 属性の構造体で次の形式を使用してその要素を指定する必要があります。

  ```
  structureName.link[1]
  ```

  たとえば、Atom 1.0 フィードに link メタデータエントリを指定するには、次のコードを使用します。

  ```
  <cfset meta.link = arrayNew(1)>
  <cfset meta.link[1] = structNew()>
  <cfset meta.link[1].href = "http://www.myCo,com">
  ```
- 要素内に複数の属性または少なくとも 1 つの属性と値が含まれる可能性がある場合は、1 つの属性または値しか指定されていない場合でも、その要素は構造体として表現されます。
- 要素内に属性と値 (本文) が含まれる場合は、要素構造体の value フィールドで値が指定されています。たとえば、Atom フィードの 3 番目の entry の summary 要素のテキストは、次のような名前のフィールドに格納されます。

  ```
  structureName.entry[3].summary.value.
  ```
- cffeed タグを使用してフィードを読み込むと、次の形式で日付がレポートされます。

  Atom の場合 : W3C 形式 (例 : 2006-07-11T18:19:00Z)

  RSS の場合 : RFC 822 形式 (例 : Thu, 05 Oct 2006 18:19:00 GMT)。
- cffeed タグを使用してフィードを作成するときは、どちらのフィードタイプでも W3C 形式または RFC 822 形式を使用できます。それ以外にも、ColdFusion で使用可能な標準の日時形式をすべて使用できます。

クエリーオブジェクトのルール

query 属性で指定されるクエリーオブジェクトは、次のルールに準拠します。

- クエリーオブジェクトの形式は複数のフィード形式をサポートしますが、多くの場合、すべてのオプションのフィード属性や要素がフィードに含まれているわけではありません。このため次のようになります。
  - フィードを読み込む場合、返されるクエリーオブジェクトには標準のすべての RSS フィールドと Atom フィールドのエントリが含まれ、フィードタイプでサポートされていないフィールドのエントリも含まれます。そのフィードで使用されていない列には、空の文字列または未定義の値が格納されます。
  - フィードを読み込む場合、そのフィードに iTunes 拡張要素が含まれている場合は、クエリーオブジェクトにすべての iTunes 拡張フィールドが含まれます。また、Dublin Core 拡張要素が含まれている場合はすべての Dublin Core 拡張フィールドが含まれます。それ以外の場合、クエリー結果に拡張フィールドは含まれません。

  - フィードを作成する場合、定義するクエリーに設定する必要があるのは、そのフィードのデータが含まれている列のみです。使用しない列は省略できます。
- フィードのエントリまたは項目に同じ名前の子要素が複数ある場合は、クエリー列でカンマ区切りリストとして要素値が表現されます。RSS 2.0 の項目には、category 要素が複数存在する場合があります。Atom 1.0 のエントリには、category、author、contributor、および link 要素が複数存在する場合があります。Dublin Core 拡張機能では、すべての要素タイプがすべて複数で存在できます。
- 複数のインスタンスが存在する可能性があるエントリまたは項目要素の多くには複数の属性がありますが、すべてのインスタンスですべての属性が必要とされるわけではありません。エントリまたは項目に特定の要素のインスタンスが複数存在し、いずれかのインスタンスで属性が省略されている場合、省略された属性はリスト内でスペースとして表現されます。たとえば、次の XML では、Atom エントリに 3 つの author 要素が含まれています。

```
<author>
    <person>Anthony</person>
    <uri>http://www.MyCo.com</uri>
    <email>Tony@MyCo.com</email>
</author>
<author>
    <person>Beverly</person>
</author>
<author>
    <person>Cathy</person>
    <email>cathy@MyCo.com</email>
</author>
```

ColdFusion クエリーでは、これらの列が次のように表現されます。

AUTHOR_PERSON	AUTHOR_URI	AUTHOR_EMAIL
Anthony,Beverly,Cathy	http://www.MyCo.com, ,	Tony@MyCo.com, ,cathy@MyCo.com

次の表に、query 属性で指定される標準のクエリーオブジェクトの列を示します。RSS フィードに Dublin Core 拡張機能または iTunes 拡張機能のいずれかが含まれている場合、クエリーには追加の列が含まれます。これらのフィールドの詳細については、「Dublin Core の拡張機能」および「Apple iTunes 拡張機能」を参照してください。

列	Atom エントリ	RSS 項目
AUTHOREMAIL	author 要素の email 属性	author 項目
AUTHORNAME	author 要素の name 属性	使用しません
AUTHORURI	author 要素の uri 属性	使用しません
CATEGORYLABEL	category 要素の label 属性	category 項目の値
CATEGORYSCHEME	category 要素の scheme 属性	category 項目の domain 属性
CATEGORYTERM	category 要素の term 属性	使用しません
COMMENTS	使用しません	comments 項目の値
CONTENT	content 要素の値	description 項目の値
CONTENTMODE	content 要素の mode 属性 (Atom 0.3 のみ)	使用しません
CONTENTSRC	content 要素の src 属性	使用しません
CONTENTTYPE	content 要素の type 属性	使用しません
CONTRIBUTOREMAIL	contributor 要素の email 属性	使用しません
CONTRIBUTORNAME	contributor 要素の name 属性	使用しません

列	Atom エントリ	RSS 項目
CONTRIBUTORURI	contributor 要素の uri 属性	使用しません
CREATEDDATE	created 要素の値 (Atom 0.3 のみ)	使用しません
EXPIRATONDATE	使用しません	expirationDate 項目の値 (RSS 0.93 のみ)
ID	id 要素の値	guid 項目の値
IDPERMALINK	使用しません	guid 項目の ispermalink 属性
LINKHREF	link 要素の href 属性	enclosure 項目の url 属性
LINKHREFLANG	link 要素の hreflang 属性	使用しません
LINKLENGTH	link 要素の length 属性	enclosure 項目の length 属性
LINKREL	link 要素の rel 属性	使用しません
LINKTITLE	link 要素の title 属性	使用しません
LINKTYPE	link 要素の type 属性	enclosure 項目の type 属性
PUBLISHEDDATE	published 要素の値 (Atom 0.3 では issued)	pubDate 項目の値
RIGHTS	rights 要素の値 (Atom 0.3 では copyright)	使用しません
RSSLINK	使用しません	link 項目の値
SOURCE	使用しません	source 項目の値
SOURCEURL	使用しません	source 項目の url 属性
SUMMARY	summary 要素の値	使用しません
SUMMARYMODE	summary 要素の mode 属性 (Atom 0.3 のみ)	使用しません
SUMMARYSRC	整形式の Atom フィードの場合は空白になります。Atom 1.0 フィードの summary 要素で content 要素形式が使用されている場合にのみ、データが含まれます。	使用しません
SUMMARYTYPE	summary 要素の type 属性	使用しません
TITLE	title 要素の値	title 項目の値
TITLETYPE	title 要素の type 属性	使用しません
UPDATEDDATE	updated 要素の値 (Atom 0.3 では modified)	使用しません
URI	使用しません	RSS 1.0 の link 項目の rdf:about 属性
XMLBASE	content 要素の xml:base 属性	使用しません

フィードメタデータの表現

フィードの作成時には、name 構造体と properties 構造体で、RSS 2 フィードまたは Atom 1 フィードのすべての標準メタデータを「name 構造体と properties 構造体のルール」に記載されている形式で表すことができます。同様に、フィードの読み込み時には、受信したすべてのメタデータが構造体で表されます。name 構造体と properties 構造体の特定のフィードメタデータフィールドには、次のルールが適用されます。

- version フィールドでは、**format_versionNumber** という形式でフィードのバージョンが示されます。create アクションの場合は、atom_1.0 または rss_2.0 を指定します。RSS 0.91 フィードを読み込む場合、version フィールドの値は rss_0.91 ではなく rss_0.91U になります。

- feedExtension フィールドでは、フィードに iTunes または Dublin Core の拡張コンテンツが含まれているかどうかが示されます。有効な値は、itunes および DublinCore です。iTunes 拡張機能が含まれるフィードの作成時にこのフィールドを指定する必要はありません。拡張フィールドを指定していることは自動的に判別されます (Dublin Core 拡張機能が含まれるフィードは作成できません)。
- read アクションの場合は、encoding フィールドで XML のエンコーディング属性 (iso-8859-1 など) が示されます。create アクションの場合は、encoding フィールドを指定しないでください。現在、ColdFusion では、すべてのフィードが UTF-8 形式で生成され、encoding の値を指定しても無視されます。
- RSS フィードの場合、skiphours フィールドには 0 ～ 23 までの数値を 24 個まで格納できるカンマ区切りリストが含まれています。これらの数値は、フィードを読み込まない時刻を示します。0 は午前 0 時を示します。アプリケーションでこのフィールドを使用すると、いつフィードを読み込むかを指定できます。
- RSS フィードの場合、skipdays フィールドには曜日名を 7 個まで格納できるカンマ区切りリストが含まれています。これらの値は、フィードを読み込まない曜日を示します。有効な名前は、Monday、Tuesday、Wednesday、Thursday、Friday、Saturday、および Sunday です。アプリケーションでこのフィールドを使用すると、いつフィードを読み込むかを指定できます。

Dublin Core の拡張機能

Dublin Core 拡張要素は、フィードまたは項目に関する追加のメタデータを提供します。cffeed タグを使用すると、Dublin Core Metadata Element Set 仕様に準拠している要素が含まれているフィードを、メタデータ (チャネル要素) または項目要素として読み込むことができます。Dublin Core 拡張要素の詳細については、Dublin Core Metadata Element Set 仕様を参照してください。本マニュアル作成時点での、この仕様の入手先は http://dublincore.org/documents/dces/ です。

ColdFusion による Dublin Core 拡張機能のサポートには次の制限があります。

- Dublin Core 拡張要素が含まれているフィードは作成できません。
- トップレベルに含まれている Dublin Core 拡張要素 (メタデータ) の image 要素は取得できません。ColdFusion ではこれらの要素は無視されます。
- ColdFusion でサポートされるのは Dublin Core Metadata Element Set のみです。追加の Dublin Core Metadata Initiative の要素および要素精密化はサポートされていません。

フィードの項目に Dublin Core 拡張機能が含まれている場合、query 属性で指定したクエリーには、次の表に示されているすべての列が含まれます。フィードに Dublin Core 拡張要素が含まれていない場合、これらの列はクエリーに含まれません。DC_SUBJECT_TAXONOMURI 列と DC_SUBJECT_VALUE 列を除いて、各列名 (DC_ 接頭辞以外の部分) は、Dublin Core 拡張要素名に対応しています。

列	説明
DC_CONTRIBUTOR	リソースに寄与した人々または組織です。
DC_COVERAGE	リソースの内容の範囲です。
DC_CREATOR	このリソースを作成した個人または組織です。
DC_DATE	このリソースに関連する日付または日付と時刻です。
DC_DESCRIPTION	リソースの内容の要約です。
DC_FORMAT	リソースのファイル形式、物理メディア、または次元です。
DC_IDENTIFIER	リソースを明確に識別するために使用される文字列です。
DC_LANGUAGE	リソースが記述されている言語です。
DC_PUBLISHER	リソースを使用可能にした個人または組織です。
DC_RELATION	通常は関連リソースの識別子です。

列	説明
DC_RIGHT	リソースの所有権に関する情報です。
DC_SOURCE	このリソースの基となった資料への参照です。
DC_SUBJECT_TAXONOMYURI	Dublin Core の subject 要素の taxonomyURI 属性です。
DC_SUBJECT_VALUE	Dublin Core の subject 要素の値です。リソースのトピックを定義する文字列です。
DC_TITLE	リソースに使用する名前です。
DC_TYPE	リソースの種類またはジャンルです。

Dublin Core 要素が含まれているフィードのデータを構造体として取得すると、要素名はこの表に示したクエリー列名と同じになります。ただし Dublin Core の subject 要素の表現を除きます。subject 要素は、構造体の形式では dc_subject エントリとして表され、このエントリは構造体の配列で構成されます。配列に含まれる構造体にはキーがあり、その名前は要素の値の場合は value、taxonomyURI 属性の場合は taxonomyURI です。

Apple iTunes 拡張機能

cffeed タグを使用すると、Apple iTunes の RSS Podcast 仕様で定義された要素が含まれているフィードの作成または読み込みができます。iTunes 拡張形式の詳細については、Apple iTunes の RSS 仕様を参照してください。本マニュアル作成時点での、この仕様の入手先は http://www.apple.com/itunes/store/podcaststechspecs.html です。

作成できるのは、iTunes の RSS 拡張機能のサブセットのみが含まれるフィードです。フィードを読み込む場合、サポートされているサブセットに存在しない iTunes 拡張要素はすべて無視されます。

次の表に、サポートされている要素の構造体エントリの名前 (クエリー列名) を示します (これらの名前は、ITUNES_ 接頭辞と iTunes の拡張要素名で構成されます)。また、各要素がメタデータで使用されるか、個々の項目で使用されるか、または両方で使用可能であるかも示します。

要素	使用対象	説明
ITUNES_AUTHOR	両方	アーティスト名です。
ITUNES_BLOCK	両方	yes は、Podcast または項目 (エピソード) が表示されないように要求する値です。 ColdFusion でフィードを読み込む場合は、このフィールドの値が判別され、適切なアクションが実行されます。
ITUNES_DURATION	項目	項目の長さです。秒数または HH:MM:SS の形式です。
ITUNES_EXPLICIT	両方	項目に露骨な表現が含まれているかどうかを示す文字列です。有効な値は、yes、no、および clean です。
ITUNES_KEYWORDS	両方	iTunes Music Store での検索時に使用される語句のカンマ区切りのリストです。
ITUNES_SUBTITLE	両方	短い説明テキスト。通常は数語で指定します。
ITUNES_SUMMARY	両方	長い説明 (半角文字で最大 4000 文字) です。

name または properties 構造体では次のチャネル要素も使用できます。

要素	説明
itunes_category	iTunes Music Store のカテゴリを指定する構造体です。この構造体には次の 2 つのフィールドがあります。 • category • subcategory これらの要素名には itunes_ 接頭辞が付いていないことに注意してください。
itunes_image	Podcast の作品の URL です。
itunes_owner	Podcast の所有者と連絡を取るための連絡先情報が記載されている構造体です。この構造体には次の 2 つのフィールドがあります。 • itunes_email • itunes_mail

フィードの作成

フィードを作成するときは、name 構造体、または query オブジェクトと properties 構造体の組み合わせを使用して、フィードのコンテンツを指定します。cffeed タグは、フィード XML を生成して、xmlVar 属性で指定されている変数、または outputFile 属性で指定されているファイル (またはその両方) にフィード XML を保存します。

RSS 2.0 フィードを作成するには、name 構造体または properties 構造体で次のメタデータフィールドを指定する必要があります。その他の RSS2.0 メタデータフィールドと項目フィールドはすべて省略可能です。

- title
- link
- description
- version (rss_2.0 に設定する必要があります)

cffeed タグ自体は、作成した Atom フィード構造体に対して何のルールも適用しません。フィードが有効であるかどうかはユーザーが確認する必要があります。

ほとんどの場合、データベーステーブルでは、フィードの作成に使用する必要がある列名とは異なる列名が使用されています。したがって、必要な列名に入力クエリー列名をマッピングするには、columnmap 属性を使用します。この属性は構造体であり、キーは cffeed タグで必要な列名、値は対応する入力クエリー列です。

注意：データベース列名が大文字であるかどうかにかかわらず、入力クエリー列名は常に大文字にする必要があります。

次の例では、cfartgallery データソースの注文テーブルを使用してフィードを作成します。注文テーブルの ORDERDATE 列がクエリーの publisheddate 列に、ADDRESS 列が content 列に、以下同様にマッピングされます。次に、サンプルコードでは、生成されたクエリー XML が表示され、結果が表示されます。

```
<!--- Get the feed data as a query from the orders table. --->
<cfquery name="getOrders" datasource="cfartgallery">
    SELECT * FROM orders
</cfquery>

<!--- Map the orders column names to the feed query column names. --->
<cfset columnMapStruct = StructNew()>
<cfset columnMapStruct.publisheddate = "ORDERDATE">
<cfset columnMapStruct.content = "ADDRESS">
<cfset columnMapStruct.title = "CUSTOMERFIRSTNAME">
<cfset columnMapStruct.rsslink = "ORDERID">

<!--- Set the feed metadata. --->
<cfset meta.title = "Art Orders">
<cfset meta.link = "http://feedlink">
<cfset meta.description = "Orders at the art gallery">
<cfset meta.version = "rss_2.0">

<!--- Create the feed. --->
<cffeed action="create"
    query="#getOrders#"
    properties="#meta#"
    columnMap="#columnMapStruct#"
    xmlvar="rssXML">

<cfdump var="#XMLParse(rssXML)#">
```

フィードの読み込み

cffeed タグは、読み込んだフィードの妥当性を検証しません。無効なフィードや構造的に正しくないフィードも読み込めますが、無効なコンテンツは一部またはすべて無視されます。たとえば、Atom フィードに複数の rights 要素が含まれている場合 (このフィードは無効です)、cffeed タグは最初の要素のみを読み込んで後の要素を無視しますが、エラーにはなりません。

読み込まれるフィードの日付と時刻は、W3C 形式または RFC 822 形式で記述されている必要があります。また、ColdFusion では、iTunes Music Store で通常使用されている形式で iTunes 拡張日付を読み込むこともできます。

例

次の例では、RSS フィードを作成します。フィードの title、link、description 要素フィールドを入力します。また、項目ごとに title、link、description フィールドも入力します。2 番目以降の項目は省略可能です。このフィードは、CFML ページが存在するディレクトリにある feedTest サブディレクトリの createRSSOutput.xml ファイルに保存されます。

```
 <!--- Generate the feed when the user submits a filled in form. --->
<cfif isDefined("Form.Submit")>
    <cfscript>

        // Create the feed data structure and add the metadata.
        myStruct = StructNew();
        mystruct.link = form.link;
        myStruct.title = form.title;
        mystruct.description = form.description;
        mystruct.pubDate = Now();
        mystruct.version = "rss_2.0";

        /* Add the feed items. A more sophisticated application would use dynamic variables
            and support varying numbers of items. */
        myStruct.item = ArrayNew(1);
        myStruct.item[1] = StructNew();
        myStruct.item[1].description = StructNew();
        myStruct.item[1].description.value = form.item1text;
        myStruct.item[1].link = form.item1link;
        myStruct.item[1].pubDate = Now();
        myStruct.item[1].title = form.item1title;
        myStruct.item[2] = StructNew();
        myStruct.item[2].description = StructNew();
        myStruct.item[2].description.value = form.item2text;
        myStruct.item[2].link = form.item2link;
        myStruct.item[2].pubDate = Now();
        myStruct.item[2].title = form.item2title;

    </cfscript>

    <!--- Generate the feed and save it to a file and variable. --->
    <cffeed action = "create"
        name = "#myStruct#"
        outputFile = "feedTest/createRSSOutput.xml"
        overwrite = "yes"
        xmlVar = "myXML">

</cfif>

<!--- The user input form. --->
<cfform format="xml" preservedata="yes" style="width:500" height="700">
    <cfformitem type = "text"> Enter The Feed Metadata</cfformitem>
    <cfinput type = "text" label = "title" name = "title"
        style = "width:435" required = "yes"> <br />
    <cfinput type = "text" label = "link" name = "link"
```

```
        style = "width:435" required = "yes" validate = "url"> <br />
    <cftextarea name = "description"
        style = "width:435; height:70" required = "yes" />

    <cfformitem type = "text"> Enter Item 1</cfformitem>
    <cfinput type="text" label="title" name="item1title"
        style="width:435" required="yes"> <br />
    <cfinput type="text" label="link" name="item1link"
        style="width:435" required="yes" validate="url"> <br />
    <cftextarea name = "item1text"
        style = "width:435; height:70" required = "yes" /> <br />

    <cfformitem type = "text"> Enter Item 2</cfformitem>
    <cfinput type = "text" label = "title" name = "item2title" style = "width:435"> <br />
    <cfinput type = "text" label = "link" name = "item2link" style = "width:435"
        validate = "url"> <br />
    <cftextarea name = "item2text" style = "width:435; height:70" /> <br />

    <cfinput type = "Submit" name = "submit" value = "Submit" >
</cfform>
```

次に、RSS と Atom を処理する簡単なフィードリーダーアプリケーションの例を示します。このアプリケーションは、項目またはエントリごとにフィードタイトルを表示し、タイトルをリンクとして表示して、発行日と項目またはエントリのコンテンツを表示します。最初のアプリケーションで作成したフィードを読み込むには、最初のアプリケーションで作成したファイルの URL を入力します。たとえば、次のように入力します。
http://localhost:8500/cffeed/feedTest/createRSSOutput.xml

```
 <!--- Process the feed data if the user submitted the form --->
<cfif isDefined("Form.Submit")>
    <cffeed source = "#theURL#"
        properties = "myProps"
        query = "myQuery">

    <!--- Display the feed output.
            Use conditional logic for to handle different feed formats. --->
    <cfoutput>
        <h2>#myProps.title#</h2>
    </cfoutput>
    <cfoutput query = "myQuery">
        <cfif myProps.version IS "atom_1.0">
            <h3><a href = "#linkhref#">#title#</a></h3>
            <p><b>Published:</b> #DateFormat(publisheddate)#</p>
        <cfelse>
            <h3><a href = "#rsslink#">#title#</a></h3>
            <p><b>Published:</b> #publisheddate#</p>
        </cfif>
        <p>#content#</p>
    </cfoutput>
</cfif>

<!--- The form for specifying the feed URL or file --->
<cfform name = "SetFeed" preserveData = "yes">
        Enter Feed URL:
        <cfinput type = "text" size = "60" name = "theURL"><br><br>
        <cfinput type = "Submit" name = "submit" value = "Submit">
</cfform>
```

cffile

説明
サーバーファイルとの操作を管理します。

次のセクションでは、cffile タグのアクションについて説明します。

- cffile action = "append"
- cffile action = "copy"
- cffile action = "delete"
- cffile action = "move"
- cffile action = "read"
- cffile action = "readBinary"
- cffile action = "rename"
- cffile action = "upload"
- cffile action = "uploadAll"
- cffile action = "write"

注意：このタグを実行するには、ColdFusion Administrator でタグを有効にする必要があります。詳細については、『ColdFusion 設定と管理』を参照してください。

複数のカスタマが使用するサーバーで ColdFusion アプリケーションを実行する場合、cffile でアップロードまたは操作できるファイルについて、セキュリティを考慮する必要があります。詳細については、『ColdFusion 設定と管理』を参照してください。

カテゴリ
ファイル管理タグ

シンタックス
タグのシンタックスは action 属性の値によって異なります。それぞれの属性値のセクションを参照してください。

関連項目
cfdirectory

履歴
ColdFusion 10：accept 属性に対する変更

ColdFusion 9: uploadAll アクション

ColdFusion 8: cfimages の読み込みと書き込みが可能になりました。

ColdFusion MX 7:

- result 属性が追加されました。この属性では、結果パラメータを受け取る代替変数を指定することができます。action="upload" アクションの場合に使用します。
- action = "append" アクションと action = "write" アクションに fixnewline 属性が追加されました。

ColdFusion MX 6.1:

- ファイルパスの必要条件が変更されました。ファイルの絶対パスを指定しなかった場合は、GetTempDirectory 関数から返される ColdFusion のテンポラリディレクトリを基準とした相対パスになります。

- action="read" の動作が変更されました。BOM (Byte Order Mark) で始まるファイルの場合は、ColdFusion がそれを使って文字エンコードを判別します。
- action="upload"nameConflict="MakeUnique" の動作が変更されました。ColdFusion はファイル名の最後に増分番号を付加することで、固有の名前を作成します (1 番目のファイルには 1 を付け、2 番目のファイルには 2 を付ける、という具合です)。ColdFusion では、各ファイルの名前に "1" を付加することで固有の名前を作成していました (つまり、1、11、111、という具合です)。

ColdFusion MX:

- パス内のスラッシュの使い方が変更されました。UNIX システムでも Windows システムでも、パス内にスラッシュ (/) または円記号 (¥) を使用することができます。
- ファイル階層の必要条件が変更されました。このタグで操作するファイルやディレクトリを、Web サーバーのドキュメントディレクトリのルートの下に置く必要はありません。
- destination 属性のディレクトリパスの必要条件が変更されました。destination 属性で指定するディレクトリパスには、末尾のスラッシュを付ける必要はありません。
- attributes 属性の system 値が非推奨になりました。
- attributes 属性の temporary 値が非推奨になりました。ColdFusion では、これは normal と同義です。このタグは、これ以降のリリースでは機能しない可能性があります。
- action 属性の read、write、append、move オプションが変更されました。これらのオプションは、charset という新しい属性をサポートします。
- attributes 属性の archive 値は廃止されたため、効果はありません。

タグ本文でのファイルコンテンツの指定

cffile action = "append" および cffile action = "write" の場合、タグ本文でファイルコンテンツを指定できます。

タグ本文と output 属性の両方でファイルコンテンツを指定した場合は、エラーが発生します。

次の例では、本文で指定したテキストが、現在のディレクトリの myfile.txt に書き込まれます。

ファイルが存在しない場合は、新しい myfile.txt ファイルが作成されます。ファイルが存在する場合は上書きされます。

```
<!--- In this case, file content is passed via both - output attribute and inside tag body. Tag body will
get preference --->
<cfset filename = expandpath('./myfile.txt')>
<cftry>
    <cffile action="write" file="#filename#">
        some tag body
    </cffile>
    <cfset content = FileRead(filename)>
    <cfoutput>File Length = #Len(content)#</cfoutput>
<cfcatch type="any">
    <cfoutput>
        #cfcatch.message#
        <br>#cfcatch.detail#
        <br>
    </cfoutput>
</cfcatch>
</cftry>
```

同様に、cffile action="append" を使用すると、タグ本文の内容が myfile.txt ファイルの内容に追加されます。

空のファイルを作成するには、次のコードに示すように、タグ本文で最低 1 行の空の行を指定する必要があります。

```
<cffile action="write" file="#filename#">
    <!--- Leave a blank line here--->
</cffile>
```

ColdFusion 10 での accept 属性に対する変更

accept 属性では、任意またはすべての拡張子のカンマ区切りリスト、MIME タイプ、または MIME タイプのリストを値として使用します。

拡張子は、. の接頭辞を使用して指定します。正確に言うと、.txt のみがサポートされ、txt や *.txt、*.* はサポートされません。ただし、すべてのファイルを受け入れるワイルドカードとして * を使用できます。

- **ファイル名の拡張子を値として指定した場合**、ファイルがチェックされ、カンマ区切りリストで指定した拡張子のリストにファイルが一致するかどうかのみが確認されます。一致した場合、ファイルがアップロードされます。
- **値が MIME タイプまたは MIME タイプのリストであり**、strict 属性を true に設定した場合は、ファイルの最初の数バイトが読み込まれ、MIME タイプが判別されます。指定したものと MIME タイプが一致した場合はアップロードが実行され、そうでない場合はエラーが発生します。strict のデフォルト値は false です。
- **ファイル名の拡張子と MIME タイプの両方を指定し**、strict を false に設定した場合は、値を指定した順序に基づいて検証が行われます。例えば、最初の値が .txt である場合、.txt ファイルがアップロードされます。

  strict を true に設定した場合、拡張子は無視されます。

例：accept 属性を使用したファイル名の拡張子の検証

upload.cfm

```
<cftry>
    <cfset variables.URL =
"http://#cgi.server_name#:#cgi.server_port##getDirectoryFromPath(cgi.script_name)#_upload.cfm">
    <cfhttp method="Post" url="#variables.URL#">
        <cfhttpparam type="FILE" name="myfile" file="#expandpath('./sample.txt')#">
    </cfhttp>
    <cfoutput>#cfhttp.filecontent#</cfoutput>
<cfcatch>
    <cfoutput>
        #CFCATCH.message#
        <br>#CFCATCH.detail#
        <br>
    </cfoutput>
</cfcatch>
</cftry>
```

_upload.cfm

```
<cfset uploadDirectory = "#GetTempDirectory()#cf_upload">
<!--- create upload directory  --->
<cfif not directoryExists(uploadDirectory)>
    <cfdirectory action="create" directory="#uploadDirectory#">
</cfif>
<cftry>
    <cffile action="UPLOAD" destination="#uploadDirectory#" filefield="form.myFile"
            nameconflict="MAKEUNIQUE" accept=".txt">
<cfcatch>
    <cfoutput>
        #CFCATCH.message#
        <br>#CFCATCH.detail#
        <br>
    </cfoutput>
</cfcatch>
</cftry>
<!--- read directory --->
<cfdirectory action="LIST" directory="#uploadDirectory#" name="qFileList">
<cfoutput>#qFileList.recordcount#
    file(s) uploaded
    <br>
</cfoutput>
<!--- delete all the uploaded files--->
<!---
<cfloop query="qFileList">
    <cffile action="delete" file="#uploadDirectory#/#Name#">
</cfloop>
--->
<!--- delete directory --->
<!---
<cfdirectory action="delete" directory="#uploadDirectory#">
--->
```

この例では、accept を .txt に設定し、strict は指定しません（したがって、デフォルトの false になります）。そのため、.txt の拡張子を持つファイルのみがアップロード対象と見なされます。

ファイルがもともとは PDF（sample.pdf）で、名前が変更されて sample.txt になっている場合も、（strict を true に設定していないので）ファイルはアップロードされます。

strict=true を指定した場合、ファイルが初めから .txt であれば（かつ他のタイプから名前が変更されたファイルでなければ）、正しい MIME タイプが指定されている場合にのみ、ファイルがアップロードされます。つまり、strict=true の場合は、accept 属性で MIME タイプを指定する必要があります。そのため、_upload.cfm で明示的に accept="text/plain" を記述する必要があります。

例 2：MIME タイプの使用

例 1 を変更して、_upload.cfm で次のように accept="application/pdf" を指定します。

```
<cffile action="UPLOAD"
    destination="#uploadDirectory#"
    filefield="form.myFile"
    nameconflict="MAKEUNIQUE"
    accept="application/pdf"
    strict=true>
```

strict = true であるので、タイプが PDF のファイルのみがアップロード対象と見なされます。

別のファイルを使用するには、upload.cfm の次のセクションを変更します。

```
<cfhttpparam type="FILE" name="myfile" file="#expandpath('./sample.txt')#">
```

例 3：拡張子と MIME タイプの両方を使用

例 1 を変更して、_upload.cfm で次のように指定します。

```
<cffile
    action="UPLOAD"
    destination="#uploadDirectory#"
    filefield="form.myFile"
    nameconflict="MAKEUNIQUE"
    accept=".txt, application/pdf"
    strict=true>
```

strict = true であるので、PDF ファイルのみがアップロードされます。

strict を false に設定すると、両方のファイルがアップロードされます。

別のファイルを使用するには、upload.cfm の次のコードを変更します。

```
<cfhttpparam type="FILE" name="myfile" file="#expandpath('./sample.txt')#">
```

例

```
 <!--- This shows how to write, read, update, and delete a file using CFFILE.
This is a view-only example. --->
<!---
<cfif IsDefined("form.formsubmit") is "Yes">
    <!--- The form has been submitted, now do the action. --->
    <cfif form.action is "new">
        <!--- Make a new file. --->
        <cffile action="Write"
            file="#GetTempDirectory()#foobar.txt"
            output="#form.the_text#">
    </cfif>
    <cfif form.action is "read">
        <!--- Read existing file. --->
        <cffile action="Read"
            file="#GetTempDirectory()#foobar.txt"
            variable="readText">
    </cfif>

    <cfif form.action is "add">
        <!--- Update existing file. --->
        <cffile action="Append"
            file="#GetTempDirectory()#foobar.txt"
            output="#form.the_text#">
    </cfif>

    <cfif form.action is "delete">
        <!--- Delete existing fil. --->
        <cffile action="Delete"
            file="#GetTempDirectory()#foobar.txt">
    </cfif>
</cfif>
<!--- Set some variables. --->
<cfparam name="fileExists" default="no">
<cfparam name="readText" default="">
<!--- First, check whether canned file exists. --->
<cfif FileExists("#GetTempDirectory()#foobar.txt") is "Yes">
    <cfset fileExists="yes">
</cfif>
<!--- Now, make the form that runs the example. --->
<form action="index.cfm" method="POST">
<h4>Type in some text to include in your file:</h4> <p>
<cfif fileExists is "yes">
    <p>A file exists (foobar.txt, in <cfoutput>#GetTempDirectory()#</cfoutput>).
```

```
    You may add to it, read from it, or delete it. </p>
</cfif>
<!--- If reading from a form, let that information display in textarea. --->
<textarea name="the_text" cols="40" rows="5">
    <cfif readText is not "">
        <cfoutput>#readText#</cfoutput>
    </cfif></textarea>
<!--- Select from the actions depending on whether the file exists. --->
<select name="action">
<cfif fileExists is "no">
    <option value="new">Make new file
</cfif>
<cfif fileExists is "yes">
    <option value="add">Add to existing file
    <option value="delete">Delete file
    <option value="read">Read existing file
</cfif>
</select>
<input type="Hidden" name="formsubmit" value="yes">
<input type="Submit" name="" value="make my changes">
</form> --->
```

cffile action = "append"

説明

サーバー上のテキストファイルにテキストを追加します。

シンタックス

```
<cffile
    action = "append"
    file = "full pathname"
    output = "string"
    addNewLine = "yes|no"
    attributes = "file attributes list"
    charset = "characterset option"
    fixnewline = "yes|no"
    mode = "mode">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfdirectory

履歴

メインの cffile タグのページの「履歴」を参照してください。

属性

属性	必須 / オプション	デフォルト	説明
action	必須		このタグが実行するファイル操作のタイプです。
file	必須		output 属性のコンテンツの追加先となるファイルのパス名です。 絶対パス (ドライブ文字とコロン、あるいはスラッシュまたは円記号から始まるパス) を指定しない場合は、GetTempDirectory 関数から返される ColdFusion テンポラリディレクトリを基準とする相対パスを指定します。
output	必須		ファイルに追加する文字列です。
addNewLine	オプション	yes	• yes: ファイルに書き込まれるテキストに改行文字を追加します。 • no
attributes	オプション		Windows に適用されます。ファイルに設定する属性をカンマ区切りリストで指定します。 この属性を省略した場合、ファイルの属性が保持されます。 値はそれぞれ明示的に指定する必要があります。たとえば、attributes="readOnly" を指定した場合、他の属性がすべて上書きされます。 • readOnly • hidden • normal

属性	必須 / オプション	デフォルト	説明
charset	オプション	JVM のデフォルトのファイル文字セット	ファイルコンテンツをエンコードする文字エンコードを指定します。一般的に使用される値を次に示します。 • utf-8 • iso-8859-1 • windows-1252 • us-ascii • shift_jis • iso-2022-jp • euc-jp • euc-kr • big5 • euc-cn • utf-16 文字エンコードの詳細については、www.w3.org/International/O-charset.html を参照してください。
fixnewline	オプション	No	• yes: 文字列変数に埋め込まれた行終了文字を、オペレーティングシステム固有の行終了文字に変更します。 • no: (デフォルト) 文字列変数に埋め込まれた行終了文字を変更しません。 この属性を使用する例については、cffile action = "write" を参照してください。
mode	オプション		UNIX および Linux だけに適用されます。許可を表す UNIX chmod コマンドの 8 進数値です。所有者、グループ、および他の利用者それぞれに割り当てられます。例 : • 644: 所有者に読み取り / 書き込み許可、グループおよび他の利用者に読み取り許可を割り当てます。 • 777: 利用者全員に読み取り / 書き込み / 実行許可を割り当てます。

例

```
<!--The first example creates the file \temp\foo on a windows system and sets attributes to normal. --->
<cffile action = "write" file = "\temp\foo" attributes = normal output = "some text">

<!--- The second example appends to the file. --->
<cffile action = "append" file = "\temp\foo" attributes = normal output = "Is this a test?">
```

cffile action = "copy"

説明

サーバー上のあるディレクトリから別のディレクトリにファイルをコピーします。

シンタックス

```
<cffile
    action = "copy"
    destination = "full pathname"
    source = "full pathname"
    attributes = "file attributes list"
    mode = "mode">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfdirectory

履歴

メインの cffile タグのページの「履歴」を参照してください。

属性

属性	必須 / オプション	デフォルト	説明
action	必須		このタグが実行するファイル操作のタイプです。
destination	必須		ファイルのコピー先となる Web サーバー上のディレクトリまたはファイルのパス名です。ディレクトリパスを明示しないでファイル名を指定した場合、ColdFusion は source で指定されたディレクトリの相対パスにファイルをコピーします。
source	必須		コピーするファイルのパス名です。 絶対パス (ドライブ文字とコロン、あるいはスラッシュまたは円記号から始まるパス) を指定しない場合は、GetTempDirectory 関数から返される ColdFusion テンポラリディレクトリを基準とする相対パスを指定します。
attributes	オプション		Windows に適用されます。ファイルに設定する属性をカンマ区切りリストで指定します。 この属性を省略した場合、ファイルの属性が保持されます。 値はそれぞれ明示的に指定する必要があります。たとえば、attributes="readOnly" を指定した場合、他の属性がすべて上書きされます。 • readOnly • hidden • normal
mode	オプション		UNIX および Linux だけに適用されます。許可を表す UNIX chmod コマンドの 8 進数値です。所有者、グループ、および他の利用者それぞれに割り当てられます。例 : • 644: 所有者に読み取り / 書き込み許可、グループおよび他の利用者に読み取り許可を割り当てます。 • 777: 利用者全員に読み取り / 書き込み / 実行許可を割り当てます。

例

この例では、"keymemo.doc" ファイルを c:¥files¥backup¥ ディレクトリにコピーします。

```
<cffile action = "copy" source = "c:\files\upload\keymemo.doc"
    destination = "c:\files\backup\">
```

cffile action = "delete"

説明
サーバー上のファイルを削除します。

シンタックス
```
<cffile
    action = "delete"
    file = "full pathname">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目
cfdirectory

属性

属性	必須 / オプション	デフォルト	説明
action	必須		このタグが実行するファイル操作のタイプです。
file	必須		削除するファイルのパス名です。 絶対パス (ドライブ文字とコロン、あるいはスラッシュまたは円記号から始まるパス) を指定しない場合は、GetTempDirectory 関数から返される ColdFusion テンポラリディレクトリを基準とする相対パスを指定します。

例
次の例では、指定したファイルを削除します。

```
<cffile action = "delete"
    file = "c:\files\upload\#Variables.DeleteFileName#">
```

cffile action = "move"

説明
サーバー上のある場所から別の場所にファイルを移動します。

シンタックス
```
<cffile
    action = "move"
    destination = "full pathname"
    source = "full pathname"
    attributes = "file attributes list"
    charset = "character set option"
    mode = "mode">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目
cfdirectory

履歴

メインの cffile タグのページの「履歴」を参照してください。

属性

属性	必須 / オプション	デフォルト	説明
action	必須		このタグが実行するファイル操作のタイプです。
destination	必須		移動先となるディレクトリまたはファイルのパス名です。絶対パスを指定しない場合は、ソースディレクトリを基準とした相対パスを指定します。
source	必須		移動するファイルのパス名です。 絶対パス (ドライブ文字とコロン、あるいはスラッシュまたは円記号から始まるパス) を指定しない場合は、GetTempDirectory 関数から返される ColdFusion テンポラリディレクトリを基準とする相対パスを指定します。
attributes	オプション		Windows に適用されます。ファイルに設定する属性をカンマ区切りリストで指定します。 この属性を省略した場合、ファイルの属性が保持されます。 値はそれぞれ明示的に指定する必要があります。たとえば、attributes="readOnly" を指定した場合、他の属性がすべて上書きされます。 • readOnly • hidden • normal
charset	オプション	JVM のデフォルトのファイル文字セット	ファイルコンテンツをエンコードする文字エンコードを指定します。一般的に使用される値を次に示します。 • utf-8 • iso-8859-1 • windows-1252 • us-ascii • shift_jis • iso-2022-jp • euc-jp • euc-kr • big5 • euc-cn • utf-16 文字エンコードの詳細については、www.w3.org/International/O-charset.html を参照してください。
mode	オプション		UNIX および Linux だけに適用されます。許可を表す UNIX chmod コマンドの 8 進数値です。所有者、グループ、および他の利用者それぞれに割り当てられます。例 : • 644: 所有者に読み取り / 書き込み許可、グループおよび他の利用者に読み取り許可を割り当てます。 • 777: 利用者全員に読み取り / 書き込み / 実行許可を割り当てます。

例

次の例では、"keymemo.doc" ファイルを、c:¥files¥upload¥ ディレクトリから Windows 上の c:¥files¥memo¥ ディレクトリに移動します。

```
<cffile
    action = "move"
    source = "c:\files\upload\keymemo.doc"
    destination = "c:\files\memo\">
```

この例では、保存先ディレクトリは "memo" です。

cffile action = "read"

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

説明

サーバー上のテキストファイルを読み取ります。ファイルは、ページ内で利用できるダイナミックなローカル変数内に読み込まれます。たとえば、次のようになります。

- テキストファイルを読み取り、そのファイルの内容をデータベースに挿入します。
- テキストファイルを読み取り、検索置換機能を使用してその内容を修正します。

注意：このアクションによって、ファイルがローカル Variables スコープ内の変数に読み込まれます。サーバーがダウンするおそれがあるため、ログなどの大きなファイルには使用しないでください。

シンタックス

```
<cffile
    action = "read"
    file = "full pathname"
    variable = "variable name"
    charset = "character set option">
```

関連項目

cfdirectory

履歴

メインの cffile タグのページの「履歴」を参照してください。

属性

属性	必須 / オプション	デフォルト	説明
action	必須		このタグが実行するファイル操作のタイプです。

属性	必須 / オプション	デフォルト	説明
file	必須		読み取るファイルのパス名です。 絶対パス (ドライブ文字とコロン、あるいはスラッシュまたは円記号から始まるパス) を指定しない場合は、GetTempDirectory 関数から返される ColdFusion テンポラリディレクトリを基準とする相対パスを指定します。
variable	必須		テキストファイルの内容が入る変数の名前です。
charset	オプション	文字エンコードは、そのファイルに BOM が存在する場合は BOM により、ない場合は JVM のデフォルトファイル文字セットにより決定します。	ファイルコンテンツをエンコードする文字エンコードを指定します。一般的に使用される値を次に示します。 • utf-8 • iso-8859-1 • windows-1252 • us-ascii • shift_jis • iso-2022-jp • euc-jp • euc-kr • big5 • euc-cn • utf-16 ファイルが BOM (Byte Order Mark) で始まる場合に、この属性を別の文字エンコードに設定すると、エラーになります。 文字エンコードの詳細については、www.w3.org/International/O-charset.html を参照してください。

使用方法

次の例では、ファイル message.txt の内容に対して Message という名前の変数が作成されます。

```
<cffile action = "read"
    file = "c:\web\message.txt"
    variable = "Message">
```

これにより、変数 Message をページ内で使用できます。たとえば、次のようにして最終的な Web ページに "message.txt" ファイルの内容を表示できます。

```
<cfoutput>#Message#</cfoutput>
```

ColdFusion では、テキストファイルの内容を操作するための関数がサポートされています。また、cffile action = "read" オペレーションで作成された変数を ArrayToList 関数や ListToArray 関数で使用することもできます。

注意：Latin-1 文字セットの Windows Cp1252 (windows--1252) エンコードでエンコードされたファイルを、デフォルトの文字エンコードが Cp1252 であるシステム上で cffile タグを使って読み取る場合、そのファイル内に 16 進数の 8x ～ 9x の範囲でエンコードされた文字が含まれているときには、charset="windows-1252" 属性を指定します (これはデフォルトのエンコードですが、指定が必要です)。そうしないと、16 進数の 8x ～ 9x の範囲にある一部の文字が正しくマップされず、正しく表示されません。

cffile action = "readBinary"

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

説明

ページ内で使用できるバイナリオブジェクトのパラメータに、サーバー上の実行可能ファイルやイメージファイルなどのバイナリファイルを読み込みます。このファイルを HTTP や SMTP などの Web プロトコルを介して送信するか、またはデータベースに保管するには、ToBase64 関数を使用してファイルをまず Base64 に変換してください。

注意：このアクションによって、ファイルがローカル Variables スコープ内の変数に読み込まれます。サーバーがダウンするおそれがあるため、ログなどの大きなファイルには使用しないでください。

シンタックス

```
<cffile
    action = "readBinary"
    file = "full pathname"
    variable = "variable name">
```

関連項目

cfdirectory

属性

属性	必須 / オプション	デフォルト	説明
action	必須		このタグが実行するファイル操作のタイプです。
file	必須		読み取るバイナリファイルのパス名です。 絶対パス (ドライブ文字とコロン、あるいはスラッシュまたは円記号から始まるパス) を指定しない場合は、GetTempDirectory 関数から返される ColdFusion テンポラリディレクトリを基準とする相対パスを指定します。
variable	必須		バイナリファイルの内容が入る変数の名前です。

使用方法

バイナリファイルを Base64 に変換して、それを別のサイトに転送します。

ColdFusion では、イメージファイルをバイナリとして読み込み、その結果を cfimage に渡すことができます。例 :

```
<!--- Convert a JPG image to a binary object. --->
<cffile action="readBinary" file="maxwell05.jpg" variable="binaryObject">
<!--- Create a cfimage from the binary object variable. --->
<cfset myImage=ImageNew(binaryObject)>
```

例

次の例では、バイナリファイル somewhere.jpg を読み込み、それを somewhereB.jpg として別のフォルダに書き込み、次にその新規ファイルを表示します。

```
 <cffile action = "readBinary" file = "C:\inetpub\wwwroot\cfdocs\getting_started\photos\somewhere.jpg"
variable = "aBinaryObj">

<!--- Output binary object to JPEG format for viewing. --->
<cffile action="write" file = "c:\files\updates\somewhereB.jpg"
    output = "#toBinary(aBinaryObj)#">

<!--- HTML to view image. --->
<img src="C:\files\updates\somewhereB.jpg">
```

cffile action = "rename"

説明

サーバー上のファイルの名前を変更するか、ファイルを移動します。

シンタックス

```
<cffile
    action = "rename"
    destination = "pathname"
    source = "full pathname"
    attributes = "file attributes list"
    mode = "mode">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfdirectory

履歴

メインの cffile タグのページの「履歴」を参照してください。

属性

属性	必須 / オプション	デフォルト	説明
action	必須		このタグが実行するファイル操作のタイプです。
destination	必須		変更先となるファイルまたはディレクトリです。絶対パスを指定しない場合は、ソースディレクトリを基準とした相対パスを指定します。

属性	必須 / オプション	デフォルト	説明
source	必須		名前を変更するファイルのパス名です。 絶対パス (ドライブ文字とコロン、あるいはスラッシュまたは円記号から始まるパス) を指定しない場合は、GetTempDirectory 関数から返される ColdFusion テンポラリディレクトリを基準とする相対パスを指定します。
attributes	オプション		Windows に適用されます。ファイルに設定する属性をカンマ区切りリストで指定します。 この属性を省略した場合、ファイルの属性が保持されます。 値はそれぞれ明示的に指定する必要があります。たとえば、attributes="readOnly" を指定した場合、他の属性がすべて上書きされます。 • readOnly • hidden • normal
mode	オプション		UNIX および Linux だけに適用されます。許可を表す UNIX chmod コマンドの 8 進数値です。所有者、グループ、および他の利用者に割り当てられます。たとえば、次のようになります。 • 644: 所有者に読み取り / 書き込み許可、グループおよび他の利用者に読み取り許可を割り当てます。 • 777: 利用者全員に読み取り / 書き込み / 実行許可を割り当てます。

使用方法

rename アクションでは、ファイルの名前を変更するか、ファイルを移動します。destination 属性には、新しいファイル名だけでなくパス名を指定する必要があります。変更先がディレクトリの場合は、ファイルの名前を変更せずに移動することになります。

例

Windows の例 :

```
<!--- Source Document is read-only but when renamed it becomes normal (not hidden or
    read-only). --->
<cffile action = "rename" source = "c:\files\memo\readonlymemo.doc"
    destination = "c:\files\memo\normalmemo.doc" attributes="normal">
```

UNIX の例 :

```
<cffile action = "rename" source = "#myWR#/memo/sample.txt"
    destination = "#myWR#/memo/other_sample.txt" mode="666">
```

cffile action = "upload"

説明

サーバー上のディレクトリにファイルをコピーします。

シンタックス

```
<cffile
    action = "upload"
    destination = "full pathname"
    fileField = "form field"
    accept = "MIME type|file type"
    attributes = "file attribute or list"
    mode = "permission"
    nameConflict = "behavior"
    result = "result name">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfdirectory

履歴

メインの cffile タグのページの「履歴」を参照してください。

属性

属性	必須 / オプション	デフォルト	説明
action	必須		このタグが実行するファイル操作のタイプです。
destination	必須		ファイルのアップロード先となるディレクトリのパス名です。絶対パス (ドライブ文字とコロン、あるいはスラッシュまたは円記号から始まるパス) を指定しない場合は、GetTempDirectory 関数から返される ColdFusion テンポラリディレクトリを基準とする相対パスを指定します。 指定した宛先が存在しない場合は、指定された宛先の名前のファイルが作成されます。たとえば、宛先 C:¥XYZ を指定した場合は、C: ドライブにファイル XYZ が作成されます。
fileField	必須		ファイルを選択するときに使用したフォームフィールドの名前です。 フィールド名を指定するときにシャープ記号 (#) を使用しないでください。
accept	オプション		受け入れる MIME タイプを制限します。カンマ区切りリスト . たとえば、次のコードでは、JPEG ファイルと Microsoft Word ファイルのアップロードが許可されます。 accept="image/jpg, application/msword" ブラウザは、ファイル拡張子に基づいてファイルタイプを判別します。
attributes	オプション		Windows に適用されます。ファイルに設定する属性をカンマ区切りリストで指定します。 この属性を省略した場合、ファイルの属性が保持されます。 値はそれぞれ明示的に指定する必要があります。たとえば、attributes="readOnly" を指定した場合、他の属性がすべて上書きされます。 • readOnly • hidden • normal (normal 以外に他の属性が指定されている場合、normal よりも他の属性が優先されます)

属性	必須 / オプション	デフォルト	説明
mode	オプション		UNIX および Linux だけに適用されます。許可を表します。chmod コマンドの 8 進数値を使用し、所有者、グループ、および他の利用者それぞれに割り当てられます。例 : • 644: 所有者に読み取り / 書き込み許可、グループおよび他の利用者に読み取り許可を割り当てます。 • 777: 利用者全員に読み取り / 書き込み / 実行許可を割り当てます。
nameConflict	オプション	Error	ファイル名がディレクトリ内のファイルと同じ場合に実行するアクションです。 • Error: ファイルは保存されません。ColdFusion によりページの処理が停止され、エラーが返されます。 • Skip: ファイルは保存されません。このオプションでは、ファイルプロパティに基づいて、動作をカスタマイズできます。 • Overwrite: ファイルを上書きします。 • MakeUnique: アップロード用に固有のファイル名を作成します。この名前は、file オブジェクト変数 serverFile に保管されます。
result	オプション		cffile が結果またはステータスのパラメータを返す変数の名前を指定することができます。この属性の値を指定しない場合、cffile は cffile という接頭辞を使用します。詳細については、「使用方法」を参照してください。

使用方法

ファイルのアップロードが終了すると、ファイルアップロードのパラメータを使用してステータス情報を取得できます。パラメータを参照するには、cffile 接頭辞を使用するか、result 属性で代替名を指定した場合はその指定した名前を使用します。たとえば、result 属性で名前を指定しなかった場合は、#cffile.fileExisted# として fileExisted パラメータにアクセスします。result 属性を myResult に設定した場合は、#myResult.fileExisted# として fileExisted パラメータにアクセスします。

ステータスパラメータは、他の ColdFusion パラメータを使用できる場所であればどこでも使用できます。

cfform タグまたは HTML フォームタグを使用して、アップロードするファイルと同時にフォームも送信する場合は、このタグの例で示すように enctype="multipart/form-data" を指定します。ColdFusion のデフォルトでは、application/x-www-form-urlencoded エンコーディングを使用してフォームが送信されるので、cffile タグでエラーが発生します。

result 属性では、複数のページから同時に呼び出される関数または CFC について、一方の呼び出しの結果が他方の呼び出しの結果を上書きしないようにすることができます。

注意：cffile 接頭辞を優先するために、file 接頭辞は非推奨になりました。新規アプリケーションでは file 接頭辞を使用しないでください。

ページのフォーム上で選択されたファイルまたは multipart/form-data HTTP メッセージ経由でページに送信されたファイルをアップロードする場合には、CGI.content_length 変数の値を調べれば、ファイルのおおよそのサイズがわかります。この変数には、ファイルの長さだけでなく、その他のリクエストコンテンツの長さも格納されています。

アップロードを終えた後は、次に示すファイルアップロードステータスのパラメータを使用できます。

パラメータ	説明
attemptedServerFile	ファイルを保存するために使用された初期名です。
clientDirectory	クライアントのシステムからアップロードされたファイルのディレクトリ位置です。
clientFile	クライアントのシステムからアップロードされたファイルの名前です。
clientFileExt	クライアントシステム上のアップロードされたファイルの拡張子です (ピリオドは含みません)。

パラメータ	説明
clientFileName	クライアントシステム上のアップロードされたファイルの名前です (拡張子は含みません)。
contentSubType	保存ファイルの MIME コンテンツサブタイプです。
contentType	保存ファイルの MIME コンテンツタイプです。
dateLastAccessed	アップロードされたファイルが最後にアクセスされた日付と時刻です。
fileExisted	同じパスを持つファイルが存在しているかどうかを yes または no で示します。
fileSize	アップロードされたファイルのサイズです。
fileWasAppended	アップロードされたファイルが他のファイルに追加されたかどうかを yes または no で示します。
fileWasOverwritten	ファイルが上書きされたかどうかを yes または no で示します。
fileWasRenamed	名前の重複を回避するために、アップロードされたファイルの名前が変更されたかどうかを yes または no で示します。
fileWasSaved	ファイルが保存されたかどうかを yes または no で示します。
oldFileSize	ファイルのアップロード処理によって上書きされたファイルのサイズです。
serverDirectory	サーバー上に実際に保存されたファイルのディレクトリです。
serverFile	サーバー上に保存されたファイルの名前です。
serverFileExt	サーバー上にアップロードされたファイルの拡張子です (ピリオドは含みません)。
serverFileName	サーバー上にアップロードされたファイルの名前です (拡張子は含みません)。
timeCreated	アップロードされたファイルが作成された時刻です。
timeLastModified	アップロードされたファイルが最後に修正された日付と時刻です。

注意：ファイルステータスのパラメータは読み取り専用です。これらの変数には、最新の cffile 操作の結果が設定されます。2 つの cffile タグを実行すると、result 属性で異なる結果変数を指定していない限り、最初の結果は 2 番目の結果によって上書きされます。

例

次の例では、Windows 上でのファイルアップロード時に名前が競合している場合に、固有のファイル名を作成します。

```
 <!--- Windows Example --->
<!--- Check to see if the Form variable exists. --->
<cfif isDefined("Form.FileContents") >
    <!--- If TRUE, upload the file. --->
    <cffile action = "upload"
        fileField = "FileContents"
        destination = "c:\files\upload\"
        accept = "text/html"
        nameConflict = "MakeUnique">
<cfelse>
    <!--- If FALSE, show the Form. --->
    <form method="post" action=<cfoutput>#cgi.script_name#</cfoutput>
        name="uploadForm" enctype="multipart/form-data">
        <input name="FileContents" type="file">
        <br>
        <input name="submit" type="submit" value="Upload File">
    </form>
</cfif>
```

cffile action = "uploadAll"

説明
HTTP リクエストでページに送信されたすべてのファイルを、サーバー上のディレクトリにコピーします。

シンタックス
```
<cffile
    action = "uploadAll"
    destination = "full pathname"
    accept = "list of MIME types"
    attributes = "file attribute or list"
    mode = "permission"
    nameConflict = "behavior"
    result = "result name">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目
cffile action = "upload"、cfdirectory

履歴
メインの cffile タグのページの「履歴」を参照してください。

属性

属性	必須 / オプション	デフォルト	説明
action	必須		このタグが実行するファイル操作のタイプです。
destination	必須		ファイルのアップロード先となるディレクトリのパス名です。絶対パス (ドライブ文字とコロン、あるいはスラッシュまたは円記号から始まるパス) を指定しない場合は、GetTempDirectory 関数から返される ColdFusion テンポラリディレクトリを基準とする相対パスを指定します。
accept	オプション		受け入れる MIME タイプを制限します。カンマ区切りリスト . たとえば、次のコードでは、JPEG ファイルと Microsoft Word ファイルのアップロードが許可されます。 accept="image/jpg, application/msword" ブラウザは、ファイル拡張子に基づいてファイルタイプを判別します。
attributes	オプション		Windows に適用されます。ファイルに設定する属性をカンマ区切りリストで指定します。 この属性を省略した場合、ファイルの属性が保持されます。 値はそれぞれ明示的に指定する必要があります。たとえば、attributes="readOnly" を指定した場合、他の属性がすべて上書きされます。 • readOnly • hidden • normal (normal 以外に他の属性が指定されている場合、normal よりも他の属性が優先されます)

属性	必須 / オプション	デフォルト	説明
mode	オプション		UNIX および Linux だけに適用されます。許可を表します。chmod コマンドの 8 進数値を使用し、所有者、グループ、および他の利用者それぞれに割り当てられます。例 : • 644: 所有者に読み取り / 書き込み許可、グループおよび他の利用者に読み取り許可を割り当てます。 • 777: 利用者全員に読み取り / 書き込み / 実行許可を割り当てます。
nameConflict	オプション	Error	ファイル名がディレクトリ内のファイルと同じ場合に実行するアクションです。 • Error: ファイルは保存されません。ColdFusion によりページの処理が停止され、エラーが返されます。 • Skip: ファイルは保存されません。このオプションでは、ファイルプロパティに基づいて、動作をカスタマイズできます。 • Overwrite: ファイルを上書きします。 • MakeUnique: アップロード用に固有のファイル名を作成します。この名前は、ファイルの結果の構造体の serverFile フィールドに保管されます。
result	オプション		cffile が結果またはステータスのパラメータを返す変数の名前を指定することができます。この属性の値を指定しない場合、cffile は cffile という接頭辞を使用します。詳細については、「使用方法」を参照してください。

使用方法

一度に 1 つのファイルのみをアップロードする cffile action="upload" とは異なり、cf fileaction="uploadall" を使用すると複数のファイルがアップロードされるので、複数の cffile action="upload" ステートメントを記述する必要がなくなります。

cffileupload コントロールの action 属性で指定されたページで、このタグを使用します。このタグのアップロードは、ユーザーが保存ボタンをクリックしたときに cffileupload コントロールによって送信されるファイルを保存します。

ファイルのアップロードが終了すると、このタグによって、結果パラメータで指定した構造体の配列が作成されます。配列の各構造体には、1 つのファイルのアップロード結果情報が格納されます。結果の構造体の内容については、cffile action = "upload" を参照してください。

注意：サーバーの [リクエストのスロットルしきい値] または Administrator の [サーバーの設定] の [設定] ページを指定することによって、アップロードの最大ファイルサイズを制御できます。

例

次の例では、cffileupload タグによってアップロードされたファイルを一時ディレクトリにコピーします。

```
<cfif isdefined("form.submit")>
    <cffile action="uploadall" destination="#expandpath('./upload')#">
</cfif>
<cfform action="#cgi.script_name#" enctype="multipart/form-data">
    <cfinput type="file" name="attachment1"><br>
    <cfinput type="file" name="attachment2"><br>
    <cfinput type="file" name="attachment3"><br>
    <cfinput type="submit" name=" submit" value="submit">
</cfform>
```

cffile action = "write"

説明
ダイナミックコンテンツをベースにして、サーバーにテキストファイルを書き込みます。コンテンツからスタティックな HTML ファイルを作成したり、テキストファイル内にアクションを記録したりできます。

シンタックス
```
<cffile
    action = "write"
    file = "full pathname"
    output = "content"
    addNewLine = "yes|no"
    attributes = "file attributes list"
    charset = "character set option"
    fixnewline = "yes|no"
    mode = "permission">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目
cfdirectory

履歴
メインの cffile タグのページの「履歴」を参照してください。

属性

属性	必須 / オプション	デフォルト	説明
action	必須		このタグが実行するファイル操作のタイプです。
file	必須		書き込み先となるファイルのパス名です。 絶対パス (ドライブ文字とコロン、あるいはスラッシュまたは円記号から始まるパス) を指定しない場合は、GetTempDirectory 関数から返される ColdFusion テンポラリディレクトリを基準とする相対パスを指定します。
output	必須		作成するファイルの内容です。
addNewLine	オプション	yes	• yes: ファイルに書き込まれるテキストに改行文字を追加します。 • no
attributes	オプション		Windows に適用されます。ファイルに設定する属性をカンマ区切りリストで指定します。 この属性を省略した場合、ファイルの属性が保持されます。 値はそれぞれ明示的に指定する必要があります。たとえば、attributes="readOnly" を指定した場合、他の属性がすべて上書きされます。 • readOnly • hidden • normal

属性	必須 / オプション	デフォルト	説明
charset	オプション	JVM のデフォルトのファイル文字セット	ファイルコンテンツをエンコードする文字エンコードを指定します。一般的に使用される値を次に示します。 • utf-8 • iso-8859-1 • windows-1252 • us-ascii • shift_jis • iso-2022-jp • euc-jp • euc-kr • big5 • euc-cn • utf-16 文字エンコードの詳細については、www.w3.org/International/O-charset.html を参照してください。
fixnewline	オプション	no	• yes: 文字列変数に埋め込まれた行終了文字を、オペレーティングシステム固有の行終了文字に変更します。 • no: 文字列変数に埋め込まれた行終了文字を変更しません。
mode	オプション		UNIX および Linux だけに適用されます。許可を表す UNIX chmod コマンドの 8 進数値です。所有者、グループ、および他の利用者それぞれに割り当てられます。例 : • 644: 所有者に読み取り / 書き込み許可、グループおよび他の利用者に読み取り許可を割り当てます。 • 777: 利用者全員に読み取り / 書き込み / 実行許可を割り当てます。

例

この例では、ユーザーが HTML 挿入フォームに入力した情報を使用してファイルを作成します。

```
<cffile action = "write"
    file = "c:\files\updates\#Form.UpdateTitle#.txt"
    output = "Created By: #Form.FullName#
    Date: #Form.Date#
    #Form.Content#">
```

ユーザーが次のフォームを送信した場合、

```
UpdateTitle = "FieldWork"
FullName = "World B. Frueh"
Date = "10/30/01"
Content = "We had a wonderful time in Cambridgeport."
```

c:¥files¥updates¥ ディレクトリに FieldWork.txt という名前のファイルが作成され、ファイルに次のテキストが加えられます。

```
 Created By: World B. Frueh
Date: 10/30/01
    We had a wonderful time in Cambridgeport.
```

この例は、UNIX での mode 属性の使用方法を示しています。ファイル /tmp/foo を作成し、rw-r--r-- (owner = read/write, group = read, other = read) というアクセス許可を指定します。

```
<cffile action = "write"
    file = "/tmp/foo"
    mode = 644>
```

この例では、ファイルへの追加を行い、利用者全員に読み取り / 書き込み (rw) 許可を設定します。

```
<cffile action = "append"
    destination = "/home/tomj/testing.txt"
    mode = 666
    output = "Is this a test?">
```

この例では、ファイルをアップロードして、アクセス許可を owner/group/other = read/write/execute に設定します。

```
cffile action = "upload"
    fileField = "fieldname"
    destination = "/tmp/program.exe"
    mode = 777>
```

この例では、fixnewline 属性を使用して、xmlData から派生する xmlString に埋め込まれた行終了文字を、オペレーティングシステム固有の行終了文字に変更します。

```
<cfxml variable="xmlData">
    <docroot>
        <payload type="string">This is some plain text</payload>
    </docroot>
</cfxml>
<cfset xmlString = toString(xmlData)>

<cfset key = createUUID()>
<cfset encString=encrypt(xmlString, key)>
<cffile action="write" addnewline="yes"
    file="C:\ColdFusion9\wwwroot\test\store.dat"
    output="#encString#" fixnewline="yes">
<cffile action="read" file="C:\ColdFusion9\wwwroot\test\store.dat"
    variable="retrievedString">
<cfset decString=decrypt(retrievedString, key)>
<cfdump var="#decString#">
<cfset newXML = xmlParse(decString)>
<cfdump var="#newXML#">
```

ColdFusion では、cffile を使用してイメージを書き込むことができます。例 :

```
<!--- Create a new cfimage. --->
<cfset myImage=ImageNew("",200,200)>
<!--- Draw a square on the image. --->
<cfset ImageDrawRect(myImage,10,10,100,100)>
<!--- Use cffile to write the cfimage to a JPG. --->
<cffile action="write" output="#myImage#" file="c:\cfpix\square.jpg">
```

cffileupload

説明

ユーザーのシステムから複数のファイルをアップロードするためのダイアログボックスを表示します。

機能強化されたダイアログボックスには、次の機能があります。

- アップロードするファイルの最大数とファイルの最大サイズを指定できます。
- プログレスバーに、ファイルのアップロードタスクの全体的な進行状況が視覚的に示されます。また、もう 1 つのプログレスバーに、個別のファイルのアップロードの進行状況が示されます。
- ファイルのアップロードごと、およびアップロードタスク全体に関して、成功または失敗を示すメッセージが表示されます。
- アップロードタスクの任意の時点で、アップロードをキャンセルできます。

カテゴリ

ファイル管理タグ、フォームタグ

シンタックス

```
<cffileupload>
    addbuttonlabel= "label"
    align = align="center|left|right"
    bgcolor = "color"
    clearbuttonlabel = "label"
    deletebuttonlabel = "label"
    extensionfilter = "none|jpg,jpeg,png"
    height= "number of pixels"
    hideUploadButton = "true|false"
    maxfileselect = "number of files"
    maxuploadsize = "file size in mega bytes"
    name = "File uploader name"
    oncomplete = "JavaScript function name"
    onerror = "JavaScript function name"
    onUploadComplete = "JavaScript function name"
    progressbar = "true|false"
    stoponerror = "true|false"
    style = "style specification"
    title = "Title panel name"
    uploadbuttonlabel = "label"
    url = "URL"
    width = "number of pixels"
    wmode = "window|opaque|transparent"
</cffileupload>
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cffile action = "uploadAll"

履歴

ColdFusion 9: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
addbuttonlabel	オプション	ファイルを追加	[追加] ボタンのラベルです。
align	オプション	left	デフォルトの配置を指定します。 使用できる値は次のとおりです。 • center • left • right
bgcolor	オプション		ファイルアップロードコントロールの背景色です。先頭の「#」を除いた 16 進数値または認識可能なカラー名 (red など) です。
clearbuttonlabel	オプション	すべてをクリア	[クリア] ボタンのラベルです。
deletebuttonlabel	オプション	Delete	[削除] ボタンのラベルです。
extensionfilter	オプション	なし	この属性は、アップロードを許可するファイルタイプを指定する場合に使用します。たとえば、イメージファイルのみをアップロード可能にするには、.jpg、.jpeg、.png などのファイル拡張子を指定できます。 none に設定すると、拡張子フィルタが適用されずにファイルがアップロードされます。
height	オプション	300	ファイルアップロードコントロールの高さです (単位 : ピクセル)。
hideUploadButton	オプション	false	ファイルアップロードダイアログボックスに [アップロード] ボタンを表示するかどうかを指定するブール値です。 • true • false
maxfileselect	オプション		アップロード可能なファイルの最大数です。
maxuploadsize	オプション	10 MB	1 回のオペレーションでアップロード可能なファイルの最大サイズ (MB) です。 maxuploadsize 属性の値が、ColdFusion Administrator で指定されているスロットルまたは送信データサイズの設定を超えると、ColdFusion でエラーが発生します。
name	オプション		ファイルアップロードコンポーネントの名前です。
onComplete	オプション		ファイルのアップロードが終了したときに実行する JavaScript 関数です。 デフォルトでは、JavaScript オブジェクトが、パラメータとして次のプロパティとともにこの関数に渡されます。 • STATUS - HTTP ステータスコードに基づいた数値です。 • MESSAGE - 成功または失敗を示します。 • FILENAME - アップロード対象として選択されたファイルの名前です。 また、status および message のパラメータを指定した構造体を作成することによって JavaScript オブジェクトを渡し、その JavaScript オブジェクトに対して serializeJSON() を呼び出すこともできます。

属性	必須 / オプション	デフォルト	説明
onError	オプション		ファイルのアップロードが失敗したときに実行する JavaScript 関数です。この場合のエラーとは、ネットワークエラーまたはサーバーサイドエラーとなります。 デフォルトでは、JavaScript オブジェクトが、パラメータとして次のプロパティとともにこの関数に渡されます。 • STATUS - HTTP ステータスコードに基づいた数値です。 • MESSAGE - 成功または失敗を示します。 • FILENAME - アップロード対象として選択されたファイルの名前です。 また、status および message のパラメータを指定した構造体を作成することによって JavaScript オブジェクトを渡し、その JavaScript オブジェクトに対して serializeJSON() を呼び出すこともできます。
onUploadComplete	オプション		すべてのファイルがアップロードされた後に実行する JavaScript 関数です。
progressbar	オプション	true	ファイルのアップロード中にプログレスバーを表示するかどうかを指定します。 • true • false
stoponerror	オプション	true	このオペレーションに関する例外を無視するかどうかを指定します。 • true - アップロードを中止して、該当するエラーを表示します。 • false - アップロードを続行して、該当するエラーを表示します。
style	オプション		レイアウトのスタイルを定義する CSS スタイル指定です。
title	オプション		アップロードダイアログボックスのタイトルです。
uploadbuttonlabel	オプション	アップロード	[アップロード] ボタンのラベルです。
url	オプション		ファイルのアップロード先となるサーバーの URL です。 この属性はオプションで、デフォルトでは cgi.script_name が設定されます。
width	オプション	420	ファイルアップロードコントロールの幅です (単位 : ピクセル)。
wmode	オプション	window	ブラウザでの絶対位置設定およびレイヤー作成の機能を指定します。 • window: Web ページ上の独自の長方形のウィンドウ内でメディアプレーヤーを再生します。 • opaque: Web ページ上でメディアプレーヤーの背後にあるものをすべて非表示にします。 • transparent: メディアプレーヤーの透明の部分に Web ページの背景が透けて見えるようにします。

使用方法

このタグは、ユーザーがサーバーに複数のファイルをアップロードできるようにする、SWF ファイル形式のファイルアップロードコントロールを作成する場合に使用します。

サーバーにファイルをアップロードするには、サーバーサイドテンプレートを定義します。定義したテンプレートは、アップロードリクエストを読み込み、選択されたファイルをサーバーにアップロードします。

ColdFusion 9.0.1 で行われた機能強化

ColdFusion 9.0.1 では、Application.cfc か Application.cfm のいずれかでセッション管理がオンになっている場合、ファイルアップロードコントロールで暗黙的にセッション情報が対象ページに渡されます。

たとえば、ファイルアップロードコントロールが URL 属性なしで定義されているとします。この場合、ユーザーがアップロードボタンを使用してデータをアップロードしようとすると、コントロールによって同じページに戻されます。ユーザーは、form.fieldnames をチェックしてアップロードを実行できます。次に例を示します。

Upload.cfm

```
<cfif isdefined("form.FIELDNAMES")>
      <cffile action = "upload" destination = "#ExpandPath('.')#" nameconflict="makeunique">
</cfif>
<cffileupload name="myuploader">
```

この場合、url には、デフォルトで CGI.script_name が設定されます。

ファイルアップロードコントロールと URL の間のセッションを維持するには、ユーザーがセッション管理をオンにする必要があります。これを実行するには、Application.cfc で this.sessionmanagement=true を設定します。[J2EE セッション変数の使用] ([ColdFusion Administrator]-[サーバーの設定]-[メモリ変数]) が選択されていない場合は、この設定によって CFID と CFtoken が URL の一部として渡されるようになります。選択されている場合は、JsessionID が URL の一部として渡されます。

サポートされているスタイル

サポートされるスタイルは次のとおりです。

スタイル	説明
headercolors	形式は色です。DateChooser コントロール上部のバンドの色を指定します。2 つの値をカンマで区切って指定します。塗りつぶしのバンドの場合、両方の値に同じ色を使用します。デフォルト値は ##E6EEEE、##FFFFFF です。
textcolor	テキストの色です。16 進数値またはカラー名で指定します。 16 進数の値を入力するには、"##xxxxxx" という形式を使用します。ここで、x は 0 ～ 9 または A ～ F です。シャープ記号 (#) は 2 つ使用するか、または使用しないでください。
titletextalign	タイトルテキストを揃えます。認識される値は、left、right、および center です。デフォルト値は right です。
titletextcolor	タイトルテキストの色を指定します。
bgcolor	ファイルアップロードコントロールの背景色です。先頭の「#」を除いた 16 進数値または認識可能なカラー名 (red など) です。
rollovercolor	マウスカーソルが上に置かれたときに値を表示します。
selectcolor	選択されている項目の背景色です。16 進数値またはカラー名で指定します。 16 進数の値を入力するには、"##xxxxxx" という形式を使用します。ここで、x は 0 ～ 9 または A ～ F です。シャープ記号 (#) は 2 つ使用するか、または使用しないでください。サポートされる色の名前については、cfchart を参照してください。

例

```
 <h3>Instructions</h3>
<p>Create a folder Upload in your C: drive
<br>Try uploading files using the file upload component and check if the files have been appropriately saved
in the Upload folder.</p>
<script>
    var foo = function(result)
    {
        alert(ColdFusion.JSON.encode(result));
    }
</script>
<cffileupload
    url="uploadFiles.cfm"
    progressbar="true"
    name="myupload"
    addButtonLabel = "Add File"
    clearButtonlabel = "Clear it"
    hideUploadButton = "true"
    width=600
    height=400
    title = "File Upload"
    maxuploadsize="30"
    extensionfilter="*.jpg, *.png, *.flv, *.txt"
    BGCOLOR="##FFFFFF"
    MAXFILESELECT=10
    UPLOADBUTTONLABEL="Upload now"/>
```

uploadfiles.cfm は次のように記述します。

```
<cffile action="upload" destination="#expandpath('./upload')#" nameconflict="makeunique">
<cfoutput>#serializeJSON({STATUS=200,MESSAGE='Passed'})#</cfoutput>
```

この例では、サーバーサイドテンプレート uploadfiles.cfm にユーザー指定のファイルを送信しています。定義したテンプレートファイルでは、cffile タグで定義された upload または uploadall アクションを使用できます。

注意：upload アクションの filefield 属性はオプションです。

ファイルの保存場所を定義するには、cffile タグで destination 属性を使用します。uploadfiles.cfm のコードについては、cffile action = "uploadAll" を参照してください。

cffinally

説明

cftry タグの内部で使用します。cffinally ブロック内のコードは、メインの cftry コードおよび (例外が発生した場合の) cfcatch コードの後で処理されます。cffinally ブロックコードは、例外があるかどうかにかかわらず常に実行されます。

カテゴリ

例外処理タグ

シンタックス

```
 <cftry>
    try code<cfcatch>
        catch code<cfcatch>
    ...
    <cffinally>
        final code</cffinally>
</cftry>
```

関連項目
cftry、cfcatch、cferror、cfrethrow、cfthrow、onError、『ColdFusion アプリケーションの開発』の Handling Errors

履歴
ColdFusion 9: タグが追加されました。

使用方法
cftry ブロック内の cffinally タグはオプションです。このブロックには cffinally タグを 1 つだけ指定できます。cffinally タグは、すべての cftry ブロックの最後に、cfcatch ブロックに続けて配置します。このタグには終了タグが必要です。cftry/cfcatch/cffinally ブロックはネストできます。

cffinally タグは、例外が発生するかどうかにかかわらず実行する必要があるコードに使用します。たとえば、リソースを解放するために使用します。

例
```
<h3>cffinally Example</h3>
<!--- Open a cftry block. --->
<cftry>
    ....
    <cfcatch type = "Database">
    ....
    </cfcatch>
    <cffinally>
    ....
    <!--- Do some cleanup here before leaving cftry block --->
    ....
    </cffinally
</cftry>
```

cfflush

説明
現在使用可能なデータをクライアントにフラッシュします。

カテゴリ
データ出力タグ、ページ処理タグ

シンタックス
```
<cfflush
    interval = "integer number of bytes">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目
cfcache、cfheader、cfinclude、cfsetting、cfsilent

属性

属性	必須 / オプション	デフォルト	説明
interval	オプション		整数。使用可能なバイト数がこの数値に達するたびに、出力をフラッシュします。HTML ヘッダと、このタグの実行時に既に使用可能になっているデータは、このカウントから除外されます。

使用方法

ページ内でこのタグが最初に検出されると、HTML ヘッダおよびその他の使用可能な HTML が返されます。同じページ内の、以降の cfflush タグについては、最後のフラッシュの後で生成された出力のみが送信されます。

少量の情報しかフラッシュしない場合、ブラウザによってはレスポンスしないものもあります。データをフラッシュするときは十分な情報があることを確認してください。また、interval 属性は数百バイトに設定してください。数千バイトでは大きすぎます。

interval 属性は、大きなクエリーの cfloop や cfoutput などにおいて、大量の出力をクライアントに送信する場合にのみ使用します。このフォームをグローバルに使用すると ("Application.cfm" ファイル内など)、HTML ヘッダを修正する CFML タグの実行時に予期せぬエラーが発生する可能性があります。

cfflush タグは、このタグが実行されるときにブラウザにデータを送信するので、次の制限があります。

- ページ上で cfflush タグの後に cfcontent、cfcookie、cfform、cfheader、cfhtmlhead、cflocation、SetLocale といったタグや関数を使用すると、エラーが起きるか、予期せぬ結果になります。同様に、cfdiv、cflayout、cflayoutarea、cfpod、cfsprydataset、cftooltip、cfwindow などの AJAX 機能を使用するタグ、あるいは HTML 形式の cfgrid、cftree、cftextarea、cfinput (autosuggest 属性または datefield 属性を使用) タグは使用しないでください。これらのタグと関数はすべて、通常は HTML ヘッダを修正するものですが、cfflush タグによってヘッダが送信されてしまうので、cfflush タグの後に記述すると正常に処理を実行できません。
- cfflush タグを含むページ上の任意の場所で cfset タグを使用して Cookie を設定しても、ブラウザ内に Cookie は設定されません。
- cfsavecontent タグ、cfquery タグ、カスタムタグなどの本文の中で cfflush タグを使用すると、エラーになります。
- クライアント変数を Cookie として保存する場合、cfflush タグの後で設定したクライアント変数はブラウザ内に保存されません。

注意：通常、cferror タグでは現在の出力バッファーが破棄され、エラーページの内容に置き換えられます。cfflush タグでは現在のバッファーが破棄されます。結果として、cfflush の後で使用する cferror タグから生成された Error.GeneratedContent 変数には、フラッシュされていない出力バッファーの内容が保存されます。この内容はクライアントには送信されません。エラーページには、送信されたバイトの後の内容がクライアントに対して表示されます。

例

次の例では、cfloop タグと、乱数を生成する rand 関数を使用してデータ表示を遅らせます。ここではデータ生成に時間がかかるページをシミュレートします。

```
 <h1>Your Magic numbers</h1>
<p>It will take us a little while to calculate your ten magic numbers. It takes a lot of work to find numbers
that truly fit your personality. So relax for a minute or so while we do the hard work for you.</p>
<H2>We are sure you will agree it was worth the short wait!</H2>
<cfflush>

<cfflush interval=10>
<!--- Delay Loop to make it seem harder. --->
<cfloop index="randomindex" from="1" to="200000" step="1">
    <cfset random=rand()>
</cfloop>

<!--- Now slowly output 10 random numbers. --->
<cfloop index="Myindex" from="1" to="10" step="1">
    <cfloop index="randomindex" from="1" to="100000" step="1">
        <cfset random=rand()>
    </cfloop>
    <cfoutput>
        Magic number #Myindex# is:  #RandRange
        (100000, 999999)#<br><br>
    </cfoutput>
</cfloop>
```

cfform

説明

CFML カスタムコントロールタグを使用してフォームを作成します。CFML カスタムコントロールタグは、一般的な HTML フォーム入力要素より優れた機能を備えています。結果フォームを HTML または Adobe Flash コンテンツとしてクライアントページに含めることができます。フォームは XML および XSLT を使用して生成できます。

カテゴリ

フォームタグ

シンタックス

```
 <cfform
    accessible = "yes|no"
    action = "form action"
    archive = "URL"
    codeBase = "URL"
    format = "HTML|Flash|XML"
    height = "pixels|percent"
    id = "HTML id" method = "POST|GET"
    name = "name"
    onError = "JavaScript function name or ActionScript code"
    onLoad = "load event script" onReset = "reset event script" onSubmit = "JavaScript"
    onSuccess = "JavaScript function name"
    preloader = "yes|no"
    preserveData = "yes|no"
    scriptSrc = "path"
    skin = "Flash skin|XSL skin"
    style = "style specification"
    timeout = "seconds"
    width = "pixels|percent"
    wMode = "window|transparent|opaque">

    ...

</cfform>
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfajaximport、cfapplet、cfcalendar、cfformgroup、cfformitem、cfgrid、cfinput、cfselect、cfslider、cftextarea、cftree、『ColdFusion アプリケーションの開発』の Requesting and Presenting Information

履歴

ColdFusion 8:

- PDF フォームにインタラクティブフィールドを追加できるようになりました。
- onSuccess 属性が追加され、AJAX コントロールで onError 属性がサポートされるようになりました。

ColdFusion MX 7:

- ColdFusion Administrator で scriptSrc 属性のデフォルト値を設定できるようになりました。
- passthrough 属性は非推奨になりました。このタグがすべての HTML form タグ属性を直接サポートするようになりました。
- method 属性が追加され、GET メソッドを使用できるようになりました。
- format、height、width、preloader、timeout、wMode、accessible、skin の各属性を含む Flash および XML 出力のサポートが追加されました。
- cfformgroup、cfformitem、および cftextarea の子タグが追加されました。
- onReset 属性が追加されました。

ColdFusion MX:

- enableCAB 属性は非推奨になりました。今後のリリースではこのタグは機能せずエラーを引き起こす可能性があります。
- name 属性と action 属性は必須ではなくなりました。
- 整数値を要求するための整数の検証が変更されました。以前のリリースでは、浮動小数点値が整数に変換されました。

属性

次の表は、ColdFusion が直接使用する属性のリストです。HTML 形式のフォームでは、このリストに記載されていない標準の HTML form タグ属性もこのタグでサポートされ、ブラウザに直接渡されます。ColdFusion には、XML でサポートされる HTML 属性もすべて含まれています。

属性	適用対象	必須 / オプション	デフォルト	説明
accessible	Flash	オプション	no	スクリーンリーダーを Flash フォームに対応可能にするかどうかを指定します。スクリーンリーダーのサポートにより、クライアントに送信する SWF ファイルに約 80 KB が加わります。
action	Flash HTML XML	オプション	「説明」を参照	処理のためにフォームを送信するときに実行する ColdFusion ページの名前です。 この属性を省略した場合に、method が get の場合は、フォームは CGI.SCRIPT_NAME 変数で特定されるページ（フォームを表示するようにリクエストされたページ）に送信されます。method が post の場合は、フォームは CGI.QUERY_STRING 変数で特定されるページに送信されます。
archive	HTML および XML のアプレット	オプション	/CFIDE/classes/cfapplets.jar です。	cfgrid、cfslider、および cftree のアプレットコントロール用のダウンロード可能な Java クラスの URL です。

属性	適用対象	必須 / オプション	デフォルト	説明
codeBase	HTML および XML のアプレット	オプション	/CFIDE/classes/cf-j2re-win.cab です。	Internet Explorer 用のダウンロード可能な JRE プラグインの URL です。cfgrid、cfslider、および cftree の Java アプレットコントロールに使用します。
format	Flash HTML XML	オプション	HTML	• HTML: HTML フォームを生成してクライアントに送信します。cfgrid および cftree の子コントロールは Flash 形式またはアプレット形式にできます。 • Flash: Flash フォームを生成し、クライアントに送信します。すべてのコントロールは Flash 形式です。 • XML: XForms 準拠の XML を生成し、name 属性で指定された変数に結果を保存します。デフォルトでは、ColdFusion は XSL スキンを適用して結果を表示します。詳細については、skin 属性を参照してください。
height	Flash XML	オプション	100%	フォームの高さです。ピクセル数を数値で指定します。Flash では、"height=60%" などのパーセント値を使用して、表示可能な高さの割合を指定できます。表示される高さは、指定したサイズよりも小さい場合があります。 **メモ :** 表の内部で Flash フォームを使用する場合、width 属性と height 属性は必須です。
id			name 属性の値	フォームの HTML id です。
method	Flash HTML XML	オプション	post	ブラウザがフォームデータをサーバーに送信するために使用するメソッドです。 • post: HTTP Post メソッドを使用してデータを送信します。このメソッドでは、データを個別のメッセージとしてサーバーに送信します。 • get: HTTP Get メソッドを使用してデータを送信します。このメソッドでは、フォームフィールドの内容を URL クエリー文字列に挿入します。
name	Flash HTML XML	オプション	CFForm_**n**	フォームの名前です。 HTML 形式の場合、この属性を省略して id 属性を指定すると、ColdFusion ではブラウザに送信される HTML に name 属性が含まれません。この動作により、cfform タグを使用して XHTML 準拠のフォームを作成することができます。name 属性および id 属性を省略すると、ColdFusion では CFForm_**n** というフォーム名が生成されます。ここで、**n** はページ上のフォームに連続的に割り当てられた番号です。
onError	Flash HTML	オプション		Flash 形式フォームの場合 : onSubmit 検証または onBlur 検証の場合にのみ適用されます。onServer 検証の場合は効果がありません。 検証エラーがあるフォームをユーザーが送信する場合に実行する、1 つまたは複数の ActionScript 式です。 HTML 形式フォームの場合 : cfdiv、cflayout、cfpod、または cfwindow コントロール内のフォームにのみ適用されます。フォームの非同期送信に失敗した場合に実行する JavaScript 関数の名前です。詳細については、「使用方法」を参照してください。
onLoad	HTML XML	オプション		フォームのロード時に実行する JavaScript です。

属性	適用対象	必須 / オプション	デフォルト	説明
onReset	HTML XML	オプション		ユーザーが [リセット] ボタンをクリックしたときに実行する JavaScript です。
onSubmit	Flash HTML XML	オプション		フォームの送信前にデータの前処理を行うために実行する JavaScript または ActionScript 関数です。子タグが onSubmit フィールド検証を指定した場合、ColdFusion はこの JavaScript の実行前に検証を行います。
onSuccess	HTML	オプション		cfdiv、cflayout、cfpod、または cfwindow コントロール内のフォームにのみ適用されます。フォームの非同期送信に成功したときに実行する JavaScript 関数の名前です。詳細については、「使用方法」を参照してください。
preloader	Flash	オプション	yes	Flash フォームのロード時に進行状況表示バーを表示するかどうかを指定します。
preserveData	HTML XML	オプション	no	cfform の action 属性によりフォームを含むページに戻るときに、コントロールの値を送信された値で上書きするかどうかを決定します。 • no: コントロールタグの属性に指定された値を使用します。 • yes: 対応する送信値を使用します。 次のコントロールに適用されます。 • cfinput、cfslider、cftextinput: value 属性の値を上書きします。 • クエリーから挿入される cfselect コントロール : selected 属性を上書きします。cfselect を参照してください。 • cftree コントロール : cftreeitem の expand 属性を上書きします。yes の場合、以前に選択された要素を展開します。cftree の completePath 属性を yes に設定する必要があります。 • cfgrid コントロール : 効果はありません。このコントロールがデータをデータベースに再送信したかどうかの混乱を避けるために使われます。

属性	適用対象	必須 / オプション	デフォルト	説明
scriptSrc	Flash HTML XML	オプション	「説明」を参照	ColdFusion JavaScript ファイル (このタグおよび子タグが使用するクライアントサイドの JavaScript を含む cfform.js ファイルなど) が存在するディレクトリの Web ルートからの相対 URL を指定します。XML 形式フォームの場合、このディレクトリは XSLT スキンのデフォルトディレクトリも兼ねています。 この属性を使用する場合、指定したディレクトリは、/CFIDE/scripts ディレクトリと同じ構造である必要があります。たとえば、scriptsrc="/resources/myScripts" を指定した場合、ColdFusion の AJAX 機能で使用される JavaScript ファイルは、/resources/myScripts/ajax ディレクトリに存在している必要があります。 この属性は、ファイルがデフォルトの場所にない場合に役立ちます。この属性は、/CFIDE ディレクトリへのアクセスをブロックする一部のホスティング環境および構成で必要になることがあります。 場所は ColdFusion Administrator で設定します (デフォルトでは /CFIDE/scripts/ に設定されています)。 **メモ :** この属性を指定する場合は、CFIDE/scripts ディレクトリから指定したディレクトリに CF_RunActiveContent.js ファイルをコピーします。 cfajaximport タグの scriptsrc 属性など、1 ページに使用できる scriptsrc 属性は 1 つのみです。複数の cfform タグを使用する場合は、cfajaximport タグで scriptsrc 属性を指定でき、その指定がすべての HTML 形式のフォームに適用されます。

属性	適用対象	必須 / オプション	デフォルト	説明
skin	Flash XML	オプション	Flash: haloGreen XML: default.xsl	**Flash:** Halo カラーを使用して出力のスタイルを設定します。スキンによって、強調表示された要素や、選択された要素の表示色が決まります。 • haloSilver • haloBlue • haloGreen • haloOrange **XML:** XSL スキンを適用して結果の HTML をクライアントに表示するかどうかを指定します。次のいずれかを指定します。 • ColdFusion スキン名 : 指定したスキンを適用します。 • XSL ファイル名 : 指定したパスにあるスキンを適用します。 • none: XSL スキンを適用しません。CFML ページでは、name 属性で指定された変数に保存された XML を処理して、結果を表示する必要があります。 • 省略した場合または default: ColdFusion のデフォルトのスキンを使用します。 次の ColdFusion スキンを指定することができます。これらのスキンは、<ColdFusion の Web ルート >¥CFIDE¥scripts¥css にあります。 • basic • basiccss • beige • blue • lightgray • red • silver ファイル名は次のいずれかです。 • 絶対 URL • Web ルートに関連付けられた URL • 絶対ファイルパス • scripts フォルダまたは <ColdFusion の Web ルート >¥CFIDE¥scripts ディレクトリのサブディレクトリにあるファイルの名前。この場合は、.xsl 接尾辞を指定しないでください。

属性	適用対象	必須 / オプション	デフォルト	説明
style	HTML、Flash、XML	オプション		フォームに適用するスタイルです。HTML フォームまたは XML フォームでは、ColdFusion は style 属性をブラウザまたは XML に渡します。 Flash 形式では、CSS 形式のスタイル指定でなければなりません。Flash スタイルの指定に関する詳細については、『ColdFusion アプリケーションの開発』の Creating Forms in Flash を参照してください。
timeout	Flash	オプション	0	Flash のフォームデータをサーバーにキャッシュする秒数を示す整数値です。この値を 0 に設定すると、データはキャッシュされません。詳細については、『ColdFusion アプリケーションの開発』の Caching data in Flash forms を参照してください。
width	Flash XML	オプション	100%	フォームの幅です。ピクセル数を数値で指定します。Flash では、"width=60%" などのパーセント値を使用して、表示可能な幅の割合を指定できます。 メモ : 表の内部で Flash フォームを使用する場合、width 属性と height 属性は必須です。
wMode	Flash	オプション	window	Flash フォームが HTML ページ上の同じスペースを使用する他の表示可能コンテンツと相対して表示される方法を指定します。 • window: Flash フォームはページの最上位レイヤーにあり、ダイナミックな HTML ドロップダウンリストなど、同じスペースを共有する要素を見えなくします。 • transparent: Flash フォームは dhtml の z-index に従うので、他のコンテンツをフォームの上位レイヤーに置くことができます。Flash フォームが他のコンテンツの上位にある場合、下位のコンテンツはフォームの透過領域に表示されます。 • opaque: Flash フォームは dhtml の z-index に従うので、他のコンテンツをフォームの上位レイヤーに置くことができます。Flash フォームが他のコンテンツの上位にある場合、フォームは下位のコンテンツをブロックします。

注意 : XML でサポートされていない属性は、ColdFusion に同梱されているスキンでは処理されません。ただし、生成される XML には、form タグの HTML 名前空間属性として含まれています。

使用方法

このタグには終了タグが必要です。

cfform タグ内で次の ColdFusion フォームコントロールタグを使用することができます。

- cfapplet: HTML 形式および XML 形式の場合にのみ使用します。登録された Java アプレットを埋め込みます。
- cfformgroup: Flash 形式および XML 形式の場合にのみ使用します。子コントロールをグループ化して整列します。
- cfformitem: Flash 形式および XML 形式の場合にのみ使用します。水平方向の基準、垂直方向の基準、およびテキストをフォームに追加します。
- cfgrid: 表形式のデータを表示するためのグリッドコントロールを作成します。
- cfinput: 入力要素を作成します。
- cfselect: ドロップダウンリストボックスを作成します。
- cfslider: HTML 形式および XML 形式の場合にのみ使用します。スライダコントロールを作成します。
- cftextarea: 複数行のテキスト入力ボックスを作成します。

- cftree: ツリーコントロールを作成します。

HTML 形式の場合はすべてのタグが、Flash 形式の場合は cftree タグと cfgrid タグが、ブラウザ上で JavaScript のサポートを必要とします。cfapplet タグおよびアプレット形式の cfgrid タグ、cfslider タグ、cftree タグでは、クライアントが Java アプレットをダウンロードする必要があります。

cfform タグで Flash 形式を指定した場合、ColdFusion はフォーム本文の HTML をすべて無視します。すべてのフォームコントロールに対して、cfinput などの ColdFusion タグを使用する必要があります。HTML 形式の cfform タグに、Flash 形式の個別の cfgrid コントロールおよび cftree コントロールを含めることができます。

Flash 形式の場合、フォームで機密データ (クレジットカード情報など) をリクエストしないときは、timeout 属性の設定を検討してください。この属性を設定すると、ユーザーがブラウザの [戻る] ボタンを使って元のフォームに戻るときに " フォームデータが期限切れになりました。このページをブラウザにリロードしてください。" などのエラーメッセージが表示されないようにすることができます。詳細については、『ColdFusion アプリケーションの開発』の Caching data in Flash forms を参照してください。

注意：Flash 形式の場合、height 属性および width 属性を指定しないときは、ブラウザウィンドウの領域に等しいブラウザスペースが使用されます。フォームの後に他の出力が続く場合、ユーザーはスクロールしてその出力を表示しなければなりません。したがって、Flash フォームの後に他の出力を続けるときは、height および width の値を指定してください。表の内部で Flash フォームを使用する場合、width 属性と height 属性は必須です。

属性値のテキストに引用符が含まれる場合は、引用符を 2 つ続けて使用し、エスケープ処理する必要があります。

Flash フォームでの onError 属性の使用

onSubmit 検証または onBlur 検証を使用する場合、onError 属性を使用すると、検証エラーのある Flash フォームをユーザーが送信しようとするときに実行する ActionScript コードを指定することができます。次のとおりです。

- 有効な Flash 式を指定 (複数可) した場合、Flash はその式を実行します。
- この属性を省略した場合、Flash は適用可能なすべてのエラーメッセージを含むダイアログボックスを表示します。
- onError="" (空の文字列) を指定した場合、Flash は何もメッセージを表示しませんが、フォームは送信されません。

ActionScript ではエラー変数を使用してフィールドおよびエラーを判別することができます。エラーオブジェクトには次のフィールドがあります。

フィールド	内容
name	コントロールの CFML タグの name 属性です。
field	フィールドの名前として Flash で使われる内部名です。たとえば、_level0.field1 などです。
value	フィールド内の値です。
message	コントロールの CFML タグの message 属性です。

次の例は、onError 属性を使用する cfform タグを示しています。このタグでは、無効なエントリが入力された lastName フィールドがある accordion または tab navigator のタブを選択します。

```
<cfform name="form1" format="flash" width="800" height="500"
        onError="if (errors['lastName'] != undefined
            ){tabA.selectedIndex=0; _root.lastName.setFocus();}">
```

HTML フォームタグと属性の併用

HTML 形式の場合、cfform タグを使用すると、次の標準の HTML 要素を組み込むことができます。これらの要素は、Flash 形式では無視されます。

- HTML form タグの標準の属性とその値。これらの属性と値は、cfform がページに出力する form タグに挿入されます。たとえば、cfform で target や onMouseOver のような form タグ属性を使用することができます。

- HTML form タグ内に通常指定できる HTML タグ。たとえば、HTML の input タグを使用して、cfinput の他の機能を使用せずに、cfform 内に [送信] ボタンを作成することができます。

```
<cfform>
    <input type = "Submit" value = " update... ">
</cfform>
```

cfdiv、cflayout、cffpod、および cfwindow コントロールでのフォームの使用

cfdiv、cflayout、cffpod、および cfwindow タグでは、インタラクティブフォームのコンテナとして使用できる AJAX ベースのコントロールが作成されます。このような構造体を使用する場合は、フォーム情報が送信されたときに新しいページを表示するよりも、ダイナミックコードを使用してページを完全にリロードすることなく既存のページを修正するほうが便利です。そのためには、onSuccess 属性および onError 属性を使用します。

フォームデータが正しく送信された場合は、onSuccess 属性で指定された関数が呼び出されます。この関数は、ポッド、レイアウト、またはウィンドウを更新して送信結果を反映します。たとえば、追加データの表示や確認ウィンドウのポップアップ表示などを実行できます。この関数には引数を渡せません。

フォームデータの送信時にエラーが発生した場合は、onError 属性で指定された関数が呼び出されます。この関数は、エラーメッセージの表示などのエラー処理を実行します。この関数は 2 つの引数 (エラー番号とエラーメッセージ) を取る必要があります。

PDF フォームへのインタラクティブフィールドの追加

ColdFusion では、cfform タグを使用してスタティックなフォームフィールドだけでなくインタラクティブフォームフィールドも含む PDF フォームを作成できます。format="pdf" に設定された cfform タグは、cfdocument タグ内に存在する必要があります。cfdocument タグ内に存在できる cfform タグは 1 つのみです。

入力済みのフォームを HTTP Post としてサーバーに送信することもできますが、PDF 全体をバイナリストリームとして送信することもできます。PDF を送信する場合は、次のように cffile タグを使用して、入力済みの PDF フォームをハードドライブに保存できます。

<cffile action="write" file="c:¥savedpdf.pdf" output="#PDF.content#">

このタグを使用して出力の操作と抽出を行えます。

PDF フォームの生成では、次の cfform 属性のみがサポートされています。

- action
- format
- method
- name
- onSubmit
- skin
- style

LiveCycle Designer または Acrobat によって生成された既存の PDF フォームを埋め込むには、このタグを使用します。

例

```
 <h3>cfform Example</h3>
<!--- If Form.oncethrough exists, the form has been submitted. --->
<cfif IsDefined("Form.oncethrough")>
    <cfif IsDefined("Form.testVal1")>
        <h3>Results of Radio Button Test</h3>
        <cfif Form.testVal1>Your radio button answer was yes
        <cfelse>Your radio button answer was no
        </cfif>
    </cfif>
    <h3>Results of Checkbox Test</h3>
    <cfif IsDefined("Form.chkTest2")>
        Your checkbox answer was yes
    <cfelse>
        Your checkbox answer was no
    </cfif>
    <cfif IsDefined("Form.textSample") AND Form.textSample is not "">
        <h3>Results of Credit Card Input</h3>
        Your credit card number, <cfoutput>#Form.textSample#</cfoutput>,
        was valid under the MOD 10 algorithm.
    </cfif>
    <cfif IsDefined("Form.sampleSlider")>
        <cfoutput>
            <h3>You gave this page a rating of #Form.sampleSlider#</h3>
        </cfoutput>
    </cfif>
    <hr noshade="True">
</cfif>

<!--- Begin by calling the cfform tag. --->
<cfform name="cfformexample">
    <h4>This example displays radio button input type for cfinput.</h4>
    Yes <cfinput type = "Radio" name = "TestVal1" value = "Yes" checked>
    No <cfinput type = "Radio" name = "TestVal1" value = "No">
    <h4>This example displays checkbox input type for cfinput.</h4>
    <cfinput type = "Checkbox" name = "ChkTest2" value = "Yes">
    <h4>This shows client-side validation for cfinput text boxes.</h4>
    (<i>This item is optional</i>)<br>
    Please enter a credit card number:
    <cfinput type = "Text" name = "TextSample"
        message = "Please enter a Credit Card Number"
        validate = "creditcard" required = "No">
    <h4>This example shows the use of the cfslider tag.</h4>
    Rate your approval of this example from 1 to 10 by sliding control.<br>
    1 <cfslider name = "sampleSlider" width="100"
            label = "Page Value: " range = "1,10"
            message = "Please enter a value from 1 to 10"> 10
    <p><cfinput type = "submit" name = "submit" value = "show me the result">
    <cfinput type = "hidden" name = "oncethrough" value = "Yes"></p>
</cfform>
```

簡単な XML フォーム

次の例は、簡単な XML 形式のフォームを示しています。ColdFusion に同梱されている default.xsl 変換を使用して、表示する HTML 出力を生成します。

```
<cfform name="testXForm" format="XML" skin="basic">
<!--- Use cfformgroup to put the first and last names on a single line. --->
    <cfformgroup type="horizontal">
        <cfinput type="text" name="firstname" label="First Name:" value="Robert">
        <cfinput type="text" name="lastname" label="Last Name:" value="Smith">
</cfformgroup>
<cfinput type="password" name="password" label="Password:" value="">
<cfinput type="hidden" name="hidden" label="hidden:" value="">
<cfselect name="state" style="width:200" label="State">
    <option>California</option>
    <option selected>Utah</option>
    <option>Iowa</option>
    <option selected>New York</option>
    </cfselect>
    <cftextarea name="description" label="Description:" rows="5" cols="40">
    this is sample text.</cftextarea>
</cfform>
```

簡単な PDF フォーム

```
<cfdocument format="pdf">
    <cfdocumentsection ../>
    ...
    ...
    <cfform type="html/xform">
        <cfinput type="textbox" name="employeeName" value="#fullName#" readonly="true">
        <cfinput type="textbox" name="employeeID" value="#id#" readonly>
        <cfselect name="contributionPercentage" options="#optionsStruct#" required="true">
        <cfinput type="submit" name="SubmitAsHTTPPost">
        <cfinput type="submit" name="SubmitAsPDF" submitType="PDF">
    </cfform>
    ...
    ...
    <cfdocumentsection ../>
</cfdocument>
```

cfformgroup

説明

複数のフォームコントロール用のコンテナコントロールを作成します。Adobe Flash フォームおよび XML フォームの cfform タグ本文で使用します。HTML フォームでは無視されます。

カテゴリ

フォームタグ

シンタックス

```
<cfformgroup
    type = "group type"
    label = "label"
    style = "style specification"
    selectedIndex = "page number">
    width = "pixels"
    height = "pixels"
    enabled = "yes|no"
    visible = "yes|no"
    onChange = "ActionScript expression"
    tooltip = "text"
    id = "unique identifier">
    ...ColdFusion forms controls...
</cfformgroup>

OR

<cfformgroup
    type = "repeater"
    query = "query object"
    maxrows = "integer">
    startrow = "row number"
    ...ColdFusion forms controls
</cfformgroup>
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfapplet、cfcalendar、cfform、cfformitem、cfgrid、cfinput、cfselect、cfslider、cftextarea および cftree

履歴

ColdFusion MX 7: このタグが追加されました。

属性

次の表に、Flash フォームの属性とその動作を示します。XML の場合、特記なき限り、属性は XML に渡されますが、ColdFusion に同梱されている基本的な XSL スタイルシートでは解釈されません。

注意：XML でサポートされていない属性は、ColdFusion に同梱されているスキンでは処理されません。ただし、これらの属性は生成された XML に含まれます。

属性	必須 / オプション、形式	デフォルト	説明
type	必須、 Flash および XML		**XML**: XSLT で定義されたすべての XForms グループタイプです。ColdFusion に同梱されている XSL スキンは、次のタイプをサポートします。 • horizontal: フォーム内で子タグを水平方向に整列して、このタグの label 属性を子の左に配置します。 • vertical: フォーム内で子タグを垂直方向に整列して、このタグの label 属性を子の左に配置します。 • fieldset: HTML fieldset タグに対応します。通常、子の周囲をボックスで囲み、その最上部の一部を凡例テキストに置き換えることにより、子をグループ化します。凡例を指定するには、label 属性を使用します。ボックスの大きさを指定するには、style 属性の height および width の値を使用します。子タグを垂直方向に整列するには、このフォームグループ内で cfformgrouptype="vertical" を明示的に使用します。
			Flash: 次のいずれかになります。 • repeater: クエリーオブジェクトの各行について、cfformgroup の子タグのインスタンスをダイナミックに作成します。その際に、行の数が変わっても ColdFusion で Flash SWF ファイルを再コンパイルする必要はありません。 • horizontal: フォーム内で子タグを水平方向に整列して、このタグの label 属性を子の左に配置します。このタグを使用して、個々のコントロールを水平方向に整列します。 • vertical: フォーム内で子タグを垂直方向に整列して、このタグの label 属性を子の左に配置します。このタグを使用して、個々のコントロールを垂直方向に整列します。 • hbox: 子を水平方向に整列します。このタイプを使用して、フォームコントロールのグループを水平方向に整列します。この属性を使用して個々のコントロールを水平方向に整列しないでください。その場合、子コントロールは正常に整列されません。代わりに、horizontal を使用してください。 • *vbox*: 子を垂直方向に整列します。このタイプを使用して、フォームコントロールのグループを垂直方向に整列します。この属性を使用して個々のコントロールを垂直方向に整列しないでください。その場合、子コントロールは正常に整列されません。代わりに、vertical を使用してください。
			• hdividedbox: 子を水平方向に整列します。それぞれの子はボーダー付きのボックス内にあり、ボックス間には分割線があります。ユーザーは、この分割線を動かして子の相対サイズを変更することができます。この属性を含むタグを使用して、フォームコントロールのグループを水平方向に整列します。この属性を使用して、個々のコントロールを水平方向に整列することはできません。 • vdividedbox: 子を垂直方向に整列します。それぞれの子はボーダー付きのボックス内にあり、ボックス間には分割線があります。ユーザーは、この分割線を動かして子の相対サイズを変更することができます。このタイプを使用して、フォームコントロールをグループ化します。たとえば、hbox フォームグループ内の 1 つの単位としてグループ化します。この属性を使用して、個々のタグを垂直方向に整列しないでください。 • panel: label 属性のテキスト、ボーダー、および垂直方向に整列された子を持つコンテンツ領域を含むタイトルバーで構成されたコンテナです。 • tile: 子を長方形のグリッド内に配置します。 • accordion: 拡張および収縮するアコーディオンプリーツ内に子を配置します。cfformgroup type="page" タグを使用して、各プリーツを定義します。 • tabnavigator: タブ付きのダイアログボックス内に子を配置します。cfformgroup type="page" タグを使用して、各タブを定義します。 • page: 子タグを、垂直方向に整列して、親 tabnavigator の単一のタブ内またはアコーディオンコンテナのプリーツ内に配置します。

属性	必須 / オプション、形式	デフォルト	説明
query	type= repeater の場合に必須、それ以外の場合は無視 Flash		repeater とともに使用するクエリーです。Flash は、クエリー内の各行について、cfformgroup タグのそれぞれの子タグのインスタンスを作成します。子タグ内の bind 属性を使って、インスタンスのクエリー行のデータを使用できます。
enabled	オプション、 Flash	yes	フォームグループ内のコントロールが有効であるかどうかを指定するブール値です。無効なコントロールはライトグレーで表示されます。
height	オプション、 Flash		グループコンテナの高さです (単位 : ピクセル)。この属性を省略すると、Flash はコンテナの高さを自動的にサイズ設定します。Flash の repeater タイプの場合は無視されます。
id	オプション、 Flash		フォームグループの一意の識別子です。 tabnavigator または accordion タイプを使用するときは、カスタムの ActionScript でコントロールを参照できるように id 属性を指定します。
label	オプション、 Flash および XML		フォームグループに適用するラベルです。 Flash では次を実行します。 • ページまたはパネルグループの場合、対応するアコーディオンプリーツ、tabnavigator タブ、またはパネルタイトルバーに配置するラベルを指定します。Flash の水平または垂直フォームグループの場合、グループの左に配置するラベルを指定します。 • repeater、hbox、hdividedbox、vbox、vdividedbox、tile、accordion、および tabnavigator の各タイプについては、Flash では無視されます。
maxrows	オプション、 Flash		repeater タイプの場合にのみ使用され、それ以外の場合は無視されます。 Flash フォームの repeater で使用するクエリー行の最大数を指定します。startrow 属性とこの値の合計を超える数のクエリー行がある場合、repeater では残りの行を使用しません。
onChange	オプション、 Flash		tabnavigator タイプおよび accordion タイプのみ : 新しいタブまたは accordion ページが選択されたときに実行する 1 つまたは複数の ActionScript 式です。 **メモ :**onChange イベントが発生するのは、当該フォームが最初に表示されるときです。
selectedIndex	オプション、 Flash のみ		accordion タイプおよび tabnavigator タイプの場合にのみ使用されます。それ以外の場合は無視されます。開くときに表示するページコントロールを指定します。ここで、0 (1 ではない) はグループに定義された最初のページコントロールを指定します。
startrow	オプション、 Flash	0	repeater タイプの場合にのみ使用され、それ以外の場合は無視されます。 Flash フォームの repeater で使用するクエリーの最初の行の行番号を指定します。この属性では 0 が基準です。最初の行は 0 です。ほとんどの ColdFusion タグのような 1 ではありません。
style	オプション、 Flash および XML		**Flash**: CSS 形式の Flash スタイル指定です。Flash スタイルの指定に関する詳細については、『ColdFusion アプリケーションの開発』の Creating Forms in Flash を参照してください。 **XML**: インライン CSS スタイル指定です。
tooltip	オプション、 Flash		マウスポインタをフォームグループ領域の上に置いたときに表示されるテキストです。フォームグループ内のコントロールでもツールチップを示す場合、マウスポインタをそのコントロールの上に置くと Flash はそのコントロールのツールチップを表示します。
visible	オプション、 Flash	yes	フォームグループ内のコントロールが表示されるかどうかを指定するブール値です。コントロールが表示されない場合、表示されるコントロールが使用するスペースは空白になります。
width	オプション、 Flash および XML		グループコンテナの幅です (単位 : ピクセル)。この属性を省略すると、Flash はコンテナの幅を自動的にサイズ設定します。Flash の repeater タイプの場合は無視されます。

使用方法

このタグには終了タグが必要です。cfform タイプが HTML の場合、このタグは無視されます。すべてのタグ本文の内容は、cfformgroup で囲まれていないかのように解釈されます。

Flash 形式のフォームでは、このタグがフォームの内容を構成します。このタグが子タグをグループ化して整列します。このタグの本文には、次のタグを含めることができます。その他のタグとテキストはすべて無視されます。

- cfformitem
- cfcalendar
- cfgrid
- cfinput
- cfselect
- cftextarea
- cftree

Flash フォームでのこのタグの使用方法については、『ColdFusion アプリケーションの開発』の Creating Forms in Flash を参照してください。

XML 形式では、ColdFusion はこのタグとその属性を XML に渡します。XML を処理するのは、スキン XSLT です。ColdFusion の基本的なスキンは、horizontal、vertical、および dualselectlist のスタイルのみをサポートします。XML フォームでのこのタグの使用方法については、『ColdFusion アプリケーションの開発』の Creating Forms in Flash を参照してください。

例

単一の cfformgroup タグを使用する XML フォームの簡単な例については、cfform を参照してください。

次の例は、cfformgroup タグを使用して Flash フォームに要素を整列する方法を示しています。hdividedbox コンテナを作成します。各 hdividedbox コンテナには、vbox コンテナが 1 つ含まれています。左のボックスには、見出しテキストと 2 つのラジオボタンがあります。右のボックスには、見出しテキストと 3 つのチェックボックスがあります。

```
 <h3>Simple cfformgroup Example</h3>
<cfform name="myform" height="450" width="500" format="Flash" >
    <cfformgroup type="hdividedbox" >
        <cfformgroup type="VBox">
            <cfformitem type="text" height="20">
                Pets:
            </cfformitem>
            <cfinput type="Radio" name="pets" label="Dogs" value="Dogs" checked>
            <cfinput type="Radio" name="pets" label="Cats" value="Cats">
        </cfformgroup>

        <cfformgroup type="VBox">
            <cfformitem type="text" height="20">
                Fruits:
            </cfformitem>
            <cfinput type = "Checkbox" name="chk1" Label="Apples" value="Apples">
            <cfinput type="Checkbox" name="chk2" Label="Bananas" value="Bananas">
            <cfinput type="Checkbox" name="chk3" Label="Pears" value="Pears">
        </cfformgroup>
    </cfformgroup>
</cfform>
```

さらに複雑な次の例は、cfformgroup タグを使用して Flash フォームにコントロールを整列する方法をより詳細に示しています。cfformgroup の本文で使用できるテキストの書式設定機能も多数示しています。フォームを送信する際には、送信データを示すために、ページでは Form スコープの内容をダンプします。

```
 <h2>cfformgroup Example</h2>
<cfif IsDefined("form.oncethrough")>
    <h3>The form submitted the following information to ColdFusion:</h3>
    <cfdump var="#form#"><br><br><br>
</cfif>

<h3>A Flash form using cfformgroup tags</h3>
<cfform name="myform" height="450" width="500" format="Flash">

<!--- The following formgroup shows how you can present formatted text. --->
    <cfformitem type="html">
        <b><font color="#FF0000" size="+4" face="serif">
        This form has two tabs, asking for the following:</font></b><br>
        <li>contact information</li>
        <li><i>preferences</i></li>
        <b>Try entering information on both tabs</b><br>
        Submit the form and see what ColdFusion gets in the Forms scope.</b><br>
        <a href="http://www..com/" target="_blank">
        <font color="#0000FF"><u>
        This link displays the home page in a new browser window
        </u></font></a><br>
         <br>
    </cfformitem>

<!--- Use a tabnavigator with two tabs for user input. --->
    <cfformgroup type="tabnavigator" height="220">
        <cfformgroup type="page" label="Contact Information">
            <!--- Align the first and last name fields horizontally --->
            <cfformgroup type="horizontal" label="Your Name">
                <cfinput type="text" required="Yes" name="firstName" label="First"
                    value="" width="100"/>
                <cfinput type="text" required="Yes" name="lastName" label="Last"
                    value="" width="100"/>
            </cfformgroup>
            <cfformitem type="html"><textformat indent="95"><font size="-2">
                Flash fills the email field in automatically.
                You can replace any of the text.
                </font></textformat>
            </cfformitem>
            <!--- The bind attribute gets the field contents from the firstname and lastName
                fields as they get filled in. --->
            <cfinput type="text" name="email" label="email"
                bind="{firstName.text}.{lastName.text}@mm.com">

            <cfinput type="text" name="phone" validate="telephone" required="Yes"
                label="Phone Number">
        </cfformgroup>

        <cfformgroup type="page" label="Preferences">
            <cfformitem type="text" height="30">
                <b>Tell us your preferences</b>
            </cfformitem>
            <!--- Put the pet selectors to the left of the fruit selectors. --->
            <cfformgroup type="hbox">
            <!--- Group the pet selector box contents, aligned vertically. --->
                <cfformgroup type="vbox">
                    <cfformitem type="text" height="20">
                        Pets:
                    </cfformitem>
                    <cfformgroup type="vertical">
                        <cfinput type="Radio" name="pets" label="Dogs" value="Dogs"
                            checked>
```

```
                    <cfinput type="Radio" name="pets" label="Cats" value="Cats">
                </cfformgroup>
            </cfformgroup>
        <!--- Group the fruit selector box contents, aligned vertically. --->
            <cfformgroup type="vbox">
                <cfformitem type="text" height="20">
                    Fruits:
                </cfformitem>
                <cfformgroup type="tile" width="200" label="Tile box">
                    <--- Flash requires unique names for all controls --->
                    <cfinput type = "Checkbox" name="chk1" Label="Apples"
                        value="Apples">
                    <cfinput type="Checkbox" name="chk2" Label="Bananas"
                        value="Bananas">
                    <cfinput type="Checkbox" name="chk3" Label="Pears"
                        value="Pears">
                    <cfinput type="Checkbox" name="chk4" Label="Oranges"
                        value="Oranges">
                    <cfinput type="Checkbox" name="chk5" Label="Grapes"
                        value="Grapes">
                    <cfinput type="Checkbox" name="chk6" Label="Cumquats"
                        value="Cumquats">
                </cfformgroup>
            </cfformgroup>
        </cfformgroup>
    </cfformgroup>

    <cfformgroup type="horizontal">
        <cfinput type = "submit" name="submit" width="100" value = "Show Results">
        <cfinput type = "reset" name="reset" width="100" value = "Reset Fields">
        <cfinput type = "hidden" name="oncethrough" value = "Yes">
    </cfformgroup>
</cfform>
```

cfformitem

説明

Flash フォームに水平線、垂直線、スペース、またはテキストを挿入します。Flash フォームおよび XML フォームの cfform または cfformgroup タグの本文で使用されます。HTML フォームでは無視されます。

カテゴリ

フォームタグ

シンタックス

```
<cfformitem
    type = "hrule|vrule|spacer"
    height = "pixels"
    style = "style specification"
    visible = "yes|no"
    width = "pixels"/>

OR

<cfformitem
    type = "html|text|script"
    bind = "bind expression"
    enabled = "yes|no"
    height = "pixels"
    style = "style specification"
    tooltip = "text"
    visible = "yes|no"
    width = "pixels">
    ...text</cfformitem>
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfapplet、cfform、cfformgroup、cfgrid、cfinput、cfselect、cfslider、cftextarea、cftree、『ColdFusion アプリケーションの開発』の Adding text, images, rules, and space with the cfformitem tag

履歴

ColdFusion MX 7.01: type 属性の値として "script" が追加されました。

ColdFusion MX 7: タグが追加されました。

属性

次の表に、Flash フォームの属性とその動作を示します。XML 形式の場合、特記なき限り、属性は XML に渡されますが、ColdFusion に同梱されている基本的な XSL スタイルシートでは解釈されません。

注意："Flash の場合のみ " と記載されている属性は、ColdFusion に付属するスキンでは処理されません。ただし、text タイプと html タイプを除くすべてのコントロールでは、生成される XML にこれらの属性が含まれます。

属性	必須 / オプション、形式	デフォルト	説明
type	必須、 Flash および XML		**Flash:** • html: このタグの本文のテキストをフォームに配置します。Flash フォームでは、テキスト内に a、b、br、font、i、img、li、p、textformat、u のテキスト書式設定タグを使用できます。ほとんどのタグが HTML タグに対応します。これらの書式設定タグの使用方法の詳細については、Flash のドキュメントを参照してください。style 属性には、テキストの書式に対するタイプの効果はありません。 • text: このタグの本文のテキストを、マークアップを解釈せずにフォームにそのまま配置します。テキストの全体的な外観は style 属性を使用して制御できます。 • spacer: 指定した高さと幅の非表示のスペースをフォームに配置します。フォームコントロールの間にスペースを配置するために使用します。このタグには子タグを指定できません。 • hrule: 水平方向の罫線をフォームに配置します。このタグには子タグを指定できません。 • vrule: 垂直方向の罫線をフォームに配置します。このタグには子タグを指定できません。 • script: Flash フォームに関数を作成でき、64 KB の制限に達する危険性を少なくします。
			XML: • html: XML の xf:output 要素の CDATA セクションに CFML タグの本文のテキストを配置します。 • text: CFML タグの本文のテキストを XML 形式 (< などの文字をエスケープ処理) にして、XML の xf:output 要素の CDATA セクションに配置します。 • hrule: hr タグを出力に配置します。style 属性を使用して、高さや幅などのすべての罫線特性を指定します。このタグには子タグを指定できません。 その他の文字列 : appearance 属性としてのタイプ名で XML の xf:group 要素を生成します。CFML タグの本文は、cf:attributename="body" 要素の CDATA セクションに配置されます。ColdFusion に同梱されている XSL 変換では、これらの要素は無視されます。
bind	オプション、 Flash		フィールドに他のフォームのフィールドの情報を挿入する Flash のバインド式です。この属性を使用した場合、cftextitem タグの本文に指定したテキストは無視されます。この属性は、cfformitem タグが cfformgrouptype="repeater" タグ内にある場合に役立ちます。 詳細については、cfinput タグの説明の「Flash フォームデータのバインディング」を参照してください。
enabled	オプション、 Flash	yes	コントロールを有効にするかどうかを指定するブール値です。無効なテキストはライトグレーで表示されます。スペースや罫線には効果がありません。
height	オプション、 Flash		項目の高さです (単位 : ピクセル)。この属性を省略すると、Flash は高さを自動的にサイズ設定します。ColdFusion の XSL スキンでは、代わりに style 属性を使用します。
style	オプション、 Flash および XML		**Flash:** • CSS 形式のスタイル指定でなければなりません。 • type 属性が html の場合は無視されます。 Flash スタイルの指定に関する詳細については、『ColdFusion アプリケーションの開発』の Creating Forms in Flash を参照してください。spacer タイプと一緒には使用しません。 **XML:** • style 属性が XML に渡されます。ColdFusion のスキンの場合は、生成された HTML に style 属性が含まれています。

属性	必須 / オプション、形式	デフォルト	説明
tooltip	オプション、Flash		マウスポインタをコントロールの上に置いたときに表示されるテキストです。スペースには効果がありません。
visible	オプション、Flash	yes	コントロールを表示するかどうかを指定するブール値です。表示されないコントロールが使用するスペースは空白です。スペースには効果がありません。
width	オプション、Flash		項目の幅です (単位 : ピクセル)。この属性を省略すると、Flash は幅を自動的にサイズ設定します。ColdFusion の XSL スキンでは、代わりに style 属性を使用します。

使用方法

このタグには、終了タグ、または開始タグの終了文字の前にスラッシュ (次の例を参照) が必要です。

```
<cfformitem type="hrule" />
```

Flash フォームでのこのタグの使用方法については、『ColdFusion アプリケーションの開発』の Creating Forms in Flash を参照してください。

例

次の例は、水平方向の基準およびテキストを使った簡単な Flash フォームを示しています。

```
<h3>cfformitem Example</h3>
<cfform name="myform" height="450" width="500" format="Flash" >
    <cfformitem type="hrule" />
    <cfformitem type="text">
        This simple form has two hrule cfformitem tags around the cfformitem tag that
            contains this text.
    </cfformitem>
    <cfformitem type="hrule" />
</cfform>
```

さらに複雑なフォームについては、cfformgroup を参照してください。

cfftp

説明

ユーザーによる FTP (File Transfer Protocol: ファイル転送プロトコル) のオペレーションの実装を許可します。

カテゴリ

ファイル管理タグ、インターネットプロトコルタグ

シンタックス

タグのシンタックスは action 属性の値によって異なります。次のセクションを参照してください。

- cfftp: FTP サーバー接続の確立と切断
- cfftp: セキュア FTP サーバー接続の確立と切断
- cfftp: 接続 : ファイルおよびディレクトリのオペレーション
- cfftp action = "listDir"

関連項目

cfhttp、cfldap、cfmail、cfpop、『ColdFusion アプリケーションの開発』の Interacting with Remote Servers の Performing file operations with cfftp

履歴

ColdFusion 8：セキュア FTP をサポートするために、fingerprint、key、paraphrase、および secure 属性が追加されました。action 属性に values ="quote"、"site"、"allo"、および "acct" が追加されました。

ColdFusion MX 7: ファイルとディレクトリのオペレーションについての result 属性が追加されました。

ColdFusion MX: agentname 属性は非推奨になりました。今後のリリースではこのタグは機能せずエラーを引き起こす可能性があります。

使用方法

cfftp タグは、ColdFusion サーバーと FTP サーバーとの間でファイルを移動するために使用します。

このタグを使用して、ColdFusion サーバーとクライアントブラウザとの間でファイルを移動することはできません。この操作を行うには、次のようにします。

- クライアントから ColdFusion サーバーにファイルを転送するには、cffile action = "upload" を使用します。
- ColdFusion サーバーからクライアントにファイルを転送するには、cfcontent タグを使用します。

セキュリティ設定

ColdFusion のセキュリティ設定で、cfftp タグの実行を抑制することができます。複数のカスタマが使用するサーバー上で ColdFusion アプリケーションを実行する場合、カスタマが移動できるファイルのセキュリティを考慮する必要があります。詳細については、『ColdFusion 設定と管理』の Administering Security を参照してください。

cfftp: FTP サーバー接続の確立と切断

説明

FTP サーバーとの接続を確立するには、open アクションと connection 属性を使用します。

シンタックス

```
<cfftp
    action = "open|close|quote|site|allo|acct"
    actionparam = "command or account information"
    buffersize = "number"
    connection = "name"
    passive = "yes|no"
    password = "password"
    port = "port"
    proxyServer = "proxy server"
    retryCount = "number"
    server = "server"
    stopOnError = "yes|no"
    timeout = "time-sout in seconds"
    username = "name">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfhttp、cfldap、cfmail、cfpop

属性

属性	必須 / オプション	デフォルト	説明
action	必須		実行する FTP オペレーションです。 • open: FTP 接続を確立します。 • close: FTP 接続を終了します。 • quote: FTP サーバーにコマンドをそのまま送信します。 • site: サイト固有のコマンドを実行します。 • allo: サーバー上のオペレーション (大きなファイルの配置など) に使用するメモリを割り当てます。 • acct: アカウント情報を要求するシステムにアカウント情報を送信します。
actionparam	オプション		action が quote、site、または acct に設定されているときにのみ使用します。action が quote または site のときは、コマンドを指定します。action が acct のときは、アカウント情報を指定します。
buffersize	オプション		バッファーサイズです (単位 : バイト)。
connection	オプション (ただし open または close のときは常に使用)		FTP 接続の名前です。username、password、および server 属性を指定した場合で、対応する接続がないときは、ColdFusion により新しい接続が作成されます。同じ接続名で cfftp を呼び出すと、同じ接続が再利用されます。
passive	オプション	no	• yes: passive モードを有効にします。 • no
password	action="open" の場合は必須		ユーザーがログインするためのパスワードです。
port	オプション	21	接続先のリモートポートです。
proxyServer	オプション		文字列。プロキシアクセスを指定した場合に使用するプロキシサーバーの名前です。
retryCount	オプション	1	エラーがレポートされるまでの再試行回数です。
server	action="open" の場合は必須		接続先の FTP サーバーです (例 : ftp.myserver.com)。
stopOnError	オプション	yes	• yes: 処理が停止し、該当するエラーが表示されます。 • no: secure="no" の場合は、次の変数に値が挿入されます。 • cfftp.succeeded: yes または no です。 • cfftp.errorCode: エラー番号です。IETF Network Working Group RFC 959: File Transfer Protocol (FTP) を参照してください (www.ietf.org/rfc/rfc0959.txt)。 • cfftp.errorText: メッセージテキストです。 条件付きオペレーションには、cfftp.errorCode を使用します。cfftp.errorText は使用しないでください。
timeout	オプション	30	個々のデータリクエストオペレーションなど、すべてのオペレーションに関するタイムアウト値です (単位 : 秒)。
username	action="open" の場合は必須		FTP オペレーションで渡すユーザー名です。

使用方法

cfftp action="open" で接続を確立するときに connection 属性に名前を指定しておくと、他の FTP オペレーションを実行するときにその接続を再使用できるように、接続がキャッシュされます。キャッシュされた接続を後続の FTP オペレーションで使用する場合、username、password、server といった接続属性を指定する必要はありません。同じ connection 名を使用する FTP オペレーションは、キャッシュされた接続に保管されている情報を自動的に使用します。キャッシュされた接続を使用すると、接続時間を節約し、ファイル転送のパフォーマンスを向上させることができます。

1 つの単純な FTP オペレーション (GetFile や PutFile など) を行うたびに接続を開く必要はありません。

close 以外のアクションでは、buffersize を指定して内部バッファーサイズを設定できます。quote、site、allo、acct のいずれかをアクションとして指定して secure="yes" に設定した場合は、エラーが発生します。site または quote をアクションとして指定するときは、FTP サーバーに送信するコマンドを actionparam 属性で指定します。アクションが site のときは、actionparam 属性を使用してサイト固有の情報を指定します。

セッションの間またはセッション後も接続を開いたままにするには、その接続名を Session スコープまたは Application スコープに挿入します。たとえば、connection="Session.FTPConnection" を指定します。ただし、この方法を使う場合は、すべての FTP オペレーションで完全な変数名を指定し、終了時に close アクションを使用する必要があります。接続を開いたままにしておくと、他のユーザーが FTP サーバーを使用できません。このため、接続はできるだけ早く閉じるようにしてください。接続名をセッション変数またはアプリケーション変数に割り当てなかった場合は、接続を現在のページのためだけに開いたままになるので、手動で閉じる必要はなくなります。

キャッシュされている接続を変更するとき (retryCount 値や timeout 値を変更する場合など) には、接続の再確立が必要になることがあります。

例

```
 <p>cfftp lets users implement File Transfer Protocol operations. By default, cfftp caches
    an open connection to an FTP server.</p>
<p>cfftp operations are usually of two types:</p>
<ul>
    <li>Establishing a connection
    <li>Performing file and directory operations
</ul>
<p>This example opens and verifies a connection, lists the files in a directory, and closes
    the connection.</p>
<p>Open a connection</p>
<cfftp action = "open"
    username = "anonymous"
    connection = "My_query"
    password = "youremail@email.com"
    server = "ftp.tucows.com"
    stopOnError = "Yes">
<p>Did it succeed? <cfoutput>#cfftp.succeeded#</cfoutput>
<p>List the files in a directory:
<cfftp action = "LISTDIR"
    stopOnError = "Yes"
    name = "ListFiles"
    directory = "/"
    connection = "my_query">
<cfoutput query = "ListFiles">
    #name#<br>
</cfoutput>

<p>Close the connection:</p>
<cfftp action = "close"
    connection = "My_query"
    stopOnError = "Yes">
<p>Did it succeed? <cfoutput>#cfftp.succeeded#</cfoutput>
```

cfftp: セキュア FTP サーバー接続の確立と切断

説明

セキュア FTP サーバーとの接続を確立するには、open アクションと connection 属性を使用して、secure = "yes" を指定し、key、passphrase、および fingerprint を適切に指定します。

```
<cfftp
    action = "open|close"
    connection = "name"
    fingerprint = "ssh-dss.ssh-rsa"
    key = "private key"
    passive = "yes|no">
    passphrase = "passphrase"
    password = "password"
    port = "port"
    proxyServer = "proxy server"
    retryCount = "number"
    secure = "yes|no"
    server = "server"
    stopOnError = "yes|no"
    timeout = "time-out in seconds"
    username = "name">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfhttp、cfldap、cfmail、cfpop

属性

属性	必須 / オプション	デフォルト	説明
action	必須		実行する FTP オペレーションです。 • open: FTP 接続を確立します。 • close: FTP 接続を終了します。
connection	オプション (ただし open または close のときは常に使用)		FTP 接続の名前です。username、password、および server 属性を指定した場合で、対応する接続がないときは、ColdFusion により新しい接続が作成されます。同じ接続名で cfftp を呼び出すと、同じ接続が再利用されます。
fingerprint	オプション。server、username、および password が指定されている場合にのみに使用		ssh-dss.ssh-rsa という形式を取るホストキーのフィンガープリントです。これは、指定された server 属性を示す 16 バイトの一意の識別子です。fingerprint は hh:hh:hh:hh::hh:hh:hh:hh という形式を取る 8 対の 16 進数で構成されます。ColdFusion は、fingerprint の値が指定されている場合にのみリモートサーバーの fingerprint を確認します。
key	action="open" の場合は必須 (secure="yes" の場合は、password または key が必須)		公開キーを利用した認証です。ユーザーの秘密キーの絶対パスを指定します。秘密キーを所有している場合は、秘密キーを使用して作成したシグネチャを送信することによって認証を行えます。そのキーがユーザーの認証に有効であり、シグネチャが有効であることがサーバーによって確認される必要があります。両方が有効でないと認証は受け入れられません。
passive	オプション	no	secure="no" の場合にのみ有効です。 • yes: passive モードを有効にします。 • no

属性	必須 / オプション	デフォルト	説明
passphrase	オプション。key が指定されている場合にのみ使用		秘密キーは暗号化されてクライアントホストに保存されているので、シグネチャの生成を有効にするためには、ユーザーがパスフレーズを入力する必要があります。
password	action="open" の場合は必須 (secure="yes" の場合は、password または key が必須)		ユーザーがログインするためのパスワードです。
port	オプション	21	接続先のリモートポートです。
proxyServer	オプション		文字列。プロキシアクセスを指定した場合に使用するプロキシサーバーの名前です。
retryCount	オプション	1	エラーがレポートされるまでの再試行回数です。
secure	オプション	no	• yes: セキュア FTP を有効にします。 • no
server	必須 (action = "open" の場合)		接続先の FTP サーバーです (例 : ftp.myserver.com)。
stopOnError	オプション	no	• yes: 処理が停止し、該当するエラーが表示されます。 • no: secure="yes" の場合は、次の変数に値が挿入されます。 • セキュア FTP サーバーへの接続に失敗した場合は、処理が停止されて該当するエラーメッセージが表示されます。 • cfftp.succeeded: yes または no です。 • cfftp.errorCode: エラー番号です。 • cfftp.errorText: メッセージテキストです。 • すべてのファイルオペレーションに関して、次のエラーコードが返されます。 SSH-CONNECT 25 SSH_MSG_USERAUTH_FAILURE 51 SSH_MSG_USERAUTH_SUCCESS 52 SSH_MSG_REQUEST_SUCCESS 81 SSH_MSG_REQUEST_FAILURE 82 条件付きオペレーションには、cfftp.errorCode を使用します。cfftp.errorText は使用しないでください。
timeout	オプション	30	個々のデータリクエストオペレーションなど、すべてのオペレーションに関するタイムアウト値です (単位 : 秒)。
username	action="open" の場合は必須		FTP オペレーションで渡すユーザー名です。

使用方法

cfftp タグでは、対称暗号化または非対称暗号化を使用して Secure Shell (SSH) サーバーとの接続を確立できます。対称暗号化を使用するには、secure="yes"、ユーザー名、パスワード、接続、およびフィンガープリントを指定します。非対称暗号化を使用するには、サーバーにアクセスする権限を持つユーザーごとに、秘密キーと公開キーのペアを生成します。各認証ユーザーの公開キーは、サーバー上に保管されます。各ユーザーの秘密キーは暗号化されてユーザーのコンピュータ上に格

納されます。SSH サーバーとの接続を確立するには、secure="yes"、ユーザー名、パスワード (または秘密キーとそれを復号するためのパスフレーズ)、接続、およびフィンガープリントを指定します。SSH サーバーとの接続が確立された後は、その接続を使用して cfftp タグでサポートされているすべてのアクションを実行できます。

セッションの間またはセッション後も接続を開いたままにするには、その接続名を Session スコープまたは Application スコープに挿入します。たとえば、connection="Session.FTPConnection" を指定します。ただし、この方法を使う場合は、すべての FTP オペレーションで完全な変数名を指定し、終了時に close アクションを使用します。接続を開いたままにしておくと、他のユーザーが FTP サーバーを使用できません。このため、接続はできるだけ早く閉じるようにしてください。接続名をセッション変数またはアプリケーション変数に割り当てなかった場合は、接続を現在のページのためだけに開いたままになるので、手動で閉じる必要はなくなります。

キャッシュされている接続を変更するとき (retryCount 値や timeout 値を変更する場合など) には、接続の再確立が必要になることがあります。

例

```
<!--- This example uses symmetric encryption. --->

<!--- Open the secure connection. --->
<cfftp action = "open"
    username = "myusername"
    connection = "My_query"
    password = "mypassword"
    fingerprint = "12:34:56:78:AB:CD:EF:FE:DC:BA:87:65:43:21"
    server = "ftp.tucows.com"
    secure = "yes">
<p>Did it succeed? <cfoutput>#cfftp.succeeded#</cfoutput>
<cfdump var ="#My_query# label="connection">

<!--- Transfer files to the remote server. --->
<cfset absolutePathToLocalFile="C:\one\two\myfile.htm">
    <cfif FileExists(absolutePathToLocalFile)>
        <cfftp action = "putFile"
            connection="My_query"
            localFile="#variables.absolutePathToLocalFile#"
            remoteFile="/home/myname/sftptest/myfile.htm">
    <cfelse>
        <!--- Put error handling code here. --->
    </cfif>
<p>Did it succeed? <cfoutput>#cfftp.succeeded#</cfoutput>

<!--- Close the connection. --->
<cfftp action="close" connection="My_query">
```

例

```
<!--- This example uses asymmetric encryption. --->

<!--- Open the secure connection. --->
<cfftp action = "open"
    username = "myusername"
    connection = "My_query"
    key="C:\mykeys\myprivatekey"
    passphrase = "zHx628Fg"
    fingerprint = "12:34:56:78:AB:CD:EF:FE:DC:BA:87:65:43:21"
    server = "ftp.tucows.com"
    secure = "yes">
<p>Did it succeed? <cfoutput>#cfftp.succeeded#</cfoutput>
<cfdump var ="#My_query# label="connection">

<!--- List files on the remote server. --->
<cftry>
    <!--- List the files in a directory. --->
        <cfftp action = "listDir"
            connection="My_query"
            stopOnError="yes"
            name="ListFiles"
            directory="/">
    <cfcatch>
        <!--- Close the connection. --->
        <cfftp action="close" connection="My_query" stopOnError="no">
    </cfcatch>
</cftry>
```

cfftp: 接続 : ファイルおよびディレクトリのオペレーション

説明

cfftp を使用してファイルおよびディレクトリのオペレーションを行うには、この形式の cfftp タグを使用します。

シンタックス

```
<cfftp
    action = "action"
    ASCIIExtensionList = "extensions"
    connection = "connection name"
    directory = "directory name"
    existing = "file or directory name"
    failIfExists = "yes|no"
    item = "directory or file"
    localFile = "filename"
    name = "query name"
    new = "file or directory name"
    passive = "yes|no"
    password = "password"
    proxyServer = "proxy server"
    remoteFile = "filename"
    result = "result name"
    server = "server"
    timeout = "time-out in seconds"
    transferMode = "ASCII FTP|Binary FTP|Auto FTP"
    username = "name">
```

注意 : このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfhttp、cfldap、cfmail、cfpop

属性

属性	必須 / オプション	デフォルト	説明
action	接続がキャッシュされていない場合は必須		実行する FTP オペレーションです。 • changedir • createDir • listDir • removeDir • getFile • putFile • rename • remove • getCurrentDir • getCurrentURL • existsDir • existsFile • exists
ASCIIExtensionList	オプション	txt;htm;html; cfm;cfml;shtm; shtml;css;asp;asa	transferMode="auto" のときに強制的に ASCII 転送モードを使用するファイル拡張子のリストです。セミコロンで区切って指定します。
connection	action = "open" または "closed" の場合は必須		FTP 接続の名前です。 新規 FTP 接続をキャッシュするか、または既存の接続を再利用する場合に使用します。
directory	action = "changedir"、"createDir"、"listDir"、または "existsDir" の場合は必須		オペレーションの対象となるディレクトリです。
existing	action = "rename" の場合は必須		リモートサーバー上にあるファイルまたはディレクトリの現在の名前です。
failIfExists	オプション	yes	• yes: 同じ名前のローカルファイルが存在する場合、getFile アクションは失敗します。 • no
item	action = "exists" または "remove" の場合は必須		これらのアクションの対象となるファイルまたはディレクトリです。
localFile	action = "getFile" または "putFile" の場合は必須		ローカルファイルシステム上にあるファイルの名前です。詳細については、「使用方法」を参照してください。

属性	必須 / オプション	デフォルト	説明
name	action = "listDir" の場合は必須		ディレクトリリストのクエリー名です。
new	action = "rename" の場合は必須		リモートサーバー上にあるファイルまたはディレクトリの新規名です。
passive	オプション	no	• yes: passive モードを有効にします。 • no
password	action="open" の場合は必須		ユーザーがログインするためのパスワードです。
proxyServer	オプション		文字列。プロキシアクセスを指定した場合に使用するプロキシサーバーの名前です。
remoteFile	action = "getFile"、"putFile"、または "existsFile" の場合は必須		FTP サーバーファイルシステム上にあるファイルの名前です。
result	オプション		cfftp で returnValue 変数を格納する格納先の構造体の名前を指定します。指定すると、その値が接頭辞として cfftp の代わりに使用され、returnVariable にアクセスするときに使われます。詳細については、「使用方法」を参照してください。
server	FTP 接続がキャッシュされていない場合は必須		接続先の FTP サーバーです (例 : ftp.myserver.com)。
timeout	オプション	30 秒	FTP サーバーからの応答を ColdFusion が待機する時間を秒単位で指定します。 キャッシュされた接続の場合は、action = "open" とともに使用されます。
transferMode	オプション	auto	• ASCII FTP 転送モード • Binary FTP 転送モード • Auto FTP 転送モード
username	接続がキャッシュされていない場合は必須		FTP オペレーションで渡すユーザー名です。

使用方法

アクティブな FTP 接続への接続キャッシュを使用する場合は、username、password、または server 接続属性を再指定する必要はありません。

キャッシュされている接続を変更するとき (retryCount 値や timeout 値を変更する場合など) には、接続の再確立が必要になることがあります。

action = "listDir" の場合、attributes 列は directory または normal を返します。その他の hidden や system といったプラットフォーム特有の値は、現在はサポートされていません。

action = "listDir" の場合は、mode 列が返されます。この列には、UNIX アクセス許可を表す 8 進数の文字列表記が含まれています (例 : "777")。

cfftp.returnValue 変数は、次のアクションについての戻り値を返します。

- getCurrentDir
- getCurrentURL

- existsDir
- existsFile
- exists

詳細については、『ColdFusion アプリケーションの開発』を参照してください。

詳しくは、『ColdFusion アプリケーションの開発』の「cfftp タグによるファイルの操作」の節を参照してください。

localFile 属性

次のシンタックスを使用して、ディスクに書き込まれないメモリ内のファイルをローカルファイルとして指定します。メモリ内のファイルを使用すると、一時的データの処理が速くなります。

```
ram:///filepath
```

ファイルパスには、ram:///petStore/images/poodle.jpg のようにディレクトリを含めることができます。ファイルを指定する前に、パスに含まれるディレクトリを作成しておく必要があります。メモリ内のファイルの使用方法について詳しくは、『ColdFusion アプリケーションの開発』の Working with in-memory files を参照してください。

アクションと cfftp.ReturnValue 変数

アクションの結果によって、returnValue 変数の値が次の表のように決まります。

cfftp アクション	cfftp.returnValue の値
getCurrentDir	文字列。現在のディレクトリです。
getCurrentURL	文字列。現在の URL です。
existsDir	yes または no です。
existsFile	yes または no です。
exists	yes または no です。

returnValue 変数にアクセスするには、result 属性で指定される値 (その値が設定されている場合) または cfftp を変数の前に付ける必要があります。result 属性は、複数のページから同時に行われる可能性がある cfftp 呼び出しについて、一方の呼び出しの結果が他方の呼び出しの結果を上書きしないようにするための方法を提供します。result 属性を myResult に設定した場合は、たとえば myResult.returnVariable として returnVariable 変数にアクセスします。あるいは、cfftp.returnVariable としてこの変数にアクセスします。

例

次の例では、接続を開き、ファイル名またはディレクトリ名、パス、URL、長さ、および修正日がリストされたファイルを取得します。

```
 <p>Open a connection
<cfftp connection = "myConnection"
    username = "myUserName"
    password = "myUserName@allaire.com"
    server = "ftp.allaire.com"
    action = "open"
    stopOnError = "Yes">

<p>Did it succeed? <cfoutput>#cfftp.succeeded#</cfoutput>
<cfftp connection = "myConnection"
    action = "LISTDIR"
    stopOnError = "Yes"
    name = "ListDirs"
    directory = "/">

<p>FTP Directory Listing:<br>
<cftable query = "ListDirs" HTMLTable = "Yes" colHeaders = "Yes">
    <cfcol header = "<b>Name</b>" text = "#name#">
    <cfcol header = "<b>Path</b>" text = "#path#">
    <cfcol header = "<b>URL</b>" text = "#url#">
    <cfcol header = "<b>Length</b>" text = "#length#">
    <cfcol header = "<b>LastModified</b>"
     text = "#DateFormat(lastmodified)#">
    <cfcol header = "<b>IsDirectory</b>" text = "#isdirectory#">
</cftable>

<p>Move Image File to Remote Server:<br></p>
<!--- The image will be put into the root directory of the FTP server unless
    otherwise noted, i.e., remoteFile = "somewhere_put.jpg" vs remoteFile = "/support/somewhere_put.jpg"
--->
<cfftp
    connection = "myConnection"
    action = "putFile"
    name = "uploadFile"
    transferMode = "binary"
    localFile = "C:\files\upload\somewhere.jpg"
    remoteFile = "somewhere_put.jpg">
<p>Did it succeed? <cfoutput>#cfftp.succeeded#</cfoutput>

<p>Close the connection:
<cfftp connection = "myConnection"
    action = "close"
    stopOnError = "Yes">
<p>Did it succeed? <cfoutput>#cfftp.succeeded#</cfoutput>
```

cfftp action = "listDir"

説明

クエリーオブジェクトの列にアクセスするには、このタグの action 属性を "listDir" に設定します。

使用方法

このアクションを使用する場合は、name 属性の値を指定します。この値には、クエリーオブジェクトの listDir アクションの結果が保持されます。クエリーオブジェクトを構成する列には、queryname.columnname[row] という形式で参照できます。ここで、queryname は name 属性で指定されているクエリーの名前で、columnname はクエリーオブジェクトで返される列を示します。**row** 値は、listDir オペレーションによって返されるファイルエントリまたはディレクトリエントリの行番号です。エントリごとに 1 つの行が作成されます。

cfftp クエリーオブジェクトの列	説明
Name	現在の要素のファイル名です。
Path	現在の要素のファイルパスです (ドライブ指定なし)。
URL	現在の要素 (ファイルまたはディレクトリ) の完全な URL です。
Length	現在の要素のファイルサイズです。
LastModified	現在の要素の日付時刻値です。形式は指定されません。
Attributes	文字列。現在の要素の属性を表します (normal または Directory)。
IsDirectory	ブール値です。オブジェクトがファイルであるかディレクトリであるかを示します。
Mode	UNIX および Linux だけに適用されます。許可を表します。8 進数値です。

注意：以前にクエリー列の値としてサポートされていた hidden や system などのシステム特有の情報は、現在はサポートされていません。

cffunction

説明

CFML 内で呼び出すことのできる関数を定義します。ColdFusion コンポーネントのメソッドを定義するために必要です。

履歴

ColdFusion 10：restPath、httpMethod、produces、consumes

ColdFusion 8:

- returnformat 属性、secureJSON 属性、および verifyClient 属性が追加されました。
- component が ReturnType 属性の有効な値として追加されました。

ColdFusion MX 7: description 属性が追加されました。returntype 属性に XML 値が追加されました。

ColdFusion MX: このタグが追加されました。

カテゴリ

拡張タグ

シンタックス

```
<cffunction
    name = "method name"
    access = "method access"
    description = "function description"
    displayName = "name"
    hint = "hint text"
    output = "yes|no"
    returnFormat = "not specified|JSON|plain|WDDX"
    returnType = "data type"
    roles = "securityRoles"
    secureJSON = "yes|no"
    verifyClient = "no|yes">
```

関連項目

cfargument、cfcomponent、cfinterface、cfinvoke、cfinvokeargument、cfobject、cfproperty、cfreturn、SerializeJSON

属性

属性	必須 / オプション	デフォルト	説明
name	必須		文字列です。cfcomponent タグ内で使用されるコンポーネントメソッドを指定します。
access	オプション	public	メソッドを呼び出すことができるクライアントセキュリティコンテキストです。 使用できる値は次のとおりです。 • private: メソッドを宣言するコンポーネント、および自分が定義されているコンポーネントを拡張するコンポーネントのみから使用できます。 • package: メソッドを宣言するコンポーネント、コンポーネントを拡張するコンポーネント、またはパッケージ内の他のコンポーネントでのみ使用できます。同じパッケージ内の CFC ページでメソッドを使用できます。 • public: ローカルで実行中のページまたはコンポーネントメソッドから使用できます。 • remote: ローカルまたはリモートで実行中のページまたはコンポーネントメソッドから使用できる他に、URL、Flash、または Web サービスを介してリモートクライアントから使用することができます。関数を Web サービスとしてパブリッシュする場合、このオプションは必須です。
description	オプション		関数を短いテキストで説明します。
displayname	オプション		CFC メソッドのパラメータの場合にのみ意味があります。イントロスペクションを使用して CFC についての情報を示すときに、関数名に続く括弧内に表示される値です。
hint	オプション		CFC メソッドのパラメータの場合にのみ意味があります。イントロスペクションを使用して CFC についての情報を示すときに表示されるテキストです。hint 属性の値は、関数の説明のシンタックス行の次に表示されます。
output	オプション	関数の本文は標準の CFML として処理されます。	関数がどのような条件下で HTML 出力を生成するかを指定します。 使用できる値は次のとおりです。 • yes: 関数の本文全体が cfoutput タグ内にある場合と同様に処理されます。シャープ記号 (#) で囲まれた変数名は、自動的に実際の値に置き換えられます。 • no: 関数が cfsilent タグ内にある場合と同様に処理されます。 この属性を指定しない場合、関数の本文は標準 CFML として処理されます。すべての変数は cfoutput タグ内に置く必要があります。
returnformat		WDDX または XML 形式。「説明」を参照。	リモートの呼び出し元に返される値の形式です。この属性は、ローカルの呼び出し元に返される値の形式には影響しません。 使用できる値は次のとおりです。 • json: リモートに値を返す前に、戻り値を JSON 形式にシリアル化します。 • wddx: リモートに値を返す前に、戻り値を WDDX 形式にシリアル化します。 • plain: 戻り値の型が ColdFusion で文字列に直接変換できる型であることを確認して、シリアル化せずに文字列値を返します。有効な形式には、数値や XML オブジェクトなどのすべての単純型が含まれます。戻り値が配列やバイナリ値などの複合型の場合は、エラーになります。returntype 属性を指定する場合は、その値として any、boolean、date、guid、numeric、string、uuid、variablename、XML のいずれかを指定する必要があります。そうしないと、エラーになります。 デフォルトでは、XML 以外の戻り型 (単純型を含む) はすべて WDDX 形式にシリアル化され、XML データは XML テキストとして返されます。 リモート CFC 関数を呼び出すときに、returnformat を HTTP リクエストパラメータとして使用することもできます。このパラメータには returnformat 属性と同じ効果があり、cffunction タグで指定された returnformat 属性の値よりも優先されます。

属性	必須 / オプション	デフォルト	説明
returnType	Web サービスの場合は必須、その他の場合はオプション	any	タイプ名の文字列です。関数の戻り値のデータ型を指定します。 • any • array • binary • boolean • component: 戻り値は ColdFusion コンポーネントである必要があります。 • date • guid: この引数は **xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx** という形式の UUID または GUID でなければなりません。**x** は 16 進数の 1 文字 (0 ～ 9、A ～ F) を表します。 • numeric • query • string • struct • uuid: この引数は **xxxxxxxx-xxxx-xxxx-xxxxxxxxxxxxxxxx** という形式の ColdFusion UUID でなければなりません。**x** は 16 進数の 1 文字 (0 ～ 9、A ～ F) を表します。 • variableName: ColdFusion 変数のネーミング規則に従った形式の文字列です。 • void: 値を返しません。 • xml: Web サービス関数で CFML XML オブジェクトと XML 文字列を返すことができます。 • コンポーネント名：returntype 属性の値が上記のどれにも当てはまらない場合、ColdFusion はそれを ColdFusion コンポーネントの名前として扱います。関数を実行したときに、返された値が指定の名前を持つ CFC でない場合はエラーになります。 **メモ：**関数が値を返さず、returnType の値が string の場合は、エラーが発生します。他のタイプの場合、エラーは発生しません。
roles	オプション	"" (空)	メソッドを呼び出すことができる ColdFusion セキュリティロールのリストをカンマで区切って指定します。指定のロールでログインしているユーザーだけがこの関数を実行できます。この属性を指定しない場合、すべてのユーザーがこのメソッドを呼び出すことができます。
secureJSON	オプション	「説明」を参照	リモートの呼び出しへの応答として関数が JSON 形式で返す値の前に、セキュリティの接頭辞を追加するかどうかを指定するブール値です。 デフォルト値は、Application.cfc ファイルの This.secureJSON 変数の値、または cfapplication タグの secureJSON 属性の値です。あるいは、secureJSON アプリケーションの設定が存在しない場合は、Administrator の [サーバーの設定]-[設定] ページの [シリアル化 JSON への接頭辞付加] 設定の値 (デフォルトは false) です。 詳細については、『ColdFusion アプリケーションの開発』の Improving security を参照してください。
verifyClient	オプション	no	暗号化されたセキュリティトークンを含めるためにリモートの関数呼び出しが必要かどうかを指定するブール値です。ColdFusion AJAX アプリケーションでのみ使用します。 詳細については、『ColdFusion アプリケーションの開発』の Improving security を参照してください。

属性	必須 / オプション	デフォルト	説明
restPath	オプション		CFC で使用するサブリソースパスを指定します。 パスの大文字と小文字は区別されます。また、パスを指定するときは特殊文字を使用しないことが望ましいです。パス内には正規表現を含めることができます。 指定可能な表現について詳しくは、cfcomponent の restPath 属性を参照してください。
httpMethod			使用する HTTP メソッドです。次のいずれかを指定する必要があります。 • GET：サーバーに情報をリクエストします。リクエストされた情報をサーバーが識別するために必要なデータはすべて、URL 内か cfhttp type="URL" タグ内に含まれている必要があります。 • POST：処理する情報をサーバーに送信します。cfhttpparam タグが必要です。フォームに似たデータを送信するときによく使われます。 • PUT：指定の URL にメッセージ本文を保管するようサーバーにリクエストします。このメソッドは、サーバーにファイルを送信するときに使われます。 • DELETE：指定の URL を削除するようサーバーにリクエストします。 • HEAD：GET メソッドと同じですが、サーバーはレスポンス内にメッセージ本文を送信しません。このメソッドは、ハイパーテキストリンクの有効性やアクセシビリティのテスト、ドキュメントのタイプや修正時刻の確認、またはサーバーのタイプの判別に使用します。HEAD を指定していない場合、デフォルトで GET メソッドが呼び出されます。ただし、レスポンス内にメッセージ本文は送信されません。 • OPTIONS：サーバーまたは指定の URL で使用できる通信オプションについての情報をリクエストします。このメソッドを使用すると、ColdFusion アプリケーションは、追加のサーバー活動を要求しなくても、URL に関連付けられているオプションや要件、あるいはサーバーの機能を判別することができます。OPTIONS を指定していない場合は、ColdFusion によってレスポンスが送信されます。 コンポーネントレベルで指定した値よりも優先されます。
produces	オプション	*/*	受け入れ可能な MIME タイプのカンマ区切りリストです。 リソースが生成可能な MIME メディアタイプの表現を指定するために使用します。例：produces="text/plain,text/html" リソースが複数の MIME メディアタイプを生成可能な場合、選択される関数は、最も受け入れ可能なメディアタイプとしてクライアントで宣言されたメディアタイプに対応します。 値を指定しない場合、デフォルトで「*/*」が設定されます（これは、すべての MIME タイプが使用されることを意味します）。 コンポーネントレベルで指定した値よりも優先されます。
consumes	オプション	*/*	受け入れ可能な MIME タイプのカンマ区切りリストです（例：consumes="text/plain,text/html"）。 リソースが受け入れ可能（つまり使用可能）なクライアントからの MIME メディアタイプの表現を指定するために使用します。 この値は、コンポーネントレベルの同じ属性よりも優先されます。 値を指定しない場合、デフォルトで「*/*」が設定されます（これは、すべての MIME タイプが使用されることを意味します）。 produces および consumes の属性を指定するときは、競合するケースが発生しないようにすることをお勧めします。例えば、関数 1 で produces を text/xml に、consumes を text/* に指定し、関数 2 で produces を text/* に、consumes を text/xml に指定するといったことは避けてください。この場合、accept = text/xml で consumes が text/xml のリクエストに対して、両方の関数が有効になります。

使用方法

cffunction タグを使用すると、ColdFusion のビルトイン関数と同じ方法で呼び出すことのできる関数を定義できます。

ColdFusion コンポーネント (CFC) のメソッドを定義するには、cffunction タグを使用します。

次の例は、ColdFusion Query オブジェクトを返す単純な CFC メソッドを定義するための cffunction タグの属性です。

```
<cffunction
    name="getEmployees"
    access="remote"
    returnType="query"
    hint="This query returns all records in the employee database. It candrill-down or narrow the search,
based on optional input parameters.">
```

ColdFusion コンポーネントに対する cffunction タグの使用方法については、『ColdFusion アプリケーションの開発』の Building and Using ColdFusion Components を参照してください。

returnformat="json" が指定されている場合に関数からクエリーが返されたときは、2 つのエントリ、列名の配列、および列データ配列の配列を含む JSON オブジェクトにクエリーがシリアル化されます。詳細については、SerializeJSON を参照してください。

roles 属性を指定した場合、この関数は、指定したロールのいずれかに所属するユーザーがログインした場合にのみ実行されます。

returnType 属性に variableName を指定した場合、この関数は、ColdFusion の変数ネーミング規則に沿った文字列を返さなければなりません。つまり、文字、アンダースコア (_)、あるいは Unicode 通貨記号で始まり、文字、数字、アンダースコア、ピリオド、Unicode 通貨記号のみから成る文字列を返す必要があります。ColdFusion は、その値が既存の ColdFusion 変数に対応するかどうかをチェックしません。

例

```
<cfcomponent>
    <cffunction name="getEmp">
         <cfquery
                name="empQuery" datasource="ExampleApps" >
                SELECT FIRSTNAME, LASTNAME, EMAIL
                FROM tblEmployees
         </cfquery>
         <cfreturn empQuery>
    </cffunction>
    <cffunction name="getDept">
        <cfquery name="deptQuery" datasource="ExampleApps" >
         SELECT *
         FROM tblDepartments
         </cfquery>
         <cfreturn deptQuery>
    </cffunction>
</cfcomponent>
```

タグ g ～ h

cfgraph

説明

このタグは非推奨となっています。代わりに cfchart、cfchartdata、および cfchartseries タグを使用してください。

データをグラフで表示します。

履歴

ColdFusion MX: このタグは非推奨になりました。このタグは ColdFusion 5 とは異なる動作をします。なお、これ以降のリリースでは機能しない可能性があります。

このタグの ColdFusion MX での実装において、以前のリリースと互換性のないものを次に示します。

cfgraph タグの属性	ColdFusion MX での機能
Title	無視されます。
Titlefont	無視されます。
Barspacing	無視されます。
Bordercolor	ボーダー、グリッド線、およびテキスト表示に使われる色です。
Colorlist	bar、pyramid、area、horizontalbar、cone、cylinder、step、および pie の各チャートのそれぞれのデータポイントに使用する色のリストです。
Valuelabelfont	値ラベルのテキストフォントを設定します。Valuelabelfont、Itemlabelfont、および Legendfont の値が異なる場合は、タグ内に最後に指定した値が使用されます。 Arial はサポートされません。これは Dialog にマッピングされます。
Itemlabelfont	項目ラベルのテキストフォントを設定します。Valuelabelfont、Itemlabelfont、および Legendfont の値が異なる場合は、タグ内に最後に指定した値が使用されます。 Arial はサポートされません。これは Dialog にマッピングされます。
Legendfont	凡例のテキストフォントを設定します。Valuelabelfont、Itemlabelfont、および Legendfont の値が異なる場合は、タグ内に最後に指定した値が使用されます。 Arial はサポートされません。これは Dialog にマッピングされます。
ShowLegend	• above、below、left、right: これらのオプションを指定した場合、凡例が表示されますが、位置には影響ありません。 • none: 凡例が表示されません。
Valuelabelsize	値ラベルのテキストサイズを設定します。Valuelabelsize と Itemlabelsize の値が異なる場合は、タグ内に最後に指定した値が使用されます。
Itemlabelsize	項目ラベルのテキストサイズを設定します。
Itemlabelorientation	無視されます。ColdFusion により、ラベルとグラフのサイズに基づいて最適な向きが計算されます。
Borderwidth	• 0 以外の数の場合 : 指定した数値に関係なく、デフォルト幅のボーダーが表示されます。 • 0: ボーダーは表示されません。
Depth	• 0: グラフに 2 次元効果を付けて表示します。 • その他の値 : グラフに 3 次元効果を付けて表示します。
Linewidth	無視されます。
Showvaluelabel	• yes: マウスクリック時に値を表示します。 • no: 値を表示しません。 • rollover: マウスカーソルが上に置かれたときに値を表示します。
Valuelocation	無視されます。

cfgraph タグの属性	ColdFusion MX での機能
url	グラフ内のいずれかの項目がクリックされたときに開くページの URL です。 URL 内では次の変数を使用することができます。これらの変数は、URL へアクセスする前に実際の値に置き換えられます。 • "$value$": 選択された行または列の値、あるいは空の文字列です。 • "$itemlabel$": 選択された項目 (列) の値、または空の文字列です。 • "$serieslabel$": 選択された系列 (行) の値、または空の文字列です。 • "javascript:...": クライアントサイドのスクリプトを実行します。
Urlcolumn	無視されます。
Type="HorizontalBar"	(0,0) 座標を左下隅に置きます。
ScaleFrom	データ内の最小値が scaleFrom より小さい場合、またはデータ内の最大値が scaleTo より大きい場合に、それぞれのデータ値を Y 軸のスケールの最小値または最大値として使用します。これにより、scaleFrom 値や scaleTo 値に関係なく、すべてのデータ値が表示されます。

cfgraphdata

説明

このタグは非推奨となっています。代わりに cfchart、cfchartdata、および cfchartseries タグを使用してください。

グラフ内にデータポイントを表示します。cfgraph タグ内で使用します。

履歴

ColdFusion MX: このタグは非推奨になりました。このタグは ColdFusion 5 とは異なる動作をします。なお、これ以降のリリースでは機能しない可能性があります。

cfgrid

説明

cfform タグ内で使用します。ColdFusion フォーム内にグリッドコントロール (データのテーブル) を作成します。グリッド列および行のデータを指定するには、cfgridcolumn タグと cfgridrow タグを使用するか、query 属性を使用した上で必要に応じて cfgridcolumn タグを併用します。

先頭にゼロが付く数値データ（例えば、郵便番号の 02674）を返す CFC メソッドでは、returnformat="string" を設定していても、このゼロはバインド式で 8 進数値として解釈され、それに相当する 10 進数値（この例の場合は 1468）として解釈されます。この問題を解決するには、URL バインド、または JavaScript 関数を介して（例えば、cfajaxproxy を使用して）送ったバインドで、returnformat=plain を設定して数値を保持することができます。また、先頭のゼロは、autosuggest コントロールの候補リストからは取り除かれます。

カテゴリ

フォームタグ

シンタックス

```
<cfgrid
    name="name"
    align="value"
    appendKey="yes|no"
    autoWidth="yes|no"
    bgColor="web color"
    bind="bind expression"
    bindOnLoad="yes|no"
    bold="yes|no"
    colHeaderAlign="left|right|center"
    colHeaderBold="yes|no"
    colHeaderFont="font_name"
    colHeaderFontSize="size"
    colHeaderItalic="yes|no"
    colHeaders="yes|no"
    colHeaderTextColor="web color"
    collapsible="false|true"
    delete="yes|no"
    deleteButton="text"
    enabled="yes|no"
    font="column_font"
    fontSize="size"
    format="applet|Flash|html|xml"
    gridDataAlign="left|right|center"
    gridLines="yes|no"
    groupfield="column name"
    height="integer"
    highlightHref="yes|no"
    href="URL"
    hrefKey="column_name"
    hSpace="integer"
    insert="yes|no"
    insertButton="text"
    italic="yes|no"
    maxRows="number"
    multirowselect="yes|no"
    notSupported="text"
    onBlur="ActionScript"
    onChange="ActionScript or bind expression"
    onError="JavaScript function name"
    onFocus="ActionScript function"
    onLoad="JavaScript function name"
    onValidate="JavaScript function name"
    pageSize="number of rows"
    pictureBar="yes|no"
    preservePageOnSort="yes|no"
    query="query name"
    rowHeaderAlign="left|right|center"
    rowHeaderBold="yes|no"
    rowHeaderFont="font name"
    rowHeaderFontSize="size"
    rowHeaderItalic="yes|no"
    rowHeaders="yes|no"
    rowHeaderTextColor="web color"
    rowHeight="pixels"
```

```
    selectColor="web color"
    selectMode="mode"
    selectOnLoad="yes|no"
    sort="yes|no"
    sortAscendingButton="text"
    sortDescendingButton="text"
    stripeRowColor="web color"
    stripeRows="yes|no"
    style= "style specification"
    target="URL_target"
    textColor="web color"
    title="text"
    tooltip="text"
    visible="yes|no"
    vSpace="integer"
    width="integer">

        zero or more cfgridcolumn and cfgridrow tags

</cfgrid>
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfajaximport、cfapplet、cfcalendar、cfgridcolumn、cfgridrow、cfgridupdate、cfform、cfformgroup、cfformitem、cfinput、cfselect、cfslider、cftextarea、cftree、『ColdFusion アプリケーションの開発』の Using HTML grids

履歴

ColdFusion9.0.1：multirowselect 属性が追加されました。HTML グリッドのみでサポートされています。

ColdFusion 9:

- collapsible、groupfield、onLoad、および title 属性が追加されました。HTML グリッドのみでサポートされています。
- HTML グリッドで insert 属性を使用できるようになりました。

ColdFusion 8: HTML 形式のグリッドがサポートされるようになりました (format 属性の html 値と、次の属性など : bind、bindOnLoad、pageSize、preservePageOnSort、stripeRows、stripeRowColor)。

ColdFusion MX 7.01: onBlur イベントと onFocus イベントがサポートされるようになりました。

ColdFusion MX 7:

- format 属性が追加され、Flash 形式と XML 形式の出力がサポートされるようになりました。
- enabled、onChange、style、tooltip、および visible の各属性が追加されました (Flash 形式の場合のみ)。

ColdFusion MX: rowHeaderWidth 属性が変更されました。ColdFusion では、rowHeaderWidth 属性を使用しません。この属性は省略できます。

属性

注意：XML 形式では、ColdFusion はすべての属性を XML に渡します。用意された XSLT スキンでは、XML 形式のグリッドは処理または表示されませんが、アプレット形式および Flash 形式のグリッドは表示されます。

属性	必須 / オプション、形式	デフォルト	説明
name	必須、すべて		グリッドコントロールの名前です。
align	オプション、 アプレット		グリッドセルの内容の整列です。 • Top • Left • Bottom • Baseline • Texttop • Absbottom • Middle • Absmiddle • Right
appendKey	オプション、 HTML、アプレット	yes	• yes: href とともに使用するときは、"CFGRIDKEY=" および選択したアイテムに関する情報を追加します。詳細については、「href 属性の使用」を参照してください。 • no
autoWidth	オプション、 HTML、アプレット	no	• yes: グリッド幅内にすべての列が表示されるように、列幅を設定します。各列の幅が等しいか、関連する cfgridcolumn の width 属性で指定された値に応じた比率の幅になります。水平スクロールバーは使用できません。 • no: 各列を等幅に設定するか、cfgridcolumn の width 属性で指定された値に設定します。
bgColor	オプション、 すべて		コントロールの背景色です。 ほとんどの形式で、16 進数値またはカラー名を指定できます。16 進数の値を入力するには、"##xxxxxx" という形式を使用します。ここで、x は 0 ～ 9 または A ～ F です。シャープ記号 (#) は 2 つ使用するか、または使用しないでください。サポートされる色の名前については、cfchart を参照してください。 • 制約 : HTML 形式の場合は有効な Web 色、Flash 形式の場合は 16 進数の値を指定する必要があります。 • Flash 形式の場合のみ : 反復パターンの行の背景色を指定する場合は、2 つの色をカンマで区切ります。
bind	オプション、 HTML		グリッドのコンテンツを埋め込むためのバインド式です。query 属性と一緒には使用できません。 詳細については、『ColdFusion アプリケーションの開発』の Using Ajax Data and Development Features の Binding data to form fields を参照してください。
bindOnLoad	オプション、 HTML	yes	• yes: 最初にフォームをロードしたときに bind 属性式を実行します。 • no: 最初のバインドイベントが発生するまで bind 属性式を実行しません。 bind 属性がない場合は無視されます。 詳細については、『ColdFusion アプリケーションの開発』の Using the bindOnLoad attribute を参照してください。

属性	必須 / オプション、形式	デフォルト	説明
bold	オプション、すべて	no	• yes: テキストをボールドで表示します。 • no
colHeaderAlign	オプション、アプレット	left	• left: 列ヘッダのテキストを左揃えにします。 • right: 列ヘッダのテキストを右揃えにします。 • center: 列ヘッダのテキストを中央揃えにします。
colHeaderBold	オプション、すべて	no	• yes: 列ヘッダをボールドで表示します。 • no
colHeaderFont	オプション、すべて		列ヘッダのフォントです。
colHeaderFontSize	オプション、すべて		列ヘッダテキストのサイズです (単位 : ポイント)。
colHeaderItalic	オプション、すべて	no	• yes: 列ヘッダをイタリックで表示します。 • no
colHeaders	オプション、アプレット、Flash	yes	• yes: 列のヘッダを表示します。 • no
colHeaderTextColor	オプション、すべて		列ヘッダの色です。 • オプション : textColor 属性の場合と同じです。
collapsible	オプション、HTML	False	ユーザーがタイトルバーの矢印をクリックしてグリッド全体を折り畳むことができるかどうかを指定するブール値です。この属性を指定するとグリッドにタイトルバーが追加されます。
delete	オプション、HTML、アプレット	no	• yes: ユーザーはグリッドから行データを削除できます。selectmode="edit" の場合にのみ有効です。 • no
deleteButton	オプション、HTML、アプレット	Delete	削除ボタンのテキストです。selectmode="edit" の場合にのみ有効です。
enabled	オプション、Flash	yes	Flash 形式の場合のみ : コントロールを有効にするかどうかを指定するブール値です。無効なコントロールはライトグレーで表示されます。
font	オプション、すべて		テキストのフォントです。
fontSize	オプション、すべて		テキストのサイズです (単位 : ポイント)

属性	必須 / オプション、形式	デフォルト	説明
format	オプション、すべて	applet	• applet: Java アプレットを生成します。 • Flash: Flash グリッドコントロールを生成します。 • html: データのバインドをサポートする AJAX ベースの HTML グリッドコントロールを生成します。 • xml: XML 表記のグリッドを生成します。 XML 形式フォームでは、生成された XML がフォーム内に含まれます。 HTML 形式フォームでは、name 属性で指定された名前を持つ文字列変数に XML が格納されます。
gridDataAlign	オプション、アプレット	left	• left: 列内のデータを左揃えにします。 • right: 列内のデータを右揃えにします。 • center: 列内のデータを中央揃えにします。
gridLines	オプション、アプレット、Flash	yes	• yes: 行と列の基準を有効にします。 • no
groupField	オプション、HTML	グループ化しない	グリッド行をグループ化します。この属性で指定した列によって編成します。各グループは折り畳むことができ、ヘッダに列名、グループフィールド値、およびグループ内のエントリ数が表示されます。 このオプションを設定すると、列プルダウンメニューに 2 つのグループ化オプションが表示されます。[グループ] 表示オプションにより、列のグループ化の有効と無効を切り替えます。[フィールド別のグループ] オプションにより、選択した列をグループ化で使用するように設定します。ユーザーは、マウスカーソルを列ヘッダ上に置いて表示された下矢印をクリックすることによって、プルダウンメニューを表示します。 この属性は、スタティックなグリッドでのみ使用できます。バインド式を使ってデータを取得するダイナミックグリッドでは使用しないでください。
height	オプション、すべて	300 (アプレットの場合のみ)	コントロールの高さです (単位 : ピクセル)。 Flash 形式でこの属性を省略した場合、グリッドでは自動的にサイズ設定されます。
highlightHref	オプション、アプレット	yes	• yes: href 属性の値に関連付けられたリンクを強調表示します。 • no
href	オプション、HTML、アプレット		各グリッドセルとハイパーリンクさせる URL またはその URL を含むクエリー列の名前です。
hrefKey	オプション、HTML、アプレット		appendKey="True" の場合に、各セルの href URL に追加された値のために使用するクエリー列です。cfgridcolumn タグを使用する場合、この列はこれらのタグのいずれかで指定する必要があります。
hSpace	オプション、アプレット		コントロールの左右に確保する水平方向の間隔です (単位 : ピクセル)。
insert	オプション、アプレット、HTML	no	• yes: ユーザーはグリッドに行データを挿入できます。selectmode="edit" の場合にのみ有効です。 • no

属性	必須 / オプション、形式	デフォルト	説明
insertButton	オプション、アプレット	Insert	挿入ボタンのテキストです。selectmode="edit" の場合にのみ有効です。
italic	オプション、すべて	no	• yes: テキストをイタリックで表示します。 • no
maxRows	オプション、すべて		グリッド内に表示する行の最大数です。
multirowselect	オプション、HTML	no	複数行の選択を許可します。特にこれは、バッチ処理が必要な場合 (複数のレコードを一度に移動する場合など) に便利です。 yes の場合、グリッドの最初の列にチェックボックスが表示され、複数のレコードを選択できます。また、[すべてを選択] と [すべてを選択解除] のオプションも表示されます。 **注意：** multirowselect="yes" の場合は行データが構造体の配列で送信されるのに対し、mutirowselect="no" の場合は構造体で送信されます。また、グリッドデータがユーザーによって操作される場合 (たとえば JavaScript を使用して、ボタンがクリックされたときにレコードを移動する場合)、method に POST を設定します。GET メソッドでは送信可能なデータ量に制限がかかるので、この設定が必要です。
notSupported	オプション、アプレット	「説明」を参照	ブラウザが Java をサポートしない場合やブラウザの Java サポートが無効になっている場合に表示するテキストです。 デフォルトのテキストは、"ColdFusion Java アプレットを表示するには、ブラウザが Java に対応していなければなりません。" です。
onBlur	オプション、Flash		グリッドがフォーカスを失ったときに実行される ActionScript です。
onChange	オプション、HTML、Flash、		**Flash 形式 :** コントロール内のユーザーのアクションに応じてコントロールに変化があったときに実行される ActionScript です。 **HTML 形式 :**bind 属性を指定して edit の値を selectMode に設定した HTML 形式のグリッドでは必須になります。データソースを更新する CFC メソッド、JavaScript 関数、または URL を呼び出すバインド式です。 URL を呼び出す場合、データは JSON 形式で URL ページに渡されるため、DeserializeJSON 関数を使用します。 JavaScript バインドを使用して cfgridrow 引数と cfgridchanged 引数を URL に渡す場合、これらの引数を JSON 文字列にシリアル化する必要があります。
onError	オプション、HTML、アプレット		HTML 形式のグリッドの場合 : エラーが発生したときに実行する JavaScript 関数の名前です。 アプレット形式のグリッドの場合 : 検証が失敗したときに実行する JavaScript 関数の名前です。
onFocus	オプション、Flash		グリッドがフォーカスを取得したときに実行される ActionScript です。
onLoad	オプション		グリッドがロードされてレンダリングされるときに実行するカスタム JavaScript 関数です。
onValidate	オプション、アプレット		ユーザー入力を検証する JavaScript 関数です。フォームオブジェクト、入力オブジェクト、および入力オブジェクト値が、関数に渡されます。検証に成功すると true が返されます。検証に失敗すると false が返されます。

属性	必須 / オプション、形式	デフォルト	説明
pageSize	オプション、HTML	10	ダイナミックグリッドの各ページに表示する行の最大数です。グリッドに表示する行数がページサイズを超えた場合は、指定した数の行のみが 1 ページに表示されます。その場合、ユーザーはすべてのデータを表示するために、ページ間を移動する必要があります。グリッドはデータを表示する必要がある場合にのみ、各ページごとにデータを取得します。 query 属性が指定されている場合、この属性は無視されます。
pictureBar	オプション、アプレット	no	• yes: イメージ (テキストなし) を、挿入、削除、ソートの各ボタンに配置します。 • no: テキスト (イメージなし) を、挿入、削除、ソートの各ボタンに配置します。
preservePageOnSort	オプション、HTML	no	グリッドのソート (または再ソート) を実行した後に、現在と同じ番号のページを表示するのか、1 ページ目を表示するのかを指定します。この属性が yes の場合は、グリッドのソート時に現在の選択ページが保持されます。
query	オプション、すべて		コントロールに関連付けるクエリーの名前です。bind 属性とともに使用することはできません。
rowHeaderAlign	オプション、アプレット	left	• left: 行ヘッダのテキストを左揃えにします。 • right: 行ヘッダのテキストを右揃えにします。 • center: 行ヘッダのテキストを中央揃えにします。
rowHeaderBold	オプション、アプレット	no	• yes: 行ラベルのテキストをボールドで表示します。 • no
rowHeaderFont	オプション、アプレット		行ラベルのフォントです。
rowHeaderFontSize	オプション、アプレット		行ラベルのテキストサイズです (単位 : ポイント)
rowHeaderItalic	オプション、アプレット	no	• yes: 行ラベルのテキストをイタリックで表示します。 • no
rowHeaders	オプション、アプレット	yes	• yes: 数字の行ラベルを示す列を表示します。 • no
rowHeaderTextColor	オプション、アプレット	black	グリッドコントロールの行ヘッダのテキストの色です。 • オプション : textColor 属性の場合と同じです。
rowHeight	オプション、アプレット、Flash、XML		行の最大高さです (単位はピクセル)。cfgridcolumntype="Image" とともに使用します。行内でグラフィックを表示するためのスペースを定義します。
selectColor	オプション、すべて		選択されている項目の背景色です。 • オプション : textColor 属性の場合と同じです。

属性	必須 / オプション、形式	デフォルト	説明
selectMode	オプション、すべて	アプレット形式の場合 : Browse HTML、Flash 形式の場合 : Row	コントロール内での項目の選択モードです。 • Edit: ユーザーがグリッドデータを編集できます。セルを選択するとセルを編集できます。 • Row: ユーザーの選択範囲が、選択したセルを含む行に自動的に拡大されます。 次のものはアプレット形式でのみ使用されます。HTML および Flash 形式では、これらは Row として解釈されます。 • Single: ユーザーの選択範囲が、選択したセルに制限されます。 • Column: ユーザーの選択範囲が、選択したセルを含む列に自動的に拡大されます。 • Browse: ユーザーはグリッドデータのブラウズのみを行うことができます。
selectOnLoad	オプション、HTML	yes	• yes: グリッドのロード時にグリッドの最初の行を選択します。 • no: グリッドのロード時に行を選択しません。
sort	オプション、アプレット	no	ソートボタンを追加して、ユーザーが選択した列で簡単なテキストソートを行います。 • yes: ソートボタンをグリッドコントロール上に配置します。 • no この設定にかかわらず、ユーザーは列ヘッダをクリックして列をソートすることができます。selectMode="browse" の場合、表はソートできません。
sortAscendingButton	オプション、アプレット	A > Z	ソートボタンのテキストです。
sortDescendingButton	オプション、アプレット	Z > A	ソートボタンのテキストです。
stripeRowColor	オプション、HTML		1 行おきに色を付ける場合の色を指定します。この色を付けない行の色は bgColor の設定によって決定されます。
stripeRows	オプション、HTML	no	1 行おきに色を付けるかどうかを示すブール値です。
style	オプション、Flash		CSS 形式のスタイル指定でなければなりません。type="text" の場合は無視されます。
target	オプション、HTML、アプレット		href URL を表示するターゲットフレームまたはウィンドウです。たとえば、"_blank" です。
textColor	オプション Flash、アプレット		テキストの色です。16 進数値またはカラー名で指定します。 16 進数の値を入力するには、"##xxxxxx" という形式を使用します。ここで、x は 0 ～ 9 または A ～ F です。シャープ記号 (#) は 2 つ使用するか、または使用しないでください。 サポートされる色の名前については、cfchart を参照してください。
title	オプション、HTML		グリッドの上部にタイトルとして表示されるテキストです。この属性を指定するとグリッドにタイトルバーが追加されます。
tooltip	オプション、Flash		Flash 形式の場合のみ : マウスポインタをコントロールの上に置いたときに表示されるテキストです。

属性	必須 / オプション、形式	デフォルト	説明
visible	オプション、Flash	yes	Flash 形式の場合のみ : コントロールを表示するかどうかを指定するブール値です。表示されないコントロールが使用するスペースは空白です。
vSpace	オプション、アプレット		コントロールの上下に確保する垂直方向の間隔です (単位 : ピクセル)。
width	オプション、すべて	300 (アプレットの場合のみ)	コントロールの幅。 Flash およびアプレット形式の場合は、ピクセル数で指定する必要があります。HTML 形式の場合は、有効なすべての CSS 寸法単位で指定でき、数値のみの値はピクセル数を示します。 Flash 形式または HTML 形式でこの属性を省略した場合、グリッドは自動的にサイズ設定されます。

使用方法

次の段落の大半は、すべてまたは 2 つ以上のグリッド形式に適用されるグリッド機能について説明しています。Flash フォームに固有の情報については、『ColdFusion アプリケーションの開発』の Creating Forms in Flash を参照してください。HTML 形式のグリッドに固有の情報については、『ColdFusion アプリケーションの開発』の Using HTML grids を参照してください。

このタグは、cfform タグブロックの内部で使用する必要があります。

アプレット形式のグリッドでは、クライアントが Java アプレットをダウンロードする必要があります。また、クライアントに最新の Java プラグインがインストールされていない場合、アプレット形式のグリッドを表示するために最新の Java プラグインをダウンロードしなければならないこともあります。Flash 形式のグリッドでは Flash コントロールを使用します。このツリーは、HTML 形式の cfform タグに埋め込むことができます。このタグを Flash 形式またはアプレット形式のいずれかで正しく動作させるには、JavaScript 対応のブラウザを使用する必要があります。

注意 : HTML 形式フォームでこのタグを使用して Flash 形式を指定し、height 属性と width 属性を指定しない場合、Flash 形式での表示は画面上の表示可能領域を超えるサイズになります。グリッドの後に他の出力 (フォームコントロールなど) が続く場合、それを表示するにはスクロールする必要があります。したがって、HTML フォームの Flash グリッドの後に他の出力を続ける場合は、height と width の値を指定してください。

cfgrid には、cfquery からのデータを挿入できます。cfgrid 本文で cfgridcolumn タグを何も指定しない場合、ColdFusion は次のものを備えたグリッドを生成します。

- クエリーの各列用の列。
- テーブルの列名にあるハイフンやアンダースコア記号がスペースに置き換えられて作成された、各列のデフォルトのヘッダ。先頭の文字とスペースの後ろの文字は大文字に変更されます。他の文字はすべて小文字です。

このタグには終了タグが必要です。

注意 : グリッドセルの編集中に送信ボタンをクリックすると、セルの変更内容が失われることがあります。変更内容を正しく送信するには、セル内のデータを更新した後に、別のセルをクリックしてからフォームを送信することをお勧めします。

cfgrid データをアクションページに返す方法

次の情報は、すべての cfgrid 形式に当てはまります。また、HTML 形式のグリッドは、バインド式を使用してデータをダイナミックに取得できます。詳細については、『ColdFusion アプリケーションの開発』の Using HTML grids を参照してください。

ユーザーがフォームを送信すると、cfgrid タグはフォームのアクションページに送信されるデータにフォーム変数を設定することによって、ユーザーのアクションに関する情報を送信します。このデータはタグの SelectMode 属性値によって異なるため、返されるフォーム変数もこの属性の値によって異なります。

一般に、返されるデータは次のいずれかのカテゴリに当てはまります。

- 単純な選択オペレーションから返される簡単なデータ
- 挿入、更新、および削除オペレーションから返される複雑なデータ

簡単な選択データ (SelectMode 属性が Single、Column、または Row の場合)
フォーム変数が cfform のアクションページに返すデータには、ユーザーが選択したセルについての情報が含まれます。このデータは、一般に、Form スコープ内の ColdFusion 変数という形でアクションページ内で使用できます。この変数には form.#GridName#.#ColumnName# というネーミング規則が適用されます。

SelectMode 属性の値に応じて、次のフォーム変数が返されます。

```
SelectMode="single"
form.#GridName#.#ColumnName# = "SelectedCellValue"
SelectMode="column"
form.#GridName#.#ColumnName# = "ValueOfCellRow1,
ValueOfCellRow2, ValueOfCellRowN"
SelectMode="row"
form.#GridName#.#Column1Name# = "ValueOfCellInSelectedRow"
form.#GridName#.#Column2Name# = "ValueOfCellInSelectedRow"
form.#GridName#.#ColumnNName# = "ValueOfCellInSelectedRow"
```

複雑な更新データ (SelectMode 属性が Edit の場合)
ユーザーがグリッドに行った挿入、更新、または削除の操作をアクションページに通知するために、グリッドからは大量のデータが返されます。ほとんどの場合、cfgridupdate タグを使用してフォーム変数からデータを自動的に収集できます。このタグはデータの収集、SQL 呼び出しの記述、およびデータソースの更新を行います。

cfgridupdate を使用できない場合は (たとえば、返されたデータを複数のデータソースに分散する必要がある場合など)、フォーム変数を読み取るコードを記述します。このモードでは、ColdFusion により各 cfgrid に対して、次の Form スコープの配列変数が作成されます。

```
form.#GridName#.#ColumnName#
form.#GridName#.original.#ColumnName#
form.#GridName#.RowStatus.Action
```

更新、挿入、または削除情報が含まれているテーブルの各行は、それぞれの配列に並行したエントリを持つことになります。変更情報をすべて確認するには、次のようにして配列を反復処理します。送信された cfform 上で cfgrid とともに機能するようにするには、GridName 変数をそのグリッドの名前に設定し、ColNameList をグリッド列のリストに設定します。

```
<cfloop index="ColName" list="#ColNameList#">
    <cfif IsDefined("form.#GridName#.#ColName#")>
        <cfoutput><br>form.#GridName#.#ColName#:<br></cfoutput>

        <cfset Array_New = form[#GridName#][#ColName#]>
        <cfset Array_Orig = form[#GridName#]['original'][#ColName#]>
        <cfset Array_Action = form[#GridName#]RowStatus.Action>

        <cfif NOT IsArray(Array_New)>
            <b>The form variable is not an array!</b><br>
        <cfelse>
            <cfset size = ArrayLen(Array_New)>
            <cfoutput>
            Result Array Size is #size#.<br>
            Contents:<br>
            </cfoutput>

            <cfif size IS 0>
                <b>The array is empty.</b><br>
            <cfelse>
                <table BORDER="yes">
```

```
                    <tr>
                        <th>Loop Index</TH>
                        <th>Action</TH>
                        <th>Old Value</TH>
                        <th>New Value</TH>
                    </tr>
                    <cfloop index="LoopCount" from="1" to=#size#>
                    <cfset Val_Orig = Array_Orig[#LoopCount#]>
                    <cfset Val_New = Array_New[#LoopCount#]>
                    <cfset Val_Action = Array_Action[#LoopCount#]>
                    <cfoutput>
                    <tr>
                        <td>#LoopCount#</td>
                        <td>#Val_Action#</td>
                        <td>#Val_Orig#</td>
                        <td>#Val_New#</td>
                    </tr>
                    </cfoutput>
                    </cfloop>
                </table>
            </cfif>
        </cfif>

    <cfelse>
        <cfoutput>form.#GridName#.#ColName#: NotSet!</cfoutput><br>
    </cfif>
</cfloop>
```

href 属性の使用

href 属性を使用してグリッド項目付きの URL を指定するときには、追加するキー値を 1 つのグリッド項目に制限するか、1 つのグリッド列またはグリッド行に拡大するかが selectMode 属性の値によって決まります。リンクされているグリッド項目をユーザーがクリックすると、cfgridkey 変数が次の形式で URL に追加されます。

```
http://myserver.com?cfgridkey=selection
```

appendKey 属性を no に設定すると、グリッド値は URL に追加されません。

selection の値は、selectMode および属性の値によって決まります。

- hrefKey 属性を指定した場合、**selection** はこの属性で指定した列のフィールド値となります。それ以外の場合は、次のいずれかです。
- selectMode="Single" の場合、**selection** はクリックした列の値となります。
- selectMode="Row" の場合、**selection** は、クリックした行に含まれる列の値をカンマで区切ったリストとなります (行内の最初のセルから順に含まれます)。
- selectMode="Column" の場合、**selection** は、クリックした列に含まれる行の値をカンマで区切ったリストとなります (列内の最初のセルから順に含まれます)。

href 属性を使用する場合は、target 属性も指定できます。target 属性では、標準の HTML ターゲット指定子 (_blank、_parent、_self、_top)、または特定のフレーム名を指定できます。

ColdFusion 9.0.1 で行われた機能強化

- ColdFusion 9 では、ダイナミックグリッドを使用したフォームの場合、フォーム送信時に最初の行のデータを使用できます。ColdFusion 9.0.1 では、このデータは使用できません。
- type が Boolean で selectmode が browse（つまり select=false）の場合、列はチェックボックスで表示されますが、クリックしても反応しません。

例

次の例では、cfdocexamples データベース内の CourseList テーブルにある利用可能なコースのセットを表示する Flash フォームを作成します。cfgrid タグを使用するさらに複雑な例については、cfgridcolumn、cfgridrow、および cfgridupdate を参照してください。

```
<!--- Query the database to fill up the grid. --->
<cfquery name = "GetCourses" dataSource = "cfdocexamples">
    SELECT Course_ID, Dept_ID, CorNumber,
    CorName, CorLevel
    FROM CourseList
    ORDER by Dept_ID ASC, CorNumber ASC
</cfquery>

<h3>cfgrid Example</h3>
<i>Currently available courses</i>
<!--- cfgrid must be inside a cfform tag. --->
<cfform>
    <cfgrid name = "FirstGrid" format="Flash"
        height="320" width="580"
        font="Tahoma" fontsize="12"
        query = "GetCourses">
    </cfgrid>
</cfform>
```

cfgridcolumn

説明

cfform タグ内で cfgrid タグとともに使用します。列を形式設定し、必要に応じてクエリーから列を挿入します。cfgridcolumn 内で使用するフォントと配置に関する属性は、cfgrid 内で定義されているグローバルなフォント設定や配置設定よりも優先されます。

カテゴリ

フォームタグ

シンタックス

```
<cfgridcolumn
    autoExpand = "yes|no"
    name = "column name"
    bgColor = "web color|expression"
    bold = "yes|no"
    dataAlign = "left|right|center"
    display = "yes|no"
    font = "column font"
    fontSize = "size"
    header = "header"
    headerAlign = "left|right|center"
    headerBold = "yes|no"
    headerFont = "font name"
    headerFontSize = "size"
    headerIcon = "icon path"
    headerItalic = "yes|no"
    headerMenu = "yes|no"
    headerTextColor = "web color"
    href = "URL"
    hrefKey = "column name"
    italic = "yes|no"
    mask= "format mask"
    numberFormat = "format"
    select = "yes|no"
    target = "URL target"
    textColor = "web color|expression"
    type = "type"
    values = "comma-separated strings and/or numeric range"
    valuesDelimiter = "delimiter character"
    valuesDisplay = "comma-separated strings and/or numeric range"
    width = "column width">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfgrid、cfgridrow、cfgridupdate、cfform、cfapplet、cfinput、cfselect、cfslider、cftextarea、cftree

履歴

ColdFusion 9.0.1：autoExpand 属性と headerMenu 属性が追加されました。HTML グリッドのみでサポートされています。

ColdFusion 9: boolean、date、numeric、および string_noCase が type 属性値に追加されました。HTML グリッドでサポートされています。

ColdFusion MX 7: mask 属性、および通貨の type 属性値が追加されました。

ColdFusion MX: select="no" の場合の動作が変更されました。cfgrid selectmode 属性の値に関係なく、ユーザーはセルデータを選択および編集できません。セルをクリックすると、セルのボーダー (selectColor の値によってはセルの背景も) の色が変わりますが、セルのデータを編集することはできません。

属性

注意：XML 形式では、ColdFusion はすべての属性を XML に渡します。用意された XSLT スキンでは、XML 形式のグリッドは処理または表示されませんが、アプレット形式および Flash 形式のグリッドは表示されます。

属性	必須 / オプション、形式	デフォルト	説明
autoExpand	オプション、HTML	最初の列は yes、残りの列は no	特定の列につき、指定した列を展開できます。 複数の列で autoExpand="yes" を設定すると、エラーが発生します。また、display 属性を no に設定する場合、autoExpand は yes に設定できません。設定した場合はエラーが発生します。
name	必須、 すべて		グリッド列要素の名前です。グリッドでクエリーを使用する場合、この属性は、グリッド列に挿入するクエリー列の名前でなければなりません。
bgColor	オプション、 すべて		グリッド列の背景の色です。 • オプション : textColor 属性の場合と同じです。
bold	オプション、 すべて	cfgrid で指定した値	• yes: グリッドコントロールのテキストをボールドで表示します。 • no
dataAlign	オプション、 アプレット、Flash、HTML	cfgrid で指定した値	列データの配置です。 • left • right • center
display	オプション、 すべて	yes	• yes • no: 列を非表示にします。
font	オプション、 すべて	cfgrid で指定した値	列内のデータのフォントです。
fontSize	オプション、 すべて	cfgrid で指定した値	列内のテキストのサイズです。
header	オプション、 すべて	yes	列ヘッダのテキストです。cfgrid の colHeaders 属性が yes の場合にのみ使用します。デフォルト値は yes です。
headerAlign	オプション、 アプレット	cfgrid で指定した値	列ヘッダのテキストの配置です。 • left • right • center
headerBold	オプション、 HTML、アプレット	cfgrid で指定した値	• yes: ヘッダをボールドで表示します。 • no
headerFont	オプション、 HTML、アプレット	cfgrid で指定した値	列ヘッダのフォントです。
headerFontSize	オプション、 HTML、アプレット	cfgrid で指定した値	列ヘッダのテキストのサイズです (単位 : ピクセル)
headerIcon	オプション		グリッドのヘッダ列のアイコンとして使用するイメージファイルの場所です。

属性	必須 / オプション、形式	デフォルト	説明
headerMenu ColdFusion 9.0.1 で追加	オプション、HTML	no	グリッド列のヘッダーメニューのオン／オフを切り替えることができます。 ヘッダメニューとは、マウスポインタが上に置かれたときにグリッドのヘッダ列に表示されるドロップダウンリストです。この属性は、グリッドのヘッダにイメージを使用する場合に便利です。
headerItalic	オプション、HTML、アプレット	cfgrid で指定した値	• yes: 列ヘッダをイタリックで表示します。 • no
headerTextColor	オプション、HTML、アプレット		グリッドコントロールの列ヘッダのテキストの色です。 • オプション : textColor 属性の場合と同じです。
href	オプション、HTML、アプレット		URL、または各グリッド列のハイパーリンク先 URL を含んでいるクエリー列の名前です。
hrefKey	オプション、HTML、アプレット		列の値の代わりに、各列の href URL に追加される値として使用するクエリー列です。
italic	オプション、すべて	cfgrid で指定した値	• yes: グリッドコントロールのテキストをイタリックで表示します。 • no
mask	オプション、Flash、HTML		フォームに表示される文字パターン、またはユーザーが入力して ColdFusion に送信できる文字パターンを制御するマスクパターンです。 通貨の type 属性がある列の場合、mask は通貨記号を指定します。数値の前に通貨記号が自動的に挿入されます。 テキスト値や数値がある列の場合、mask は表示する形式またはユーザーが入力できる形式を、次のように指定します。 • A = [A-Za-z] • X = [A-Za-z0-9] • 9 = [0-9] • ? = 任意の文字 • これら以外のすべての文字 = リテラル文字を挿入 列の値が日付またはタイムスタンプの場合、ColdFusion はこのマスクパターンを使用して、選択された日付を形式設定します。 日付時刻のマスク形式の詳細については、「mask 属性の日付時刻形式」を参照してください。 マスキングでは、HTML グリッドがサポートされます。デフォルトの形式は m/d/y です（例：05/06/75）。m は先頭に 0 が付いた月、d は先頭に 0 が付いた日、y は 2 桁表示の年です。詳細については、次の URL を参照してください。 http://www.extjs.com/deploy/dev/docs/output/Date.html
numberFormat	オプション、アプレット		グリッド内で数値データを表示するための形式です。後述の numberFormat 属性のマスク文字の表を参照してください。

属性	必須 / オプション、形式	デフォルト	説明
select	オプション、 すべて	yes	cfgrid の selectmode 属性の値が column、edit、または single の場合の選択動作を指定します。row や browse の場合は無視されます。 • yes: ユーザーは selectmode 属性での指定に応じて、列を選択するか、列内のセルを選択または編集することができます。 • no: ユーザーは列を選択することも、列内のセルを選択または編集することもできません。
target	オプション、 HTML、アプレット		href で指定されたリンクを開くためのフレームまたは標準の HTML ターゲットです。
textColor	オプション、 アプレット、Flash、HTML		列内のグリッド要素のテキストの色です。16 進数の値またはテキスト名で指定します。 16 進数の値を入力するには、"##**xxxxxx**" という形式を使用します。ここで、x は 0 ～ 9 または A ～ F です。シャープ記号は、2 つ使用するか、または使用しません。 制約 : HTML 形式の場合は、有効な HTML カラーを指定する必要があります。アプレット形式の場合は、次のいずれかを指定します。 • 16 進数形式の任意の色 • Black • Red • Blue • Magenta • Cyan • Orange • Darkgray • Pink • Gray • White • Lightgray • Yellow

属性	必須 / オプション、形式	デフォルト	説明
type	オプション、 すべて		すべての形式で次の値を指定することができます。 • boolean: 列をチェックボックスとして表示します。セルが編集可能である場合、ユーザーはチェックマークを変更できます。onchange イベントのスタティックなグリッドとダイナミックなグリッドでは、渡されるデータは、データベースでのブール値の表示形式に変換されます。 • combobox: values 属性と valuedisplay 属性で指定された値をオプションとして含むドロップダウンリストを表示します。 • numeric: グリッドデータを数値でソートできます。HTML 形式では、セルが編集可能である場合、ユーザーは数値を入力できます。 • string_noCase: 大文字小文字を区別しないテキストとしてグリッドデータをソートできます。HTML 形式では、セルが編集可能である場合、ユーザーはテキスト値を入力できます。 次の値はアプレット形式と Flash 形式では指定できますが、HTML 形式では機能しません。 • image: グリッドには、列内の URL で指定されたイメージが表示されます。相対 URL を使用する場合、イメージは CFIDE¥classes ディレクトリまたはサブディレクトリに置かれている必要があります。イメージが列セルよりも大きい場合は、セル内に収まるように切り取られます。Flash のイメージは JPEG ファイルでなければなりません。アプレット形式のイメージは JPEG ファイルか GIF ファイルです。 次の値はアプレット形式では指定できますが、Flash 形式と HTML 形式では機能しません。 • image: 列の値では、イメージファイルへのパスを使用する以外に、ColdFusion のビルトインイメージ名 (cd、computer、document、element、folder、floppy、fixed、remote) を使用することもできます。 次の値は Flash 形式では指定できますが、アプレット形式と HTML 形式では機能しません。 • currency: 列データを通貨として形式設定します。小数点を基準に整列します。この列を使用するグリッドをソートすると、通貨について正しくソートされます。mask 属性を使用して通貨記号を指定します。デフォルト値はドル記号 ($) です。 次の値は HTML 形式では指定できますが、アプレット形式と Flash 形式では機能しません。 • date: 列には日付値が入ります。グリッドの selectMode 属性の値が edit である場合、セルは編集可能です。編集可能なセルをクリックすると、アイコンが表示されます。このアイコンをクリックすると、日付ピッカーが開き、日付を選択できます。
values	オプション、 HTML、アプレット		列内のセルをドロップダウンリストボックスの形式にします。ドロップダウンリストの項目を指定します。例 : `values = "arthur, scott, charles, 1-20, mabel"`

属性	必須 / オプション、形式	デフォルト	説明
valuesDelimiter	オプション、 HTML、アプレット	, (カンマ)	values 属性と valuesDisplay 属性の区切り文字です。
valuesDisplay	オプション、 HTML、アプレット		values 属性内の要素を、ドロップダウンリストで表示する文字列にマッピングします。文字列や数値の範囲を区切り文字で区切って指定します。
width	オプション、 すべて	列ヘッダの幅	列の幅です (単位 : ピクセル)。

次のマトリックスは、type="boolean" の動作を示しています。

変換前	変換後
Y	N
T	F
1	0
true (スタティックなグリッドの場合)	false (スタティックなグリッドの場合)
true (ダイナミックなグリッドの場合)	NO (ダイナミックなグリッドの場合)
ブール値以外または null の場合	Y

アプレット形式の場合に限り、次の numberFormat 属性のマスク文字を使用して、出力を米国式の数値形式や通貨形式に設定することができます。これらのマスク文字の使い方の詳細については、1158 ページの「NumberFormat」を参照してください。cfgridcolumn タグでは、国際数値形式はサポートされません。

文字	説明
_	(アンダースコア) 桁プレースホルダーです。
9	桁プレースホルダーです。
.	(ピリオド) 必須の小数点の位置です。
0	必須の小数点の左または右に置かれ、0 (ゼロ) が埋め込まれます。
( )	数値が 0 より小さい場合は、マスクを丸括弧で囲みます。
+	正の数値の前にプラス記号を置き、負の数値の前にマイナス記号を置きます。
-	正の数値の前にスペースを置き、負の数値の前にマイナス記号を置きます。
,	(カンマ) 3 桁ごとにカンマで区切ります。
L、C	マスク列の幅の中で数値を左揃えまたは中央揃えにします。L または C はマスクの先頭に指定する必要があります。デフォルトは右揃えです。
$	形式設定された数値の先頭にドル記号を置きます。マスクの先頭に指定します。
^	(キャレット) 右側の形式と左側の形式を区切ります。

mask 属性の日付時刻形式

日付時刻値は、デフォルトでは Flash で Oct 29 2004 11:03:21 のような表示形式を使ってグリッド列に表示されます。mask 属性を使用すると、日付または時刻を別の形式で表示できます。それらの形式について次の表で説明します。

パターン文字	説明
Y	年。パターン文字の数が 2 の場合、年は 2 桁に切り詰められます。それ以外の場合、4 桁で表示されます。次の例の 3 番目に示すように、指定された桁数になるように 0 が追加されます。 例： YY = 03 YYYY = 2003 YYYYY = 02003
M	月。形式は、次の条件により決まります。 • パターン文字の数が 1 つの場合、形式は 1 桁または 2 桁の数値として解釈されます。 • パターン文字の数が 2 つの場合、形式は 2 桁の数値として解釈されます。 • パターン文字の数が 3 つの場合、形式は省略したテキストとして解釈されます。 • パターン文字の数が 4 つの場合、形式は省略されないテキストとして解釈されます。 例： M = 7 MM= 07 MMM=Jul MMMM= July
D	日。 例： D=4 DD=04 DD=10
E	曜日。形式は、次の条件により決まります。 • パターン文字の数が 1 つの場合、形式は 1 桁または 2 桁の数値として解釈されます。 • パターン文字の数が 2 つの場合、形式は 2 桁の数値として解釈されます。 • パターン文字の数が 3 つの場合、形式は省略したテキストとして解釈されます。 • パターン文字の数が 4 つの場合、形式は省略されないテキストとして解釈されます。 例： E = 1 EE = 01 EEE = Mon EEEE = Monday
A	AM/PM を示すインジケータ。
J	24 時間形式の時間 (0 ～ 23)。
H	24 時間形式の時間 (1 ～ 24)。
K	am/pm 形式の時間 (0 ～ 11)。
L	am/pm 形式の時間 (1 ～ 12)。

パターン文字	説明
N	分。 例： N = 3 NN = 03
S	秒。
その他のテキスト	その他のテキストをパターン文字列に追加して、文字列をさらに設定することができます。句読点や数字、任意の小文字を使用できます。パターン文字として解釈される可能性があるので、大文字は使用しないでください。 例： EEEE, MMM.D, YYYY at H:NN A = Tuesday, Sept. 8, 2003 at 1:26 PM

例

次の例では、cfdocexamples データベースにある CourseList テーブルの特定のフィールドを更新します。cfgridcolumn タグを使用して、テーブルを構築します。

```
<!--- If the gridEntered field exists, the form has been submitted.
        Update the database. --->
<cfif IsDefined("form.gridEntered")>
    <cfgridupdate grid = "FirstGrid" dataSource = "cfdocexamples"
        tableName = "CourseList" keyOnly = "Yes">
</cfif>

<!--- Query the database to fill up the grid. --->
<cfquery name = "GetCourses" dataSource = "cfdocexamples">
    SELECT Course_ID, Dept_ID, CorNumber,CorName, CorLevel, CorDesc
    FROM CourseList
    ORDER by Dept_ID ASC, CorNumber ASC
</cfquery>

<html>
<head>
<title>cfgrid Example</title>
</head>
<body>
<h3>cfgrid Example</h3>
<I>You can update the Name, Level, and Description information for courses.</i>
<!--- The cfform tag must surround a cfgrid control. --->
<cfform action = "#CGI.SCRIPT_NAME#">
```

```
    <cfgrid name = "FirstGrid" width = "500"
            query = "GetCourses" colheaderbold="Yes"
            font = "Tahoma" rowHeaders = "No"
            selectColor = "Red" selectMode = "Edit" >
        <!--- cfgridcolumn tags arrange the table and control the display. --->
        <!--- Hide the primary key, required for update --->
        <cfgridcolumn name = "Course_ID" display = "No">
        <!--- select="No" does not seem to have any effect!!! --->
        <cfgridcolumn name = "Dept_ID" header = "Department" Select="No" width="75"
            textcolor="blue" bold="Yes">
        <cfgridcolumn name = "CorNumber" header = "Course ##" Select="No" width="65">
        <cfgridcolumn name = "CorName" header = "Name" width="125">
        <cfgridcolumn name = "CorLevel" header = "Level" width="85">
        <cfgridcolumn name = "CorDesc" header = "Description" width="125">
    </cfgrid>
    <br>
    <cfinput type="submit" name="gridEntered">
</cfform>
</body>
</html>
```

cfgridrow

説明

クエリーを行データのソースとして使用しない cfgrid コントロールを定義することができます。query 属性が cfgrid タグで指定されている場合、cfgridrow タグは無視されます。

カテゴリ

フォームタグ

シンタックス

```
<cfgridrow
    data = "col1, col2, ..."
    delimiter = "delimiter character">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfgrid、cfgridcolumn、cfgridupdate、cfform、cfinput、cfselect、cfslider、cftextarea、cftree

属性

属性	必須 / オプション	デフォルト	説明
data	必須		列の値の区切りリストです。値に区切り文字が含まれる場合は、区切り文字をもう 1 つ使用してエスケープする必要があります。
delimiter	オプション	,(カンマ)	列の値の間に挿入する区切り文字です。

例

次の例は、cfgridrow タグを使用してリストデータから cfgrid タグを挿入する方法を示しています。

```
<!--- Set two lists, each with the data for a grid column. --->
<cfset cities = "Rome,Athens,Canberra,Brasilia,Paris">
<cfset countries = "Italy,Greece,Australia,Brazil,France">

<cfform name = "cities">
    <cfgrid name="GeoGrid" autowidth = "yes" vspace = "4" height = "120"
            font="tahoma" rowheaders="no">
        <cfgridcolumn name="City" header="City">
        <cfgridcolumn name="Country" header="Country">
        <!--- Loop through the lists using cfgridrow to poplulate the grid. --->
        <cfloop index="i" from="1" to="#ListLen(cities)#">
            <cfgridrow data ="#ListGetAt(cities, i)#,#ListGetAt(countries, i)#">
        </cfloop>
    </cfgrid><br><br>
</cfform>
```

cfgridupdate

説明

cfgrid タグとともに使用します。編集したグリッドデータからデータソースを直接更新します。このタグは、データソースへの直接のインターフェイスを提供します。

このタグでは、最初に delete 行アクションが適用され、次に insert 行アクション、最後に update 行アクションが適用されます。エラーが発生すると行の処理は停止されます。

カテゴリ

フォームタグ

シンタックス

```
<cfgridupdate
    grid = "grid name"
    dataSource = "data source name"
    tableName = "table name"
    keyOnly = "yes|no">
    password = "data source password"
    tableOwner = "table owner"
    tableQualifier = "qualifier"
    username = "data source user name">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfgrid、cfgridcolumn、cfgridrow、cfform、cfapplet、cfinput、cfselect、cfslider、cftextinput、cftree

履歴

ColdFusion 10：clientInfo 属性が追加されました。

ColdFusion MX: connectString、dbName、dbServer、dbtype、provider、および providerDSN 属性は非推奨になりました。ColdFusion 5 以降のリリースでは、これらは機能せず、エラーを引き起こす可能性があります。

属性

属性	必須 / オプション	デフォルト	説明
clientInfo	オプション		データベース接続で設定するクライアントのプロパティが含まれる構造体です。
grid	必須		更新アクションのソースとなる cfgrid フォーム要素の名前です。
dataSource	必須		更新アクションのデータソースの名前です。
tableName	必須		更新するテーブルの名前です。 ORACLE ドライバの場合、エントリは大文字で指定する必要があります。 Sybase ドライバの場合、エントリの大文字と小文字が区別されるため、テーブルの作成時に使用した名前と大文字小文字を同じにする必要があります。
keyOnly		no	update アクションに適用されます。 • yes: WHERE 条件がキー値に制限されます。 • no: WHERE 条件には、キー値と、変更されたフィールドの元の値が含まれます。
password	オプション		ODBC セットアップで指定されている password の値よりも優先されます。
tableOwner	オプション		テーブル所有者です。サポートされている場合に限ります。
tableQualifier	オプション		テーブル識別子です。サポートされている場合に限ります。内容は次のとおりです。 • SQL Server および Oracle ドライバの場合 : テーブルが含まれているデータベースの名前 • Intersolv dBASE ドライバの場合 : DBF ファイルのディレクトリ
username	オプション		ODBC セットアップで指定されている username の値よりも優先されます。

例

次の例では、cfgrid タグを使用してデータベースを更新し、レコード全体を追加および削除するか、個々のセルのデータを更新します。cfgridupdate タグは、送信されたフォームのデータを処理し、データベースを更新します。

```
 <!--- If the gridEntered form field exists, the form was submitted. Perform gridupdate. --->
<cfif IsDefined("form.gridEntered") is True>
    <cfgridupdate grid = "FirstGrid" dataSource = "cfdocexamples" Keyonly="true"
        tableName = "CourseList">
    </cfif>

<!--- Query the database to fill up the grid. --->
<cfquery name = "GetCourses" dataSource = "cfdocexamples">
    SELECT Course_ID, Dept_ID, CorNumber, CorName, CorLevel, CorDesc
    FROM CourseList
    ORDER by Dept_ID ASC, CorNumber ASC
</cfquery>

<h3>cfgrid Example</h3>
<I>Try adding a course to the database, and then deleting it.</i>
<cfform>
<cfgrid name = "FirstGrid" width = "450"
    query = "GetCourses" insert = "Yes" delete = "Yes"
    font = "Tahoma" rowHeaders = "No"
    colHeaderBold = "Yes"
    selectMode = "EDIT"
    insertButton = "Insert a Row" deleteButton = "Delete selected row" >
</cfgrid><br>
<cfinput type="submit" name="gridEntered">
</cfform>...
```

cfheader

説明
クライアントに返すカスタム HTTP レスポンスヘッダを生成します。

カテゴリ
データ出力タグ、ページ処理タグ

シンタックス
```
<cfheader
    charset="character set"
    name = "header name"
    value = "header value">

OR

<cfheader
    statusCode = "status code"
    statusText = "status text">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目
cfcache、cfflush、cfhtmlhead、cfinclude、cfsetting、cfsilent、cfcontent

履歴
ColdFusion MX 6.1: name 属性の動作が変更されました。**cfheader name="Content-Disposition"** の場合は、このヘッダの値をエンコードするときにデフォルトのファイル文字エンコードが使用されるので、そのファイルで使用されている文字エンコード内の文字をファイル名に含めることができます。

属性

属性	必須 / オプション	デフォルト	説明
charset	オプション	UTF-8	ヘッダの値をエンコードする文字エンコードです。一般的に使用される値を次に示します。 • utf-8 • iso-8859-1 • windows-1252 • us-ascii • shift_jis • iso-2022-jp • euc-jp • euc-kr • big5 • euc-cn • utf-16 文字エンコードの詳細については、www.w3.org/International/O-charset.html を参照してください。
name	statusCode を指定しない場合は必須		ヘッダ名です。
statusCode	name を指定しない場合は必須		数値。HTTP ステータスコードを表します。
statusText	オプション		ステータスコードの説明です。
value	オプション		HTTP ヘッダの値です。

使用方法
ページ内で cfflush タグの後にこのタグを使用するとエラーが発生します。

例
```
 <h3>cfheader Example</h3>

<p>cfheader generates custom HTTP response headers to return to the client.
<p>This example forces browser client to purge its cache of requested file.
<cfheader name="Expires" value="#GetHttpTimeString(Now())#">
```

cfhtmlhead

説明
生成される HTML ページの head セクションにテキストを書き込みます。

カテゴリ
ページ処理タグ

シンタックス

```
<cfhtmlhead
    text = "text">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfcache、cfflush、cfheader、cfinclude、cfsetting、cfsilent

属性

属性	必須 / オプション	デフォルト	説明
text	必須		HTML ページの <head> 領域に追加するテキストです。

使用方法

このタグは、HTML ページのヘッダ内に JavaScript コードを埋め込む場合や、meta、link、title、base などの他の HTML タグを配置する場合などに使用します。

ページ内で cfflush タグの後にこのタグを使用するとエラーが発生します。

例

```
<!--- This example adds a favicon to the HTML Head of every page view. The extra CRLF at
    the end cleans up view source. Note the embedded tag uses double quotes and the cfhtmlhead
    text attribute uses single quotes, but it could be the other way around too.
--->

<cfhtmlhead
    text='<link href="/blog/custom/img/favicon.ico" rel="shortcut icon" type="image/x-
icon">#chr(13)##chr(10)#'>
```

cfhttp

説明

HTTP リクエストを生成し、サーバーからのレスポンスを処理します。

カテゴリ

インターネットプロトコルタグ

シンタックス

```
<cfhttp
    url = "server URL"
    charset = "character encoding"
    clientCert = "filename"
    clientCertPassword = "password"
    columns = "query columns"
    delimiter = "character"
    file = "filename"
    firstrowasheaders = "yes|no"
    getAsBinary = "auto|yes|no|never"
    method = "method name"
    multipart = "yes|no"
    name = "query name"
    password = "password"
    path = "path"
    port = "port number"
    proxyServer = "host name"
    proxyPort = "port number"
    proxyUser = "username"
    proxyPassword = "password"
    redirect = "yes|no"
    resolveURL = "yes|no"
    result = "result name"
    textQualifier = "character"
    throwOnError = "yes|no"
    timeout = "time-out period in seconds"
    username = "username"
    userAgent = "user agent">

    cfhttpparam tags [optional for some methods]

</cfhttp>
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfhttpparam、GetHttpRequestData、cfftp、cfldap、cfmail、cfpop

履歴

ColdFusion 8: clientCert 属性と clientCertPassword 属性が追加されました。

ColdFusion MX 7.01: getAsBinary 属性の値として "never" が追加されました。

ColdFusion MX 7: result 属性が追加されました。

ColdFusion MX 6.1:

- HEAD、PUT、DELETE、OPTIONS、および TRACE の各メソッドのサポートが追加されました。
- multipart、getAsBinary、proxyUser、および proxyPassword の各属性が追加されました。
- httpparam 動作が変更されました。すべてのオペレーションに httpparam タグを指定できます。
- cfhttp.errorDetail という戻り変数が追加されました。
- レスポンスの本文コンテンツのタイプがテキストと見なされるように変更されました。
- 複数ヘッダーの動作が変更されました。同じタイプの複数ヘッダーは配列で返されるようになりました。
- HTTPS プロキシトンネリングのサポートが追加されました。

- コードとマニュアルのバグが修正されました。

ColdFusion MX:

- charset 属性と firstrowasheaders 属性が追加されました。
- SSL (Secure Sockets Layer) のサポートが変更されました。ColdFusion は Sun JSSE ライブラリを使用します。このライブラリは、SSL をサポートするために 128 ビット暗号化をサポートします。

属性

次に示す属性は HTTP トランザクションを制御するためのもので、どの HTTP メソッドでも使用できます。

属性	必須 / オプション	デフォルト	説明
url	必須	http プロトコルを使用	リクエストを処理するサーバー上のリソースのアドレスです。この URL にはホスト名か IP アドレスを含める必要があります。 トランザクションプロトコル (http:// または https://) を指定しなかった場合は、デフォルトのプロトコルである http が使用されます。 この属性でポート番号を指定した場合は、port 属性の値が上書きされます。 cfhttpparam タグの URL 属性を使用すると、クエリー文字列の属性と値のペアがこの URL に追加されます。
charset	オプション	リクエストの場合：UTF-8 レスポンスの場合：レスポンスの Content-Type ヘッダに指定された charset、レスポンスが文字セットを指定していない場合は UTF-8	URL クエリー文字列、フォームまたはファイルデータ、レスポンスなどのリクエストの文字エンコードです。一般的に使用される値を次に示します。 • utf-8 • iso-8859-1 • windows-1252 • us-ascii • shift_jis • iso-2022-jp • euc-jp • euc-kr • big5 • euc-cn • utf-16 文字エンコードの詳細については、www.w3.org/International/O-charset.html を参照してください。
clientCert	オプション		リクエストのためのクライアント証明書が含まれている PKCS12 形式のファイルへの絶対パスです。
clientCertPassword	オプション		クライアント証明書を復号するために使用されるパスワードです。
compression	オプション		ターゲット Web サーバーの圧縮ステータスです。サポートされている値は none のみです。ターゲットの Web サイトが、HTTP 圧縮が有効な IIS で実行されている場合は、この属性を使用して、GET または POST オペレーションを実行するときの接続エラーを回避できます。

属性	必須 / オプション	デフォルト	説明
getAsBinary	オプション	no	• no: ColdFusion がレスポンス本文のタイプをテキストとして認識しない場合は、それを ColdFusion オブジェクトに変換します。 • auto: ColdFusion がレスポンス本文のタイプをテキストとして認識しない場合は、それを ColdFusion Binary タイプデータに変換します。 • yes: レスポンス本文のコンテンツを必ず ColdFusion Binary タイプデータに変換します。ColdFusion がレスポンス本文のタイプをテキストとして認識する場合でも同様です。 • never: 特定の MIME タイプから ColdFusion の Binary タイプデータへの自動変換が防止され、返される内容が常にテキストとして処理されます。 ColdFusion は、次の場合にレスポンス本文をテキストとして認識します。 • ヘッダにコンテンツタイプが指定されていない場合 • コンテンツタイプが "text" で始まっている場合 • コンテンツタイプが "message" で始まっている場合 • コンテンツタイプが "application/octet-stream" である場合 ColdFusion がレスポンス本文をテキストとして認識せず、オブジェクトに変換した場合でも、その本文がテキストから構成されている場合は、cfoutput タグを使って表示することができます。cfoutput タグでは Binary タイプデータを表示できません。バイナリデータをテキストに変換するには、ToString 関数を使用します。
method	オプション	GET	• GET: サーバーに情報をリクエストします。リクエストされた情報をサーバーが識別するために必要なデータはすべて、URL 内か cfhttptype="URL" タグ内に含まれている必要があります。 • POST: 処理する情報をサーバーに送信します。cfhttpparam タグが必要です。フォームに似たデータを送信するときによく使われます。 • PUT: 指定の URL にメッセージ本文を保管するようサーバーにリクエストします。このメソッドは、サーバーにファイルを送信するときに使われます。 • DELETE: 指定の URL を削除するようサーバーにリクエストします。 • HEAD: GET メソッドと同じですが、サーバーはレスポンス内にメッセージ本文を送信しません。このメソッドは、ハイパーテキストリンクの有効性やアクセシビリティのテスト、ドキュメントのタイプや修正時刻の確認、またはサーバーのタイプの判別に使用します。 • TRACE: 受信した HTTP ヘッダをレスポンス本文に挿入して送信者にエコーバックするようサーバーにリクエストします。トレースリクエストには本文がありません。このメソッドを使用すると、ColdFusion アプリケーションはサーバー側で何を受信したかを調べ、そのデータをテストや診断情報に利用することができます。 • OPTIONS: サーバーまたは指定の URL で使用できる通信オプションについての情報をリクエストします。このメソッドを使用すると、ColdFusion アプリケーションは、追加のサーバー活動を要求しなくても、URL に関連付けられているオプションや要件、あるいはサーバーの機能を判別することができます。
password	オプション		基本認証のターゲット URL にパスワードを渡すために使用します。username と組み合わせて、認証ヘッダで渡される base64 エンコード文字列を形成します。統合 Windows 認証、NTLM 認証、または Kerebos 認証には対応しません。

属性	必須 / オプション	デフォルト	説明
port	オプション	http の場合は 80、 https の場合は 443	リクエストの送信先となるサーバー上のポート番号です。url 属性のポート値は、この値よりも優先されます。
proxyServer	オプション		リクエストの送信先となるプロキシサーバーのホスト名または IP アドレスです。
proxyPort	オプション	80	プロキシサーバー上で使用するポート番号です。
proxyUser	オプション		プロキシサーバーに提供するユーザー名です。
proxyPassword	オプション		プロキシサーバーに提供するパスワードです。
redirect	オプション	yes	レスポンスヘッダに Location フィールドが含まれ、ColdFusion が 300 系列 (リダイレクション) のステータスコードを受け取った場合、そのフィールド内で指定された URL に実行をリダイレクトするかどうかを指定します。 • yes: 指定のページに実行をリダイレクトします。 • no: 実行を停止し、レスポンス情報を cfhttp 変数に返します。あるいは、throwOnError 属性が true の場合はエラーを返します。 cfhttp.responseHeader.Location 変数にはリダイレクトパスが含まれています。ColdFusion では、1 つのリクエストで最大 4 回までリダイレクトが行われます。これよりも多くなると、redirect ="no" を指定した場合と同じ処理が行われます。 **メモ** : cflocation タグは、Location ヘッダ値として url 属性を使用する HTTP 302 レスポンスを生成します。

属性	必須 / オプション	デフォルト	説明
resolveURL	オプション	no	• no: レスポンス本文の中の URL を変換しません。その結果、レスポンス本文に含まれる相対 URL リンクはすべて機能しなくなります。 • yes: 取得したページ内でもリンクが機能するように、レスポンス本文中の URL (ポート番号も含む) を絶対 URL に変換します。次の HTML タグに適用されます。 • img • src • a href • form action • applet code • script src • embed src • embed pluginspace • body background • frame src • bgsound src • object data • object classid • object codebase • object usemap file 属性および path 属性を使用する場合は、URL を変換しません。
result	オプション		結果を受け取る代替変数を指定することができます。
throwOnError	オプション	no	• yes: サーバーがエラーレスポンスコードを返す場合に、cftry と cfcatch (または ColdFusion エラーページ) を使って検出できる例外を返します。 • no: エラーレスポンスを返す場合に例外を返しません。この場合、アプリケーションは、エラーの有無とその原因を cfhttp.StatusCode 変数から判別することができます。

属性	必須 / オプション	デフォルト	説明
timeout	オプション		リクエストの最大待機時間を秒数で指定します。レスポンスがないままタイムアウトになった場合は、そのリクエストが失敗したものと見なされます。 クライアントが URL 検索パラメータ内でタイムアウトを指定した場合は (例 : ?RequestTime=120)、その URL タイムアウトと timeout 属性値のうち、短いほうが採用されます。これにより、リクエストが確実にページよりも先に (あるいは同時に) タイムアウトになります。 URL にタイムアウトが指定されていない場合は、Administrator タイムアウトと timeout 属性値のうち、短いほうが採用されます。 上記のどの方法でもタイムアウト値が設定されていない場合は、ColdFusion は cfhttp リクエストの処理を無限に待機することになります。
userAgent	オプション	ColdFusion	ユーザーエージェントリクエストヘッダに配置されるテキストです。リクエストクライアントソフトウェアを識別するために使われます。ColdFusion アプリケーションをブラウザのように見せることができます。
username	オプション		基本認証のターゲット URL にユーザー名を渡すために使用します。password と組み合わせて、認証ヘッダで渡される base64 エンコード文字列を形成します。統合 Windows 認証、NTLM 認証、または Kerberos 認証には対応しません。

PUT メソッドでは、次の属性を使用して、httpparam type="formField" で指定されたデータの送信方法を設定します。

属性	必須 / オプション	デフォルト	説明
multipart	オプション	no (リクエストが File タイプデータを含んでいる場合にのみマルチパートとして送信)	cfhttpparam type="formField" タグで指定されたすべてのデータを、マルチパートフォームデータ (Content-Type が multipart/form-data) として送信することを ColdFusion に指示します。ColdFusion のデフォルトでは、formField データだけを含む、Content Type が application/x-www-form-urlencoded の cfhttp リクエストが送信されます。ただし、リクエストに File タイプデータも含まれている場合は、すべての部分に multipart/form-data コンテンツタイプが使用されます。 この属性を yes に設定した場合は、Content-Type の説明の中でリクエストの文字セットも送信されます。すべてのフォームフィールドデータをこの文字エンコードでエンコードする必要があります。ColdFusion はこのデータに URLEncode を実行しません。フィールド名は ISO-88591-1 または ASCII である必要があります。一部の http パーサ (以前のバージョンの ColdFusion で使われていたものも含む) は、マルチパートフォームフィールドの文字エンコードの説明を無視します。

次の属性はマルチパートヘッダフィールドを設定し、YouTube にビデオをアップロードする場合などに使用されます。

属性	必須 / オプション	デフォルト	説明
multipartType	オプション	form-data	マルチパートヘッダフィールドを related または form-data に設定できます。デフォルトでは、値は form-data です。

例 :

```
<!--- Get Video --->
<cfset videoName = "<vedio path>\hello.wmv">
<cfset videoFileName = "hello.wmv">
<cfoutput>
<!--- Set User Account Data --->
<cfset clientKey = "client key from google"/>
<cfset devKey = "<dev key from google>"/>
<cfset youTubeUploadURL = "http://uploads.gdata.youtube.com/feeds/api/users/default/uploads"/>
<!--- Authenticate with Google / YouTube --->
<cfhttp url="https://www.google.com/accounts/ClientLogin" method="post" result="result" charset="utf-8">
    <cfhttpparam type="formfield" name="accountType" value="HOSTED_OR_GOOGLE">
    <cfhttpparam type="formfield" name="Email" value="<gmail id>">
    <cfhttpparam type="formfield" name="Passwd" value="<password>">
    <cfhttpparam type="formfield" name="service" value="youtube">
    <cfhttpparam type="formfield" name="source" value="youtubecode">
</cfhttp>
<!--- Create Auth Token --->
<cfset content = result.filecontent>
<cfset authdata = structNew()>
<cfloop index="line" list="#content#" delimiters="#chr(10)#">
<cfset dtype = listFirst(line, "=")>
<cfset value = listRest(line, "=")>
<cfset authdata[dtype] = value></cfloop>
<!--- Create ATOM XML and save to a file to be sent with video --->
    <cfsavecontent variable="meta"><cfoutput>
    <entry xmlns="http://www.w3.org/2005/Atom"
    xmlns:media="http://search.yahoo.com/mrss/"
    xmlns:yt="http://gdata.youtube.com/schemas/2007">
    <media:group>
    <media:title type="plain">WithOutQuotes</media:title>
    <media:description type="plain">Test Description</media:description>
    <media:category
            scheme="http://gdata.youtube.com/schemas/2007/categories.cat">People
    </media:category>
    <media:keywords>yourvideo</media:keywords>
    </media:group>
    </entry>
</cfoutput>
</cfsavecontent>
<cfset tmpfile = expandPath("./meta.xml")/>
<cffile action="write" file="#tmpfile#" output="#trim(meta)#" />

<!--- Upload video --->
<cfhttp url="#youTubeUploadURL#" result="result" method="POST"  timeout="450" multipartType="related">
<cfhttpparam type="header" name="Authorization" value="GoogleLogin auth=#authdata.auth#">
<cfhttpparam type="header" name="X-GData-Client" value="#variables.clientkey#">
<cfhttpparam type="header" name="X-GData-Key" value="key=#variables.devkey#">
<cfhttpparam type="header" name="Slug" value="#videoFileName#">
    <!---<CFHTTPPARAM type="HEADER" name="Connection" value="Keep-Alive"> --->
    <!--- Send 2 files --->
    <cfhttpparam type="file" name="API_XML_Request" file="#tmpfile#"
    mimetype="application/atom+xml">
<cfhttpparam type="file" name="file" file="#videoName#" mimetype="video/*">
</cfhttp>
<cfdump var="#result#"/>
</cfoutput>
```

次の属性を使用すると、必要なオペレーションの結果を返すために使用する変数の名前を指定することができます。指定した名前が、戻り値にアクセスする際の接頭辞として cfhttp と置き換わります。たとえば、result 属性を myResult に設定した場合は、#myResult.FileContent# として FileContent にアクセスします。

result 属性では、複数のページから同時に呼び出される関数または CFC について、一方の呼び出しの結果が他方の呼び出しの結果を上書きしないようにすることができます。cfhttp get オペレーションで返される変数の詳細については、「使用方法」の「cfhttp の get オペレーションで返される変数」を参照してください。

属性	必須 / オプション	デフォルト	説明
result	オプション		結果を返すために使用する変数の名前を指定します。

次の属性は、HTTP レスポンス本文をファイルに保管するときに使用します。GET、POST、PUT、DELETE、OPTIONS、TRACE の各メソッドでレスポンス本文をファイルに保管することができますが、DELETE メソッドと OPTIONS メソッドではほとんど役に立ちません。

属性	必須 / オプション	デフォルト	説明
file	path 属性を指定していて、GET メソッドでない場合は必須	「説明」を参照	レスポンス本文の保管先となるファイルの名前です。 GET オペレーションでは、URL 内でリクエストされたファイルがデフォルトになります (ファイルが指定されている場合)。たとえば、GET メソッドの URL が http:www.myco.com/test.htm である場合、デフォルトファイルは test.htm です。 この属性ではディレクトリへのパスを指定しないでください。パスは path 属性で指定します。
path	file 属性を指定している場合は必須		HTTP レスポンス本文をファイルに保管することを ColdFusion に指示します。ファイルの保管先となるディレクトリへの絶対パスを指定します。

ファイルのメモリ内のディレクトリを指定するには、path 属性で次のシンタックスを使用します。メモリ内のファイルを使用すると、一時的データの処理が速くなります。

```
ram:///filepath
```

ファイルパスには、複数のディレクトリを含めることができます (例 : ram:///petStore/images/)。パスに含まれるディレクトリを事前に作成しておく必要があります。メモリ内のファイルの使用方法について詳しくは、『ColdFusion アプリケーションの開発』の Working with in-memory files を参照してください。

次の属性は、HTTP レスポンス本文を ColdFusion クエリーオブジェクトに変換するときに使用します。これらの属性は GET メソッドと POST メソッドでのみ使用できます。

属性	必須 / オプション	デフォルト	説明
columns	オプション	レスポンスの 1 行目には列名が含まれます。	このクエリーの列名をカンマで区切って指定します。スペースは使用しません。列名は文字で始める必要があります。2 文字目以降には、文字、数字、アンダースコア文字 (_) を使用できます。 レスポンス内に列名ヘッダがない場合は、列名を識別するためにこの属性を指定します。 この属性を設定したときに、firstrowasHeader 属性が true (デフォルト) だった場合には、この属性で指定した列名によってレスポンスの 1 行目が置き換えられます。この動作を利用して、リクエストによって取得された列名を独自の名前に置き換えることができます。 この属性またはレスポンスからの列名の中で重複する列ヘッダがある場合、その名前にアンダースコアが付加され、固有の名前が作成されます。 この属性で指定した列の数が HTTP レスポンス本文の中の列数と等しくない場合は、エラーが発生します。
delimiter	オプション	, (カンマ)	クエリー列を区切る文字です。レスポンス本文はこの文字を使用してクエリー列を区切る必要があります。
firstrowasheaders	オプション	yes	ColdFusion で、クエリーのレコードセットの最初の行を処理する方法を決定します。 • yes: 1 行目を列ヘッダとして扱います。columns 属性を指定した場合は、ファイルの 1 行目が無視されます。 • no: 1 行目をデータとして扱います。columns 属性を指定しなかった場合は、"column_1" のように、"column" という単語に数字を付加した列名が作成されます。
name	オプション		返された HTTP レスポンス本文から指定の名前のクエリーオブジェクトを作成するよう ColdFusion に指示します。
textQualifier	オプション	" [二重引用符]	テキスト列の始まりと終わりを示すオプションの文字です。レスポンス本文中のテキストフィールドがフィールド値の一部に区切り文字を含んでいる場合は、テキストフィールドをこの文字で囲む必要があります。 この文字を列テキストの中に含めるときには、文字を 2 つ指定してエスケープします。たとえば、テキスト修飾子が二重引用符である場合に、この文字をエスケープするには "" と指定します。

使用方法

cfhttp タグは、HTTP リクエストを作成し、返された結果を処理するための汎用ツールです。このタグを使用すると、ほとんどの標準 HTTP リクエストタイプを生成することができます。リクエストヘッダと本文コンテンツを指定するには、このタグ内で cfhttpparam タグを使用します。

ColdFusion は、cfhttp リクエストに対するレスポンスを受け取ると、そのレスポンス本文をファイルまたは cfhttp.FileContent 文字列変数に保管することができます。本文テキストが結果セットとして構築されている場合は、その本文テキストをクエリーオブジェクトに入れることができます。さらに、返されたすべてのヘッダ値へのアクセス、エラーステータスとリダイレクションの処理方法の指定、リクエストをハングさせないためのタイムアウトの指定もできます。

HTTP プロトコルは World Wide Web のバックボーンであり、すべての Web トランザクションで使用されます。cfhttp タグはほとんどのリクエストタイプを生成できるので、非常に柔軟に使用することができます。たとえば次のような使い方があります。

- Web サービスとしては使用できないダイナミックな Web サイトやサービスとやり取りする (SOAP Web サービスへのアクセスには cfinvoke タグを使用)
- Web サーバー上のイメージなど、HTML ページやその他のファイルのコンテンツを取得し、それを CFML ページ内で使用したり、ファイルに保管したりする
- url 属性に https プロトコルを指定して、サーバーに安全なリクエストを送信する

- POST メソッドを使用して、multipart/form-data スタイルのデータを、そのようなデータを処理して結果を返すことのできる任意の URL (CGI 実行可能ファイルや、その他の ColdFusion ページなど) に送信する
- PUT メソッドを使用して、FTP リクエストを受け付けないサーバーにファイルをアップロードする

このタグの本文には、cfhttpparam タグを含めることができます。また、PUT リクエストと POST リクエストの場合はこのタグは必須です。このタグに cfhttpparam タグを含める場合は、</cfhttp> 終了タグが必要です。

cfhttp タグを含む HTTPS を使用するためには、各 Web サーバーに対する証明書を、ColdFusion が使用する JRE のキーストアに手動でインポートすることが必要になる場合があります。JSSE (Java Secure Sockets Extension) が認識する機関 (Verisign など) によって証明書が署名 (発行) される場合、つまり、署名する機関が既に cacerts にある場合、この手順は必要ありません。ただし、SSL (Secure Sockets Layer) 証明書を自己発行している場合は、この手順が必要になることがあります。

証明書を手動でインポートするには :

1. SSL サーバー上の当該ページに進みます。
2. 鍵のアイコンをダブルクリックします。
3. [詳細] タブをクリックします。
4. [ファイルにコピー] をクリックします。
5. base64 オプションを選択してファイルを保存します。
6. CER ファイルを C:¥ColdFusion9¥runtime¥jre¥lib¥security (または JRE ColdFusion が使用する任意の場所) にコピーします。
7. 同じディレクトリの次のコマンドを実行します (keytool.exe は C:¥ColdFusion9¥runtime¥jre¥bin にあります)。

```
keytool -import -keystore cacerts -alias giveUniqueName -file filename.cer
```

cfhttp の get オペレーションで返される変数

cfhttp タグは、次の変数を返します。result 属性を設定した場合は、割り当てた名前が接頭辞として cfhttp と置き換わります。その他の情報については、result 属性を参照してください。

名前	説明
cfhttp.charSet	レスポンスの Content-Type ヘッダで指定されたレスポンス文字セット (文字エンコード) です。
cfhttp.errorDetail	HTTP サーバーへの接続が失敗した場合に、失敗の詳細を記録します (例 : "Unknown host: my.co.com")。それ以外の場合は、空の文字列になります。エラー状態について、他の変数を確認する前にこの変数を確認することをお勧めします。
cfhttp.fileContent	レスポンス本文です。たとえば、GET オペレーションによって取得された HTML ページのコンテンツなどです。レスポンスをファイルに保管した場合は空になります。
cfhttp.header	すべてのヘッダ情報を 1 つの文字列に格納した生のレスポンスヘッダです。cfhttp.responseHeader 変数と同じ情報を含みます。
cfhttp.mimeType	レスポンスの Content-Type ヘッダで指定されている MIME タイプです (例 : text/html)。

名前	説明
cfhttp.responseHeader	構造体の形式にされたレスポンスヘッダです。要素キーは、Content-Type や Status_Code などのヘッダ名です。あるヘッダタイプのインスタンスが複数ある場合は、そのタイプの値が配列に格納されます。 よく使用されるテクニックは、#cfhttp.resonseHeader[fieldVariable]# のように、cfhttp.responseHeader 構造体をダイナミック配列としてダイナミックにアクセスすることです。
cfhttp.statusCode	HTTP Explanation ヘッダ値の後に続く HTTP status_code ヘッダ値です (例 : "200 OK")。
cfhttp.text	ブール値。レスポンス本文のコンテンツタイプがテキストである場合は true になります。ColdFusion は、次の場合にレスポンス本文をテキストとして認識します。 • ヘッダにコンテンツタイプが指定されていない場合 • コンテンツタイプが "text" で始まっている場合 • コンテンツタイプが "message" で始まっている場合 • コンテンツタイプが "application/octet-stream" である場合

区切り形式のテキストファイルからクエリーを構築する方法

cfhttp タグでは、レスポンス本文から ColdFusion クエリーオブジェクトを作成することができます。これを行うには、レスポンス本文が、数行から成るテキストを含んでいて、その各行が列区切り文字によって区切られたいくつかのフィールドを含んでいる必要があります。デフォルトの区切り文字はカンマ (,) です。レスポンス本文では、テキスト修飾子を使用することもできます。デフォルトのテキスト修飾子は二重引用符 (") です。文字列フィールドをテキスト修飾子で囲むと、そのフィールドに区切り文字を含めることができます。フィールドテキスト内にテキスト修飾子を含める場合は、その文字を重ねて指定してエスケープします。次の例は、クエリーに変換される 2 行のリクエスト本文を示しています。各行はカンマで区切られた 3 つのフィールドを含んでいます。

```
Field1,Field2,Field3
"A comma, in text","A quote: ""Oh My!""",Plain text
```

ColdFusion がこのデータをどのように処理するかを確認するには、次のコードを実行します。

```
<cfhttp method="Get"
    url="127.0.0.1:8500/tests/escapetest.txt"
    name="onerow">
<cfdump var="#onerow#"><br>
```

列名は次の 3 通りの方法で指定できます。

- デフォルトでは、レスポンスの 1 行目は列名として使用されます。
- columns 属性にカンマ区切りの値を指定した場合は、その属性で指定した名前が列名として使用されます。レスポンスの 1 行目にデータが含まれている場合は、firstRowAsHeaders="no" を設定します。データが含まれていない場合、1 行目は無視されます。
- columns 属性を指定せず、firstrowasheaders="no" を設定した場合は、Column_1、Column_2、... というように列名が生成されます。

cfhttp タグは、このタグによって返されるデータ内の列名をチェックして、列名が文字で始まっていること、および文字、数字、アンダースコア文字 (_) だけを含んでいることを確認します。

ColdFusion は無効な列名をチェックします。列名は文字で始める必要があります。2 文字目以降には、文字、数字、アンダースコア (_) を使用できます。列名が無効な場合は、エラーが発生します。

注意事項

- ColdFusion Administrator のタイムアウトと URL のタイムアウトを有効にするには、ColdFusion Administrator の [サーバーの設定] ページでタイムアウトを有効にします。詳細については、『ColdFusion 設定と管理』を参照してください。

- cfhttp タグはすべてのオペレーションで基本認証をサポートします。
- cfhttp では、SSL を使用してセキュアトランザクション通信が行われます。
- HTTP レスポンス本文をファイルに保管した場合、それは CFHTTP.FileContent 変数には格納されず、クエリーオブジェクトも生成されません。レスポンス本文をファイルに保管しなかった場合、それは CFHTTP.FileContent 変数に格納されます。このとき、name 属性を指定していればクエリーオブジェクトが生成されます。
- cfhttp タグは、NTLM 認証やダイジェスト認証をサポートしません。
- Microsoft IIS を使用している場合、HTTP ヘッダサイズの制限はありません。HTTP ヘッダサイズの制限を指定するには、IIS で設定します。

例

```
<!--- This example displays the information provided by
    the Designer & Developer Center XML feed,
    http://www.adobe.com/devnet/resources/_resources.xml
    See http://www.adobe.com/devnet/articles/xml_resource_feed.html
    for more information on this feed. --->

<!--- Set the URL address. --->
<cfset urlAddress="http://www.adobe.com/devnet/resources/_resources.xml">

<!--- Use the CFHTTP tag to get the file content represented by urladdress.
        Note that />, not an end tag, terminates this tag. --->
<cfhttp url="#urladdress#" method="GET" resolveurl="Yes" throwOnError="Yes"/>

<!--- Parse the XML and output a list of resources. --->
<cfset xmlDoc = XmlParse(CFHTTP.FileContent)>
<!--- Get the array of resource elements, the xmlChildren of the xmlroot. --->
<cfset resources=xmlDoc.xmlroot.xmlChildren>
<cfset numresources=ArrayLen(resources)>

<cfloop index="i" from="1" to="#numresources#">
    <cfset item=resources[i]>
    <cfoutput>
        <strong><a href=#item.url.xmltext#>#item.title.xmltext#</strong></a><br>
        <strong>Author</strong>  #item.author.xmltext#<br>
        <strong>Applies to these products</strong><br>
        <cfloop index="i" from="4" to="#arraylen(item.xmlChildren)#">
            #item.xmlChildren[i].xmlAttributes.Name#<br>
        </cfloop>
        <br>
    </cfoutput>
</cfloop>
```

cfhttpparam

説明

cfhttp タグ本文でのみ使用できます。cfhttp の POST オペレーションに必須です。他のオペレーションではオプションです。HTTP リクエストを作成するためのパラメータを指定します。

カテゴリ

インターネットプロトコルタグ

シンタックス

```
<cfhttpparam
    type = "transaction type"
    encoded = "yes|no"
    file = "filename"
    mimeType = "MIME type designator"
    name = "data name"
    value = "data value">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfhttp、GetHttpRequestData、cfftp、cfldap、cfmail、cfmailparam、cfpop

履歴

ColdFusion MX 6.1:

- header タイプと body タイプが追加されました。
- encoded 属性と mimeType 属性が追加されました。
- HTTP メソッドの動作が変更されました。すべての HTTP メソッドで httpparam タグを使用することができます。
- name 属性の必要条件が変更されました。すべてのタイプに必須ではなくなりました。

属性

属性	必須 / オプション	デフォルト	説明
type	必須		情報のタイプです。 • header: HTTP ヘッダを指定します。ColdFusion はこのヘッダの URL エンコードを行いません。 • CGI: HTTP ヘッダを指定します。ColdFusion はデフォルトでこのヘッダの URL エンコードを行います。 • body: HTTP リクエストの本文を指定します。ColdFusion は、Content-Type ヘッダまたは本文にふくまれる URL エンコードの設定を自動的に行いません。Content-Type を指定するには、type=header が指定された個別の cfhttpparam タグを使用します。 • XML: リクエストの Content-Type が text/xml であると識別します。value 属性に HTTP リクエストの本文が含まれることを示します。宛先 URL に XML を送信するときに使用します。ColdFusion はこの XML データの URL エンコードを行いません。 • file: 指定のファイルのコンテンツを送信することを ColdFusion に指示します。ColdFusion はこのファイルコンテンツの URL エンコードを行いません。 • URL: cfhttp の url 属性に URL クエリー文字列の名前と値のペアを付加することを示します。ColdFusion はこのクエリー文字列の URL エンコードを行います。 • formField: 送信するフォームフィールドを指定します。ColdFusion はデフォルトでフォームフィールドの URL エンコードを行います。 • cookie: HTTP ヘッダとして送信する Cookie を指定します。ColdFusion はこの Cookie の URL エンコードを行います。
encoded	オプション	yes	FormField タイプと CGI タイプに適用されます。他のタイプでは無視されます。フォームフィールドまたはヘッダに URL エンコードを行うかどうかを指定します。
file	type="File" の場合のみ必須		File タイプに適用されます。他のタイプでは無視されます。リクエスト本文の中で送信されるファイルへの絶対パスを指定します。
mimeType	オプション		File タイプに適用されます。他のタイプでは無効です。ファイルコンテンツの MIME メディアタイプを指定します。このコンテンツタイプには、ファイルの文字エンコードの識別子を含めることができます。たとえば text/html; charset=ISO-8859-1 は、そのファイルが ISO Latin-1 文字エンコードの HTML テキストであることを示します。
name	必須です。Body タイプと XML タイプではオプションであり、無視される		渡されるデータの変数名です。Body タイプと XML タイプでは無視されます。File タイプの場合は、このリクエストで送信するファイル名を指定します。
value	必須です。File タイプではオプションであり、無視される		送信されるデータの値です。File タイプでは無視されます。type 属性が Body 以外のときは、文字列データまたは ColdFusion が文字列に変換できるデータが値に含まれる必要があります。Body タイプのときは、文字列またはバイナリ値を指定できます。

使用方法

HTTP リクエストで送信するヘッダまたは本文データを指定します。type 属性は、そのパラメータが指定する情報を示します。cfhttp タグには、次の制限の下で複数の cfhttpparam タグを含めることができます。

- XML の type 属性は、別の XML の type 属性や、body、file、または formField の type 属性とは併用できません。
- body の type 属性は、別の body の type 属性や、XML、file、または formField の type 属性とは併用できません。
- XML および body の type 属性は、cfhttp タグの TRACE メソッドでは使用できません。

- file の type 属性は、cfhttp タグの POST および PUT メソッドでのみ意味があります。
- formField の type 属性は、cfhttp タグの POST および GET メソッドでのみ意味があります。

ColdFusion ページに HTTP リクエストを送信する場合は、CGI タイプで送信されるヘッダだけでなく、すべての HTTP ヘッダを CGI スコープ変数として使用することができます。ただし、"myVar" などのカスタム変数は CGI スコープのダンプには表示されません。

type="file" 属性を使用してファイルを送信する場合、そのファイルのコンテンツは multipart/form-data リクエストの本文の中で送信されます。このファイルを ColdFusion ページに送信した場合は、受信側のページの Form スコープに、cfhttpparam タグの name 属性に指定した名前をキーとするエントリが格納されます。この変数の値には、送信したファイルを保持するテンポラリファイルへのパスが含まれています。フォームフィールドのデータも送信した場合、form.fieldnames キーリスト内でのこのファイル名の場所は、フォームデータを含む cfhttp タグに相対的なファイルを含む cfhttpparam タグの場所によって決まります。

URL エンコードにより、アンパサンドなどの特殊文字はサーバーに渡されても保持されます。詳細については、1334 ページの「URLEncodedFormat」関数を参照してください。

" 未処理 " の HTTP メッセージ内の任意のデータを送信するには、type="body" 属性を含む cfhttpparam タグを使用して本文の内容を指定し、type="header" 属性を含む cfhttpparam タグを使用してヘッダを指定します。

例

```
<!--- This example consists of two CFML pages.
        The first page posts to the second. --->

<!--- The first, posting page.
        This page posts variables to another page and displays the body of the response from
        the second page. Change the URL and port as necessary for your environment. --->

<cfhttp
        method="post"
        url="http://127.0.0.1/tests/http/cfhttpparamexample.cfm"
        port="8500"
        throwonerror="Yes">
    <cfhttpparam name="form_test" type="FormField" value="This is a form variable">
    <cfhttpparam name="url_test" type="URL" value="This is a URL variable">
    <cfhttpparam name="cgi_test" type="CGI" value="This is a CGI variable">
    <cfhttpparam name="cookie_test" type="Cookie" value="This is a cookie">
</cfhttp>

<!--- Output the results returned by the posted-to page. --->
<cfoutput>
    #cfhttp.fileContent#
</cfoutput>


<!--- This is the cfhttpparamexample.cfm page that receives and processes the Post request. Its response
body is the generated HTML output. --->

<h3>Output the passed variables</h3>
<cfoutput>
    Form variable: #form.form_test#
    <br>URL variable: #URL.url_test#
    <br>Cookie variable: #Cookie.cookie_test#
    <br>CGI variable: #CGI.cgi_test#<br>
    <br>Note that the CGI variable is URL encoded.
</cfoutput>
```

タグ i

cfif

説明

CFML で簡単な条件ステートメントや複合条件ステートメントを作成できます。式、変数、関数の戻り値、または文字列をテストします。オプションで cfelse タグと cfelseif タグを組み合わせて使用します。

カテゴリ

フロー制御タグ

シンタックス

```
<cfif expression>
    HTML and CFML tags<cfelseif expression>
    HTML and CFML tags
    <cfelse>
    HTML and CFML tags
</cfif>
```

関連項目

cfelse、cfelseif、cfabort、cfbreak、cfexecute、cfexit、cflocation、cfloop、cfswitch、cfthrow、cftry

使用方法

cfif タグ内の式の値が true の場合、それ以降のすべてのコードは、cfelseif タグまたは cfelse タグに到達するまで実行され、その後は cfif 終了タグまでスキップされます。false の場合、cfif タグの直後のコードは実行されず、cfelseif タグまたは cfelse タグが存在すればそのコードが実行され、あるいは、cfif 終了タグの後のコードが実行されます。

ブール値を返す関数の戻り値をテストするときには、true 条件を明示的に定義する必要はありません。この例では、IsArray 関数を使用しています。

```
<cfif IsArray(myarray)>
```

関数が成功した場合、IsArray は yes と評価されます。この文字列は、ブール値の true に相当します。true の条件を次のように明示的に定義する方法よりも、この方法をお勧めします。

```
<cfif IsArray(myarray) IS True>
```

このタグには終了タグが必要です。

例

この例では、変数をシャープ記号で囲んでいます。これは必須ではありません。

```
<!--- This example shows the interaction of cfif, cfelse, and cfelseif. --->
<!----- First, perform a query to get some data. ----->
<cfquery name="getCenters" datasource="cfdocexamples">
    SELECT Center_ID, Name, Address1, Address2, City, State, Country, PostalCode,
        Phone, Contact
    FROM Centers
    ORDER by City, State, Name
</cfquery>
<p>CFIF gives us the ability to perform conditional logic based on a condition
    or set of conditions.</p>
<p>For example, we can output the list of Centers from the snippets datasource
    by group and only display them <b>IF</b> City = San Diego.</p>
<hr>
<!---- Use CFIF to test a condition when outputting a query. ----->
<p>The following centers are in San Diego:</p>
<cfoutput query="getCenters">
        <cfif Trim(City) is "San Diego">
            <br><b>Name/Address:</b>#Name#, #Address1#, #City#, #State#
            <br><b>Contact:</b> #Contact#
            <br>
        </cfif>
</cfoutput>
<hr>
<p>If we would like more than one condition to be the case, we can ask for a list of the
centers in San Diego <b>OR</b> Santa Ana. If the center does not follow this condition, we
can use CFELSE to show only the names and cities of the other centers.</p>
<p>Notice how a nested CFIF is used to specify the location of the
    featured site (Santa Ana or San Diego).</p>
<!----- Use CFIF to specify a conditional choice for multiple options;
    also note the nested CFIF. --->
<p>Complete information is shown for centers in San Diego or Santa Ana.
    All other centers are listed in italic:</p>
<cfoutput query="getCenters">
    <cfif Trim(City) is "San Diego" OR Trim(City) is "Santa Ana">
        <h4>Featured Center in
            <cfif Trim(City) is "San Diego">
                San Diego
            <cfelse>
                Santa Ana
            </cfif>
            </h4> <b>Name/Address:</b>#Name#, #Address1#, #City#, #State#
            <br><b>Contact:</b> #Contact#<br>
    <cfelse>
        <br><i>#Name#, #City#</i>
    </cfif>
</cfoutput>
```

```
<hr>
<p>Finally, we can use CFELSEIF to cycle through a number of conditions and
produce varying output. Note that you can use CFCASE and CFSWITCH for a more
elegant representation of this behavior.
<!--- Use CFIF in conjunction with CFELSEIF to specify more than one
    branch in a conditional situation. --->
<cfoutput query="getCenters">
    <cfif Trim(City) is "San Diego" OR Trim(City) is "Santa Ana">
        <br><i>#Name#, #City#</i> (this one is in
                <cfif Trim(City) is "San Diego">San Diego
                <cfelse>Santa Ana
                </cfif>)
            <cfelseif Trim(City) is "San Francisco">
                <br><i>#Name#, #City#</i> (this one is in San Francisco)
            <cfelseif Trim(City) is "Suisun">
                <br><i>#Name#, #City#</i> (this one is in Suisun)
            <cfelse> <br><i>#Name#</i>
                <b>Not in a city we track</b>
            </cfif>
</cfoutput>
```

cfimage

説明

ColdFusion イメージを作成します。cfimage タグを使用すると、Image 関数へのショートカットとして一般的なイメージ操作オペレーションを実行できます。cfimage タグは、単独で使用することも、Image 関数と併用することもあります。

履歴

ColdFusion 10：cfimage action = "resize" に interpolation 属性が追加されました。

ColdFusion 8: このタグが追加されました。

カテゴリ

その他のタグ

シンタックス

```
Add a border to an image
<cfimage
    required
    action = "border"
    source = "absolute pathname|pathname relative to the web root|URL|#cfimage variable#"
    optional
    color = "hexadecimal value|web color"
    destination = "absolute pathname|pathname relative to the web root"
    isBase64 = "yes|no"
    name = "cfimage variable"
    overwrite = "yes|no"
    thickness = "number of pixels">

Create a CAPTCHA image
<cfimage
    required
    action = "captcha"
    height = "number of pixels"
    text = "text string"
    width = "number of pixels"
    optional
```

```
    destination = "absolute pathname|pathname relative to the web root"
    difficulty = "high|medium|low"
    overwrite = "yes|no"
    fonts = "comma-separated list of font names"
    fontSize = "point size">

Convert an image file format
<cfimage
    required
    action = "convert"
    destination = "absolute pathname|pathname relative to the web root"
    source = "absolute pathname|pathname relative to the web root"|URL|#cfimage variable#
    optional
    isBase64 = "yes|no"
    name = "cfimage variable"
    overwrite = "yes|no">

Retrieve information about an image
<cfimage
    required
    action = "info"
    source = "absolute pathname|pathname relative to the web root|URL|#cfimage variable#"
    structname = "structure name"
    optional
    isBase64 = "yes|no">

Read an image into memory
<cfimage
    required
    name = "cfimage variable"
    source = "absolute pathname|pathname relative to the web root|URL|#cfimage variable#"
    optional
    action = "read"
    isBase64 = "yes|no">

Resize an image
<cfimage
    required
    action = "resize"
    height = "number of pixels|percent%"
    source = "absolute pathname|pathname relative to the web root|URL|#cfimage variable#"
    width = "number of pixels|percent%"
    optional
    destination = "absolute pathname|pathname relative to the web root"
    isBase64 = "yes|no"
    name = "cfimage variable"
    overwrite = "yes|no">
    interpolation = "interpolation algorithm"

Rotate an image
<cfimage
    required
    action = "rotate"
    angle = "angle in degrees"
    source = "absolute pathname|pathname relative to the web root|URL|#cfimage variable#"
    optional
    destination = "absolute pathname|pathname relative to the web root"
    isBase64= "yes|no"
    name = "cfimage variable"
    overwrite = "yes|no">
```

```
Write an image to a file
<cfimage
    required
    action = "write"
    destination = "absolute pathname|pathname relative to the web root"
    source = "absolute or relative pathname|URL|#cfimage variable#"
    optional
    isBase64= "yes|no"
    overwrite = "yes|no"
    quality = "JPEG image quality">

Write an image to the browser
<cfimage
    required
    action = "writeToBrowser"
    source = "absolute pathname|pathname relative to the web root|URL|#cfimage variable#"
    optional
    format = "png|jpg|jpeg"
    isBase64= "yes|no">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

ImageAddBorder、ImageInfo、ImageNew、ImageRead、ImageReadBase64、ImageResize、ImageRotate、ImageWrite、ImageWriteBase64、『ColdFusion アプリケーションの開発』の Creating and Manipulating ColdFusion Images

属性

属性	アクション	必須 / オプション	デフォルト	説明
action	N/A	オプション	read	実行するアクションです。次のいずれかになります。 • border • captcha • convert • info • read • resize • rotate • write • writeToBrowser デフォルトのアクションは read です。このアクションを明示的に指定する必要はありません。
angle	rotate	必須		イメージを回転させる角度です (単位 : 度)。 この値は整数で指定します。
color	border	オプション	black	ボーダーの色です。 16 進数値またはサポートされているカラー色を指定します。「有効な HTML カラー名」にある名前のリストを参照してください。16 進数値で指定する場合は、"##xxxxxx" または "xxxxxx" という形式を使用します。ここで、x は 0 ～ 9 または A ～ F です。シャープ記号 (#) は 2 つ使用するか、または使用しないでください。
destination	border captcha convert resize rotate write	オプション (「説明」を参照)		イメージの出力先の絶対パス名または相対パス名です。イメージの形式は、ファイル拡張子によって決定されます。 convert および write アクションの場合、destination 属性は必須です。border、captcha、resize、および rotate アクションの場合は、name 属性または destination 属性が必須になります。両方を指定することもできます。ColdFusion では、PNG 形式の CAPTCHA イメージのみがサポートされています。 CAPTCHA イメージが配置される場所は、次のように決まります。 • destination を指定した場合、イメージは、（指定した絶対パス名または相対パス名に基づいて）ファイルに書き込まれます。 • name を指定した場合、イメージはイメージ変数に書き込まれます。 • destination と name のどちらも指定しない場合、イメージはブラウザーに書き込まれます。
difficulty	captcha	オプション	low	CAPTCHA テキストの複雑度のレベルです。テキストの歪みのレベルを次のいずれかの値で指定します。 • low • medium • high

属性	アクション	必須 / オプション	デフォルト	説明
fonts	captcha	オプション		CAPTCHA テキストで使用する有効なフォントです。複数のフォントを指定する場合は、カンマで区切ります。 指定されたフォントを使用できない場合は、システムフォントが代用されます。
fontSize	captcha	オプション	24	CAPTCHA イメージ内のテキストのフォントサイズです。 この値は整数で指定する必要があります。
format	writeToBrowser	オプション	PNG	ブラウザに表示するイメージの形式です。形式を指定しない場合、イメージは PNG 形式で表示されます。 ブラウザで GIF イメージを表示することはできません。GIF イメージは PNG 形式で表示されます。
height	captcha resize	必須		イメージの高さです (単位 : ピクセル)。 resize の場合は、高さをパーセントで指定することもできます (整数値の後にパーセント (%) 記号を付けます)。 イメージのサイズを変更する場合は、幅の値を指定して高さを "" に指定すると、ColdFusion で縦横比が計算されます。 指定する場合、値は整数で指定する必要があります。
interpolation	resize	オプション	highestQuality	名前（hamming など）、イメージ品質（mediumQuality など）、またはパフォーマンス（highestPerformance など）を基準に特定の補間アルゴリズムを指定します。有効な値は次のとおりです。 • highestQuality • highQuality • mediumQuality • highestPerformance • highPerformance • mediumPerformance • nearest • bilinear • bicubic • bessel • blackman • hamming • hanning • hermite • lanczos • mitchell • quadratic

属性	アクション	必須 / オプション	デフォルト	説明
isBase64	border convert info read resize rotate write writeToBrowser	オプション	no	ソースが Base64 文字列であるかどうかを指定します。 • yes: ソースは Base64 文字列です。 • no: ソースは Base64 文字列ではありません。
name	border convert read resize rotate	オプション (「説明」を参照)		作成する ColdFusion イメージ変数の名前です。 read アクションの場合、name 属性は必須です。 border、resize、および rotate アクションの場合は、name 属性または destination 属性が必須になります。両方を指定することもできます。
overwrite	border captcha convert read resize rotate write	オプション	no	destination 属性が指定されている場合にのみ有効です。overwrite の値は次のとおりです。 • yes: 書き込み先のファイルを上書きします。 • no: 書き込み先のファイルを上書きしません。 書き込み先のファイルが既に存在する場合は、overwrite アクションが yes に設定されていないとエラーが発生します。
quality	write	オプション	0.75	JPEG 形式の書き込み先ファイルの品質です。拡張子が JPG または JPEG のファイルに対してのみ適用されます。有効な値は、0 ～ 1 の間の小数です (値が小さいほど品質は低くなります)。
source	border convert info read resize rotate write writeToBrowser	必須		• ソースイメージの URL (例 : "http://www.google.com/images/logo.gif") • 絶対パス名または Web ルートからの相対パス名 (例 : "c:¥images¥logo.jpg") • 別のイメージ、BLOB、またはバイト配列を含む ColdFusion イメージ変数 (例 : "#myImage#") • Base64 文字列 (例 : ".....")
structName	info	必須		作成する ColdFusion 構造体の名前です。

属性	アクション	必須 / オプション	デフォルト	説明
text	captcha	必須		CAPTCHA イメージ内に表示するテキスト文字列です。読みやすいように大文字を使用してください。CAPTCHA イメージ内では見分けられないので、スペースは使用しないでください。
thickness	border	オプション	1	ボーダーの幅です (単位 : ピクセル)。ボーダーはソースイメージの外縁に追加されるため、ボーダーの幅に応じてイメージ領域が広くなります。 この値は整数で指定する必要があります。
width	captcha resize	必須		イメージの幅です (単位 : ピクセル)。 resize の場合は、幅をパーセントで指定することもできます (整数値の後に % 記号を付けます)。 イメージのサイズを変更する場合は、高さの値を指定して幅を "" に指定すると、ColdFusion で縦横比が計算されます。 指定する場合、値は整数で指定する必要があります。

使用方法

ColdFusion には、cfimage タグに加え、ColdFusion イメージと呼ばれる概念があります。ColdFusion イメージとは、イメージデータを含む ColdFusion 固有の構造です。メモリ内で ColdFusion イメージを操作して、ファイル、データベース、ブラウザなどに書き込むことができます。既存のイメージファイルから ColdFusion イメージを作成して、簡単なイメージアクション (回転やサイズ変更など) を実行するには、cfimage タグを使用します。また、ImageNew 関数を使用すると、何もない状態から ColdFusion イメージを作成することも、既存のイメージから ColdFusion イメージを作成することもできます。Image 関数を使用すると、cfimage タグまたは ImageNew 関数で作成した ColdFusion イメージに対して複雑なイメージ操作オペレーションを実行できます。

ColdFusion イメージに対しては、次のようなタスクを実行できます。

- イメージのファイル形式を変換する。たとえば、BMP ファイルを JPEG ファイルに変換したり、Base64 文字列を GIF に変換することができます。
- サーバーにアップロードするファイルのサイズを統一する。
- JPEG イメージの品質を変更してサイズを制限する。
- ColdFusion イメージをファイルに保存する。または、ブラウザに直接出力する。
- cfquery タグ内で ImageGetBlob 関数を使用して、ColdFusion イメージを BLOB (Binary Large Object Bitmap) としてデータベースに挿入する。データベースから BLOB を抽出して ColdFusion イメージを生成することもできます。
- 透かしイメージを作成する。
- サムネールイメージを作成する。
- スパムを防止するために CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart) イメージを作成する。CAPTCHA イメージとは、機械では読み取れないように (ただし、人間には読み取れるように) 歪められたテキストを含むイメージであり、本人確認などに使用されます。

詳細な例については、『ColdFusion アプリケーションの開発』の Creating and Manipulating ColdFusion Images を参照してください。

ファイル属性

次のシンタックスを使用して、destination 属性および source 属性で指定されたディスクに書き込まれない、メモリ内のファイルを指定します。メモリ内のファイルを使用すると、一時的データの処理が速くなります。

```
ram:///filepath
```

ファイルパスには、ram:///petStore/images/poodle.jpg のようにディレクトリを含めることができます。ファイルを指定する前に、パスに含まれるディレクトリを作成しておく必要があります。メモリ内のファイルの使用方法について詳しくは、『ColdFusion アプリケーションの開発』の Working with in-memory files を参照してください。

サポートされるイメージファイル形式

cfimage タグは、さまざまなファイル形式をサポートします。ColdFusion アプリケーションがデプロイされているサーバー上でサポートされている形式のリストを取得するには、GetReadableImageFormats 関数および GetWriteableImageFormats 関数を使用します。

Macintosh、Windows、および Unix オペレーティングシステムにおいて ColdFusion がサポートするデフォルトのイメージ形式は次のとおりです。

- JPEG
- GIF
- TIFF
- PNG
- BMP

次のイメージ形式はサポートされていません。

- アニメーション GIF
- マルチページ TIFF
- PSD
- AI

CMYK サポート

cfimage タグでは、CMYK イメージの読み込みと書き込みがサポートされていますが、イメージの変換が必要なアクションはサポートされていません。たとえば、CMYK イメージは、read、write、writeToBrowser、resize、rotate、および info アクションで使用できます。一方、CMYK イメージは、convert、captcha、および border アクションでは使用できません。同じルールがイメージ関数にも適用されます。たとえば、ImageNew、ImageRead、および ImageWrite 関数では CMYK イメージがサポートされますが、ImageAddBorder 関数ではサポートされません。

有効な HTML カラー名

次の表に、color 属性で使用できる W3C HTML 4 カラー名または 16 進値を示します。

カラー名	RGB 値
Black	##000000
Blue	##0000FF
Red	##FF0000
Gray	##808080
LightGray	##D3D3D3
DarkGray	##A9A9A9
Green	##008000
Pink	##FFC0CB
Cyan	##00FFFF

カラー名	RGB 値
Magenta	##FF00FF
Orange	##FFA500
White	##FFFFFF
Yellow	##FFFF00

その他の色を指定する場合は、16 進数値を入力します。RGB 値を示す 6 桁の値を指定します。00 ～ FF の範囲の値を使用できます。

イメージの品質

デフォルトでは、cfimage タグで生成されるイメージにはアンチエイリアス処理が適用されます (輪郭のジャギーが除去されます)。また、補間方式は highestQuality に設定されます。この方式では高品質のイメージが生成されますが、処理速度は低下します。アンチエイリアスをオフにするには、ImageSetAntialiasing 関数を使用します。補間方式の変更や、イメージ属性の制御には、次の関数を使用します。

- ImageResize
- ImageRotate
- ImageScaleToFit
- ImageShear
- ImageTranslate

border アクション イメージの外縁に矩形のボーダーを作成するには、border アクションを使用します。このアクションでは、ボーダーの幅と色を制御できます。その他の属性を制御するには、ImageAddBorder 関数を使用します。次の例では、ボーダーの幅と色を設定します。

```
<!--- This example shows how to create a ColdFusion image from an existing JPEG file, add a
    five-pixel-wide red border to the image, and save it to a new JPEG file. --->
<cfimage source="../cfdocs/images/artgallery/jeff05.jpg" action="border" thickness="5"
    destination="jeff05.jpg" color="red" overwrite="yes">
```

captcha アクション 機械では読み取れないように (ただし、人間には読み取れるように) 歪められたテキストイメージを作成するには、captcha アクションを使用します。CAPTCHA イメージを作成するには、CAPTCHA イメージで表示するテキストを指定します。このテキストが ColdFusion によってランダムに変形されます。このアクションでは、テキスト領域の高さと幅を指定できます。この設定は、文字の間隔、フォントサイズ、CAPTCHA テキストで使用するフォント、難読度に影響するため、テキストの読みやすさを左右します。次の例は、CAPTCHA イメージをブラウザに直接出力する方法を示しています。

```
<cfimage action="captcha" fontSize="25" width="400" height="150" text="rEadMe"
    fonts="Arial,Verdana,Courier New">
```

注意：CAPTCHA イメージを表示するには、width を、"fontSize × text で指定した文字の数 × 1.08" より大きい値に設定する必要があります。この例の場合、width の最小値は 162 です。

ColdFusion では、PNG 形式の CAPTCHA イメージのみがサポートされています。

注意：複数のユーザーが CAPTCHA イメージにアクセスした場合にファイルが上書きされないようにするには、CAPTCHA イメージファイルに一意の名前を付けます。

次の例は、difficulty を medium に設定して CAPTCHA イメージを作成し、ファイルに書き込む方法を示しています。

```
<!--- Use the GetTickCount function to generate unique names for the CAPTCHA files. --->
<cfset tc = GetTickCount()>
<cfimage action="captcha" fontSize="15" width="180" height="50" text="rEadMe"
    destination="images/rEadMe#tc#.png" difficulty="medium">
```

詳細な例については、『ColdFusion アプリケーションの開発』の Creating and Manipulating ColdFusion Images を参照してください。

convert アクション イメージのファイル形式を変換するには、convert アクションを使用します。ファイル形式の詳細については、「サポートされるイメージファイル形式」を参照してください。次の例は、JPEG ファイルを PNG ファイルに変換する方法を示しています。

```
<!--- This example shows how to convert a JPEG image to a PNG image. --->
<cfimage source="../cfdocs/images/artgallery/aiden02.jpg" action="convert"
    destination="aiden02.png">
```

注意：イメージのファイル形式を変換する作業には時間がかかります。また、イメージの品質が低下する場合もあります。たとえば、PNG イメージでは 24 ビットの色がサポートされていますが、GIF イメージでは 256 色しかサポートされていません。透明のイメージ (アルファ値を持つイメージ) を変換すると、品質が低下する場合があります。

info アクション イメージのカラーモデル、高さ、幅、ソースなど、イメージに関する情報を格納する ColdFusion 構造体を作成するには、info アクションを使用します。この構造体は、ImageInfo 関数が返す構造体と同じものです。次の例は、イメージに関するすべての情報を取得する方法を示しています。

```
<!--- This example shows how to retrieve and display image information. --->
<cfimage source="../cfdocs/images/artgallery/viata03.jpg" action="info" structName="viatoInfo">
<cfdump var="#viatoInfo#">

<!--- Alternatively, you can use the cfoutput tag to display specific image information, as
    shown in the following example. --->
<cfoutput>
    <p>height: #viatoInfo.height# pixels</p>
    <p>width: #viatoInfo.width# pixels</p>
    <p>source: #viatoInfo.source#</p>
    <p>transparency: #viatoInfo.colormodel.transparency#</p>
    <p>pixel size: #viatoInfo.colormodel.pixel_size#</p>
    <p>color model: #viatoInfo.colormodel.colormodel_type#</p>
    <p>alpha channel support: #viatoInfo.colormodel.alpha_channel_support#</p>
    <p>color space: #viatoInfo.colormodel.colorspace#</p>
</cfoutput>
```

read アクション 指定したローカルのファイルパス名または URL からイメージを読み込んで、メモリ内に ColdFusion イメージを作成するには、read アクションを使用します。ColdFusion イメージ変数は、別の cfimage タグや Image 関数のソースとして使用できます。read アクションは、ImageRead 関数と同じオペレーションを実行します。次の例は、JPEG ファイルから ColdFusion イメージを作成し、ImageGrayscale 関数を使用してイメージを操作する方法を示しています。

```
<!--- This code shows how to create a ColdFusion image from a JPEG file.
    --->
<cfimage source="../cfdocs/images/artgallery/jeff01.jpg" name="myImage">
<!--- This code shows how to convert the image to grayscale. --->
<cfset ImageGrayscale(myImage)>
<!--- This code shows how to write the grayscale image to a JPEG file. --->
<cfimage source="#myImage#" action="write" destination="myGrayscaleImage.jpg" overwrite="yes">
```

resize アクション イメージの高さと幅を変更するには、resize アクションを使用します。高さと幅を設定するには、ピクセル数またはパーセント値を指定します。

```
<!--- This example shows how to specify the height and width of an image in pixels. --->
<cfimage source="../cfdocs/images/artgallery/jeff01.jpg" action="resize" width="100" height="100"
destination="jeff01_thumbnail.jpg" overwrite="yes">
<!--- This example shows how to specify the height and width of an image as percentages. --->
<cfimage source="../cfdocs/images/artgallery/jeff02.jpg" action="resize"
    width="50%" height="50%" destination="jeff02_thumbnail.jpg" overwrite="yes">
<!--- This example shows how to specify the height of an image in pixels and its width as a
    percentage. --->
<cfimage source="../cfdocs/images/artgallery/jeff03.jpg" action="resize"
    width="50%" height="100" destination="jeff03_thumbnail.jpg" overwrite="yes">
```

サイズ変更に関するその他の属性を制御するには、ImageResize 関数を使用します。

rotate アクション イメージを指定した角度だけ回転させるには、rotate アクションを使用します。

```
<!--- This example shows how to rotate an image by 30 degrees. --->
<cfimage source="../cfdocs/images/artgallery/maxwell01.jpg" action="rotate" angle="30"
    name="maxwellAngle">
<!--- Display the rotated image in a browser. --->
<cfimage source="#maxwellAngle#" action="writeToBrowser">
```

回転に関するその他の属性を制御するには、ImageRotate 関数を使用します。

write アクション 指定したパスにイメージを書き込むには、write アクションを使用します。新しいイメージは、destination 属性で指定したファイルタイプに変換されます。write アクションは、ImageWrite 関数と同じオペレーションを実行します。イメージを JPEG ファイルに書き込む場合のデフォルトのイメージ品質は、元のイメージの 75% に設定されます。イメージのサイズを制御するには、write アクションの quality 属性を使用します。

write アクションを使用して JPEG イメージの品質を変更すると、ファイルサイズを小さくすることができます。次の例は、イメージの品質を 0.5 に変更する方法を示しています。

```
<!--- This example shows how to create a PNG file from a JPEG file by using the write action. --->
<cfimage source="../cfdocs/images/artgallery/aiden01.jpg" action="write"
    destination="aiden01.png">
<!--- This example shows how to create a low-quality JPEG image. --->
<cfimage source="../cfdocs/images/artgallery/jeff05.jpg" action="write"
    destination="jeff05_lq.jpg" quality=".5">
<!--- This example shows how to write a JPEG file to a new location. --->
<cfimage source="../cfdocs/images/artgallery/jeff05.jpg" action="write"
    destination="jeff05.jpg">
```

writeToBrowser アクション ColdFusion イメージをファイルに書き込まずに直接ブラウザに表示するには、writeToBrowser アクションを使用します。イメージは PNG 形式で表示されます。次の例は、イメージのサイズを小さくしてからブラウザにイメージを表示する方法を示しています。

```
<!--- This example shows how to create a ColdFusion image from a JPEG file, resize it, and
    then display it in the browser as a PNG image. --->
<cfimage source="../cfdocs/images/artgallery/jeff05.jpg" action="resize"
    width="50%" height="50%" name="smLogo">
<cfimage source="#smLogo#" action="writeToBrowser">
```

例

次の例は、ColdFusion イメージを作成し、Image 関数を使用してイメージを操作する方法を示しています。

```
<!--- Create the ColdFusion image variable "myImage" from a JPEG file. --->
<cfimage source="../cfdocs/images/artgallery/jeff05.jpg" name="myImage">
<!--- Pass the ColdFusion image to the Image functions to blur the image by a radius of 5,
    flip the image 90 degrees, and convert the image to grayscale. --->
<cfset ImageBlur(myImage,5)>
<cfset ImageFlip(myImage,"90")>
<cfset ImageGrayscale(myImage)>
<!--- Write the transformed image to a browser. --->
<cfimage source="#myImage#" action="writeToBrowser">
```

cfimap

説明
複数のフォルダ内のメールを取得および管理するように IMAP サーバーに問い合わせます。

カテゴリ
通信タグ、インターネットプロトコルタグ

シンタックス
```
<cfimap
  action ="DELETE|DELETEFOLDER|CREATEFOLDER|OPEN|CLOSE|RENAMEFOLDER|LISTALLFOLDERS|
MARKREAD|MOVEMAIL|GETALL|GETHEADERONLY"
  attachmentpath = "string"
  connection = "string"
  folder = "string"
  generateuniquefilenames = "yes|no"
  maxrows = "integer"
  messagenumber ="integer"
  name = "string"
  newfolder = "string"
  password = "string"
  port = "integer"
  recurse = "true|false"
  secure = "yes|no"
  server = "IMAP server address"
  startrow = "integer"
  secure = "yes|no"
  stoponerror = "true|false"
  uid = "integer or comma-delimited list of integers"
  username = "SMTP user ID">
```

関連項目
cfmailparam、cfmailparam、cfmailpart、cfpop、cfftp、cfhttp、cfldap、Wrap、『ColdFusion アプリケーションの開発』の Sending and Receiving E-Mail の Using ColdFusion with mail servers

履歴
ColdFusion 9 で新規のタグが導入されました。

属性

属性	必須 / オプション	デフォルト	説明
action	オプション	GetHeaderOnly	取得したすべてのメールのメッセージヘッダ情報を返します。この属性で指定できるその他の値は次のとおりです。 GetAll: メールを返します。情報には、メッセージヘッダ情報、メッセージテキスト、および添付ファイルが含まれます。添付ファイルを取得するには、AttachmentPath 属性を設定します。 Delete: フォルダからメッセージを削除します。 Open: IMAP サーバーとのセッションまたは接続を開始します。 Close: IMAP サーバーとのセッションまたは接続を終了します。 MarkRead: フォルダから読み込んだメッセージすべてにマークを付けます。 CreateFolder: メールボックス内にフォルダを作成します。 DeleteFolder: メールボックス内のフォルダを削除します。 RenameFolder: 既存のユーザー定義フォルダの名前を変更します。 ListAllFolders: メールボックス内または Folder 属性で定義したフォルダ名の下にある既存のすべてのフォルダのリストを表示します。 MoveMail: フォルダ間でメールを移動します。
attachmentpath	GetAll アクションの場合はオプション		ColdFusion が添付ファイルを取得するフォルダの名前を指定します。このフォルダが存在しない場合は、自動的に作成されます。フォルダを指定しない場合、添付ファイルは保存されません。 メモ : 相対パスを指定した場合、添付ファイルはテンポラリフォルダに保存されます。このフォルダは、GetTempDirectory 関数を使用した場合に返されるフォルダと同じフォルダになります。
Connection	Open および Close アクションの場合は必須		接続およびセッションのための変数を指定します。サーバー属性に無効な IP アドレスまたは無効なドメイン名が指定されている場合、接続は失敗してエラーメッセージが返されます。
Folder	DeleteFolder、CreateFolder、および RenameFolder のアクションの場合は必須 LISTALLFOLDERS、MOVEMAIL、MARKREAD、GETALL、GETHEADERONLY、および DELETE の場合はオプション	INBOX (ListAllFolders アクションの場合、デフォルトのフォルダは mailbox です。)	メール関連のアクションの場合 : メッセージを取得、移動、または削除するフォルダの名前を指定します。フォルダ名が無効の場合、デフォルトでは INBOX となります。 フォルダのアクションの場合 : 削除 (DeleteFolder)、作成 (CreateFolder)、または名前変更 (RenameFolder) されたフォルダの名前を指定します。 サブフォルダを選択する場合は、必要に応じてピリオド (.) 文字を使用します。たとえば、フォルダ 'Product.Version.Module' のフォルダ 'Module' 内のメールを削除する場合は、Folder 属性を 'Product.Version.Module' と定義します。
GenerateUniqueFilenames	オプション		各添付ファイルに対して、固有のファイル名が生成されるようにします。これにより、同じファイル名の添付ファイルで名前の競合が発生しなくなります。
MaxRows	オプション		既読としてマーク、削除、またはフォルダ間で移動する行数を指定します。値が 1 の場合、StartRow で決定された行が示されます。増分値がある場合は、StartRow から開始される行にマークが付けられます。UID または MessageNumber 属性を指定した場合、MaxRows は無視されます。

属性	必須 / オプション	デフォルト	説明
MessageNumber	オプション		取得、削除、既読としてマーク、またはメールを移動するメッセージのメッセージ番号、またはメッセージ番号のカンマ区切りのリストを指定します。 無効なメッセージ番号を設定した場合、IMAP オペレーションは無視されます。たとえば、存在しないメッセージ番号を削除するように cfimap で指定した場合、そのオペレーションは無視されます。 UID 属性を指定した場合、MessageNumber 属性は無視されます。
Name	オプション GetAll、GetHeaderOnly、および ListAllFolders のアクションの場合は必須		取得したメッセージの情報が含まれるクエリーオブジェクトの名前を指定します。
NewFolder	オプション RenameFolder、MoveMail のアクションの場合は必須		フォルダ名を変更する場合は新しいフォルダ名を指定し、すべてのメールを移動する場合は移動先フォルダの名前を指定します。
Password	オプション action="open" の場合は必須、ユーザー名とパスワードの指定が必要		ユーザーの電子メールアカウントにアクセスするためのパスワードを指定します。
Port	オプション	143 または 993	IMAP ポート番号を指定します。セキュリティ保護されていない接続の場合は 143、保護されている接続の場合は 993 を使用します。
Recurse	オプション	False	ColdFusion がサブフォルダ内で CFIMAP コマンドを実行するかどうかを指定します。 Recurse は action="ListAllFolders" の場合に機能します。recurse が true に設定されている場合、すべてのフォルダおよびサブフォルダが解析され、フォルダまたはサブフォルダの名前とメール情報が返されます。
Secure	オプション	False	IMAP サーバーで Secure Sockets Layer を使用するかどうかを指定します。
Server	オプション Open アクションの場合は必須		IMAP サーバー識別子を指定します。ホスト名または IP アドレスを IMAP サーバー識別子として割り当てることができます。
StartRow	オプション		読み取りまたは削除対象となる最初の行番号を定義します。UID または MessageNumber 属性を指定した場合、StartRow は無視されます。また、移動するメールについて StartRow を指定することもできます。
StopOnError	オプション	True	このオペレーションに関する例外を無視するかどうかを指定します。値が true の場合、処理が停止されて該当するエラーが表示されます。
Timeout	オプション	60	IMAP サーバーへの接続がタイムアウトになるまで待機する秒数を指定します。タイムアウトが発生すると、エラーメッセージが表示されます。
Uid	オプション		取得、削除、または移動対象の一意の ID または Uid のカンマ区切りリストを指定します。無効な Uid を指定した場合は無視されます。
Username	オプション		ユーザー名を指定します。通常、ユーザー名は電子メールログインと同じになります。

使用方法

IMAP サーバーとのセッションまたは接続を確立します。セッションを開くには、サーバー、ユーザー名、およびパスワード属性を定義します。IMAP サーバーとの接続を開始するには、サーバー、ユーザー名、パスワード、および接続属性の値を指定します。セキュリティ保護された接続を使用する場合は、secure="true" を指定します。後続の CFIMAP タグでは、サーバー、ユーザー名、パスワード属性を指定しなくても接続属性を再利用できます。接続を確立した後、次のアクションを実行できます。

- メールの取得 : GetHeaderOnly または GetAll 属性を使用してメールを取得し、クエリーオブジェクトに情報を格納します。cfdump コマンドを使用して、クエリーオブジェクトの内容を表示します。また、AttachmentPath 属性で定義されているとおり、添付ファイルを ColdFusion テンポラリフォルダまたは新規フォルダにダウンロードできます。
- 不要なメールを削除するか、フォルダを削除します。ユーザー作成のフォルダを削除できます。受信トレイ、送信トレイ、送信済みなどの標準フォルダは削除できません。
- 複数のメールを既読としてマークを付けます。
- フォルダの作成、フォルダ名の変更、またはフォルダ間でのメールの移動によって、メールフォルダを管理します。サブフォルダを使用する場合は、ピリオド (.) を使用して正確なパスを指定します。

すべてのアクションを実行した後、セッションを閉じるか、または IMAP サーバーとの接続を閉じます。たとえば、Gmail IMAP サーバーとの接続を設定し、Gmail の電子メールアカウントのメールを取得できます。ログイン (ユーザー名) を定義し、セキュリティ保護された接続を設定できます。次に、GetHeaderOnly 属性を使用して電子メールのトップレベルスナップショットをすばやく取得するか、または GetAll 属性を使用して電子メールに関する完全な情報にアクセスできます。

注意 : Gmail では完全には IMAP が実装されていないので、標準的な IMAP サーバーの一部の機能が Gmail では動作しない場合があります。

次の表に、さまざまな cfimap 属性によって返されるクエリー情報 (列名) を示します。

"action" 属性の値	**列**
GetHeaderOnly	ANSWERED、CC、DELETED、DRAFT、FLAGGED、FROM、HEADER、LINES、MESSAGEID、MESSAGENUMBER、RECENT、REPLYTO、RXDDATE、SEEN、SENTDATE、SIZE、SUBJECT、TO、UID
GetAll	ANSWERED、ATTACHMENTFILES、ATTACHMENTS、BODY、CC、CIDS、DELETED、DRAFT、FLAGGED、FROM、HEADER、HTMLBODY、LINES、MESSAGEID、MESSAGENUMBER、RECENT、REPLYTO、RXDDATE、SEEN、SENTDATE、SIZE、SUBJECT、TEXTBODY、TO、UID CID は、CFIMAP タグが取得する、HTML 電子メール内のイメージの正しい場所を検出する場合に使用されます。電子メールメッセージに複数のイメージが埋め込まれている場合は、最後に埋め込まれたイメージのみを使用できます。
ListAllFolders	FULLNAME (ディレクトリ構造全体を指定します)、NAME、NEW、TOTALMESSAGES、および UNREAD

注意 : cfimap コマンドは IMAP4 revision1 上での動作が最適です。IMAP4 revision1 には、IMAP2 および IMAP2bis バージョンとの下位互換性があります。以前のバージョンは積極的には使用されなくなっています。

次のシナリオでは、エラーが発生する可能性があります。

- 無効なサーバー接続へのアクセスが確立されている場合。ネットワーク状態と、適切なサーバー IP アドレスおよびドメイン名を使用しているかどうかをチェックします。有効な電子メールユーザー名およびパスワードを使用してください。
- 存在しないフォルダにアクセスしている場合。アクセス先のフォルダが存在しているかどうかを確認します。有効な名前のフォルダを作成するか、名前を変更します。コアフォルダの名前は変更できません。既存のフォルダ内でメールを移動させてください。

- ネットワークが遅い場合。timeout 属性の値を大きくする必要があるかどうかを確認します。CreateFolder などのアクションの場合は、実行に要する時間がより長くなることがあります。このような場合は、timeout 属性の値を調整します。
- cfimap の属性を正しく使用していない場合。適切な属性を使用しているかどうかを確認します。たとえば、フォルダ内に 15 件の電子メールがあり、startrow または maxrow 属性の値が 18 の場合は、エラーが返されます。
- 電子メールクライアントでは、IMAP アクセスが認識されません。IMAP アクセスを許可するように電子メールがセットアップされているかどうかを確認します。電子メールクライアントの接続文字列セクション内の必要な IMAP アクセスを完成させます。
- 属性のシンタックスを正しく使用していない場合。すべての属性がシンタックスに従って定義されていることを確認します。

例 : 1

```
<!--- Retrieving e-mails from a folder --->
<html >
    <head>
    <title>IMAP Mail Client</title>
    </head>
    <body>
    <!--- Replace your username and password with valid IMAP email account name and password. Replace "server
address" with your IMAP server address--->
    <cfimap
        server = "server address" <!------>
        username = "yourname"
        action="open"
        secure="yes"
        password = "yourpassword"
        connection = "test.cf.gmail">
    <!--- Retrieve header information from the mailbox. --->
    <cfimap
        action="getHeaderOnly"
        connection="test.cf.gmail"
        name="queryname">
    <cfdump var="#queryname#">
    <cfimap
        action="close"
        connection = "test.cf.gmail">
    </body>
</html>
```

例 : 2

```
<!--- Create a folder; copy mail from Inbox and list all folder info--->
<html >
    <head>
    <title>IMAP Mail Client</title>
    </head>
<body>
    <!--- Replace yourname and yourpassword with valid IMAP email account name and password. Replace "server
address" with your IMAP server address--->
    <cfimap
        server = "server address"
        username = "yourname"
        action="open"
        secure="yes"
        password = "yourpassword"
        connection = "test.cf.gmail">
    <!--- Create a new folder, named Folder1 --->
    <cfimap
        action="CreateFolder"
        folder="Folder1"
        connection="test.cf.gmail">
    <!--- Move first 2 mails from INBOX to Folder1. --->
    <cfimap
        action="MoveMail"
        newfolder="Folder1"
        messagenumber="1,2"
        stoponerror="true"
        connection="test.cf.gmail">
    <!--- List all folders and get the information in a query. --->
    <cfimap action="listallfolders"
        connection="test.cf.gmail"
        name="queryname">
    <!--- Display query containing all folders information. --->
    <cfdump var="#queryname#">
    <cfimap
        action="close"
        connection = "test.cf.gmail">
</body>
</html>
```

例 : 3

```
<!--- Use form-based entry to access cfimap mail account. Save the form with name login.cfm. --->
<html>
<head>
<title>IMAP Mail Client</title>
</head>
<body>
<cfif IsDefined("form.server")>
    <!--- Make sure server, username are not empty. --->
    <cfif form.server is not "" and form.username is not "">
        <cfimap
           server = "#form.server#"
           username = "#form.username#"
           action="open"
           Secure="yes"
           password = "#form.pwd#"
           connection = "#form.server#">
        <cfimap
           action="ListAllFolders"
           connection="#form.server#"
           name="queryname">
```

```
        <h3>Folders in your Inbox <cfoutput>#form.username#</cfoutput></h3>
        <ol>
            <cfoutput query = "queryname">
                <li>#NAME# - #TOTALMESSAGES# <b>(#UNREAD#)</b></li>
            </cfoutput>
        </ol>
        <!---<cfdump var="#queryname#">--->
        <cfimap  action="close"   connection = "#form.server#">
    </cfif>
    <cfelse>
    <form action = "login.cfm " method = "post">
        <table>
            <tr>
            <td>Enter IMAP mail server</td><td><input type = "Text" name = "server"></td>
            </tr>
            <tr>
            <td>Enter your username</td><td><input type = "Text" name = "username"></td>
            </tr>
            <tr>
            <td>Enter your password</td><td><input type = "password" name = "pwd"></td>
            </tr>
            <tr>
            <td colspan="2"><input type="Submit" value="Get Folder List" name="getFolderList"></td>
            </tr>
        </table>
    </form>
</cfif>
</body>
</html>
```

cfimpersonate

説明

このタグは廃止になりました。新しいセキュリティツールを使用してください。詳細については、711 ページの「変換関数」および『ColdFusion アプリケーションの開発』の Securing Applications を参照してください。

履歴

ColdFusion MX: このタグは廃止されました。ColdFusion MX およびこれ以降のリリースでは機能しません。

cfimport

説明

cfimport タグを使用すると、次のいずれかをインポートすることができます。

- カスタムタグライブラリとして、ディレクトリ内のすべての ColdFusion ページ。
- Java Server Page (JSP) タグライブラリ。JSP タグライブラリは、JSP 1.1 タグ拡張 API に準拠しているタグハンドラのパッケージセットです。

カテゴリ

アプリケーションフレームワークタグ

シンタックス

```
<cfimport
    prefix = "custom"
    taglib = "tag library location">
```

関連項目

cfapplication

履歴

ColdFusion MX: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
taglib	必須		タグライブラリの URI です。このパスは、Web ルートを基準とする相対パス (/ で始まる) か、現在のページの場所、または Administrator ColdFusion マッピングページ内で指定したディレクトリを基準とする相対パスである必要があります。 • カスタム ColdFusion タグが保管されているディレクトリ。この場合は、このディレクトリ内のすべての cfm ページがタグライブラリ内のカスタムタグとして扱われます。 • Web アプリケーションの JAR へのパス (例 : "/WEB-INF/lib/sometags.jar")。 • タグライブラリディスクリプタへのパス (例 : "/sometags.tld")。 **メモ :** JSP カスタムタグライブラリは /WEB-INF/lib ディレクトリ内に置く必要があります。この制限は ColdFusion ページには適用されません。
prefix	必須		インポートされたカスタム JSP タグにアクセスするための接頭辞です。 CFML カスタムタグディレクトリをインポートするときに、この属性に空の値 "" を指定した場合は、そのカスタムタグを接頭辞なしで呼び出すことができます。JSP タグライブラリについては、接頭辞を指定して使用する必要があります。

使用方法

次の例では、"myCustomTags" ディレクトリからタグをインポートしています。

```
<cfimport
    prefix="mytags"
    taglib="myCustomTags">
```

1 つの接頭辞で複数のタグライブラリをインポートできます。ライブラリ内に重複タグがある場合は、1 番目のタグが優先されます。

JSP タグには固定の属性があります。ただし、タグが実行時属性式をサポートしている場合は、ほとんどのタグライブラリで #expressions# というシンタックスを使用できます。

CFML ページ内で JSP タグを参照する場合は、<prefix:tagname> というシンタックスを使用します。この prefix 属性には接頭辞の値を設定します。

ColdFusion ページでカスタム JSP タグを使用するには :

1. myjsptags.jar などの JSP タグライブラリの JAR ファイルを、ColdFusion サーバーのディレクトリ wwwroot/WEB-INF/lib に置きます。タグライブラリが独立した TLD ファイルを持っている場合は、TLD ファイルを JAR ファイルと同じディレクトリに配置します。
2. CFML ページの先頭に次のようなコードを挿入します。

```
   <cfimport
      prefix="mytags"
      taglib="/WEB-INF/lib/myjsptags.jar">
```

JAR ファイルから JSP タグを参照するには、次のシンタックスを使用します。

```
<cfoutput>
    <mytags:helloTag message="#mymessage#" />
<cfoutput>
```

cfimport タグは、インポートされたタグを使用するページ上に存在する必要があります。たとえば、cfinclude 呼び出しでインクルードするページ上に cfimport タグを記述した場合は、cfinclude タグを含んでいるページ上ではインポートされたタグを使用できません。同様に、Application.cfm ページ上に cfimport タグを記述した場合、インポートされたタグを使用できるのは Application.cfm ページだけです。アプリケーション内のその他のページでは使用できません。この状況でエラーが返されることはありませんが、インポートされたタグは機能しません。

cfimport タグを使用してタグライブラリからの出力を抑制することはできません。

詳細については、JSP 1.1 の仕様書を参照してください。

例

```
<h3>cfimport example</h3>
<p>This example uses the random JSP tag library that is available from the
    Jakarta Taglibs project, at http://jakarta.apache.org/taglibs/</p>

<cfimport taglib="/WEB-INF/lib/taglibs-random.jar" prefix="randomnum">

<randomnum:number id="randPass" range="000000-999999" algorithm="SHA1PRNG" provider="SUN"/>
<cfset myPassword = randPass.random>
<cfoutput>
    Your password is #myPassword#<br>
</cfoutput>
```

cfinclude

説明

CFML 内に ColdFusion ページへの参照を埋め込みます。cfinclude タグは再帰的に埋め込むことができます。CFML をカプセル化する他の方法については、451 ページの「cfmessagebox」を参照してください。ColdFusion ページは、以前は ColdFusion テンプレートまたは単にテンプレートと呼ばれることがありました。

カテゴリ

フロー制御タグ、ページ処理タグ

シンタックス

```
<cfinclude
    template = "template name"
    runOnce = "true|false">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfcache、cfflush、cfheader、cfhtmlhead、cfsetting、cfsilent

履歴

ColdFusion 10：runOnce 属性が追加されました。

ColdFusion MX: エラーの動作が変更されました。このタグを使用して長さ 0 バイトの CFML ページをインクルードしても、エラーにはなりません。

属性

属性	必須 / オプション	デフォルト	説明
template	必須		ColdFusion ページへの論理パスです。
runOnce	オプション	false	true に設定した場合、指定されたテンプレートは（既に処理済みの場合）、所定のリクエストでは再処理されません。

使用方法

ColdFusion は、インクルード対象ファイルを次の場所から検索します。

1 現在のページのディレクトリまたは現在のページに関連付けられたディレクトリ

2 ColdFusion Administrator でマップされたディレクトリ

インクルード対象ファイルの絶対 URL またはファイルシステムパスを指定することはできません。インクルード元ページのディレクトリまたは ColdFusion Administrator の [マッピング] ページで登録したディレクトリからの相対パスのみを使用できます。次の cfinclude ステートメントは、指定したディレクトリに "myinclude.cfm" ファイルが存在する場合に機能します。

```
<cfinclude template="myinclude.cfm">
<cfinclude template="../myinclude.cfm">
<cfinclude template="/CFIDE/debug/myinclude.cfm">
```

次のステートメントは機能しません。

```
<cfinclude template="C:\ColdFusion\wwwroot\doccomments\myinclude.cfm">
<cfinclude template="http://localhost:8500/doccomments/myinclude.cfm">
```

インクルード対象のファイルは、正しいシンタックスで記述された完全な CFML ページである必要があります。たとえば、インクルードされるページ内からデータを出力するには、参照する側のページ上ではなく、インクルードされるページ上に cfoutput タグを終了タグも含めて記述する必要があります。同様に、cfif タグを参照側のページとインクルードされるページとの間にまたがらせることはできません。このタグはインクルードされるページ内で完結していなければなりません。

次の例に示すように、template 属性に対して変数を指定できます。

```
<cfset templatetouse="../header/header.cfm">
<cfinclude template="#templatetouse#">
```

例

```
<!--- This example shows the use of cfinclude to paste CFML or HTML code into another page
    dynamically. --->

<h4>This example includes the dochome.htm page from the CFDOCS directory. The images do not
    display, because they are located in a separate directory. However, the page appears
    fully rendered within the contents of this page.</h4>
<cfinclude template = "../cfdocs/dochome.htm">
```

cfindex

説明

検索エンジンのコレクションにメタデータを挿入し、それを検索するためのインデックスを作成します。このエンジン（Solr）を使用すると、様々なタイプの物理ファイルやデータベースクエリを検索できます。クエリーからの結果のデータベース列にインデックスを付けると、複数の SQL クエリーを使用して同じデータを返していた場合に比べてはるかに迅速にクエリーデータを検索できるようになります。

コレクションにインデックスを付ける前に、ColdFusion Administrator または cfcollection タグを使用してコレクションを定義する必要があります。

ColdFusion Administrator を使用して、コレクションにインデックスを付けることもできます。

コレクションの作成、インデックス作成、検索の詳細については、『ColdFusion アプリケーションの開発』の Building a Search Interface を参照してください。

カテゴリ

拡張タグ

シンタックス

```
cfindex supports script style syntax:

new index(collection="<collection_name>");

<cfindex
    action = "update|delete|purge|refresh|fullimport|deltaimport|status|abort"
    autoCommit = "yes|no"
    collection = "collection name"
    body = "body"
    category = "category name"
    categoryTree = "category tree"
    custom1 = "custom value"
    custom2 = "custom value"
    custom3 = "custom value"
    custom4 = "custom value"
    extensions = "file extensions"
    key = "ID"
    language = "language"
    prefix = "location of documents"
    query = "query name"
    recurse = "yes|no"
    status = "status"
    title = "title"
    type = "type"
    URLpath = "URL">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfcollection、cfexecute、cfobject、cfreport、cfsearch、cfwddx

履歴

ColdFusion 10:

- 新しい fullimport、deltaimport、status、abort の各アクションが追加されました。

- autoCommit、fieldBoost、docBoost 属性が追加されました。

ColdFusion 9: Solr 検索エンジンを使用できるようになりました。

ColdFusion MX 7.0.1: prefix 属性が追加されました。

ColdFusion MX 7:

- update および refresh アクションに、category 属性、categoryTree 属性、custom3 属性、および custom4 属性が追加されました。
- update、refresh、delete、および purge の各アクションに status 属性が追加されました。
- 外部コレクションへの参照が削除されました。
- cflock の推奨使用方法が削除されました。

ColdFusion MX:

- action 属性の値 optimize が廃止になりました。この値は ColdFusion MX では機能せず、エラーが発生する可能性があります。
- external 属性の動作が変更されました。external 属性を指定する必要はなくなりました。内部コレクションか外部コレクションかは ColdFusion が自動的に検出します。
- Verity オペレーションの動作が変更されました。ColdFusion で、Acrobat PDF ファイルに対する Verity オペレーションがサポートされるようになりました。
- 返す例外が変更されました。このタグでは SEARCHENGINE 例外を返すことができます。
- コレクションのネーミング規則が変更され、スペースを含むコレクション名が使用できるようになりました。
- クエリー結果の動作が変更されました。cfindex タグでは、cfsearch タグのクエリー結果にインデックスを付けることができます。

属性

属性	必須 / オプション	デフォルト	説明
action	必須		• update: コレクションを更新し、インデックスに key を追加します。 • delete: key 属性で指定したコレクションのドキュメントを削除します。 • purge: コレクションのすべてのドキュメントを削除します。コレクションがオフラインになり、検索ができないようになります。 • refresh: コレクションのすべてのドキュメントを削除し、更新を実行します。 • fullimport：データベース全体のインデックスを作成するためのものです。例えば、初めてデータベースのインデックスを作成するときに使用します。 • deltaimport：インデックスを部分的に作成するためのものです。例えば、データベースに更新があった場合、fullimport ではなく、deltaimport を実行してコレクションを更新できます。 • status：処理されたドキュメントの総数やステータス（アイドルや実行中など）など、インデックス作成の状態を提供します。 • abort：実行中のインデックス作成タスクを中止します。
autoCommit	オプション		yes の場合、検索サーバーに対する変更は自動的にコミットされます。 no の場合は、cfindex action="commit" を使用して明示的にコミットを実行しない限り、インデックスが作成されたドキュメントはコミットされません。デフォルトでは、値は true に設定されます。
collection	必須		ColdFusion によって登録されているコレクションの名前。"personnel" のように指定します。
body	type=custom の場合は必須		• インデックスを付ける ASCII テキスト • query にクエリー名を指定している場合は、クエリー列名です。 区切りリストで列を指定できます。例 : "emp_name, dept_name, location" この属性は、type が file または path の場合は無視されます。action が delete の場合は無効です。
category	オプション		インデックスを付けるデータの検索カテゴリを指定する文字列です (複数指定可)。1 つのインデックスについて、複数のカテゴリをカンマで区切って定義することができます。
categoryTree	オプション		検索対象の階層構造のカテゴリまたはカテゴリツリーを指定する文字列値です。スラッシュ ("/") で区切られた、カテゴリの系列です。指定できるカテゴリツリーは 1 つだけです。
custom1	オプション		コレクションレコード内の不連続な値にインデックスを付けて、特定のレコードを検索できるようにする場合に使用します。一方、body 属性で指定した値は、連結され、指定の条件を使用してテキスト本文として検索されます。 type = custom の場合は、クエリ列名です。type が file または path の場合は、string です。
custom2	オプション		使用方法は、custom1 の場合と同じです。
custom3	オプション		使用方法は、custom1 の場合と同じです。
custom4	オプション		使用方法は、custom1 の場合と同じです。
docBoost	オプション		インデックスの作成時にドキュメント全体を増強します。docBoost は、ドキュメントのスコアを強化し、それによって検索結果のランキングを強化します。

属性	必須 / オプション	デフォルト	説明
extensions	オプション	htm、 html、 cfm、 cfml、 dbm、 dbml	type="Path" の場合に、ColdFusion がファイルのインデックス作成に使用するファイル拡張子をカンマ区切りで指定します。 "*." を入力すると、拡張子のないファイルが返されます。".*" を入力すると、すべてのファイルが返されます。 たとえば、次のコードでは、リストされている拡張子のファイルまたは拡張子のないファイルが返されます。 extensions = ".htm、.html、.cfm、.cfml、'*.'"
fieldBoost	オプション	htm、 html、 cfm、 cfml、 dbm、 dbml	インデックスの作成時に特定のフィールドを増強します。 fieldBoost は、フィールドのスコアを強化し、それによって検索結果のランキングを強化します。複数のフィールドを増強するには、カンマ区切りリストで値を指定します。
key	必須	(空の文字列)	key に対して指定される値は、type 属性によって決まります。 • type="file" のときは、そのファイルのディレクトリパスとファイル名です。 • type="path" のときは、ファイルの場所のディレクトリパスです。 • type="custom" のときは、データの場所を指定する固有の識別子です。たとえば、クエリーの場合、プライマリキーを格納する列の名前です。クエリー以外の場合、Web ページの URL などの識別子です。
language	オプション	English	オプションについては、cfcollection を参照してください。
prefix	オプション		K2 検索サービスと ColdFusion が別々のコンピュータにインストールされていて、type 属性を path に設定してファイルにインデックスを付ける場合のファイルの場所を指定します。
query	オプション		コレクションを生成する対象のクエリーの名前です。
recurse	オプション	no	• yes: type="path" の場合、key 属性で指定したパス以下のディレクトリにある指定されたファイルにインデックスを付けます。 • no
status	オプション		ColdFusion がステータス情報を返す先の構造体の名前です。
title	オプション		ドキュメントのタイトルをドキュメントから取得できない場合、そのタイトルを提供します。
type	オプション	query 属性を指定している場合は custom。指定していない場合は file。	• file: action の値をファイル名 (パスを含む) に適用します。key 属性のファイル名が必要です。 • path: action の値を、extensions フィルタを渡すディレクトリパス内のファイルに適用します。key 属性のディレクトリ名が必要です。 • dih：データインポートハンドラーの場合に使用します。 • *custom*: action の値をカスタムデータ (たとえば、クエリーからのデータ) に渡します。
URLpath	オプション		type が file または path の場合、URL パスを指定します。インデックスの作成時には、このパス名がファイル名の先頭に付き、url として検索から返されます。

使用方法

cfindex タグに必要な属性の設定は、query 属性を設定するかどうかによって決まります。query 属性を有効なクエリー名に設定した場合は、cfindex がディスク内のドキュメントにインデックスを付けるのではなく、クエリー内のデータにインデックスを付けるように指定します。query 属性を設定しない場合、cfindex はファイル (type = file)、ディレクトリパス内のファイルのセット (type = path)、または body 属性で指定されたテキスト (type = custom) にインデックスを付けると想定します。

query 属性を有効なクエリー名に設定した場合、cfindex タグは、次の属性およびその値で指定されたとおりにインデックスを作成します。

タイプ	属性値
File	key 属性は、完全ファイル名 (パスを含む) を含むクエリー内の列名です。
Path	key 属性は、ディレクトリのパス名を含むクエリー内の列名です。
	extensions 属性および recurse 属性が指定されている場合、これらの属性はどのタイプのファイルが含まれているかを示します。アクションが delete の場合、cfindex はコレクションのキーを削除します。
Custom	key 属性は、目的のものを含む列名を指定します。たとえば、データベースのプライマリキー値などです。これはコレクションのプライマリキーであるため、固有のものでなければなりません。アクションが delete の場合、key 属性は、削除するキーを含むクエリー内の列の名前です。
	body 属性は必須で、インデックス作成の対象となるテキストを含んだ列の名前がカンマで区切られて一覧表示されます。

query 属性を設定しない場合、cfindex タグは、次の属性およびその値で指定されたとおりにインデックスを作成します。

タイプ	属性値
File	key 属性が必要で、その値はファイルへのフルパスです。
Path	key 属性が必要で、その値はディレクトリのパス名です。
	extensions 属性および recurse 属性が指定されている場合、これらの属性はどのタイプのファイルが含まれているかを示します。アクションが delete の場合、キーおよびドキュメントファイルがどちらも削除されます。
Custom	key 属性は、キーを指定する任意の識別子です。アクションが delete の場合、key 属性は削除するドキュメントキーです。
	body 属性が必要で、その値はインデックスを付ける対象のテキストです。

type を指定しないで query を設定した場合、ColdFusion は type を custom のデフォルト値に設定します。

type と query のいずれも設定しない場合、ColdFusion は type を file のデフォルト値に設定します。

type が custom の場合、key および body 以外のすべての属性で、列名だけでなくリテラル値も指定することができます。これにより、コレクション内でフィールドを空に変更することができます。

status 属性

status 属性は、cfindex オペレーションの結果に関する次の情報と診断を提供します。

キー	型	説明
BADKEYS	Struct	無効なキーのインデックス作成に関する診断メッセージを含む、キーの構造体です。無効なキーがない場合、このキーは存在しません。
DELETED	Number	削除されたキーの数です。
MESSAGES	Array	診断メッセージの配列で、重大でないエラーと警告が含まれます。メッセージがない場合、このキーは存在しません。
INSERTED	Number	コレクションに挿入されたキーの数です。
UPDATED	Number	コレクションで更新されたキーの数です。

例

```
 <!--- EXAMPLE #1 Index a file, type = "file". ---------------------------->
<!--- Example dumps content of status variable (info). ------------------->
<cfindex collection="CodeColl"
    action="refresh"
    type="file"
    key="C:\ColdFusion\wwwroot\vw_files\cfindex.htm"
    urlpath="http://localhost:8500/vw_files/"
    language="English"
    title="Cfindex Reference page"
    status="info">

<!--- Search for Attributes. --->
<cfsearch
    name = "mySearch"
    collection = "CodeColl"
    criteria = "Attributes"
    contextpassages = "1"
    maxrows = "100">
<cfoutput>
    key=#mySearch.key#<br />
    title=#mySearch.title#<br />
    context=#mySearch.context#<br />
    url=#mySearch.url#<br />
</cfoutput>

<cfdump var="#info#">

<!--- EXAMPLE #2 Index a path (type = "path"). ------------------------------>
<cfindex collection="CodeColl"
    action="refresh"
    type="path"
    key="C:\inetpub\wwwroot\vw_files\newspaper\sports"
    urlpath="http://localhost/vw_files/newspaper/sports"
    extensions = ".htm, .html"
    recurse="no"
    language="English"
    categoryTree="vw_files/newspaper/sports"
    category="Giants">

<!--- Search for any references to criteria. --->
<cfsearch
    name = "mySearch"
    collection = "CodeColl"
    categoryTree="vw_files/newspaper/sports"
    category="Giants"
    criteria = "Williams"
    contextpassages = "1"
    maxrows = "100">
<cfoutput>
    key=#mySearch.key#<br />
    title=#mySearch.title#<br />
    context=#mySearch.context#<br />
    url=#mySearch.url#<br />
</cfoutput>

<!---EXAMPLE #3: Index a QUERY (type = "custom") using custom1. ------------>
<!--- Retrieve data from the table. --->
<cfquery name="getCourses" datasource="cfdocexamples">
    SELECT * FROM COURSES
</cfquery>
```

```
<!--- Update the collection with the above query results. --->
<!--- key is Course_ID in the Courses table. ---->
<!--- body specifies the columns to be indexed for searching. --->
<!--- custom1 specifies the value of the Course_Number column. --->

<cfindex
    query="getCourses"
    collection="CodeColl"
    action="Update"
    type="Custom"
    key="Course_ID"
    title="Courses"
    body="Course_ID,Descript"
    custom1="Course_Number"
>
<h2>Indexing Complete</h2>
<!--- cno supplies value for searching custom1; could be form input instead. --->
<cfset cno = "540">
<cfsearch
    name = "mySearch"
    collection = "CodeColl"
    criteria = "CF_CUSTOM1 <MATCHES> #cno#"
    contextpassages = "1"
    maxrows = "100">
<!--- Returns indexed values (Course_ID and Descript) for
            Course_Number 540. --->
<cfoutput>
    key=#mySearch.key#<br />
    title=#mySearch.title#<br />
    context=#mySearch.context#<br />
    url=#mySearch.url#<br />
</cfoutput>

<!--- EXAMPLE #4 Index a FILE within a QUERY (type= "file"). ---------------->
<!--- Retrieve row with a column that contains a filename (Contract_File). --->
<cfquery name="getEmps" datasource="cfdocexamples">
    SELECT * FROM EMPLOYEE WHERE EMP_ID = 1
</cfquery>

<!--- Update the collection with the above query results. --->
<!--- key specifies the column that contains a complete filename. --->
<!--- file is indexed in same way as if no query involved. --->
<cfindex
    query="getEmps"
    collection="CodeColl"
    action="Update"
    type="file"
    key="Contract_File"
    title="Contract_File"
    body="Emp_ID,FirstName,LastName,Contract_File">

<h2>Indexing Complete</h2>
<cfsearch
    name = "mySearch"
    collection = "CodeColl"
    criteria = "vacation"
    contextpassages = "1"
    maxrows = "100">
<cfoutput>
    key=#mySearch.key#<br />
    title=#mySearch.title#<br />
    context=#mySearch.context#<br />
```

```
    url=#mySearch.url#<br />
</cfoutput>

<!--- EXAMPLE # 5 Index a PATH within a QUERY. ---------------------------->
<!--- Retrieve a row with a column that contains a path (Project_Docs). --->
<cfquery name="getEmps" datasource="cfdocexamples">
    SELECT * FROM EMPLOYEE WHERE Emp_ID = 15
</cfquery>

<!--- Update the collection with the above query results. --->
<!--- key specifies a column that contains a directory path. --->
<!--- path is indexed in same way as if no query involved. --->
<cfindex
    query="getEmps"
    collection="CodeColl"
    action="update"
    type="path"
    key="Project_Docs"
    title="Project_Docs"
    body="Emp_ID,FirstName,LastName,Project_Docs">

<h2>Indexing Complete</h2>

<cfsearch
    name = "getEmps"
    collection = "CodeColl"
    criteria = "cfsetting"
    contextpassages = "1"
    maxrows = "100">
<cfoutput>
    key=#getEmps.key#<br />
    title=#getEmps.title#<br />
    context=#getEmps.context#<br />
    url=#getEmps.url#<br />
</cfoutput>

<!--- EXAMPLE #6 Deletes keys in the CodeColl collection for html files --->
<!--- in the specified directory (but not in subdirectories). ------------->

<cfindex collection="CodeColl"
    action="delete"
    type="path"
    key="C:\ColdFusion\wwwroot\vw_files\newspaper"
    urlpath="http://localhost:8500/vw_files/newspaper"
    extensions = ".htm, .html"
    recurse="no">

<!--- EXAMPLE #7 Purges all keys in the CodeColl collection --->
<!--- with recursion. ------------------------------------------------------>

<cfindex collection="CodeColl"
    action="purge"
    type="path"
    key="C:\ColdFusion\wwwroot\vw_files\newspaper">
```

cfinput

説明

cfform タグ内で使用します。入力検証をサポートする入力コントロールをフォームに配置します。

カテゴリ

フォームタグ

シンタックス

```
<cfinput
    name = "name"
    autosuggest = "list or bind expression"
    autosuggestBindDelay = "integer number if seconds"
    autosuggestMinLength = "integer"
    bind = "bind expression"
    bindAttribute = "attribute name"
    bindOnLoad = "no|yes"
    checked = "yes|no"
    dayNames = "day of week labels separated by commas"
    delimiter = "character"
    disabled = "disabled"
    enabled = "yes|no"
    firstDayOfWeek = "day name"
    height = "number of pixels"
    id = "HTML id"
    label = "text"
    matchContains = "true|false"
    mask = "masking pattern"
    maxLength = "number"
    maxResultsDisplayed = "number"
    message = "text"
    monthNames = "month labels"
    onBindError = "JavaScript function name"
    onChange = "JavaScript or ActionScript"
    onClick = "JavaScript or ActionScript"
    onError = "script name"
    onKeyDown = "JavaScript or ActionScript"
    onKeyUp = "JavaScript or ActionScript"
    onMouseDown = "JavaScript or ActionScript"
    onMouseUp = "JavaScript or ActionScript"
    onValidate = "script name"
    pattern = "regular expression"
    range = "minimum value, maximum value"
    required = "yes|no"
    showAutosuggestLoadingIcon = "yes|no"
    size = "integer"
    sourceForToolTip = "URL"
    src = "image URL"
    style = "style specification"
    tooltip = "text"
    type = "input type"
    typeahead = "no|yes"
    validate = "data type"
    validateAt = "onBlur|onServer|onSubmit"
    value = "initial value"
    visible = "yes|no"
    width = "integer number of pixels">
```

一部の属性は、特定の表示形式のみに適用されます。詳細については、「属性」の表を参照してください。

HTML 形式フォームの場合、上に記載されていない標準の HTML 入力コントロール属性は HTML に直接渡され、通常どおりに機能します。

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfajaximport、cfapplet、cfcalendar、cfform、cfformgroup、cfformitem、cfgrid、cfselect、cfslider、cftextarea、cftree、『ColdFusion アプリケーションの開発』の Using Ajax User Interface Components and Features の Using Ajax form controls and features

履歴

ColdFusion 8

- autosuggest、autosuggestBindDelay、autosuggestMinLength、delimiter、maxResultsDisplayed、showAutosuggestLoadingIcon、および typeahead の各属性が追加されました。
- HTML フォームで bind 属性が使用できるようになり、bindAttribute、bindOnload、および onBindError の各属性が追加されました。
- sourceForTooltip 属性が追加されました。
- HTML フォームの type 属性の datefield 値に対するサポートが追加されました。

ColdFusion MX 7:

- ボタン、ファイル、非表示、イメージ、リセット、および送信の各コントロールに対するサポートが追加されました。
- Flash コントロールおよび XML コントロール (cfform タグで指定) の生成に対するサポートが追加されました。
- datefield タイプ (Flash フォームの場合のみ)、および dayNames 属性と monthNames 属性に対するサポートが追加されました。
- Flash フォームで使用できる bind、enabled、height、label、tooltip、visible、および width の各属性が追加されました。
- onBlur 検証と onServer 検証に対するサポートが追加されました (validateAt 属性など)。
- validate 属性の値として、USdate、range、boolean、email、URL、uuid、guid、maxlength、noblanks、および submitOnce が追加されました。
- 複数の送信を回避できるようになりました。
- mask 属性が追加されました。
- passthrough 属性は非推奨になりました。すべての HTML input タグ属性を直接サポートするようになりました。

ColdFusion MX: cfform タグの preserveData 属性の動作が変更されました。この属性が true に設定されている場合、ColdFusion は、ラジオボタンとチェックボックスの値がそのコントロールの送信値と一致する場合にのみ、ラジオボタンまたはチェックボックスにチェックを付けます。以前のリリースでは、送信値が cfinput チェックボックスまたはラジオボタンのどれにも一致しない場合、checked 属性の値が使用されていました。

属性

次の表は、ColdFusion が直接使用する属性のリストです。このリストに記載されていない HTML form タグ属性もすべてサポートされ、ブラウザに直接渡されます。

注意：「すべて」または「XML」と記載されていない属性は、ColdFusion に同梱されているスキンでは処理されません。ただし、これらの属性は生成された XML に含まれます。

属性	必須 / オプション、形式	デフォルト	説明
name	必須、 すべて		フォーム入力要素の名前です。
autosuggest	オプション、 HTML		ユーザーがテキストを入力するときに表示する入力候補を指定します。ユーザーは入力候補を選択してテキスト入力を完成させることができます。 有効な値は、以下のいずれかです。 • delimiter 属性で指定した区切り文字によって区切られた入力候補値を含む文字列。 • 現在の入力テキストに基づいて入力候補値を取得するバインド式。ユーザー入力を表現するために、バインド式では cfautosuggestvalue バインドパラメータを渡す必要があります。 cfinput type="text" の場合にのみ有効です。 詳細については、『ColdFusion アプリケーションの開発』の Using autosuggest text input fields を参照してください。
autosuggestBindDelay	オプション、 HTML	0.5 秒	autosuggest バインド式を呼び出す最小の間隔を指定するゼロ以外の整数です (単位 : 秒)。また、この値は、ユーザーが最小の長さのテキストをフィールドに入力してから入力候補リストを表示するまでの時間も指定します。この属性を使用すると、ユーザーがテキストを入力するときにサーバーに送信されるリクエストの数を制限できます。 cfinput type="text" の場合にのみ有効です。 **メモ :** この属性を省略した場合のみ、デフォルトの動作が有効になります。属性を指定する場合、時間はゼロ以外の整数値である必要があります。
autosuggestMinLength	オプション、 HTML	1	テキストボックスに何個以上の文字が入力された場合に入力候補のバインド式を呼び出すかを指定します。 cfinput type="text" の場合にのみ有効です。
bind	オプション、 HTML、 Flash		コントロールの属性をダイナミックに設定するバインド式です。詳細については、「使用方法」を参照してください。
bindAttribute	オプション、 HTML	value	bind 属性を使用して値を設定する HTML タグ属性を指定します。ColdFusion 固有の属性ではなく、ブラウザの HTML DOM ツリーに含まれる属性のみを指定できます。 bind 属性がない場合は無視されます。 cfinput type="text" の場合にのみ有効です。
bindOnLoad	オプション、 HTML	no	フォームを最初にロードするときに bind 属性の式を実行するかどうかを指定するブール値です。 bind 属性がない場合は無視されます。 cfinput type="text" の場合にのみ有効です。
checked	オプション、 すべて	no	ラジオボタンまたはチェックボックスコントロールにチェックマークを付けます。 • yes • no HTML 形式の場合、checked に値を指定するか、またはこの属性を値なしで指定することで、項目が選択されたかどうかを示すことができます。

属性	必須 / オプション、形式	デフォルト	説明
dayNames	オプション、すべて	S、M、T、W、T、F、S	datefield タイプにのみ適用されます。カレンダーに表示される曜日名を設定するカンマ区切りのリストです。日曜日 (Sunday) が先頭で、残りの曜日が通常どおりにその後に続きます。
delimiter	オプション、HTML	カンマ (,)	スタティックな入力候補リストのエントリを区切るために使用する区切り文字です。この属性は、autosuggest 属性で区切り値の文字列が指定されている場合にのみ効果を持ちます。
disabled	オプション、すべて	no	ユーザー入力を無効にし、コントロールを読み取り専用にします。属性の動作は、次のようにフォームの形式により決まります。 • HTML 形式の場合 : ColdFusion はこの属性を直接 HTML に渡します。入力を無効にするには、disabled を値なしで指定するか (HTML 形式の場合)、値 disabled を指定します (XHTML 準拠の場合)。入力を有効にするには、この属性を省略します。 • Flash 形式の場合 : 入力を無効にするには、属性を省略して disabled を指定するか、disabled="yes" (または ColdFusion で使用する true などの真を表すブール値) を指定します。入力を有効にするには、属性を省略するか、disabled="no" (または ColdFusion で使用する false などの偽を表すブール値) を指定します。
enabled	オプション、Flash	yes	コントロールを有効にするかどうかを指定するブール値です。無効なコントロールはライトグレーで表示されます。disabled 属性の逆です。
firstDayOfWeek	オプション、すべて	0	datefield タイプにのみ適用されます。カレンダーでその週の最初の曜日を指定する 0 ～ 6 の範囲の整数です。0 は日曜日、6 は土曜日を示します。
height	オプション、「説明」を参照		ほとんどの Flash のタイプに適用されます。一部のブラウザでは、HTML の image タイプに適用されます。コントロールの高さをピクセル単位で指定します。表示される高さは、指定したサイズよりも小さい場合があります。
id	オプション、HTML	name 属性の値	フォームの HTML ID です。
label	オプション、Flash、XML		Flash フォーム内でコントロールの横に配置するラベルです。button、hidden、image、reset、または submit の各タイプには使用しません。

属性	必須 / オプション、形式	デフォルト	説明
mask	オプション、Flash、HTML		type="text" のタグの場合 : ユーザーが入力できる文字パターン、またはフォームが ColdFusion に送信する文字パターンを制御するマスクパターンです。マスク文字および対応する有効な入力文字は次のとおりです。 • A = [A-Za-z] • X = [A-Za-z0-9] • 9 = [0-9] • ? = 任意の文字 • これら以外のすべての文字 = リテラル文字を挿入 type="datefield" のタグの場合、ユーザーがカレンダーで選択する日付の形式を制御するパターンです。マスク文字は次のとおりです。 • D は日です。0 ～ 2 個のマスク文字を使用できます。 • M は月です。0 ～ 4 個のマスク文字を使用できます。 • Y は年です。0、2、または 4 個の文字を使用できます。 • E は曜日です。0 ～ 4 個の文字を使用できます。 詳細については、「使用方法」を参照してください。
matchContains	オプション	false	true の場合、返される一致にはクエリー文字列が含まれます。デフォルトでは、クエリー文字列で始まる結果が返されます。
maxLength	オプション、すべて		type="Text" または "password" の場合に、入力テキストの最大長を設定します。総合的な長さ検証の場合、validate 属性で maxLength 検証を指定します。それ以外の場合は、この属性を使用すると、ユーザーが指定の長さを超えて入力しないようにできますが、指定の長さを超えた値をペーストしないようにすることはできません。
maxResultsDisplayed	オプション、HTML	10	入力候補リストに表示するエントリの最大数です。 cfinput type="text" の場合にのみ有効です。
message	オプション、すべて		検証に失敗した場合に表示されるメッセージテキストです。
monthNames	オプション、すべて	January、February、March、April、May、June、July、August、September、October、November、December	datefield タイプにのみ適用されます。カレンダーの最上部に表示される月名のカンマ区切りのリストです。
onBindError	オプション、HTML	「説明」を参照	バインド式 (入力候補のバインド式を含む) の評価でエラーが発生した場合に実行する JavaScript 関数の名前です。この関数は 2 つの属性 (HTTP ステータスコードとメッセージ) を取る必要があります。 この属性を省略し、(ColdFusion.setGlobalErrorHandler 関数を使用して) グローバルエラーハンドラを指定すると、エラーメッセージが表示されます。それ以外の場合は、デフォルトのエラーポップアップが表示されます。
onChange	オプション、すべて		ユーザーのアクションに応じてコントロールに変化があったときに実行される JavaScript (HTML/XML の場合) または ActionScript (Flash の場合) です。Flash の場合、datefield、password、および text の各タイプにのみ適用されます。

属性	必須 / オプション、形式	デフォルト	説明
onClick	オプション、すべて		ユーザーがコントロールをクリックしたときに実行される JavaScript (HTML/XML の場合) または ActionScript (Flash の場合) です。Flash の場合、button、checkbox、image、radio、reset、および submit の各タイプにのみ適用されます。
onError	オプション、HTML、XML		検証に失敗した場合に実行するカスタム JavaScript 関数の名前です。
onKeyDown	オプション、すべて		ユーザーがコントロールでキーボードのキーを押したときに実行される JavaScript (HTML/XML の場合) または ActionScript (Flash の場合) です。
onKeyUp	オプション、すべて		ユーザーがコントロール内でキーボードのキーを離したときに実行される JavaScript (HTML/XML の場合) または ActionScript (Flash の場合) です。
onMouseDown	オプション、すべて		ユーザーがコントロール内でマウスボタンを離したときに実行される JavaScript (HTML/XML の場合) または ActionScript (Flash の場合) です。
onMouseUp	オプション、すべて		ユーザーがコントロール内でマウスボタンを押したときに実行される JavaScript (HTML/XML の場合) または ActionScript (Flash の場合) です。
onValidate	オプション、HTML、XML		ユーザー入力の検証に使用するカスタム JavaScript 関数の名前です。このルーチンには、フォームオブジェクト、入力オブジェクト、および入力オブジェクトの値が渡されます。このルーチンは、検証に成功した場合は true、失敗した場合は false を返さなければなりません。この属性を使用する場合、validate 属性は無視されます。
pattern	validate = "regex" の場合は必須、HTML、XML		入力を検証するための、JavaScript 正規表現のパターンです。この属性は、validate 属性で **regex** が指定されている場合にのみ効果を持ちます。 先頭および末尾のスラッシュ記号は削除されます。例とシンタックスについては、『ColdFusion アプリケーションの開発』の Building Dynamic Forms with cfform Tags を参照してください。
range	オプション、すべて		指定可能な数値の最大値と最小値です。この属性は、validate 属性で **range** が指定されている場合にのみ効果を持ちます。 単一の値を指定するか、単一の値の後にカンマを付けて指定した場合、その値は最小値として処理されます。最大値はありません。カンマの後に単一の値を指定した場合、その値は最大値として設定されます。最小値はありません。 **メモ :** XML 形式のフォームで onsubmit または onBlur の検証を使用した場合、range 属性は処理されません。
readonly	オプション、Flash、HTML		HTML フォームおよび Flash フォームに適用されます。cfinput type = "text" の場合にのみ有効です。他のすべての入力タイプでは、この属性は無視されます。
required	オプション、すべて	no	• yes: フィールドにはデータが必要です。 • no: フィールドは空でもかまいません。
showAutosuggestLoadingIcon	オプション、HTML	true	テキスト入力の候補値をロードするときにアニメーションのアイコンを表示するかどうかを指定するブール値です。
size	オプション、すべて		入力コントロールのサイズです。type="radio" または "checkbox" の場合は無視されます。 Flash フォームで指定した場合、ColdFusion はコントロール幅のピクセル値を指定サイズの 10 倍に設定し、width 属性を無視します。

属性	必須 / オプション、形式	デフォルト	説明
sourceForTooltip	オプション、HTML		ツールヒントとして表示するページの URL です。このページには書式を制御するための HTML マークアップを含めることができ、ヒントにはイメージを含めることができます。 この属性を指定すると、ヒントのロード中にアニメーションのアイコンと「ロード中 ...」というテキストが表示されます。
src	オプション、Flash、HTML		Flash の button、reset、submit、image の各タイプ、および HTML の image タイプに適用されます。ボタンに使用するイメージの URL です。
style	オプション、すべて		HTML フォームまたは XML フォームでは、ColdFusion は style 属性をブラウザまたは XML に渡します。 Flash 形式では、CSS 形式のスタイル指定でなければなりません。Flash スタイルの指定に関する詳細については、『ColdFusion アプリケーションの開発』の Creating Forms in Flash を参照してください。 XML 形式では、ColdFusion は style 属性を XML に渡します。
tooltip	オプション、Flash、HTML		マウスポインタをコントロールの上に置いたときに表示されるテキストです。 sourceForTooltip 属性が指定されている場合は無視されます。
type	オプション、すべて	text	作成する入力コントロールのタイプです。 • button: プッシュボタン • checkbox: チェックボックス • datefield: (HTML および Flash の場合のみ) ユーザーが日付を選択するためのカレンダーを呼び出せる日付入力フィールド。HTML 形式の場合のみ、ユーザーはこのフィールドに日付を入力することもできます。 • file: ファイルセレクタ。Flash ではサポートされません。Ajax フォームの送信を使用してページから非同期でフォームを送信する場合はサポートされません。 • hidden: 非表示コントロール • image: イメージ付きのクリック可能ボタン • password: パスワード入力コントロール。入力した値は表示されません。 • radio: ラジオボタン • reset: フォームリセットボタン • submit: フォーム送信ボタン • text: テキスト入力ボックス この属性では、HTML 5 のすべての入力タイプ（email、range、date など）がサポートされます。
typeahead	オプション、HTML	no	入力候補リストの最初のエントリを使用して自動的にユーザー入力を完成させるかどうかを指定するブール値です。 cfinput type="text" の場合にのみ有効です。
validate	オプション、すべて		実行する検証のタイプです。使用可能な検証のタイプとアルゴリズムは、形式によって異なります。詳細については、「使用方法」を参照してください。

属性	必須 / オプション、形式	デフォルト	説明
validateAt	オプション、 すべて	onSubmit	検証の実行方法です。次の値を指定します (複数指定可)。 • onSubmit • onServer • onBlur onBlur 値と onSubmit 値は、Flash フォームでは同じです。複数の値を指定する場合は、カンマ区切りのリストを使用します。 詳細については、「使用方法」を参照してください。
value	type の設定により異なる、 すべて		HTML: HTML の value 属性に対応します。使用方法はコントロールのタイプによって異なります。 Flash: (オプション) ボタンタイプ入力のテキストを button、submit、および image の中から指定します。
visible	オプション、Flash	yes	コントロールを表示するかどうかを指定するブール値です。表示されないコントロールが使用するスペースは空白です。
width	オプション、「説明」を参照		ほとんどの Flash のタイプに適用され、一部のブラウザでは、HTML の image タイプに適用されます。コントロールの幅をピクセル単位で指定します。Flash フォームでは、size 属性の値が指定されている場合、この属性は無視されます。

注意：XML でサポートされない属性は、ColdFusion に同梱されているスキンでは処理されません。ただし、これらの属性は生成された XML に含まれます。

使用方法

このタグを正しく動作させるには、JavaScript 対応のブラウザを使用する必要があります。

cfform の preserveData 属性が true に設定されていて、そのフォームが同じページに送信された場合は、cfinput の value 属性または checked 属性ではなく、cfinput コントロールの送信値が使用されます。

Flash フォームの datefield 入力コントロールから日付を選択するには、次のようにキーボードを使用します。Tab キーを押してこのフィールドに移動し、スペースバーを押してメニューを表示します。上下左右の矢印キーを使用すると、現在選択されている日付を変更できます。Home キーと End キーを使用すると、月の開始日と終了日にそれぞれ進むことができます。Page Up キーと Page Down キーを使用すると、前月と翌月にそれぞれ進むことができます。

注意：Flash 形式フォームで datefield エントリをクリアするには、フィールドを選択してメニューを開き、選択した日付をクリックします。

詳細については、cfform を参照してください。このタグで JavaScript 正規表現を使用する方法については、『ColdFusion アプリケーションの開発』の Building Dynamic Forms with cfform Tags を参照してください。

検証

次のセクションでは、cfinput タグでの検証の実行方法について説明します。

検証メソッド ColdFusion では、cfinput の text および password フィールドの検証メソッドが 4 つ用意されています。

validateAt 属性で次のメソッドのいずれか 1 つまたは複数のメソッドを組み合わせて指定できます。

- onSubmit: フォームをサーバーに送信する前に検証を実行する JavaScript 関数が、ブラウザ上のフォームページに含まれます。Flash 形式では、このオプションは onBlur と同じです。

- onBlur: HTML 形式の場合、フィールドがフォーカスを失ったときに検証を実行する JavaScript 関数が、ブラウザ上のフォームページに含まれます。Flash 形式の場合、この属性は onSubmit と同じです。OnBlur 検証では、onSubmit 検証と同じアルゴリズムを使用します。OnBlur 検証は ColdFusion MX 7 で追加されました。
- onServer: ColdFusion はサーバー上で検証を実行します。一部の onServer アルゴリズムは、onSubmit アルゴリズムと異なります。OnServer の Date および Time 検証では、onSubmit 検証よりも多くのパターンを使用できます。OnServer 検証は ColdFusion MX 7 で追加され、非表示フィールドを自動的に生成して検証をサポートします。

validate 属性を省略して、別の非表示フォームフィールドでフィールドの検証のタイプを指定することもできます。このタイプの検証は onServer 検証と同じですが、フィールドに対して検証を実行するたびに異なるメッセージを指定することができます。この検証は、以前の ColdFusion リリースとの間に後方互換性があります。非表示フォームフィールドを使用した検証の詳細については、cfform および『ColdFusion アプリケーションの開発』の Validating form data using hidden fields の Validating form data using hidden fields を参照してください。

検証のタイプ： validate 属性で次の値を使用して、すべての検証メソッドの入力検証を指定します。ほとんどの属性は、パスワードフィールドまたはテキストフィールドにのみ適用されます。カンマ区切りリストで複数の検証タイプを指定できますが、意味があるのは一部の組み合わせだけです。

タイプ	説明
date	validateAt="onServer" の場合、IsDate 関数で true を返す任意の日付形式を使用できます。それ以外の場合は USdate と同じです。
USdate	mm/dd/yy、mm-dd-yy、または mm.dd.yy の書式を使用する米国の日付 (1 ～ 2 桁の月と日、1 ～ 4 桁の年を使用) です。
eurodate	dd/mm/yy の書式を使用する日付 (1 ～ 2 桁の月と日、1 ～ 4 桁の年を使用) です。区切り文字として /、-、または . を使用できます。
time	時刻形式 (hh:mm:ss) です。
float または numeric	数値です。整数を使用できます。
integer	整数。
range	数値の範囲です。
boolean	ブール値 (yes、no、true、false、または数値) に変換可能な値です。
telephone	米国の標準の電話番号の形式。最初に国番号 1 を付加することや、最大 5 桁の内線番号を付加することができます。x で始めることもできます。
zipcode	米国の 5 桁または 9 桁の郵便番号。#####-#### の形式。区切り文字にはハイフン (-) かスペースを使用できます。
creditcard	空白とダッシュは削除されます。mod10 アルゴリズムを使用して番号が検証されます。番号は 13 ～ 16 桁であることが必要です。
ssn または social_security_number	米国の社会保障番号。###-##-#### の形式。区切り文字にはハイフン (-) かスペースを使用できます。
email	有効な電子メールアドレス。名前 @ サーバー . ドメインの形式。形式が正しいかどうかは確認されますが、アクティブで有効な電子メールアドレスであるかどうかは確認されません。
URL	有効な URL パターン。http、https、ftp ファイル、mailto、および news URL がサポートされています。
guid	Microsoft/DCE 形式に準拠する一意の識別子 (xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx)。x には 16 進数の数字が入ります。
uuid	ColdFusion の形式 (xxxxxxxx-xxxx-xxxx-xxxxxxxxxxxxxxxx) に準拠する UUID (Universally Unique Identifier)。x には 16 進数の数字が入ります。
maxlength	入力可能な文字数を制限します。

タイプ	説明
noblanks	空白のみで構成されるフィールドを許可しません。
regex または regular_expression	入力内容を pattern 属性と照合します。HTML および XML 形式でのみ有効です。Flash 形式では無視されます。
SubmitOnce	submit タイプおよび image タイプでのみ使用されます。次のページがロードされるまでの間、ユーザーが同じフォームを複数回送信できないようにします。たとえば、注文が送信されたときに、その注文が確認されるまで、同じ注文を再送信できないようにします。HTML および XML 形式でのみ有効です。Flash 形式では無視されます。

検証動作の相違： 前の表に記載されている内容は、一般的な検証動作についての説明です。使用される検証コードは、検証メソッドやフォームのタイプによって異なります。したがって、次のように、場合によって使用されるアルゴリズムが異なります。

- 日付時刻値に使用される検証アルゴリズムは、onSubmit/OnBlur と OnServer の間で異なります。
- Flash の onSubmit/OnBlur 検証に使用されるアルゴリズムは、HTML/XML 形式で使用されるアルゴリズムとは異なり、一般的には、より単純なルールに準拠します。

前の表に記載されている内容は、HTML 形式での onSubmit/OnBlur の動作に関する説明です。OnServer 検証アルゴリズムの詳細については、『ColdFusion アプリケーションの開発』の Data validation types の Data validation types を参照してください。

各検証タイプの利点と欠点など、検証の詳細については、『ColdFusion アプリケーションの開発』の Validating Data を参照してください。

入力データのマスキング

HTML フォームと Flash フォームでは、mask 属性を使用して、text フィールドに入力可能なデータの形式、または datefield 入力コントロールカレンダーで選択されるデータの形式を制御できます。HTML 形式では、ユーザーがマスクに従わない日付を datefield 入力コントロールに入力しないようにすることはできません。1 つのフィールドで、マスキングと検証を組み合わせて使用することもできます。

テキストフィールドの場合、パターンに含まれるリテラル文字は ColdFusion によって自動的に挿入されます (電話番号のハイフン (-) など)。ユーザーは、フィールドの可変部分のみを入力する必要があります。

次のパターンを使用した場合、ユーザーは EB-1234-c1-098765 という形式でパーツ番号を入力する必要があります。ユーザーは、最初の数字から入力を開始します。先頭の EB と「-」は ColdFusion によって自動的に挿入されます。ユーザーは、4 桁の数字を入力した後、2 つのアルファベット文字を入力し、最後に 6 桁の数字を入力する必要があります。

```
<cfinput type="text" name="newPart" mask="EB-9999-XX-999999"/>
```

注意：すべての文字を大文字または小文字で入力させるには、アクションページで ColdFusion の UCase 関数または LCase 関数を使用します。

type="datefield" を使用するタグ (および cfcalendar タグ) の場合、ユーザーがカレンダーで日付を選択するとき、次のようにパターン文字の数によって出力の形式が決まります。

マスク	パターン
D	1 桁または 2 桁で表される日付 (1 や 28 など)
DD	2 桁で表される日付 (01 や 28 など)
M	1 桁または 2 桁で表される月 (1 や 12 など)
MM	2 桁で表される月 (01 や 12 など)
MMM	月の省略名 (Jan や Dec など)

マスク	パターン
MMMM	月の名前 (January や December など)
YY	2 桁で表される年 (05 など)
YYYY	4 桁で表される年 (2005 など)
E	1 桁で表される曜日 (1 または 7 など)
EEE	曜日の省略名 (Mon や Sun など)
EEEE	曜日名 (Monday や Sunday など)

次のパターンを Flash フォームで使用すると、datefield 入力コントロールで選択された日付が、「04/29/2004」という形式のテキストとして ColdFusion に送信されます。

```
<cfinput name="stDate" type="datefield" label="date:" mask="mm/dd/yyyy"/>
```

Flash フォームデータのバインディング

bind 属性を使用すると、他のフォームフィールドの内容を別のフォームフィールドにできます。Flash 形式の cfinput タグの bind 属性で他のフィールドのテキストを指定するには、次の形式を使用します。

{**sourceTagName**.text}

たとえば、次の例では、firstName および lastName フィールドの値を使用して電子メールアドレスが作成されます。ユーザーはこの値を変更できます。

```
<cfinput type="text" name="email" label="email"
    bind="{firstName.text}.{lastName.text}@mm.com">
```

HTML フォームデータのバインディング

bind 属性を使用すると、cfinput 属性をダイナミックに設定できます。たとえば、名前フィールドとドメインフィールドの値に基づいて、電子メールフィールドにテキストを自動的に挿入することができます。

HTML 形式の場合、bind 属性では、バインド式を指定します。バインド式には次の形式を使用できます。

- バインドパラメータ、または少なくとも 1 つのバインドパラメータを含む文字列。bind パラメータでは、フォームコントロール値または他の属性を指定し、オプションでイベントを指定します。基本的な形式では、バインド対象となるコントロールの name 属性または id 属性を中括弧 ({}) で囲んだものがバインドパラメータになります。cfinput コントロール属性の値は、バインドパラメータで指定したコントロール属性の値によって決定されます。
- CFC 関数、JavaScript 関数、または URL。通常は、関数パラメータとしてバインドパラメータを使用します。cfinput 属性の値は、関数または URL から返されたデータによって設定されます。

HTML フォームデータのバインディングの使用方法の詳細については、『ColdFusion アプリケーションの開発』の Binding data to form fields を参照してください。

注意： button の type 属性を使用して cfinput コントロールにバインドするには、ボタンにバインドするコントロールのバインド式で click などのバインドイベント設定を指定します。デフォルトのイベントの onChange は何の影響も与えません。

例 : バインドなし

```
<!--- This example shows the use of cfinput within a cfform to ensure simple
    validation of text items. --->
<cfform action = "cfinput.cfm">
<!--- Phone number validation. --->
Phone Number Validation (enter a properly formatted phone number): <br>
<cfinput
    type = "Text" name = "MyPhone"
    message = "Enter telephone number, formatted xxx-xxx-xxxx (e.g. 617-761-2000)"
    validate = "telephone" required = "yes">
    <font size = -1 color = red>Required</font>
<!--- Zip code validation. --->
<p>Zip Code Validation (enter a properly formatted zip code):<br>
<cfinput
    type = "Text" name = "MyZip"
    message = "Enter zip code, formatted xxxxx or xxxxx-xxxx"
    validate = "zipcode" required = "yes">
    <font size = -1 color = red>Required</font>
<!--- Range validation. --->
<p>Range Validation (enter an integer from 1 to 5): <br>
<cfinput
    type = "Text" name = "MyRange" range = "1,5"
    message = "You must enter an integer from 1 to 5"
    validate = "integer" required = "no">
<!--- Date validation. --->
<p>Date Validation (enter a properly formatted date):<br>
<cfinput
    type = "Text" name = "MyDate"
    message = "Enter a correctly formatted date (dd/mm/yy)"
    validate = "date" required = "no">
<input
    type = "Submit" name = ""
    value = "send my information">
</cfform>
```

例 : バインドあり

次の例では、バインドを使用して、名、姓、ドメイン名の入力コントロールに基づいてデフォルトの電子メールアドレスを生成し、その結果を電子メールテキスト入力フィールドに挿入します。

CFML ページには次のコードが含まれています。

```
<h3>CFC bind using onkeyup() event</h3>
<ul>
    <li>When you type in the First Name field, the first initial appears in
            the Email control.</li>
    <li>Each key you enter in the Last Name and Domain fields appears in the
            Email control.</li>
</ul>

<cfform name="mycfform">
    First Name<font color="#FF0000">*</font>:   
        <cfinput type="text" name="firstname" required="yes"><br>
    Last Name<font color="#FF0000">*</font>:   
        <cfinput type="text" name="lastname" required="yes"><br>
    Domain<font color="#FF0000">*
        </font>:         
        <cfinput type="text" name="domain" required="yes"><br>
    <br>
        <cfinput type="text" name="email1"
            bind="cfc:bindFcns.getEmailId({firstname@keyup},{lastname@keyup},
                {domain@keyup})">
    <br><br>
<font color="#FF0000">*</font> indicates the field is required.
</cfform>
```

bindFcns CFC には次のコードが含まれています。

```
<cfcomponent>
    <cffunction name="getEmailId" access="remote">
        <cfargument name="firstname">
        <cfargument name="lastname">
        <cfargument name="domain">
        <cfreturn "#left(firstname,1)#.#lcase(arguments.lastname)#@#
    lcase(domain)#">
    </cffunction>
</cfcomponent>
```

cfinsert

説明

ColdFusion フォームまたは Form スコープ内のデータからデータソースにレコードを挿入します。

カテゴリ

データベース操作タグ

シンタックス

```
<cfinsert
    dataSource = "data source name"
    tableName = "table name"
    fetchClientInfo = "true|false"
    formFields = "formfield1, formfield2, ..."
    password = "password"
    tableOwner = "owner"
    tableQualifier = "table qualifier"
    username = "user name">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfprocparam、cfprocresult、cfquery、cfqueryparam、cfstoredproc、cftransaction、cfupdate

履歴

ColdFusion 10：fetchClientInfo 属性と clientInfo 属性が追加されました。

ColdFusion MX: connectString、dbName、dbServer、dbtype、provider、および providerDSN 属性は非推奨になりました。ColdFusion 5 以降のリリースでは、これらは機能せず、エラーを引き起こす可能性があります。

属性

属性	必須 / オプション	デフォルト	説明
dataSource	必須		テーブルが含まれているデータソースです。
clientInfo	オプション		データベース接続で設定するクライアントのプロパティが含まれる構造体です。
tableName	必須		フォームフィールドを挿入するテーブルです。 ORACLE ドライバの場合 : 大文字で指定する必要があります。 Sybase ドライバの場合 : 大文字と小文字が区別されます。テーブルの作成時に使用した名前と大文字小文字を同じにする必要があります。
fetchClientInfo	オプション	false	true に設定した場合、最後のクエリによって渡された、キーと値のペアを持つ構造体が返されます。
formFields	オプション	(キーを除くフォーム上の全フィールド)	挿入するフォームフィールドのカンマ区切りリストです。指定しない場合は、フォーム内のすべてのフィールドが対象となります。 フォームフィールドがデータベース内の列名と一致しない場合は、エラーが発生します。 データベーステーブルのキーフィールドがフォームに存在している必要があります。キーフィールドは非表示になっている場合があります。
password	オプション		ODBC セットアップで指定されたパスワードよりも優先されます。
tableOwner	オプション		テーブル所有権をサポートしているデータソース (SQL Server、Oracle、Sybase SQL Anywhere など) の場合は、このフィールドを使用してテーブルの所有者を指定します。
tableQualifier	オプション		テーブル修飾子をサポートしているデータソースの場合は、このフィールドを使用してテーブルの修飾子を指定します。テーブル修飾子の内容は、ドライバによって異なります。SQL Server および Oracle の場合、修飾子は、テーブルが含まれているデータベースの名前を表します。Intersolv dBASE ドライバの場合、修飾子は DBF ファイルが置かれているディレクトリを表します。
username	オプション		ODBC セットアップで指定されたユーザー名よりも優先されます。

例

```
 <!--- This example shows how to use cfinsert instead of cfquery to put data in a
    datasource. --->
<!--- If form.POSTED exists, we insert new record, so begin cfinsert tag. --->
<cfif IsDefined ("form.posted")>
    <cfinsert dataSource = "cfdocexamples"
        tableName = "COMMENTS"
        formFields = "Email,FromUser,Subject,MessText,Posted">
    <h3><I>Your record was added to the database.</i></h3>
</cfif>

<cfif IsDefined ("form.posted")>
    <cfif Server.OS.Name IS "Windows NT">
        <cfinsert datasource="cfdocexamples" tablename="COMMENTS"
            formfields="EMail,FromUser,Subject,MessText,Posted">
    <cfelse>
        <cfinsert datasource="cfdocexamples" tablename="COMMENTS"
            formfields="CommentID,EMail,FromUser,Subject,MessText,Posted">
    </cfif>
    <h3><i>Your record was added to the database.</i></h3> </cfif>

<!--- Use a query to show the existing state of the database. --->
<cfquery name = "GetComments" dataSource = "cfdocexamples">
    SELECT
        CommentID, EMail, FromUser, Subject, CommtType, MessText, Posted, Processed
    FROM
        COMMENTS
</cfquery>

<html>
<head></head>
<h3>cfinsert Example</h3>
<p>First, show a list of the comments in the cfdocexamples datasource.
<!--- Show all the comments in the db. --->
<table>
    <tr>
        <td>From User</td><td>Subject</td><td>Comment Type</td>
        <td>Message</td><td>Date Posted</td>
    </tr>
    <cfoutput query = "GetComments">
        <tr>
            <td valign = top><a href = "mailto:#Email#">#FromUser#</A></td>
            <td valign = top>#Subject#</td>
            <td valign = top>#CommtType#</td>
            <td valign = top><font size = "-2">#Left(MessText, 125)#</font></td>
```

```
            <td valign = top>#Posted#</td>
        </tr>
    </cfoutput>
</table>
<p>Next, we'll offer the opportunity to enter a comment:
<!--- Make a form for input. --->
<form action = "cfinsert.cfm" method = "post">
    <pre>
    Email:   <input type = "Text" name = "email">
    From:    <input type = "Text" name = "fromUser">
    Subject: <input type = "Text" name = "subject">
    Message: <textarea name = "MessText" COLS = "40" ROWS = "6"></textarea>
    Date Posted:  <cfoutput>#DateFormat(Now())#</cfoutput>
    <!--- Dynamically determine today's date. --->
    <input type = "hidden"
        name = "posted" value = "<cfoutput>#Now()#</cfoutput>">
    </pre>
    <input type = "Submit"
        name = "" value = "insert my comment">
</form>
```

注意：cfinsert タグでは、パラメータ化されたクエリーは内部的に使用されます。

cfinterface

説明

一連の関数シグネチャで構成されるインターフェイスを定義します。インターフェイスには完全な関数定義は含まれません。代わりに、ColdFusion コンポーネント (CFC) で関数を実装します。このタグによって定義されたインターフェイスを使用すると、再利用可能なアプリケーションフレームワークの構造を作成できます。

履歴

ColdFusion 8: このタグが追加されました。

カテゴリ

アプリケーションフレームワークタグ、拡張タグ

シンタックス

```
<cfinterface
    displayName = "descriptive name"
    extends = "interfaceName1[,interfaceName2]..."
    Hint = "hint text">
        <cffunction ...>
            <cfargument ... >
            <cfargument ... >
            ...
        </cffunction>
        <cffunction ...>
            ...
        </cffunction>
        ...
</cfinterface>
```

関連項目

cfargument、cfcomponent、cffunction、GetComponentMetaData、IsInstanceOf

属性

属性	必須 / オプション	デフォルト	説明
displayName	オプション		イントロスペクションを使用してインターフェイスの記述名を示すときに表示される値です。
extends	オプション		このインターフェイスが拡張するインターフェイスのカンマ区切りリストです。インターフェイスを実装する CFC は、このプロパティで指定されたインターフェイスの関数もすべて実装する必要があります。 他のインターフェイスを拡張するインターフェイスが存在し、親インターフェイスと子インターフェイスの間で同じ名前の関数が指定されている場合は、両方の関数の属性が一致している必要があります。一致していない場合、エラーが発生します。
hint	オプション		イントロスペクションを使用してインターフェイスに関する情報を示すときに表示されるテキストです。hint 属性の値は、関数の説明のシンタックス行の次に表示されます。

使用方法

cfinterface タグは、そのインターフェイスを実装する ColdFusion コンポーネント (CFC) で定義する必要がある関数を宣言します。このインターフェイスは関数のシグネチャを指定しますが、関数は実装しません。代わりに、そのインターフェイスを実装する CFC で完全な関数定義を記述する必要があります。

たとえば、CRUD というインターフェイスを作成して、4 つのオペレーション (作成、読み込み、更新、削除) の基本シグネチャを定義したとします。このインターフェイスを実装するコンポーネントは、インターフェイスのシグネチャに準拠している必要があります。このコンポーネントを複数のインターフェイスで実装することにより、異なる種類のデータソースを管理できます。すべてのコンポーネントが同じインターフェイスを実装するため、各アプリケーションで使用するデータソースの種類に応じて簡単にコンポーネントを置き換えることができます。

インターフェイスを定義するには、.cfc という拡張子を持つ ColdFusion ファイルを作成し、そのファイルの唯一の最上位タグとして cfinterface タグを指定します。インターフェイス名はファイル名に基づいて決定されます。たとえば、myInterface.cfc は myInterface インターフェイスを定義します。cfinterface タグでは、どのような属性でも指定できますが、ColdFusion に対して意味を持つのは「属性」の表に記載されている名前のみです。ファイル名にカンマとピリオドを含めることはできません (ただし、拡張子 cfc の前のピリオドは除きます)。

cfinterface タグの本文では、インターフェイスの関数を宣言することによってインターフェイスを指定します。インターフェイスの定義は、次の基本ルールに準拠している必要があります。

- cfinterface タグの本文には、cffunction タグとコメントのみを含めることができます。
- cffunction タグの本文には、関数の引数を宣言する cfargument タグとコメントのみを含めることができます。
- cffunction タグの本文は省略可能です。

次の例は、一般的なインターフェイス定義の形式を示しています。

```
<cfinterface extends="IBasicInterface">
    <cffunction name="hello" description="Should print a greeting containing the input
         argument or 'world'.">
        <cfargument name="whom" type="string" default="world">
    </cffunction>
    <cffunction name="calculateTwo" returnType="numeric" output="no"
            description="calculates a result using two numbers and returns the result">
        <cfargument name="first" type="numeric" required="yes"/>
        <cfargument name="second" type="numeric" required="no" default="0"/>
    </cffunction>
    <cffunction name="disclaimer"/>
</cfinterface>
```

このインターフェイスは IBasicInterface インターフェイスを拡張します。したがって、このインターフェイスを実装するコンポーネントは、IBasicInterface インターフェイスのメソッドも実装する必要があります。このインターフェイスを実装するコンポーネントでは、次の 3 つの関数を定義する必要があります。

- hello 関数 : オプションで文字列型の引数を 1 つ取ります。デフォルト値は "world" です。
- calculateTwo 関数 : 数値型の引数を必ず 1 つ取り、オプションで数値型の引数を 1 つ取り (デフォルト値は 0)、数値を返します。
- disclaimer 関数 : 引数は取らず、任意の型を返します。

インターフェイスを実装する CFC では、cfcomponent タグの implements 属性を使用してインターフェイス名を指定します。そのコンポーネントは、インターフェイスの cffunction タグで指定されているすべてのメソッドを実装する必要があります。関数の引数の順序は、インターフェイス定義とコンポーネント定義の間で一致している必要があります。

次の表に、cffunction タグと cfargument タグで使用できる属性を示します。また、インターフェイスの定義とコンポーネントの実装で使用する場合の要件と制約について説明します。

属性	インターフェイスの要件	実装の要件
cffunction		
access	オプション、public のみ指定できます。	オプション、public または remote のみ指定できます。
description	オプション	インターフェイスの値と異なっていてもかまいません。
displayName	オプション	インターフェイスの値と異なっていてもかまいません。
hint	オプション	インターフェイスの値と異なっていてもかまいません。
name	必須	インターフェイスの値と一致している必要があります。
output	オプション	インターフェイスの値と一致している必要はありません。 インターフェイスでこの属性を省略していても、実装でこの属性を使用することができます。同様に、インターフェイスでこの属性を使用して、実装で省略することもできます。
returnType	オプション	インターフェイスの値と一致している必要があります。ただし、省略された type オプションと any のオプション値は同等で、ColdFusion は一致として扱います。
roles	指定できません。	任意の有効な値を指定できます。
cfargument		
default	オプション	インターフェイスの値と一致している必要があります。 インターフェイスでこの属性を省略した場合は、実装で任意の値を指定できます。
displayName	オプション	インターフェイスの値と異なっていてもかまいません。
hint	オプション	インターフェイスの値と異なっていてもかまいません。
name	必須	インターフェイスの値と一致している必要があります。
required	オプション	インターフェイスで yes と指定した場合は、この属性も yes と指定する必要があります。インターフェイスで no と指定した場合、またはこの属性を省略した場合は、no と指定するか、属性を省略することができます。
type	オプション	インターフェイスの値と一致している必要があります。ただし、省略された type オプションと any のオプション値は同等で、ColdFusion は一致として扱います。

1 つの CFC で複数のインターフェイスを実装することもできます。

注意：1 つの CFC で複数のインターフェイスを実装し、複数のインターフェイスで同じ名前の関数を定義する場合は、すべてのインターフェイスでその関数のシグネチャが一致している必要があります。ColdFusion は関数のオーバーロードをサポートしていません。

ColdFusion は、コンポーネントを検索する場合と同じルールを使用してインターフェイスを検索します。GetComponentMetaData 関数を使用して、インターフェイスに関する情報を取得できます。

ファイルに名前を付けるときに、その名前がインターフェイス名であることを識別できるようにしておくことをお勧めします。たとえば、ファイル名の先頭に大文字の I を付けておくと、インターフェイス名であることが識別しやすくなります。また、1 つのインターフェイスのみを実装するコンポーネントには、そのインターフェイスに似た名前を付けることをお勧めします (たとえば、最初の文字だけを大文字の C に変更します)。たとえば、インターフェイスファイルに IresourceInfo.cfc という名前を付けた場合は、対応するコンポーネントファイルに CresourceInfo.cfc という名前を付けます。

例

次の例では、加算、減算、乗算、除算の 4 つのオペレーションで構成される IBasicMath インターフェイスを定義します。integerMath CFC は、これらのオペレーションの整数計算バージョンを定義することにより、このインターフェイスを実装します。testMath.cfm アプリケーションは、integerMath 関数を使用して、2 つの 10 進数値に対し (pi と e の値を使用して) 算術演算を実行します。

練習として、任意の型の値を取るようにインターフェイスの定義を変更し、PrecisionEvaluate 関数を使用して任意精度の結果を返す別の CFC を実装することをお勧めします (これらのバージョンは省略してあります)。

IBasicMath.cfc ファイルに含まれるインターフェイスの定義は次のとおりです。

```
<cfinterface>
    <cffunction name = add returntype = "numeric" output = "no"
            description = "Add two values">
        <cfargument name = "first" type="numeric" required = "no" default ="0">
        <cfargument name = "second" type = "numeric" required = "no" default = "0">
    </cffunction>
    <cffunction name = subtract returntype = "numeric" output = "no"
            description = "Subtract two values">
        <cfargument name = "first" type = "numeric" required = "no" default = "0">
        <cfargument name ="second" type = "numeric" required = "no" default = "0">
    </cffunction>
    <cffunction name = multiply returntype = "numeric" output = "no"
            description = "Multiply two values">
        <cfargument name = "first" type = "numeric" required = "no" default = "0">
        <cfargument name = "second" type = "numeric" required = "no" default = "0">
    </cffunction>
    <cffunction name = divide returntype = "numeric" output = "no"
            description = "Divide two values">
        <cfargument name = "first" type = "numeric" required = "no" default="0">
        <cfargument name = "second" type="numeric" required = "no" default="1">
    </cffunction>
</cfinterface>
```

integerMath.cfc ファイルでは、IBasicMath インターフェイスを実装する integerMath コンポーネントを次のように定義します。

```
<cfcomponent implements = "IBasicMath" >
    <cffunction name = add returntype = "numeric" output = "no"
            description = "Add two values">
        <cfargument name = "first" type = "numeric" required = "no" default = "0">
        <cfargument name = "second" type = "numeric" required = "no" default = "0">
        <cfreturn Round(first + second)>
    </cffunction>
    <cffunction name = subtract returntype = "numeric" output = "no"
            description = "Subtract two values">
        <cfargument name = "first" type = "numeric" required = "no" default = "0">
        <cfargument name = "second" type = "numeric" required = "no" default = "0">
        <cfreturn Round(first - second)>
    </cffunction>
    <cffunction name = multiply returntype = "numeric" output = "no"
            description = "Multiply two values">
        <cfargument name = "first" type = "numeric" required = "no" default = "0">
        <cfargument name = "second" type = "numeric" required = "no" default = "0">
        <cfreturn Round(first * second)>
    </cffunction>
    <cffunction name = divide returntype = "numeric" output = "no"
description = "Divide two values">
        <cfargument name = "first" type = "numeric" required = "no" default = "0">
        <cfargument name = "second" type = "numeric" required = "no" default = "1">
        <cfreturn Round(first / second)>
    </cffunction>
</cfcomponent>
```

testMath.cfm ファイルは、次のように integerMath コンポーネントのメソッドを使用して整数値を計算します。

```
<cfscript>
    arguments = StructNew();
    arguments.first = pi();
    arguments.second = "2.71828182845904523536028747135 2";
</cfscript>

<cfobject name = "iMathObj" component = "integerMath">
<cfoutput>
<h3>Function Arguments</h3>
argument 1: #arguments.first#<br>
argument 2: #arguments.second#<br>

<h3>Addition</h3>
#iMathObj.add(argumentCollection = arguments)#
<h3>Subtraction</h3>
#iMathObj.subtract(argumentCollection = arguments)#
<h3>Multiplication</h3>
#iMathObj.multiply(argumentCollection = arguments)#
<h3>Division</h3>
#iMathObj.divide(argumentCollection = arguments)#
</cfoutput>
```

cfinvoke

説明

次のいずれかを行います。

- ColdFusion ページまたは ColdFusion コンポーネント内からコンポーネントメソッドを呼び出します。
- Web サービスを呼び出します。

このタグは、次のような処理を行います。

- 一時的にコンポーネントまたは Web サービスのインスタンスを作成し、そのメソッドを呼び出します。
- インスタンス化されているコンポーネントまたは Web サービスのメソッドを呼び出します。

このタグでは、次の方法でメソッドにパラメータを渡すことができます。

- cfinvokeargument タグを使用します。
- 1 つのパラメータにつき 1 つの属性を、名前付き属性と値のペアとして渡します。
- argumentCollection 属性で構造体として渡します。

カテゴリ

拡張タグ

シンタックス

```
 <!--- Syntax 1: This syntax invokes a method of a component. --->
<cfinvoke
    component = "component name or reference"
    method = "method name"
    returnVariable = "variable name"
    argumentCollection = "argument collection"
    ...>

OR

<!--- Syntax 2: This syntax can invoke a method of a component only from within the component. --->
<cfinvoke
    method = "method name"
    returnVariable = "variable name"
    argumentCollection = "argument collection"
    ...>

OR

<!--- Syntax 3: This syntax invokes a web service. --->
<cfinvoke
    webservice = "Web service name or WSDL URL"
    method = "operation name"
    password = "password"
    proxyPassword = "password for proxy server"
    proxyPort = "port on proxy server"
    proxyServer = "WSDL proxy server URL"proxyUser = "user ID for proxy server"
    returnVariable = "variable name"
    refreshWSDL = "yes|no"
    servicePort = "WSDL port name"
    timeout = "request timeout in seconds"
    username = "user name"
    wsdl2javaArgs = "argument string">

OR

<!--- Syntax 4A: This syntax invokes a component.
    This syntax shows instantiation with the cfobject tag.
    This cfinvoke syntax applies to instantiating a component
    with the cfobject tag and to instantiating a component
    with the CreateObject function. --->
<cfobject
    component = "component name"
    name = "name for instantiated component">
```

```
    <cfinvoke
    <!--- Value is object name, within number signs. --->
    component = "#name of instantiated component#"
    method = "method name"
    returnVariable = "variable name"
    argumentCollection = "argument collection"
    ...>

OR

<!--- Syntax 4B: This syntax invokes a web service.
    This syntax shows instantiation with the cfobject tag.
    This cfinvoke syntax applies to instantiating a web service with the cfobject tag and to
    instantiating a web service with the CreateObject function. --->
<cfobject
    webservice = "web service name or WSDL URL"
    name = "name for instantiated object"
    (optional cfobject attibutes)>
<cfinvoke
    webservice = "#cfobject name attribute value#"
    method = "method name"
    password = "password"
    proxyPassword = "password for proxy server"
    proxyPort = "port on proxy server"
    proxyServer = "name or IP address of WSDL proxy server"
    proxyUser = "user ID for proxy server"
    returnVariable = "variable name"
    refreshWSDL = "yes|no"
    servicePort = "WSDL port name"
    timeout = "request time-out in seconds"
    username = "user name"
    wsdl2javaArgs = "argument string">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfargument、cfcomponent、cffunction、cfinvokeargument、cfobject、cfproperty、cfreturn

履歴

ColdFusion 8: refreshWSDL 属性と wsdl2javaArgs 属性が追加されました。

ColdFusion MX 7: servicePort 属性が追加されました。

ColdFusion MX 6.1: timeout、proxyServer、proxyPort、proxyUser、および proxyPassword の各属性が追加されました。

ColdFusion MX: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
argumentCollection	オプション		構造体の名前です。メソッドに渡す引数の連想配列です。
component	「使用方法」を参照		文字列またはコンポーネントオブジェクトです。コンポーネントへの参照、またはインスタンス化するコンポーネントを指定します。
input_params ...			入力パラメータです。名前付き入力パラメータごとに **paramName=paramValue** を指定します。

属性	必須 / オプション	デフォルト	説明
method	「使用方法」を参照		メソッドの名前です。Web サービスの場合は、オペレーションの名前です。
password	オプション	存在する場合には、Administrator で設定されたパスワード	Web サービスにアクセスするためのパスワードです。Administrator で設定された Web サービス名が webservice 属性で指定されている場合は、Administrator のエントリで指定されたユーザー名ではなく、ここで指定されたユーザー名が使用されます。
proxyPassword	オプション	存在する場合には、http.proxyPassword システムプロパティ	プロキシサーバー上で使用するユーザーのパスワードです。
proxyPort	オプション	存在する場合には、http.proxyPort システムプロパティ	プロキシサーバー上で使用するポートです。
proxyServer	オプション	存在する場合には、http.proxyHost システムプロパティ	webservice の URL にアクセスするために必要なプロキシサーバーです。
proxyUser	オプション	存在する場合には、http.proxyUser システムプロパティ	プロキシサーバーに送信するユーザー ID です。
refreshWSDL	オプション	no	• yes: WSDL ファイルをリロードし、Web サービスを利用するために必要な生成結果を再生成します。 • no
returnVariable	オプション		呼び出しの結果を表す変数の名前です。
servicePort	オプション	WSDL で最初に見つかったポート	Web サービスのポート名です。この値は、service 要素内の port 要素の name 属性に対応します。大文字と小文字が区別されます。 Web サービスに複数のポートが含まれている場合、この属性を指定します。
timeout	オプション	0 (タイムアウトなし)	Web サービスリクエストのタイムアウトです (単位 : 秒)。
username	オプション	存在する場合には、Administrator で設定されたユーザー名	Web サービスにアクセスするためのユーザー名です。Administrator で設定された Web サービス名が webservice 属性で指定されている場合は、Administrator のエントリで指定されたユーザー名ではなく、ここで指定されたユーザー名が使用されます。

属性	必須 / オプション	デフォルト	説明
webservice	「使用方法」を参照		次のいずれかです。 • Web サービス WSDL の絶対 URL • ColdFusion Administrator で Web サービスに割り当てられた名前 (文字列)
wsdl2javaArgs	「使用方法」を参照		WSDL2Java ツールに渡す引数のスペース区切りリストを含む文字列です。WSDL2Java ツールは Web サービス用の Java スタブを生成するためのツールです。使用可能な引数は次のとおりです。 • -W または --noWrapped: wrapped document/literal スタイルオペレーションの特別な処理を無効にします。 • -a または --all: 参照されないものも含め、WSDL 内の全要素のコードを生成します。 • -w または --wrapArrays: ラップされた XML 配列型に対してストレートな配列を作成するのではなく bean を作成します。このスイッチは Axis のマニュアルには記載されていません。 有効な引数の詳細については、『Apache Axis WSDL2Java Reference』を参照してください。
wsVersion	オプション		使用する軸のバージョンを指定します。wsversion を 1 に指定した場合は、軸 1 が使用されます。2 にした場合は、軸 2 が使用されます。

注意 : プロキシサーバーの属性を指定せず、対応するシステムプロパティが設定された場合 (通常は JVM スタートアップ引数)、そのシステムプロパティ値が使用されます。

使用方法

どのシンタックスでどの属性が使用できるかを次の表にまとめます。

各属性の指定要件 (必須、オプション、無視、または無効):	cfinvoke タグのシンタックス :				
	シンタックス 1	シンタックス 2	シンタックス 3	シンタックス 4A	シンタックス 4B
argumentCollection	オプション	オプション	オプション	オプション	オプション
component	必須	オプション	無効	必須	無効
input_params ...	オプション	オプション	オプション	オプション	オプション
method	必須	必須	必須	必須	必須
password	無視	無視	オプション	無視	オプション
proxyPassword	無効	無効	オプション	無効	オプション
proxyPort	無効	無効	オプション	無効	オプション
proxyServer	無効	無効	オプション	無効	オプション
proxyUser	無効	無効	オプション	無効	オプション
returnVariable	オプション	オプション	オプション	オプション	オプション
servicePort	無効	無効	オプション	無効	オプション
timeout	無効	無効	オプション	無効	オプション

各属性の指定要件 (必須、オプション、無視、または無効):	cfinvoke タグのシンタックス :				
	シンタックス 1	シンタックス 2	シンタックス 3	シンタックス 4A	シンタックス 4B
username	無視	無視	オプション	無視	オプション
webservice	無効	無効	必須	無効	必須
wsdl2javaArgs	無効	無効	オプション	無効	オプション

component 属性でコンポーネント名を指定した場合は、対応する名前を持つコンポーネントのインスタンスが作成され、リクエストされたメソッドが呼び出されます。その後すぐ、そのコンポーネントのインスタンスは廃棄されます。インスタンス化されているコンポーネントオブジェクトへの参照をこの属性で指定した場合は、コンポーネントのインスタンス作成や廃棄は行われません。

UNIX システムでは、まず指定のコンポーネント名と同じ名前 (ただしすべて小文字) のファイルが検索されます。該当するファイルがない場合は、大文字と小文字もまったく同じ名前のファイルが検索されます。

メソッド引数は次のどの方法でも渡すことができます。同じ名前の引数が複数の方法で渡された場合は、次に示すとおりの優先順序が適用されます。

1. cfinvokeargument タグを使用します。
2. cfinvoke タグの属性として直接渡します。このとき、cfinvoke タグの登録済み属性と同じ名前 (method、component、webservice、returnVariable) を使用することはできません。
3. argumentCollection 属性を使用して、構造体キーとして渡します。

たとえば、params 構造体に a=1、b=1、c=1 という 3 つのキーが含まれているとします。次の呼び出しは、引数が a=3、b=2、c=1 という順序でメソッドに渡された場合と同様に評価されます。

```
<cfinvoke ... a=2 b=2 argumentCollection=params>
        <cfinvokeargument name="a" value="3">
    </cfinvoke>
```

注意：cfinvoke タグでは、component、method、argumentCollection、および result の各属性名は予約されています。これらの名前は、引数名として使用できません。

例 1

この例では、シンタックス 1 を使用しています。

```
<!--- Immediate instantiation and destruction. --->
<cfinvoke
    component="nasdaq.quote"
    method="getLastTradePrice"
    returnVariable="res">
    <cfinvokeargument
        name="symbol"
        value="macr">
</cfinvoke>
<cfoutput>#res#</cfoutput>
```

例 2

この例では、シンタックス 1 を使用しています。

```
<!--- Passing the arguments using argumentCollection. --->
<cfset args = StructNew()>
<cfset args.symbol = "macr">
<cfinvoke
    component="nasdaq.quote"
    method="getLastTradePrice"
    argumentCollection="#args#"
    returnVariable="res">
<cfoutput>#res#</cfoutput>
```

例 3

この例では、シンタックス 2 を使用しています。

```
<!--- Called only from within a component, MyComponent. --->
<cfinvoke
    method = "a method name of MyComponent"
    returnVariable = "variable name">
```

例 4

この例では、シンタックス 3 を使用しています。

```
<!--- Using cfinvoke to consume a web service using a ColdFusion component. --->
<cfinvoke
    webservice="http://www.xmethods.net/sd/2001/TemperatureService.wsdl"
    method="getTemp"
    returnvariable="aTemp">
<cfinvokeargument name="zipcode" value="55987"/>
</cfinvoke>
<cfoutput>The temperature at zip code 55987 is #aTemp#</cfoutput>
```

Web サービスの詳細については、『ColdFusion アプリケーションの開発』の Using Web Services を参照してください。

例 5

この例では、シンタックス 4A を使用しています。

```
<!--- Separate instantiation and method invocation; useful for multiple invocations using
    different methods or values. --->
<cfobject
    name="quoteService"
    component="nasdaq.quote">
<cfinvoke
    component="#quoteService#"
    method="getLastTradePrice"
    symbol="macr"
    returnVariable="res_macr">
<cfoutput>#res#</cfoutput>
<cfinvoke
    component="#quoteService#"
    method="getLastTradePrice"
    symbol="mot"
    returnVariable="res_mot">
<cfoutput>#res#</cfoutput>
```

cfinvokeargument

説明

パラメータの名前と値をコンポーネントメソッドまたは Web サービスに渡します。このタグは、cfinvoke タグ内で使用します。

カテゴリ
拡張タグ

シンタックス
```
<cfinvokeargument
    name="argument name"
    value="argument value"
    omit = "yes|no">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目
cfargument、cfcomponent、cffunction、cfinvoke、cfobject、cfproperty、cfreturn

履歴
ColdFusion MX 7: omit 属性が追加されました。

ColdFusion MX: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
name	必須		引数名です。
value	必須		引数の値です。
omit	オプション	no	Web サービスを呼び出すときに、パラメータを省略できるようにします。cfinvoke の webservice 属性を指定していない場合に omit="yes" を指定すると、エラーになります。 yes: Web サービスを呼び出すときに、このパラメータを省略します。 no: Web サービスを呼び出すときに、このパラメータを省略しません。

使用方法
cfinvoke タグ本文には複数の cfinvokeargument タグを記述できます。

cfinvokeargument タグを使用すると、渡す引数をダイナミックに決定できます。たとえば、条件式に基づいた引数名の決定や、cfif タグを利用した cfinvokeargument タグを実行するかどうかの決定ができます。

Web サービスを呼び出す場合、omit 属性を "yes" に設定すると、パラメータを省略できます。WSDL で、引数が nillable であると指定されている場合、関連付けられている引数は null に設定されます。WSDL で minoccurs=0 と指定されている場合、その引数は WSDL から省略されます。

例 1
```
<cfinvoke
    component="nasdaq.quote"
    method="getLastTradePrice"
    returnVariable="res">
    <cfinvokeargument name="symbol" value="mot">
    <cfinvokeargument name="symbol" value="macr">
</cfinvoke>

<cfoutput>#res#</cfoutput>
```

例 2

```
<!--- Using cfinvoke to consume a web service using a ColdFusion component. --->
<cfinvoke
    webservice="http://www.xmethods.net/sd/2001/TemperatureService.wsdl"
    method="getTemp"
    returnvariable="aTemp">
        <cfinvokeargument name="zipcode" value="55987"/>
</cfinvoke>
<cfoutput>The temperature at zip code 55987 is #aTemp#</cfoutput>
```

タグ j ～ l

cflayout

説明

コンテナ (ブラウザウィンドウや cflayoutarea タグ) の領域を作成します。ボーダー付きの領域、水平または垂直方向に配置されたボックス、タブ付きナビゲータなど、特定のレイアウト動作を適用できます。

カテゴリ

表示管理タグ

シンタックス

```
<cflayout
    type="accordion|border|hbox|tab|vbox"
    activeOnTop="false|true"
    align="center|justify|left|right"
    fillHeight="true|false"
    fitToWindow="true|false"
    height="integer"
    name="string"
    padding="integer"
    style="CSS style specification"
    tabHeight="measurement"
    tabPosition="top|bottom"
    tabStrip="true|false"
    titleCollapse="true|false"
    width="integer">

        cflayoutarea tags

</cflayout>
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfajaximport、cfdiv、cflayoutarea、cfpod、cfwindow、『ColdFusion アプリケーションの開発』の Using Ajax User Interface Components and Features

履歴

ColdFusion 9:

- type 属性の accordion 値、および activeOnTop、fillHeight、titleCollapse 属性が追加されました。

- height 属性と width 属性が hbox タイプと vbox タイプでサポートされるようになりました。

ColdFusion 8: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	適用対象	説明
type	必須		すべて	レイアウトのタイプです。使用できる値は次のとおりです。 • accordion: 複数のレイアウト領域があるコンテナです。一度に 1 つの領域だけが表示されます。各レイアウト領域には、常にタイトルバーが表示されます。ユーザーは、レイアウト領域のタイトルバーで [+] または [-] ボタンをクリックすることによって、各レイアウト領域を展開したり折り畳んだりすることができます。 • border: ボーダー付きで、最大 5 つのレイアウト領域を含むボックスです。それぞれのレイアウト領域にボーダーが付きます。詳細については、「使用方法」を参照してください。 • hbox: 水平方向のボックスです。直下の子であるすべての cflayoutarea コントロールが水平に配置されます。 • tab: タブ付きの表示です。現在の子の cflayoutarea タグが、レイアウトの表示領域を占有します。各レイアウト領域にはタブが付き、ユーザーはタブを選択して各領域のコンテンツを表示できます。 • vbox: 垂直方向のボックスです。直下の子であるすべての cflayoutarea コントロールが垂直に配置されます。
activeOnTop	オプション	false	accordion	アクティブなパネルをレイアウトの一番上に移動して最初のパネルにするかどうかを指定します。
align	オプション	ブラウザで設定されているレイアウト方向	すべて	子レイアウト領域のコンテンツに適用するデフォルトの配置を指定します。それぞれの cflayoutarea タグで alignment 属性が指定されている場合は、この値よりも優先されます。 使用できる値は次のとおりです。 • center • justify • left • right
fillHeight	オプション	true	accordion	アクティブなレイアウト領域の高さを調整してレイアウトコントロールコンテナ内の使用可能なスペースを占めるようにするかどうかを指定するブール値です。 • true • false fillHeight に false を指定すると、overflow が hidden に設定されます。

属性	必須 / オプション	デフォルト	適用対象	説明
fitToWindow	オプション	false	border	ボーダーレイアウトが、ウィンドウの幅および高さの 100% を占めるようにするかどうかを指定するブール値です。 • true • false 基盤となる実装で viewport が使用されると、レイアウトの外側にある内容がレイアウト内に納まりません。 ネストされたボーダーレイアウトを使用している場合、fitToWindow を true に設定すると、ボーダーレイアウト内にネストされているレイアウトが、使用可能なスペースに自動的に納められません。このため、ネストされたレイアウトの場合は、fitToWindow ではなく width または size を使用することをお勧めします。
height	オプション	ボーダーレイアウトの場合は 600、その他の場合は自動設定	すべて	レイアウトの高さです (単位 : ピクセル)。 タブレイアウトの場合、height 属性には、tabheight 属性と同じ機能があります。height 属性および tabheight 属性の両方を指定した場合、height のほうが tabheight よりも優先されます。ここで指定した高さの値は、style 属性を使用して指定した高さの値よりも優先されます。
name	オプション		すべて	レイアウト領域の名前です。ページ内で一意である必要があります。
padding	オプション	0	hbox、vbox	• hbox レイアウトの場合は、それぞれの子レイアウト領域の右側に追加するパディングを指定します。 • vbox レイアウトの場合は、それぞれの子レイアウト領域の下部に追加するパディングを指定します。 この属性では、有効な CSS の長さ形式またはパーセント形式 (10、10%、10px、10em など) をすべて使用できます。 パディングは子レイアウト領域に追加され、そのレイアウト領域のスタイルが適用されます。
style	オプション		すべて	レイアウトのスタイルを定義する CSS スタイル指定です。
tabHeight	オプション		tab	すべての子レイアウト領域のコンテンツ領域に適用する高さを指定します。この設定は、個々の cflayoutarea タグの style 属性に高さ設定を指定して上書きできます。
tabPosition	オプション	top	tab	タブ領域のコンテンツを基準としたタブの相対的な位置を指定します。 • bottom: レイアウトの下部にタブが表示されます。 • top: レイアウトの上部にタブが表示されます。
tabStrip	オプション	true	tab	true の場合、背景タブストリップが表示されます。
titleCollapse	オプション	true	accordion	各レイアウト領域のタイトルバーの任意の場所をクリックすることによってレイアウト領域を展開したり折り畳んだりするかどうかを指定します。false の場合、ユーザーは、タイトルバーの [+] または [-] ボタンをクリックしてレイアウト領域を展開したり折り畳んだりする必要があります。
width	オプション		すべて	レイアウトの幅です (単位 : ピクセル)。この値は、style 属性を使用して定義された幅よりも優先されます。値が指定されていない場合、autoWidth が適用されるため、コンテンツが画面全体に表示されます。

使用方法

cflayout タグの直下の子は、cflayoutarea タグ、または本文の最上位レベルに cflayoutarea タグが含まれる非表示タグである必要があります。たとえば、cflayout タグの子として cfloop や cfquery を使用でき、これらのタグの本文に cflayoutarea タグを含めることもできます。

border タイプのレイアウトには次の特徴があります。

- レイアウトコントロールと直下の子レイアウト領域が、それぞれボーダーで囲まれます。
- コントロールには、レイアウトの左、右、中央、上部、下部に配置される子を 5 つまで格納できます。
- ユーザーが子レイアウト領域を展開または折り畳んだり、完全に閉じることができるように、子レイアウト領域にスプリッタを設定できます (中央の領域を除く)。
- 中央の子レイアウト領域は、レイアウト内の他のレイアウト領域が使用していないスペースをすべて占有します。
- レイアウトの高さを指定するには、style 属性の height 設定を使用します。

  注意：DOCTYPE 宣言があるページにボーダー付きレイアウトを指定すると、高さが正しく設定されないため、cflayout タグの style 属性で高さを指定する必要があります。

  次の JavaScript 関数を使用すると、border および tab タイプの cflayout コントロールに対して、基盤となる Ext JS - JavaScript Library オブジェクトにアクセスできます。

関数	説明
ColdFusion.Layout.getBorderLayout	指定された border タイプの cflayout コントロールに対して、基盤となる Ext JS - JavaScript Library オブジェクトを取得します。
ColdFusion.Layout.getTabLayout	指定された tab タイプの cflayout コントロールに対して、基盤となる Ext JS - JavaScript Library オブジェクトを取得します。
ColdFusion.Layout.getAccordionLayout	指定されたアコーディオンタイプの cflayout コントロールに対して、基盤となる Ext JS - JavaScript Library オブジェクトを取得します。

  レイアウト領域の設定の詳細については、cflayoutarea を参照してください。

例

次の例は、ネストされたレイアウトのセットを示しています。外側のレイアウトは 2 つのレイアウト領域を持つ vbox です。上部のレイアウト領域にはボーダー付きレイアウトが配置されます。下部のレイアウト領域にはボタンを含むフォームが配置され、このボタンでボーダー付きレイアウト領域の表示を制御できます。

```
<html>
<head>
</head>
<body>
<cflayout name="outerlayout" type="vbox">
    <cflayoutarea style="height:400;">
        <cflayout name="thelayout" type="border">
            <!--- The 100% height style ensures that the background color fills
                the area. --->
            <cflayoutarea position="top" size="100" splitter="true"
                    style="background-color:##00FFFF; height:100%">
                This is text in layout area 1: top
            </cflayoutarea>
            <cflayoutarea title="Left layout area" position="left"
                    closable="true"
                    collapsible="true" name="left" splitter="true"
                    style="background-color:##FF00FF; height:100%">
                This is text in layout area 2: left<br />
                You can close and collapse this area.
            </cflayoutarea>
            <cflayoutarea position="center"
                    style="background-color:##FFFF00; height:100%">
                This is text in layout area 3: center<br />
            </cflayoutarea>
            <cflayoutarea position="right" collapsible="true"
```

```
                    title="Right Layout Area" initcollapsed="true"
                    style="background-color:##FF00FF; height:100%" >
                This is text in layout area 4: right<br />
                You can collapse this, but not close it.<br />
                It is initially collapsed.
            </cflayoutarea>
            <cflayoutarea position="bottom" size="100" splitter="true"
                     style="background-color:##00FFFF; height:100%">
                This is text in layout area 5: bottom
            </cflayoutarea>
        </cflayout>
    </cflayoutarea>

    <cflayoutarea style="height:100; ; background-color:##FFCCFF">
        <h3>Change the state of Area 2</h3>
        <cfform>
            <cfinput name="expand2" width="100" value="Expand Area 2" type="button"
                onClick="ColdFusion.Layout.expandArea('thelayout', 'left');">
            <cfinput name="collapse2" width="100" value="Collapse Area 2" type="button"
                onClick="ColdFusion.Layout.collapseArea('thelayout', 'left');">
            <cfinput name="show2" width="100" value="Show Area 2" type="button"
                onClick="ColdFusion.Layout.showArea('thelayout', 'left');">
            <cfinput name="hide2" width="100" value="Hide Area 2" type="button"
                onClick="ColdFusion.Layout.hideArea('thelayout', 'left');">
        </cfform>
    </cflayoutarea>
</cflayout>
</body>
</html>
```

cflayoutarea

説明

タブ付きレイアウトの各タブなど、cflayout タグ本文内の領域を定義します。

カテゴリ

表示管理タグ

シンタックス

```
 In a border layout
<cflayoutarea
    required
    position="bottom|center|left|right|top"
    optional
    align="left|center|justify|right"
    collapsible="false|true"
    initcollapsed="false|true"
    inithide="false|true"
    maxSize="number of pixels"
    minSize="number of pixels"
    name="string"
    onBindError = "JavaScript function name"
    overflow = "auto|hidden"
    size="number of pixels"
    source="URL"
    splitter="false|true"
    style="CSS style specification"
    title="string">

    area elements

</cflayoutarea>

In a hbox or vbox layout
<cflayoutarea
    optional
    name="string"
    onBindError = "JavaScript function name"
    overflow = "auto|hidden|scroll|visble"
    size="number of pixels"
    source="URL"
    style="CSS style specification">

    area elements

</layoutarea>

In a tab layout
<cflayoutarea
    optional
    bindonload="false|true"
    closable="false|true"
    disabled="false|true"
    inithide="false|true"
    name="string"
    onBindError = "JavaScript function name"
    overflow = "auto|hidden|scroll|visble"
    refreshOnActivate = "false|true"
    selected="false|true"
    source="URL"
    style="CSS style specification"
    tabTip="text"
    title="string">

    area elements
```

```
</layoutarea>

In an accordion layout
<cflayoutarea
    optional
    bindonload="false|true"
    closable="false|true"
    name="string"
    onBindError = "JavaScript function name"
    overflow = "auto|hidden|scroll|visble"
    refreshOnActivate = "false|true"
    selected="false|true"
    source="URL"
    style="CSS style specification"
    title="string"
    titleicon="icon location">

    area elements
```

source 属性を指定した場合は、すべての子タグが無視されます。子タグを含めない場合は、タグを /> で閉じます。

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfdiv、cflayout、cfpod、cfwindow、AJAX JavaScript 関数、『ColdFusion アプリケーションの開発』の Using layouts

履歴

ColdFusion 8: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	適用対象	説明
align	オプション	cflayout タグの align 属性の値	すべて	レイアウト領域内の子コントロールをどのように配置するかを指定します。使用できる値は次のとおりです。 • center • justify • left • right
bindLoad	オプション	true	tab、accordion	layoutarea が初めてロードされたときに source 属性の式を実行するかどうかを指定するブール値です。
closable	オプション	false	tab	領域を閉じることが可能かどうかを指定するブール値です。この属性を指定すると、タブまたはタイトルバーに x のアイコンが表示されます。ユーザーはこのアイコンをクリックして領域を閉じることができます。
collapsible	オプション	false	border、accordion	領域を折り畳むことが可能かどうかを指定するブール値です。この属性を指定すると、タイトルバーに >> または << のアイコンが表示されます。ユーザーはこのアイコンをクリックして領域を折り畳むことができます。 position 属性の値が center に設定されているボーダー付きレイアウト領域では、この属性を使用できません。

属性	必須 / オプション	デフォルト	適用対象	説明
disabled	オプション	false	tab	タブを無効にするかどうか (ユーザーがタブを選択してそのコンテンツを表示できるようにするかどうか) を指定するブール値です。無効なタブはグレーで表示されます。 selected 属性の値が true の場合は無視されます。
initCollapsed	オプション	false	border	初期状態で領域を折り畳むかどうかを指定するブール値です。 position 属性の値が center に設定されているボーダー付きレイアウト領域では、この属性を使用できません。 collapsible 属性の値が false の場合は無視されます。
initHide	オプション	false	border、tab、accordion	初期状態で領域を非表示にするかどうかを指定するブール値です。初期状態で非表示になっている領域を表示するには、ColdFusion.Layout.showArea または ColdFusion.Layout.showTab 関数を使用します。 position 属性の値が center に設定されているボーダー付きレイアウト領域では、この属性を使用できません。
maxSize	オプション	-1 (制限なし)	border	position 属性の値が top または bottom の場合は、スプリッタをドラッグして設定できる領域の高さの最大値です (ピクセル単位)。 position 属性の値が left または right の場合は、領域の幅の最大値です。 position 属性の値が center に設定されているボーダー付きレイアウト領域では、この属性を使用できません。
minSize	オプション	-1 (制限なし)	border	position 属性の値が top または bottom の場合は、スプリッタをドラッグして設定できる領域の高さの最小値です (ピクセル単位)。 position 属性の値が left または right の場合は、領域の幅の最小値です。 position 属性の値が center に設定されているボーダー付きレイアウト領域では、この属性を使用できません。
name	オプション		すべて	レイアウト領域の名前です。
onBindError	オプション	「説明」を参照	すべて	バインド式の評価でエラーが発生した場合に実行する JavaScript 関数の名前です。この関数は 2 つの属性 (HTTP ステータスコードとメッセージ) を取る必要があります。 この属性を省略し、(ColdFusion.setGlobalErrorHandler 関数を使用して) グローバルエラーハンドラを指定すると、エラーメッセージが表示されます。それ以外の場合は、デフォルトのエラーポップアップが表示されます。

属性	必須 / オプション	デフォルト	適用対象	説明
overflow	オプション	auto アコーディオンの場合、cflayout タグの fillheight 属性が false に設定されていると、デフォルト値は hidden になります。	すべて	子コンテンツのサイズが大きいためにコントロールがウィンドウの境界を超えてしまう場合に、子コンテンツをどのように表示するかを指定します。使用できる値は次のとおりです。 • auto: 必要に応じてスクロールバーを表示します。 • hidden: 境界からはみ出したコンテンツにアクセスできないようにします。 • scroll: 必要がない場合でも常に水平および垂直方向のスクロールバーを表示します。 • visible: レイアウト領域の外側にコンテンツを表示します。 **注意事項 :** • ボーダー付きレイアウト内のレイアウト領域で visible または scroll を指定することはできません。 • Internet Explorer では、レイアウト領域を visible に設定した場合、レイアウト領域を超えてコンテンツを表示するのではなく、コンテンツのサイズに合わせてレイアウト領域が拡張されます。
position	cflayout タイプが border の場合は必須		border	レイアウト内での領域の位置です。有効な値は次のとおりです。 • top: レイアウト全体の上部に領域を配置します。 • bottom: レイアウト全体の下部に領域を配置します。 • left: レイアウトの左側に領域を配置します。上部と下部に可視の領域がある場合は、その間に配置します。 • right: レイアウトの右側に領域を配置します。上部と下部に可視の領域がある場合は、その間に配置します。 • center: 上部、下部、左側、右側の領域に重ならないスペースに領域を配置します。 ボーダー付きスタイルのレイアウトでは、各タイプのレイアウト領域を 1 つずつ指定できます。
refreshOnActivate	オプション	false	tab、accordion	• true: バインドイベントの発生時に加え、タブまたはパネル表示領域が表示される (たとえば、ユーザーがタブを選択する) たびに source バインド式を実行して、タブまたはアコーディオンパネルのコンテンツを更新します。 • false: バインド式がバインドイベントによって実行された場合のみ、タブまたはアコーディオンパネル表示領域を更新します。 この属性を使用する場合は、source 属性も指定します。
selected	オプション	最初のタブが選択された状態	tab、accordion	このタブを最初から選択された状態にして、そのコンテンツをレイアウト内に表示するかどうかを指定するブール値です。

属性	必須 / オプション	デフォルト	適用対象	説明
size	オプション	-1 初期サイズをダイナミックに計算	border、hbox、vbox	hbox レイアウト、および position 属性の値が top または bottom に設定されているボーダー付きレイアウトの場合は、領域の初期の高さです。 vbox レイアウト、および position 属性の値が left または right に設定されているボーダー付きレイアウトの場合は、領域の初期の幅です。 hbox および vbox レイアウトの場合は、有効な CSS の長さ形式またはパーセント形式 (10、10%、10px、10em など) をすべて使用できます。 border レイアウトの場合は、ピクセル数を整数で指定する必要があります。 position 属性の値が center に設定されているボーダー付きレイアウト領域では、この属性を使用できません。中央のサイズは他のレイアウト領域のサイズに基づいて自動的に決定されます。 **メモ :** ボーダー付きレイアウト内のレイアウト領域に HTML 形式の cftree タグなどの AJAX コントロールのみが含まれる場合は、size 属性を指定します。それ以外の場合、AJAX コンポーネントはレイアウト領域のサイズが変更されるまで表示されません。
source	オプション		すべて	レイアウト領域のコンテンツを返す URL です。ColdFusion では標準のページパス解決規則が使用されます。この属性では依存関係を持つバインド式を使用できます。 この属性で指定したファイルに AJAX 機能を使用するタグ (cfform、cfgrid、cfpod など) が含まれている場合は、cflayoutarea タグを含むページで cfajaximport タグを使用します。詳細については、cfajaximport を参照してください。 source 属性の詳細については、「使用方法」を参照してください。
splitter	オプション	false	border	レイアウト領域とそれに隣接する layoutarea コントロールの間にスプリッタを配置するかどうかを指定するブール値です。ユーザーは、このスプリッタをドラッグして領域の相対サイズを変更できます。 position 属性が left または right に設定されているレイアウト領域でこの属性を true に設定した場合は、スプリッタによってその領域と隣接領域のサイズが水平方向に変更されます。position 属性が top または bottom に設定されているレイアウト領域でこの属性を true に設定した場合は、スプリッタによってレイアウトのサイズが垂直方向に変更されます。 position 属性の値が center に設定されているボーダー付きレイアウト領域では、この属性を使用できません。
style	オプション		すべて	領域の外観を制御する CSS スタイル指定です。
tabTip	オプション		tab	定義されている場合、タブヒントが表示されます。
title	オプション (タブ付きレイアウトの場合は必須)		border、tab、accordion	タブ付きレイアウトの場合は、タブに表示されるテキストです。 ボーダー付きレイアウトの場合は、この属性を指定すると、レイアウト領域のタイトルバーが作成され、指定したテキストがタイトルとして使用されます。デフォルトでは、closable または collapsible が true に設定されていないボーダー付きレイアウトの場合、タイトルバーは作成されません。 position 属性の値が center に設定されているボーダー付きレイアウト領域では、この属性を使用できません。
titleicon	オプション		accordion	タイトルとともに表示するアイコンの位置を指定します。

使用方法

cflayoutarea タグは常に cflayout タグの子にする必要があります。cflayoutarea タグの直下の子として cflayoutarea タグを含めることはできませんが、cflayout タグを含めることは可能です。ただし、cflayoutarea タグを cflayout タグの直下の子にする必要はありません。代わりに、cflayout タグの子として cfloop や cfquery を使用でき、cfloop や cfquery タグの本文に cflayoutarea タグを含めることができます。このルールにより、さまざまなレイアウトを複雑に組み合わせることが可能になります。

注意：tab タイプのレイアウトの中に border タイプのレイアウトを配置することはできません。

size 属性の値を指定しなかった場合は、レイアウト領域のコンテンツに必要なサイズが自動的に計算されます。ただし、レイアウト領域に AJAX コントロールが含まれている場合など、必要なサイズを計算できないこともあります。AJAX コントロールが表示されるようにするには、size 属性を指定する必要があります。その場合は、レイアウト領域にスクロールバーが表示されます。

レイアウト領域のコンテンツを指定するには、source 属性またはタグ本文を使用します。両方とも指定されている場合は、source 属性で指定されたコンテンツが使用され、タグ本文は無視されます。source 属性を使用すると、コンテンツの取得中にアニメーションのアイコンと「ロード中 ...」というテキストが表示されます。

JavaScript 関数の定義を含むページを source 属性で指定する場合、そのページの関数定義は次の形式に準拠している必要があります。

```
functionName = function(arguments) {function body}
```

次のような関数定義は、正常に動作しない可能性があります。

```
function functionName (arguments) {function body}
```

ただし、カスタム JavaScript すべてを外部の JavaScript ファイルにインクルードして、アプリケーションのメインページにインポートすることをお勧めします。source 属性を使用して取得するコードには、カスタム JavaScript をインラインで記述しないでください。インポートするページには、関数定義に関するこのような制限はありません。

source 属性を使用する場合は、バインド式を使用してフォームフィールドの値やその他のフォームコントロール属性をソース指定の一部として含めることができます。バインド可能なフォームコントロールは、HTML 形式のフォームコントロールのみです。バインド式の使用方法の詳細については、『ColdFusion アプリケーションの開発』の Using AJAX Data and Development Features を参照してください。

border タイプのレイアウトでは、position 属性を center に設定した cflayoutarea タグを指定していない場合でも、他の領域で使用されていないスペースが中央のレイアウト領域によってすべて占有されます。したがって、いずれかの方向に 2 つのレイアウト領域のみを配置する場合は、一方の領域を中央の領域として指定するか、2 つの領域のサイズを明示的に指定してレイアウト領域全体が完全に埋まるようにする必要があります。

レイアウトをネストする場合は、内側のレイアウト領域が必ず表示されるように初期サイズを正しく設定します。

次の JavaScript 関数を使用すると、レイアウト領域の有効化、無効化、表示、非表示、展開、折り畳み、選択などが可能です。

関数	説明
ColdFusion.Layout.createTab	既存のタブ付きレイアウトにタブを作成します。
ColdFusion.Layout.disableTab	特定のタブを無効にして選択できないようにします。
ColdFusion.Layout.enableTab	特定のタブを有効にし、ユーザーがそのタブを選択して領域のコンテンツを表示できるようにします。
ColdFusion.Layout.hideTab	タブを非表示にします。
ColdFusion.Layout.selectTab	タブを選択して、レイアウト領域のコンテンツを表示します。
ColdFusion.Layout.showTab	inithide 属性または hideTab() 関数によって非表示にされたタブを表示します。
ColdFusion.Layout.collapseArea	ボーダー付きレイアウトの領域を折り畳みます。

関数	説明
ColdFusion.Layout.expandArea	ボーダー付きレイアウトの折り畳まれている領域を展開します。
ColdFusion.Layout.getTabLayout	ボーダー付きレイアウトの領域を非表示にします。
ColdFusion.Layout.hideArea	ボーダー付きレイアウトの領域を非表示にします。
ColdFusion.Layout.showArea	inithide 属性または hideArea() 関数によって非表示にされたボーダー付きレイアウトの領域を表示します。
ColdFusion.Layout.hideAccordion	アコーディオンを非表示にします。
ColdFusion.Layout.showAccordion	inithide 属性または hideArea() 関数によって非表示にされたアコーディオンを表示します。
ColdFusion.Layout.selectAccordion	アコーディオンを選択して、レイアウト領域のコンテンツを表示します。
ColdFusion.Layout.collapseAccordion	アコーディオンレイアウト領域を折り畳みます。
ColdFusion.Layout.expandAccordion	アコーディオンレイアウトの折り畳まれている領域を展開します。

注意：style 属性を使用してボーダー付きレイアウト領域の背景色を指定するとき、レイアウト領域全体を背景色で塗りつぶすには、height スタイルを 100% に設定します。スタイル指定はレイアウト領域そのものではなく、レイアウト領域内のコンテンツ領域に適用されます。そのため、100% を指定するとレイアウト領域全体がコンテンツ領域によって占有されます。

例

次の例では、3 つのタブ付きレイアウトを作成し、ボタンを使用して 2 番目のタブをダイナミックに制御します。

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
</head>
<body>
<h3>Atab</h3>
<cflayout type="tab" name="thelayout" tabheight="175" style="background-color:##CCffFF;
        color:red; height:200">
    <cflayoutarea title="Tab 1" style="background-color:##FFAAFF;" closable="true">
        This is text in layout area 1
    </cflayoutarea>
    <cflayoutarea name="area2" title="Tab 2" inithide="true"
            style="background-color:##FFCCFF" >
        This is text in layout area 2
    </cflayoutarea>
    <cflayoutarea title="Tab 3" style="background-color:##FF99FF;">
        This is text in layout area 3
    </cflayoutarea>
</cflayout>
<br />
<cfform>
    <cfinput name="show" width="40" value="show tab" type="button"
        onClick="ColdFusion.Layout.showTab('thelayout', 'area2');">
    <cfinput name="hide" width="40" value="hide tab" type="button"
        onClick="ColdFusion.Layout.hideTab('thelayout', 'area2');">
    <cfinput name="enable" width="40" value="enable tab" type="button"
        onClick="ColdFusion.Layout.enableTab('thelayout', 'area2');">
    <cfinput name="disable" width="40" value="disable tab" type="button"
        onClick="ColdFusion.Layout.disableTab('thelayout', 'area2');">
    <cfinput name="select" width="40" value="select tab" type="button"
        onClick="ColdFusion.Layout.selectTab('thelayout', 'area2');">
</cfform>
</body>
</html>
```

cfldap

説明

Netscape Directory Server などの LDAP (Lightweight Directory Access Protocol) ディレクトリサーバーへのインターフェイスを提供します。

カテゴリ

インターネットプロトコルタグ

シンタックス

```
<cfldap
    action = "action"
    server = "server name"
    attributes = "attribute, attribute"
    delimiter = "delimiter character"
    dn = "distinguished name"
    filter = "filter"
    maxRows = "number"
    modifyType = "replace|add|delete"
    name = "name"
    password = "password"
    port = "port number"
    rebind = "yes|no"
    referral = "number of allowed hops"
    returnAsBinary = "column name, column name"
    scope = "scope"
    secure = "multifield security string"
    separator = "separator character"
    sort = "attribute[, attribute]..."
    sortControl = "nocase|desc|asc"
    start = "distinguished name"
    startRow = "row number"
    timeout = "milliseconds"
    username = "user name">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfftp、cfhttp、cfmail、cfmailparam、cfpop、『ColdFusion アプリケーションの開発』の Managing LDAP Directories

履歴

ColdFusion 8: returnAsBinary 属性で変数のリストを指定するときに、区切り文字としてカンマを使用できるようになりました。たとえば、returnAsBinary="objectGUID,objectSID" のように指定できます。以前は、区切り文字として使用できるのは空白のみでした。

ColdFusion MX 7: returnAsBinary 属性が追加されました。SSL V2 クライアントベースの認証が可能になり、ColdFusion で CFSSL_CLIENT_AUTH オプションがサポートされるようになりました。CFSSL_CLIENT_AUTH が選択されている場合、ColdFusion は cacerts (認証データベース) の最初の証明書にクライアント証明書が格納されているものと想定します。

ColdFusion MX:

- name 属性の動作が変更されました。このタグでは、name 属性のクエリー名が検証されます。

- ソートの動作が変更されました。このタグでは、クライアントサイドでのクエリー結果のソートがサポートされません。サーバーサイドでのソートはサポートされています (sort および sortcontrol 属性を使用します)。
- 結果のソート方法が変更されました。サーバーサイドのソート結果は、ColdFusion 5 の場合と少し異なります。ソートがサポートされていないサーバーに対してソートを実行しようとすると、ColdFusion MX でエラーが発生します。
- filterConfig 属性および filterFile 属性は非推奨になりました。これらの属性は今後のリリースでは機能せず、エラーが発生する可能性があります。

属性

属性	必須 / オプション	デフォルト	説明
action	必須	query	• query: LDAP エントリ情報のみを返します。name、start、および attributes 属性が必要です。 • add: LDAP エントリを LDAP サーバーに追加します。attributes 属性が必要です。 • modify: LDAP サーバー上で LDAP エントリを修正します (識別名 dn 属性は除く)。dn 属性が必要です。modifyType 属性を参照してください。 • modifyDN: LDAP サーバー上で LDAP エントリの識別名属性を修正します。dn 属性が必要です。 • delete: LDAP サーバー上の LDAP エントリを削除します。dn 属性が必要です。
server	必須		LDAP サーバーのホスト名または IP アドレスです。
attributes	action="Query"、"Add"、"ModifyDN"、または "Modify" の場合は必須		クエリーの場合は、取得したい属性のカンマ区切りリストを指定します。クエリーの場合、すべての属性を取得するには "*" を指定します。 action="add" または "modify" の場合は、更新する列のリストを指定できます。属性をセミコロンで区切って指定します。 action="ModifyDN" の場合は、シンタックスチェックを行わずに属性がそのまま LDAP サーバーに渡されます。
delimiter	オプション	; (セミコロン)	属性の名前と値のペアの区切り文字です。この属性は次の場合に使用します。 • attributes 属性で複数の項目を指定する場合。 • 属性にデフォルトの区切り文字 (セミコロン) が含まれる場合。例 : mgrpmsgrejecttext;lang-en query、add、modify アクションにより、また、cfldap により、複数の値を持つ属性を出力するために使用されます。 たとえば、$ (ドル記号) の場合、"cn=DoubleTreeInn$street=1111Elm;Suite100 のように指定できます。ここでは、セミコロンは番地名の一部です。
dn	action="Add"、"Modify"、"ModifyDN"、または "delete" の場合は必須		update アクションの識別名です。例 : "cn=BobJensen,o=AceIndustry,c=US"
filter	オプション	"objectclass = *"	action="query" の場合の検索条件です。 属性を "(attributeoperatorvalue)" という形式でリストします。例 : "(sn=Smith)"

属性	必須 / オプション	デフォルト	説明
maxRows	オプション		LDAP クエリーに含まれるエントリの最大数です。
modifyType	オプション	replace	複数値リスト内で属性を処理する方法を指定します。 • add: 指定された属性を既存の属性に追加します。 • delete: 属性のセットから属性を削除します。 • replace: 属性を、指定された属性に置き換えます。 既に存在する属性や空の属性を追加することはできません。
name	action="Query" の場合は必須		LDAP クエリーの名前です。このタグは、この値を検証します。
password	secure="CFSSL_BASIC" の場合は必須		ユーザー名に対応するパスワードです。 secure ="CFSSL_BASIC" の場合、V2 によりパスワードが送信前に暗号化されます。
port	オプション	389	ポートです。
rebind	オプション	no	• yes: リファラルコールバックの再バインドを試み、元の証明情報を使用して、参照アドレスでクエリーを再発行します。 • no: 参照接続は匿名です。
referral	オプション		整数。リファラルで許可されるホップの数です。値を 0 に設定すると、LDAP 用の参照アドレスが無効になります。したがって、データは返されません。
returnAsBinary	オプション		バイナリ値として返される列の空白区切りリストです。
scope	オプション	oneLevel	action="Query" の場合に、start 属性に指定したエントリからどこまでを検索スコープにするかを指定します。 • oneLevel: 1 つ下のレベルにあるエントリ • base: そのエントリのみ • subtree: エントリとその下の全レベル
secure	オプション		使用するセキュリティと、必要な情報です。この属性を指定する場合、値は CFSSL_BASIC である必要があります。これにより、V2 SSL の暗号化とサーバー認証が可能になります。
separator	オプション	, (カンマ)	複数の値を持つ属性の、属性値の区切り文字です。query、add、modify アクションにより、また、cfldap により、複数の値を持つ属性を出力するために使用されます。 たとえば、$ (ドル記号) に設定すると、attributes 属性に "objectclass=top$person" という値を指定できます。この場合、objectclass の最初の値は top に、2 番目の値は person になります。これにより、値にカンマが含まれる場合の混乱を回避します。
sort	オプション		クエリー結果のソートの基準にする属性です。カンマ区切りで指定します。
sortControl	オプション	asc	• nocase: 大文字と小文字を区別しないでソートします。 • asc: 大文字と小文字を区別し、昇順 (a から z へ) でソートします。 • desc: 大文字と小文字を区別し、降順 (z から a へ) でソートします。 sortControl="nocase,asc" のように、ソートタイプを組み合わせて入力することもできます。

属性	必須 / オプション	デフォルト	説明
start	action="Query" の場合は必須		検索を開始するために使用するエントリの識別名です。
startRow	オプション	1	action="query" とともに使用します。ColdFusion クエリーに挿入する LDAP クエリーの最初の行です。
timeout	オプション	60000	LDAP 処理の最大待機時間をミリ秒単位で指定します。
username	secure="CFSSL_BASIC" の場合は必須	(匿名)	ユーザー ID です。

使用方法

query アクションを使用すると、cfldap によってクエリーオブジェクトが作成されるため、次のようにしてクエリー変数の情報にアクセスできます。

変数名	説明
queryname.recordCount	クエリーによって返されるレコードの数です。
queryname.currentRow	cfoutput が現在処理しているクエリーの行です。
queryname.columnList	クエリーの列名です。

security="CFSSL_BASIC" オプションを使用した場合、ColdFusion は、サーバーの資格情報と ColdFusion で使用される JRE の jre/lib/security/cacerts キーストア内の情報とを比較することで、サーバーを信頼するかどうかを判断します。ColdFusion のデフォルトの cacerts ファイルには、数多くの認証機関についての情報が含まれています。このファイルに情報を追加して更新する必要がある場合は、ColdFusion の "jre/bin" ディレクトリにある keytool ユーティリティを使用して、X.509 形式の証明書をインポートできます。たとえば、次の行を入力します。

```
keytool -import -keystore cacerts -alias ldap -file ldap.crt -keypass bl19mq
```

次に ColdFusion を再起動します。キーツールユーティリティの初期の keypass パスワードは、"change it" です。keytool ユーティリティの使用方法の詳細については、Sun JDK のマニュアルを参照してください。

ColdFusion では無効な文字が、LDAP の属性名に使用されていることがあります。結果として、cfldap タグが作成するクエリー結果セットの中に、CFML でアクセスできない無効な文字を含む名前の列が含まれる可能性があります。ColdFusion では、無効な文字は自動的にアンダースコア文字にマッピングされます。したがって、クエリー結果セットの列名は LDAP 属性の名前とは正確には一致しない場合があります。

使用例については、『ColdFusion アプリケーションの開発』を参照してください。

例

```
 <h3>cfldap Example</h3>
<p>Provides an interface to LDAP directory servers. The example uses the
University of Connecticut public LDAP server. For more public LDAP servers,
see <a href="http://www.emailman.com">http://www.emailman.com</a>.</p>
<p>Enter a name and search the public LDAP resource.
An asterisk before or after the name acts as a wildcard.</p>
<!--- If form.name exists, the form was submitted; run the query. --->
<cfif IsDefined("form.name")>
    <!--- Check to see that there is a name listed. --->
    <cfif form.name is not "">
        <!--- Make the LDAP query. --->
        <cfldap
             server = "ldap.uconn.edu"
             action = "query"
             name = "results"
             start = "dc=uconn,dc=edu"
             filter = "cn=#name#"
             attributes = "cn,o,title,mail,telephonenumber"
             sort = "cn ASC">
        <!--- Display results. --->
        <center>
        <table border = 0 cellspacing = 2 cellpadding = 2>
            <tr>
                <th colspan = 5>
                    <cfoutput>#results.recordCount# matches found </cfoutput></TH>
            </tr>
            <tr>
                <th><font size = "-2">Name</font></TH>
                <th><font size = "-2">Organization</font></TH>
                <th><font size = "-2">Title</font></TH>
                <th><font size = "-2">E-Mail</font></TH>
                <th><font size = "-2">Phone</font></TH>
            </tr>
            <cfoutput query = "results">
                <tr>
                    <td><font size = "-2">#cn#</font></td>
                    <td><font size = "-2">#o#</font></td>
                    <td><font size = "-2">#title#</font></td>
                    <td><font size = "-2">
                        <A href = "mailto:#mail#">#mail#</A></font></td>
                    <td><font size = "-2">#telephonenumber#</font></td>
                </tr>
            </cfoutput>
            </table>
            </center>
        </cfif>
</cfif>

<form action="#cgi.script_name#" method="POST">
    <p>Enter a name to search in the database.</p>
    <input type="Text" name="name">
    <input type="Submit" value="Search" name="">
</form>
```

cflocation

説明

現在のページの実行を停止し、ColdFusion ページまたは HTML ファイルを開きます。

カテゴリ
フロー制御タグ、ページ処理タグ

シンタックス
```
<cflocation
    url = "URL"
    addToken = "yes|no"
    statusCode = "300|301|302|303|304|305|307">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目
cfabort、cfbreak、cfexecute、cfexit、cfif、cfloop、cfswitch、cfthrow、cftry

履歴
ColdFusion 8: statusCode 属性が追加されました。

属性

属性	必須 / オプション	デフォルト	説明
url	必須		開く HTML ファイルまたは CFML ページの URL です。
addToken	オプション		clientManagement 属性を有効にする必要があります。cfapplication を参照してください。 • *yes*: URL にクライアント変数情報を追加します。 • *no*
statusCode	オプション		次のような HTTP ステータスコードです。 • 300 HTTP_MULTIPLE_CHOICES: リクエストされたアドレスが複数のエンティティを指しています。 • 301 HTTP_MOVED_PERMANENTLY: ページに新しい URI が割り当てられます。この変更は恒久的なものです。 • 302 HTTP_MOVED_TEMPORARILY: ページに新しい URI が割り当てられます。この変更は一時的なものです。 • 303 HTTP_SEE_OTHER: 別のネットワークアドレスを試してください。 • 304 HTTP_NOT_MODIFIED: リクエストされたリソースは変更されていません。 • 305 HTTP_USE_PROXY: リクエストされたリソースは、Location フィールドで指定されたプロキシ経由でアクセスする必要があります。 • 307 HTTP_TEMPORARY_REDIRECT: リクエストされたデータは一時的に別の場所に置かれています。 ステータスコードが 304 ～ 307 の場合は、URL で指定されたページにリダイレクトされません。リダイレクトされるようにするには、最新の HTTP RFC で定められているガイドラインに従う必要があります。

使用方法
基本的なメッセージやレスポンスをファイルに書き込んでおき、複数のアプリケーションから呼び出すことができます。このタグを使用して、ユーザーのブラウザを標準ファイルにリダイレクトします。

このタグをページ内の cfflush タグの後に記述しても、効果はありません。

注意： cfabort、cflocation または cfcontent タグを使用した場合は、OnRequestEnd ではなく OnAbort メソッドが呼び出されます。

例

```
<h3>cflocation Example</h3>
<p>This tag redirects the browser to a web resource; normally, you would use this tag to go to a CF page or
an HTML file on the same server. The addToken attribute lets you send client information to the target
page.</p>
<p>If you remove the comments, this code redirects you to CFDOCS home page:</p>

<!--- <cflocation url = "http://localhost:8500/cfdocs/dochome.htm" addToken = "no"> --->
```

cflock

説明

共有データの整合性を維持します。次のロックを適用します。

- Exclusive: 本文内の CFML 構文に対してシングルスレッドのアクセスを行うことができます。このタグの本文は、一度に 1 つのリクエストでしか実行できません。あるリクエストが排他的ロックをかけている間は、タグ内のコードを他のリクエストが開始することはできません。ColdFusion は、排他的ロックを早い者勝ちで適用します。
- Read-only: 複数のリクエストがタグ本文内の CFML 構文に同時にアクセスできます。読み取り専用ロックは、共有データを読み取る場合にのみ使用し、共有データを修正する場合には使用しないでください。あるリクエストが共有データの排他的ロックを取得している場合、新規リクエストはその排他的ロックが解除されるのを待機することになります。

カテゴリ

アプリケーションフレームワークタグ

シンタックス

```
<cflock
    timeout = "time-out in seconds"
    name = "lock name"
    scope = "Application|Server|Session|Request"
    throwOnTimeout = "yes|no"
    type = "readOnly|exclusive">
    <!--- CFML to be synchronized. --->
</cflock>
```

注意： このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfapplication、cfassociate、cfmessagebox、『ColdFusion アプリケーションの開発』の Using Persistent Data and Locking

履歴

ColdFusion 8: scope 属性の値として Request が追加されました。

属性

属性	必須 / オプション	デフォルト	説明
timeout	必須		ロックを取得するときに待機する最大時間を秒単位で指定します。ロックが取得されると、タグの実行は継続します。ロックを取得できない場合の動作は、throwOnTimeout 属性の値によって異なります。timeout="0" を指定した場合、タイムアウトは ColdFusion Administrator の [設定] ページでの " リクエストタイムアウト **x**" の設定 (この設定が有効なとき) により決まります。ただし、この設定が有効でない場合に timeout="0" を指定すると、ColdFusion はロックを取得するまで無限に待機することになります。
name	オプション		ロック名です。scope 属性と併用することはできません。特定の名前を持つ cflock タグ内のコードを実行できるのは、一度に 1 つのリクエストだけに限られます。空の文字列は指定できません。 アプリケーションのさまざまな部分からのリソースへのアクセスを同期化できます。ロック名は、ColdFusion サーバーに対してグローバルです。ロック名は、アプリケーションおよびユーザーセッション間で共有できますが、クラスタ化された複数のサーバー間にわたって共有することはできません。
scope	オプション		ロックのスコープです。name 属性とは排他の関係です。指定のスコープ内では、一度に 1 つのリクエストだけがこのタグ内のコード (または同じロックスコープの別の cflock タグ内のコード) を実行できます。 • Application • Request • Server • Session
throwOnTimeout	オプション	yes	タイムアウト条件を処理する方法です。 • yes: タイムアウト時に例外が生成されます。 • no: このタグの時間を経過しても実行が継続されます。
type	オプション	exclusive	• readOnly: 複数のリクエストで共有データを読み取ることができます。 • exclusive: 1 つのリクエストで共有データを読み書きできます。

注意：共有データの構造体、ファイル、および CFX を更新するコードのスコープを制限してください。排他的ロックはこれらの更新の整合性を確保するために必要ですが、読み取り専用ロックのほうが高速に処理されます。パフォーマンスを重視するアプリケーションの場合は、共有データを読み取るときなどは、できるかぎり排他的ロックではなく読み取り専用ロックを使用してください。

使用方法

ColdFusion はマルチスレッドサーバーです。したがって、一度に複数のページリクエストを処理することができます。cflock タグは、次の目的で使用します。

- 同時に実行されるリクエストでの共有データやオブジェクトに対する修正が、整然とした順序で行われるようにします。
- ファイル操作の構造をこのタグで囲み、ファイルの更新時に他のアプリケーションやタグがそのファイルを書き込み用に開いていることがないようにします。
- CFX 呼び出しをこのタグで囲み、スレッドセーフな方法で実装されていない CFX を ColdFusion で安全に呼び出せるようにします (これは、C++ で開発された CFX にのみ該当します)。

  ColdFusion を安全に動作させるには、共有 (グローバル) データ構造体の管理と操作を行う C++ CFX がスレッドセーフでなければなりません。しかし、これには高度な知識が必要です。CFML カスタムタグラッパーで CFX を囲むと、呼び出しをスレッドセーフにすることができます。

共有スコープで変数の表示、設定、更新を行う場合は、scope 属性を使用して Server、Application、Session のいずれかを特定してください。

デッドロック

デッドロックとは、ページ内のロックされた部分のコードをどのリクエストも実行できない状態です。デッドロックが発生すると、どのユーザーもロックを解除できなくなります。なぜなら、ロックタイムアウトによってデッドロックが解消されるまで、ページ内の保護されたセクションへのリクエストはすべて拒否されるからです。

cflock タグでは、カーネルレベルの同期オブジェクトを使用します。このオブジェクトは、それを使用しているスレッドがタイムアウトするか異常終了した時点で、自動的に解放されます。したがって、ColdFusion が cflock タグの処理中に永遠にデッドロックされることはありません。しかし、タイムアウト時間が長すぎると、リクエストスレッドが長時間にわたってブロックされ、スループットが急激に低下する可能性があります。これを避けるには、常に最低限のタイムアウト値を使用してください。

リクエストスレッドがブロックされる別の原因としては、cflock タグのネスト構造やロック名が一貫していないことが考えられます。ロックをネストする場合は、ロックされた変数にアクセスするコードが、一貫して同じ順序で cflock タグをネストする必要があります。そうしないと、デッドロックが発生する可能性があります。

次に、デッドロックが発生する例を示します。

ユーザーが 2 人の場合のデッドロックの例	
ユーザー 1	**ユーザー 2**
Session スコープをロックします。	Application スコープをロックします。
デッドロック : Application スコープをロックしようとしますが、Application スコープはユーザー 2 によって既にロックされています。	デッドロック : Session スコープをロックしようとしますが、Session スコープはユーザー 1 によって既にロックされています。

読み取りロックの中に排他的ロックをネストしようとすると、次のようなデッドロックが発生する可能性があります。

ユーザーが 1 人の場合のデッドロックの例
ユーザー 1
Session スコープを読み取りロックでロックします。
Session スコープを排他的ロックでロックしようとします。
デッドロック : Session スコープは既に読み取りロックでロックされているので、Session スコープを排他的ロックでロックすることはできません。

次のコードは、この事例を示しています。

```
<cflock timeout = "60" scope = "SESSION" type = "readOnly">
    ..............
    <cflock timeout = "60" scope = "SESSION" type = "Exclusive">
    .........
    </cflock>
</cflock>
```

デッドロックを避けるには、ロックをネストするすべてのコードが、一貫した順序でネストし、一貫した名前を付ける必要があります。Server、Application、および Session スコープへのアクセスをロックする必要がある場合は、次の順序に従ってください。

1. Session スコープをロックします。cflock タグ内で、scope = "session" を指定します。
2. Application スコープをロックします。cflock タグで scope = "Application" を指定します。
3. Server スコープをロックします。cflock タグ内で、scope = "server" を指定します。
4. Server スコープのロックを解除します。

5 Application スコープのロックを解除します。

6 Session スコープのロックを解除します。

注意：スコープをロックする必要がない場合は、そのスコープのロックとロック解除の手順を省略できます。たとえば、Server スコープをロックする必要がない場合は、手順 3 と 4 を省略できます。名前付きロックにも同様のルールが適用されます。

詳細については、次の資料を参照してください。

- 『ColdFusion アプリケーションの開発』の Using Persistent Data and Locking
- 『ColdFusion アプリケーションの開発』の Locking thread data and resource access (cfthread タグを使用してマルチスレッド ColdFusion アプリケーションを作成するときに Request スコープをロックする方法についての説明)
- Adobe Web サイトの ColdFusion Locking Best Practices

例

```
<!---
This example shows how cflock can guarantee consistency of data updates to variables in the
    Application, Server, and Session scopes. --->

<!--- Copy the following code into an Application.cfm file in the
    application root directory. --->
<!----------------   Beginning of Application.cfm code   ---------------->
<!--- cfapplication defines scoping for a ColdFusion application and enables or disables
    storing of application and session variables. Put this tag in a special file called
    Application.cfm. It is run before any other ColdFusion page in its directory. --->

<!--- Enable session management for this application. --->
<cfapplication name = "ETurtle"
    sessionTimeout = #CreateTimeSpan(0,0, 0, 60)#
    sessionManagement = "yes">

<!--- Initialize session and application variables used by E-Turtleneck. Use session scope
    for the session variables. --->
<cflock scope = "Session"
    timeout = "30" type = "Exclusive">
    <cfif NOT IsDefined("session.size")>
        <cfset session.size = "">
    </cfif>
    <cfif NOT IsDefined("session.color")>
        <cfset session.color = "">
    </cfif>
</cflock>

<!--- Use an application lock for the application-wide variable that keeps track of the
    number of turtlenecks sold. For a more efficient, but more complex, way of handling
    Application scope locking, see the "Developing ColdFusion Applications"--->
<cflock scope = "Application" timeout = "30" type = "Exclusive">
    <cfif NOT IsDefined("application.number")>
        <cfset application.number = 0>
    </cfif>
</cflock>

<!----------------------- End of Application.cfm ----------------------->

<h3>cflock Example</h3>

<cfif IsDefined("form.submit")>
<!--- The form has been submitted; process the request. --->
    <cfoutput>
```

```
        Thanks for shopping E-Turtleneck. You chose size <b>#form.size#</b>,
        color <b>#form.color#</b>.<br><br>
    </cfoutput>

<!--- Lock the code that assigns values to session variables. ----->
    <cflock scope = "Session" timeout = "30" type = "Exclusive">
        <cfparam name = session.size Default = #form.size#>
        <cfparam name = session.color Default = #form.color#>
    </cflock>

<!---- Lock the code that updates the Application scope number of turtlenecks sold. --->
    <cflock scope = "Application" timeout = "30" type = "Exclusive">
        <cfset application.number = application.number + 1>
    <cfoutput>
        E-Turtleneck has now sold #application.number# turtlenecks!
    </cfoutput>
    </cflock>

<cfelse>
<!--- Show the form only if it has not been submitted. --->
    <cflock scope = "Application" timeout = "30" type = "Readonly">
        <cfoutput>
            E-Turtleneck has sold #application.number# turtlenecks to date.
        </cfoutput>
    </cflock>

    <form method="post" action="cflocktest.cfm">
        <p>Congratulations! You selected the most comfortable turtleneck in the world.
        Please select color and size.</p>
        <table cellspacing = "2" cellpadding = "2" border = "0">
            <tr>
                <td>Select a color.</td>
                <td><select type = "Text" name = "color">
                        <option>red
                        <option>white
                        <option>blue
                        <option>turquoise
                        <option>black
                        <option>forest green
                    </select>
                </td>
            </tr>
            <tr>
                <td>Select a size.</td>
                <td><select type = "Text" name = "size" >
                        <option>XXsmall
                        <option>Xsmall
                        <option>small
                        <option>medium
                        <option>large
                        <option>Xlarge
                    </select>
                </td>
            </tr>
            <tr>
                <td>Press Submit when you are finished making your selection.</td>
                <td><input type = "Submit" name = "submit" value = "Submit"> </td>
            </tr>
        </table>
    </form>
</cfif>
```

cflog

説明
メッセージをログファイルに書き込みます。

カテゴリ
データ出力タグ

シンタックス
```
<cflog
    text = "text"
    type = "information|warning|error|fatal"
    application = "yes|no"
    file = "filename"
    log = "log type">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目
cfcol、cfcontent、cfoutput、cftable

履歴
ColdFusion MX: thread、date、および time 属性は非推奨になりました。これらの属性は今後のリリースでは機能せず、エラーが発生する可能性があります。以前のリリースでは、これらの属性によって、それぞれのデータ項目がログに出力されるかどうかが決まっていました。ColdFusion MX では、常にこのデータが出力されます。

属性

属性	必須 / オプション	デフォルト	説明
text	必須		ロギングするメッセージテキストです。
application	オプション	yes	• yes: cfapplication タグまたは "Application.cfc" ファイルでアプリケーション名が指定されている場合は、アプリケーション名を記録します。 • no

属性	必須 / オプション	デフォルト	説明
file	オプション		メッセージファイルです。ファイル名の拡張子以外の部分だけを指定します。たとえば、"Testing.log" ファイルにロギングするときは、"Testing" と指定します。 このファイルは、デフォルトのログディレクトリに格納されます。ディレクトリパスは指定できません。指定したファイルが存在しない場合は、自動的にそのファイルが作成されて .log という拡張子が付けられます。
log	オプション		file 属性を省略した場合は、メッセージは標準ログファイルに書き込まれます。file 属性を指定した場合は、この属性が無視されます。 • Application: Application.log に記録します。通常は、アプリケーション固有のメッセージに対して使用されます。 • Scheduler: Scheduler.log に記録します。通常は、スケジューリングされたタスクの実行をロギングするときに使用されます。
type	オプション	Information	メッセージのタイプ (厳格度) です。 • Information • Warning • Error • Fatal

使用方法

このタグは、カスタムメッセージを標準ログファイルまたはカスタムログファイルにロギングします。ログメッセージ用のファイルを指定することも、デフォルトのアプリケーションログやスケジューラログにメッセージを送信することもできます。ログメッセージには ColdFusion の式を含めることができます。ログファイルは拡張子 ".log" を持ち、ColdFusion のログディレクトリに格納される必要があります。

ログエントリは、次のフィールドから成るカンマ区切りリストとして記述されます。

- type
- thread
- date
- time
- application
- text

値は二重引用符 (") で囲まれます。application 属性に no を指定すると、リスト内の対応するエントリは空になります。

cflog タグの実行を無効にすることができます。詳細については、ColdFusion Administrator の [基本セキュリティ] ページを参照してください。

次の例では、アプリケーションにログオンするユーザーの名前をロギングします。このメッセージは ColdFusion ログディレクトリの "myAppLog.log" ファイルにロギングされます。このログには、日付、時刻、およびスレッド ID が含まれますが、アプリケーション名は含まれません。

```
<cflog file="myAppLog" application="no"
    text="User #Form.username# logged on.">
```

たとえば、ユーザーがフォームのユーザー名フィールドに「Sang Thornfield」と入力した場合は、次のエントリが "myApplog.log" ファイルエントリに追加されます。

"Information","153","02/28/01","14:53:40",,"User Sang Thornfield logged on."

cflogin

説明

ユーザーログインと認証コードのためのコンテナです。ユーザーがまだログインしていない場合に、このタグ内のコードが実行されます。このタグ内には、ユーザーを認証し、そのユーザーのロールを識別するためのコードを記述します。cfloginuser タグとともに使用します。

カテゴリ

セキュリティタグ

シンタックス

```
<cflogin
    applicationToken = "token"
    cookieDomain = "domain"
    idletimeout = "value">
    ...
    <cfloginuser
        name = "name"
        password = "password"
        roles = "roles">
</cflogin>
```

関連項目

cfloginuser、cflogout、GetAuthUser、GetUserRoles、IsUserInAnyRole、IsUserInRole、IsUserLoggedIn、『ColdFusion アプリケーションの開発』の Securing Applications

履歴

ColdFusion 8: applicationtoken 属性を使用して、アプリケーションごとに固有のアプリケーション ID を指定したり、複数のアプリケーションに対して同じ値を指定できます。

ColdFusion MX 6.1: 動作が変更されました。ColdFusion が NTLM 認証情報またはダイジェスト (HTTP Negotiated ヘッダ) 認証情報を含むリクエストを受け取るときは、cflogin 変数が存在します。

ColdFusion MX: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
applicationtoken	オプション	現在のアプリケーション名	アプリケーションに適用するログインです。ユーザーが 1 つのアプリケーションのみにログインできるようにするには、そのアプリケーションの固有の値を指定します。ユーザーが複数のアプリケーションにログインできるようにするには、それらのアプリケーションに同じ値を指定します。applicationtoken 属性の値を設定しない場合、デフォルト値は CFAUTHORIZATION_< アプリケーション名 > です。
cookiedomain	オプション		ユーザーがログインしていることを示すために使われる Cookie のドメインです。この属性を使うことで、ユーザーのログイン Cookie を同じドメイン内の複数のクラスタサーバーで使用できるようにします。
idletimeout	オプション	1800	ColdFusion がユーザーをログオフさせることになるまでの待機時間 (秒単位) です。

使用方法

このタグの本文は、ログインしているユーザーがいない場合にのみ実行されます。アプリケーションベースのセキュリティを使用する場合は、cflogin タグの本文に、ユーザーが指定した ID とパスワードをログイン ID のデータソース、LDAP ディレクトリ、またはその他のレポジトリと対照して確認するためのコードを挿入します。タグ本文には cfloginuser タグを指定して、認証済みユーザーを ColdFusion に識別させる必要があります。

ユーザーはデータソースを制御し、cflogin タグ内の SQL のコーディングを担当します。また、関連付けられたデータベースにユーザー、パスワード、およびロールに関する情報があるようにしなければなりません。

cflogin タグは、そのページが次のいずれかに対するレスポンスとして実行されている場合には、cflogin.name と cflogin.password という 2 つの変数から成るビルトインの cflogin 構造体を含んでいます。

- j_username および j_password という名前の入力フィールドを含むフォームの送信。
- HTTP 基本認証を使用し、ユーザー名とパスワードが入っている認証ヘッダを含むリクエスト。
- NTLM または Digest 認証を使用するリクエスト。この場合、ユーザー名とパスワードは Authorization ヘッダの一方向アルゴリズムでハッシュされます。ColdFusion は Web サーバーからユーザー名を取得し、cflogin.password 値に空の文字列を設定します。

  これらの値は、cflogin タグの本文内でユーザーを認証するために使用され、さらに cfloginuser タグ内でユーザーをログインさせるために使用されます。この構造体は cflogin タグ本文内でのみ使用できます。

例

次の例は、簡単な認証を示しています。このコードは、通常は Application.cfc の onRequestStart メソッドまたは application.cfm ページに置かれます。

```
<cflogin>
    <cfif NOT IsDefined("cflogin")>
        <cfinclude template="loginform.cfm">
        <cfabort>
    <cfelse>
        <cfif cflogin.name eq "admin">
            <cfset roles = "user,admin">
        <cfelse>
            <cfset roles = "user">
        </cfif>
        <cfloginuser name = "#cflogin.name#" password = "#cflogin.password#"
            roles = "#roles#"/>
    </cfif>
</cflogin>
```

次の参照専用の例では、ユーザー ID とパスワードをデータソースと対照して確認します。

```
<cfquery name="qSecurity"
    datasource="UserRolesDb">
    SELECT Roles FROM SecurityRoles
    WHERE username=<cfqueryparam value='#cflogin.name#' CFSQLTYPE="CF_SQL_VARCHAR"
    AND password=<cfqueryparam value='#cflogin.password#' CFSQLTYPE='CF_SQL_VARCHAR'
</cfquery>

<cfif qSecurity.recordcount gt 0>
<cfloginuser name = "#cflogin.name#"
    password = "#cflogin.password#"
    roles = "#trim(qSecurity.Roles)#" >
</cfif>
```

cfloginuser

説明
ColdFusion に対して認証済みのユーザーを識別します。ユーザー ID とロールを指定します。cflogin タグ内で使用します。

カテゴリ
セキュリティタグ

シンタックス
```
<cfloginuser
    name = "name"
    password = "password"
    roles = "roles">
```

関連項目
cflogin、cflogout、GetAuthUser、GetUserRoles、IsUserInAnyRole、IsUserInRole、IsUserLoggedIn、『ColdFusion アプリケーションの開発』の Securing Applications

履歴
ColdFusion MX 6.1: 動作が変更されました。Session スコープが有効で、cfapplication タグの loginStorage 属性が Session に設定されている場合、ユーザーのログインは、そのセッションが期限切れになるか、ユーザーが cflogout タグによってログアウトされるまでは有効なままになります。

ColdFusion MX: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
name	必須		ユーザー名です。
password	必須		ユーザーのパスワードです。
roles	必須		ロール識別子のカンマ区切りリストです。 ColdFusion は、リスト要素内のスペースを要素の一部として処理します。

使用方法
cflogin タグ内で、認証済みユーザーを ColdFusion に識別させるために使われます。このタグを呼び出した後は、GetAuthUser と IsUserInRole はこのユーザー名とロールの情報を返します。

注意：デフォルトでは、このユーザー情報はメモリ上の Cookie に格納されます。cfapplication タグまたは Application.cfc This.loginStorage 変数を使用すると、このログイン情報を Session スコープ内に格納するように指定できます。

例
cflogin を参照してください。

cflogout

説明
現在のユーザーをログアウトさせます。ユーザー ID、パスワード、ロールの情報をサーバーから削除します。このタグを使用しない場合、ユーザーはセッションの終了時に自動的にログアウトします。

カテゴリ
セキュリティタグ

シンタックス
```
<cflogout>
```

関連項目
cflogin、cfloginuser、GetAuthUser、GetUserRoles、IsUserInAnyRole、IsUserInRole、IsUserLoggedIn、『ColdFusion アプリケーションの開発』の Securing Applications

履歴
ColdFusion MX 6.1: 動作が変更されました。Session スコープが有効な場合、ユーザーのログインは、セッションが期限切れになるか、ユーザーが cflogout タグによってログアウトされるまでは有効なままになります。

ColdFusion MX: このタグが追加されました。

例
```
<cflogin>
    <cfloginuser
        name = "foo"
        password ="bar"
        roles = "admin">
</cflogin>
<cfoutput>Authorized user: #getAuthUser()#</cfoutput>
<cflogout>
<cfoutput>Authorized user: #getAuthUser()#</cfoutput>
```

cfloop

説明
ループは、条件 (複数指定可) が満たされるまで、一連の命令や出力表示を繰り返すプログラミング手法です。このタグでは、次の種類のループがサポートされます。

- cfloop: インデックスループ
- cfloop: 条件付きループ
- cfloop: 日付または時刻の範囲に対するループ
- cfloop: クエリーに対するループ
- cfloop: リスト、ファイル、または配列に対するループ
- cfloop: COM コレクションまたは構造体に対するループ

詳細については、cfloop and cfbreak、および『ColdFusion アプリケーションの開発』の Populating arrays with data を参照してください。

カテゴリ
フロー制御タグ

cfloop: インデックスループ

説明

インデックスループは、数値で指定された回数だけ繰り返されます。インデックスループは FOR ループとも呼ばれます。

シンタックス

```
<cfloop
    index = "parameter name"
    from = "beginning value"
    to = "ending value"
    step = "increment"
    charset "charset to read in a file">
        HTML or CFML code ...
</cfloop>
```

関連項目

cfabort、cfbreak、cfcontinue、cfdirectory、cfexecute、cfexit、cfif、cflocation、cfrethrow、cfswitch、cfthrow、cftry、『ColdFusion アプリケーションの開発』の cfloop and cfbreak

属性

属性	必須 / オプション	デフォルト	説明
index	必須		インデックスの値です。この値は最初 from 属性の値に設定され、step 属性の値ずつ、to 属性の値までインクリメントまたはデクリメントされます。
from	必須		インデックスの開始値です。
to	必須		インデックスの終了値です。
step	オプション	1	インデックス値をインクリメントまたはデクリメントする数値です。
charset	オプション		ファイルを行ごとに読み込むときに使用する文字セットです。

使用方法

インデックスループの from 属性と to 属性に整数値以外の値を指定すると、予期せぬ結果になる可能性があります。たとえば、インデックスループを通じて 1 ～ 2 の間で 0.1 ずつインクリメントしていく場合、ColdFusion は "1、1.1、1.2、...、1.9" まで出力しますが、"2" は出力しません。これは、浮動小数点数の内部表現に関するプログラミング言語の問題です。

注意：to 値は、cfloop タグが検出されたときに一度だけ評価されます。loop ブロック内、またはこの値を評価する式内でこの値を変更しても、ループが実行される回数に影響はありません。

例

次の例では、ループが 5 回行われ、そのたびに index 値が表示されます。

```
<cfloop index = "LoopCount" from = "1" to = "5">
    The loop index is <cfoutput>#LoopCount#</cfoutput>.<br>
</cfloop>
```

このループの出力は次のとおりです。

```
The loop index is 1.
The loop index is 2.
The loop index is 3.
The loop index is 4.
The loop index is 5.
```

次の例では、ループが 4 回行われ、そのたびに index 値が表示されます。j の値はループが繰り返されるたび 1 ずつデクリメントされます。to の値はループに入る前に j からコピーされたものなので、このデクリメントの影響を受けません。

```
<cfset j = 4>
<cfloop index = "LoopCount" from = "1" to = #j#>
    <cfoutput>The loop index is #LoopCount#</cfoutput>.<br>
    <cfset j = j - 1>
</cfloop>
```

このループの出力は次のとおりです。

```
The loop index is 1.
The loop index is 2.
The loop index is 3.
The loop index is 4.
```

前述のように、j の値はループが繰り返されるたびにデクリメントされますが、to 値は j がループに入る前に作成されたコピーであるため影響はありません。

この例では、step にはデフォルト値 1 が使用されています。次のコードでは、インデックス値がデクリメントされます。

```
<cfloop index = "LoopCount"
    from = "5"
    to = "1"
    step = "-1">
The loop index is <cfoutput>#LoopCount#</cfoutput>.<br>
</cfloop>
```

このループの出力は次のとおりです。

```
The loop index is 5.
The loop index is 4.
The loop index is 3.
The loop index is 2.
The loop index is 1.
```

cfloop: 条件付きループ

説明

条件付きループは、条件が true であるかぎり、一連の命令を繰り返します。このタイプのループを正しく使用するためには、ループが繰り返されるたびに条件を変更する命令を記述し、条件が false になるまでそれを行う必要があります。条件付きループは、" この条件が TRUE である間はループする " という意味で、WHILE ループとも呼ばれます。

シンタックス

```
<cfloop
    condition = "expression">
    ...
</cfloop>
```

関連項目

cfabort、cfbreak、cfcontinue、cfexecute、cfexit、cfif、cflocation、cfswitch、cfthrow、cftry、『ColdFusion アプリケーションの開発』の cfloop and cfbreak

属性

属性	必須 / オプション	デフォルト	説明
condition	必須		ループを制御するための条件です。

例

次の例では、CountVar が 1 ～ 5 まで 1 ずつインクリメントされています。

```
<!--- Set the variable CountVar to 0. --->
<cfset CountVar = 0>
<!--- Loop until CountVar = 5. --->
<cfloop condition = "CountVar LESS THAN 5">
    <cfset CountVar = CountVar + 1>
    The loop index is <cfoutput>#CountVar#</cfoutput>.<br>
</cfloop>
```

このループの出力は次のとおりです。

```
The loop index is 1.
The loop index is 2.
The loop index is 3.
The loop index is 4.
The loop index is 5.
```

cfloop: 日付または時刻の範囲に対するループ

説明

from 属性と to 属性で指定された日付と時刻の範囲をループします。デフォルトの数値単位は 1 日ですが、タイムスパンを作成することによりこの値を変更できます。このタイプの cfloop タグは、cfoutput タグ内で使用できないタグに対してループを行います。

シンタックス

```
<cfloop
    from = "start time"
    to = "end time"
    index = "current value"
    step = "increment">
</cfloop>
```

関連項目

cfabort、cfbreak、cfcontinue、cfdirectory、cfexecute、cfexit、cfif、cflocation、cfrethrow、cfswitch、cfthrow、cftry、『ColdFusion アプリケーションの開発』の cfloop and cfbreak

属性

属性	必須 / オプション	デフォルト	説明
from	必須		日付または時刻の範囲の開始時です。
to	必須		日付または時刻の範囲の終了時です。
index	必須	1 day	インデックスの値です。この値は最初 from 属性の値に設定され、step 属性の値ずつ、to 属性の値までインクリメントされます。
step	オプション		インデックスをインクリメントする数値単位です。タイムスパンとして表されます。

例

次の例では、今日の日付から今日の日付に 30 日を加えた日付まで、一度に 7 日ずつループして、日付を表示します。

```
<cfset startDate = Now()>
<cfset endDate = Now() + 30>
<cfloop from="#startDate#" to="#endDate#" index="i" step="#CreateTimeSpan(7,0,0,0)#">
<cfoutput>#dateformat(i, "mm/dd/yyyy")#<br /></cfoutput>
</cfloop>
```

次の例では、夜中の 0 時から 23 時 59 分 59 秒まで、時間を 30 分ずつインクリメントして表示します。

```
<cfset startTime = CreateTime(0,0,0)>
<cfset endTime = CreateTime(23,59,59)>
<cfloop from="#startTime#" to="#endTime#" index="i" step="#CreateTimeSpan(0,0,30,0)#">
    <cfoutput>#TimeFormat(i, "hh:mm tt")#<br /></cfoutput>
</cfloop>
```

cfloop: クエリーに対するループ

説明

クエリーに対するループは、クエリーレコードセット内のレコードごとに実行されます。結果は cfoutput タグと同様です。ループが繰り返されるごとに、現在行の列が出力されます。このタイプの cfloop タグは、cfoutput タグ内で使用できないタグに対してループを行います。

シンタックス

```
<cfloop
    query = "query name"
    startRow = "row number"
    endRow = "row number"
    group = "Query column">
</cfloop>
```

関連項目

cfabort、cfbreak、cfcontinue、cfexecute、cfexit、cfif、cflocation、cfoutput、cfswitch、cfthrow、cftry。詳細については、『ColdFusion アプリケーションの開発』の cfloop and cfbreak を参照してください。

属性

属性	必須 / オプション	デフォルト	説明
query	必須		ループを制御するためのクエリーです。 query 属性を使用する場合に、文字列の使用に加えて、次のコードに示すような動的な参照を使用できるようになりました。 `<cfloop query = "#getEmployees()#">`
startRow	オプション		ループに含めるクエリーの最初の行です。
endRow	オプション		ループに含めるクエリーの最後の行です。
group	オプション		レコードのセットをグループ化するために使用するクエリー列です。データをソートする場合、隣り合う重複行は削除されます。1 つ以上のクエリ列で順序付けられたレコードセットを取得した場合に使用します。例えば、レコードセットが "Customer_ID" に従って順序付けられている場合は、"Customer_ID" に関する出力をグループ化することができます。

例

```
<cfquery name = "MessageRecords" dataSource = "cfdocexamples">
    SELECT * FROM Messages
</cfquery>
<cfloop query = "MessageRecords">
<cfoutput>#Message_ID#</cfoutput><br>
</cfloop>
```

cfloop タグでは、レコードセットの開始ポイントと終了ポイントをダイナミックに指定してループを繰り返すこともできます。これにより、後続の **n** セットのレコードをクエリーから取得します。この例では、MessageRecords クエリーで返される 5 番目のレコードから 10 番目のレコードまでループします。

```
<cfset Start = 5>
<cfset End = 10>
<cfloop query = "MessageRecords"
    startRow = "#Start#"
    endRow = "#End#">
<cfoutput>#MessageRecords.Message_ID#</cfoutput><br>
</cfloop>
```

レコードがなくなるか、現在のレコードのインデックスが endRow 属性の値を上回ると、ループは停止します。次の例では、cfinclude タグを使用して、ページ名リストのクエリーによって返されたページ群を 1 つのドキュメントに結合します。

```
<cfquery name = "GetTemplate" dataSource = "Library" maxRows = "5">
    SELECT TemplateName
    FROM Templates
</cfquery>
<cfloop query = "GetTemplate">
    <cfinclude template = "#TemplateName#">
</cfloop>
```

cfloop: リスト、ファイル、または配列に対するループ

説明

リストに対してループを行うと、次のエンティティ内に含まれている要素が順次使用されます。

- 変数
- 式から返される値
- 配列
- ファイル

  ファイルに対してループを行うと、メモリ内でファイル全体は開かれません。

シンタックス

```
<cfloop
    index = "index name"
    array = "array"
    characters = "number of characters"
    delimiters = "item delimiter"
    file = "absolute path and filename">
    list = "list items"
    ...
</cfloop>
```

関連項目

cfabort、cfbreak、cfcontinue、cfexecute、cfexit、cfif、cflocation、cfswitch、cfthrow、cftry、『ColdFusion アプリケーションの開発』の cfloop and cfbreak

履歴

ColdFusion 8: characters、file、および array 属性が追加されました。

属性

属性	必須 / オプション	デフォルト	説明
index	必須		リスト、ファイルまたは配列のループ内で、次の要素を受け取る変数です。
list	file 属性でファイル名を指定しない場合は必須		リスト、変数、またはファイル名です。これらにはリストが含まれます。
array	オプション		配列です。
characters	オプション		ループを繰り返すたびに file 属性で指定されたファイルから読み込む文字数です。characters 属性の値がファイル内の文字数よりも大きい場合は、ファイル内の文字数が使用されます。
delimiters	オプション		リスト内の項目を区切る文字です。
file	オプション		読み込むディスク上またはメモリ内のテキストファイルの絶対パス名です。一度に 1 行ずつ読み込まれます。ファイルの現在行を格納するインデックス変数の値を再利用できるので、サイズの大きいテキストファイルを読み込む場合に便利です。ループが完了すると、ファイルは閉じられます。

例

このループでは 4 つの名前を表示します。

```
<cfloop index = "ListElement" list = "John,Paul,George,Ringo">
    <cfoutput>#ListElement#</cfoutput><br>
</cfloop>
```

delimiters 属性に、任意の順序で複数の文字を入れることができます。たとえば、次のループではカンマ、コロン、スラッシュがリストの区切り文字として処理されます。

```
<cfloop index = "ListElement" list = "John/Paul,George::Ringo" delimiters = ",:/">
    <cfoutput>#ListElement#</cfoutput><br>
</cfloop>
```

リスト要素間にある、連続する 2 つめ以降の区切り文字はスキップされます。したがって、前の例では、"George" と "Ringo" の間の 2 つのコロンは、1 つの区切り文字として処理されます。

ファイルの各行をループするには、このタグを次のように使用します。

```
<cfloop file="c:\temp\simplefile.txt" index="line">
    <cfoutput>#line#</cfoutput><br>
</cfloop>
```

ループの繰り返すたびに指定した文字数をテキストファイルから読み込むには、このタグを次のように使用します。

```
<cfloop file="c:\temp\simplefile.txt" index="chars" characters="12">
    <cfoutput>#chars#</cfoutput><br>
</cfloop>
```

次のテキストファイルを読み込むと、ループが繰り返されるたびに 12 文字が読み込まれ、次の結果が表示されます。

テキストファイル	結果
This is line 1. This is line 2. This is line 3. This is line 4. This is line 5. This is line 6. This is line 7. This is line 8. This is line 9. This is line 10. This is line 11.	This is line 1. This is line 2. Th is is line 3 . This is l ine 4. This is line 5. This is lin e 6. This i s line 7. T his is line 8. This is line 9. Thi s is line 10 . This is l ine 11.

配列に対してルールを行うには、次のように設定します。

```
<cfset x = ["mars","earth", "venus", "jupiter"]>
<cfloop array="#x#" index="name">
    <cfoutput>#name#</cfoutput><br>
</cfloop>
```

cfloop: COM コレクションまたは構造体に対するループ

説明

cfloop タグの collection 属性では、COM/DCOM コレクションオブジェクト内のすべてのオブジェクトや、構造体内のすべての要素に対してループを行います。

- COM/DCOM コレクションオブジェクトは、グループとして参照される一連の類似項目です。たとえば、アプリケーション内で開かれているドキュメントのグループはコレクションとなります。
- 構造体は、関連する一連の項目を格納するもので、連想配列として使用できます。ループは、構造体を連想配列として使用する場合に特に有効です。

  ループ内では、各項目が、item 属性に指定した変数名によって参照されます。すべての項目にアクセスが行われるまで、ループが実行されます。

  collection 属性は item 属性とともに使用します。次の例では、item 属性に file2 という変数を指定して、cfloop の各サイクルでコレクション内の各項目を参照できるようにしています。cfoutput セクションでは、file2 項目の name プロパティを参照し、表示しています。

  詳細については、『ColdFusion アプリケーションの開発』の Integrating COM and CORBA Objects in CFML Applications を参照してください。

例

この例では、COM オブジェクトを使用してファイルのリストを出力します。この例では、FFunc は file2 オブジェクトのコレクションです。

```
<cfobject
    class = FileFunctions.files
    name = FFunc
    action = Create>
<cfset FFunc.Path = "c:\">
<cfset FFunc.Mask = "*.*" >
<cfset FFunc.attributes = 16 >
<cfset x = FFunc.GetFileList()>
<cfloop collection = #FFUNC# item = "file2">
    <cfoutput> #file2.name# <br> </cfoutput>
</cfloop>
<!---Loop through a structure that is used as an associative array: --->
...
<!--- Create a structure and loop through its contents. --->
<cfset Departments = StructNew()>
<cfset val = StructInsert(Departments, "John ", "Sales ")>
<cfset val = StructInsert(Departments, "Tom ", "Finance ")>
<cfset val = StructInsert(Departments, "Mike ", "Education ")>
<!--- Build a table to display the contents --->
<cfoutput>
<table cellpadding = "2 " cellspacing = "2 ">
    <tr>
        <td><b>Employee</b></td>
        <td><b>Dept.</b></td>
    </tr>
<!--- Use item to create the variable person to hold value of key as loop runs. --->
<cfloop collection = #Departments# item = "person ">
    <tr>
        <td>#person#</td>
        <td>#StructFind(Departments, person)#</td></tr>
</cfloop>
</table>
</cfoutput>
```

タグ m ～ o

cfmail

説明
SMTP サーバーを使用して、オプションでクエリー出力を含む電子メールメッセージを送信します。

カテゴリ
通信タグ、インターネットプロトコルタグ

シンタックス

```
<cfmail
    from = "e-mail address"
    to = "comma-delimited list"
    bcc = "comma-delimited list"
    cc = "comma-delimited list"
    charset = "character encoding"
    debug = "yes|no"
    failto = "e-mail address"
    group = "query column"
    groupcasesensitive = "yes|no"
    mailerid = "header id"
    maxrows = "integer"
    mimeattach = "path"
    password = "string"
    port = "integer"
    priority = "integer or string priority level"
    query = "query name"
    remove = "yes|no"
    replyto = "e-mail address"
    server = "SMTP server address"
    spoolenable = "yes|no"
    startrow = "query row number"
    subject = "string"
    timeout = "number of seconds"
    type = "mime type"
    username = "SMTP user ID"
    useSSL = "yes|no"
    useTLS = "yes|no"
    wraptext = "column number"
    sign = "true|false"
    keystore = "location of keystore"
    keystorepassword = "password of keystore"
    keyalias = "alias of key"
    keypassword = "password for private key">

(Optional) Mail message body and/or cfmailparam tags

</cfmail>
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfmailparam、cfmailpart、cfpop、cfftp、cfhttp、cfldap、Wrap、『ColdFusion アプリケーションの開発』の Sending and Receiving E-Mail の Using ColdFusion with mail servers

履歴

ColdFusion 8.0.1：Remove 属性が追加されました。

ColdFusion 8: priority、useSSL、および useTLS 属性が追加されました。

ColdFusion MX 7:

- cfmail タグの本文に MIME エンコードされたメッセージ全体を埋め込んでマルチパートメールを送信することができなくなりました。代わりに cfmailpart タグを使用してください。
- cfmail タグでは等幅フォントがプロポーショナルフォントと同様にレンダリングされます。この動作変更は ColdFusion 5 からのものです。ColdFusion MX 7 では UTF-8 が使用され、これがメールヘッダに埋め込まれて送信されます (Content-Type: text/plain; charset=UTF-8)。ColdFusion 5 では ISO-8859-1 (Latin 1) が使用されます。この動作を

抑制するには、charset="ISO-8859-1" 属性を追加して ColdFusion 5 のデフォルトのエンコードに戻します。また、ColdFusion Administrator の [メール] ページでエンコードを変更することもできます。

ColdFusion MX 6.1:

- charset、failto、replyto、username、password、および wraptext の各属性が新たに追加されました。
- server 属性で複数のメールサーバーを指定できるようになりました。
- ColdFusion Administrator の [メール設定] ページにいくつかの設定オプションが追加されました。

ColdFusion MX: SpoolEnable 属性が追加されました。

ColdFusion 9: メールに電子署名を追加できるようになりました。新規に追加された関連属性は、sign、keystore、keystorepassword、keyalias、keypassword、および remove です。

属性

属性	必須 / オプション	デフォルト	説明
bcc	オプション		メッセージのコピー先のアドレスです。メッセージのヘッダにはリストされません。複数のアドレスを指定するには、各アドレスの間をカンマで区切ります。
cc	オプション		メッセージのコピー先のアドレスです。複数のアドレスを指定するには、各アドレスの間をカンマで区切ります。
charset	オプション	ColdFusion Administrator の [メール] ページで選択した文字エンコード。 utf-8	ヘッダを含むメールメッセージの文字エンコードです。一般的に使用される値を次に示します。 • utf-8 • iso-8859-1 • windows-1252 • us-ascii • shift_jis • iso-2022-jp • euc-jp • euc-kr • big5 • hz-gb-2312 • euc-cn • utf-16 文字エンコードの詳細については、www.w3.org/International/O-charset.html を参照してください。
debug	オプション	no	• yes: デバッグ出力を標準出力に送信します。デフォルトでは、コンソールウィンドウを使用できない場合、ColdFusion は出力をサーバー設定内の <ColdFusion のルートディレクトリ >¥runtime¥logs¥coldfusion-out.log に送信します。 • no: デバッグ出力を生成しません。
failto	オプション		メールシステムから配達失敗の通知が送信されるアドレスです。メールエンベロープの逆順パスの値を設定します。

属性	必須 / オプション	デフォルト	説明
from	必須		電子メールメッセージの送信者の名前です。 • スタティックな文字列 (例 : "support@mex.com")。 • 変数 (例 : "#GetUser.EMailAddress#")。 この属性は、有効なインターネットアドレスでなくてもかまいません。空白を含まない任意のテキスト文字列を使用できます。
to	必須		メッセージ受信者の電子メールアドレスです。 • スタティックなアドレス (例 : "support@.com")。 • アドレスが含まれている変数 (例 : "#Form.Email#")。 • アドレスが含まれているクエリー列の名前 (例 : "#EMail#")。返される行ごとに電子メールメッセージが送信されます。 複数のアドレスを指定するには、各アドレスの間をカンマで区切ります。
subject	必須		メッセージの表題です。ダイナミックに作成することができます。たとえば、顧客に最新情報を通知するメッセージを送信するには、" 注文番号 #Order_ID# のステータス " などを指定します。
group	オプション	CurrentRow	レコードのセットをグループ化して、1 つのメッセージとして送信する場合に使用するクエリー列です。たとえば、顧客に 1 セットの請求書を送信する場合、"Customer_ID" でグループ化します。大文字と小文字は区別されます。指定したフィールドでデータをソートする場合、隣り合う重複データは削除されます。
groupcasesensitive	オプション	No	ブール値です。group 属性を使用するときに、大文字と小文字を区別するかどうかを指定します。大文字と小文字の区別があるレコードをグループ化する場合は、この属性を Yes に設定します。
keyalias	オプション		証明書と秘密キーをキーストアに保存するために使用するキーのエイリアスです。指定しない場合は、キーストアの最初のエントリがエイリアスとして使用されます。
keypassword	オプション		秘密キー用のパスワードです。指定しない場合は、keystorepassword が使用されます。
keystore	オプション		キーストアファイルの場所です (C:¥OpenSSL¥bin¥keystore.jks など)。
keystorepassword	オプション		キーストア用のパスワードです。ColdFusion 設定ファイルに保存されます。
mailerid	オプション	ColdFusion アプリケーションサーバー	X-Mailer SMTP ヘッダに渡されるメーラー ID です。この ID によって、メーラーのアプリケーションが識別されます。
maxrows	オプション		クエリーをループするときに送信するメッセージの最大数です。
mimeattach	オプション		メッセージに添付するディスク上またはメモリ内のファイルのパスです。添付ファイルは MIME 形式でエンコードされます。ファイルの MIME タイプは ColdFusion によって判断されます。MIME タイプを指定して添付ファイルを送信するには、cfmailparam タグを使用します。
password	オプション		認証を要求する SMTP サーバーに送信するパスワードです。username 属性が必要です。
port	オプション		SMTP サーバーがリクエストをリスンする TCP/IP ポートです (通常は 25)。ここで設定する値は、Administrator で設定する値よりも優先されます。

属性	必須 / オプション	デフォルト	説明
priority	オプション	3	メッセージの優先度レベルです。次のいずれかの値を指定します。 • 1 ～ 5 の範囲の整数。1 が最も高い優先度を表します。 • 次のいずれかの文字列。これらの文字列は数値に対応します。highest または urgent、high、normal、low、lowest または non-urgent。
query	オプション		メッセージのデータを取り出す cfquery の名前です。複数のメッセージを送信する場合や、メッセージ内でクエリー結果を送信する場合は、この属性を使用します。
remove	オプション	no	yes の場合、メールが正常に送信された後に添付ファイル (ある場合) が削除されます。
replyto	オプション		受信者が返信するときに宛先として使用するアドレスです。
server	オプション		メッセージの送信に使用する SMTP サーバーアドレス、または (エンタープライズ版のみ) サーバーアドレスのカンマ区切りのリストです。ここか ColdFusion Administrator で、少なくとも 1 つのサーバーを指定する必要があります。ここで設定する値は、Administrator で設定する値よりも優先されます。ポートの指定を含む値は、port 属性で設定する値よりも優先されます。詳細については、「使用方法」を参照してください。
sign			メールに電子署名を追加します。true に設定されている場合、送信するすべてのメッセージに電子署名が追加されます。
spoolenable	オプション		メールをスプールするか、常にメールを直ちに送信するかを指定します。ColdFusion Administrator の [配達されるメールメッセージをスプール] で設定する値よりも優先されます。 • yes: 送信オペレーションが終了するまで、メッセージのコピーを保存します。ページでこのオプションを使用すると、No オプションを使用するページよりも処理が遅くなることがあります。 • no: 送信オペレーションが終了するまでコピーを保管せずに、メッセージを送信キューに挿入します。このオプションを No に設定したときに配達エラーが発生すると、アプリケーション例外が発生し、mail.log ファイルにエラーのログが記録されます。
startrow	オプション	1	処理を開始するクエリー内の行です。
timeout	オプション		SMTP サーバーへの接続がタイムアウトになるまで待機する秒数です。ここで設定する値は、Administrator で設定する値よりも優先されます。
type	オプション	text/plain	メッセージの MIME タイプです。有効な MIME メディアタイプまたは次のいずれかの値を指定できます。 • text: text/plain タイプを指定します。 • plain: text/plain タイプを指定します。 • html: text/html タイプを指定します。 登録されているすべての MIME メディアタイプのリストについては、www.iana.org/assignments/media-types/ を参照してください。
username	オプション		認証を要求する SMTP サーバーに送信するユーザー名です。password 属性が必要です。

属性	必須 / オプション	デフォルト	説明
useSSL	オプション		Secure Sockets Layer を使用するかどうかを指定します。
useTLS	オプション		Transport Level Security を使用するかどうかを指定します。
wraptext	オプション	テキストをラップしない	メールテキストの最大行を文字数で指定します。指定した文字数よりも行が長い場合は、指定した位置の直前の空白文字 (タブやスペースなど) が改行に置き換わります。行に空白文字がない場合は、指定した位置に改行が挿入されます。この属性の一般的な値は 72 です。

使用方法

指定したアドレスにメールメッセージを送信します。メールメッセージには、添付ファイルを含めることができます。タグ本文に CFML コードを入れて、メール出力を生成できます。cfmailparam タグと cfmailpart タグは、cfmail タグ本文内でのみ使用できます。

メールメッセージは、シングルまたはマルチパートにできます。マルチパートのメールメッセージを送信する場合は、メッセージの内容をすべて cfmailpart タグ内に入れる必要があります。cfmailpart タグ内にないマルチパートのメッセージテキストは無視されます。

注意：cfmail タグは、メールをディスクにスプールするときに添付ファイルのコピーを作成しません。スプール機能を有効にした添付ファイルのメッセージを送信するために cfmail タグを使用し、その添付ファイルを削除するために cffile タグを使用する場合、ファイルが削除された後でメール処理が実行される可能性があるため、メールが送信されないことがあります。この場合は、メールログに FileNotFound 例外が記録され、電子メールは送信されません。属性で SpoolEnable="No" を設定するか、ColdFusion Administrator でスプール機能を無効にすると、この問題を防ぐことができます。スプール機能を無効にすると、電子メールは直ちに配達されるようになります。

type="text" を設定すると、送信するメッセージ内の空白文字が圧縮される場合があります。この問題を解決するには、ColdFusion Administrator で [サーバーの設定]-[設定] に移動し、[空白抑制の有効化] オプションを選択解除します。

メールアドレスの指定

メールアドレスは、次のどの形式でも指定できます。

形式	例
user@server	rsmith@company.com
<user@server>	<rsmith@company.com>
表示名 <user@server>	Rob Smith <rsmith@company.com>
" 表示名 " <user@server>	"Rob Smith" <rsmith@company.com>
user@server (表示名)	rsmith@company.com (Rob Smith)

メールサーバーの指定

server 属性では、複数のメールサーバーを指定できます。

注意：ColdFusion スタンダード版で複数のメールサーバーを指定した場合、cfmail タグではその中の最初のサーバーのみが使用されます。メールログファイルに警告メッセージのログが記録され、残りのサーバーは無視されます。

サーバーごとに、オプションでユーザー名、パスワード、およびポートを指定できます。ここで設定する値は、対応する属性で設定した値よりも優先されます。server 属性は、次の形式で指定します。

```
[user:password@]server[:port],[user:password@]server[:port],....
```

たとえば、次の行では、デフォルトのポートを使用し、ユーザーとパスワードを使用しない mail.myco.com というサーバーを指定しています。次に、ユーザー、パスワード、および特定のポートを使用する 2 番目のサーバーを指定しています。

```
server=mail.myco.com,mail_admin:adm2qzf@mail2.myco.com:24
```

ColdFusion エンタープライズ版で複数のメールサーバーを指定すると、いずれかのサーバーに接続されるまで、指定した順序で利用可能なサーバーへの接続が試行されます。直前の 60 秒間利用できなかったサーバーには接続は試行されません。

電子署名

メールに電子署名を追加するには、sign、keystore、keystorepassword、keyalias、および keypassword の各属性を次の例に示すように指定します。

```
<cfmail from="Sender@Company.com" server="sendmail.myCo.com" sign="true"
keystore="C:\OpenSSL\bin\hello.jks" keystorepassword="digital" to="Recepient@Company.com"
keyalias="crypto" keypassword="signature" subject="Mail with Digital Signature">
```

送信するすべてのメッセージに電子署名を追加するには、タグに属性を追加するのではなく、ColdFusion Administrator の [サーバーの設定]-[設定] ページで設定を指定します。

タグで属性を指定しない場合は、Administrator 設定が適用されます。また、タグで sign = "true" を設定し、keystore、keystorepassword、keyalias、および keypassword の各属性を指定しない場合は、ColdFusion Administrator を使用して指定したこれらの属性の値が適用されます。

例

```
<h3>cfmail Example</h3>

<!--- Delete the surrounding comments to use this example.

<cfif IsDefined("form.mailto")>
    <cfif form.mailto is not "" AND form.mailfrom is not "" AND form.Subject is not "">
        <cfmail to = "#form.mailto#" from = "#form.mailFrom#" subject = "#form.subject#">
                This message was sent by an automatic mailer built with cfmail:
                = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
                #form.body#
        </cfmail>
        <h3>Thank you</h3>
        <p>Thank you, <cfoutput>#mailfrom#: your message, #subject#, has been sent to
            #mailto#</cfoutput>.</p>
    </cfif>
</cfif>
<p>
<form action = "cfmail.cfm" method="POST">
    <pre>
    TO: <input type = "Text" name = "MailTo">
    FROM: <input type = "Text" name = "MailFrom">
    SUBJECT: <input type = "Text" name = "Subject">
    <hr>
    MESSAGE BODY:
    <textarea name ="body" cols="40" rows="5" wrap="virtual"></textarea>
    </pre>
    <!--- Establish required fields. --->
    <input type = "hidden" name = "MailTo_required" value = "You must enter a recipient">
    <input type = "hidden" name = "MailFrom_required" value = "You must enter a sender">
    <input type = "hidden" name = "Subject_required" value = "You must enter a subject">
    <input type = "hidden" name = "Body_required" value = "You must enter some text">
    <p><input type = "Submit" name = ""></p>
</p>
</form>
```

cfmailparam

説明
電子メールメッセージへのファイルの添付やヘッダの追加を行います。

カテゴリ
通信タグ、インターネットプロトコルタグ

シンタックス
```
<cfmail
    to = "recipient"
    subject = "message subject"
    from = "sender"
        more attributes... >
    <cfmailparam
        contentID = "content ID"
        disposition = "disposition type">
        file = "filename"
        type ="media type"

OR

    <cfmailparam
        name = "header name"
        value = "header value">
    ...
</cfmail>
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目
cfmail、cfmailpart、cfftp、cfhttp、cfldap、cfpop、『ColdFusion アプリケーションの開発』の Sending and Receiving E-Mail の Using the cfmailparam tag

履歴
ColdFusion 8.0.1：Content 属性と Remove 属性が追加されました。

ColdFusion MX 6.x: Disposition 属性および ContentID 属性が追加されました。

ColdFusion MX 6.1: type 属性が追加されました。

属性

属性	必須 / オプション	デフォルト	説明
content			ColdFusion 変数の内容を添付ファイルとして送信することができます。これを行うには、次の例のように、# 記号で囲んだ変数を content 属性の値として指定します。 `<cfmailparam file="anyname" content="#variablename#">`
contentID	オプション		添付ファイルの識別子です。この ID はグローバルに固有な値である必要があり、添付ファイルのコンテンツを参照するメール本文中の IMG やその他のタグ内でファイルを識別するために使用されます。
disposition	オプション	attachment	添付するファイルの処理方法です。次のいずれかを指定します。 • attachment: ファイルを添付ファイルとして添付します。 • inline: ファイルのコンテンツをメッセージ内に表示します。
file	name 属性を指定しない場合は必須		メッセージにファイルを添付します。name 属性の値とは排他の関係です。ファイルは送信前に MIME でエンコードされます。
name	file 属性を指定しない場合は必須		ヘッダの名前です。大文字と小文字は区別されません。file 属性の値とは排他の関係です。
remove	オプション	no	yes の場合、メールが正常に送信された後に添付ファイル (ある場合) が削除されます。
type	オプション		ファイルの MIME メディアタイプです。name 属性と一緒には使用しません。有効な MIME メディアタイプまたは次のいずれかの値を指定できます。 • text: text/plain タイプを指定します。 • plain: text/plain タイプを指定します。 • html: text/html タイプを指定します。 タイプを指定すると、指定した値が Content-Type ヘッダになります。タイプを指定しない場合は、ColdFusion によって Content-Type ヘッダが生成されます。 **メモ** : 登録されているすべての MIME メディアタイプのリストについては、www.iana.org/assignments/media-types/ を参照してください。
value	オプション		ヘッダの値です。file 属性と一緒には使用しません。

使用方法

このタグは、電子メールメッセージへのファイルの添付やヘッダの追加を行います。cfmail タグ内でのみ使用できます。cfmail タグ内で複数の cfmailparam タグを使用できます。

このタグを使用して、画像などのファイルを HTML メールメッセージに含めることができます。例 2 に示すように、ファイルは、HTML メッセージ内にインラインで表示することも、添付ファイルとすることもできます。複数のファイルを含めるには、複数の cfmailparam タグを使用します。

メールメッセージ内でのファイルのインラインでの表示

1. cfmail タグで type="html" を指定します。
2. cfmailparam タグで disposition="inline" および ContentID 属性を指定します。
3. src="cid:<ContentID の値 >" 属性を使用して、img タグなどの HTML タグに含めるコンテンツを特定します。

例

例 1: この参照専用の例では、cfmailparam タグを使用して、メッセージへのヘッダの追加、ファイルの添付、送信者への受信通知の返信を行います。

```
<cfmail from = "peter@domain.com" To = "paul@domain.com"
    Subject = "See Important Attachments and Reply">
    <cfmailparam name = "Importance" value = "High">
    Please review the new logo. Tell us what you think.
    <cfmailparam file = "c:\work\readme.txt" type="text/plain">
    <cfmailparam file = "c:\work\logo.gif" type="image/gif">
    <cfmailparam name="Disposition-Notification-To" value="peter@domain.com">
</cfmail>
```

例 2: この参照専用の例では、HTML メッセージの本文に画像を表示します。

```
<cfmail type="HTML"
        to = "#form.mailto#"
        from = "#form.mailFrom#"
        subject = "Sample inline image">
    <cfmailparam file="C:\Inetpub\wwwroot\web.gif"
        disposition="inline"
        contentID="image1">
    <p>There should be an image here</p>
    <img src="cid:image1">
    <p>After the picture</p>
</cfmail>
```

cfmailpart

説明

マルチパート電子メールメッセージの 1 つのパートを指定します。cfmail タグ内でのみ使用できます。cfmail タグ内では、複数の cfmailpart タグを使用できます。

カテゴリ

通信タグ、インターネットプロトコルタグ

シンタックス

```
<cfmail
    ... >
    (Optional cfmailparam entries)
    <cfmailpart
        charset="character encoding"
        type="mime type"
        wraptext="number"
    >
    Mail part contents
    </cfmailpart>
    ...
</cfmail>
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfmail、cfmailparam、cfpop、cfftp、cfhttp、cfldap、cfcontent、Wrap、『ColdFusion アプリケーションの開発』の E-mail

履歴
ColdFusion MX 6.1: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
charset	オプション	cfmail タグの charset 属性で指定した文字エンコード	パートテキストをエンコードする文字エンコードを指定します。一般的に使用される値を次に示します。 • utf-8 • iso-8859-1 • windows-1252 • us-ascii • shift_jis • iso-2022-jp • euc-jp • euc-kr • big5 • hz-gb-2312 • euc-cn • utf-16 文字エンコードの詳細については、www.w3.org/International/O-charset.html を参照してください。
type	必須		パートの MIME メディアタイプです。有効な MIME メディアタイプまたは次のいずれかの値を指定できます。 • text: text/plain タイプを指定します。 • plain: text/plain タイプを指定します。 • html: text/html タイプを指定します。 **メモ :** 登録されているすべての MIME メディアタイプのリストについては、www.iana.org/assignments/media-types/ を参照してください。
wraptext	オプション	テキストをラップしない	メールテキストの最大行を文字数で指定します。指定した文字数よりも行が長い場合は、指定した位置の直前の空白文字 (タブやスペースなど) が改行に置き換わります。行に空白文字がない場合は、指定した位置に改行が挿入されます。この属性の一般的な値は 72 です。

使用方法
このタグは、メールメッセージと、その内容を複数の形式で複製した代替版のメッセージを作成するために使用します。最も一般的な使い方は、どのメールリーダーでも読み取れるプレーンテキストのメッセージの後ろに、HTML 互換のメールリーダーで表示できる HTML 形式のメッセージを付けて送信することです。最も単純な形式のメッセージを最初に指定し、より複雑な形式のメッセージを後から指定します。詳細については、www.ietf.org/rfc/rfc2046.txt を参照してください。

例

```
<h3>cfmailpart Example</h3>
<cfmail from = "peter@domain.com" To = "paul@domain.com"
    Subject = "Which version do you see?">
    <cfmailpart type="text" wraptext="74">
        You are reading this message as plain text, because your mail reader does not handle
            HTML text.
    </cfmailpart>
    <cfmailpart type="html">
        <h3>HTML Mail Message</h3>
        <p>You are reading this message as <strong>HTML</strong>.</p>
        <p>Your mail reader handles HTML text.</p>
    </cfmailpart>
</cfmail>
```

cfmap

説明

ColdFusion Web ページ内に地図を埋め込みます。

現在 ColdFusion でサポートされているのは、Google マップの埋め込みのみです。マップを生成するには、有効な Google MAP API キーを入力し、場所の緯度および経度か、またはその場所の住所を指定します。

Google MAP API キーは、次の方法で指定できます。

1. cfajaximport タグを使用します。MAP API キーを params 属性に次のように指定します。

   ```
   <cfajaximport params="#{googlemapkey='Map API Key'}#">
   ```

2. 次のように Application.cfc を使用します。

   ```
   <cfset this.googlemapkey="Map API Key">
   ```

3. ColdFusion Administrator の [設定] ページを使用します。[Google Map API キー] フィールドに MAP API キーを指定します。また、runtime.cfc 内に MAP API キーを指定することもできます。

カテゴリ

表示管理タグ

シンタックス

```
<cfmap
        centeraddress="address"
        centerlatitude="latitude in degrees"
        centerlongitude="longitude in degrees"
        collapsible="true|false"
        continuouszoom="true|false"
        doubleclickzoom="true|false"
        height="integer"
        hideborder="true|false"
        initshow="true|false"
        markerbind="bind expression"
        markercolor="marker color"
        markericon="icon path"
        markerwindowcontent="content"
        name="name"
        onerror="JavaScript function name"
        onload="JavaScript function name"
        overview="true|false"
        scrollwheelzoom="true|false"
        showallmarkers="true|false"
        showcentermarker="true|false"
        showmarkerwinodw="true|false"
        showscale="true|false"
        showUser="true|false"
        tip="center property marker tips"
        title="string"
        type="map|satellite|hybrid|earth|terrain"
        typecontrol="none|basic|advanced"
        width="integer"
        zoomcontrol="none|small|large|small3d|large3d"
        zoomlevel="integer">
</cfmap>
```

関連項目

cfdiv、cfwindow、cfmapitem

履歴

ColdFusion 10：showUser 属性が追加されました。

ColdFusion 9.0.1：initShow 属性が追加されました。

ColdFusion 9: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
centeraddress	centerlatitude および centerlongitude が指定されていない場合は必須		マップの中央として設定される、場所の住所です。
centerlatitude	centeraddress が指定されていない場合は必須		場所の緯度の値です (単位 : 度数)。この値がマップの中央として設定されます。 この属性は、centerlatitude 属性とともに使用する必要があります。 centerlatitude の有効な値は -90 ～ + 90 です。
centerlongitude	centeraddress が指定されていない場合は必須		場所の経度の値です (単位 : 度数)。この値がマップの中央として設定されます。 この属性は、centerlongitude 属性とともに使用する必要があります。 centerlongitude の有効な値は -180 ～ +180 です。
collapsible	オプション	false	周囲のパネルに対して collapsible プロパティを指定するかどうかを示します。 • true • false collapsible を true に設定した場合は、hideborders を true に設定することができません。
continuouszoom	オプション	true	マップのスムーズなズーミングを可能にするズームコントロールを指定するかどうかを示します。 • true • false
doubleclickzoom	オプション	true	ダブルクリックによるズームを有効にするかどうかを指定します。 • true • false
height	オプション	400 ピクセル	マップの高さです (単位 : ピクセル)。
hideborder	オプション	true	周囲のパネルのボーダーを非表示するかどうかを指定します。 • true • false hideborder を true に設定した場合は、collapsible を true に設定することができません。
initshow	オプション	true	ページロード時のマップの表示／非表示を設定するために使用します。 これは、折り畳み可能な div や折り畳み可能な spry 領域で、リンクやボタンのクリック時にマップを表示する必要がある場合に便利です。
markerbind	オプション		マーカーアイコンをクリックしたときに開くウィンドウにデータをダイナミックに挿入するためのバインド式です。バインド式は、CFC 関数、JavaScript 関数、または URL を指定できます。
markercolor	オプション		マーカーの色を 16 進数値で指定します。 デフォルトでは、centermarker は緑色です。 markericon 属性と markercolor 属性は排他の関係です。
markericon	オプション		マーカーアイコンとして使用するイメージファイルの場所です。markericon 属性と markercolor 属性は排他の関係です。

属性	必須 / オプション	デフォルト	説明
markerwindowcontent	オプション		マーカーウィンドウに表示されるスタティックコンテンツです。この属性は、markerbind 属性と排他の関係にあります。
name	必須		マップの名前です。 JavaScript 関数を呼び出す場合、name 属性は必須です。
onerror	オプション		Google Map API エラーが発生したときに実行する JavaScript 関数です。 JavaScript 関数は 2 つのパラメータとともに渡されます。これらのパラメータは、Google マップのステータスコードとエラーメッセージです。
onload	オプション		ベントの登録など、マップをロードした後に実行するカスタム JavaScript 関数です。
overview	オプション	false	マップに概要パネルを追加するかどうかを指定します。 • true • false
showmarkerwindow	オプション	false	true に設定されている場合、マーカーウィンドウが表示されます。markerbind 属性を使用する場合、この属性を true に設定しないと、マーカーウィンドウは表示されません。 markerwindowcontent が true に設定されている場合、この属性は無視されます。
showUser	オプション	false	true を設定すると、HTML 準拠のブラウザーの場合、ユーザーの場所がマップに表示されます。 HTML 5 に準拠しないブラウザーの場合、住所は centerAddress で指定した値に置換されます。値が指定されていない場合、centerLatitude および centerLongitude で指定した値に置換されます。 ユーザーは、ユーザーの場所が追跡されるようにするためにサイトを認証する必要があります。例えば、Google Chrome では、物理的な場所の追跡を許可するように促されます。
scrollwheelzoom	オプション	true	マウスホイールのズーミングコントロールを有効にするかどうかを指定します。 • true • false
showallmarkers	オプション	true	マップに追加されているすべてのマーカーを表示するかどうかを指定します。 • true • false showallmarkers に true を指定した場合、マップ領域内にすべてのマーカーを表示するために、マップに対して指定されているズームレベルをオーバーライドできます。
showcentermarker	オプション	true	マップの中央を識別するマーカーアイコンを表示するかどうかを指定します。 • true • false
showscale	オプション	false	スケールコントロールを表示するかどうかを指定します。 • true • false
tip	オプション		ツールヒントとして表示される中央の場所の短い説明です。
title	オプション		パネルのタイトルです。 hideborder を true に設定した場合は、タイトルを定義できません。

属性	必須 / オプション	デフォルト	説明
type	オプション	map	Google マップのタイプです。 • map • satellite • hybrid • terrain • earth: type="earth" を使用する場合、Google Earth 3D プラグインのダウンロードを求められます。
typecontrol	オプション	basic	マップを切り替えることができるタイプコントロールを指定するかどうかを示します。 • basic：map、satellite および hybrid のオプションを提供するマーカータイプが表示されます。 • none • advanced: type 属性で定義される 5 つのオプションのドロップダウンリストが表示されます。
width	オプション	400 ピクセル	マップの幅です (単位 : ピクセル)。
zoomcontrol	オプション	small	ズームコントロールを有効にするかどうかを指定します。 • none • small • large • large3d • small3d
zoomlevel	オプション	3	開始ズーム値を指定します。

使用方法

このタグは、マップを作成するために、HTML ページ、div タグ、または新規ウィンドウ内で使用できます。このタグを新規ウィンドウ内で使用する場合は、cfwindow タグ内で cfmap タグを使用する必要があります。

zoomcontrol 属性を使用すると、埋め込みマップのサイズを変更できます。ズーム値を大きくすると、マップのクローズアップビューを取得できます。または、ズーム値を小さくすると、マップのより広範囲が小さいサイズで表示されます。ズーム値を変更するたびに、マップ全体が更新されるわけではありません。マップの変更される部分のみが更新されるため、データの表示が高速になります。

cfmap タグでは、map、satellite、terrain、earth、hybrid という 5 つの形式のマップ表示がサポートされます。map 形式では、標準的なロードマップイメージが表示されます。satellite 形式では、マップの衛星イメージが表示されます。hybrid 形式では、マップのロードマップと衛星イメージの組み合わせが表示されます。この場合、衛星イメージ上で、主要な通りの名前と場所がマークされます。

type="earth" の場合、Zoomlevel、showScale、overview、tip、zoomControl、showCenterMarker、および showAllMarkers 属性は機能しません。

Safari 3.x および Google Chrome で cfmap タグを機能させるには、HTML の head タグ (<head></head>) を指定する必要があります。

例

```
<h3>cfmap Example using latitude and longitude attributes</h3>
<cfmap name="gmap01"
    centerlatitude="71.094224"
    centerlongitude="42.339641"
    doubleclickzoom="true"
    overview="true"
    scrollwheelzoom="true"
    showscale="true"
    tip="My Map"
    zoomlevel="4"/>

<h3>cfmap Example using center address</h3>
<cfmap name="gmap02"
    centeraddress="345 Park Avenue, san jose, CA 95110-2704, USA"
    doubleclickzoom="true"
    scrollwheelzoom="true"
    showscale="false"
    tip="My Map"/>
```

cfmapitem

説明

cfmapitem タグは cfmap タグの子タグです。このタグによってマップ上のマーカーが作成されます。マップ内にマーカーを指定するには、cfmapitem タグを使用するか、または ColdFusion.Map.AddMapMarker JavaScript API を使用します。詳細については、1415 ページの「ColdFusion.Map.addMarker」を参照してください。

カテゴリ

表示管理タグ

シンタックス

```
<cfmapitem
        address="address"
        latitude="latitude in degrees"
        longitude="longitude in degrees"
        markercolor="marker color"
        markericon="icon path"
        markerwindowcontent="content"
        name="name of the map"
        showmarkerwinodw="true|false"
        showUser="true|false"
        tip="marker tip" />
```

関連項目

cfdiv、cfwindow、cfmap

履歴

ColdFusion 10：showUser 属性が追加されました。

ColdFusion 9: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
address	latitude および longitude が指定されていない場合は必須		マップマーカーを設定する場所の住所です。
latitude	address が指定されていない場合は必須		マーカーの緯度の値です (単位 : 度数)。 latitude の有効な値は -90 ～ + 90 です。
longitude	address が指定されていない場合は必須		マーカーの経度の値です (単位 : 度数)。 longitude の有効な値は -180 ～ + 180 です。
markercolor	オプション	green	マーカーの色を 16 進数値で指定します。 デフォルトでは、centermarker は緑色です。 markericon 属性と markercolor 属性は排他の関係です。
markericon	オプション		マーカーアイコンとして使用するイメージファイルの場所です。markericon 属性と markercolor 属性は排他の関係です。
markerwindowcontent	オプション		マーカーウィンドウに表示されるスタティックコンテンツです。この属性は、cfmap タグで定義されている markerbind 属性を無視します。
name	オプション		マップの名前です。
showmarkerwindow	オプション	親の cfmap 設定を継承します。	true に設定されている場合、マーカーウィンドウが表示されます。markerbind 属性を使用する場合、この属性を true に設定しないと、マーカーウィンドウは表示されません。
showUser	オプション	false	true を設定すると、HTML 準拠のブラウザーの場合、ユーザーの場所がマップに表示されます。
tip	オプション		ツールヒントとして表示されるマーカーの場所の短い説明です。

使用方法

このタグは、cfmap タグ内で使用する必要があります。

次の継承ルールが適用されます。

- cfmap タグの showmarkerwindow 属性に指定されている値は、すべての cfmapitem タグに継承されます。
- 子の cfmapitem タグは、値を変更することで cfmap タグの showmarkerwindow 属性よりも優先することができます。
- markerbind を使用して定義されたバインド式は、cfmapitem タグで markerwindowcontent 属性が定義されている場合は無視されます。

例

```
<h3>cfmapitem example using latitude and longitude attributes</h3>
<cfmap name="gmap01"
    centerlatitude="71.094224"
    centerlongitude="42.339641"
    doubleclickzoom="true"
    overview="true"
    scrollwheelzoom="true"
    showscale="true"
    tip="My Map"
    zoomlevel="4">
<cfmapitem name="marker01"
    latitude="70.50"
    longitude="42.50"
    tip="New marker"/>

<h3>cfmap Example using address address</h3>
<cfmap name="gmap02"
    centerlatitude="71.094224"
    centerlongitude="42.339641"
    doubleclickzoom="true"
    overview="true"
    scrollwheelzoom="true"
    showscale="true"
    tip="My Map"
    zoomlevel="4">
<cfmapitem name="marker02"
    address="345 Park Avenue, san jose, CA 95110-2704, USA"
    tip="New marker"/>
```

cfmediaplayer

説明

FLV、MPEG-3 および MPEG-4 ファイルに加えて、HTML 5 準拠のブラウザーでサポートされているすべての形式のビデオを再生可能な組み込みメディアプレーヤーを作成します。

FLV ファイルは任意の Web サーバーから再生できます。MPEG-3 および MPEG-4 は、RTMP を使用する Flash Media Server からのみ再生できます。また、MP3 形式のオーディオファイルも再生できます。

カテゴリ

表示管理タグ

シンタックス

```
<cfmediaplayer
    align="alignment option"
    autoplay="true|false"
    bgcolor="hexadecimal value"
    hideborder="true|false"
    hidetitle="true|false"
    controlbar="true|false"
    fullScreenControl="yes|no"
    name="name"
    onComplete="JavaScript function name"
    onError="JavaScript function name"
    onPause="JavaScript function name"
    onLoad="JavaScript function name"
    onStart="JavaScript function name"
    posterImage="URL"
    quality="low|high|medium"
    repeat="true|false"
    skin="XML_Path"
    source="source name"
    style="style specification"
    title="title"
    type="html|flash">
    height="integer"
    width="integer"
    wmode="window|opaque|transparent">
</cfmediaplayer>
```

履歴

ColdFusion 10：type、repeat、posterImage、title、skin、onPause、onError

ColdFusion 9: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
align	オプション	left	メディアプレーヤーの水平方向の整列を指定します。left、right、および center から選択できます。
autoplay	オプション	false	メディアプレーヤーで、CFM ページのロード時に FLV ファイルを自動的に再生するかどうかを指定するブール値です。 • true • false
bgcolor	オプション	6b7c85	16 進数値または認識可能なカラー名前 (red など) で指定した、メディアプレーヤーの背景色です。 HTML プレーヤーの場合、この属性は、wmode 属性を transparent に設定した場合にのみ適用されます。 Flash Player を使用している場合、この依存関係はありません。
controlbar	オプション	true	メディアプレーヤーのコントロールパネルを表示するかどうかを指定するブール値です。 • true • false

属性	必須 / オプション	デフォルト	説明
hideborder	オプション	true	メディアプレーヤーパネルのボーダーを表示するかどうかを指定するブール値です。 • true • false
hidetitle	オプション	true	true の場合、ビデオファイル名は表示されなくなります。
fullScreenControl	オプション	yes	全画面表示を有効にするかどうかを指定します。 • yes • no 次の特性が適用されます。 • 1 回クリックすると、メディアプレーヤーが再生または一時停止されます。 • ボーダー (定義されている場合) は全画面モードでは表示されません。 • 全画面モードを使用する場合にダブルクリックします。 • (全画面モードの場合) Esc キーを押すかダブルクリックすると、プレーヤーが標準モードに戻ります。 この属性は、HTML の再生ではサポートされません。また、Flash Player の全画面表示は無効にできません。
height	オプション	275 ピクセル	メディアプレーヤーの高さです (単位 : ピクセル)。
name	source が定義されていない場合は必須		メディアプレーヤーの名前です。 JavaScript 関数を呼び出す場合、name 属性は必須です。
onComplete	オプション		FLV ファイルの再生が終了したときに実行するカスタム JavaScript 関数です。
onError			再生でエラーが発生したときに実行するカスタム JavaScript 関数です。
onLoad	オプション		プレーヤーコンポーネントのロード時に実行するカスタム JavaScript 関数です。
onPause			ビデオが一時停止されたときに実行するカスタム JavaScript 関数です。
onStart	オプション		FLV ファイルの再生が開始されたときに実行するカスタム JavaScript 関数です。
posterImage			ビデオの再生用のポスターイメージを設定します。 URL または相対アドレスを値として使用します。
quality	オプション	high	メディア再生の品質です。 • low • medium • high
repeat		false	true の場合、メディアプレーヤーがビデオの最後に達すると、引き続き、最初から最後までフレームが再生されます。
skin			スキン設定オプションが指定されている XML へのパスです。Flash にのみ適用されます。 次に例を示します。 <cfmediaplayer source="myvideo.mp4" skin="./skin/myskin.xml">

属性	必須 / オプション	デフォルト	説明
source	name が定義されていない場合は必須		FLV ファイルの URL です。このパラメータでは、現在のページからの相対 URL を指定できます。 FLV ファイルは、ColdFusion サーバーまたはその他のストリーミングサーバー上に格納できます。
style	オプション		サポートされるスタイルは次のとおりです。 • bgcolor: メディアプレーヤーの背景色です。 • borderbottom: 数値。デフォルト値は 10 です。 • bordertop: 数値。デフォルト値は 10 です。 • borderleft: 数値。デフォルト値は 10 です。 • borderright: 数値。デフォルト値は 10 です。 • titletextcolor: RGB カラーの 16 進数値。たとえば、白の場合は ##FFFFFF または FFFFFF と指定します。デフォルトは黒です。 • titlebgcolor: RGB カラーの 16 進数値。デフォルトは黒です。 • progresscolor: プログレスバーの前景色です。RGB カラーの 16 進数値。デフォルトは黒です。 • progressbgcolor: プログレスバーの背景色です。RGB カラーの 16 進数値。デフォルトは黒です。 • controlscolor: コントロールパネル内のコントロールの前景色です。RGB カラーの 16 進数値。デフォルトは黒です。 • controlbarbgcolor: コントロールの背景色です。RGB カラーの 16 進数値。デフォルトは黒です。
title			メディアプレーヤー上のタイトルを設定します。 タイトルは、メディアプレーヤーの左上隅に表示されます。title を指定して hideTitle を指定しない場合、hideTitle は false に設定されます。また、再生が Flash の場合、Flash Player の wmode 属性は opaque に設定され、デフォルト値およびユーザーが指定した値は無視されます。
type		flash	メディアプレーヤーのタイプです（html または flash）。
width	オプション	480 ピクセル	メディアプレーヤーの幅です (単位 : ピクセル)。
wmode	オプション	window	ブラウザでの絶対位置設定およびレイヤー作成の機能を指定します。 • window: Web ページ上の独自の長方形のウィンドウ内でメディアプレーヤーを再生します。 • opaque: Web ページ上でメディアプレーヤーの背後にあるものをすべて非表示にします。 • transparent: メディアプレーヤーの透明の部分に Web ページの背景が透けて見えるようにします。 再生タイプを Flash に設定した場合、この属性の値は opaque に設定され、デフォルト値およびユーザーが指定した値は無視されます。

例

この例では、FLV ファイルは、ColdFusion サーバーが使用する Web ルートに保存されています。FLV ファイル (mediafile.flv) は <Web ルート >**¥xyz¥** の場所に格納する必要があります。これで、次の内容を含むメディアプレーヤーを作成できます。

```
<h3>cfmediaplayer Example</h3>
<cfmediaplayer
    name="Myvideo"
    source="/xyz/mediafile.flv"
    width=500
    height=400
    align="center"
    quality="high"
    fullscreencontrol="true"/>
```

次のコードは、メディアプレーヤーのスタイルを設定する方法を示しています。

```
<cfset bgColorTheme = "EDC393">
<cfset titleColorTheme = "800517">
<cfset controlsColorTheme = titleColorTheme>
<cfset progressColorTheme = "E67451">
<cfset progressbgColorTheme = "FFF8C6">
<cfmediaplayer name="player2" style="bgcolor:#bgColorTheme#;
titletextcolor:#titleColorTheme#;titlebgcolor:#bgColorTheme#;controlbarbgcolor:#bgColorTheme#;controlscol
or:#controlsColorTheme#;progressbgcolor:#progressbgColorTheme#;progresscolor:#progressColorTheme#;borderl
eft:20;borderright:20;bordertop:10;borderbottom:13" hideborder="false" hideTitle=false controlbar="true"
source="#defaultFlvfile#">
```

cfmenu

説明

水平方向または垂直方向に配置されるメニューを作成します。任意のメニューアイテムをサブメニューのトップレベルにすることができます。

カテゴリ

表示管理タグ

シンタックス

```
<cfmenu
    bgcolor="HTML color value"
    childStyle="CSS style specification"
    font="HTML font family"
    fontColor="HTML color value"
    fontSize="Number of pixels"
    menuStyle="CSS style specification"
    name="string"
    selectedFontColor="HTML color value"
    selectedItemColor="HTML color value"
    type="horizontal|vertical
    width="Number of pixels">

        cfmenuitem tags

</cfmenu>
```

cfmenu タグの本文には、メニューアイテムを定義する cfmenuitem タグを少なくとも 1 つ含める必要があります。また、このタグには終了タグ </cfmenu> が必要です。

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfajaximport、cfmenuitem、『ColdFusion アプリケーションの開発』の Using Ajax User Interface Components and Features の Using menus and toolbars

履歴

ColdFusion 8: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
bgColor	オプション	メニューの背景色スタイル	メニューの背景の色です。有効な HTML カラーを指定できます。 指定による動作は次のとおりです。 • この属性で指定した色よりも、このタグの menuStyle 属性および cfmenuitem タグで指定した色のほうが優先されます。 • childStyle 属性によって背景が指定されるサブメニューの背景色を制御します。
childStyle	オプション		次のメニューアイテムに適用される CSS スタイル指定です。 • トップレベルのメニューアイテム • すべての子メニューアイテム (サブメニューの子にも適用されます) この属性を使用すると、すべてのメニューアイテムに特定のスタイル仕様を適用できます。
font	オプション	ブラウザのデフォルトフォント	すべての子メニューアイテムで使用されるフォントです。有効な HTML フォントファミリースタイル属性を使用します。たとえば、serif、sans-serif、Times、Courier、Arial などの値を指定できます。
fontColor	オプション	black	メニューのテキストの色です。有効な HTML カラーを指定します。
fontSize	オプション	メニューアイテムのフォントサイズ	フォントのサイズです。文字のピクセル数を指定するには、8 などの数値を使用します。デフォルトのフォントサイズに対する相対的なサイズを指定するには、80% などのパーセント値を使用します。 20 ピクセルを超えるフォントサイズを指定すると、サブメニューのテキストがメニューの境界からはみ出す可能性があります。
menuStyle	オプション		メニューに適用される CSS スタイル指定です。アイテムが存在しないメニュー部分にも適用されます。 cfmenuitem タグでスタイル情報を指定しない場合は、この属性によってトップレベルアイテムのスタイルが決定されます。
name	オプション		メニューの名前です。
selectedFontColor	オプション	black	フォーカスを持つメニューアイテムのテキストの色です。有効な HTML カラーを指定します。

属性	必須 / オプション	デフォルト	説明
selectedItemColor	オプション	light blue	フォーカスを持つメニューアイテムを強調表示する色です。有効な HTML カラーを指定できます。
type	オプション	horizontal	メニューの向きです。使用できる値は次のとおりです。 • horizontal: メニューアイテムを水平方向に配置します。 • vertical: メニューアイテムを垂直方向に配置します。 どちらのタイプでも、サブメニューは常に垂直方向に配置されます。
width	オプション	コンテナの幅	垂直方向に配置されたメニューの幅です。水平方向に配置されたメニューには適用されません。 ピクセル数を指定するには、50 などの数値を使用します。親要素に対する相対的なサイズを指定するには、30% などのパーセント値を使用します。

使用方法

cfmenu タグは、水平方向または垂直方向に配置される ColdFusion メニューを定義します。cfmenu タグを使用してメニューの全般的な特性を定義し、子タグの cfmenuitem を使用して個々のメニューエントリとサブメニューを定義します。cfmenuitem タグの本文に cfmenuitem タグを配置すると、サブメニューが作成されます。

フォーム、cfmenu タグ、または cfmenuitem タグの中に cfmenu タグをネストすることはできません。

例

次の例では簡単なメニューバーを作成します。メニューバーのエントリをクリックすると、選択した製品に関するアドビ システムズ社の Web サイトページがブラウザに表示されます。アイコンをクリックすると ColdFusion アイテムが展開されます。そこでアイテムを選択すると、特定の ColdFusion Web ページを表示できます。

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
</head>
<body>
<cfmenu name="menu" type="horizontal" fontsize="14" bgcolor="##CCFFFF">
    <cfmenuitem name="acrobat" href="http://www.adobe.com/acrobat" display="Acrobat"/>
    <cfmenuitem name="aftereffects" href="http://www.adobe.com/aftereffects"
        display="After Effects"/>
    <!--- The ColdFusion menu item has a pop-up menu. --->
    <cfmenuitem name="coldfusion"
            href="http://www.adobe.com/products/coldfusion" display="ColdFusion">
        <cfmenuitem name="buy"
            href="http://www.adobe.com/products/coldfusion/buy/" display="Buy"/>
        <cfmenuitem name="devcenter"
            href="http://www.adobe.com/devnet/coldfusion/" display="Developer Center"/>
        <cfmenuitem name="documentation"
            href="http://www.adobe.com/support/documentation/en/coldfusion/"
                display="Documentation"/>
        <cfmenuitem name="support" href="http://www.adobe.com/support/coldfusion/"
            display="Support"/>
    </cfmenuitem>
    <cfmenuitem name="flex" href="http://www.adobe.com/flex" display="Flex"/>
</cfmenu>
</body>
</html>
```

cfmenuitem

説明
サブメニューの頭になるアイテムを含め、メニュー内の個々のエントリを定義します。

カテゴリ
表示管理タグ

シンタックス
```
<cfmenuitem
    display="string"
    childStyle="CSS style specification"
    href="URL or JavaScript function"
    image="path"
    menuStyle="CSS style specification"
    name="string"
    style="CSS style specification"
    target="location identifier">
        Optional child menuitem tags
</cfmenuitem>

OR

<cfmenuitem
    divider[="true"]/>
```

cfmenuitem タグの本文に終了タグ </cfmenuitem> がない場合は、大なり記号の前にスラッシュを付けて (/>) タグを閉じます。たとえば、<cfmenuitem divider="true"/> のように記述します。

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目
cfmenu、『ColdFusion アプリケーションの開発』の Using Ajax User Interface Components and Features の Using menus and toolbars

履歴
ColdFusion 8: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
display	divider 属性を指定しない場合は必須		メニューアイテムのラベルとして表示されるテキストです。
childStyle	オプション	親のスタイルにより決定	すべての子メニューアイテムに適用される CSS スタイル指定です。サブメニューの子にも適用されます。
divider	オプション		そのアイテムがディバイダであることを指定します。この属性を指定した場合、その他の属性は指定できません。この属性は、次の例のように値なしで使用できます。 <cfmenuitem divider /> この属性はトップレベルの水平方向のメニューには使用できません。
href	オプション		ユーザーがメニューアイテムをクリックしたときに呼び出す URL リンクまたは JavaScript 関数です。

属性	必須 / オプション	デフォルト	説明
image	オプション		メニューアイテムの左側に表示する画像の URL です。ブラウザで表示可能なすべてのファイルタイプを指定できます。 通常は 15x15 ピクセルの画像を使用する必要があります。それ以上のサイズになると、メニューアイテムのテキストに重なる場合があります。
menuStyle	オプション	親のスタイルにより決定	このメニューアイテムのサブメニュー全体に適用される CSS スタイル指定です。この属性は現在のメニューアイテムのサブメニューを制御しますが、サブメニューの子は制御しません。
style	オプション	親のスタイルにより決定	現在のメニューアイテムのみに適用される CSS スタイル指定です。childStyle 属性よりも優先されます。
name	オプション		メニューアイテムの名前です。
target	オプション	現在のウィンドウおよびフレーム (存在する場合)	href 属性で返されたコンテンツを表示するターゲットです。ブラウザのウィンドウ名やフレーム名の他に、_self などの HTML ターゲット値を指定できます。

使用方法

cfmenuitem タグは、cfmenu タグまたは cfmenuitem タグの子にする必要があります。サブメニューを作成するには、親メニューのサブメニュールートの cfmenuitem タグ本文内に、サブメニューアイテムの cfmenuitem タグを配置します。簡単なサブメニューの例については、cfmenu を参照してください。

例

次のメニューは、さまざまなスタイル属性がメニューやメニューアイテムの外観にどのような効果を与えるかを示しています。

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
</head>
<body>
<cfmenu name="menu" type="horizontal" fontsize="14" bgcolor="##FF9999"
    childStyle="font-weight:bold; font-size:12px; border:medium; background-color:##99FF99"
    menuStyle="font-weight:bold; font-style:italic; font-size:14px;
    background-color:##9999FF">
    <cfmenuitem name="acrobatInfo"
        href="http://www.adobe.com/acrobat" display="Acrobat"/>
    <cfmenuitem name="aftereffectsInfo"
        href="http://www.adobe.com/aftereffects" display="After Effects"/>
    <!--- The ColdFusion menu item has a pop-up menu. --->
    <cfmenuitem name="cfInfo"
        childStyle="font-weight:bold; font-size:12px; border:medium;
        background-color:##FF0000" style="font-weight:bold;
        font-style:italic; font-size:16px; border:medium; background color:##00FF00"
        menuStyle="font-weight:bold; font-style:italic; font-size:16px;
        border:medium; background-color:##0000FF"
        href="http://www.adobe.com/products/coldfusion" display="ColdFusion">
    <cfmenuitem name="cfbuy"
        href="http://www.adobe.com/products/coldfusion/buy/" display="Buy"/>
        <cfmenuitem divider="true"/>
        <cfmenuitem name="cfdevcenter"
            href="http://www.adobe.com/devnet/coldfusion/" display="Developer Center"/>
        <cfmenuitem name="cfdocumentation"
            href="http://www.adobe.com/support/documentation/en/coldfusion/"
            display="Documentation">
            <cfmenuitem name="cfmanuals"
                href="http://www.adobe.com/support/documentation/en/coldfusion/
```

```
                index.html##manuals" display="Product Manuals"/>
            <cfmenuitem name="cfrelnotes"
                href="http://www.adobe.com/support/documentation/en/coldfusion/
                releasenotes.html" display="Release Notes"/>
        </cfmenuitem>
        <cfmenuitem name="cfsupport"
                href="http://www.adobe.com/support/coldfusion/" display="Support"/>
    </cfmenuitem>
    <cfmenuitem name="flexInfo" href="http://www.adobe.com/flex" display="Flex">
        <cfmenuitem name="fldocumentation"
            href="http://www.adobe.com/support/documentation/en/flex/"
            display="Documentation" >
            <cfmenuitem name="flmanuals"
                href="http://www.adobe.com/support/documentation/en/flex/
                index.html##manuals" display="Product Manuals" />
        </cfmenuitem>
    </cfmenuitem>
</cfmenu>
</body>
</html>
```

cfmessagebox

説明

ポップアップメッセージを表示するコントロールを定義します。標準の警告ボックスとは異なり、このコントロールには、ボックス内にプロンプトと入力フィールドを表示する機能などが備わっています。

カテゴリ

表示管理タグ

シンタックス

```
<cfmessagebox
    bodyStyle = "CSS style specification"
    buttonType = "yesno|yesnocancel"
    callbackHandler = "function name"
    icon = "error|info|question|warning"
    labelCancel = "Cancel button label text"
    labelNo = "No button label text"
    labelOk = "OK button label text"
    labelYes= "Yes button label text"
    message = "message text"
    modal = "yes|no"
    multiline = "false|true"
    name = "control name"
    title = "title"
    type = "alert|confirm|prompt"
    width = "number of pixels"
    x = "numeic pixel coordinate"
    y = "numeic pixel coordinate"/>
```

履歴

ColdFusion 9: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
bodyStyle	オプション		メッセージボックス本体の CSS スタイル指定です。原則として、この属性は色とフォントスタイルを設定する目的に使用します。
buttonType	オプション	yesno	コントロールのタイプ - confirm に適用されます。 メッセージボックスに表示するボタンは次のとおりです。 • yesno: [はい] と [いいえ] の 2 つのボタンを表示します。 • yesnocancel: [はい]、[いいえ]、および [キャンセル] のボタンを表示します。
callbackhandler	オプション		ユーザーがいずれかのボタンをクリックしたときにコントロールによって呼び出される関数です。詳細については、「使用方法」を参照してください。
icon	オプション		次の CSS クラスを指定します。 • error: エラーアイコンを示します。エラーメッセージを表示する場合にこのアイコンを使用できます。 • info: 情報アイコンを示します。情報を表示する場合にこのアイコンを使用できます。 • question: 質問アイコンを示します。ユーザーレスポンスを要求する確認メッセージボックスでこのアイコンを使用できます。 • warning: 警告アイコンを示します。警告メッセージを表示する場合にこのアイコンを使用できます。
labelCancel	オプション	Cancel	prompt タイプのメッセージボックスのキャンセルボタンに表示されるテキストです。
labelOk	オプション	OK	アラートボタンと、prompt タイプのメッセージボックスの OK ボタンに表示されるテキストです。
labelNo	オプション	No	confirm タイプのメッセージボックスの NO のボタンに表示されるテキストです。
labelYes	オプション	Yes	confirm タイプのメッセージボックスの YES のボタンに表示されるテキストです。
message	オプション		メッセージボックス内部に表示されるテキストです。
modal	オプション	yes	メッセージボックスがモーダルウィンドウかどうかを指定するブール値です。 • yes • no
multiline	オプション	false	prompt タイプのメッセージボックスでのみ有効です。prompt タイプのメッセージボックスの入力テキストボックスが単一行であるか、または複数行であるかを指定するブール値です。
name	必須		コントロールの名前です。JavaScript 内でコントロールを参照するために使用されます。
title	オプション		メッセージボックスのタイトルです。 タイトルを指定しない場合、ColdFusion では、コントロールのタイプ値がデフォルトタイトルとして割り当てられます。

属性	必須 / オプション	デフォルト	説明
type	必須		コントロールタイプです。次のいずれかになります。 • alert - OK ボタンが 1 つ表示されるメッセージです。 • confirm - 2 つ (YES と NO) または 3 つ (YES、NO、CANCEL) のボタンが表示されるメッセージボックスです。 • prompt - 単一行または複数行のテキスト入力領域、および OK と CANCEL のボタンが表示されるメッセージです。
width	オプション		メッセージボックスの幅です (単位 : ピクセル)。
x	オプション		メッセージボックスの左上隅の x (水平) 座標です。 y 属性を設定しない場合、この属性は無視されます。
y	オプション		メッセージボックスの左上隅の y (垂直) 座標です。 x 属性を設定しない場合、この属性は無視されます。

使用方法

cfmessagebox はメッセージボックスを作成しますが、表示はしません。JavaScript コードでたとえば mymessagebox というメッセージボックスを表示するには、次のようにします。

```
ColdFusion.MessageBox.show("mymessagebox");
```

callbackhandler を指定した場合、メッセージボックス内のボタンをクリックすると、ボタンラベルをパラメータとして渡すことによって、callbackhandler が呼び出されます。プロンプトボックスの場合は、プロンプトテキストを含む追加のパラメータも渡されます。

alert および confirm タイプのボックス :

```
var function_name = function(button);
```

prompt タイプのボックス :

```
var function_name = function(button, promptmessage);
```

EventObject パラメータは、押されたボタンの JavaScript ID です。名前ではありません。

textmessage パラメータは、プロンプトテキストボックスの内容を含む文字列です。

例

次に、それぞれのタイプのメッセージボックスを表示する 3 つのボタンの例を示します。メッセージボックスのラベルはカスタマイズされており、クリックされたボタンのタイプがメッセージボックスのコールバック関数によって表示されます。

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Untitled Document</title>

<script  type="text/javascript">
    //Function to to show result of a message box.
    var showResult1 = function(btn,message){
        alert("You entered: "+message);
    }
    //Function to show results of other message boxes.
    var showResult2 = function(btn){
        alert("You clicked button: "+btn);
    }

    //The button onClick handler displays the message boxes.
    function showMB(mbox)  {
        ColdFusion.MessageBox.show(mbox);
    }
</script>
</head>

<body>
<cfform>
    <p>Click a button display the corresponding message box.</p>
    <cfinput name="Prompt" type="button" value="Prompt"
        onclick="showMB('mymessagebox01')">
    <cfinput name="Prompt" type="button" value="Prompt"
        onclick="showMB('mymessagebox02')">
    <cfinput name="Prompt" type="button" value="Prompt"
        onclick="showMB('mymessagebox03')">
    </cfform>

<!--- Code to define the message boxes. --->
<cfmessagebox name="mymessagebox01" type="prompt"
        message="Write a short description about yourself"
        labelOK="This is OK" labelCANCEL="Cancel this"
           callbackhandler="showResult1" multiline="true"/>

<cfmessagebox name="mymessagebox02" type="confirm"
        message="Is it OK to save the planet?"
        labelNO="Dont Save" labelYES="Sure"
        callbackhandler="showResult2"/>

<cfmessagebox name="mymessagebox03" type="alert"
        message="You have been ALERTED!"
        callbackhandler="showResult2" />

</body>
</html>
```

cfmodule

説明

ColdFusion アプリケーションページで使用するカスタムタグを呼び出します。このタグにより、カスタムタグの名前の重複が処理されます。

カテゴリ
アプリケーションフレームワークタグ

シンタックス
```
<cfmodule
    attributeCollection = "collection structure"
    attribute_name1 = "valuea"
    attribute_name2 = "valueb"
    name = "tag name"
    template = "path"
    ...>
```

関連項目
cfapplication、cfassociate、cflock、『ColdFusion アプリケーションの開発』の Creating and Using Custom CFML Tags

履歴
ColdFusion MX: このタグをカスタムタグ内で使用したときの動作が変更されました。attribute_name パラメータが attributeCollection パラメータ内の key 要素と同じである場合は、attributeCollection パラメータ内の name 値が使用されるようになりました。以前のリリースでは、この処理は一貫していませんでした。

属性

属性	必須 / オプション	デフォルト	説明
attributeCollection	オプション		構造 . 属性名とその値を表す、キーと値のペアのコレクションです。キーと値のペアを複数指定できます。この属性は、一度だけ指定することができます。 **メモ :** この属性の機能は、他のほとんどのタグでサポートされている attributeCollection 属性とは異なります。name 属性と template 属性は、attributeCollection 構造体内ではなく、cfmodule タグの直接の属性として指定する必要があります。
attribute_name	オプション		カスタムタグの属性です。この属性の複数のインスタンスを入れて、カスタムタグのパラメータを指定することができます。
name	template 属性が使用されない場合は必須		template 属性とは排他の関係です。"Name.Name.Name..." という形式でカスタムタグの名前を指定します。ColdFusion タグのルートディレクトリの下の、カスタムタグページが含まれているサブディレクトリを特定します。例 : (Windows 形式の場合) <cfmodule name = ".Forums40. GetUserOptions"> これにより、ColdFusion ルートディレクトリの下にあるディレクトリ CustomTags¥¥Forums40 内のページ GetUserOptions.cfm が特定されます。
template	name 属性が使用されない場合は必須		name 属性とは排他の関係です。タグを実装するページへのパスです。 • 相対パスは、現在のページから展開されます。 • 絶対パスは、ColdFusion のマッピングを使用して展開されます。 物理パスは有効ではありません。

使用方法
カスタムタグ定義が含まれている ColdFusion ページにパスを含めた名前を付けるには、template 属性を使用します。位置をドット表記で示して ColdFusion インストールディレクトリ内のカスタムタグを参照するには、name 属性を使用します。

UNIX システムでは、まず、name 属性と一致するファイル (ただしすべて小文字) が検索されます。該当するファイルが見つからない場合は、大文字と小文字の区別も含めて属性と一致するファイル名が検索されます。

同じ呼び出し内で、attributeCollection 属性と明示的なカスタムタグ属性を使用することができます。

カスタムタグコード内では、attributeCollection を使用して渡された属性は独立した属性値として保存され、その属性がカスタムタグの呼び出し側によって構造体にグループ化されたことは示されません。

同様に、カスタムタグが cfassociate タグを使用してその属性を保存している場合、attributeCollection によって渡された属性は独立した属性値として保存され、その属性がカスタムタグの呼び出し側によって構造体にグループ化されたことは示されません。

終了タグを cfmodule に指定する場合、ColdFusion では開始タグと終了タグの両方がある場合と同様にカスタムタグを呼び出します。詳細については、『ColdFusion アプリケーションの開発』の Handling end tags を参照してください。

例

```
 <h3>cfmodule Example</h3>
<p>This view-only example shows use of cfmodule to call a custom tag inline.</p>
<p>This example uses a sample custom tag that is saved in myTag.cfm in the snippets directory.
You can also save ColdFusion custom tags in the CFusionMX7\CustomTags directory.</p>
<cfset attrCollection1 = StructNew()>
    <cfparam name="attrCollection1.value1" default="22">
    <cfparam name="attrCollection1.value2" default="45">
    <cfparam name="attrcollection1.value3" default="88">
<!--- Call the tag with CFMODULE with Name--->
<cfmodule
    Template="myTag.cfm"
    X="3"
    attributeCollection=#attrCollection1#
    Y="4">
<!--- Show the code. --->
<HR size="2" color="#0000A0">
<P>Here is one way in which to invoke the custom tag, using the TEMPLATE attribute.</P>
<cfoutput>#HTMLCodeFormat(" <CFMODULE
    Template=""myTag.cfm""
    X=3
    attributeCollection=##attrCollection1##
    Y=4>")#
</cfoutput>
<p>The result: <cfoutput>#result#</cfoutput></p>
<!--- Call the tag with CFMODULE with Name.--->
<!---
<CFMODULE
    Name="myTag"
    X="3"
    attributeCollection=#attrCollection1#
    Y="4">
--->
<!--- Show the code. --->
<HR size="2" color="#0000A0">
<p>Here is another way to invoke the custom tag, using the NAME attribute.</p>
<cfoutput>#HTMLCodeFormat(" <CFMODULE
    NAME='myTag'
    X=3
```

```
    attributeCollection=##attrCollection1##
    Y=4>")#
</cfoutput>
<P>The result: <cfoutput>#result#</cfoutput></p>
<!--- Call the tag using the shortcut notation. --->
<!---
<CF_myTag
    X="3"
    attributeCollection=#attrCollection1#
    Y="4">
--->

<!--- Show the code. --->
<p>Here is the short cut to invoking the same tag.</p>
<cfoutput>#HTMLCodeFormat("<cf_mytag
    x = 3
    attributeCollection = ##attrcollection1##
    y = 4>")#
</cfoutput>
<p>The result: <cfoutput>#result#</cfoutput></p>
```

cfNTauthenticate

説明

ColdFusion サーバーが稼動している Windows NT ドメインに対してユーザー名とパスワードを認証し、必要に応じてユーザーのグループを取得します。

カテゴリ

セキュリティタグ

シンタックス

```
<cfNTauthenticate
    domain="NT domain"
    password="password"
    username="user name"
    listGroups = "yes|no"
    result="result variable"
    throwOnError = "yes|no">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cflogin、cfloginuser、IsUserInAnyRole、GetAuthUser

履歴

ColdFusion MX 7: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
domain	必須		ユーザーを認証する対象のドメインです。ColdFusion J2EE サーバーはこのドメインで稼動している必要があります。
password	必須		ユーザーのパスワードです。
username	必須		ユーザーのログイン名です。
listGroups	オプション	No	ユーザーのグループをカンマ区切りで示すリストを結果の構造体に含めるかどうかを指定するブール値です。
result	オプション	cfntauthenticate	結果を返す変数の名前です。
throwOnError	オプション	no	認証が失敗したときに例外が発生するかどうかを指定するブール値です。この属性が yes の場合、ユーザー名またはパスワードが無効のときは ColdFusion でエラーが発生します。アプリケーションでは、try/catch ブロックまたは ColdFusion エラーハンドラページでこうしたエラーを処理する必要があります。

使用方法

この関数を使用して、Windows NT ドメインに対してユーザーを認証し、必要に応じてユーザーのグループを取得します。この関数は、Microsoft Active Directory ディレクトリサービスでは機能せず、UNIX システムおよび Linux システムでは何も実行しません。通常は、後の例に示すように、このタグを cflogin タグ内で使用して cfloginuser タグに対してユーザーを認証します。

注意：ColdFusion は、指定されたドメイン内の他のユーザーを認証する権限があるユーザーとして実行する必要があります。

result 属性で指定された構造体には、次の情報が格納されています。

フィールド	値
auth	ユーザーを認証するかどうかを指定します。 • yes • no
groups	指定されたドメイン内のユーザーのグループを示すカンマ区切りのリストです。構造体にこのフィールドが含まれるのは、listGroups 属性が yes の場合のみです。
name	ユーザー名です。タグの username 属性と同じです。
status	認証のステータスです。次のいずれかです。 • success • UserNotInDirFailure: ユーザーはディレクトリに存在しません。 • AuthenticationFailure: ユーザーはディレクトリに存在しますが、パスワードが無効です。

このタグは、認証の処理について、ステータス確認および例外処理という 2 つのモデルを提供します。throwOnError 属性が no の場合は、結果変数の auth フィールドと status フィールドを使用して、ユーザーが認証されたかどうかを確認します。認証されなかった場合は、認証失敗の原因を確認します。throwOnError 属性が yes の場合、ユーザーが無効のときに ColdFusion では例外エラーが発生します。その場合は、try/catch エラー処理を使用します。catch ブロックでは、あらゆる認証の失敗を処理します。

例

次の例では、auth フィールドと status フィールドを使用して、ユーザーが認証されるかどうかおよびエラーが発生するかどうかを確認します。同じディレクトリに置いた次の 3 つのファイルで構成されています。

- メインの cfntauthexample.cfm ページ。ユーザーが認証された場合にそのユーザー名を表示します。ログアウトリンクを含んでいます。
- ログインフォームページ。ユーザーがログインしていない場合に表示されます。
- Application.cfm ページ。ログイン、認証、およびログアウトのすべての処理コードを含んでいます。

ログイン処理の詳細については、『ColdFusion アプリケーションの開発』を参照してください。この例の詳細については、コード内のコメントを参照してください。

次のページを cfntauthenticateexample.cfm として保存します。この例を実行するには、ブラウザまたは IDE からこのページをリクエストします。

```
<!--- The Application.cfm page, which is processed each time a user
    requests this page, ensures that you log in first. --->
<cfoutput>
    <h3>Welcome #GetAuthUser()#</h3>
    <!--- A link to log out the user. --->
    <a href="#CGI.script_name#?logout=Yes">Log Out</a>
</cfoutput>
```

次のページを loginform.cfm として保存します。

```
<!--- A simple login form that posts back to the page whose request initiated the login. --->
<h2>Please Log In</h2>
<cfform action="#CGI.script_name#">
    <!--- j_username and j_password are special names that populate cflogin tag
        variables. --->
    User Name: <cfinput type="text" name="j_username" value="cfqa_user1" required="Yes"><br>
    Password: <cfinput type="password" name="j_password" value="cfqa_user1"
        required="Yes"><br>
    Domain: <cfinput type="text" name="domain" value="rnd" required="Yes"><br>
    <input type="submit" value="Log In">
</cfform>
```

次のページを Application.cfm として保存します。

```
<!--- If this page is executing in response to the user clicking a logout link,
        log out the user. The cflogin tag code will then run. --->
<cfif IsDefined("URL.logout") AND URL.logout>
    <cflogout>
</cfif>

<!--- The cflogin body code runs only if a user is not logged in. --->
<cflogin>
    <!--- cflogin variable exists only if login credentials are available. --->
    <cfif NOT IsDefined("cflogin")>
        <!--- Show a login form that posts back to the page whose request
            initiated the login, and do not process the rest of this page. --->
        <cfinclude template="loginform.cfm">
        <cfabort>
    <cfelse>
        <!--- Trim any leading or trailing spaces from the username and password
            submitted by the form. --->
        <cfset theusername=trim(form.j_username)>
        <cfset thepassword=trim(form.j_password)>
        <cfset thedomain=trim(form.domain)>
        <cfntauthenticate username="#theusername#" password="#thepassword#"
            domain="#thedomain#" result="authresult" listgroups="yes">
        <!--- authresult.auth is True if the user is authenticated. --->
```

```
        <cfif authresult.auth>
            <!--- Log user in to ColdFusion and set roles to the user's Groups. --->
            <cfloginuser name="#theusername#" password="#thepassword#"
                roles="#authresult.groups#">
        <cfelse>
            <!--- The user was not authenticated.
                    Display an error message and the login form. --->
            <cfoutput>
                <cfif authresult.status IS "AuthenticationFailure">
                    <!--- The user is valid, but not the password. --->
                    <h2>The password for #theusername# is not correct<br>
                        Please Try again</h2>
                <cfelse>
                    <!--- There is one other status value, invalid user name. --->
                    <H2>The user name #theusername# is not valid<br>
                        Please Try again</h2>
                </cfif>
            </cfoutput>
            <cfinclude template="loginform.cfm">
            <cfabort>
        </cfif>
    </cfif>
</cflogin>
```

cfobject

説明
指定されたタイプの ColdFusion オブジェクトを作成します。

注意：ColdFusion Administrator の ColdFusion セキュリティの [サンドボックスセキュリティ] で、このタグを有効または無効にすることができます。

カテゴリ
拡張タグ

シンタックス
タグのシンタックスはオブジェクトタイプによって異なります。type 属性を使用するタイプと、使用しないタイプがあります。次のセクションを参照してください。

- 461 ページの「cfobject: .NET オブジェクト」
- 464 ページの「cfobject: COM オブジェクト」
- 466 ページの「cfobject: コンポーネントオブジェクト」
- 467 ページの「cfobject: CORBA オブジェクト」
- 468 ページの「cfobject: Java または EJB オブジェクト」
- 470 ページの「cfobject: Web サービスオブジェクト」

注意：UNIX 上では、このタグは COM オブジェクトをサポートしません。

関連項目
cfargument、cfcomponent、cffunction、cfinvoke、cfinvokeargument、cfproperty、cfreturn、『ColdFusion アプリケーションの開発』の Using Java objects

履歴
ColdFusion 8:

- Web サービスオブジェクトで使用するために、password、proxyPassword、proxyPort、proxyServer、proxyUser、refreshWSDL、userName、wsdl2JavaArgs、wsportname 属性が追加されました。
- .NET タイプおよび dotnet タイプと、それに関連する assembly、port、protocol、secure 属性が追加されました。

ColdFusion MX:

- インスタンス作成時の動作が変更されました。このタグと CreateObject 関数によって、ColdFusion コンポーネント (CFC) のインスタンスを作成できるようになりました。これらは cfscript タグ内で使用できます。
- CORBA オブジェクトの場合 : アドレスのネーミングサービスセパレータの形式が、ピリオドからスラッシュに変更されました。たとえば、あるクラスに "context=NameService" が指定されている場合、class パラメータには、次のいずれかの形式を使用します。
  - "/Eng/CF"
  - ".current/Eng.current/CF"

  以前のリリースでは、形式は ".Eng.CF" でした。
- CORBA オブジェクトの場合 : locale 属性が変更されました。この属性では、プロパティファイルを含む Java 設定を指定します。

cfobject: .NET オブジェクト

説明
.NET オブジェクトを作成します。このオブジェクトは、ローカルまたはリモートの .NET アセンブリに存在するクラスにアクセスするための ColdFusion プロキシになります。

シンタックス

```
<cfobject
    class="class name"
    name="instance name"
    type=".NET|dotnet"
    action="create"
    assembly="absolute path"
    port="6086"
    protocol="tcp|http"
    secure="no|yes"
    server = "localhost">
```

注意 : このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目
CreateObject: .NET オブジェクト、DotNetToCFType、『ColdFusion アプリケーションの開発』の Using Microsoft .NET Assemblies

履歴
ColdFusion 8: .NET と dotnet タイプの値、assembly、port、protocol、および secure の各属性が追加されました。

属性

属性	必須 / オプション	デフォルト	説明
class	必須		オブジェクトとしてインスタンス化する .NET クラスの名前です。
name	必須		文字列です。アプリケーションで使用するコンポーネントの参照名を指定します。
type	.NET の場合は必須		オブジェクトのタイプです。.NET オブジェクトの場合は .NET または dotnet に設定する必要があります。
action	オプション	create	実行するアクションです。create に設定する必要があります。
assembly	オプション	.NET コアクラスを含む mscorlib.dll	**ローカル .NET アセンブリの場合** : .NET クラスとそのサポートクラスにアクセスするために使用するアセンブリ (EXE または DLL ファイル) の絶対パスです (複数指定可)。アセンブリ内のクラスが他のアセンブリ内のサポートクラスを必要とする場合は、それらのアセンブリも指定する必要があります。ただし、次のタイプのサポートクラスについては、サポートアセンブリの指定を省略できます。 • .NET コアクラス (mscorlib.dll に含まれるクラス) • GAC (Global Assembly Cache) に存在するアセンブリ内のクラス 複数のアセンブリを指定するには、カンマ区切りのリストを使用します。 **リモート .NET アセンブリの場合** : アセンブリの代わりとして使用するローカルプロキシ JAR ファイルの絶対パスを指定する必要があります (複数指定可)。 ローカルに .NET がインストールされていない場合にこの属性を省略すると、このタグは機能せず、エラーも生成されません。ローカルに .NET がインストールされている場合、この属性を省略して、指定したクラスが .NET コアクラスに存在しないときは、エラーが生成されます。
port	オプション	6086	.NET サイドのエージェントがリスンするポート番号です。
protocol	オプション	tcp	ColdFusion と .NET の通信に使用するプロトコルです。有効な値は次のとおりです。 • http: HTTP/SOAP 通信プロトコルを使用します。tcp より遅くなりますが、ファイアウォール経由でアクセスする場合に必要になることがあります。 • tcp: バイナリの TCP/IP プロトコルを使用します。HTTP よりも効率的です。
secure	オプション	false	.NET サイドのエージェントとの通信を暗号化するかどうかを指定します。true を指定すると、.NET との通信に SSL が使用されます。
server	オプション	localhost	.NET サイドのエージェントが実行されているサーバーのホスト名または IP アドレスです。次のいずれかの形式で指定します。 • サーバー名 (例 : myserver) • IP アドレス (例 : 127.0.0.1) リモートサーバー上の .NET コンポーネントにアクセスする場合は、この属性を指定する必要があります。

使用方法

cfobject タグで type 属性の値を .NET または dotnet に設定すると、指定したクラスの .NET オブジェクトへの参照が作成されます。この参照を使用して、.NET オブジェクトのフィールドとメソッドにアクセスできます。.NET クラスは必ずしもローカルに存在している必要はないので、.NET がインストールされていないシステム (UNIX ベース、OS-X ベースのシステムも含む) でも cfobject タグを使用できます。

.NET アセンブリにアクセスする手順は次のとおりです。

- アセンブリがインストールされているシステムに ColdFusion .NET Extension をインストールして、.NET Extension サービスを実行します。リモートアセンブリのみにアクセスする ColdFusion システムでは、Extension をインストー

ルして Extension サービスを実行する必要はありません。インストール手順については、『ColdFusion インストール』を参照してください。

- アセンブリがリモートシステム上に存在する場合は、使用する .NET クラス用の Java プロキシを作成し、そのプロキシを ColdFusion システムにコピーします。次に、プロキシからアクセスできるように、リモートシステムを設定します。これらの手順については、『ColdFusion アプリケーションの開発』の Using Microsoft .NET Assemblies を参照してください。.NET アセンブリが ColdFusion システム上にある場合は、この手順を実行する必要はありません。

メソッドとフィールドへのアクセス

他の ColdFusion オブジェクトのメソッドを使用する場合と同様に .NET メソッドを呼び出します。簡単なアプリケーションでは、コード内で次の形式を使用して .NET クラスのメソッドを呼び出します。

```
<cfobject type=".NET" name="mathInstance" class="mathClass">
    assembly="C:/Net/Assemblies/math.dll">
<cfset myVar=mathInstance.multiply(1,2)>
```

.NET クラスに複数のコンストラクタがある場合、デフォルトのコンストラクタを使用してオブジェクトを作成しないようにするには、コンストラクタの引数を使用して ColdFusion オブジェクトの特別な init メソッドを呼び出すことにより、特定のコンストラクタを呼び出します。たとえば、次のタグを使用すると com.foo.MyClass(int, int) をインスタンス化できます。

```
<cfobject type=".NET" class="com.foo.MyClass"
    assembly="c:\temp\myLib.dll" name="myObj" >
<cfset myObj.init(10, 5)>
```

.NET クラスのパブリックフィールドにアクセスして変更を加えるには、次のメソッドを呼び出します。

```
Get_fieldName()
Set_fieldName()
```

たとえば、.NET クラスに account という名前のパブリックフィールドがある場合、そのフィールドの値にアクセスして変更を加えるには Get_acount() メソッドと Set_account() メソッドを使用します。

final フィールドは、アクセスのみが可能で変更は行えません。したがって、呼び出せるのは Get_**fieldName**() のみになります。

例

次の例では、.NET System.Diagnostics.Process クラスの GetProcess メソッドを使用して、ローカルシステムで実行されているプロセスの情報を取得して表示します。この例では .NET コアクラスを使用するため、cfobject タグでアセンブリ名を指定する必要はありません (コアクラス用のプロキシは ColdFusion によって自動的に生成されます)。

カスタムの .NET クラスを使用するさらに複雑な例については、『ColdFusion アプリケーションの開発』の Using Microsoft .NET Assemblies を参照してください。

```
<cfobject type=".NET" name="proc" class="System.Diagnostics.Process">
<cfset processes = proc.GetProcesses()>
<cfset arrLen = arrayLen(processes)>

<table border=0 cellspacing="3" cellpadding="3">
    <tr bgcolor="#33CCCC">
        <td style="font-size:12px; font-weight:bold" nowrap>Process ID</td>
        <td style="font-size:12px; font-weight:bold" nowrap>Name</td>
        <td style="font-size:12px; font-weight:bold" nowrap>Memory (KB)</td>
        <td style="font-size:12px; font-weight:bold" nowrap>Peak Memory (KB)</td>
        <td style="font-size:12px; font-weight:bold" nowrap>Virtual Memory Size (KB)</td>
        <td style="font-size:12px; font-weight:bold" nowrap>Start Time</td>
        <td style="font-size:12px; font-weight:bold" nowrap>Total Processor Time</td>
    </tr>
    <cfloop from = 1 to="#arrLen#" index=i>
        <cfset process = processes[i]>
        <cfset id = process.Get_Id()>
        <cfif id neq 0>
            <cfoutput>
            <tr>
                <td align="right">#process.Get_Id()#</td>
                <td>#process.Get_ProcessName()#</td>
                <td align="right">#process.Get_PagedMemorySize()/1000#</td>
                <td align="right">#process.Get_PeakPagedMemorySize()/1000#</td>
                <td align="right">#process.Get_VirtualMemorySize()/1000#</td>
                <td>#process.Get_StartTime()#</td>
                <td>#process.Get_TotalProcessorTime()#</td>
            </tr>
            </cfoutput>
        </cfif>
    </cfloop>
</table>
```

cfobject: COM オブジェクト

説明

COM (Component Object Model) オブジェクトを作成および操作します。また、登録されている自動サーバーオブジェクトタイプを呼び出します。

OLEView、COM、および DCOM については、Microsoft OLE 開発 Web サイト www.microsoft.com を参照してください。

このタグを使用するには、オブジェクトのプログラム ID またはファイル名、IDispatch インターフェイスから利用できるメソッドとプロパティ、およびオブジェクトのメソッドの引数と戻り値のタイプを指定します。ほとんどの COM オブジェクトについて、OLEView ユーティリティを使用してこの情報を得ることができます。

注意：UNIX 上では、cfobject タグは COM オブジェクトをサポートしません。

シンタックス

```
<cfobject
    class = "program ID"
    name = "instance name"
    action = "create|connect"
    context = "inproc|local|remote"
    server = "server name">
    type = "com"
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

ReleaseComObject、cfcollection、cfexecute、『ColdFusion アプリケーションの開発』の COM

属性

属性	必須 / オプション	デフォルト	説明
class	必須		呼び出すオブジェクトのコンポーネントプログラム ID です。Java スタブを使用して COM オブジェクトに接続するときは、class に COM オブジェクトのプログラム ID を指定する必要があります。
name	必須		文字列です。インスタンス化されたコンポーネント名を指定します。
action	オプション	create	• create: メソッドまたはプロパティを呼び出す前に COM オブジェクト (通常は DLL) のインスタンスを作成します。 • connect: サーバー上で実行されている COM オブジェクト (通常は EXE) に接続します。
context	オプション		• inproc • local • remote Windows では、この属性を指定しない場合、レジストリの設定が使用されます。
server	context = "Remote" の場合は必須		サーバー名です。Universal Naming Convention (UNC) または Domain Name Serve (DNS) の表記規則を使用して、次のいずれかの形式で指定します。 • ¥¥lanserver • lanserver • http://www.servername.com • www.servername.com • 127.0.0.1
type	オプション		オブジェクトのタイプです。com は COM オブジェクトを意味します。

例

```
<h3>cfobject (COM) Example</h3>
<!--- Create a COM object as an inproc server (DLL). (class = prog-id)--->
<cfobject action = "Create"
    type = "COM"
    class = Allaire.DocEx1.1
    name = "obj">

<!--- Call a method. Methods that expect no arguments should be called by using
    empty parentheses. --->
<cfset obj.Init()>

<!--- This is a collection object. It should support, at a minimum:
    Property : Count
    Method : Item(inarg, outarg)
    and a special property called _NewEnum
--->
<cfoutput>
    This object has #obj.Count# items.
    <br> <HR>
</cfoutput>

<!--- Get the 3rd object in the collection. --->
<cfset emp = obj.Item(3)>
<cfoutput>
    The last name in the third item is #emp.lastname#.
    <br> <HR>
</cfoutput>
<!---Loop over all the objects in the collection.--->
<p>Looping through all items in the collection:
<br>
<cfloop
    collection = #obj#
    item = file2>
    <cfoutput>Last name: #file2.lastname# <br></cfoutput>
</cfloop>
```

cfobject: コンポーネントオブジェクト

説明

ColdFusion コンポーネント (CFC) オブジェクトのインスタンスを作成します。

シンタックス

```
<cfobject
    component = "component name"
    name = "instance name"
    type = "component">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfcollection、cfcomponent、cfexecute、cfindex、IsInstanceOf、cfreport、cfsearch、cfwddx、『ColdFusion アプリケーションの開発』の Using ColdFusion components

属性

属性	必須 / オプション	デフォルト	説明
component	必須		インスタンス化するコンポーネントの名前です。
name	必須		文字列です。インスタンス化されたコンポーネント名を指定します。名前の最初と最後の文字にピリオドを使用することはできません。
type	オプション	component	オブジェクトのタイプです。この属性は省略できますが、使用する場合は component を指定します。オブジェクトのタイプは自動的にコンポーネントに設定されます。

使用方法

cfobject タグが CFC のインスタンスを作成するときに、CFC 内のコンストラクタコードが実行されます。つまり、メソッド定義にないコードが実行されます。

UNIX システムでは、まず指定のコンポーネント名と同じ名前 (ただしすべて小文字) のファイルが検索されます。これに該当するファイルが見つからない場合は、大文字小文字の違いも含めて、指定されたコンポーネント名とまったく同じ名前のファイルが次に検索されます。

例

```
<!--- Separate instantiation and method invocation; --->
<!--- permits multiple invocations. --->
<cfobject
    name="quoteService"
    component="nasdaq.quote">
<cfinvoke
    component="#quoteService#"
    method="getLastTradePrice"
    symbol="macr"
    returnVariable="res">
<cfoutput>#res#</cfoutput><br>

<cfinvoke
    component="#quoteService#"
    method="getLastTradePrice"
    symbol="mot"
    returnVariable="res">
<cfoutput>#res#</cfoutput>
```

cfobject: CORBA オブジェクト

説明

登録されている CORBA オブジェクトでメソッドを呼び出します。

シンタックス

```
<cfobject
    class = "filepath or naming service"
    context = "ior|nameservice"
    name = "instance name"
    type = "corba"
    locale = "type-value arguments">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfcollection、cfexecute、cfindex、cfreport、cfsearch、cfwddx、『ColdFusion アプリケーションの開発』の CORBA

履歴

cfobject タグのメインページの「履歴」を参照してください。

属性

属性	必須 / オプション	デフォルト	説明
class	必須		• context="ior" の場合 : 文字列形式の IOR (Interoperable Object Reference) が含まれているファイルの絶対パスです。このファイルは常に、ColdFusion で読み取り可能である必要があります。つまり、このファイルは、ColdFusion サーバーのローカルファイルであるか、またはネットワーク上のアクセス可能な場所に置かれている必要があります。 • context="nameservice" の場合 : スラッシュで区切った、ネーミングサービスに対するネーミングコンテキストです。例 : Allaire//Doc/empobject
context	必須		• ior: ColdFusion では IOR (Interoperable Object Reference) を使用して CORBA サーバーにアクセスします。 • nameservice: ColdFusion ではネーミングサービスを使用してサーバーにアクセスします。このオプションは、VisiBroker Orb の InitialContext の使用時のみ有効です。
locale	オプション		init_orb の呼び出しの引数を設定します。この属性は、VisiBroker ORB に対してのみ使用してください。これは C++ バージョン 3.2 で使用可能です。この値は次の形式で指定する必要があります。 locale = " -ORBagentAddr 199.99.129.33 -ORBagentPort 19000" タイプと値の各ペアの先頭には、ハイフンを付ける必要があります。
name	必須		文字列です。インスタンス化されたコンポーネント名を指定します。アプリケーションでは、この名前を使用して CORBA オブジェクトのメソッドと属性を参照します。
type	CORBA の場合は必須		オブジェクトのタイプです。CORBA オブジェクトの場合は corba に設定する必要があります。

使用方法

ColdFusion エンタープライズ版バージョン 4.0 以降では、DII (Dynamic Invocation Interface) により CORBA がサポートされます。cfobject を CORBA オブジェクトとともに使用するには、文字列形式の IOR が含まれているファイル名、またはネーミングサービスでのオブジェクトのネーミングコンテキストのいずれかを指定します。また、オブジェクトの属性、メソッド名、およびメソッドのシグネチャも必要です。

構造体などのユーザー定義のタイプはサポートされません。

例

```
<cfobject type = "corba"
    context = "ior"
    class = "c:\\myobject.ior"
    name = "GetName">
```

cfobject: Java または EJB オブジェクト

説明

Java および EJB (Enterprise Java Bean) オブジェクトの作成と操作を行います。

シンタックス

```
<cfobject
    class = "Java class"
    type = "Java"
    name = "instance name"
    action = "create">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfcollection、cfexecute、cfindex、IsInstanceOf、cfreport、cfsearch、cfwddx、『ColdFusion アプリケーションの開発』の Using Java objects

属性

属性	必須 / オプション	デフォルト	説明
action	オプション	create	指定できるアクションはデフォルトの create のみです。このアクションはオブジェクトを作成します。
class	必須		Java クラスです。
name	必須		文字列です。インスタンス化されたコンポーネント名を指定します。
type	Java の場合は必須		オブジェクトのタイプです。Java または EJB オブジェクトの場合は java に設定する必要があります。

使用方法

ColdFusion では、Java CFX または Java オブジェクトの呼び出しに、処理に組み込まれている JVM (Java Virtual Machine) を使用します。JVM の読み込み、保存場所および設定は、ColdFusion Administrator ページで指定できます。

ColdFusion Administrator で指定したクラスパスで使用可能な Java クラスは、cfobject タグを使用して ColdFusion からロードして使用できます。

Java メソッドとフィールドへのアクセス

1 cfobject タグを呼び出し、クラスをロードします。コード例を参照してください。

2 適切な引数で init メソッドを使用して、コンストラクタを呼び出します。たとえば、次のようになります。

```
<cfset ret = myObj.init(arg1, arg2)>
```

最初に init メソッドを呼び出さずにオブジェクト上のパブリックメソッドを呼び出すと、デフォルトのコンストラクタが暗黙のうちに呼び出されます。引数および戻り値は、任意の Java タイプ (simple、array、object) になります。ColdFusion では、文字列が引数として渡された場合に変換が行われますが、戻り値として返された場合には変換は行われません。

オーバーロードされたメソッドは、引数の数が異なる場合はサポートされます。

EJB の呼び出し

EJB オブジェクトの作成と呼び出しを行うには、cfobject タグを使用します。この後の 2 番目の例では、WebLogic JNDI を使用して EJBHome インスタンスの登録と検出が行われます。

例

```
<!--- Example of a Java Object, this cfobject call loads the class MyClass
    but does not create an instance object. Static methods and fields
    are accessible after a call to cfobject. --->
<cfobject
    action = "create"
    type = "java"
    class = "myclass"
    name = "myobj">


<!---- Example of an EJB - The cfobject tag creates the Weblogic Environment
    object, which is used to get InitialContext. The context object is
    used to look up the EJBHome interface. The call to Create() results
    in getting an instance of stateless session EJB. --->

<cfobject
    action = "create"
    type = "java"
    class = "weblogic/jndi/Environment"
    name = "wlEnv">

<cfset ctx = wlEnv.getInitialContext()>
<cfset ejbHome = ctx.lookup("statelessSession.TraderHome")>
<cfset trader = ejbHome.Create()>
<cfset value = trader.shareValue(20, 55.45)>
<cfoutput>
     Share value = #value#
</cfoutput>
<cfset value = trader.remove()>
```

cfobject: Web サービスオブジェクト

説明

Web サービスプロキシオブジェクトを作成します。

シンタックス

```
<cfobject
    name = "local name">
    webservice= "service identifier"
    password = "string"
    proxyPassword = "string"
    proxyPort = "port number"
    proxyServer = "URL or IP address"
    proxyUser = "string"
    refreshWSDL = "no|yes"
    type = "webservice"
    username = "string"
    wsdl2javaArgs = "argument string"
    wsportname = "port name">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfcollection、cfexecute、cfindex、cfreport、cfsearch、cfwddx、『ColdFusion アプリケーションの開発』の Using Web Services の Consuming web services

履歴

cfobject タグのメインページの「履歴」を参照してください。

属性

属性	必須 / オプション	デフォルト	説明
name	必須		Web サービスのローカル名です。文字列です。
webservice	必須		次のいずれかです。 • Web サービスの絶対 URL • ColdFusion Administrator で Web サービスに割り当てられた名前 (文字列)
password	オプション	存在する場合には、Administrator で設定されたパスワード	Web サービスにアクセスするためのパスワードです。ColdFusion Administrator で設定された Web サービス名が webservice 属性で指定されている場合は、Administrator のエントリで指定されたパスワードではなく、ここで指定されたパスワードが使用されます。
proxyPassword	オプション	存在する場合には、http.proxyPassword システムプロパティ	プロキシサーバー上で使用するユーザーのパスワードです。
proxyPort	オプション	存在する場合には、http.proxyPort システムプロパティ	プロキシサーバー上で使用するポートです。
proxyServer	オプション	存在する場合には、http.proxyHost システムプロパティ	Web サービスの URL にアクセスするために必要なプロキシサーバーです。
proxyUser	オプション	存在する場合には、http.proxyUser システムプロパティ	プロキシサーバーに送信するユーザー ID です。
refreshWSDL	オプション	no	• yes: WSDL ファイルをリロードし、Web サービスを利用するために必要な生成結果を再生成します。 • no
type	オプション		オブジェクトのタイプです。この属性は省略できますが、使用する場合は webservice を指定します。

属性	必須 / オプション	デフォルト	説明
username	オプション	存在する場合には、Administrator で設定されたユーザー名	Web サービスにアクセスするためのユーザー名です。ColdFusion Administrator で設定された Web サービス名が webservice 属性で指定されている場合は、Administrator のエントリで指定されたユーザー名ではなく、ここで指定されたユーザー名が使用されます。
wsdl2javaArgs	オプション		WSDL2Java ツールに渡す引数のスペース区切りリストを含む文字列です。WSDL2Java ツールは Web サービス用の Java スタブを生成するためのツールです。使用可能な引数は次のとおりです。 • -W または --noWrapped: wrapped document/literal スタイルオペレーションの特別な処理を無効にします。 • -a または --all: 参照されないものも含め、WSDL 内の全要素のコードを生成します。 • -w または --wrapArrays: ラップされた XML 配列型に対してストレートな配列を作成するのではなく bean を作成します。このスイッチは Axis のマニュアルには記載されていません。 有効な引数の詳細については、『Apache Axis WSDL2Java Reference』を参照してください。
wsportname	オプション	WSDL 内の最初のポート	Web サービスのポート名です。この値は、service 要素内の port 要素の name 属性に対応します。大文字と小文字が区別されます。 Web サービスに複数のポートが含まれている場合、このパラメータを指定します。

使用方法
Web サービスのプロキシオブジェクトのインスタンスを作成します。このタグでは、絶対 URL の入力や、ColdFusion Administrator で入力された Web サービスの参照ができます。コードの保守の必要性を最小限にするためには、Administrator で Web サービスを入力し、入力した名前をこのタグで参照します。

cfobjectcache

説明
クエリーキャッシュを一括消去します。

カテゴリ
データベース操作タグ

シンタックス
```
<cfobjectcache
    action = "clear">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目
cfobject

履歴
ColdFusion 5: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
action	必須		clear: Application スコープ内で、クエリーをキャッシュから消去します。

cfoutput

説明
ColdFusion の変数および関数の処理結果が含まれる出力を表示します。データベースクエリーの結果をループすることができます。

カテゴリ
データ出力タグ

シンタックス
```
<cfoutput
    group = "query column"
    groupCaseSensitive = "yes|no"
    maxRows = "maximum rows to display"
    query = "query name"
    startRow = "start row">
</cfoutput>
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目
cfcol、cfcontent、cfdirectory、cftable

履歴
ColdFusion 4.5.0: groupCaseSensitive 属性が追加されました。

属性

属性	必須 / オプション	デフォルト	説明
group	オプション		レコードのセットをグループ化するために使用するクエリー列です。データをソートする場合、隣り合う重複行は削除されます。クエリー列で順序付けられたレコードセットを取得した場合に使用します。たとえば、cfquery タグ内で "Customer_ID" に従って順序付けられているレコードセットがある場合は、"Customer_ID" に関する出力をグループ化することができます。
groupCaseSensitive	オプション	yes	ブール値です。行のグループ化で大文字と小文字を区別するかどうかを指定します。
maxRows	オプション	すべての行を表示	表示する行の最大数です。
query	オプション		出力セクションのデータを取り出す cfquery の名前です。
startRow	オプション	1	出力を開始する行です。

使用方法

cfoutput タグ本文内では、シャープ記号 (#) で囲まれたテキストは ColdFusion 変数または関数呼び出しとして扱われます。たとえば、次のコードでは "Hello World!" というテキストが表示されます。

```
<cfset myVar="Hello World!">
<cfoutput>#myVar#</cfoutput>
```

query 属性を指定すると、このタグはクエリー行をループし、startRow 値および maxRows 値で指定された範囲内の行ごとに出力を生成します。さらに、グループ化属性の値で指定されている場合は、重複エントリをグループ化または削除します。また、**query**.currentRow 変数を、処理中の現在の行に設定します。

クエリーを処理する cfoutput ブロックをネストする場合は、最上位レベルで query 属性と group 属性を指定します。一番内側の cfoutput ブロックを除いて、内側の各ブロックに group 属性を指定できます。

このタグには終了タグが必要です。

例

```
<!--- EXAMPLE: This example shows how cfoutput operates. --->
<!--- Run a sample query. --->
<cfquery name = "GetCourses" dataSource = "cfdocexamples">
    SELECT Dept_ID, CorName, CorLevel
    FROM courseList
    ORDER by Dept_ID, CorLevel, CorName
</cfquery>
<h3>cfoutput Example</h3>
<p>cfoutput tells ColdFusion Server to begin processing, and then to hand back control of page rendering to
the web server.
<p>For example, to show today's date, you could write #DateFormat("#Now()#"). If you enclosed that expression
in cfoutput, the result would be<cfoutput>#DateFormat(Now())#</cfoutput>.

<p>In addition, cfoutput may be used to show the results of a query operation, or only a partial result, as
shown:

<p>There are <cfoutput>#getCourses.recordCount#</cfoutput> total records in our query. Using the maxRows
parameter, we are limiting our display to 4 rows.
<p><cfoutput query = "GetCourses" maxRows = 4>
    #Dept_ID# #CorName# #CorLevel#<br>
</cfoutput>

<p>EXAMPLE: The next example uses the group attribute to eliminate duplicate lines from a
    list of course levels taught in each department.</p>
<p><cfquery name = "GetCourses" dataSource = "cfdocexamples"></p>
    SELECT Dept_ID, CorLevel
    FROM courseList
    ORDER by Dept_ID, CorLevel
</cfquery>
<p><cfoutput query = "GetCourses" group="CorLevel" GroupCaseSensitive="True">
    #Dept_ID# #CorLevel#<br></p>
</cfoutput>

<p>cfoutput can also show the results of a more complex expression,
such as getting the day of the week from today's date. We first
extract the integer representing the Day of the Week from
the server function Now() and then apply the result to
the DayofWeekAsString function:</p>

<br>Today is #DayofWeekAsString(DayofWeek(Now()))#
<br>Today is <cfoutput>#DayofWeekAsString(DayofWeek(Now()))#</cfoutput>

<p> EXAMPLE: This last example shows nested cfoutput tags:</p>
<cfquery datasource="cfdocexamples" name="empSalary">
```

```
    SELECT Emp_ID, firstname, lastname, e.dept_id, salary, d.dept_name
    FROM employee e, departmt d
    WHERE e.dept_id = d.dept_id
    ORDER BY d.dept_name
</cfquery>

<!--- Outer cfoutput. --->
<cfoutput query="empSalary" group="dept_id">
    <h2>#dept_name#</h2>
    <table width="95%" border="2" cellspacing="2" cellpadding="2" >
<tr>
    <th>Employee</th>
    <th>Salary</th>
</tr>
<cfset deptTotal = 0 >
<!--- Inner cfoutput. --->
<cfoutput>
    <tr>
<td>#empSalary.lastname#, #empSalary.firstname#</td>
<td align="right">#DollarFormat(empSalary.salary)#</td>
</tr>
    <cfset deptTotal = deptTotal + empSalary.salary>
</cfoutput>
    <tr>
<td align="right">Total</td>
    <td align="right">#DollarFormat(deptTotal)#</td>
</tr>
    <cfset deptTotal = 0>
    </table>
</cfoutput>
```

タグ p ～ q

cfparam

説明

パラメータ (変数) が存在するかどうかをテストし、そのデータを検証し、デフォルト値が割り当てられていない場合は必要に応じて割り当てます。

履歴

ColdFusion 10：maxLength 属性が追加されました。

ColdFusion MX 7:

- min、max、および pattern の各属性が追加されました。
- type 属性の値として、creditcard、email、eurodate、float、integer、range、regex、regular_expression、ssn、social_security_number、time、URL、USdate、XML、および zipcode が追加されました。

カテゴリ

変数操作タグ

シンタックス

```
<cfparam
    name = "parameter name"
    default = "value"
    max = "value"
    maxLength = "number"
    min = "value"
    pattern = "regular expression"
    type = "data_type">
```

関連項目

cfcookie、cfregistry、cfsavecontent、cfschedule、cfset、『ColdFusion アプリケーションの開発』の Validating data with the IsValid function and the cfparam tag

属性

属性	必須 / オプション	デフォルト	説明
name	必須		テストするパラメータ (変数) の名前です ("Client.Email " や "Cookie.BackgroundColor " など)。省略すると、パラメータが存在しない場合はエラーが発生します。
default	オプション		パラメータが存在しない場合に、パラメータに設定する値です。パラメータが存在する場合でも、default 属性に対して任意の式が評価されます。パラメータが存在する場合、式の結果は割り当てられません。ただし、その場合でも式が副次的な効果を持つときは、その効果は発生します。
max	オプション		最大有効値です。range の検証の場合にのみ使用します。
maxLength	オプション		email、url および string の最大文字長を指定するために使用します。

属性	必須 / オプション	デフォルト	説明
min	オプション		最小有効値です。range の検証の場合にのみ使用します。
pattern	オプション		パラメータ表記の基準となる JavaScript 正規表現です。regex または regular_expression の検証の場合にのみ使用します。
type	オプション	any	有効なデータの形式です。次のいずれかです。検証アルゴリズムの詳細については、『ColdFusion アプリケーションの開発』の Validating Data の Validating form data using hidden fields を参照してください。 • any: 任意の型の値です。 • array: 値の配列です。 • binary: バイナリ値です。 • boolean: ブール値です (yes、no、true、false、または数値)。 • creditcard: mod10 アルゴリズムに準拠する 13 ～ 16 桁の数値です。 • date または time: 日付時刻値です。 • email: 有効な電子メールアドレスです。 • eurodate: 日付時刻値です。いずれの日付の部分も dd/mm/yy の形式でなければなりません。区切り文字として /、-、または . を使用できます。 • float または numeric: 数値です。 • guid: Universally Unique Identifier (ユニバーサル固有識別子) です。形式は "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX" で、'X' は 16 進数です。 • integer: 整数です。 • query: クエリーオブジェクトです。 • range: 数値の範囲です。min 属性と max 属性で指定します。 • regex または regular_expression: 入力内容を pattern 属性と照合します。 • ssn または social_security_number: 米国の社会保障番号です。 • string: 文字列値または 1 つの文字です。 • struct: 構造体です。 • telephone: 米国の標準の電話番号です。 • URL: http、https、ftp、file、mailto、または news の URL です。 • UUID: ColdFusion Universally Unique Identifier (ユニバーサル固有識別子) です。形式は 'XXXXXXXX-XXXX-XXXX-XXXXXXXXXXXXXXXX' で、'X' は 16 進数です。819 ページの「CreateUUID」を参照してください。 • USdate: mm/dd/yy 形式の米国の日付です。日および月は 1 ～ 2 桁、年は 1 ～ 4 桁です。 • variableName: ColdFusion 変数のネーミング規則に従った形式の文字列です。 • xml: XML オブジェクトと XML 文字列です。 • zipcode: 米国の 5 桁または 9 桁形式の郵便番号です。

使用方法

このタグを使用すると、次のテストを実行できます。

- 必要な変数が存在するかどうかをテストするには、name 属性のみを指定してこのタグを使用します。変数が存在しない場合、ColdFusion ではページの処理が中止され、エラーが返されます。
- 必要な変数が存在しているかどうか、およびそれが指定した型であることをテストするには、name 属性および type 属性を指定してこのタグを使用します。変数が存在しないか、変数の値が指定した型でない場合は、エラーが返されます。
- 変数のデフォルト値を設定するには、name 属性および default 属性を指定してこのタグを使用します。変数が存在しない場合は、変数が作成され、default 属性の値に設定されます。変数が存在する場合は、処理が継続されます。値は変更されません。

type 属性で variableName を指定する場合、パラメータの値は ColdFusion 変数ネーミング規則に沿った文字列にする必要があります。つまり、文字、アンダースコア (_)、または Unicode 通貨記号で始まり、文字、数字、アンダースコア、ピリオド、Unicode 通貨記号のみを含む文字列を使用する必要があります。ColdFusion は、そのパラメータ値が既存の ColdFusion 変数に対応するかどうかをチェックしません。

パフォーマンスを向上させるには、CFC メソッドを含めて、ColdFusion 関数に cfparam タグを使用しないようにします。代わりに、**CFML** ページの本文に cfparam タグを配置します。

例

```
<!--- This example shows how to use CFPARAM to define default values for page variables. --->
<cfparam name = "storeTempVar" default = "my default value">
<cfparam name = "tempVar" default = "my default value">

<!--- Check if form.tempVar was passed. --->
<cfif IsDefined("form.tempVar") is "True">
    <!--- Check if form.tempVar is not blank. --->
    <cfif form.tempVar is not "">
        <!--- If not, set tempVar to value of form.tempVar --->
        <cfset tempVar = form.tempVar>
    </cfif>
</cfif>

<body>
<h3>cfparam Example</h3>
<p>cfparam is used to set default values so that a developer does not have to
check for the existence of a variable using a function like IsDefined.</p>

<p>The default value of our tempVar is "<cfoutput>#StoreTempVar# </cfoutput>"</p>

<!--- Check if tempVar is still the same as StoreTempVar and that tempVar is not blank. --->
<cfif tempVar is not #StoreTempVar#
    and tempVar is not "">
        <h3>The value of tempVar has changed: the new value is
        <cfoutput>#tempVar#</cfoutput></h3>
</cfif>

<p>
<form action = "cfparam.cfm" method = "post">
    Type in a new value for tempVar, and hit submit:<br>
    <input type = "Text" name = "tempVar">
    <input type = "Submit" name = "" value = "submit">
</form>
```

cfpdf

説明

既存の PDF ドキュメントを操作します。cfpdf タグを使用すると、次のようなタスクを実行できます。

- 複数の PDF ドキュメントを 1 つの PDF ドキュメントにマージする
- PDF ドキュメントからページを削除する
- PDF ドキュメントのページをマージし、新規の PDF ドキュメントを生成する
- PDF ドキュメントを線形化して Web 表示を高速化する
- Acrobat® で作成されたフォームからインタラクティブ機能を除去して、フラットな PDF ドキュメントを生成する
- PDF ドキュメントを暗号化して、パスワードで保護する
- PDF ドキュメントまたは PDF ページからサムネールイメージを生成する
- PDF ドキュメントまたはページの透かしを追加または削除する
- PDF ドキュメントに関する情報 (ファイルの生成に使用されたソフトウェア、作成者など) を取得したり、PDF ドキュメントに関する情報 (タイトル、作成者、キーワードなど) を設定する
- PDF ポートフォリオを作成する
- PDF ドキュメントにヘッダまたはフッタを追加する、および PDF ドキュメントからヘッダまたはフッタを削除する
- PDF ドキュメントを最適化する

履歴

ColdFusion 8: このタグが追加されました。

ColdFusion 9: jpgdpi、maxBreadth、noAttachments、leftMargin、algo、noMetadata、noBookMarks、noJavaScripts、useStructure、noFonts、text、noComments、encodeAll、numberFormat、compressTIFFs、addQuads、rightMargin、topMargin、bottomMargin、noThumbnails、align、noLinks、maxLength、hires、hScale、overridepage、honourspaces、maxScale、package、vScale の各属性が新たに追加されました。

カテゴリ

データ出力タグ

シンタックス

```
Add a watermark to a PDF document
<cfpdf
    required
    action = "addwatermark"
    source = "absolute or relative pathname to a PDF file|PDF document variable|
        cfdocument variable"
    one of the following:
    copyfrom = "absolute or relative pathname to a PDF file from which the first page is
    used as a watermark"
    image = "absolute or relative pathname to image file|image variable used as a
    watermark"
    optional
    foreground = "yes|no"
    isBase64 = "yes|no"
    opacity = "watermark opacity"
    overwrite = "yes|no"
    pages = "page or pages to add the watermark"
    password = "user or owner password for the PDF source file"
    position = "position on the page where the watermark is placed"
    rotation = "degree of rotation of the watermark"
    showonprint = "yes|no">
    \\one of the following:
    destination = "PDF output file pathname"
    name = "PDF document variable name"
    image = "image file name to be used as the footer"
    text = "text to be used in the footer"
Add headers
<cfpdf
    required
    action = "addheader"
    source = "absolute or relative pathname to a PDF file|PDF document variable|
            cfdocument variable"
    pages = "page or pages to add the footer"
    optional
    isBase64 = "yes|no"
    overwrite = "yes|no"
    password = "user or owner password for the PDF source file"
    showonprint = "yes|no">
    align = "left|right|center"
    leftmargin = "value of the header left marign"
    rightmargin = "value of the header right margin"
    numberformat = "LOWERCASEROMAN|NUMERIC|UPPERCASEROMAN" <!---used with either
    _PAGENUMBER or _LASTPAGENUMBER--->
    opacity = "header opacity"
    topmargin = "value of the top margin of the header"
    \\one of the following:
    destination = "PDF output file pathname"
    name = "PDF document variable name"
    text = _PAGELABEL: add current page label|_LASTPAGELABEL: add last page label|
            _PAGENUMBER: add current page number|_LASTPAGENUMBER: add last page
            number \\text for the header. You can also add a normal text string.
    image = "image file name to be used as the header"

Add footer
<cfpdf
    required
    action = "addfooter"
    source = "absolute or relative pathname to a PDF file|PDF document variable|
            cfdocument variable"
    pages = "page or pages to add the footer"
    optional
```

```
    isBase64 = "yes|no"
    overwrite = "yes|no"

    password = "user or owner password for the PDF source file"
    showonprint = "yes|no">
    destination = "PDF output file pathname"
    name = "PDF document variable name"
    align = "left|right|center"
    one of the following:
    image = "image file name to be used as the footer"
    text =  _PAGELABEL: add current page label|_LASTPAGELABEL: add last page label|
            _PAGENUMBER: add current page number|_LASTPAGENUMBER: add last page
            number \\text for the header
    leftmargin = "value of the footer left marign"
    rightmargin = "value of the footer right margin"
    numberformat
    opacity = "footer opacity"
    bottommargin = "value of the bottom margin"
Delete pages from a PDF document
<cfpdf
    required
    action = "deletepages"
    pages = "page or pages to delete"
    source = "absolute or relative pathname to a PDF file|PDF document variable|
            cfdocument variable"
    optional
    overwrite = "yes|no"
    password = "PDF source file password"
        one of the following:
    destination = "PDF output file pathname"
    name = "PDF document variable name">
Delete headers and footers
<cfpdf
    required
    action = "removeheaderfooter"
    source = "absolute or relative pathname to a PDF file|PDF document variable|
        cfdocument variable"
    optional
    overwrite = "yes|no"
    pages = "page or pages to add the watermark"
    password = "user or owner password for the PDF source file"
    one of the following:
    destination = "PDF output file pathname"
    name = "PDF document variable name"
Retrieve information about a PDF document
<cfpdf
    required
    action = "getinfo"
    name = "structure variable name"
    source = "absolute or relative pathname to a PDF file|PDF document variable|
        cfdocument variable"
    optional
    password = "PDF source file password">

Merge PDF documents into an output PDF file
<cfpdf
    required
    action = "merge"
    one of the following:
    directory = "directory of PDF files to merge"
    source = "comma-separated list of PDF source files|absolute or relative pathname
        to a PDF file|PDF document variable|cfdocument variable"
```

```
    <cfpdfparam>
    \\required only when package is specified as true
    order = "name|time"
    one of the following if <cfpdfparam> is specified:
    name = "PDF document variable name"
    destination = "PDF output file pathname"
    optional
    package = "true|false" <!---create PDF packages if set to true. You can provide
        description in cfpdfparam tag, such as <cfpdfparam file="filename desc="">--->
    ascending = "yes|no"
    keepBookmark = "yes|no"
    overwrite = "yes|no"
    pages = "pages to merge in PDF source file"
    password = "PDF source file password"
    stopOnError = "yes|no"
    \\one of the following:
    destination = "PDF output file pathname"
    name = "PDF document variable name">

Use DDX instructions to manipulate PDF documents
<cfpdf
    required
    ddxfile = "DDX filepath|DDX string"
    inputfiles = "#inputStruct#"
    outputfiles = "#outputStruct#"
    name = "structure name">
    optional
    action="processddx"

Set passwords and encrypt PDF documnets
<cfpdf
    required
    action = "protect"
    source = "absolute or relative pathname to a PDF file|PDF document variable|
        cfdocument variable"
        at least one of the following:
    newUserPassword = "password"
    newOwnerPassword = "password"
        if newOwnerPassword is specified:
    permissions = "All|AllowAssembly|AllowDegradedPrinting|AllowCopy|AllowFillIn|AllowModifyAnnotations|
        AllowModifyContents|AllowPrinting|AllowScreenReaders|AllowSecure|None|
        comma-separated list"
    optional
    destination = "PDF output file pathname"
    encrypt = "RC4_40|RC4_128|RC4_128M|AES_128|none"
    overwrite = "yes|no"
    password = "source file password">

Name a PDF document variable
<cfpdf
    required
    action = "read"
    name = "PDF document variable name"
    source = "absolute or relative pathname to a PDF file|PDF document variable|
        cfdocument variable"
    optional
    password = "PDF source file password">

Remove a watermark from a PDF document
<cfpdf
    required
    action = "removeWatermark"
```

```
    source = "absolute or relative pathname to a PDF file|PDF document variable|
        cfdocument variable"
    optional
    overwrite = "yes|no"
    pages = "page or pages from which to remove the watermark"
    password = "PDF source file password">
        one of the following:
    destination = "PDF output file pathname"
    name = "PDF document variable name"

Set information about a PDF document
<cfpdf
    required
    action = "setinfo"
    info = "#structure variable name#"
    source = "absolute or relative pathname to a PDF file|PDF document variable|
        cfdocument variable"
    optional
    destination = "PDF output file pathname"
    overwrite = "yes|no"
    password = "PDF source file password">

Generate thumbnails from pages in a PDF document
<cfpdf
    required
    action = "thumbnail"
    source = "absolute or relative pathname to a PDF file|PDF document variable|
        cfdocument variable"
    optional
    destination = "directory path where the thumbnail images are written"
    format = "png|jpeg|tiff"
    imagePrefix = "string used as a prefix in the output filename"
    overwrite = "yes|no"
    password = "PDF source file password">
    pages = "page or pages to make into thumbnails"
    resolution= "low|high"
    scale = "percentage between 1 and 100"
    transparent = "yes|no">
    hires = "yes|no"
    overridepage = "yes|no"
    compresstiffs = "yes|no"
    maxscale = "maximum scale of the thumbnail"
    maxlength = "maximum length of the thumbnail"
    maxbreadth = "maximum width of the thumbnail"

Write a PDF document to an output file
<cfpdf
    required
    action = "write"
    source = "absolute or relative pathname to a PDF file|PDF document variable|
        cfdocument variable"
    \\one of the following
    destination = "PDF output file pathname"
    name = #PDF variable# <!---new variable support added now--->
    optional
    flatten = "yes|no"
    overwrite = "yes|no"
    password = "PDF source file password"
    saveOption = "linear|incremental|full"
    version = "1.1|1.2|1.3|1.4|1.5|1.6">
    encodeall = "yes|no"
```

```
Reduce the quality of a PDF document
<cfpdf
    required
    action = "optimize"
    source = "absolute or relative path of the PDF file|PDF document variable|
            cfdocument variable"
    algo = "bilinear|bicubic|nearest_neighbour" <!---algorithm for image
            downsampling--->
    pages = "*" <!----page numbers associated with the objects in the PDF document--->
    optional
    vscale= "Vertical scale of the image to be modified. Valid values are vscale>0"
    hscacle="Horizontal scale of the image to be modified. Valid values are hscale<1"
    destination = "PDF output file pathname"
    name = "PDF document variable"
    noattachments = "Discard all attachments"
    nobookmarks = "Discard all bookmarks"
    nocomments = "Discard all comments"
    nofonts = "Discard all fonts"
    nojavascripts = "Discard all JavaScript actions"
    nolinks = "Discard external cross-references"
    nometadata = "Discard document information and metadata"
    nothumbnails = "Discard embedded page thumbnails"
    overwrite = "true" <!---Overwrite the specified object in the PDF document--->
    password = "" <!--- PDF document password--->

Extract text
<cfpdf
    required
    action="extracttext" <!---extract all the words in the PDF.--->
    source= "absolute or relative path of the PDF file|PDF document variable|
            cfdocument variable"
    pages = "*" <!----page numbers from where the text needs to be extracted from the
                PDF document--->
    optional
    addquads = "add the position or quadrants for the text in the PDF"
    honourspaces = "true|false"
    overwrite = "true" <!---Overwrite the specified object in the PDF document--->
    password = "" <!--- PDF document password--->
    type = "string|xml" <!---format in which the text needs to be extracted--->
    one of the following:
    destination = "PDF output file pathname"
    name = "PDF document variable"
    usestructure = "true|false"
Extract image
<cfpdf
    required
    action = "extractimage" <!---extract images and save it to a directory--->
    source = "absolute or relative path of the PDF file|PDF document variable|
            cfdocument variable"
    pages = "*" <!---page numbers from where the images need to be extracted--->
    optional
    overwrite = "true|false" <!---overwrite any existing image when set to true--->
    format = "png|tiff|jpg" <!---format in which the images should be extracted--->
    imageprefix = "*" <!---the string that you want to prefix with the image name--->
```

```
    password = "" <!--- PDF document password--->
    destination = "PDF output file pathname"

Page level transformations
<cfpdf
    required
    action = "transform"
    source = "absolute or relative path of the PDF file|PDF document variable|
            cfdocument variable"
    pages = "page or pages to be transformed"
    optional
    hscale = "value of the horizontal scale of the page"
    overwrite = "yes|no"
    password = "PDF source file password"
    position = "x, y" <!---value in pixels--->
    rotation = "0|90|180|270"
    vscale = "length of the page to be transformed"
    one of the following:
    destination = ""Path of the directory where the PDF document will be saved"
    name = "PDF document variable"
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfdocument、cfdocumentsection、cfpdfform、cfpdfformparam、cfpdfparam、cfpdfsubform、cfprint、IsDDX、IsPDFFile、IsPDFObject、『ColdFusion アプリケーションの開発』の Assembling PDF Documents

属性

属性	アクション	必須 / オプション	デフォルト	説明
action	N/A	オプション	*processddx*	実行するアクションです。 • addWatermark • deletePages • getInfo • merge • processddx • protect • read • removeWatermark • setInfo • thumbnail • write • optimize • extracttext • extractimage • archive • addheader • addfooter • removeheaderfooter • transform
addquads	extracttext	オプション	false	サムネールの位置および象限を追加します。
align	addheader addfooter	オプション	center	PDF のヘッダとフッタを揃えます。
algo	optimize	必須		イメージダウンサンプリングのアルゴリズムを指定します。値は bilinear、bicubic、および nearest_neighbour です。
ascending	merge	オプション	no	PDF ファイルのソート順です。 • yes: ファイルは昇順でソートされます。 • no: ファイルは降順でソートされます。 directory 属性を指定した場合にのみ適用されます。
bottomMargin	addfooter	オプション		bottomMargin の値を指定します。

属性	アクション	必須 / オプション	デフォルト	説明
copyFrom	addWatermark	オプション		透かしとして使用する PDF ドキュメントのパス名です (このドキュメントの最初のページが透かしとして使用されます)。
compresstiffs	thumbnail	オプション	no	TIFF 形式のサムネールを圧縮します。
ddxfile	processddx	必須		DDX ファイルのパス名または DDX 命令を含む文字列です。
destination	addWatermark deletePages merge protect removeWatermark setInfo thumbnail write optimize extracttext extractimage addheader addfooter removeheaderfooter transform	write アクションの場合は必須 その他のアクションの場合はオプション		変更後の PDF ドキュメントのパス名です。保存先のファイルが既に存在する場合は、overwrite 属性を yes に設定します。保存先のファイルが存在せず、指定した親ディレクトリが存在する場合は、ColdFusion によってそのファイルが作成されます。 destination 属性と name 属性の両方を指定することはできません。 thumbnail アクションの場合は、イメージを書き込むディレクトリパスが保存先になります。保存先ディレクトリの相対パス名を指定する場合は、template ディレクトリからの相対パスを使用します。保存先ディレクトリを指定しない場合は、template ディレクトリの下に thumbnails というサブディレクトリが作成されます。 optimize アクションの場合、保存先は、最適化対象の PDF ドキュメントが存在するパスになります。 extracttext および extractimage の場合、保存先は、テキストまたはイメージの抽出元となる PDF ドキュメントのパスになります。 addheader、addfooter、removeheaderfooter の場合、保存先は、ヘッダまたはフッタを追加するか、ヘッダおよびフッタを削除する PDF ドキュメントのパスになります。 transform の場合、保存先では、ページレベルの変換の実行が必要な PDF ドキュメントのディレクトリパスが指定されます。
directory	merge	オプション		マージする PDF ドキュメントのディレクトリです。directory 属性または source 属性のいずれかを指定します。directory 属性を指定した場合、デフォルトでは、ファイル名の降順でドキュメントの順序が決定されます。ファイルの順序を変更するには、order 属性を使用します。
encodeall	write	オプション	no	ページ内容を最適化するために、エンコードされていないストリームをエンコードします。

属性	アクション	必須 / オプション	デフォルト	説明
encrypt	protect	オプション	RC4_128 (Acrobat 5.0 以上)	PDF 出力ファイルの暗号化方式です。 • RC4_40 • RC4_128 • RC4_128M • AES_128 • None 詳細については、「PDF ドキュメントの暗号化」を参照してください。
flatten	write	オプション	no	Acrobat で作成されたフォームにのみ適用され、インタラクティブ機能を無効にするかどうかを指定します (LiveCycle で作成されたフォームには適用されません)。 • yes: フォームフィールドのインタラクティブ機能が無効になります。 • no: フォームフィールドのインタラクティブ機能が保持されます。
foreground	addWatermark	オプション	no	ページ上の透かしの配置です。 • yes: 前景 (ページコンテンツの上) に透かしが表示されます。 • no: 背景 (ページコンテンツの背後) に透かしが表示されます。
format	thumbnail	オプション	jpg	出力するサムネールイメージのファイルタイプです。 • jpg • tiff • png
hires	thumbnail	オプション	no	yes に設定されている場合、サムネールに対して高解像度を設定します。
honourspaces	extracttext	オプション	false	このオプションを "true" に設定すると、読みやすさと間隔調整の精度が向上します。
hscale	optimize	オプション		変更されるイメージの水平比率です。有効な値は hscale<1 です。
image	addWatermark	オプション		透かしとして使用するイメージです。パス名、イメージファイルを含む変数、または ColdFusion イメージ変数を指定できます。
imagePrefix	thumbnail	オプション	ソースがパス名の場合は、ファイル名が接頭辞として使用されます。それ以外の場合は thumbnail という接頭辞が使用されます。	生成された各イメージサムネールファイルに使用する接頭辞です。イメージのファイル名には、**imagePrefix**_page_**n.format** の形式が使用されます。 たとえば、imagePrefix 属性が myThumbnail に設定されたドキュメントの 1 ページ目から生成されたサムネールは myThumbnail_page_1.jpg という名前になります。

属性	アクション	必須 / オプション	デフォルト	説明
info	setInfo	必須		関連情報の構造体変数です (例 : "#infoStruct#")。PDF 出力ファイルの作成者、題名、タイトル、キーワードを指定できます。
inputFiles	processddx	必須		DDX ファイル内の入力変数または要素とパス名の文字列に PDF ソースファイルをマップする構造体です。
isBase64	addWatermark	オプション	no	image 属性が指定されている場合にのみ有効です。透かしとして使用するイメージが Base64 形式であるかどうかを指定します。 • yes: イメージは Base64 形式です。 • no: イメージは Base64 形式ではありません。
keepBookmark	merge	オプション	no	ソース PDF ドキュメントのブックマークをマージ後のドキュメントでも保持するかどうかを指定します。 • yes: ブックマークは保持されます。 • no: ブックマークは削除されます。
leftmargin	addheader	オプション		ヘッダの左マージンの値を指定します。
maxbreadth	thumbnail	オプション		サムネールの最大幅を指定します。
maxlength	thumbnail	オプション		サムネールの最大長を指定します。
maxscale	thumbnail	オプション		サムネールの最大スケールを指定します。
name	addWatermark deletePages getInfo merge processddx protect read removeWatemark write tranform addheader addfooter removeheaderfooter	必須 : getInfo processddx read オプション : addWatermark deletePages merge protect removeWatermark tranform addheader addfooter removeheaderfooter		PDF ドキュメント変数名です (例 : *myBook*)。 ソースが PDF ドキュメント変数の場合は、name 属性を再度指定することはできません。変更後の PDF ドキュメントを保存先に書き込むことができます。 destination 属性と name 属性の両方を指定することはできません。 processddx アクションの場合、名前は出力変数の成功または失敗を示す構造体を表します。
newOwnerPassword	protect	オプション (「説明」を参照)		PDF ドキュメントのアクセス許可を設定するためのパスワードです。 デフォルトのアクセス許可を変更するには、newOwnerPassword 属性を指定します。 詳細については、「PDF ドキュメントのパスワード」を参照してください。

属性	アクション	必須 / オプション	デフォルト	説明
newUserPassword	protect	オプション (「説明」を参照)		PDF ドキュメントを開くためのパスワードです。 newUserPassword 属性または newOwnerPassword 属性のいずれかを指定します。両方を指定する場合は、異なるパスワードを使用する必要があります。詳細については、「PDF ドキュメントのパスワード」を参照してください。
noattachments	thumbnail	オプション	no	PDF ドキュメントからすべての添付ファイルを削除します。
noattachments	optimize	オプション	no	すべてのファイル添付を削除します。
nobookmarks	optimize	オプション	no	PDF ドキュメントからブックマークを削除します。
nocomments	optimize	オプション	no	PDF ドキュメントからコメントを削除します。
nofonts	optimize	オプション	no	フォントのスタイル設定を削除します。
nojavascripts	optimize	オプション	no	ドキュメントレベルのすべての JavaScript アクションを削除します。
nolinks	optimize	オプション	no	外部相互参照を削除します。
nometadata	optimize	オプション	no	ドキュメント情報およびメタデータを削除します。
nothumbnails	optimize	オプション	no	埋め込みページサムネールを削除します。
numberformat	addfooter	オプション		フッタの PDF ページの自動番号形式を指定します。
opacity	addWatermark addheader addfooter	オプション	3	透かしの不透明度です。有効な値は 0 (透明) ～ 10 (不透明) の整数です。
order	merge	オプション	time	ディレクトリ内の PDF ドキュメントをマージする順序です。 • name: ドキュメントはファイル名のアルファベット順でマージされます。 • time: ドキュメントはタイムスタンプ順でマージされます。 デフォルトでは、降順で (たとえば Z ～ A の順で) ファイルがマージされます。これを変更するには、ascending 属性を yes に設定します。
outputFiles	processddx	必須		キーとして DDX ファイルまたは文字列で指定された出力ファイルを含み、値として結果ファイルのパス名を含む構造体です。

属性	アクション	必須 / オプション	デフォルト	説明
overwrite	addWatermark deletePages merge protect removeWatermark setInfo thumbnail write tranform addheader addfooter removeheaderfooter	オプション	no	保存先のファイルが既に存在する場合、PDF 出力で上書きするかどうかを指定します。 • yes: 書き込み先のファイルを上書きします。 • no: 書き込み先のファイルを上書きしません。 thumbnail アクションの場合は、保存先ディレクトリを上書きするかどうかを指定します。overwrite を yes に設定せず、保存先のディレクトリが既に存在する場合、サムネールは生成されません。
package	merge	オプション	true	PDF パッケージを作成します。
pages	addWatermark deletePages merge removeWatermark optimize extracttext extractimage addheader addfooter removeheaderfooter transform	必須 : deletePages オプション : addWatermark merge removeWatermark thumbnail optimize extractext extractimage tranform addheader addfooter removeheaderfooter	all	アクションの対象となるソース PDF ドキュメントのページです。"1,6-9,56-89,100,110-120" のような形式で、複数のページまたはページ範囲を指定できます。 removeWatermark アクションの場合、pages 属性は透かしタイプにのみ適用されます。 重複するページ番号および総ページ数より大きい番号は無視されます。

属性	アクション	必須 / オプション	デフォルト	説明
password	addWatermark deletePages getInfo merge protect read removeWatermark setInfo thumbnail write optimize extracttext extractimage addheader addfooter removeheaderfooter transform	オプション		ソース PDF ドキュメントのオーナーパスワードまたはユーザーパスワードです (ドキュメントがパスワードで保護されている場合)。
permissions	protect	オプション	All	PDF ドキュメントのアクセス許可のタイプです。 • All • AllowAssembly • AllowCopy • AllowDegradedPrinting • AllowFillIn • AllowModifyAnnotations • AllowModifyContents • AllowPrinting • AllowScreenReaders • AllowSecure • None All と None を除き、アクセス許可のカンマ区切りリストを指定できます。アクセス許可を設定するには、newOwnerPassword 属性も設定する必要があります。
position	addWatermark	オプション		透かしを配置するページ上の位置です。この位置は、透かしの左上隅を表します。x 座標と y 座標を指定します (例 : "50,30")。

属性	アクション	必須 / オプション	デフォルト	説明
resolution	thumbnail	オプション	high	サムネールイメージの生成に使用するイメージ品質です。 • high: 高解像度を使用します (メモリ使用量が多くなります)。 • low: 低解像度を使用します。
rotation	addWatermark transform	オプション		ページ上の透かしイメージを回転させる角度です (例 : "30")。
saveOption	write	オプション	full	PDF 出力の保存オプションです。 • full: 通常の保存 (デフォルト) • incremental: 署名付き PDF ドキュメントの変更を保存する場合に必須 • linear: 表示を高速化する場合
scale	thumbnail	オプション	25	ソースページを基準としたサムネールのサイズです。1 ～ 100 のパーセントで指定します。
showOnPrint	addWatermark	オプション	no	PDF ドキュメントとともに透かしを印刷するかどうかを指定します。 • yes: PDF ドキュメントとともに透かしを印刷します。 • no: 透かしは表示専用になります。
source	addWatermark deletePages getInfo merge protect read removeWatermark setInfo thumbnail write optimize extracttext extractimage addheader addfooter removeheaderfooter transform	必須 (merge については、「使用方法」を参照してください)。		ソースとして使用する PDF ドキュメントです。指定できるソースは次のいずれかです。 • PDF ドキュメントの絶対パス名または相対パス名 (例 : c:¥work¥myPDF.pdf または myPDF.pdf)。 • cfdocument タグまたは cfpdf タグによってメモリ内に生成される PDF ドキュメント変数 (例 : "myPDFdoc")。

属性	アクション	必須 / オプション	デフォルト	説明
stopOnError	merge	オプション	no	directory 属性が指定されている場合にのみ有効です。指定したディレクトリに ColdFusion で読み込み可能な PDF ファイル以外のファイルが含まれている場合にマージ処理を停止するか続行するかを指定します。 • yes: 指定したディレクトリに無効な PDF ファイルが存在する場合、マージ処理を停止します。 • no: 指定したディレクトリに無効なファイルが存在する場合でも、マージ処理を続行します。
transparent	thumbnail	オプション	no	(format="png" のみ) イメージの背景を透明にするか不透明にするかを指定します。 • yes: 背景は透明です。 • no: 背景は不透明です。
useStructure	extracttext	オプション	true	PDF の構造に基づいて内容を抽出できます。この属性と同時に honourspaces 属性を使用すると、抽出されたテキストの読みやすさが向上します。
version	write	オプション		ドキュメントの書き込みに使用する PDF のバージョンです。 • 1.1 • 1.2 • 1.3 • 1.4 • 1.5 • 1.6 詳細については、「PDF バージョン」を参照してください。

注意：PDF ソースドキュメントに変更を加える場合は、source 属性と destination 属性で同じファイルパス名を指定し、overwrite 属性を yes に設定します。

使用方法

cfpdf タグを使用すると、既存の PDF ドキュメントを操作したり、マージしたりすることができます。cfpdf タグは Acrobat で使用可能な多くの機能を提供しますが、このタグを使用して別のファイル形式から PDF ドキュメントを生成することはできません。HTML および CFML コンテンツから PDF 出力を作成するには、cfdocument タグを使用します。

cfpdf タグを cfdocument タグ内に埋め込んだり、cfdocument タグを cfpdf タグ内に埋め込んだりすることはできません。ただし、cfdocument タグの出力を変数に書き込み、その変数を cfpdf タグに渡すことは可能です。次の例では、cfdocument タグを使用して表紙を作成し、それを別の PDF ドキュメントとマージする方法を示します。

```
<!--- Use the cfdocument tag to create a cover page and write the output to a variable called
    cfdoc. --->
<cfdocument format="PDF" name="cfdoc">
<html>
<body>
<h1>Here is a cover page</h1>
</body>
</html>
</cfdocument>

<!--- Use the cfpdf tag and cfpdfparam tags to merge individual PDF documents into a new PDF document called
new.pdf. Notice that the cfdoc variable created by using the cfdocument tag is the source value of the first
cfpdfparam tag. --->
<cfpdf action="merge" destination="/samtemp/pdfs/new.pdf" overwrite="yes">
    <cfpdfparam source="cfdoc">
    <cfpdfparam source="/samtemp/pdfs/pdf2.pdf">
    <cfpdfparam source="/samtemp/pdfs/pdf1.pdf">
</cfpdf>
```

cfpdf タグを使用して、インタラクティブ機能を持つ PDF フォームファイルを 1 つの PDF ドキュメントにマージし、write アクションで flatten 属性を指定すると、Acrobat で作成されたフォームからインタラクティブ機能を除去できます。PDF フォームのデータを処理するには、cfpdfform および関連タグを使用します。cfpdf タグを使用して、Adobe LiveCycle Designer ES で作成されたフォームをフラット化することはできません。

PDF ファイルの読み込みと書き込み

cfpdf タグには、PDF ファイルの読み込みと書き込みに関するさまざまなオプションがあります。PDF 変数または PDF ファイルをソースとして指定し、変数またはファイルに出力を書き込むことができます (ただし、両方を行うことはできません)。次の表で、読み込みおよび書き込みの操作について説明します。

タスク	属性	例
ソース PDF ファイルを上書きする	source 属性で PDF ファイルのパス名を指定し、destination 属性を指定しない	<cfpdf action="addWatermark" source="myPDF.pdf" image="myImage.jpg">
メモリ内の PDF ドキュメントをファイルに書き込む	source 属性で PDF 変数を指定し、destination 属性で PDF ファイルのパス名を指定する	<cfpdf action="addWatermark" source="myPDF" image="myImage.jpg" destination="outputFile.pdf">
PDF ドキュメントを新規ファイルに書き込む	source 属性で PDF ファイルのパス名を指定し、destination 属性で別の PDF ファイルのパス名を指定する	<cfpdf action="addWatermark" source="sourceFile.pdf" image="myImage.jpg" destination="outputFile.pdf">
PDF ファイルを PDF 変数に書き込む	source 属性で PDF ファイルのパス名を指定し、name 属性で PDF 変数を指定する	<cfpdf action="addWatermark" source="sourceFile.pdf" image="myImage.jpg" name="myPDF">
メモリ内の PDF ドキュメントを上書きする	source 属性で PDF 変数名を指定し、destination 属性を指定しない	<cfpdf action="addWatermark" source="myPDF" image="myImage.jpg">

メモリ内の PDF ファイルの操作

ColdFusion では、name 属性を使用して PDF ファイルを変数に書き込むことができます。この方法は、ドキュメントに対して複数のオペレーションを実行してからファイルに書き込む場合に便利です。ただし、この方法は多くのメモリを消費するため、小さなファイルを操作する場合にのみ使用してください。大きな PDF ドキュメントを操作する場合は、PDF ドキュメントをファイルに書き込むことを推奨します。

ソースとして変数を指定する場合は、name 属性を指定しないことを推奨します。name 属性を指定するとコピーが作成されるため、処理速度が低下します。ほとんどの場合、ファイルに書き込んだ後でも変数を再利用できるので、この属性を指定する必要はありません。

注意：try/catch ブロック内で PDF 変数を使用してエラーが発生した場合は、エラーの発生後にその変数を使用できなくなります。

PDF ドキュメントの印刷

cfprint タグを使用すると、PDF ドキュメントを印刷できます。ただし、注釈、コメント、編集者による校正などのマークアップは印刷されません。

addWatermark アクション PDF ドキュメント内の指定したページに透かしを追加するには、addwatermark アクションを使用します。次のいずれかの方法で透かしを追加できます。

- 別の PDF ドキュメントの最初のページを透かしとして使用します。イメージを拡大せずに、copyfrom で指定したページがソースドキュメントの上に重ねられます。
- 透かしとして使用するイメージファイルを指定します。
- イメージ変数を使用してメモリ内のイメージを指定します。

次のコードは、PDF ドキュメントの最初のページを透かしとして使用する方法を示しています。

```
<cfpdf action="addWatermark" source="c:\myBook.pdf" copyFrom="e:\yourBook.pdf"
    destination="ourBook.pdf" overwrite="yes">
```

デフォルトでは、出力ファイルのすべてのページに透かしが適用され、透かしイメージがページの中央に配置されます。次のコードは、JPEG イメージを透かしとして使用し、出力ファイルの最初のページに透かしを適用します。

```
<cfpdf action="addWatermark" source="Book.pdf"
    image="../cfdocs/images/artgallery/paul01.jpg" destination="newBook.pdf" pages="1"
    overwrite="yes">
```

ColdFusion イメージを透かしとして指定するには、cfimage タグまたは Image 関数を使用します。addwatermark アクションでは、RGB および ARGB イメージもサポートされます。特に、cfimage タグおよび関連する関数を使用して追加されたイメージもサポートされます。次の例では、イメージをグレースケールに変換し、それを透かしとして PDF ファイルに適用します。

```
<!--- Use the ImageNew function to create a ColdFusion image from a JPEG file. --->
<cfset myImage=ImageNew("../cfdocs/images/artgallery/jeff05.jpg")>

<!--- Use the ImageGrayscale function to convert the image to grayscale in memory. --->
<cfset ImageGrayscale(myImage)>

<!--- Specify the image variable to apply the grayscale image as a watermark in the Book.pdf file. Because
the source and destination are the same and the overwrite attribute is set to yes, ColdFusion overwrites
the source file. --->
<cfpdf action="addWatermark" source="Book.pdf" destination="Book.pdf" overwrite="yes" image="#myImage#">
```

ColdFusion イメージの詳細については、『ColdFusion アプリケーションの開発』の Creating and Manipulating ColdFusion Images を参照してください。

addfooter このアクションは、PDF ドキュメントにフッタを追加する場合に使用します。次のコードのように、PDF ドキュメントが存在するソースと、フッタのある新規 PDF ドキュメントの保存先を指定します。

```
<cfpdf action = "addfooter"
                source = "../myBook.pdf"
                destination = "../myBookwithfooter.pdf"
                image = "adobelogo.JPG" // Use this attribute to add an image in the
                                        footer
                align = "right"> // By default, the alignemnt is center
```

また、フッタに挿入するイメージやテキストに加え、align、bottommargin、leftmargin、numberformat、opacity などのさまざまな属性も指定できます。

addheader このアクションは、PDF ドキュメントにヘッダを追加する場合に使用します。次のコードのように、PDF ドキュメントのソースおよび保存先を指定し、ヘッダに挿入するテキストまたはイメージを指定します。

```
<cfpdf action = "addheader"
        source = "../myBook.pdf"
        destination = "../myBookwithheader.pdf"
        text = "Adobe"
        align = "left">
```

deletePages アクション 指定した PDF ドキュメントからページを削除するには、deletePages アクションを使用します。次のように、単一のページ、ページ範囲、またはカンマ区切りリストを指定できます。

```
<cfpdf action="deletePages" source="c:\myBook.pdf" pages="1,16-32,89,100-147"
    destination="myLittleBook.pdf">
```

extracttext extracttext アクションは、次のコードのように、PDF ドキュメント内の指定されたページ番号からのすべての単語を抽出する場合に使用します。

```
<cfpdf action = "extracttext" source = "../myBook.pdf" pages = "5-20, 29, 80" destination =
"../adobe/textdoc.txt"
```

extractimage extractimage アクションは、次のコードのように、PDF ドキュメント内の指定されたページ番号からすべてのイメージを抽出する場合に使用します。

```
<cfpdf action = "extractimage" source = "../myBook.pdf" pages = "1-200" destination = "..\mybookimages"
imageprefix = "mybook">
```

抽出されたイメージは、destination 属性で指定したディレクトリに保存されます。抽出対象のイメージの接頭辞 (imageprefix) を指定できます。それ以外の場合は、イメージ名に「cf+page number」のような接頭辞が付加されます。固有の形式でイメージを保存するには、format 属性を使用します。

getInfo アクション PDF ドキュメントに関する情報 (作成者、タイトル、作成日など) を取得するには、getInfo アクションを使用します。次のように、ファイルに関するデータを含む構造体変数の名前を指定します。

```
<cfpdf action="getInfo" source="myBook.pdf" name="PDFInfo">
<p><cfoutput>#PDFInfo.title#</cfoutput></p>
<p><cfoutput>#PDFInfo.author#</cfoutput></p>
<p><cfoutput>#PDFInfo.keywords#</cfoutput></p>
<p><cfoutput>#PDFInfo.created#</cfoutput></p>
```

情報要素の完全なリストを表示するには、次のように cfdump タグを使用します。

```
<cfdump var="#PDFInfo#">
```

注意： パスワードで保護されている PDF ドキュメントのアクセス許可を表示するには、(オーナーパスワードではなく) ユーザーパスワードを指定します。オーナーパスワードを指定した場合、アクセス許可はすべて Allowed に設定されます。

PDF ドキュメントの品質の低下

optimize アクションは、PDF ドキュメント内のイメージをダウンサンプルし、未使用のオブジェクトを破棄する場合に使用します。

optimize PDF ドキュメント内のイメージをダウンサンプルするために、algos 属性を、bilinear、bicubic、および nearest_neighbour の値とともに使用します。次のコードでは、イメージのダウンサンプリング後の PDF が生成されます。

```
<cfpdf action = "optimize" algo = "bicubic" source "..\myBook.pdf" name = #mybook#>
```

また、optimize アクションで次の属性を使用して、コメント、JavaScript、添付ファイル、ブックマーク、メタデータをなどの未使用のオブジェクトを PDF ドキュメントから破棄することもできます。

```
<cfpdf action = "optimize"
        noJavaScripts
        noThumbnails
        noBookmarks
        noComments
        noMetadata
        noFileAttachments
        noLinks
        nofonts>
```

PDF ドキュメント内のページの変換

1 つのページのサイズを拡大または縮小し、PDF ドキュメント内の複数のページの位置や回転の値を指定できます。

transform transform アクションには、ページのサイズ (hscale、vscale)、位置 (position)、および回転 (rotation) を定義する 4 つの属性があります。次のコードに使用法を示します。

```
<cfpdf action = "transform"
        required
        source = "..\myBook.pdf"
        optional
        destination = "..\new\myBook.pdf">
        hscale = ".5"
        vscale = ".15"
        position = "8, 10"
        rotation = "180">
```

回転の値は、0、90、180、270 の段階で指定する必要があります。その他の値を指定すると、エラーが発生します。

PDF ファイルの情報要素

次の表に、getinfo アクションで取得できる情報要素を示します。

要素	例	説明
Application	Acrobat PDFMaker 7.0.7 for Word	PDF ドキュメントの作成に使用されたアプリケーションです。この値は読み取り専用です。
Author	Harper Lee	PDF ドキュメントの作成者です。setInfo アクションでテキスト文字列を指定して変更できます。
CenterWindowOnScreen	[空の文字列]	PDF ドキュメントを最初に開いたときの表示設定です。この設定を変更するには、DDX 要素 InitialViewProfile とともに processddx アクションを使用します。
ChangingDocument	Not Allowed	PDF コンテンツの編集に関して割り当てられたアクセス許可です。この設定を変更するには、protect アクションとともに permissions 属性を使用します。
Commenting	Allowed	PDF ドキュメントへのコメントの追加に関して割り当てられたアクセス許可です。この設定を変更するには、protect アクションとともに permissions 属性を使用します。
ContentExtraction	Allowed	PDF ドキュメントからのコンテンツの抽出に関して割り当てられたアクセス許可です。この設定を変更するには、protect アクションとともに permissions 属性を使用します。
CopyContent	Allowed	PDF ドキュメントからのコンテンツのコピーに関して割り当てられたアクセス許可です。この設定を変更するには、protect アクションとともに permissions 属性を使用します。
Created	D:20061121155226-05'00'	システムによって生成された、PDF ドキュメントの作成日です。setInfo アクションでテキスト文字列を指定して変更できます。
DocumentAssembly	Not Allowed	他の PDF ドキュメントとのマージに関して割り当てられたアクセス許可です。この設定を変更するには、protect アクションとともに permissions 属性を使用します。
Encryption	Password Security	PDF ファイルがパスワードで保護されているかどうかを示します。暗号化アルゴリズムを変更する場合、またはパスワードを追加する場合は、protect アクションを使用します。
FilePath	C:¥ColdFusion¥wwwroot¥lion¥myDoc.pdf	PDF ファイルの絶対パス名です。この値は読み取り専用です。
FillingForm	Allowed	フォームフィールドへのデータ入力に関して割り当てられたアクセス許可です。この設定を変更するには、protect アクションとともに permissions 属性を使用します。
FitToWindow	[空の文字列]	PDF ドキュメントを最初に開いたときの表示設定です。この設定を変更するには、DDX 要素 InitialViewProfile とともに processddx アクションを使用します。
HideMenubar	[空の文字列]	PDF ドキュメントを最初に開いたときの表示設定です。この設定を変更するには、DDX 要素 InitialViewProfile とともに processddx アクションを使用します。
HideToolbar	[空の文字列]	PDF ドキュメントを最初に開いたときの表示設定です。この設定を変更するには、DDX 要素 InitialViewProfile とともに processddx アクションを使用します。
HideWindowUI	[空の文字列]	PDF ドキュメントを最初に開いたときの表示設定です。この設定を変更するには、DDX 要素 InitialViewProfile とともに processddx アクションを使用します。
Keywords	marketing,sales,production	PDF ドキュメントの検索用キーワードです。setInfo アクションでキーワードのカンマ区切りリストを指定して変更できます。
Language	EN-US	PDF ドキュメントのソースファイルの作成に使用された言語バージョンです。この値は読み取り専用です。

要素	例	説明
Modified	D:20061121155226-06'00'	システムによって生成された、PDF ファイルの最終変更日のタイムスタンプです。setInfo アクションでテキスト文字列を指定して変更できます。
PageLayout	OneColumn	PDF ドキュメントを最初に開いたときの表示設定です。この設定を変更するには、DDX 要素 InitialViewProfile とともに processddx アクションを使用します。
Printing	Allowed	ドキュメントの印刷に関して割り当てられたアクセス許可です。この設定を変更するには、protect アクションとともに permissions 属性を使用します。
Producer	Acrobat Distiller 7.0.5 (Windows)	PDF ドキュメントの生成に使用された Acrobat Distiller のバージョンです。この値は読み取り専用です。
Properties	[空の文字列]	この値は読み取り専用です。
Secure	Not Allowed	PDF ドキュメントがパスワードで保護されているかどうかを示す表示設定です。
ShowDocumentsOption	[空の文字列]	PDF ドキュメントを最初に開いたときの表示設定です。この設定を変更するには、DDX 要素 InitialViewProfile とともに processddx アクションを使用します。
ShowWindowsOption	[空の文字列]	PDF ドキュメントを最初に開いたときの表示設定です。この設定を変更するには、DDX 要素 InitialViewProfile とともに processddx アクションを使用します。
Signing	Allowed	PDF ドキュメントへの電子署名の追加に関して割り当てられたアクセス許可です。この設定を変更するには、protect アクションとともに permissions 属性を使用します。
Subject	製品マーケティング	PDF ドキュメントに割り当てられた題名です。setInfo アクションでテキスト文字列を指定して変更できます。
Title	第 1 章 : はじめに	PDF ドキュメントに割り当てられたタイトルです。setInfo アクションでテキスト文字列を指定して変更できます。
TotalPages	25	PDF ドキュメントの総ページ数です。この値は読み取り専用です。
Trapped	[空の文字列]	PDF ドキュメントにトラッピングを適用するかどうかを示します。トラッピングとは、隣合う 2 つインク色の隙間をなくすための印刷技術です。setInfo アクションでテキスト文字列を指定して変更できます。
Version	1.6	PDF ドキュメントの作成に使用された Adobe PDF Generator のバージョンです。この設定を変更するには、write アクションとともに version 属性を使用します。詳細については、「PDF バージョン」を参照してください。

merge アクション 複数の PDF ドキュメント、または PDF ソースファイルのページを 1 つの出力ファイルにマージするには、merge アクションを使用します。次のコードは、指定したディレクトリに存在するすべての PDF ファイルをマージする方法を示しています。

```
<cfpdf action="merge" directory="c:\myPDFfiles" destination="oneBigFile.pdf"
    overwrite="yes">
```

デフォルトでは、タイムスタンプの降順でファイルがマージされます。次のコードは、ソースファイルをファイル名の降順でマージする方法を示しています。

```
<cfpdf action="merge" directory="c:\book" order="name" ascending="yes"
    destination="c:\book\output1.pdf" overwrite="yes">
```

この方法は、ソースファイルに Chap0.pdf、Chap1.pdf、Chap2.pdf のような名前が付いている場合に便利です。

デフォルトでは、指定したディレクトリ内で有効な PDF ドキュメント以外のファイルが検出された場合でも、マージ処理は継続されます。指定したディレクトリに有効な PDF ドキュメント以外のファイルが含まれている場合にマージを停止するには、次のように stopOnError 属性を yes に設定します。

```
<cfpdf action="merge" directory="c:\bookfiles" destination="book.pdf" overwrite="yes"
    order="name" ascending="yes" keepBookmark="yes" stopOnError="yes">
```

ドキュメント内の特定のページから PDF ファイルを作成するには、pages 属性とともに source 属性を使用します。次のコードは、ソースドキュメントの 1 ～ 5 ページ目からファイルを作成する方法を示しています。

```
<cfpdf action="merge" source="myBigBook.pdf" pages="1-5" destination="myShortBook.pdf"
    overwrite="yes">
```

複数のファイルを 1 つのドキュメントにマージするには、次のようにファイルの絶対パス名をカンマ区切りリストで指定します。

```
<cfpdf action="merge" source="c:\PDFdocs\myBook\Chap1.pdf,
    c:\PDFdocs\myBook\Chap2.pdf,c:\PDFdocs\myBook\Chap3,pdf" destination="myBook.pdf"
    package = "true" overwrite="yes">
```

これで、package = "true" 属性を merge アクションとともに使用して PDF パッケージを作成できるようになります。

ファイルのマージ順をさらに詳細に指定したり、異なる場所にあるファイルをマージしたり、複数の PDF ファイルからページを抽出したりするには、merge アクションとともに cfpdfparam タグを使用します。PDF ファイルのマージの詳細については、『ColdFusion アプリケーションの開発』の Assembling PDF Documents を参照してください。

cfpdf action="merge" および package="yes" を指定した場合は、すべてのファイル形式をソースとして使用できます。次のコード例では、ソースとして ZIP および JPEG ファイル形式を使用しています。

```
<cfpdf action="merge" package="yes" destination="./myBook/adobetest.pdf" overwrite="yes">
 <cfpdfparam source="./inputFiles/c.zip" >
<cfpdfparam source="./inputFiles/d.jpg" >
</cfpdf>
```

processddx アクション DDX (Document Description XML) 命令を使用して複数の PDF ファイルをマージするには、proccessddx アクションを使用します。DDX とは、Adobe® LiveCycle® Assembler で使用される宣言型マークアップ言語です。DDX 命令を使用すると高度なタスクを実行できます。たとえば、目次ページ、ヘッダ、フッタ、自動ページ番号、テキスト文字列の透かしなどを PDF ドキュメントに追加できます。

ColdFusion には、一部の LiveCycle Assembler 機能が組み込まれています。以下のセクションにある表を参照して、ColdFusion で実行できるタスクを確認し、LiveCycle Assembler を購入する必要があるかどうかを検討してください。

DDX の完全なシンタックスについては、『Adobe LiveCycle Assembler Document Description XML Reference』を参照してください。

サポートされる DDX 要素

次の表に、ColdFusion でサポートされる DDX 要素を示します。

About	Author	Background
Center	DatePattern	DDX
DocumentInformation	DocumentText	Footer
Header	InitialViewProfile	Keyword
Keywords	Left	MasterPassword
Metadata	NoBookmarks	OpenPassword
PageLabel	Password	PasswordAccessProfile

PasswordEncryptionProfile	PDF (「メモ」を参照)	PDFGroup
Permissions	Right	StyledText
StyleProfile	Subject	TableOfContents
TableOfContentsEntryPattern	TableOfContentsPagePattern	Title
Watermark		

注意：ColdFusion では、PDF 要素の certification および mergeLayers 属性はサポートされていません。

サポートされない DDX 要素

次の表に、ColdFusion でサポートされない DDX 要素を示します。

ArtBox	AttachmentAppearance	Bookmarks
BlankPage	BleedBox	Comments
Description	FileAttachments	FilenameEncoding
LinkAlias	Links	NoBackgrounds
NoComments	NoFileAttchments	NoFooters
NoForms	NoHeaders	NoLinks
NoPageLabels	NoThumbnails	NoWatermarks
NoXFA	PageMargins	PageSize
PageRotation	PageOverlay	PageUnderlay
PDFsFromBookmarks	Transform	TrimBox

簡単な DDX 命令

DDX 命令は任意のテキストエディタで作成でき、DDX の拡張子を付けてファイルに保存できます。次のコードは、複数のドキュメントをマージし、ソース PDF ドキュメントのブックマークを使用して目次を生成する DDX 命令の例です。

```
<?xml version="1.0" encoding="UTF-8"?>
<DDX xmlns="http://ns.adobe.com/DDX/1.0/"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://ns.adobe.com/DDX/1.0/ coldfusion_ddx.xsd">
<PDF result="Out1">
    <PDF source="Title"/>
    <TableOfContents/>
    <PDF source="Doc1"/>
    <PDF source="Doc2"/>
    <PDF source="Doc3"/>
</PDF>
</DDX>
```

ColdFusion での DDX 命令の処理

以下のコードを使用すると、ColdFusion で DDX 命令が処理されます。

```
<!--- The following code verifies that the DDX file exists and the DDX instructions are valid. --->
<cfif IsDDX("Book.ddx")>

    <!--- The following code maps the PDF source files to the PDF source variables in the
    DDX file. --->
    <cfset inputStruct=StructNew()>
    <cfset inputStruct.Title="Title.pdf">
    <cfset inputStruct.Doc1="Chap1.pdf">
    <cfset inputStruct.Doc2="Chap2.pdf">
    <cfset inputStruct.Doc3="Chap3.pdf">

    <!--- The following code maps the PDF output file to the PDF result variable in the DDX
    file. --->
    <cfset outputStruct=StructNew()>
    <cfset outputStruct.Out1="output.pdf">

    <!--- The following code process the DDX instructions in the Book.ddx file to generate
    a merged document. --->
    <cfpdf action="processddx" ddxfile="Book.ddx" inputfiles="#inputStruct#"
        outputfiles="#outputStruct#" name="ddxVar">
<cfelse>
    <p>The DDX instructions are not valid.</p>
</cfif>

<!--- The following code displays a success or failure message. --->
<cfoutput>#ddxVar.Out1#</cfoutput>
```

name 属性は、処理が成功したかどうかを確認するための変数を定義します。成功または失敗のメッセージを表示するには、上の例のように cfoutput タグを使用します。また、構造体を表示するには、次の例のように cfdump タグを使用します。

```
<cfdump var="#ddxVar#">
```

このコードは、構造体で指定されている出力ファイルごとに次の情報を返します。

- "Successful": ファイルが正常にマージされた場合。
- "Reason for failure": ファイルが正常にマージされず、失敗の理由が判明している場合。
- "Failure": ファイルが正常にマージされず、失敗の理由が不明の場合。

DDX ファイルまたは一連の DDX 命令が有効であるかどうかを検証するには、IsDDX 関数を使用します。

詳細な例については、『ColdFusion アプリケーションの開発』の Assembling PDF Documents を参照してください。

protect アクション PDF 出力ファイルをパスワードで保護したり、アクセス許可を設定したり、PDF 出力ファイルを暗号化するには、protect アクションを使用します。

protect アクションを使用する場合は、newUserPassword または newOwnerPassword を設定します。(両方を設定する場合は、異なるパスワードを指定する必要があります)。ドキュメントにユーザーパスワードを割り当てると、その PDF ドキュメントを開くときに、すべてのユーザーがこのパスワードを使用する必要があります。次のコードは、PDF ドキュメントにユーザーパスワードを追加します。

```
<cfpdf action="protect" source="Finances.pdf" destination="myFinances.pdf"
    newUserPassword="keepOut">
```

出力ファイルのアクセス許可を設定するには、newOwnerPassword を設定します。PDF ファイルにアクセスする際にオーナーパスワードを入力したユーザーは、そのファイルのオーナーと見なされます。次の例は、新しいオーナーパスワードを設定する方法を示しています。

```
<cfpdf action="protect" encrypt="AES_128"source="Book.pdf" destination="MysteryBook.pdf"
    overwrite="yes" newOwnerPassword="pssst" permissions="AllowDegradedPrinting">
```

この例では、アクセス許可が AllowDegradedPrinting に設定されているため、ユーザーは 150 DPI でドキュメントを印刷できますが、その他のアクションはすべて禁止されます。たとえば、ユーザーがファイルを削除しようとすると、入力された

パスワードが正しくないことを示すエラーメッセージ、またはそのアクションが許可されていないことを示すエラーメッセージが表示されます。

ColdFusion ではアクセス許可が保持されません。newUserPassword 属性を追加した場合は、アクセス許可も明示的に設定する必要があります。

myVar を使用するには、パスワードとして newownerpw を指定します。

PDF ドキュメントのパスワード

PDF ドキュメントには、ユーザーパスワードとオーナーパスワードという 2 種類のパスワードを設定できます。次の表で、2 種類の ColdFusion パスワードと、それらに相当する Acrobat パスワードについて説明します。

ColdFusion パスワード	相当する Acrobat パスワード	説明
ユーザーパスワード	ドキュメントを開くパスワード、ユーザーパスワード	その PDF ドキュメントを開くときに、すべてのユーザーが指定されたパスワードを入力する必要があります。ユーザーパスワードを入力した場合、PDF ドキュメント内で制限されている機能を変更することはできません。
オーナーパスワード	権限パスワード、マスタパスワード	このパスワードを入力したユーザーは、PDF ドキュメント内の機能へのアクセスを制限することができます。

PDF を保護すると、指定したパスワードに変更されます。ColdFusion の変数に保存されているパスワードは、指定されたパスワードに更新されます。ただし、両方のパスワードを指定した場合は、オーナーパスワードが使用されます。

PDF を保護するには、次のように設定します。

```
<cfpdf action="protect" source="myVar" password="oldpassword"
    permissions="none" newuserpassword="newuserpw"
    newownerpassword="newownerpw">
```

PDF のすべてのプロパティを取得するには、次のように設定します。

```
<cfpdf action="info" source="myVar" name="info">
```

ユーザーに許可されているプロパティのみを取得するには、次のように設定します。

```
<cfpdf action="info" source="myVar" password=" newuserpw" name="info">
```

PDF ドキュメントのアクセス許可

次の表に、オーナーが PDF ドキュメントに対して設定できるアクセス許可を示します。

アクセス許可	説明
All	この PDF ドキュメントの機能は制限されていません。
AllowAssembly	ユーザーはこの PDF ドキュメントを他のドキュメントにマージできます。
AllowCopy	ユーザーはこのファイルのコンテンツ (テキスト、イメージなど) をコピーできます。この設定は、thumbnail アクションを使用してサムネールイメージを生成する場合に必須です。
AllowDegradedPrinting	ユーザーはこのドキュメントを低解像度 (150 DPI) で印刷できます。
AllowFillIn	ユーザーはこの PDF フォームのフィールドにデータを入力できます。ユーザーは PDF フォームに電子署名を追加できます。
AllowModifyAnnotations	ユーザーはこの PDF ドキュメントのコメントを追加または変更できます。
AllowModifyContents	ユーザーはこのファイルのコンテンツを変更できます。ユーザーはこの PDF ドキュメントを他のドキュメントにマージできます。
AllowPrinting	ユーザーはこのドキュメントを高解像度で印刷できます。この設定は、cfprint タグを使用する場合に必須です。

アクセス許可	説明
AllowScreenReaders	ユーザーはこの PDF ドキュメントからコンテンツを抽出できます。
AllowSecure	ユーザーはこの PDF ドキュメントに電子署名を追加できます。
None	ユーザーはこのドキュメントの表示のみを行えます。

PDF ドキュメントの暗号化

encrypt 属性は、パスワードで保護されているドキュメントを開くための暗号化方式を設定します。デフォルトでは、PDF ファイルの暗号化には RC4 128 ビットの暗号化アルゴリズムが使用されます。暗号化アルゴリズムを変更するには、protect アクションとともに encrypt 属性を使用します。次のコードは、PDF 出力ファイルを AES アルゴリズムで暗号化します。

```
<cfpdf action="protect" encrypt="AES_128" source="Book.pdf" destination="MysteryBook.pdf"
    overwrite="yes" newOwnerPassword="pssst" permissions="AllowDegradedPrinting">
```

ColdFusion は、次の暗号化アルゴリズムをサポートします。

暗号化アルゴリズム	互換性	説明
AES_128	Adobe Acrobat 7.0 以降	Advanced Encryption Standard (AES) では、Rijndael アルゴリズムと呼ばれる、128 ビットのデータブロックを処理可能な対称ブロック暗号化方式が定義されています。これは最高レベルの暗号化方式です。 ユーザーはこの暗号化アルゴリズムを使用して次のことを行えます。 • すべてのドキュメントコンテンツを暗号化する • メタデータを除くすべてのドキュメントコンテンツを暗号化する • 添付ファイルのみを暗号化する
RC4_128M	Adobe Acrobat 6.0 以降	RC4 では、インターネットトラフィックを保護する Secure Sockets Layer (SSL) や、ワイヤレスネットワークを保護する WEP などのアルゴリズムに適用可能な RSA Security ソフトウェアストリーム暗号化方式が定義されています。 ユーザーはこの暗号化アルゴリズムを使用して次のことを行えます。 • すべてのドキュメントコンテンツを暗号化する • メタデータを除くすべてのドキュメントコンテンツを暗号化する
RC4_128	Adobe Acrobat 5.0 以降	RC4 128 ビットの暗号化方式です。ユーザーはこの暗号化アルゴリズムを使用してドキュメントのコンテンツを暗号化することはできますが、ドキュメントのメタデータを暗号化することはできません。
RC4_40	Adobe Acrobat 3.0 以降	RC4 40 ビットの暗号化方式です。これは最低レベルの暗号化方式です。
None		ドキュメントは暗号化されません。

注意：ドキュメントのメタデータはインターネット検索で使用されます。検索エンジンは、メタデータが暗号化されている PDF ドキュメントを検索できません。新しいバージョンの Acrobat でサポートされている方式を使用して暗号化された PDF ドキュメントを以前のバージョンの Acrobat で開くことはできません。たとえば、AES 128 暗号化を指定した場合、Acrobat 6.0 以前でそのドキュメントを開くことはできません。

read アクション ソース PDF ドキュメントを name 変数に読み込むには、次のコードのように read アクションを使用します。

```
<cfif IsPDFFile("Book.pdf")>
    <cfpdf action="read" source="Book.pdf" name="myBook">
...
</cfif>
```

removeWatermark アクション PDF ドキュメントまたはドキュメント内の指定したページから透かしを削除するには、removewatermark アクションを使用します。次の例では、PDF ドキュメントの最初のページから透かしを削除し、出力を新規ファイルに書き込みます。

```
<cfpdf action="removeWatermark" source="Book.pdf" pages="1" destination="newBook.pdf" overwrite="yes">
```

removeheaderfooter アクション PDF ドキュメント、またはドキュメント内の指定されたページからヘッダおよびフッタを削除するには、このアクションを使用します。次の例では、ドキュメント全体からヘッダおよびフッタを削除します。

```
<cfpdf action = "removeheaderfooter" source="..\mybook.pdf" destination = "new.pdf">
```

setInfo アクション PDF ドキュメントに関する情報を指定してドキュメントとともに保存するには、setinfo アクションを使用します。関連情報を含む構造体を作成します。cfpdf タグの info 属性で、その構造体を参照します。次のコードは、setInfo アクションを使用して変更可能な要素を示しています。

```
<cfset PDFinfo=StructNew()>
<cfset PDFinfo.Title="Make Way for Ducklings">
<cfset PDFinfo.Author="Donald Duck">
<cfset PDFinfo.Keywords="Huey,Dewy,Louie">
<cfset PDFinfo.Subject="Ducks">

<cfpdf action="setInfo" source="chap1.pdf" info="#PDFinfo#" destination="meta1.pdf" overwrite="yes">
```

thumbnail アクション ソース PDF ドキュメントからサムネールイメージを生成するには、thumbnail アクションを使用します。

サムネールファイルの保存先ディレクトリを指定しない場合は、CFM ページが存在するディレクトリ内にサムネール用のディレクトリが作成されます。ソースとしてファイル名を指定した場合、サムネールディレクトリの名前は、ソースファイルの名前に _Thumbnails を追加したものになります。たとえば、次のコードは、myBook.pdf の各ページからサムネールイメージを生成して、myBook_thumbnails というディレクトリに保存します。

```
<cfpdf action="thumbnail" source="myBook.pdf">
```

CFM ページが c:¥myProject¥genThumbnails.cfm ディレクトリに存在する場合、サムネールディレクトリのパス名は c:¥myProject¥myBook_thumbnails になります。

デフォルトでは、サムネールファイルは JPEG 形式で生成され、イメージのサイズはオリジナルの 25% に縮小されます。

ソースドキュメント内の個々のページを指定してサムネールを生成できます。また、サムネールのサイズ、解像度、出力形式 (JPEG、PNG、または TIFF)、サムネールファイル名に使用する接頭辞を変更することもできます。次のコードは、ソースドキュメントの最初のページからオリジナルサイズの 50% に縮小した低解像度のサムネールを生成します。

```
<cfpdf action="thumbnail" source="myBook.pdf" pages="1" destination="c:\myBook\images"
    imagePrefix="Cover" format="png" scale="50" resolution="low">
```

完全な出力ファイルのパス名は次のとおりです。

```
c:\myBook\images\Cover_page_1.png
```

注意：サムネールイメージを生成するには、ソースドキュメントのアクセス許可に AllowCopy が含まれている必要があります。詳細については、cfpdf の「PDF ドキュメントのアクセス許可」を参照してください。

ColdFusion 9 で、次の新しい属性が thumbnail アクションに導入されました。

- hires: この属性を true に設定すると、ページから高解像度のイメージを抽出できます。この属性は、ドキュメントに高解像度のイメージが含まれていて、それらのイメージの解像度を維持したい場合に便利です。

  たとえば、次のようになります。

  ```
  <cfpdf action="thumbnail" source="./WORK/myBook.pdf" destination="./WORK/Testing_CFPDF"
  overwrite="true" hires="yes">
  ```

- overridepage: この属性を true に設定すると、PDF のページサイズではなく、そのページ内に存在するイメージのサイズに準拠したサムネールが生成されます。イメージが存在しない場合、サイズはページの最大サイズに設定されます。

- compresstiffs: サムネールイメージのサイズを圧縮するには、この属性を使用します。属性の名前が示唆しているように、この属性は TIFF 形式に対してのみ有効です。次に例を示します。

```
<cfpdf action="thumbnail" source="C:\WORK\myBook.pdf" destination="C:\WORK\Testing_CFPDF"
overwrite="true" hires="yes" format="tiff" compresstiffs="yes">
```

- maxscale: サムネールイメージの最大スケールに整数値を指定するには、この属性を使用します。
- maxlength: サムネールイメージの最大長の整数値を指定するには、この属性を使用します。
- maxbreadth: サムネールイメージの最大幅の整数値を指定するには、この属性を使用します。

  次の例は、maxscale、maxlength、および maxbreadth の使用方法を示しています。

```
<cfpdf action="thumbnail" source="./WORK/myBook.pdf" destination="./WORK/Testing_CFPDF"
overwrite="true" format="jpg" maxscale="3" maxlength="300" maxbreadth="200" hires="yes" scale="100">
```

  注意：通常、maxscale 属性を使用する場合は、scale 属性の値を 100 に設定します。

write アクション ソース PDF ドキュメント、またはメモリに変数として格納されている PDF ドキュメントをファイルに書き込むには、write アクションを使用します。次のコードは、メモリに格納されている PDF ファイルを別の PDF バージョンに変換し、出力を新規ファイルに書き込みます。

```
<cfpdf action="read" source="Book.pdf" name="myBook">
<cfpdf action="write" source="myBook" destination="myBook1.pdf"
    version="1.4">
```

これで、write アクションで name 属性または destination 属性のいずれかを使用できるようになります。name 属性は、PDF ドキュメント変数として値を取ります。たとえば、上記のコードを次のように記述することができます。

```
<cfpdf action="read" source="Book.pdf" name="myBook">
<cfpdf action="write" source="myBook" name=#myBook#
    version="1.4">
```

新規の encodeall 属性では、ソース内のエンコードされていないストリームがすべてエンコードされます。ただし、LZW のようなダムエンコーディングと、flate のようなエンコーディングとが区別されないため、エンコードされていないストリームのみが flate エンコードされます。

注意：これで、フォント管理画面を使用してサムネールフォントを登録できるようになります。

PDF バージョン

以前のバージョンの Acrobat または Adobe Reader を使用しているユーザーがファイルを開けるようにするには、PDF バージョンを変更します。次の表に、PDF のバージョンと Acrobat および Adobe Reader のバージョンの互換性を示します。

PDF バージョン	互換性
1.1	Acrobat および Adobe Reader 2
1.2	Acrobat および Adobe Reader 3
1.3	Acrobat および Adobe Reader 4
1.4	Acrobat および Adobe Reader 5
1.5	Acrobat および Adobe Reader 6
1.6	Acrobat および Adobe Reader 7

Web での表示を高速化するために PDF ドキュメントを線形化するには、次のように saveOption 属性を linear に設定します。

```
<cfpdf action="write" source="myBook" destination="myBook1.pdf" saveOption="linear"
    overwrite="yes">
```

PDF フォームのインタラクティブ機能を維持する場合や、PDF ドキュメントに電子署名を追加できるようにする場合には、saveOption で linear を指定しないでください。電子署名を許可するには、次のように saveOption 属性を incremental に設定します。

```
<cfpdf action="write" source="myDraft" destination="mySignedDoc.pdf"
    saveOption="incremental" overwrite="yes">
```

Acrobat で作成されたフォームからインタラクティブ機能を除去するには、flatten 属性を使用します。

```
<cfpdf action="write" source="myAcrobatForm.pdf"
    destination="myFlatForm.pdf" flatten="yes" overwrite="yes">
```

注意：ColdFusion では、Adobe® LiveCycle® で作成されたフォームのフラット化はサポートされていません。LiveCycle および Acrobat で作成されたフォームの詳細については、『ColdFusion アプリケーションの開発』の Manipulating PDF Forms in ColdFusion を参照してください。

例

次の例では、PDF ドキュメント内のページからサムネールイメージを生成して、それらのイメージを PDF ドキュメント内のページにリンクします。

```
<h3>PDF Thumbnail Demo</h3>

<!--- Create a variable for the name of the PDF document. --->
<cfset mypdf="myBook">
<cfset thisPath=ExpandPath(".")>
<!--- Use the getInfo action to retrieve the total page count for the
    PDF document. --->
<cfpdf action="getInfo" source="#mypdf#.pdf" name="PDFInfo">
<cfset pageCount="#PDFInfo.TotalPages#">

<!--- Generate a thumbnail image for each page in the PDF source document,
    create a directory (if it doesn't already exist) in the web root that is
    a concatenation of the PDF source name and the word "thumbnails", and
    save the thumbnail images in that directory. --->
<cfpdf action="thumbnail" source="#mypdf#.pdf" overwrite="yes"
    destination="#mypdf#_thumbnails" scale=60>

<!--- Loop through the images in the thumbnail directory and generate a link
    from each image to the corresponding page in the PDF document. --->
<cfloop index="LoopCount" from ="1" to="#pageCount#" step="1">
    <cfoutput>
        <!--- Click the thumbnail image to navigate to the page in the PDF
                document. --->
        <a href="#mypdf#.pdf##page=#LoopCount#" target="_blank">
        <img src="#mypdf#_thumbnails/#mypdf#_page_#LoopCount#.jpg"></a>
    </cfoutput>
</cfloop>
```

cfpdfform

説明

Adobe® Acrobat® および Adobe® LiveCycle® Designer で作成された既存のフォームを操作します。cfpdfform タグを使用すると、次のようなタスクを実行できます。

- Acrobat LiveCycle で作成されたインタラクティブフォームを PDF ドキュメント内に埋め込みます。cfdocument タグ内に PDF フォームを埋め込むには、cfpdfform タグを使用します。

- Acrobat または LiveCycle で作成された既存のフォームをレンダリングします。これには、データベースや XML データファイルからフィールドにデータを挿入したり、HTTP POST や PDF 送信からのフォームデータを処理する手順が含まれます。
- PDF フォームの値を抽出または挿入し、出力をファイルに保存したり、データソースの更新に使用します。

履歴

ColdFusion 8: このタグが追加されました。

カテゴリ

フォームタグ

シンタックス

```
populate
<cfpdfform
    required
    action = "populate"
    source = "PDF file pathname|byte array"
    optional
    XMLdata = "XML object|XML string|XML data filename|
        URL that returns XML data"
    destination = "output file pathname"
    overwrite = "yes|no"/
    fdf = "true|false> <!---New attribute that populates data in FDF format instead of
    XML with subforms and params--->
    fdfdata = "file name to be imported" <!--- New attribute populates data in FDF format
    from the AcroForm--->
read
<cfpdfform
    required
    action = "read"
    source = "pathname|byte array"
        at least one of the following:
    XMLdata = "variable name for XML data"
    result = "structure containing form field values"
    optional
    overwrite = "yes|no"/>
    fdfdata = "filename to be exported to"
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfdocument、cfdocumentsection、cfform、cfinput、cfpdf、cfpdfformparam、cfpdfparam、cfpdfsubform、cfprint、IsPDFFile、IsPDFObject、『ColdFusion アプリケーションの開発』の Manipulating PDF Forms in ColdFusion

属性

属性	アクション	必須 / オプション	デフォルト	説明
action	該当なし	必須		ソースに対して実行するアクションです。 • populate • read
destination	populate	オプション	ブラウザに出力	出力ファイルのパス名です。絶対パス名またはコンテキストルートからの相対パス名を指定できます。 ファイル拡張子は PDF または XDP である必要があります。ファイルの形式はファイル拡張子によって決定されます (XDP 形式は LiveCycle フォームにのみ適用されます)。 出力先を指定しない場合は、ブラウザにフォームが表示されます。 PDF ドキュメントにフォームを埋め込む場合は、出力先を指定しないでください。
overwrite	populate read	オプション	no	出力先ファイル (action="populate" の場合) またはデータファイル (action="read" の場合) を上書きするかどうかを指定します。 • yes • no
overwriteData	populate	オプション	no	PDF フォームフィールドの既存のデータをデータソースのデータで上書きするかどうかを指定します。 • yes: フォームフィールドの既存のデータをデータソースのデータで上書きします。 • no: フォームフィールドの既存のデータを保持し、データを含んでいないフィールドにのみデータを挿入します。 この属性は、XML データソース、cfpdfparam タグ、および cfpdfsubform タグから渡されるデータに適用されます。
result	read	オプション (「説明」を参照)		フォームフィールドの値を含む ColdFusion 構造体です。 XMLdata 属性または result 属性のいずれかを指定します (両方を指定することもできます)。
source	populate read	必須		ソース PDF のパス名 (ディスク上またはメモリ内の絶対パス、あるいはコンテキストルートからの相対パス)、または PDF を表すバイト配列です。
XMLdata	populate read	オプション (「説明」を参照)		XML データファイルのパス名です。 • action="populate" の場合は、このファイル、XML オブジェクト、または XML 文字列のデータがフォームフィールドに挿入されます。コンテキストルートからの相対パス名または絶対パス名を指定できます。 • action="read" の場合は、変数にデータが書き込まれます。 read アクションの場合は、XMLdata 属性または result 属性のいずれかを指定します (両方を指定することもできます)。
fdf	populate	オプション	false	true に設定した場合、XML ではなく subforms および params を含む FDF が作成されます。
fdfdata	populate read	オプション		populate の場合、FDF データのインポート元となるファイル名を指定します。 read の場合、FDF データのエクスポート先となるファイル名を指定します。

使用方法

ColdFusion でサポートされるインタラクティブフォームは、Adobe Acrobat 6.0 以前で作成されたフォームと、Adobe LiveCycle で作成されたフォームの 2 種類です。Adobe Acrobat Professional および Standard 7.0 では、PDF フォームの作成に Adobe® LiveCycle® Designer が使用されます。ColdFusion は、LiveCycle Designer 7.0 以降で作成されたフォームをサポートしています。

Acrobat で作成されたフォームの構造はフラットになります (つまり、一連のフィールドが同一レベルに配置されます)。LiveCycle Designer で作成されたフォームは階層構造になり、多くの場合はネストされたサブフォームを含みます。データをフォームフィールドにマップするには、cfpdfsubform タグを使用して、ColdFusion でフォームの構造を再作成します。例については、cfpdfsubform タグの「使用方法」、および『ColdFusion アプリケーションの開発』の Manipulating PDF Forms in ColdFusion を参照してください。

populate アクション 指定したデータファイルに含まれるデータを PDF フォームフィールドに挿入するには、populate アクションを使用します。出力を書き込むファイルを指定できます。また、データを挿入したフォームをブラウザに直接表示することもできます。インタラクティブな PDF フォームをブラウザに表示する場合は、出力先を指定しないようにします。

次の例は、XML データファイルを使用して PDF フォームにデータを挿入し、完成したフォームをブラウザに表示する方法を示しています。

```
<cfpdfform source="c:\payslipTemplate.pdf" action="populate" XMLdata="c:\formdata.xml"/>
```

次の例は、XML データファイルを使用して PDF フォームにデータを挿入し、完成したフォームを新規の PDF ファイルに書き込む方法を示しています。

```
<!--- Specify an XML file to populate a PDF form. --->
<cfpdfform source="c:\payslipTemplate.pdf"
    destination="c:\employeeid123.pdf" action="populate"
    XMLdata="c:\formdata.xml"/>
```

XML データを返す URL を指定することもできます。次の例では、"http://test1.com/xyz" から XML コンテンツが返されます。

```
<cfpdfform source= "#sourcefile#" action="populate" XMLdata=
    "http://test1.com/xyz" destination="#resultfile#" overwrite="true"/>
```

Acrobat で作成されたフォームの場合、出力は PDF ファイルにのみ書き込むことができます。LiveCycle で作成されたフォームの場合は、出力を XML Data Package (XDP) ファイルに書き込むこともできます。XDP ファイルは、PDF ファイルの XML 表現です。

注意：Acrobat または LiveCycle Designer で作成されたフォームフィールドに挿入される値の大文字と小文字は区別されます。たとえば、フォーム内のチェックボックスに "Yes" という値が必要な場合、"yes" という値はフィールドに挿入されません。

ファイルの形式はファイル拡張子によって決定されます。出力を XDP 形式で保存するには、出力先のファイル名に XDP 拡張子を使用します。

```
<!--- Specify a an XML file to populate a PDF form. --->
<cfpdfform source="c:\payslipTemplate.pdf"
    destination="c:\employeeid123.xdp" action="populate"
    XMLdata="c:\formdata.xml"/>
```

cfpdfform タグ内で cfpdfformparam タグを使用すると、PDF フォーム内の個々のフィールドにデータを挿入できます。

次の例は、Acrobat で作成された既存のフォーム (payslipTemplate.pdf) にデータを挿入し、employeeID フィールドと salary フィールドに値が入力された PDF フォーム (employeeid123.pdf) を作成する方法を示しています。

```
<!--- This example shows how to populate two fields in a form created in Acrobat. --->
<cfpdfform source="c:\payslipTemplate.pdf"
    destination="c:\employeeid123.pdf" action="populate">
        <cfpdfformparam name="employeeId" value="123">
        <cfpdfformparam name="salary" value="$85,000">
</cfpdfform>
```

ColdFusion では、フィールドのデータを提供するソース PDF フォームの構造を正確に再現する必要があります。ColdFusion で PDF フォームの構造を確認するには、cfpdfform タグの read アクションを使用した後、cfdump タグを使用して結果の構造体を表示します。構造体のレベルごとに cfpdfsubform タグを使用します。詳細については、『ColdFusion アプリケーションの開発』の Manipulating PDF Forms in ColdFusion を参照してください。

次の例は、LiveCycle で作成されたフォームにデータを挿入する方法を示しています。多くの場合、LiveCycle でテンプレートから作成されたフォームには、form1 というサブフォームが含まれています。ColdFusion でサブフォームを作成するには、cfpdfsubform タグを使用します。

```
<!--- This example shows how to populate two fields in a LiveCycle form.
    --->
<cfpdfform source="c:\payslipTemplate.pdf"
    destination="c:\employeeid123.pdf" action="populate">
    <cfpdfsubform name="form1">
        <cfpdfformparam name="employeeId" value="123">
        <cfpdfformparam name="salary" value="$85,000">
    </cfpdfsubform>
</cfpdfform>
```

これで、populate アクションを使用して、FDF 形式のファイルをインポートできるようになります。次の例に、使用方法を示します。

```
<cfpdfform source= "write_acrroform.pdf" action="populate" fdfdata="abc.fdf" destination="hello.pdf">
</cfpdfform>
```

populate アクションの fdf 属性が true に設定されている場合、次の例に示すように、XML ではなく、subforms および params を含む FDF 形式のデータを挿入できます。

```
<cfpdfform source="acroform2.pdf" destination="source_result17.pdf" action="populate" overwrite="true"
fdf="true">
<cfpdfsubform name="Text1">
        <cfpdfsubform name="0">
        <cfpdfformparam name="0" value="Test1.0.0">
        <cfpdfformparam name="1" value="Test1.0.1">
        <cfpdfformparam name="2" value="Test1.0.2">
</cfpdfsubform>
<cfpdfsubform name="1">
        <cfpdfformparam name="0" value="Test1.1.0">
        <cfpdfformparam name="1" value="Test1.1.1">
        <cfpdfformparam name="2" value="Test1.1.2">
    </cfpdfsubform>
</cfpdfsubform>

<cfpdfsubform name="Text2">
    <cfpdfformparam name="0" value="Test2.0">
        <cfpdfformparam name="1" value="Test2.1">
            <cfpdfformparam name="2" value="Test2.2">
            <cfpdfformparam name="3" value="Test2.3">
        </cfpdfsubform>
<cfpdfformparam name="Text3" value="Test3">
<cfpdfformparam name="Text4" value="Test4">
<cfpdfformparam name="checkbox1" value="Yes">
<cfpdfformparam name="listbox1" value="item4">
<cfpdfformparam name="radiobutton1" value="2">
</cfpdfform>
```

read アクション ソース PDF フォームからデータを読み込み、フォームフィールドとその値を含む結果の構造体を生成するには、read アクションを使用します。また、read アクションを使用して、PDF ソースファイルから XML データファイルを生成することもできます。

次の例は、PDF ファイルを読み込み、そのデータから結果の構造体を生成する方法を示しています。

```
<!--- Use the read action to retrieve the values from the saved PDF. --->
<cfpdfform source="c:\employeeid123.pdf" result="resultStruct" action="read"/>
```

cfdump タグを使用すると、結果の構造体を表示できます。

```
<cfdump var="#resultStruct#">
```

結果のフィールド (#resultStruct.employeeId#、#resultStruct.salary# など) は ColdFusion で使用できます。

次の例は、PDF ファイルを読み込み、そのデータを XML ファイルに書き込む方法を示しています。

```
<cfpdfform source="c:\employeeid123.pdf" result="c:\employeeid123.xml" overwrite="yes"
    action="read"/>
```

次の例は、XML データを含む変数に PDF ファイルを読み込む方法を示しています。

```
<cfpdfform source="c:\employeeid123.pdf" XMLdata="myXMLdata" action="read"/>
```

次の例は、PDF ファイルを XML データ変数に読み込み、結果の構造体を生成する方法を示しています。cffile タグを使用して XML ファイルにデータを書き込みます。

```
<cfset sourcefile = "Grant Application Updated.pdf">
<cfset resultfile = "Expandpath('datafile_result1.xml')">
<!--- Use the cfpdfform tag to read data extracted from a form into an XML data variable and
    generate a result structure. --->
<cfpdfform source= "#sourcefile#" action="read" xmldata="xmldata" result="resultstruct"/>
<!--- Use the cffile tag to write the XML data to a file. --->
<cffile action="write"file="#resultfile#" output="#xmldata#">
<!---- Use the cfdump tag to display the result structure. --->
<cfdump var="#resultstruct#">
```

PDF 送信からのデータの抽出

次のコードでは、PDF 送信からデータを抽出し、そのデータを fields という構造体に書き込みます。

```
<!--- The following code reads the submitted PDF file and generates a result structure called
    fields. --->
<cfpdfform source="#PDF.content#" action="read" result="fields"/>
```

データ構造体を表示するには、次のように cfdump タグを使用します。

```
<cfdump var="#fields#">
```

注意：PDF 送信からデータを抽出する場合は、常に "#PDF.content#" をソースとして指定します。

次のように、フォームフィールドを変数に設定することもできます。

```
<cfset empForm="#fields.form1#">
```

出力をファイルに書き込むには、cfpdfform タグの populate アクションを使用します。"#PDF.content#" をソースとして指定します。次の例では、PDF フォームのフィールドから一意のファイル名を生成しています。

```
<cfpdfform action="populate" source="#PDF.content#"
    destination="timesheets\#empForm.txtsheet#.pdf" overwrite="yes"/>
```

HTTP Post 送信からのデータの抽出

HTTP POST 送信では、PDF フォームからデータが送信されますが、フォーム自体は送信されません。PDF フォームフィールドのデータを抽出することはできますが、出力を直接ファイルに書き込むことはできません。データを抽出してデータベースを更新するには、データベースのフィールドを構造体および HTTP POST のデータに正確にマップする必要があります。

注意：送信後の HTTP POST データの構造は、送信前の PDF フォームの構造と異なります。両方の例については、『ColdFusion アプリケーションの開発』の Manipulating PDF Forms in ColdFusion を参照してください。

HTTP POST データの構造を確認するには、次のように、cfdump タグでフォーム名を変数として指定して、データ構造を表示します。

```
<cfdump var="#FORM.form1#">
```

注意：HTTP Post 送信からデータを抽出する場合は、常にソースとしてフォーム名を指定します。たとえば、LiveCycle Designer でテンプレートから生成されたフォームの場合は、"#FORM.form1#" を指定します。cfpdfform タグを使用してデータを抽出した結果として、複数のページが生成された場合は、1 つの構造体ではなく 1 ページにつき 1 つの構造体が返されます。

PDF ドキュメントへの PDF フォームの埋め込み

cfdocument タグ内で cfpdfform タグを使用すると、PDF ドキュメント内に既存のインタラクティブ PDF フォームを埋め込むことができます。cfpdfform タグは、最低 1 つの cfdocumentsection タグと併用しますが、cfdocumentsection タグの中に cfpdfform タグを含めないようにしてください。PDF フォームの埋め込みの詳細については、『ColdFusion アプリケーションの開発』の Manipulating PDF Forms in ColdFusion を参照してください。

Acrobat で作成したフォームのフラット化

Acrobat で作成されたフォームをフラット化するには、cfpdf タグを使用します。ColdFusion では、LiveCycle で作成されたフォームのフラット化はサポートされていません。詳細については、『ColdFusion アプリケーションの開発』の Assembling PDF Documents を参照してください。

フォームの印刷

Acrobat で作成されたフォームを印刷するには、cfprint タグを使用します。ただし、注釈、コメント、編集者による校正などのマークアップは印刷されません。cfprint タグを使用して、LiveCycle Designer で作成されたフォームを印刷することはできません。

PDF フォームの FDF でのエクスポート

read アクションを使用して、PDF フォームを FDF 形式でエクスポートできます。次の例に、FDF 形式での PDF フォームのエクスポート方法を示します。

```
<cfpdfform source= "acroform_export.pdf" action="read" fdfdata="abc.fdf" >
</cfpdfform>
```

例

次の例は、cfdocument タグを使用して作成された PDF ドキュメント内にインタラクティブな PDF フォームを埋め込む方法を示しています。

```
<!--- The following code extracts data from the cfdocexamples database based
    on a username entered in a login form. --->
<cfquery name="getEmpInfo" datasource="cfdocexamples">
    SELECT * FROM EMPLOYEES
    WHERE EMAIL = <cfqueryparam value="#form.username#">
</cfquery>

<!--- The following code creates a PDF document with headers
    and footers. --->
<cfdocument format="pdf">
    <cfdocumentitem type="header">
    <font size="-1" align="center"><i>Nondisclosure Agreement</i></font>
    </cfdocumentitem>
    <cfdocumentitem type="footer">
    <font size="-1"><i>Page <cfoutput>#cfdocument.currentpagenumber# of
        #cfdocument.totalpagecount#</cfoutput></i></font>
    </cfdocumentitem>

<!--- The following code creates the first section in the PDF document. --->
    <cfdocumentsection>
    <h3>Employee Nondisclosure Agreement</h3>
    <p>Please verify the information in the enclosed form. Make any of the
```

```
    necessary changes in the online form and click the <b>Print</b> button.
    Sign and date the last page. Staple the pages together and return the
    completed form to your manager.</p>
    </cfdocumentsection>

<!--- The following code embeds an interactive PDF form within the PDF
    document with fields populated by the database query. The cfpdpfform tag
    automatically creates a section in the PDF document. Do not embed the
    cfpdfform within cfdocumentsection tags. --->

    <cfpdfform action="populate" source="c:\forms\embed.pdf">
        <cfpdfsubform name="form1">
            <cfpdfformparam name="txtEmpName" value="#getEmpInfo.FIRSTNAME#
                #getEmpInfo.LASTNAME#">
            <cfpdfformparam name="txtDeptName" value="#getEmpInfo.DEPARTMENT#">
            <cfpdfformparam name="txtEmail" value="#getEmpInfo.IM_ID#">
            <cfpdfformparam name="txtPhoneNum" value="#getEmpInfo.PHONE#">
            <cfpdfformparam name="txtManagerName" value="Randy Nielsen">
        </cfpdfsubform>
    </cfpdfform>

<!--- The following code creates the last document section. Page numbering
    resumes in this section. --->
    <cfdocumentsection>
    <p>I, <cfoutput>#getEmpInfo.FIRSTNAME# #getEmpInfo.LASTNAME#</cfoutput>,
        hereby attest that the information in this document is accurate and complete.</p>
    <br/><br/>
    <table border="0" cellpadding="20">
    <tr><td width="300">
    <hr />
    <p><i>Signature</i></p></td>
    <td width="150"><hr />
    <p><i>Today's Date</i></p></td></tr>
    </cfdocumentsection>
</cfdocument>
```

cfpdfformparam

説明

cfpdfform タグに追加情報を提供します。

cfpdfformparam タグは、常に cfpdfform タグまたは cfpdfsubform タグの子タグになります。cfpdfformparam タグを使用すると、PDF フォーム内のフィールドにデータを挿入できます。

履歴

ColdFusion 8: このタグが追加されました。

カテゴリ

フォームタグ

シンタックス

```
<cfpdfform ...>
    <cfpdfformparam
    name = "field name"
    value = "ColdFusion variable"
    index = "integer">
</cfpdfform>
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfdocument、cfdocumentsection、cfform、cfinput、cfpdf、cfpdfform、cfpdfparam、cfpdfsubform、cfprint、IsPDFFile、IsPDFObject

属性

属性	必須 / オプション	デフォルト	説明
index	オプション	1	フィールド名に関連付けられたインデックスです。同じ名前のフィールドが複数ある場合は、インデックス値を使用して 1 つのフィールドを特定します。LiveCycle で作成されたフォームにのみ適用されます。
name	必須		PDF フォーム上のフィールド名です。
value	必須		フィールド名に関連付けられた値です。インタラクティブフィールドの場合は、ColdFusion 変数を指定します。

使用方法

cfpdfform タグまたは cfpdfsubform タグ内で cfpdfformparam タグを使用すると、PDF フォーム内のフィールドにデータを挿入できます。

名前が同じで値が異なるフィールドを指定するには、次のように cfpdfformparam タグの index 属性を使用します。

```
<!--- This example shows how to use multiple cfpdfformparam tags with the same name and
    different index values for a PDF form that contains fields with same name. --->
<cfpdfform source="c:\payslipTemplate.pdf"
    destination="c:\employeeid123.pdf" action="populate">
        <cfpdfformparam name="phone" value="781-869-1234" index="1"/>
        <cfpdfformparam name="phone" value="617-273-9021" index="2"/>
</cfpdfform>
```

注意：index 属性は、LiveCycle で作成されたフォームに対してのみ使用します。Acrobat で作成されたフォームには同じ名前のフィールドが複数存在する可能性はないため、index 属性は無効です。

例

cfpdfform タグの例を参照してください。

cfpdfparam

説明

cfpdf タグの追加情報を提供します。cfpdfparam タグは、cfpdf タグの merge アクションにのみ適用され、常に cfpdf タグの子タグになります。

履歴

ColdFusion 8: このタグが追加されました。

カテゴリ

フォームタグ

シンタックス

```
<cfpdf action = "merge" ..>
    <cfpdfparam
    pages = "page number|page range|comma-separated page numbers"
    password = "user or owner password"
    source = "absolute or relative pathname to a PDF file|PDF document variable|
        cfdocument variable">
</cfpdf>
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfdocument、cfdocumentsection、cfpdf、cfpdfform、cfpdfformparam、cfpdfsubform、cfprint、IsPDFFile、IsPDFObject

属性

属性	必須 / オプション	デフォルト	説明
pages	オプション		マージする PDF ソースファイルのページです。ページの範囲 (例 : "1-5") またはページのカンマ区切りリスト (例 : "1-5,9-10,18") を指定できます。
password	オプション		ユーザーパスワードまたはオーナーパスワードを指定します (ソース PDF ファイルがパスワードで保護されている場合)。
source	必須		マージするソース PDF ファイルです。PDF 変数、cfdocument 変数、またはファイルのパス名を指定できます。

使用方法

cfpdfparam タグを使用すると、複数の PDF ドキュメントを 1 つのファイルにマージできます。cfpdfparam タグを使用すると、ソースファイルの順序を明示的に指定できます。このタグを使用すると、異なる場所にある複数の PDF ドキュメントソースファイルのページをマージできます。

次のコードは、combined.pdf という PDF ドキュメントを作成します。この PDF ドキュメントには、abc.pdf ファイルの 1 ～ 3 ページおよび 5 ページ、xyz.pdf の全ページ、メモリ内の myPDFvariable 変数に格納されたファイル、abc.pdf ファイルの 10 ～ 90 ページ、という順序でページがマージされます。password 属性は、ソースファイルがパスワードで保護されている場合にのみ適用されます。

```
<cfpdf action="merge" destination="combined.pdf" overwrite="yes">
    <cfpdfparam source="c:\abc.pdf" pages="1-3,5" password="adobe">
    \\x
    <cfpdfparam source="myPDFvariable">
    <cfpdfparam source="abc.pdf" pages="10-90" password="adobe">
</cfpdf>
```

注意：cfpdfparam タグを cfpdf の merge アクションとともに使用する場合は、cfpdf タグの destination 属性または name 属性を指定する必要があります。

例

次の ColdFusion ページでは、税務関連のフォームと税務情報の冊子をダウンロードするためのフォームを作成します。

```
<h3>Downloading Federal Tax Documents</h3>
<p>Please choose the your type of business.</p>
<!--- Create the ColdFusion form to determine which PDF documents to merge. --->
<table>
<cfform action="cfpdfMergeAction.cfm" method="post">
    <tr><td><cfinput type="radio" name="businessType" Value="SoleP">
        Sole Proprieter</td></tr>
    <tr><td><cfinput type="radio" name="businessType"
        Value="Partner">Partnership</td></tr>
    <tr><td><cfinput type="radio" name="businessType" Value="SCorp">S Corporation</td></tr>
    <cfinput type = "hidden" name = "selection required" value = "must make a selection">
    <tr><td><cfinput type="Submit" name="OK" label="OK"></td></tr>
    </tr>
</cfform>
</table>
```

ColdFusion アクションページにより、フォームで選択された値に基づいて、異なる場所にある PDF ファイルがマージされます。

```
<!--- Create a merged PDF document based on the selection in the form. --->
<cfpdf action="merge" name="taxDoc">
    <cfif #form.businessType# is "SoleP">
        <cfpdfparam source="taxForms\f2106ez.pdf">
        <cfpdfparam source="taxForms\f1040.pdf">
        <cfpdfparam source="taxForms\f1040sc.pdf">
        <cfpdfparam source="taxInfo\i1040sc.pdf">
        <cfpdfparam source="taxInfo\i2106.pdf">
        <cfpdfparam source="taxInfo\i1040sc.pdf">
        <cfpdfparam source="taxInfo\p535.pdf">
        <cfpdfparam source="taxInfo\p560.pdf">
        <cfpdfparam source="taxInfo\p334.pdf">
    <cfelseif #form.businessType# is "Partner">
        <cfpdfparam source="taxForms\f1065.pdf">
        <cfpdfparam source="taxForms\f1065b.pdf">
        <cfpdfparam source="taxForms\f1065bsk.pdf">
        <cfpdfparam source="taxForms\f8804.pdf">
        <cfpdfparam source="taxForms\f8825.pdf">
        <cfpdfparam source="taxInfo\p535.pdf">
        <cfpdfparam source="taxInfo\p560.pdf">
        <cfpdfparam source="taxInfo\i1065bsk.pdf">
    <cfelseif #form.businessType# is "SCorp">
        <cfpdfparam source="taxForms\f1120s.pdf">
        <cfpdfparam source="taxForms\f2553.pdf">
        <cfpdfparam source="taxForms\f8453s.pdf">
        <cfpdfparam source="taxForms\f8825.pdf">
        <cfpdfparam source="taxInfo\i1120s.pdf">
        <cfpdfparam source="taxInfo\p542.pdf">
        <cfpdfparam source="taxInfo\p535.pdf">
        <cfpdfparam source="taxInfo\p560.pdf">
    </cfif>
</cfpdf>

<cfpdf action="write" source="taxDoc" destination="c:\taxDoc.PDF"
    overwrite="yes"/>
```

注意：cfpdf タグの merge アクションを使用すると、ColdFusion によってフォームフィールドが自動的にフラット化されます。

cfpdfsubform

説明

cfpdfform タグ内のサブフォームにデータを挿入します。

cfpdfsubform タグは cfpdfform タグの子タグにすることも、別の cfpdfsubform タグ内にネストすることもできます。

履歴

ColdFusion 8: このタグが追加されました。

カテゴリ

フォームタグ

シンタックス

```
<cfpdfform ..>
    <cfpdfsubform
    name = "field name"
    index = "integer">
    </cfpdfsubform>
</cfpdfform>
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfdocument、cfdocumentsection、cfform、cfinput、cfpdf、cfpdfform、cfpdfformparam、cfpdfparam、cfprint、IsPDFFile、IsPDFObject

属性

属性	必須 / オプション	デフォルト	説明
index	オプション	1	フィールド名に関連付けられたインデックスです。同じ名前のフィールドが複数ある場合は、このインデックス値を使用してフィールドが識別されます。
name	必須		PDF フォーム内のサブフォーム名に対応するサブフォームの名前です。

使用方法

cfpdfform タグとともに cfpdfsubform タグを使用すると、PDF フォーム内のサブフォームにデータを挿入できます。cfpdfsubform タグには複数の cfpdfformparam タグを含めることができます。また、次の例のように、サブフォームをネストすることもできます。

```
<!--- This example shows how to nest cfpdfsubform tags. --->
<cfpdfform source="c:\payslipTemplate.pdf"
            destination="c:\employeeid123.pdf" action="populate">
    <cfpdfsubform name="employeeDetail">
        <cfpdfsubform name="address">
            <cfpdfformparam name="txtAddLine1" value="572 Evergreen Terrace">
            <cfpdfformparam name="txtCity" value="Springfield">
            <cfpdfformparam name="txtState" value="Oregon">
            <cfpdfformparam name="txtZip" value="65412">
            <cfpdfformparam name="txtCountry" value="United States">
        </cfpdfsubform>
        <cfpdfformparam name="txtEmployeeId" value="879104">
        <cfpdfformparam name="numSalary" value="$85,000">
    </cfpdfsubform>
</cfpdfform>
```

サブフォームを使用する場合は、ソース PDF フォームの構造と完全に一致させる必要があります。そうしないと、ColdFusion がフォームにデータを挿入できないため、エラーが発生します。多くの場合、LiveCycle でテンプレートから生成されたフォームには、form1 というサブフォームが含まれています。次のように、これをサブフォームとして指定する必要があります。

```
<cfpdfform source="c:\forms\timesheetForm.pdf" action="populate">
    <cfpdfsubform name="form1">
        <cfpdfformparam name="txtCompanyName" value="Adobe">
        <cfpdfformparam name="txtManager" value="Randy Nielsen">
    </cfpdfsubform>
</cfpdfform>
```

ColdFusion で PDF フォームの構造を確認するには、次の例のように cfpdfform タグの read アクションを使用します。

```
<cfpdfform source="c:\forms\timesheetForm.pdf" result="resultStruct" action="read"/>
```

続いて、cfdump タグを使用してこの構造体を表示します。

```
<cfdump var="#resultStruct#">
```

例

cfpdfform タグの例を参照してください。

cfpod

説明

ポッドを作成します。ポッドとは、オプションのタイトルバーと表示要素を含む本体で構成されるブラウザウィンドウの領域またはレイアウト領域です。

カテゴリ

表示管理タグ

シンタックス

```
<cfpod
    source = "path"
    bodyStyle = "CSS style specification"
    headerStyle = "CSS style specification"
    height = "number of pixels"
    name = "string"
    onBindError = "JavaScript function name"
    title = "string"
    width = "number of pixels"/>

OR

<cfpod
    bodyStyle = "CSS style specification"
    headerStyle = "CSS style specification"
    height = "number of pixels"
    name = "string"
    onBindError = "JavaScript function name"
    title = "string"
    width = "number of pixels">
        pod contents
</pod>
```

本文と終了タグを省略する場合は、タグの最後を /> で閉じます。

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfajaximport、cfdiv、cflayout、cfwindow

履歴

ColdFusion 8: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
bodyStyle	オプション		ポッド本体の CSS スタイル指定です。 原則として、この属性は色とフォントスタイルを設定する目的に使用します。たとえば、この属性を使用して高さや幅を設定すると、予期しない結果になる場合があります。
headerStyle	オプション		ポッドヘッダの CSS スタイル指定です。 原則として、この属性は色とフォントスタイルを設定する目的に使用します。たとえば、この属性を使用して高さや幅を設定すると、予期しない結果になる場合があります。
height	オプション	100	タイトルバーとボーダーを含むコントロールの高さです (単位 : ピクセル)。
name	オプション		ポッドコントロールの名前です。
onBindError	オプション	「説明」を参照	バインド式の評価でエラーが発生した場合に実行する JavaScript 関数の名前です。この関数は 2 つの属性 (HTTP ステータスコードとメッセージ) を取る必要があります。 この属性を省略し、(ColdFusion.setGlobalErrorHandler 関数を使用して) グローバルエラーハンドラを指定すると、エラーメッセージが表示されます。それ以外の場合は、デフォルトのエラーポップアップが表示されます。

属性	必須 / オプション	デフォルト	説明
overflow	オプション	auto	子コンテンツのサイズが大きいためにコントロールがポッドの境界を超えてしまう場合に、子コンテンツをどのように表示するかを指定します。使用できる値は次のとおりです。 • auto: 必要に応じてスクロールバーを表示します。 • hidden: 境界からはみ出したコンテンツにアクセスできないようにします。 • scroll: 必要がない場合でも常に水平および垂直方向のスクロールバーを表示します。 • visible: ポッドの境界の外側にコンテンツを表示します。 **メモ :** Internet Explorer では、ポッドを visible に設定した場合、レイアウト領域を超えてコンテンツを表示するのではなく、コンテンツのサイズに合わせてポッドが拡大されます。
source	タグに本文がない場合は必須		ポッドのコンテンツを返す URL です。ColdFusion では標準のページパス解決規則が使用されます。 この属性を指定して、cfpod タグの本文を指定した場合、本文のコンテンツは無視されます。 この属性では依存関係を持つバインド式を使用できます。詳細については、「使用方法」を参照してください。 **メモ :** この属性で指定した CFML ページに AJAX 機能を使用するタグ (cfform、cfgrid、cfwindow など) が含まれている場合は、そのページで cfpod タグとともに cfajaximport タグを使用する必要があります。詳細については、cfajaximport を参照してください。
title	オプション		ポッドのタイトルバーに表示するテキストです。HTML マークアップを使用するとタイトルの外観を制御できます (たとえば、赤いイタリックフォントでテキストを表示することができます)。この属性を省略すると、ポッドのタイトルバーは作成されません。
width	オプション	500	ボーダーを含むコントロールの幅です (単位 : ピクセル)。

使用方法

ポッドのコンテンツを指定するには、source 属性またはタグ本文を使用します。両方とも指定されている場合は、source 属性で指定されたコンテンツが使用され、タグ本文は無視されます。source 属性を使用すると、コンテンツの取得中にアニメーションのアイコンと「ロード中 ...」というテキストが表示されます。

JavaScript 関数の定義を含むページを source 属性で指定する場合、そのページの関数定義は次の形式に準拠している必要があります。

```
functionName = function(arguments) {function body}
```

次のような関数定義は、正常に動作しない可能性があります。

```
function functionName (arguments) {function body}
```

ただし、カスタム JavaScript すべてを外部の JavaScript ファイルにインクルードして、アプリケーションのメインページにインポートすることをお勧めします。source 属性を使用して取得するコードには、カスタム JavaScript をインラインで記述しないでください。インポートするページには、関数定義に関するこのような制限はありません。

source 属性を使用する場合は、バインド式を使用してフォームフィールドの値やその他のフォームコントロール属性をソース指定の一部として含めることができます。バインド可能なフォームコントロールは、HTML 形式のフォームコントロールのみです。

バインド式を使用するには、URL を指定し、バインドパラメータを含む URL パラメータをページに渡します。基本的な形式では、バインド対象となるコントロールの name 属性または id 属性を中括弧 ({}) で囲んだものがバインドパラメータになります。たとえば、バインドパラメータとして city コントロールの値を使用するには、次の形式を使用します。

```
source="/myapplication/cityPod.cfm?cityname={city}"
```

バインド式の使用に関する詳細については、『ColdFusion アプリケーションの開発』の Binding data to form fields を参照してください。

例

次の CFML ページは、2 つのポッドを垂直に並べて表示します。各ポッドは displayforpod.cfm ページからコンテンツを取得します。このページは cffeed タグを使用して Atom フィードを取得します。

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
<title>Untitled Document</title>
</head>

<body>
<cflayout type="hbox" style="background-color:##CCffFF; color:red;">
    <cflayoutarea>
        <cfpod name="pod01" source="displayforpod.cfm?start=1" height="500" width="300"
            title="Comment 1"/>
    </cflayoutarea>
    <cflayoutarea>
        <cfpod name="pod02" source="displayforpod.cfm?start=2" height="500" width="450"
            title="Comment 2"/>
    </cflayoutarea>
</cflayout>
</body>
</html>
```

displayforpod.cfm ページの内容は次のとおりです。

```
<cffeed action="read" source="http://googleblog.blogspot.com/atom.xml"
            query="feedQuery" properties="feedMetadata" >

<cfloop query = "feedQuery"
        startRow = "#url.start#" endRow = "#url.start#">
    <cfoutput>#feedQuery.content#<br />
    ========================================<br/>
    </cfoutput>
</cfloop>
```

cfpop

説明

POP メールサーバーから電子メールメッセージの取得や削除を行います。

カテゴリ

通信タグ、インターネットプロトコルタグ

シンタックス

```
<cfpop
    server = "server name"
    action = "getHeaderOnly|getAll|delete"
    attachmentPath = "path"
    debug = "yes|no">
    generateUniqueFilenames = "yes|no"
    maxRows = "number"
    messageNumber = "number"
    name = "query name"
    password = "password"
    port = "port number"
    secure = "yes|no">
    startRow = "number"
    timeout = "seconds"
    uid = "number"
    username = "user name">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfftp、cfhttp、cfldap、cfmail、cfmailparam、SetLocale、『ColdFusion アプリケーションの開発』の Sending and Receiving E-Mail

履歴

ColdFusion 10：secure 属性が追加されました。

ColdFusion MX 7.01: cids クエリー変数が追加されました。

ColdFusion MX 6.1:

- テキストおよび HTML のパートを含むマルチパートメールメッセージのサポートが追加されました。
- 添付ファイル名のセパレータが変更されました。メッセージに複数の添付ファイルが含まれる場合、タブ記号が attachments および attachmentfiles クエリーフィールドの添付ファイル名のセパレータになりました。この動作は ColdFusion 5 およびそれ以前のバージョンと同じです。

ColdFusion MX: 添付ファイル名のセパレータが変更されました。メッセージに複数の添付ファイルが含まれる場合、カンマが attachments および attachmentfiles クエリーフィールドの名前を区切ります。

属性

属性	必須 / オプション	デフォルト	説明
server	必須		POP サーバーの識別子です。 • ホスト名 (例 : biff.upperlip.com) • IP アドレス (例 : 192.1.2.225)
action	オプション	getHeaderOnly	• getHeaderOnly: メッセージヘッダの情報のみを返します。 • getAll: メッセージヘッダの情報、メッセージテキスト、および attachmentPath を指定している場合は添付ファイルを返します。 • delete: POP サーバー上のメッセージを削除します。
attachmentPath	オプション		action="getAll" の場合に、添付ファイルを保存するディレクトリを指定します。指定したディレクトリが存在しない場合は、自動的に作成されます。 この属性を省略すると、添付ファイルは保存されません。相対パスを指定する場合は、GetTempDirectory 関数によって返される ColdFusion テンポラリディレクトリがパスのルートになります。
debug	オプション	no	• yes: デバッグ出力を標準出力に送信します。デフォルトでは、コンソールウィンドウを使用できない場合、ColdFusion は出力を <ColdFusion のルートディレクトリ >/runtime/logs/coldfusion-out.log に送信します。 • no: デバッグ出力を生成しません。
generateUniqueFilenames	オプション	no	• yes: 電子メールメッセージに添付したファイルに固有のファイル名を生成して、ファイル保存時の名前の重複を避けます。 • no
maxRows	オプション	取得可能なすべての行を取得	startRow の番号から数えて、取得または削除するメッセージの数です。messageNumber 属性または uid 属性を指定すると、この属性は無視されます。
messageNumber			取得または削除するメッセージ番号またはメッセージ番号のカンマ区切りリストです。無効なメッセージ番号は無視されます。 uid 属性を指定すると、この属性は無視されます。
name	action="getAll" または "getHeaderOnly" の場合は必須		取得したメッセージの情報が含まれるクエリーオブジェクトの名前です。
password	オプション		username に対応するパスワードです。
port	オプション	110	POP ポートです。
secure	オプション	no	yes に設定すると、pop リクエストの SSL が有効化されます。
startRow	オプション	1	取得または削除する最初の行番号です。messageNumber 属性または uid 属性を指定すると、この属性は無視されます。

属性	必須 / オプション	デフォルト	説明
timeout	オプション	60	メール処理を待つ最大時間です (単位 : 秒)。
uid			取得または削除する UID または UID のカンマ区切りリストです。無効な UID は無視されます。
username	オプション		ユーザー名です。

使用方法

cfpop タグは、POP サーバーからメールメッセージを取得し、取得したメッセージを 1 行 1 メッセージで、ColdFusion クエリーオブジェクトに挿入します。または、POP サーバーからメッセージを削除します。

注意：cfpop タグが不正な形式のメールメッセージを取得した場合、エラーは生成されず、空のフィールドが返されます。

パフォーマンスを最適化するために、2 つの取得オプションを利用できます。メッセージヘッダの情報は、通常は短いため、短時間で転送されます。メッセージテキストおよび添付ファイルは長い場合もあり、処理に時間がかかることもあります。

attachmentPath 属性

attachmentPath 属性で、メモリ内のディレクトリを指定するには、次のシンタックスを使用します。メモリ内のファイルを使用すると、一時的データの処理が速くなります。

```
ram:///path
```

パスには、ram:///petStore/mail/attachments のように複数のディレクトリを含めることができます。パス内のディレクトリは、ファイルを指定する前に作成する必要があります。メモリ内のファイルの使用方法について詳しくは、『ColdFusion アプリケーションの開発』の Working with in-memory files を参照してください。

cfpop のクエリー変数

次の表では、cfpop によって返されるクエリーに関する情報を提供する変数について説明します。

変数名	説明
queryname.recordCount	クエリーによって返されるレコードの数です。
queryname.currentRow	cfoutput により処理されている現在の行です。
queryname.columnList	クエリー内の列名のリストです。

クエリーメッセージヘッダおよび本文の列

次の表は、action = "getHeaderOnly" または "getAll" の場合に返されるメッセージヘッダおよび本文の列のリストです。

列名	getHeaderOnly	getAll
queryname.date	yes	yes
queryname.from	yes	yes
queryname.messagenumber	yes	yes
queryname.messageid	yes	yes
queryname.replyto	yes	yes
queryname.subject	yes	yes
queryname.cc	yes	yes
queryname.to	yes	yes

列名	getHeaderOnly	getAll
queryname.body	no	yes
queryname.textBody	no	yes
queryname.HTMLBody	no	yes
queryname.header	yes	yes
queryname.attachments	no	yes
queryname.attachmentfiles	no	yes
queryname.UID	yes	yes
queryname.cids	no	yes

メールメッセージに Content-Type が text/plain のパートが含まれる場合は、queryname.textBody 列にパートのメッセージの内容が入ります。メールメッセージに Content-Type が text/HTML のパートが含まれる場合は、queryname.HTMLBody 列にパートのメッセージの内容が入ります。これらのタイプと一致する Content-Type がない場合は、列は空になります。queryname.Body 列には、常に最初に見つかったメッセージ本文が入ります。

queryname.attachments 列には、すべての添付ファイル名を示すタブ区切りのリストが入ります。queryname.attachmentfiles 列には、添付ファイルの場所を示すタブ区切りのリストが入ります。処理が終わったテンポラリファイルを削除するには、cffile タグを使用します。

queryname.date 列のメールメッセージから抽出される日付時刻文字列から ColdFusion の日付時刻オブジェクトを作成するには、次の表のようにします。

ロケール	queryname.date から ColdFusion の日付時刻オブジェクトを作成する方法
English (US)	ParseDateTime 関数を使用します。pop-conversion 属性を指定すると、関数は日付 / 時刻オブジェクトを UTC に調整します。
その他	文字列から日付部分を抽出し、それを LSParseDateTime 関数に渡します。

注意：日付、時刻、数値、および通貨の値のデフォルトの表示形式を設定するには、SetLocale 関数を使用してください。

cfpop の詳細については、『ColdFusion アプリケーションの開発』の Sending and Receiving E-Mail を参照してください。

例

```
<!--- This view-only example shows the use of cfpop. --->
<h3>cfpop Example</h3>
<p>cfpop lets you retrieve and manipulate mail in a POP3 mailbox.
    This view-only example shows how to create one feature of
    a mail client, to display the mail headers in a POP3 mailbox.
<p>To execute this, un-comment this code and run with a mail-enabled CF Server.
<!---
<cfif IsDefined("form.server")>
    <!--- Make sure server, username are not empty. --->
    <cfif form.server is not "" and form.username is not "">
        <cfpop server = "#form.popserver# " username = #form.username# password = #form.pwd#
        action = "getHeaderOnly" name = "GetHeaders ">
        <h3>Message Headers in Your Inbox</h3>
        <p>Number of Records:
        <cfoutput>#GetHeaders.recordCount#</cfoutput></p>

        <ul>
            <cfoutput query = "GetHeaders">
            <li>Row: #currentRow#: From: #From# -- Subject: #Subject#
            </cfoutput>
        </ul>
    </cfif>
</cfif>

<form action = "cfpop.cfm " method = "post">
    <p>Enter your mail server:</p>
    <p><input type = "Text" name = "popserver"></p>
    <p>Enter your username:</p>
    <p><input type = "Text" name = "username"></p>
    <p>Enter your password:</p>
    <p><input type = "password" name = "pwd"></p>
    <p><input type = "Submit" name = "get message headers"></p>
</form>
--->
```

cfpresentation

説明

ダイナミックなスライドプレゼンテーションの外観を定義し、プレゼンテーションファイルをディスクに書き込むかどうかを決定します。cfpresentation タグは、プレゼンテーションのコンテンツを定義する cfpresentationslide タグ、およびスライドのプレゼンタに関する情報を提供する cfpresenter タグの親タグになります。

履歴

ColdFusion 9: format 属性および destination 属性が追加されました。

ColdFusion 8: このタグが追加されました。

カテゴリ

データ出力タグ

シンタックス

```
<cfpresentation
    title = "text string"
    authPassword = "authentication password"
    authUser = "authentication user name"
    autoPlay = "yes|no"
    backgroundColor = "hexadecimal color|HTML named color"
    control = "normal|brief"
    controlLocation = "right|left"
    destination = "filepath"
    directory = "pathname"
    format = "ppt|html"
    glowColor = "hexadecimal color|HTML named color"
    initialTab = "outline|search|notes"
    lightColor = "hexadecimal color|HTML named color"
    loop = "yes|no"
    overwrite = "yes|no"
    primaryColor = "hexadecimal color|HTML named color"
    proxyHost = "IP address or server name for proxy host"
    proxyPassword = "password for the proxy host"
    proxyPort = "port of the proxy host"
    proxyUser = "user name for the proxy host"
    shadowColor = "hexadecimal color|HTML named color"
    showNotes = "yes|no"
    showOutline = "yes|no"
    showSearch = "yes|no"
    textColor = "hexadecimal color|HTML named color"
    userAgent = "HTTP user agent identifier">
    presentation content...
</cfpresentation>
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfchart、cfpresentationslide、cfpresenter、cfreport、『ColdFusion アプリケーションの開発』の Creating Slide Presentations

属性

属性	必須 / オプション	デフォルト	説明
authPassword	オプション		基本認証のターゲット URL にパスワードを送信します。username と組み合わせて、認証ヘッダで渡される base64 エンコード文字列を形成します。統合 Windows 認証、NTLM 認証、または Kerberos 認証には対応しません。
authUser	オプション		基本認証のターゲット URL にユーザー名を送信します。password と組み合わせて、認証ヘッダで渡される base64 エンコード文字列を形成します。統合 Windows 認証、NTLM 認証、または Kerebos 認証には対応しません。
autoPlay	オプション	yes	プレゼンテーションを自動的に再生するかどうかを指定します。 • yes: 起動時にプレゼンテーション全体が自動的に実行されます。 • no: プレゼンテーションを開始したり次のスライドに進むには、ユーザーが手動で [再生] ボタンや [次へ] ボタンをクリックする必要があります。

属性	必須 / オプション	デフォルト	説明
backgroundColor	オプション	727971	プレゼンテーションの背景色です。"##xxxxxx" または "##xxxxxxxx" という形式を使用して、16 進数値で指定します。ここで、x は 0 ～ 9 または A ～ F です。シャープ記号 (#) は 2 つ使用するか、または使用しないでください。「カラー名」の表に示した HTML カラー名の一部を使用することもできます。
control	オプション	normal	プレゼンテーションコントロールのタイプです。 • normal • brief
controlLocation	オプション	right	プレゼンテーションコントロールの位置を指定します。 • right • left
destination	オプション		ファイルの絶対パス、または CFM ページへのファイルの相対パスです。Connect プレゼンテーションと ppt プレゼンテーションの両方に、この属性を使用することができます。format="html" の場合は必須です。
directory	オプション		プレゼンテーションを保存するディレクトリです。絶対パスまたは CFM ページからの相対パスを使用できます。また、次のファイルを含む data というサブディレクトリが ColdFusion によって作成されます。 • 各スライドの SWF ファイル • srchdata.xml (検索インターフェイスを作成するファイル) • vconfig.xml • viewer.xml • cfpresentationslide タグによって参照されるイメージ、ビデオクリップ、および SWF ファイル ディレクトリを指定しない場合は、temp ディレクトリにファイルが書き込まれ、クライアントブラウザ内でプレゼンテーションが実行されます。
format	オプション		変換するファイル形式を指定します。 • ppt を指定すると、cfpresentationslide で指定された HTML 入力を PowerPoint ファイルに変換します。 • html を指定すると、ppt を HTML プレゼンテーションに変換します。
glowColor	オプション	35D334	ボタンのグロー効果に使用する色です。"##xxxxxx" または "##xxxxxxxx" という形式を使用して、16 進数値で指定します。ここで、x は 0 ～ 9 または A ～ F です。シャープ記号 (#) は 2 つ使用するか、または使用しないでください。「カラー名」の表に示した HTML カラー名の一部を使用することもできます。
initialTab	オプション	outline	プレゼンテーションの実行時にどのタブを手前に表示するかを指定します。これは、control の値が normal である場合にのみ適用されます。 • outline • search • notes
lightColor	オプション	4E5D60	ライト / シャドウ効果に使用するライトの色です。"##xxxxxx" または "##xxxxxxxx" という形式を使用して、16 進数値で指定します。ここで、x は 0 ～ 9 または A ～ F です。シャープ記号 (#) は 2 つ使用するか、または使用しないでください。「カラー名」の表に示した HTML カラー名の一部を使用することもできます。

属性	必須 / オプション	デフォルト	説明
loop	オプション	no	プレゼンテーションを繰り返し再生するかどうかを指定します。 • yes: プレゼンテーションが終了すると自動的に最初から再生し直されます。 • no: プレゼンテーションを最初から再生し直すには、ユーザーが [再生] ボタンをクリックする必要があります。
overwrite	オプション	yes	ディレクトリ内のファイルを上書きするかどうかを指定します。directory 属性が指定されている場合にのみ有効です。 • yes: ファイルが既に存在する場合、ファイルを上書きします。 • no: 新しいファイルを作成します。
primaryColor	オプション	6F8488	プレゼンテーションの基本色です。"##xxxxxx" または "##xxxxxxxxx" という形式を使用して、16 進数値で指定します。ここで、x は 0 ～ 9 または A ～ F です。シャープ記号 (#) は 2 つ使用するか、または使用しないでください。「カラー名」の表に示した HTML カラー名の一部を使用することもできます。
proxyHost	オプション		リクエストの送信先となるプロキシサーバーのホスト名または IP アドレスです。
proxyPassword	オプション		プロキシサーバーで要求されるパスワードです。
proxyPort	オプション	80	プロキシサーバー上で接続するポートです。
proxyUser	オプション		プロキシサーバーに提供するユーザー名です。
shadowColor	オプション	000000	ライト / シャドウ効果に使用するシャドウの色です。"##xxxxxx" または "##xxxxxxxxx" という形式を使用して、16 進数値で指定します。ここで、x は 0 ～ 9 または A ～ F です。シャープ記号 (#) は 2 つ使用するか、または使用しないでください。「カラー名」の表に示した HTML カラー名の一部を使用することもできます。
showNotes	オプション	no	プレゼンテーションコントロールパネルに [メモ] タブを表示するかどうかを指定します。 • yes • no
showOutline	オプション	yes	プレゼンテーションコントロールパネルにアウトラインを表示するかどうかを指定します。 • yes • no
showSearch	オプション	yes	プレゼンテーションコントロールパネルに [検索] タブを表示するかどうかを指定します。 • yes • no
textColor	オプション	FFFFFF	プレゼンテーションのユーザーインターフェイスに表示されるすべてのテキストの色です。"##xxxxxx" または "##xxxxxxxxx" という形式を使用して、16 進数値で指定します。ここで、x は 0 ～ 9 または A ～ F です。シャープ記号 (#) は 2 つ使用するか、または使用しないでください。「カラー名」の表に示した HTML カラー名の一部を使用することもできます。
title	必須		プレゼンテーションのタイトルです。
userAgent	オプション	ColdFusion	HTTP User-Agent リクエストヘッダフィールドに配置されるテキストです。リクエストクライアントソフトウェアを識別するために使われます。

使用方法

cfpresentation タグを使用すると、スライドプレゼンテーションのコンテナを作成できます。プレゼンテーションコントロールの位置や外観、背景色、プレゼンテーションのテキストも定義できます。また、このタグでは、プレゼンテーションをファイルに書き込むか、クライアントブラウザ内で直接実行するかを指定できます。

cfpresentation タグの設定は、cfpresentationslide タグで定義されたコンテンツの外観には影響しません。

destination 属性

次のシンタックスを使用すると、destination 属性のディスクに書き込まれない、メモリ内のファイルを指定できます。メモリ内のファイルを使用すると、一時的データの処理が速くなります。

```
ram:///filepath
```

ファイルパスには、ram:///petStore/presentations/quarterlyresults.html のようなディレクトリを含めることができます。ファイルを指定する前に、パスに含まれるディレクトリを作成しておく必要があります。メモリ内のファイルの使用方法について詳しくは、『ColdFusion アプリケーションの開発』の Working with in-memory files を参照してください。

カラー名

cfpresentation タグの backgroundColor、glowColor、lightColor、primaryColor、shadowColor、および textColor の各属性では、次のカラー名を指定できます。

カラー名	16 進数値
red	FF0000
green	008000
blue	0000FF
black	000000
white	FFFFFF
yellow	FFFF00
gray	808080
darkgray	A9A9A9
lightgray	D3D3D3
cyan	00FFFF
magenta	FF00FF
orange	FFA500
pink	FFC0CB

例

```
<!--- This example shows how to create a slide presentation from --->
<!--- an HTML file and from HTML code on the CFM page and write --->
<!--- the presentation files to a directory called myPresentation, --->
<!--- which is relative to the CFM page. --->
<cfpresentation title="Sales Presentation" directory="myPresenation">
    <cfpresenter name="Shyam" title="Vice President" email="shyam@somecompany.com" image="shyam.jpg">
    <cfpresenter name="Ram" title="Sr. Vice President" email="ram@somecompany.com">

<!--- The following code creates a slide from an HTML file --->
<!--- located on the ColdFusion server. --->
    <cfpresentationslide src="introduction.htm" title="Introduction" presenter="Shyam"
        audio="myAudio.mp3" duration="36"/>

<!--- The following code creates a slide from HTML code in the CFM file. --->
    <cfpresentationslide>
        <h3>Sales</h3>
        <ul>
            <li>Overview</li>
            <li>Q1 Sales Figures</li>
            <li>Projected Sales</li>
            <li>Competition</li>
            <li>Advantages</li>
            <li>Long Term Growth</li>
        </ul>
    </cfpresentationslide>

<!--- The following code creates a slide from HTML and CFML code. --->
    <cfpresentationslide Title="Q1 Sales Figures" duration="14" presenter="Ram"
            audio="myAudio2.mp3">
        <h3>Q1 Sales Figures</h3>
        <cfchart format="png" showborder="yes" chartheight="250" chartwidth="300"
            pieslicestyle="sliced">
        <cfchartseries type="pie">
            <cfchartdata item="Europe" value="9">
            <cfchartdata item="Asia" value="20">
            <cfchartdata item="North America" value="50">
            <cfchartdata item="South America" value="21">
        </cfchartseries>
        </cfchart>
    </cfpresentationslide>
</cfpresentation>
```

cfpresentationslide

説明

ソースファイルまたは ColdFusion ページ上の HTML コードおよび CFML コードからダイナミックにスライドを作成します。cfpresentationslide は cfpresentation タグの子タグになります。

履歴

ColdFusion 9: slides 属性が追加されました。src 属性に PowerPoint ファイルサポートが追加されました。

ColdFusion 8: このタグが追加されました。

カテゴリ

データ出力タグ

シンタックス

```
<cfpresentation ...>
    <cfpresentationslide
    advance = "auto|never|click"
    audio = "pathname relative to the CFM page or the web root for audio file"
    authPassword = "authentication password"
    authUser = "authentication user name"
    duration = "duration of slide in seconds"
    marginBottom = "margin in pixels"
    marginLeft = "margin in pixels"
    marginRight = "margin in pixels"
    marginTop = "margin in pixels"
    notes = "text string"
    presenter = "presenter name"
    scale = "decimal"
    src = "absolute path|URL|path relative to CFM page"
    title = "text string"
    userAgent = "HTTP user agent identifier"
    video = "pathname relative to the CFM page
    or the web rootfor video file"
    useExternalProgram = "true|false" />
</cfpresentation>
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfchart、cfpresentation、cfpresenter、cfreport、『ColdFusion アプリケーションの開発』の Creating Slide Presentations

属性

属性	必須 / オプション	デフォルト	説明
advance	オプション	「説明」を参照	スライドごとに cfpresentation タグの autoPlay 属性を上書きします。 • auto: 現在のスライドを再生した後、自動的に次のスライドへ進みます。cfpresentation autoPlay="yes" の場合は、これがデフォルトの設定です。 • never: 現在のスライドを再生した後、ユーザーが [次へ] ボタンをクリックするまで次のスライドへ進みません。cfpresentation autoPlay="no" の場合は、これがデフォルトの設定です。 • click: 現在のスライドを再生した後、ユーザーがメインのプレゼンテーション領域内をクリックすると、次のスライドへ進みます。
audio	オプション		CFM ページまたは Web ルートを基準としたオーディオファイルの相対パス名です。オーディオファイルは MP3 ファイルである必要があります。 スライドのオーディオとビデオを両方指定することはできません。
authPassword	オプション		基本認証のターゲット URL にパスワードを渡すために使用します。username と組み合わせて、認証ヘッダで渡される base64 エンコード文字列を形成します。統合 Windows 認証、NTLM 認証、または Kerberos 認証には対応しません。
authUser	オプション		基本認証のターゲット URL にユーザー名を渡すために使用します。password と組み合わせて、認証ヘッダで渡される base64 エンコード文字列を形成します。統合 Windows 認証、NTLM 認証、または Kerberos 認証には対応しません。
duration	オプション		スライドの再生時間です (単位 : 秒)。時間を指定しない場合、そのスライドに関連付けられたオーディオクリップの再生時間がスライドの再生時間になります。
marginBottom	オプション	0	スライドの下マージンです。

属性	必須 / オプション	デフォルト	説明
marginLeft	オプション	0	スライドの左マージンです。
marginRight	オプション	0	スライドの右マージンです。
marginTop	オプション	0	スライドの上マージンです。
notes	オプション		スライドで使用するメモです。メモは cfpresentationslide タグの showNotes 属性が yes に設定されている場合にのみ表示されます。
presenter	オプション		スライドのプレゼンタです。1 つのスライドで指定できるプレゼンタは 1 人のみです。この名前は、cfpresenter タグのプレゼンタ名と一致している必要があります。
scale	オプション	1.0	スライドプレゼンテーションの HTML コンテンツに使用するスケールです。スケールを指定しない場合は、スライドのサイズに合わせてコンテンツが自動的に拡大または縮小されます。
slides	オプション	すべてのスライド	src 属性で PowerPoint ファイルを参照している場合に、エクスポートに必要なスライド番号を指定します。範囲を指定する場合はハイフンを使用し、連続していないスライドを指定する場合はカンマを使用します。たとえば、次のようになります。 slides="1-10"、slides="1,10"
src	オプション		スライドとして使用する HTML、SWF、または PPT ソースファイルです。スライドのソースとして指定できる値は次のいずれかです。 • 絶対パス • CFM ページからの相対パス • URL: ソースが HTML コンテンツを返す場合に指定します。 SWF ファイルは、ColdFusion を実行しているシステム上に存在している必要があり、絶対パスまたは CFM ページからの相対パスで指定する必要があります。 ソースファイルを指定しない場合は、HTML/CFML コードを本文に含めます。ソースファイルと HTML/CFML の両方を指定した場合はソースファイルが無視され、スライド内に HTML/CFML コンテンツが表示されます。
title	オプション		スライドのタイトルです。
userAgent	オプション	ColdFusion	HTTP User-Agent リクエストヘッダフィールドに配置されるテキストです。これにより、リクエストクライアントソフトウェアが識別されます。
useExternalProgram	オプション	true	OpenOffice ライブラリと POI ライブラリとを切り替えるためのブール値です。 • true: OpenOffice ライブラリ • false: POI ライブラリ
video	オプション		スライドのプレゼンタのビデオファイルです。スライドのビデオとプレゼンタのイメージを指定した場合は、イメージではなくビデオがスライドで使用されます。スライドのオーディオとビデオを両方指定することはできません。ビデオは FLV ファイルまたは SWF ファイルである必要があります。 ビデオファイルは CFM ページまたは Web ルートからの相対パスで指定する必要があります。

使用方法

cfpresentation タグ内で cfpresentationslide タグを使用すると、個々の SWF または HTML ソースファイルからスライドプレゼンテーションを作成できます。ソースファイルを指定しない場合は、スライド本文の HTML または CFML コードを cfpresentationslide タグ内に含めます。1 つのスライドに割り当てられるプレゼンタは 1 人のみです。cfpresentationslide タグによって参照されるプレゼンタを定義するには、cfpresenter タグを使用します。

次のコードは、既存の SWF ファイルからスライドプレゼンテーションを作成する方法を示しています。

```
<!--- The following example shows how to create a slide presentation --->
<!--- from individual SWF files located on the ColdFusion server. --->
<!--- Because no directory is specified, the presentation runs in --->
<!--- the browser. --->
<cfpresentation title="myPresentation">
    <cfpresentationslide title="1st slide" src="slide1.swf" duration="10"/>
    <cfpresentationslide title="2nd slide" src="slide2.swf"
        audio="audio1.mp3" duration="20"/>
    <cfpresentationslide title="3rd slide" src="slide3.swf"
        audio="audio2.mp3" duration="218"/>
</cfpresentation>
```

注意：cfpresentationslide タグには終了タグが必要です。スライドの内容として、(開始タグと終了タグの間の CFML および HTML コードではなく) ソースファイルを指定する場合は、終了タグのショートカットである終了スラッシュを使用します。

HTML コンテンツを返すソースファイルの URL であれば、それらの URL を参照できます。次のコードは、外部 Web サイト上にある HTML ファイルからスライドプレゼンテーションを作成する方法を示しています。

```
<!--- The following example shows how to create a slide presentation --->
<!--- from HTML files located on an external site. --->
<cfpresentation title="USGS Naming Conventions" directory="myPresenation">
    <cfpresenter name="Robert L. Payne" title="Executive Secretary">
    <cfpresenter name="Trent Palmer" title="Executive Secretary Foreign Names">
    </cfpresentationslide>
    <cfpresentationslide src="http://geonames.usgs.gov/index.html"
        duration="10" presenter="Robert L. Payne"/>
    <cfpresentationslide src="http://geonames.usgs.gov/domestic/index.html"
        duration="15" presenter="Robert L. Payne"/>
    <cfpresentationslide src="http://geonames.usgs.gov/foreign/index.html"
        duration="15" presenter="Trent Palmer"/>
    </cfpresentation>
```

注意：HTML ファイルから作成されたスライド内のリンクは機能しません。

スライドの本文として HTML および CFML コードを入力することもできます。次の例のように、コードにはチャート、グラフ、イメージを含めることができます。

```
<!--- This example shows how to create a slide presentation dynamically --->
<!--- from HTML code and CFML code. Because no directory is specified, --->
<!--- the presenation runs in the client browser. --->
<cfpresentation title="Sales Presentation">
    <cfpresenter name="Shyam" title="Vice President" email="shyam@somecompany.com">
    <cfpresenter name="Ram" title="Sr. Vice President" email="ram@somecompany.com">
    <cfpresentationslide title="Introduction" presenter="Shyam" audio="myAudio3.mp3"
            duration="10">
        <h3>Introduction</h3>
        <table>
            <tr>
                <td>
                    <ul>
                        <li>Overview</li>
                        <li>Q1 Sales Figures</li>
                        <li>Projected Sales</li>
                        <li>Competition</li>
                        <li>Advantages</li>
                        <li>Long Term Growth</li>
                    </ul>
                </td>
                <td><img src="../cfdocs/images/artgallery/paul01.jpg"/></td>
            </tr>
        </table>
    </cfpresentationslide>
    <cfpresentationslide Title="Q1 Sales Figures" duration="14" presenter="Ram"
            audio="myAudio1.mp3">
        <h3>Q1 Sales Figures</h3>
        <cfchart format="png" showborder="yes" chartheight="250" chartwidth="300"
            pieslicestyle="sliced">
            <cfchartseries type="pie">
                <cfchartdata item="Europe" value="9">
                <cfchartdata item="Asia" value="20">
                <cfchartdata item="North America" value="50">
                <cfchartdata item="South America" value="21">
            </cfchartseries>
        </cfchart>
    </cfpresentationslide>
</cfpresentation>
```

例

```
<!--- The following example shows how to create a slide presentation --->
<!--- dynamically from HTML in ColdFusion and from HTML files located --->
<!--- on an external site. ColdFusion writes the presentation files --->
<!--- to a directory relative to the CFM page. --->
<cfpresentation title="USGS Naming Conventions" directory="namingConventions">
    <cfpresenter name="Robert L. Payne" title="Executive Secretary">
    <cfpresenter name="Trent Palmer" title="Executive Secretary Foreign Names">
    <cfpresentationslide presenter="Robert L. Payne">
    <h3>USGS Naming Conventions</h3>
    <ul>
        <li>Overview</li>
        <li>General Naming Conventions</li>
        <li>Domestic Naming Conventions</li>
        <li>Foreign Naming Conventions</li>
    </ul>
    <p></p>
    </cfpresentationslide>
        duration="10" presenter="Robert L. Payne"/>
    <cfpresentationslide src="http://geonames.usgs.gov/domestic/index.html"
        duration="15" presenter="Robert L. Payne"/>
    <cfpresentationslide src="http://geonames.usgs.gov/foreign/index.html"
        duration="15" presenter="Trent Palmer"/>
</cfpresentation>
```

cfpresenter

説明

スライドプレゼンテーション内でプレゼンタの紹介を行います。1 つのスライドプレゼンテーションには複数のプレゼンタを指定できます。プレゼンタは cfpresentationslide タグで定義されたスライドから参照する必要があります。

履歴

ColdFusion 8: このタグが追加されました。

カテゴリ

データ出力タグ

シンタックス

```
<cfpresenter
    biography = "text string"
    email = "e-mail address of the presenter"
    image = "relative pathname for JPG"
    name = "text string"
    logo = "relative pathname for JPG"
    title = "text string">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfchart、cfpresentation、cfpresentationslide

属性

属性	必須 / オプション	デフォルト	説明
biography	オプション		プレゼンタに関する情報を提供するテキスト文字列です (例 : "Sally Maverick はアドビ製品の売上において過去 5 年間トップの成績を収めています。")。
email	オプション		プレゼンタの電子メールアドレス。この属性を指定するとプレゼンテーションコントロールパネルの [Contact] リンクがアクティブになり、このリンクをクリックすると電子メールメッセージが開きます。
image	オプション		プレゼンタの JPEG 形式イメージのパス名です。この JPEG ファイルは CFM ページからの相対パスで指定する必要があります。cfpresentationslide タグでビデオが指定されている場合、そのスライドではここで指定した値ではなくビデオクリップが使用されます。
name	必須		プレゼンタの名前です。プレゼンタをスライドに関連付けるには、cfpresentationslide タグの presenter 属性でこの値を使用します。
logo	オプション		プレゼンタのロゴまたはプレゼンタの組織のロゴを表示するイメージファイルのパス名です。ロゴのイメージは JPEG 形式である必要があります。このファイルは CFM ファイル Web サイトからの相対パスで指定する必要があります。
title	オプション		プレゼンタの役職です (例 : " 営業部長 ")。

使用方法

cfpresenter タグを使用すると、スライドごとに指定するプレゼンタを定義できます。プレゼンタの情報は、その担当者に関連付けられたスライドのコントロールパネルに表示されます。スライドごとのプレゼンタを指定するには、cfpresentationslide タグの presenter 属性を使用します。

プレゼンタのイメージとプレゼンタの会社のロゴを指定するには、presenter タグの image 属性と logo 属性を使用します。プレゼンタのイメージの代わりにビデオクリップを表示するには、cfpresentationslide タグの video 属性で FLV または SWF ファイルを指定します。

例

```
<!--- This example shows how to specify presenters for a slide --->
<!--- presentation and assign a presenter to each slide in the presentation. --->
<cfpresentation title="myPresentation" directory="presentation" overwrite="yes">

<!--- The following code defines three presenters. --->
<cfpresenter name="Shyam" title="President" email="shyam@somecompany.com"
    image="images\shyam01.jpg">
<cfpresenter name="Ram" title="V.P. Sales" email="ram@somecompany.com"
    image="images\ram01.jpg">
<cfpresenter name="Michelle" title="V.P. Engineering"
    email="mhatter@adobe.com" image="images\michelle01.jpg">

<!--- The following code assigns a presenter to each of three slides in the presentation. --->
<cfpresentationslide title="myFirstSlide" src="slide1.swf" duration="10"
    presenter="Shyam"/>
<cfpresentationslide title="mySecondSlide" src="slide2.swf" duration="15"
    presenter="Michelle"/>
<cfpresentationslide title="myThirdSlide" src="slide3.swf" duration="2"
    presenter="Ram"/>

<!--- In the following slide, ColdFusion uses a video clip --->
<!--- instead of the JPEG image for the presenter. --->
<cfpresentationslide title="myFourthSlide" src="slide4.swf" duration="5"
    presenter="Shyam" video="video\video1.flv"/>
</cfpresentation>
```

cfprint

説明

PDF ファイル内の指定されたページを印刷します。このタグを使用すると、バッチ印刷ジョブを自動化できます。cfprint タグでは、cfdocument、cfpdf、および cfpdfform タグによって生成されたドキュメントを含め、任意の PDF ドキュメントを印刷できます。また、PDF 形式でエクスポートされた Report Builder レポートも印刷できます。

履歴

ColdFusion 8: このタグが追加されました。

カテゴリ

データ出力タグ

シンタックス

```
<cfprint
    source = "absolute or relative pathname to a PDF file|PDF document variable"
    attributeStruct = "ColdFusion structure that contains standard print request
        key-value pairs"
    color = "yes|no"
    copies = "number of copies"
    fidelity = "yes|no"
    pages = "page or pages to print"
    password = "PDF source file owner or user password"
    paper = "letter|legal|A4|A5|B4|B5|B4-JIS|B5-JIS|any media supported by the printer"
    printer = "string that specifies the printer name"
    type = "PDF">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfdocument、cfpdf、cfpdfform、cfpdfformparam、cfpdfparam、cfpdfsubform、GetPrinterInfo、IsPDFFile、IsPDFObject

属性

属性	必須 / オプション	デフォルト	説明
attributeStruct	オプション		追加の印刷命令を指定するための ColdFusion 構造体です。個別に名前が付けられた属性は、この属性構造体に含まれるキーと値のペアよりも優先されます。キーと値のペアについては、attributeStruct の表を参照してください。
color	オプション		カラーで印刷するかモノクロで印刷するかを指定します。 • yes: カラーで印刷します。 • no: モノクロ (グレースケール) で印刷します。
copies	オプション		印刷部数です。1 以上の値を指定する必要があります。
fidelity	オプション	no	指定した条件に基づいてジョブを印刷するかどうかを指定します。有効な値は、次のとおりです。 • yes: 指定した条件どおりに印刷できない場合、そのジョブは拒否されます。 • no: 条件どおりに印刷できない場合でも、妥当であればジョブを印刷します。

属性	必須 / オプション	デフォルト	説明
pages	オプション	all	ソースファイル内で印刷するページです。ドキュメントの最初のページと最後のページの間にページが存在する場合、重複するページ番号と総ページ数を超えるページ番号は無視されます。個々のページ番号とページ範囲を組み合わせることができます (例 : 1-3,6,10-20)。pages 属性で値を指定しない場合は、ドキュメント全体が印刷されます。
paper	オプション		印刷ジョブに使用する用紙です。GetPrinterInfo 関数によって返されるすべての値を使用できます。使用できる値は次のとおりです。 • na-letter • na-legal • iso-a4 • iso-a5 • iso-b4 • iso-b5 • jis-b4 • jis-b5 詳細については、「サポートされる用紙タイプ」を参照してください。
password	オプション		PDF ソースファイルのオーナーパスワードまたはユーザーパスワードです。パスワードで保護されている PDF ファイルを印刷するには、この属性を指定します。
printer	オプション		プリンタの名前です。Windows の場合は、¥¥s1001prn02¥NTN-2W-HP_BW02 のように指定します。デフォルトの名前は、ColdFusion サーバーを実行しているアカウントのデフォルトプリンタです。プリンタ名の大文字と小文字は区別されるため、ColdFusion Administrator の [システム情報] ページに表示されるとおりに正確に入力する必要があります。詳細については、「使用方法」を参照してください。
source	必須		印刷するソースドキュメントです。次のいずれかを指定します。 • ディスク上またはメモリ内の PDF ドキュメントファイルへの絶対パス名または相対パス名 (例 : c:¥work¥myPDF.pdf または myPDF.pdf)。デフォルトのディレクトリは template ディレクトリです。 • cfdocument タグまたは cfpdf タグによってメモリ内に生成される PDF ドキュメント変数 (例 : "myPDFdoc")。
type	オプション	PDF	印刷するドキュメントのファイルタイプです。有効な値は PDF のみです。

使用方法

cfprint タグを使用すると、PDF ドキュメントのバッチ印刷を自動化できます。たとえば、PDF 形式のレポートを生成するバッチジョブを毎晩実行して、翌朝ユーザーが確認できるようにレポート全体または特定のページのみを自動的に印刷できます。

cfprint タグの大半の属性はプリンタに依存します。指定した属性がプリンタでサポートされていない場合、その命令は無視されます。また、属性のデフォルト設定もプリンタに依存します。デフォルトプリンタを設定する場合は、PDF ファイルソースのみを指定します (ファイルがパスワードで保護されている場合はパスワードも指定します)。

注意 : サポートされるプリンタ属性は、オペレーティングシステム、ネットワークプリンタサーバー、およびプリンタによって異なります。cfprint タグは Java Print Service (JPS) に依存します。JPS からアクセスできない属性をサポートしているプリンタもあります。たとえば、JDK 1.5 を実行している Macintosh OSX の JPS ではサポートされないプリンタ属性が数多くあります。JDK 1.6 にアップグレードすると一部の機能が追加されますが、仕上げ関連の属性はサポートされません。

fidelity 属性を yes に設定した場合、指定したいずれかの属性がプリンタでサポートされていなければ、ジョブは印刷されません。fidelity 属性を no に設定した場合は、プリンタで印刷ジョブが受け付けられ、サポートされていない属性が無視されるか、その属性に対して妥当な別の値に置き換えられます。

指定したプリンタでサポートされる属性を調べるには、GetPrinterInfo 関数を使用します。

サポートされる用紙タイプ

cfdocument タグでサポートされるページタイプに相当する用紙を使用できますが、これらのタイプは GetPrinterInfo 関数では返されません。

cfdocument	cfprint
• letter	• na-letter
• legal	• na-legal
• A4	• iso-a4
• A5	• iso-a5
• B4	• iso-b4
• B5	• iso-b5
• B4-JIS	• jis-b4
• B5-JIS	• jis-b5

設定済みプリンタのリストを表示するには

1. ColdFusion Administrator にログオンします。
2. Administrator のコンソールウィンドウの右上にある [システム情報] アイコンをクリックします。(このアイコンには "i" の文字が書かれています)。
3. [システム情報] ページの下部までスクロールします。[プリンタの詳細] の下に [デフォルトプリンタ] があります。デフォルトプリンタの下に [プリンタ] があり、デフォルトプリンタを含め ColdFusion から利用できる設定済みのプリンタが表示されます。

プリンタの設定はオペレーティングシステムに依存します。プリンタの設定は ColdFusion の外部で行います。

印刷ログを表示するには

1. ColdFusion Administrator にログオンします。
2. [デバッグとロギング] トピックを展開します。
3. [ログファイル] リンクをクリックします。ログファイルのリストに print.log ファイルが表示されます。

印刷に関連するアクセス許可

暗号化されている PDF ファイルを印刷するには、ファイルのアクセス許可を AllowPrinting に設定するか、オーナーパスワードを指定します。PDF ファイルのアクセス許可とパスワードを設定するには、cfpdf タグの protect アクションを使用します。詳細については、479 ページの「cfpdf」の「PDF ドキュメントのアクセス許可」を参照してください。

Security Manager がインストールされている場合、印刷ジョブリクエストを開始するには coldfusion.policy ファイルで次のアクセス許可が設定されている必要があります。

```
grant { permission java.lang.RuntimePermission "queuePrintJob"; };
```

Windows システムでは、ColdFusion サーバーを実行するアカウントに対して、使用するプリンタごとに PRINTER_ACCESS_USE アクセス権を設定する必要があります。ColdFusion を実行するアカウントに適切なアクセス許可を割り当てないと、システム上でローカルに設定されているプリンタを利用できません。

注意：デフォルトでは、ColdFusion はローカルシステムアカウントで実行されるため、プリンタキューにアクセスできません。特定のユーザーとして ColdFusion を実行する方法については、テクニカルノート (http://www.adobe.com/cfusion/knowledgebase/index.cfm?id=tn_17279) を参照してください。

attributeStruct

プリンタリクエストを指定するために使用するオプションの attributeStruct のキーと値のペアを次の表に示します。

要素	説明
autoRotateAndCenter	プリンタ属性で指定された向きと一致するようにドキュメントの向きを調整し、イメージ領域の中央にページを配置します。 • yes: 向きが指定されている場合は無視します (デフォルト)。 • no: 向きが指定されている場合は、その向きをドキュメントに適用します。
collate または sheetCollate	copies 属性でドキュメントの印刷部数が 2 部以上に設定されている場合に、1 部ごとにページ順に並べて印刷するかどうかを指定します。 • yes • no
color または chromaticity	カラーで印刷するかモノクロで印刷するかを指定します。モノクロ印刷の場合はグレースケールで表示されます。 • yes: カラーで印刷します。 • no: モノクロで印刷します。
copies	ソースドキュメントの印刷部数です。有効な値は 1 以上の整数です。
coverPage または jobSheets	ジョブで印刷するジョブ開始シートと終了シート (存在する場合) を指定します。 • none • standard
fidelity	指定した条件に基づいてジョブを印刷するかどうかを指定します。使用できる値は次のとおりです。 • yes: 指定した条件どおりに印刷できない場合、そのジョブは拒否されます。 • no: 条件どおりに印刷できない場合でも、妥当であればジョブを印刷します (デフォルト)。

要素	説明
finishings	ドキュメントを 1 部印刷するたびに実行する仕上げ処理です。 • staple-top-left • staple-bottom-left • staple-top-right • staple-bottom-right • edge-stitch-left • edge-stitch-right • edge-stitch-top • edge-stitch-bottom • dual-right • dual-top • dual-bottom • dual-left
jobHoldUntil	ジョブの印刷が可能になる日時を表す日付時刻属性です。有効な値は ColdFusion の日付時刻変数です。
jobName	印刷ジョブの名前です。
jobPriority	印刷ジョブの優先度を表す整数値です。印刷を開始できるジョブの中で、優先度の値が **n** に設定されているジョブがすべて印刷されると、**n**-1 のジョブが印刷されます。有効な値は 1 (優先度最低) ～ 100 (優先度最高) の整数です。
numberUp	用紙の片面に印刷するページ数です。1 以上の値を指定する必要があります。
orientation または orientationRequested	ページの印刷向きです。PDF ドキュメントの場合、有効な値は portrait のみです。印刷向きを横に変更するには、autoRotateAndCenter を yes に設定します (これがデフォルト値です)。autoRotateAndCenter 命令は orientation 命令よりも優先されます。
pages	ソースファイル内で印刷するページです。ドキュメントの最初のページと最後のページの間にページが存在する場合、重複するページ番号と総ページ数を超えるページ番号は無視されます。個々のページ番号とページ範囲を組み合わせることができます (例 : 1-3,6,10-20)。pages 属性で値を指定しない場合は、ドキュメント全体が印刷されます。
pageScaling	用紙上でページを拡大または縮小する方法を指定します。 • fit-to-printer-margins: 現在選択されている用紙サイズの印刷可能領域に合わせて、各ページが縮小または拡大されます。 • reduce-to-printer-margins: 大きいページは、現在選択されている用紙サイズに合わせて縮小されますが、小さいページは拡大されません。印刷領域が選択されていて、その領域が現在選択されている用紙の印刷可能領域より大きい場合は、印刷可能領域に合わせてページを縮小します (デフォルト)。 • none: 拡大または縮小を行わずに、ページの左上または中央部分を印刷します (autoRotateAndCenter が設定されている場合)。用紙に入りきらないページは自動的にトリミングされます。
pageSubset	pages 属性で指定したページのサブセットを印刷します。 • all: 指定したページ範囲のすべてのページを印刷します (デフォルト)。 • odd: 指定したページ範囲の奇数ページのみを印刷します。 • even: 指定したページ範囲の偶数ページのみを印刷します。

要素	説明
paper	印刷ジョブに使用する用紙です。GetPrinterInfo 関数によって返されるすべての値を使用できます。最も一般的な値は次のとおりです。 • na-letter • iso-a4
presentationDirection	numberUp 属性とともに使用して、複数のドキュメントページを用紙の片面に印刷する場合のレイアウトを指定します。
printer	プリンタの名前です。Windows の場合は、¥¥s1001prn02¥NTN-2W-HP_BW02 のように指定します。デフォルトの名前は、ColdFusion サーバーを実行しているアカウントのデフォルトプリンタです。プリンタ名の大文字と小文字は区別されるため、ColdFusion Administrator の [システム情報] ページに表示されるとおりに正確に入力する必要があります。印刷ログを表示する方法については、「使用方法」を参照してください。
quality	印刷ジョブの印刷品質です。 • draft • high • normal
requestingUserName	印刷ジョブを送信したエンドユーザーの名前を指定する文字列です。
reversePages	ページを逆順で印刷します。ページ範囲が指定されている場合は、入力された順序と逆に印刷されます。たとえば、[ページ] ボックスに「3-5, 7-10」と入力し、逆順印刷を選択した場合は、10 ページから 7 ページまでが印刷された後、5 ページから 3 ページまでが印刷されます。 • yes • no (デフォルト)
sides	ページを片面に印刷するか両面に印刷するかを設定します。 • one-sided (デフォルト) • duplex または two-sided-long-edge • tumble または two-sided-short-edge
usePdfPageSize	用紙のサイズではなく PDF ページのサイズを使用して、用紙の印刷領域を決定します。これは、サイズの異なるページを含む PDF ドキュメントを印刷する場合に便利です。 • yes • no (デフォルト)

例

次の例は、attributeStruct 属性と cfprint タグを使用して、レターサイズの PDF ドキュメントを両面、丁合いの設定で 5 部印刷し、左上隅を綴じる方法を示しています。

```
<cfset aset=StructNew()>
<cfset aset["sides"] = "duplex">
<cfprint type="pdf" source="myfile.pdf"
    printer="\\s1001prn02\NTN-2W-HP_BW02" copies="5" paper="letter"
    attributeStruct="#aset#">
```

次の例は、attributeStruct 属性を使用してすべての印刷属性を指定する方法を示しています。

```
<cfset aset=StructNew()>
<cfset aset["paper"] = "letter">
<cfset aset["sides"] = "duplex">
<cfset aset["copies"] = "5">
<cfset aset["printer"] = "\\s1001prn02\NTN-2W-HP_BW02">

<cfprint type="pdf" source="myfile.pdf" attributeStruct="#aset#">
```

プリンタには autoRotateAndCenter という設定があり、これはデフォルトで yes に設定されています。次の例は、デフォルトの autoRotatateAndCenter 設定の代わりに orientation 設定を使用する方法を示しています。

```
<cfset aset=StructNew()>
    <cfset aset["autoRotateAndCenter"] = "no">
    <cfset aset["orientation"] = "portrait">

    <cfprint printer="myprinter" source="_mydoc.pdf" attributeStruct="#aset#">
```

印刷ジョブを非同期で実行するには、スレッド内で印刷ジョブを開始します。次の処理に進む前に、印刷ジョブがプリンタに送信されるまで待機しないようにしてください。スレッド内で印刷ジョブを開始するには、次の例のように、cfthread の開始タグと終了タグの間に cfprint タグを挿入します。

```
<cfthread name="mythread" action="run">
    <cfprint type="pdf" source="myfile.pdf" printer="\\s1001prn02\NTN-2W-HP_BW02">
</cfthread>
....
```

詳細については、654 ページの「cfthread」を参照してください。

cfprocessingdirective

説明

現在のページの処理方法に関して ColdFusion に次の情報を与えます。

- ColdFusion によってタグ本文に生成された内容から余分な空白文字を削除するかどうかを指定します。
- ページ内容の文字エンコード (文字セット) を指定します。

カテゴリ

データ出力タグ、ページ処理タグ

シンタックス

```
<cfprocessingdirective
    pageencoding = "page-encoding literal string"/>

OR

<cfprocessingdirective
    pageEncoding = "page-encoding literal string"
    suppressWhiteSpace = "yes|no">
    CFML tags
</cfprocessingdirective>
```

関連項目

cfcol、cfcontent、cfoutput、cfsetting、cfsilent、cftable、『ColdFusion アプリケーションの開発』の Developing Globalized Applications

履歴

ColdFusion MX:

- suppresswhitespace 属性値の動作が変更されました。suppresswhitespace 属性の値を文字列変数で指定できるようになりました (ColdFusion 5 では、この属性の値は定数でしか設定できませんでした)。
- pageEncoding 属性が追加されました。

属性

属性	必須 / オプション	デフォルト	説明
pageEncoding	オプション	文字エンコードはページの BOM (Byte Order Mark) によって識別され、BOM がない場合はシステムのデフォルトのエンコードが使用される	文字列リテラルです。変数は使用できません。現在の CFML ページの文字エンコードを指定します。この属性は、cfprocessing タグ本文だけでなく、ページ全体に適用されます。この値を一重引用符または二重引用符で囲むこともできますが、囲まなくても問題ありません。 一般的に使用される値を次に示します。 • utf-8 • iso-8859-1 • windows-1252 • us-ascii • shift_jis • iso-2022-jp • euc-jp • euc-kr • big5 • euc-cn • utf-16 文字エンコードの詳細については、www.w3.org/International/O-charset.html を参照してください。
suppressWhiteSpace	オプション		ブール値です。cfprocessingdirective ブロック内の空白文字を削除するかどうかを指定します。空白文字は CFML タグによって生成され、HTML の外観にはほとんど影響しません。この設定は、HTML コード内の空白には適用されません。

使用方法

cfprocssingdirective タグには、使用する属性に依存する制限があります。このため、cfprocessingdirective タグでは、pageencoding 属性または suppresswhitespace 属性のいずれか一方だけを使用することをお勧めします。両方の値を指定する場合は、別々のタグを使用します。

ColdFusion コンポーネント (.cfc ファイル) では、cfprocessingdirective タグは cfcomponent タグの後に続く必要があります。

pageEncoding 属性を使用する場合は、次のルールが適用されます。

- このタグはページの最初の 4096 バイト内に配置する必要があります。cfsetting または cfsilent タグの後に配置できます。
- cfinclude タグや cfmodule タグ、カスタムタグの呼び出しなどを使って他のページをインクルードしているページでこのタグを使用する場合、インクルードしたページにはこのタグの設定は適用されません。

- pageEncoding 属性は、ページの実行時ではなくコンパイル時に評価されるため、このタグを条件論理式に埋め込むことはできません。たとえば、次のコードでは cfprocessingdirective タグは既に評価されているため、実行時に何の影響も与えません。

```
<cfif dynEncoding is not "dynamic encoding is not possible">
    <cfprocessingdirective pageencoding=#dynEncoding#>
</cfif>
```

- pageEncoding 属性を指定する cfprocessingdirective タグが 1 ページ内に複数ある場合は、指定する値をすべて同じにする必要があります。値が異なる場合は、エラーが発生します。
- pageencoding 属性のみを指定する場合は、単独の終了タグを使用しないでください。
- ColdFusion では、Java プラットフォームでサポートされる文字エンコード名を使用できます。無効な名前が指定された場合は、InvalidEncodingSpecification 例外が発生します。
- ページに BOM があり、pageencoding 属性で指定したエンコードが BOM と異なる場合は、エラーが発生します。

suppressWhiteSpace 属性を使用する場合は、次のルールが適用されます。

- suppresswhitespace 属性値は、定数または変数として指定することができます。変数を使用するには、変数 (たとえば、whitespaceSetting) を定義し、これに値 yes または no を割り当て、次のようなステートメントを記述します。

```
<!--- ColdFusion allows suppression option to be set at runtime --->
<cfprocessingdirective suppresswhitespace=#whitespaceSetting#>
code to whose output the setting is applied
</cfprocessingdirective>
```

- suppresswhitespace 属性は、<cfprocessingdirective> 開始タグと </cfprocessingdirective> 終了タグの間に配置されたコードにのみ適用されます。

次の例では、ネストされた cfprocessingdirective タグの使い方を示します。外側のタグは、大きなテーブルの計算時に不要な空白を削除します。内側のタグは空白を保持して、書式設定済みのテーブルを出力します。

例

```
<cfprocessingdirective suppressWhiteSpace = "Yes">
    <!--- CFML code --->
    <cfprocessingdirective suppressWhiteSpace = "No">
        <cfoutput>#table_data#
        </cfoutput>
    </cfprocessingdirective>
</cfprocessingdirective>
```

次の例では、pageencoding 属性の使い方を示します。

```
<cfprocessingdirective pageencoding = "shift_jis">
```

cfprocparam

説明

ストアドプロシージャのパラメータを定義します。このタグは、cfstoredproc タグ内にネストします。

カテゴリ

データベース操作タグ

シンタックス

```
<cfprocparam
    CFSQLType = "parameter data type"
    maxLength = "length"
    null = "yes|no"
    scale = "decimal places"
    type = "in|out|inout"
    value = "parameter value"
    variable = "variable name">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfinsert、cfprocresult、cfquery、cfqueryparam、cfstoredproc、cftransaction、cfupdate、『ColdFusion アプリケーションの開発』の Designing and Optimizing a ColdFusion Application の Optimizing ColdFusion applications

履歴

ColdFusion MX:

- maxrows 属性は廃止されました。
- dbvarname 属性の動作が変更されました。すべてのドライバで無視されるようになりました。ColdFusion では JDBC 2.2 が使用され、名前付きパラメータはサポートされません。これは非推奨になりました。
- maxLength 属性の動作が変更されました。IN および INOUT パラメータ値に適用されるようになりました。

属性

属性	必須 / オプション	デフォルト	説明
CFSQLType	必須		任意の型のパラメータをバインドする SQL 型です。ColdFusion では次の値がサポートされます。名前の末尾の要素は SQL データ型に対応しています。データベースシステムによっては、このリストの別のサブセットがサポートされる場合があります。サポートされているパラメータ型については、ご使用の DBMS のドキュメントを参照してください。 • CF_SQL_BIGINT • CF_SQL_BIT • CF_SQL_BLOB • CF_SQL_CHAR • CF_SQL_CLOB • CF_SQL_DATE • CF_SQL_DECIMAL • CF_SQL_DOUBLE • CF_SQL_FLOAT • CF_SQL_IDSTAMP • CF_SQL_INTEGER • CF_SQL_LONGVARCHAR • CF_SQL_MONEY • CF_SQL_MONEY4 • CF_SQL_NUMERIC • CF_SQL_REAL • CF_SQL_REFCURSOR • CF_SQL_SMALLINT • CF_SQL_TIME • CF_SQL_TIMESTAMP • CF_SQL_TINYINT • CF_SQL_VARCHAR ColdFusion SQL データ型から JDBC データ型へのマッピングについては、cfqueryparam を参照してください。
maxLength	オプション	0	IN または INOUT の場合での、value 属性の文字列の最大長です。maxLength を 0 に設定すると、長さは任意になります。type=out を指定するときは、maxLength 属性は不要です。
null	オプション	no	パラメータを null 値として渡すかどうかを指定します。OUT タイプのパラメータでは使用しません。 • yes: value 属性を無視します。 • no

属性	必須 / オプション	デフォルト	説明
scale	オプション	0	数値パラメータの小数点以下の桁数です。scale が 0 の場合、値は整数に制限されます。
type	オプション	in	• in: データベースシステムにデータを送信するときにのみ使用します。値でパラメータを渡します。 • out: データベースシステムからデータを受信するときにのみ使用します。パラメータをバインド変数として渡します。 • inout: データを送信および受信するときに使用します。パラメータをバインド変数として渡します。
value	type="IN" の場合は必須		ColdFusion からストアドプロシージャに渡す値です。inout パラメータの場合はオプションです。
variable	type="OUT" または "INOUT" の場合は必須		ColdFusion 変数名です。ストアドプロシージャが呼び出された後、出力パラメータに与えられる値を参照します。in パラメータの場合は無視されます。

使用方法

このタグを使用すると、ストアドプロシージャのパラメータとそのデータ型を指定できます。パラメータごとに 1 つの cfprocparam タグをコーディングします。コーディングするパラメータは、パラメータの種類と DBMS によって異なります。ColdFusion では、定位置パラメーターがサポートされます。定位置パラメータを使用する場合は、ストアドプロシージャ定義の中で出現するパラメータと同じ順序で cfprocparam タグをコーディングする必要があります。

出力変数は、variable 属性で指定された ColdFusion 変数に格納されます。

cfprocparam タグは、Oracle 8 および 9 の Reference Cursor には使用できません。代わりに、cfprocresult タグを使用します。

例

次の例では、データベースにデータを挿入する Oracle および Microsoft SQL Server のストアドプロシージャを示します。これらのストアドプロシージャは同等です。いずれのストアドプロシージャを呼び出す場合も CFML は同じです。

次の例では、Oracle のストアドプロシージャを示します。

```
CREATE OR REPLACE PROCEDURE Insert_Book (
    arg_Title Books.Title%type,
    arg_Price Books.Price%type,
    arg_PublishDate Books.PublishDate%type,
    arg_BookID OUT Books.BookID%type)
AS
    num_BookID NUMBER;
BEGIN
    SELECT seq_Books.NEXTVAL
    INTO num_BookID
    FROM DUAL;

    INSERT INTO
        Books (
            BookID,
            Title,
            Price,
            PublishDate )
    VALUES (
        num_BookID,
        arg_Title,
        arg_Price,
        arg_PublishDate );

    arg_BookID := num_BookID;
END;
/
```

次の例では、SQL Server のストアドプロシージャを示します。

```
CREATE PROCEDURE Insert_Book (
    @arg_Title VARCHAR(255),
    @arg_Price SMALLMONEY,
    @arg_PublishDate DATETIME,
    @arg_BookID INT OUT)
AS
BEGIN
INSERT INTO
    Books (
        Title,
        Price,
        PublishDate )
    VALUES (
        @arg_Title,
        @arg_Price,
        @arg_PublishDate );

    SELECT @arg_BookID = @@IDENTITY;
END;
```

いずれのストアドプロシージャを呼び出す場合も、次の CFML コードを使用します。

```
<cfset ds = "sqltst">
<!--- <cfset ds = "oratst"> --->

<!--- If submitting a new book, insert the record and display confirmation --->
<cfif isDefined("form.title")>
    <cfstoredproc procedure="Insert_Book" datasource="#ds#">
        <cfprocparam cfsqltype="cf_sql_varchar" value="#form.title#">
        <cfprocparam cfsqltype="cf_sql_numeric" value="#form.price#">
        <cfprocparam cfsqltype="cf_sql_date" value="#form.price#">
        <cfprocparam cfsqltype="cf_sql_numeric" type="out" variable="bookId">
    </cfstoredproc>

<cfoutput>
    <h3>'#form.title#' inserted into database.The ID is #bookId#.</h3>
</cfoutput>

</cfif>
<cfform action="#CGI.SCRIPT_NAME#" method="post">
    <h3>Insert a new book</h3>

    Title:
    <cfinput type="text" size="20" required="yes" name="title"/>
    <br/>

    Price:
    <cfinput type="text" size="20" required="yes" name="price" validate="float"/>
    <br/>

    Publish Date:
    <cfinput type="text" size="5" required="yes" name="publishDate" validate="date"/>
    <br/>

    <input type="submit" value="Insert Book"/>

</cfform>
```

cfprocresult

説明

ストアドプロシージャによって返される結果セットにクエリーオブジェクトを関連付けます。このクエリーオブジェクトは、他の ColdFusion タグ (cfoutput や cftable など) が結果セットにアクセスするときに使用されます。このタグは、cfstoredproc タグ内にネストします。

カテゴリ

データベース操作タグ

シンタックス

```
<cfprocresult
    name = "query name"
    maxRows = "number"
    resultSet = "1-n">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfinsert、cfprocparam、cfquery、cfqueryparam、cfstoredproc、cftransaction、cfupdate、『ColdFusion アプリケーションの開発』の Designing and Optimizing a ColdFusion Application の Optimizing database use

属性

属性	必須 / オプション	デフォルト	説明
name	必須		クエリーの結果セットの名前です。
maxRows	オプション	-1 (すべて)	結果セットに返される行の最大数です。
resultSet	オプション	1	ストアドプロシージャによって複数の結果セットが返される場合に、特定の結果セットの名前を指定します。

使用方法

ストアドプロシージャによって返されるデータにアクセスできるようにするには、cfprocresult タグを指定します。ストアドプロシージャによって複数の結果セットが返される場合は、resultSet 属性を使用して、ストアドプロシージャのどの結果セットを返すかを指定します。

resultSet 属性には、cfstoredproc タグのスコープ内で固有の値を指定する必要があります。結果セットを 2 回指定すると、最初の指定は 2 回目の指定によって上書きされます。

CFML では、パラメータをリファレンスによって渡す Oracle 8 および 9 の Reference Cursor タイプがサポートされます。この方法で渡されるパラメータは、1 つのアプリケーションの実行において、メモリを割り当てられたり、割り当て解除されたりします。パッケージやストアドプロシージャで Reference Cursor を使用するには、cfprocresult タグを使用します。このタグを使用すると、ColdFusion JDBC データベースドライバによって Oracle Reference Cursor が結果セットに組み込まれます。Oracle の ThinClient JDBC ドライバでは、この方法を使用できません。

例

```
<!--- This example executes a Sybase stored procedure that returns three result sets, two
    of which we want. The stored procedure returns status code and one output parameter, which
    we display. We use named notation for parameters. --->
<!--- cfstoredproc tag --->
<cfstoredproc procedure = "foo_proc"
    dataSource = "MY_SYBASE_TEST" username = "sa"
    password = "" dbServer = "scup" dbName = "pubs2"
    returnCode = "Yes" debug = "Yes">
    <!--- cfprocresult tags --->
    <cfprocresult name = RS1>
    <cfprocresult name = RS3 resultSet = 3>
    <!--- cfprocparam tags --->
    <cfprocparam type = "IN" CFSQLType = CF_SQL_INTEGER value = "1">
    <cfprocparam type = "OUT" CFSQLType = CF_SQL_DATE variable = FOO>
    <!--- Close the cfstoredproc tag. --->
</cfstoredproc>
<cfoutput>
    The output param value: '#foo#'<br>
</cfoutput>
<h3>The Results Information</h3>
<cfoutput query = RS1>#name#,#DATE_COL#<br>
</cfoutput>
<p></p>
<cfoutput>
    <hr>
    <p>Record Count: #RS1.recordCount# <p>Columns: #RS1.columnList#</p>
    <hr>
</cfoutput>
<cfoutput query = RS3>#col1#,#col2#,#col3#<br>
</cfoutput>
<p></p>
<cfoutput>
    <hr>
    <p>Record Count: #RS3.recordCount# <p>Columns: #RS3.columnList#</p>
    <hr>
    The return code for the stored procedure is:
    '#cfstoredproc.statusCode#'<br>
</cfoutput>
...
```

cfprogressbar

説明

進行状況表示バーを定義して、ファイルのダウンロードなどの動作の進行状況を示します。

カテゴリ

表示管理タグ

シンタックス

```
<cfprogressbar
    autoDisplay="true|false"
    name="control identifier"
    bind ="bind expression"
    duration="time value"
    height="height in pixels"
    interval="time in milliseconds"
    onComplete="function name"
    onError="JavaScript function name"
    style="style specification"
    width="pixel value">
```

履歴

ColdFusion 9: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
autoDisplay	オプション	true	プログレスバーを表示するには、true に設定します。
name	必須		コントロールの名前です。進行状況を開始するスクリプトなど、JavaScript 内でコントロールを参照するために使用します。
bind	duration が指定されていない場合は必須		クライアントの JavaScript 関数またはサーバーの CFC を指定するバインド式を示します。interval 属性で定義された時間が経過するごとに、進行状況の情報を取得するためにコントロールがこの関数を呼び出します。この属性は duration 属性と一緒には使用できません。 エラーが発生した場合、それ以降のバインド式呼び出しは実行されず、onError が呼び出されます (定義されている場合)。
duration	bind が指定されていない場合は必須		進行状況表示バーの開始時点から終了時点までの時間をミリ秒単位で示します。バインド式を使用して実際の進行状況の情報を取得しない自動の進行状況表示バーについてのみ使用します。この属性は bind 属性と一緒には使用できません。
height	オプション		バーの高さです (単位 : ピクセル)。
interval	オプション	1000	duration が指定されている場合に適用されます。進行状況表示バーが更新される時間間隔をミリ秒単位で示します。この属性はオプションですが、指定すると、表示バーの更新間隔を長くすることができます。
onComplete	オプション		進行状況が完了したときに呼び出す関数の名前です。
onError			bind を使用する場合にのみ適用されます。エラー状態が発生したときに実行する JavaScript 関数です。この場合のエラーとは、ネットワークエラーまたはサーバーサイドエラーとなります。
style	オプション		サポートされる色は次のとおりです。 • bgcolor: プログレスバーの背景色です。接頭辞 "#" が付かない 16 進数値です。 • textcolor: プログレスバーのテキスト色です。 • progresscolor: 進行状況の表示に使用される色です。
width	オプション	400	バーの幅 (長さ) をピクセル単位で示します。

使用方法

進行状況表示バーでは、次の 2 つの動作のうちいずれかを指定できます。

- 手動 : duration 属性で指定された時間の経過とともに、進行状況を示すインジケータの長さが徐々に増加します。
- 動的 : インジケータの長さを決定する関数を bind 属性で指定します。

バインド式を使用する場合、呼び出される関数は、パラメータを取らず、次の 2 つの値が含まれた構造体を返す必要があります。

- status - 小数で表された完了値 (範囲 : 0 ～ 1.0)
- message - 進行状況表示バーで表示するメッセージ (例 : " ロード中 ..."、" 完了 ")

次の 2 つの Ajax 関数を使用して、進行状況表示バーを開始および終了します。

```
ColdFusion.ProgressBar.start(barName)
ColdFusion.ProgressBar.stop(barName)
```

進行状況表示バーを開始するには、start メソッドを呼び出す必要があります。

進行状況表示バーを明示的に終了するには、stop メソッドを呼び出します。バインドメソッドで status 値 1 が返されるか、または duration 属性で指定された時間が経過すると、バーは自動的に終了します。そのため、このメソッドを使用する必要があるのは、プロセスが完了していない場合、duration 時間の前にプロセスが完了する場合、またはエラー状態などの正常でない状況が発生した場合に限られます。

例

次の例のような簡単なコメントフォームによって、タイマーを使用してフォームを処理するのにかかる時間をシミュレートすることができます。

Application.cfc ページで、セッション管理を有効にしている必要があります。

アプリケーションのメインページには、次のコードが含まれています。

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Untitled Document</title>
</head>

<script type="text/javascript">
// The function that starts the progress bar,
// called when the user clicks the Send comment button.
function startProgress() {
         ColdFusion.ProgressBar.start("mydataProgressbar");
       };

// The function called when the progress bar finishes,
// specified by the cfprogressbar tag onComplete attribute.
function onfinish() {
     alert("Done");
     };
</script>

<body>
<!--- Ensure there is no Session.STATUS value, which is used by
    the progress bar bind CFC, when the page displays. --->
<cfif IsDefined("Session.STATUS")>
    <cfscript>
        StructDelete(Session,"STATUS");
    </cfscript>
</cfif>

<!--- For code simplicity, formatting is minimal. --->
<cfform name="kitform">
    <p>To make our service better and to benefit from our special offers,
      take a moment to give us your email address and send us a comment.</p>
    <p>Name:<br />
```

```
         <cfinput type="text" name="name">  </p>
    <p>E-mail:<br />
         <cfinput type="text"  name="email">  </p>
    <p>Comment:<br />
         <cftextarea   name="cmnt"/></p>
    <p>       
            <cfinput type="button" name="" value="Send Comment"
        onClick=startProgress()></p>
    <!--- The progressbar control --->
    <div style="padding-left:3px" >
            <cfprogressbar name="mydataProgressbar"
            bind="cfc:mycfc.getstatus()"
            interval="1700"
            width="200"
        oncomplete="onfinish"/>
    </div>

</cfform>
</body>
</html>
```

mycfc.cfc ファイルには、getstatus 関数が 1 つ含まれています。

```
<cfcomponent>
<!--- This function simulates the time taken
    by and operation by using sleep functions.
    It increments the progressbar status by 1/10 each time the
    progressbar bind expression calls it (that is, each time the
    time specified by the cfprogressbar interval attribute passes.
--->
    <cffunction name="getstatus" access="remote">
            <cfset str = StructNew()>
            <cfset str.message = "Saving Data">
            <cfif  NOT IsDefined("session.STATUS")>
                <cfset session.STATUS = 0.1>
                <cfscript>
                    Sleep(200);
                </cfscript>
            <cfelseif session.STATUS LT 0.9>
                <cfset session.STATUS=session.STATUS + .1>
                <cfscript>
                    Sleep(200);
                </cfscript>
            <cfelse>
                <cfset str.message = "Done...">
                <cfset session.STATUS="1.0">
            </cfif>
            <cfset str.status = session.STATUS>
            <cfreturn str>
    </cffunction>

</cfcomponent>
```

cfproperty

説明

ColdFusion コンポーネント (CFC) のプロパティおよびその注釈を定義します。プロパティは、Web サービスの複合データ型を作成するために使用されます。注釈は、CFC 用の ORM (Object Relational Model: オブジェクトリレーショナルモデル) を定義するために使用されます。このタグの属性は、コンポーネントのメタデータとして示され、継承ルールが適用されます。

カテゴリ
拡張タグ

シンタックス
```
<cfproperty
    name="name"
    default="default value"
    displayname="descriptive name"
    hint="extended description"
    required="false|true"
    serializable="true|false"
    type="type">
```

注意：ORM 関連の属性とその使用方法については、『ColdFusion アプリケーションの開発』の Map the properties を参照してください。

関連項目
cfargument、cfcomponent、cffunction、cfinvoke、cfinvokeargument、cfobject、cfreturn、『ColdFusion アプリケーションの開発』の Building and Using ColdFusion Components の Documenting CFCs、Implicit Get and Set Functions

履歴
ColdFusion 9: CFC 用のオブジェクトリレーショナルモデルを定義するための属性が追加されました。

暗黙的な getter および setter が追加されました。validate 属性と validateparams 属性が追加されました。

ColdFusion MX: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
batchsize			これらの属性の詳細については、ColdFusion ORM を参照してください。
cascade			
catalog			
cfc			
collectiontype			
column			
constrained			
datatype			
default	オプション		コンポーネントを Web サービスで使用するときにプロパティ値が設定されていない場合、デフォルト値が指定されます。 この属性を指定する場合は、required 属性を no に設定します。あるいは、required 属性を指定しないでください。 default 属性の ORM 固有の使用方法については、ColdFusion ORM を参照してください。
displayname	オプション		イントロスペクションを使用して CFC についての情報を示すときに表示される値です。この値はプロパティ名の後ろの括弧内に表示されます。

属性	必須 / オプション	デフォルト	説明
dynamicInsert			これらの属性の詳細については、ColdFusion ORM を参照してください。
dynamicupdate			
elementColumn			
elementtype			
entityname			
fetchbatchsize			
fieldType			
fkcolumn			
formula			
generator			
getter	オプション		getter メソッドを生成するかどうかを指定します。有効な値は次のとおりです。 • true • false
hint	オプション		イントロスペクションを使用して CFC についての情報を示すときに表示されるテキストです。この属性は、パラメータの目的を説明するのに役立ちます。
index			これらの属性の詳細については、ColdFusion ORM を参照してください。
insert			
inverse			
inversejoincolumn			
joincolumn			
lazy			
length			
linkcatalog			
linkschema			
linktable			
mappedby			
missingrowIgnored			
name	必須		文字列です。プロパティ名を指定します。スタティックな値である必要があります。
notnull			これらの属性の詳細については、ColdFusion ORM を参照してください。
optimisticLock			
optimisticLockgenerated			
orderby			
orderByreadonly			

属性	必須 / オプション	デフォルト	説明
params			これらの属性の詳細については、ColdFusion ORM を参照してください。
persistent			
preceision			
readonly			
readonly			
required	オプション	no	パラメータが必須かどうかを指定します。 • yes • no
rowid			これらの属性の詳細については、ColdFusion ORM を参照してください。
scale			
`setter`			setter メソッドを生成するかどうかを指定します。有効な値は次のとおりです。 • true • false
schema			これらの属性の詳細については、ColdFusion ORM を参照してください。
selectbeforeupdate			
selectkey			
sequence			
serializable	オプション	true	このプロパティがシリアル化可能かどうかを指定します。この値を false に設定した場合、プロパティはシリアル化できないため、加えられた変更はセッションレプリケーションで保持されず、2 番目のサーバーではデフォルト値 (存在する場合) が使用されます。CFC でシリアル化可能なプロパティのシリアル化を回避するには、この属性を使用します。
source			これらの属性の詳細については、ColdFusion ORM を参照してください。
structkeycolumn			
structkeycolumn			
structkeydatatype			
structkeyType			
table			
table			

属性	必須 / オプション	デフォルト	説明
type	オプション	any	文字列です。プロパティのデータ型を指定します。 • any • array • binary • boolean • date • guid: この引数は xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx という形式の UUID または GUID でなければなりません。x は 16 進数の 1 文字 (0 ～ 9、A ～ F) を表します。 • numeric • query • string • struct • uuid: この引数は xxxxxxxx-xxxx-xxxx-xxxxxxxxxxxxxxxx という形式の ColdFusion UUID でなければなりません。x は 16 進数の 1 文字を表します (0 ～ 9、A ～ F)。 • variableName: ColdFusion 変数のネーミング規則に従った形式の文字列です。 • コンポーネント名 : type 属性の値が上記のどれにも当てはまらない場合、ColdFusion はそれを ColdFusion コンポーネントの名前として扱います。関数を実行したときに、渡された引数が指定の名前を持つ CFC でない場合はエラーになります。
unique			これらの属性の詳細については、ColdFusion ORM を参照してください。
uniquekey			
update			
where			
validate	オプション		詳細については、Validate and validateparams attributes を参照してください。
validateparam	オプション		

使用方法

cfproperty タグは、コンポーネントの最初の、実行可能コードと関数定義よりも上に配置する必要があります。

コンポーネントが Web サービスとして使用されない場合、<cfproperty> では、プロパティのメタデータ情報のみが指定されます。コンポーネントで使用できる変数の定義や値の設定は行われません。ただし、getter 属性または setter 属性が有効化されているかどうかに応じて、CFC のプロパティの暗黙的な setter および getter が作成されます。詳細については、『ColdFusion アプリケーションの開発』の Implicit Get and Set Functions を参照してください。

オブジェクトリレーショナルモデル (ORM) については、CFC のプロパティのリレーショナルマッピングを定義するために cfproperty が使用されます。詳細については、『ColdFusion アプリケーションの開発』の ColdFusion ORM を参照してください。

ColdFusion で作成する Web サービスについては、Web サービスで使用する複合変数が cfproperty タグによって定義されます。

例
たとえば、住所を表すプロパティを持つコンポーネントが、"address.cfc" ファイルで次のように定義されているとします。

```
<cfcomponent>
    <cfproperty name="Number" type="numeric">
    <cfproperty name="Street" type="string">
    <cfproperty name="City" type="string">
    <cfproperty name="State" type="string">
    <cfproperty name="Country" type="string">
</cfcomponent>
```

このコンポーネントで表される複合データ型は、次のように Web サービスとしてエクスポートされるコンポーネントで使用できます。

```
<cfcomponent>
    <cffunction name="echoAddress" returnType="address" access="remote">
        <cfargument name="input" type="address">
        <cfreturn arguments.input>
    </cffunction>
</cfcomponent>
```

cfquery

説明
クエリーまたは SQL ステートメントをデータソースに渡します。

各 cfquery タグ内で cfqueryparam タグを使用して、権限のないユーザーがデータベースにアクセスできないようにしてください。詳細については、セキュリティ情報 ASB99-04「Multiple SQL Statements in Dynamic Queries」(Security Zone、www.adobe.com/go/sn_asb99-04_jp) および『ColdFusion アプリケーションの開発』の Accessing and Retrieving Data を参照してください。

カテゴリ
データベース操作タグ

シンタックス

```
<cfquery
    name = "query name"
    blockFactor = "block size"
    cachedAfter = "date"
    cacheID = "ID"
    cacheRegion = "region"
    cachedWithin = "timespan"
    dataSource = "data source name"
    dbtype = "query"
    debug = "yes|no"
    fetchClientInfo = "yes|no"
    maxRows = "number"
    ormoptions = #orm options structure#
    password = "password"
    result = "result name"
    timeout = "seconds"
    username = "user name">
</cfquery>
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfdbinfo、cfinsert、cfprocparam、cfprocresult、cfqueryparam、cfstoredproc、cftransaction、cfupdate、『ColdFusion アプリケーションの開発』の Optimizing database use

履歴

ColdFusion 10：fetchClientInfo、cacheID、cacheRegion、clientInfo の各属性が追加されました。

ColdFusion 9.0.1：HQL クエリのサポートが導入され、ormoptions 属性が追加されました。

ColdFusion 9: データソース属性はオプションになりました。

ColdFusion 8: 行の ID を指定する結果変数が追加されました。

ColdFusion MX 7:

- 結果変数を保持する構造体の別名を指定するための result 属性が追加されました。
- 実行された SQL ステートメントの結果変数 (sql)、返されたレコードの数 (recordcount)、クエリーがキャッシュされたかどうか (cached)、cfqueryparam 値の配列 (sqlparameters)、および返されたクエリー内の列のリスト (columnlist) が追加されました。

ColdFusion MX:

- クエリーオブクエリーの動作が変更されました。サポートされる標準 SQL のサブセットが拡張されました。
- ドット表記法のサポートが変更されました。レコードセット名でのドット表記法をサポートするようになりました。レコードセット名は構造体として解釈されます。
- connectString、dbName、dbServer、provider、providerDSN、および sql の各属性、および query を除く dbtype 属性のすべての値が非推奨となりました。ColdFusion 5 以降のリリースでは、これらは機能せず、エラーを引き起こす可能性があります。
- 新しいクエリーオブジェクト変数 cfquery.ExecutionTime を使用できるようになりました。
- ネイティブドライバはサポートされなくなりました。データベース接続には JDBC (および ODBC-JDBC ブリッジ) を使用します。

属性

属性	必須 / オプション	デフォルト	説明
name	必須		クエリー名です。クエリーのレコードセットを参照するため、ページ内で使用します。文字で始める必要があります。文字、数字、およびアンダースコアを使用できます。
blockFactor	オプション	1	サーバーから同時に取得する行の最大数です。1 ～ 100 の範囲で指定します。一部のデータベースシステムではサポートされないことがあります。
cachedAfter	オプション		日付の値です (たとえば、April 16, 1999、4-16-99)。元のクエリーの日付がこの日付よりも後の場合は、ColdFusion ではキャッシュされているクエリーデータが使用されます。キャッシュされているデータを使用するには、現在のクエリーで同じ SQL ステートメント、データソース、クエリー名、ユーザー名、およびパスワードを使用する必要があります。 日付時刻オブジェクトの範囲は、西暦 100 ～ 9999 年です。 日付の値を文字列として指定するときは、文字列を引用符で囲みます。
cacheID	オプション		クエリ結果をキャッシュに保管する際に使用する ID です。 この ID は、キャッシュのクエリの取得と削除の両方で使用できます。
cacheRegion	オプション		クエリ結果をキャッシュする際に使用するキャッシュ領域です。 指定しない場合、デフォルトでは、クエリは QUERY 領域にキャッシュされます。

属性	必須 / オプション	デフォルト	説明
cachedWithin	オプション		期間です。CreateTimeSpan 関数を使用します。元のクエリーの日付がこの期間内の場合、キャッシュされているクエリーデータが使用されます。CreateTimeSpan では、現在からさかのぼって期間を定義します。クエリーのキャッシュ機能が Administrator 内で有効になっている場合のみ機能します。 キャッシュされているデータを使用するには、現在のクエリーで、同じ SQL 文、データソース、クエリー名、ユーザー名、およびパスワードを使用する必要があります。
clientInfo	オプション		データベース接続で設定するクライアントのプロパティが含まれる構造体です。
dataSource	オプション		Datasource 属性は、オプションになりました。省略すると、アプリケーションで指定されたデータソースがクエリーで使用されます。どちらによっても指定されない場合は、エラーが発生します。
dbtype	オプション		クエリーの結果を入力として指定します。dbtype または dataSource を指定します。 ColdFusion では、cfquery で HQL がサポートされます。したがって、次の例に示すように dbtype="hql" を指定できます。 <cfquery dbtype="hql" name="artists" ormoptions=#{cachename=""}#>from Artists where firstname=<cfqueryparam value="Aiden"></cfquery>
debug	オプション (値と等号は省略可能)		• yes、または値は省略された場合 : デバッグがオンになっており、Administrator の [データベースアクティビティ] オプションがオフの場合、データソースに提出された SQL と、クエリーによって返されたレコードの数が表示されます。 • no: Administrator の [データベースアクティビティ] オプションがオンの場合は表示されません。
fetchClientInfo	オプション	no	yes に設定した場合、最後のクエリによって渡された、キーと値のペアを持つ構造体が返されます。
maxRows	オプション	-1 (すべて)	レコードセットで返す行の最大数です。
ormoptions	オプション		HQL を実行する際の orm オプションが含まれる構造体です。dbtype が hql に設定されている場合にのみ適用されます。
password	オプション		データソースセットアップのパスワードよりも優先されます。
result	オプション		cfquery が結果変数を返す構造体の名前です。詳細については、「使用方法」を参照してください。
timeout			エラーが返されるようになるまでにクエリーの各アクションを実行できる秒数の最大値です。累積時間はこの値を超える可能性があります。 JDBC ステートメントの場合は、この属性が設定されます。その他のドライバについては、各ドライバのマニュアルを参照してください。
username	オプション		データソースセットアップのユーザー名よりも優先されます。

使用方法

このタグを使用して、ColdFusion データソースに対して SQL ステートメントを実行します。cfquery タグを使用して任意の SQL DDL (Data Definition Language) または DML (Data Manipulation Language) ステートメントを実行できますが、通常は SQL SELECT ステートメントを実行します。

注意：ストアドプロシージャを呼び出すには、cfstoredproc タグを使用します。

このタグではクエリーオブジェクトが作成され、次の情報がクエリー変数に提供されます。

変数名	説明
query_name.currentRow	cfoutput が処理しているクエリーの現在行です。
query_name.columnList	クエリー列のカンマ区切りリストです。
query_name.RecordCount	クエリーから返されるレコード (行) 数です。

cfquery タグは、構造体内の次の結果変数も返します。result 属性で指定した名前を接頭辞とするこれらの変数にアクセスできます。たとえば、myResult という名前を result 属性に割り当てた場合、#myResult.sql# にアクセスすることにより、実行された SQL ステートメントの名前を取得できます。result 属性は、複数のページから同時に呼び出される可能性がある関数または CFC (あるいはこれら両方) について、一方の呼び出しの結果が他方の呼び出しの結果を上書きしないようにするための方法を提供します。INSERT クエリーの結果変数には、挿入された行の自動生成 ID を示すキーと値のペアが格納されます。この結果変数は、この機能をサポートするデータベースでのみ使用できます。複数のレコードが挿入された場合の値は ID のリストになります。キー名はデータベースごとに異なります。

変数名	説明
result_name.sql	実行された SQL ステートメントです。
result_name.recordcount	クエリーから返されるレコード (行) 数です。
result_name.cached	クエリーがキャッシュされた場合は true、それ以外の場合は false です。
result_name.sqlparameters	cfqueryparam 値の順序配列です。
result_name.columnList	クエリー列のカンマ区切りリストです。
result_name.ExecutionTime	クエリーの処理に要した累積時間です。
result_name.IDENTITYCOL	(SQL Server のみ) 挿入された行の ID です。
result_name.ROWID	(Oracle のみ) 挿入された行の ID です。これは行のプライマリキーではありませんが、この ID に基づいて行を取得できます。
result_name.SYB_IDENTITY	(Sybase のみ) 挿入された行の ID です。
result_name.SERIAL_COL	(Informix のみ) 挿入された行の ID です。
result_name.GENERATED_KEY	(MySQL のみ) 挿入された行の ID です。MySQL 3 はこの機能をサポートしていません。

クエリー結果をキャッシュして、ストアドプロシージャを実行することができます。この方法と、cfquery 出力を表示する方法については、『ColdFusion アプリケーションの開発』を参照してください。

timeout 属性はクエリーの各サブ操作の最大時間に影響するだけであるため、累積時間はこの値を超える可能性があります。非常に大きな結果セットが返される可能性があるページのタイムアウトを設定するには、Administrator の [サーバーの設定]-[リクエストタイムアウト] オプションを適切な値に設定するか、cfsetting タグの RequestTimeout 属性を使用します (<cfsettingrequestTimeout="300"> など)。

ColdFusion Administrator の [キャッシュ機能] ページで、キャッシュされるクエリーの最大数を指定します。この値を 0 に設定すると、クエリーのキャッシュ機能は無効になります。

ColdFusion の予約語をクエリー名に使用することはできません。

SQL の予約語は、エスケープしなければクエリーオブクエリーの変数名または列名としては使用できません。エスケープ文字は角括弧 [] で、たとえば次のように使用します。

```
SELECT [count] FROM MYTABLE.
```

ColdFusion の予約キーワードのリストについては、『ColdFusion アプリケーションの開発』の Escaping reserved keywords を参照してください。

例

```
<!--- This example shows the use of CreateTimeSpan with CFQUERY ------>
<!--- to cache a record set. Define startrow and maxrows to ---->
<!--- facilitate 'next N' style browsing. ---->
<cfparam name="MaxRows" default="10">
<cfparam name="StartRow" default="1">
<!----------------------------------------------------------------------
Query database for information if cached database information has
not been updated in the last six hours; otherwise, use cached data.
---------------------------------------------------------------------->
<cfquery
    name="GetParks" datasource="cfdocexamples"
    cachedwithin="#CreateTimeSpan(0, 6, 0, 0)#">
    SELECT PARKNAME, REGION, STATE
    FROM Parks
    ORDER BY ParkName, State
</cfquery>
<!--- Build HTML table to display query. ------------------------->
<table cellpadding="1" cellspacing="1">
    <tr>
        <td bgcolor="f0f0f0">
             
        </td>
        <td bgcolor="f0f0f0">
            <b><i>Park Name</i></b>
        </td>
        <td bgcolor="f0f0f0">
            <b><i>Region</i></b>
        </td>
        <td bgcolor="f0f0f0">
            <b><i>State</i></b>
        </td>
    </tr>
<!--- Output the query and define the startrow and maxrows parameters.
Use the query variable CurrentCount to keep track of the row you are displaying. ------>
<cfoutput query="GetParks" startrow="#StartRow#" maxrows="#MaxRows#">
    <tr>
        <td valign="top" bgcolor="ffffed">
            <b>#GetParks.CurrentRow#</b>
        </td>
        <td valign="top">
            <font size="-1">#ParkName#</font>
        </td>
```

```
        <td valign="top">
            <font size="-1">#Region#</font>
        </td>
        <td valign="top">
            <font size="-1">#State#</font>
        </td>
    </tr>
</cfoutput>
<!--- If the total number of records is less than or equal to the total number of rows,
then offer a link to the same page, with the startrow value incremented by maxrows
(in the case of this example, incremented by 10). --------->
    <tr>
        <td colspan="4">
        <cfif (StartRow + MaxRows) LTE GetParks.RecordCount>
            <cfoutput><a href="#CGI.SCRIPT_NAME#?startrow=#Evaluate(StartRow + MaxRows)#">
            See next #MaxRows# rows</a></cfoutput>
        </cfif>
        </td>
    </tr>
</table>
```

cfqueryparam

説明

クエリーパラメータのデータ型を検証します。バインド変数をサポートする DBMS では、SQL ステートメントでバインド変数を使用できるようにします。バインド変数を使用することで、cfquery ステートメントを何度も実行するときのパフォーマンスが向上します。

このタグは、クエリー SQL ステートメント内に埋め込まれた cfquery タグ内にネストされます。オプションのパラメータを指定すると、このタグはデータ検証を実行します。

各 cfquery タグ内で cfqueryparam タグを使用して、権限のないユーザーがデータベースにアクセスできないようにしてください。詳細については、セキュリティ情報 ASB99-04「Multiple SQL Statements in Dynamic Queries」(www.adobe.com/go/sn_asb99-04_jp) および『ColdFusion アプリケーションの開発』の Accessing and Retrieving Data を参照してください。

カテゴリ

データベース操作タグ

シンタックス

```
<cfquery
    name = "query name"
    dataSource = "data source name"
    ...other attributes...
    SQL STATEMENT column_name =
    <cfqueryparam value = "parameter value"
        CFSQLType = "parameter type"
        list = "yes|no"
        maxLength = "maximum parameter length"
        null = "yes|no"
        scale = "number of decimal places"
        separator = "separator character">
    AND/OR ...additional criteria of the WHERE clause...>
</cfquery>
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfinsert、cfprocparam、cfprocresult、cfquery、cfstoredproc、cftransaction、cfupdate、『ColdFusion アプリケーションの開発』の Enhancing security with cfqueryparam

属性

属性	必須 / オプション	デフォルト	説明
value	必須		ColdFusion によって where 節の比較演算子の右側に渡される値です。 CFSQLType が日付または時刻のオプションの場合は、日付の値に必ず使用中の DBMS 固有の日付形式を使用します。CreateODBCDateTime または DateFormat および TimeFormat 関数を使用して、日付の値の形式を設定します。
CFSQLType	オプション	CF_SQL_CHAR	任意の型のパラメータがバインドされる SQL 型です。 • CF_SQL_BIGINT • CF_SQL_BIT • CF_SQL_CHAR • CF_SQL_BLOB • CF_SQL_CLOB • CF_SQL_DATE • CF_SQL_DECIMAL • CF_SQL_DOUBLE • CF_SQL_FLOAT • CF_SQL_IDSTAMP • CF_SQL_INTEGER • CF_SQL_LONGVARCHAR • CF_SQL_MONEY • CF_SQL_MONEY4 • CF_SQL_NUMERIC • CF_SQL_REAL • CF_SQL_REFCURSOR • CF_SQL_SMALLINT • CF_SQL_TIME • CF_SQL_TIMESTAMP • CF_SQL_TINYINT • CF_SQL_VARCHAR
list	オプション	no	• yes: value 属性値は区切りリストです。 • no

属性	必須 / オプション	デフォルト	説明
maxLength	オプション	value 属性値の文字列の長さ	パラメータの最大の長さです。文字列が DBMS に送信される前に ColdFusion で長さチェックを実行するようにする必要があります。その結果、悪意のある文字列の送信を回避できます。
null	オプション	no	パラメータを null 値として渡すかどうかを指定します。 • yes: value 属性を無視します。 • no
scale	オプション	0	パラメータの小数点以下の桁数です。CF_SQL_NUMERIC および CF_SQL_DECIMAL に適用されます。
separator	value 属性でリストを指定する場合は必須	, (カンマ)	value 属性内で、リストの値を区切る文字です。

使用方法

cfqueryparam タグは、ColdFusion 変数を使用する任意の SQL ステートメント (SELECT、INSERT、UPDATE、DELETE など) で使用します。

文字列データの検証の最大値を決定するために、maxlength 属性を指定します。

このタグでは次のことが実行されます。

- SQL バインドパラメータを使用可能にします。これによりパフォーマンスが向上します。
- 指定された SQL 型と変数データが一致していることを確認します。
- SQL ステートメントからの長いテキストフィールドの更新を可能にします。
- 文字列変数を一重引用符でエスケープします。

バインド変数によるパフォーマンスの向上を活かすには、すべての ColdFusion 変数に cfqueryparam を使用します。また、DBMS がバインド変数をサポートしている必要があります。DBMS がバインドパラメータをサポートしていない場合は、ColdFusion で検証が行われ、検証済みのパラメータ値が文字列に置き換えられます。検証に失敗すると、エラーメッセージが返されます。

検証ルールは次のとおりです。

- CF_SQL_SMALLINT、CF_SQL_INTEGER、CF_SQL_REAL、CF_SQL_FLOAT、CF_SQL_DOUBLE、CF_SQL_TINYINT、CF_SQL_MONEY、CF_SQL_MONEY4、CF_SQL_DECIMAL、CF_SQL_NUMERIC、および CF_SQL_BIGINT については、データ値を数値に変換することができます。
- CF_SQL_DATE、CF_SQL_TIME、CF_SQL_TIMESTAMP については、データ値をターゲットデータソースでサポートされる日付に変換できます。
- その他の型では、maxLength 属性を指定した場合、データ値は指定されている最大長を超えることはできません。

ColdFusion のデバッグ出力では、バインド変数が疑問符 (?) で表示され、クエリーの下に値が使用順に表示されます。

注意: SequelLink ODBC Socket または SequelLink Access ドライバを使用して Microsoft Access テーブルに空の文字列を挿入するには、CFSQLType 属性で CF_SQL_LONGVARCHAR を指定する必要があります。

次の表は、JDBC SQL 型を使用する ColdFusion SQL データ型のマッピング、および各データベース管理システムのデータ型を示しています。

ColdFusion	JDBC	DB2	Informix	Oracle	MSSQL
CF_SQL_ARRAY	ARRAY				
CF_SQL_BIGINT	BIGINT	Bigint	int8、serial8		bigint
CF_SQL_BINARY	BINARY	Char for Bit Data			binary timestamp
CF_SQL_BIT	BIT		boolean		bit
CF_SQL_BLOB	BLOB	Blob	blob	blob、bfile	longvarbinary
CF_SQL_CHAR	CHAR	Char	char、nchar	char、nchar	char
CF_SQL_CLOB	CLOB	Clob	clob	clob、nclob	
CF_SQL_DATE	DATE	Date	date、datetime、year to day		date
CF_SQL_DECIMAL	DECIMAL	Decimal	decimal、money	number	decimal
CF_SQL_DISTINCT	DISTINCT				
CF_SQL_DOUBLE	DOUBLE	Double			double
CF_SQL_FLOAT	FLOAT	Float	float	number	real
CF_SQL_IDSTAMP	CHAR	Char	char、nchar	char、nchar	char
CF_SQL_INTEGER	INTEGER	Integer	integer、serial		integer
CF_SQL_LONGVARBINARY	LONGVARBINARY	Long Varchar for Bit Data	byte	long raw	longvarbinary
CF_SQL_LONGVARCHAR	LONGVARCHAR	Long Varchar	text	long	longvarchar
CF_SQL_MONEY	DOUBLE	Double			double
CF_SQL_MONEY4	DOUBLE	Double			double
CF_SQL_NULL	NULL				
CF_SQL_NUMERIC	NUMERIC	Numeric			numeric
CF_SQL_OTHER	OTHER				
CF_SQL_REAL	REAL	Real	smallfloat		real
CF_SQL_REFCURSOR	REF				
CF_SQL_SMALLINT	SMALLINT	Smallint	smallint		smallint
CF_SQL_STRUCT	STRUCT				
CF_SQL_TIME	TIME	Time	datetime hour to second		time
CF_SQL_TIMESTAMP	TIMESTAMP	Timestamp	datetime year to fraction(5)、datetime year to second	date	timestamp

ColdFusion	JDBC	DB2	Informix	Oracle	MSSQL
CF_SQL_TINYINT	TINYINT				tinyint
CF_SQL_VARBINARY	VARBINARY	Rowid		raw	varbinary
CF_SQL_VARCHAR	VARCHAR	Varchar	varchar、nvarchar、lvarchar	varchar2、nvarchar2	varchar

例

```
<!--- This example shows cfqueryparam with VALID input in Course_ID. --->
<h3>cfqueryparam Example</h3>
<cfset Course_ID = 12>
<cfquery name = "getFirst" dataSource = "cfdocexamples">
    SELECT *
    FROM courses
    WHERE Course_ID = <cfqueryPARAM value = "#Course_ID#"
    CFSQLType = 'CF_SQL_INTEGER'>
</cfquery>
<cfoutput query = "getFirst">
    <p>Course Number: #Course_ID#<br> Description: #descript#</p>
</cfoutput>

<!--- This example shows the use of CFQUERYPARAM when INVALID string data is
    in Course_ID. ---->
<p>This example throws an error because the value passed in the CFQUERYPARAM tag exceeds the
    MAXLENGTH attribute</p>

<cfset LastName="Peterson; DELETE employees WHERE LastName='Peterson'">
<!------- Note that for string input you must specify the MAXLENGTH attribute
    for validation. ---------------------------------------------------->
<cfquery
    name="getFirst" datasource="cfdocexamples">
    SELECT *
    FROM employees
    WHERE LastName=<cfqueryparam
                value="#LastName#"
                cfsqltype="CF_SQL_VARCHAR"
                maxlength="17">
</cfquery>
<cfoutput
    query="getFirst"> <p>
        Course Number: #FirstName# #LastName#
        Description: #Department# </p>
</cfoutput>
```

タグ r ～ s

cfregistry

説明

このタグは、UNIX プラットフォームでは非推奨となっています。

システムレジストリ内でキーおよび値の読み取り、書き込み、および削除を行います。クライアント変数の永続ストレージを提供します。

注意：このタグを実行するには、ColdFusion Administrator でタグを有効にする必要があります。詳細については、『ColdFusion 設定と管理』を参照してください。

カテゴリ
その他のタグ、変数操作タグ

シンタックス
タグのシンタックスは action 属性の値によって異なります。次のセクションを参照してください。

- cfregistry action = "get"
- cfregistry action = "set"
- cfregistry action = "getAll"
- cfregistry action = "delete"

関連項目
cfcookie、cfparam、cfsavecontent、cfschedule、cfset、『ColdFusion アプリケーションの開発』の About resource and sandbox security および Using Persistent Data and Locking

履歴
ColdFusion MX:

- UNIX プラットフォームで、このタグが非推奨になりました。今後のリリースではこのタグは機能せずエラーを引き起こす可能性があります。
- 永続データの保管方法が変更されました。システムレジストリ外部の最も永続的なデータを XML ファイルに保管するようになりました。

cfregistry action = "getAll"

説明
ブランチで定義されているすべてのレジストリキーと値を返します。これらの値には、レコードセットにアクセスする場合と同様にアクセスできます。

シンタックス

```
<cfregistry
    action = "getAll"
    branch = "branch"
    name = "query name"
    sort = "asc|desc"
    type = "string|dWord|key|any">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目
『ColdFusion アプリケーションの開発』の Using Persistent Data and Locking

属性

属性	必須 / オプション	デフォルト	説明
action	必須		常に getall です。
branch	必須		レジストリブランチの名前です。
name	必須		返されたキーと値を格納するレコードセットの名前です。
sort	オプション	asc	クエリー列のデータをソートします (大文字と小文字の区別はありません)。「Entry」、「Type」、および「Value」の列上で、テキストとしてソートを行います。クエリー出力からの列の組み合わせを、カンマ区切りリストで指定します。たとえば、次のようになります。 `sort = "value desc, entry asc"` • asc: 昇順 (a ～ z) のソートです。 • desc: 降順 (z ～ a) のソートです。
type	オプション	string	• string: 文字列の値を返します。 • dWord: DWord 値を返します。 • key: キーを返します。 • any: キーと値を返します。

使用方法

このタグでは、cfoutput などのタグによってアクセスできるレコードセット内に #entry#、#type#、および #value# が返されます。これらの変数を完全修飾するには、name 属性に指定したものと同じレコードセット名を使用します。

#type# がキーの場合、#value# は空の文字列になります。

type="any" を指定すると、getAll によってバイナリのレジストリ値も返されます。バイナリ値の場合は、#type# 変数は UNSUPPORTED となり、#value# は空白になります。

例

```
<!--- This example uses cfregistry with the getAll action. --->
<cfregistry action = "getAll"
    branch = "HKEY_LOCAL_MACHINE\Software\Microsoft\Java VM"
    type = "Any" name = "RegQuery">
<h1>cfregistry action = "getAll"</h1>
<cftable query = "RegQuery" colHeaders HTMLTable border = "yes">
    <cfcol header = "<b>Entry</b>" width = "35" text = "#RegQuery.Entry#">
    <cfcol header = "<b>Type</b>" width = "10" text = "#RegQuery.type#">
    <cfcol header = "<b>Value</b>" width = "35" text = "#RegQuery.Value#">
</cftable>
```

cfregistry action = "get"

説明

レジストリ値にアクセスし、それを ColdFusion 変数内に保管することができます。

シンタックス

```
<cfregistry
    action = "get"
    branch = "branch"
    entry = "key or value"
    variable = "variable"
    type = "string|dWord|key">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

『ColdFusion アプリケーションの開発』の Using Persistent Data and Locking

属性

属性	必須 / オプション	デフォルト	説明
action	必須		常に get です。
branch	必須		レジストリブランチの名前です。
entry	必須		アクセスするレジストリ値です。
variable	必須		値を保管するための変数です。
type	オプション	*string*	• string: 文字列の値を返します。 • dWord: DWord 値を返します。 • key: キーのデフォルト値を返します。

使用方法

値が存在しない場合は、cfregistry タグによりエントリは作成されません。

例

```
<!--- This example uses cfregistry with the get action. --->
<cfregistry action = "get"
    branch = "HKEY_LOCAL_MACHINE\Software\Microsoft\Java VM"
    entry = "ClassPath" type = "String" variable = "RegValue">
<h1>cfregistry action = "get"</h1>
<cfoutput>
    Java ClassPath value is #RegValue#
</cfoutput>
```

cfregistry action = "set"

説明

レジストリキーの追加、値の追加、値の更新を行います。

シンタックス

```
<cfregistry
    action = "set"
    branch = "branch"
    entry = "key or value"
    type = "string|dWord|key"
    value = "data">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

『ColdFusion アプリケーションの開発』の Using Persistent Data and Locking

属性

属性	必須 / オプション	デフォルト	説明
action	必須		常に set です。
branch	必須		レジストリブランチの名前です。
entry	必須		設定するキーまたは値です。
type	オプション		• string: 文字列値を設定します (デフォルト)。 • dWord: DWord 値を設定します。 • key: キーを作成します。
value	オプション		設定する値データです。この属性を省略すると、cfregistry タグにより次のようなデフォルト値が作成されます。 • string: 空の文字列 "" を作成します。 • dWord: 値 0 (ゼロ) を作成します。

使用方法

キーまたは値が存在しない場合は、cfregistry タグによってキーまたは値が作成されます。

例

```
<!--- This example uses the cfregistry set action to modify registry value data. --->
<!--- Normally you pass in a filename instead of setting one here. --->
<cfset FileName = "dummy.cfm">
<cfregistry action = "set"
    branch = "HKEY_LOCAL_MACHINE\Software\cflangref"
    entry = "LastCFM01" type = "String" value = "#FileName#">
<h1>cfregistry action = "set"</h1>
```

cfregistry action = "delete"

説明

レジストリキーまたは値を削除します。

シンタックス

```
<cfregistry
    action = "delete"
    branch = "branch"
    entry = "key or value">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

『ColdFusion アプリケーションの開発』の Using Persistent Data and Locking

属性

属性	必須 / オプション	デフォルト	説明
action	必須		常に delete です。
branch	必須		• キーを削除する場合 : 削除するレジストリキーの名前です。entry 属性を指定しないでください。 • 値を削除する場合 : 削除する値が含まれているレジストリブランチの名前です。entry 属性を指定します。
entry	値を削除する場合は必須		削除する値です。

使用方法

キーを削除すると、キーの下に定義されている値およびサブキーも cfregistry タグによって削除されます。

例

```
<cfregistry action = "delete"
    branch = "HKEY_LOCAL_MACHINE\Software\cflangref\tempkey"
    entry = "LastCFM01">
<h1>cfregistry action = "delete"</h1>
```

cfreport

説明

次のいずれかを実行するために使用します。

- ColdFusion Report Builder で作成されたレポートを実行し、PDF、Adobe® FlashPaper®、RTF、HTML、XML、Excel のいずれかの形式で表示します。オプションで、このレポートをファイルに保存することもできます。
- 定義済みの Crystal Reports レポートを実行します。Windows システムにのみ適用されます。

カテゴリ

データ出力タグ

シンタックス

```
 ColdFusion Report Builder syntax:
<cfreport
    format = "PDF|FlashPaper|Excel|RTF|HTML|XML"
    template = "absolute pathname or pathname relative to the report file"
    encryption = "128-bit|40-bit|none"
    filename = "output filename"
    name = "ColdFusion variable"
    ownerpassword = "password"
    overwrite = "no|yes"
    permissions = "permission list"
    query = "query variable"
    resourceTimespan = #CreateTimeSpan (days, hours, minutes, seconds)#
    style = "CSS style definition or css file pathname"
    userpassword = "password">
    <cfreportparam ...>
</cfreport>

Crystal Reports syntax:
<cfreport
    report = "report path"
    dataSource = "data source name"
    formula = "formula"
    orderBy = "result order"
    password = "password"
    timeout = "number of seconds"
    type = "standard|netscape|microsoft"
    username = "username">
</cfreport>
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfcollection、cfdocument、cfdocumentitem、cfdocumentsection、cfexecute、cfindex、cfobject、cfreportparam、cfsearch、cfwddx、『ColdFusion アプリケーションの開発』の Creating Reports with Report Builder、Report Builder オンラインヘルプ

履歴

ColdFusion 8: style 属性と resourceTimespan 属性が追加されました。format 属性の値として、HTML と XML が追加されました。

ColdFusion MX 7.0.1: format 属性の値として RTF が追加され、RTF 形式でレポートを生成できるようになりました。

ColdFusion MX 7: ColdFusion Report Builder に対するサポートが追加されました。

ColdFusion MX: データソース接続の動作が変更されました。Crystal Reports は、データソースに独立した接続を確立するようになりました。この接続は、ColdFusion データソース固有の制約を受けません。たとえば、Crystal Reports サーバーは、ColdFusion Administrator で無効になっているかどうかにかかわらず、データソースにアクセスできます。

属性

属性	適用対象	必須 / オプション	デフォルト	説明
datasource	Crystal Reports	オプション		登録済みまたはネイティブのデータソースの名前です。
encryption	Report Builder	オプション	none	(format="PDF" の場合のみ) レポート出力の暗号化方式です。有効な値は、次のとおりです。 • 128-bit • 40-bit • none
filename	Report Builder	オプション		レポートが含まれるファイルの名前です。name 属性と filename 属性の両方を指定することはできません。ファイル名の拡張子は、format 属性の値と一致している必要があります。 レポート出力を HTML ファイルに書き込む場合は、*filename* で指定した出力ファイルを基準とする場所にディレクトリが自動的に作成されます。また、レポート内のチャートの PNG ファイルやレポートにインポートされるイメージファイルのコピーが生成されて、このディレクトリに格納されます。
format	Report Builder	必須		レポート出力の形式です。 • PDF • FlashPaper • Excel • RTF • XML • HTML レポート出力を HTML 形式でブラウザに直接書き込むと、テンポラリディレクトリとレポート内のイメージに対応するファイルが生成されます。イメージファイルが格納されるテンポラリディレクトリは、次の場所に生成されます。 C:¥ColdFusion9¥tmpCache¥CFFileServlet¥_cfreport¥_report[< 一意の識別子 >] ブラウザからこれらのイメージを削除する時期を指定するには、resourceTimespan 属性を使用します。

formula	Crystal Reports	オプション		名前付き式を指定します (複数指定可)。各式の最後にはセミコロンを付けます。次の形式を使用します。
				`formula = "formulaname1 = 'formula1';formulaname2 = 'formula2';"`
				式の中でセミコロンを使用する場合は、2 回 (;;) 入力してエスケープします。たとえば、次のようになります。
				`formula = "Name1 = 'Val_1a;;Val_1b';Name2 = 'Val2';"`

name	Report Builder	オプション		レポート出力が含まれている ColdFusion 変数の名前です。name と filename の両方を指定することはできません。format="HTML" の場合、この属性は有効ではありません。
orderBy	Crystal Reports	オプション		ユーザーが指定する結果の並び順です。

属性	適用対象	必須 / オプション	デフォルト	説明
overwrite	Report Builder	オプション	no	filename 属性で指定したものと同じ名前のファイルが存在する場合に上書きするかどうかを指定します。 • yes • no
ownerPassword	Report Builder	オプション		(format="PDF" の場合のみ) レポートのオーナーパスワードです。
password	Crystal Reports	オプション		データベースアクセスに必要なユーザー名に対応するパスワードです。ColdFusion Administrator でのデータソースのデフォルト設定値よりも優先されます。
permissions	Report Builder	オプション		(format="PDF" の場合のみ) 次のアクセス許可を指定します (複数指定可)。 • AllowPrinting • AllowModifyContents • AllowCopy • AllowModifyAnnotations • AllowFillIn • AllowScreenReaders • AllowAssembly • AllowDegradedPrinting 複数のアクセス許可を指定する場合は、カンマで区切ります。
query	Report Builder	オプション		レポートの入力データを含むクエリーの名前です。このクエリーは、Report Builder レポート内のクエリーより優先されます。この ColdFusion クエリーには、少なくとも Report Builder クエリー内の列がすべて含まれている必要がありますが、WHERE 節は異なっていてもかまいません。このパラメータを省略した場合、内部 SQL ステートメントまたは cfreportparam 項目のデータが Report Builder で使用されます。
report	Crystal Reports	必須		レポートのパス名です。ColdFusion ページファイルと同じディレクトリに Crystal Reports ファイルを保管します。
resourceTimespan	Report Builder	オプション	5 分	(format="HTML" の場合のみ) リソースディレクトリの有効期間です。Report Builder レポートを HTML 形式でエクスポートすると、レポート内のイメージやその他のリソースファイルがテンポラリリソースディレクトリに書き込まれます。この属性を使用すると、リソースディレクトリを削除する時期を指定できます。この値を設定するには、CreateTimeSpan 関数を使用して、日、時、分、秒をカンマで区切った期間を指定します。たとえば、1 時間後にリソースディレクトリを削除するには、「#CreateTimeSpan(0,1,0,0)#」のように指定します。 値が 0 に設定されている場合は、次にサーバーが再起動されるまでリソースディレクトリが保持されます。 リソースディレクトリが削除されるのは、format="HTML" に設定されていて、name 属性も filename 属性も指定されていない場合のみです。デフォルト値は 5 分です。これを明示的に指定すると、「#CreateTimeSpan(0,0,5,0)#」のようになります。

属性	適用対象	必須 / オプション	デフォルト	説明
style	Report Builder	オプション		Report Builder レポートで定義されているスタイルを無視して実行時に適用する CSS 形式のスタイルです。CSS ファイルの絶対パス、レポートを基準とした CSS ファイルの相対パス、または有効な CSS 形式で記述された文字列を指定できます。 スタイルを有効にするには、Report Builder レポート内の要素に割り当てられている Style Name 属性とこのスタイル名が一致している必要があります。CSS ファイルは Report Builder で生成してエクスポートすることも、テキストエディタで作成することもできます。サポートされている CSS スタイルのリストについては、「スタイルプロパティ」を参照してください。
template	Report Builder	必須		Report Builder (CFR) ファイルへのパス名 (Web ルートからの相対パス名) を指定します。
timeout	Crystal Reports	オプション		Crystal Reports ファイルへの接続が確立されるまで待機する最大時間 (秒) を指定します。
type	Crystal Reports	オプション	standard	レポートのタイプです。 • standard (Crystal Reports 8.0 では無効) • netscape • microsoft
userName	Crystal Reports	オプション		レポートの作成に使用するデータベースに対してアクセス権を持つユーザー名です。ColdFusion Administrator で指定されたデータソースのデフォルト設定よりも優先されます。
userPassword	Report Builder	オプション		(format="PDF" の場合のみ) ユーザーのパスワードです。

使用方法

このタグを使用して、ColdFusion Report Builder または Crystal Reports で作成されたレポート定義に基づくレポートを生成します。ColdFusion Report Builder の使用方法の詳細については、Report Builder を起動して F1 キーを押し、オンラインヘルプを表示してください。

注意：レポートの出力形式を Excel タイプに設定した場合は、ColdFusion レポート機能でサポートされる形式設定オプションの一部を利用できません。Excel では、イメージとチャートはサポートされず、カンマ、パーセント、通貨などの書式を含む数値データはプレーンテキストとして表示されます。Excel 形式でレポートを出力する場合は単純なレポートしかサポートされないので、デザインとレイアウトを慎重に検討することをお勧めします。

このタグには終了タグが必要です。

CSS (Cascading Style Sheet) の使用

ColdFusion では、cfreport タグの style 属性を使用して、Report Builder レポート内のカスケーディングスタイルシート (CSS) を実行時に上書きできます。

CSS ファイルは次の 2 つの方法で作成できます。1 つは Report Builder の [レポートスタイルのエクスポート] アイコンを使用してスタイルをエクスポートする方法で、もう 1 つはテキストエディタで CSS ファイルを作成する方法です。ただし、それらの CSS スタイルを有効にするには、Report Builder を使用してレポート内の要素にスタイル名を割り当てる必要があります (デフォルトスタイルは例外です。Report Builder でデフォルトスタイルが定義されていない場合でも、style 属性を使用して ColdFusion にデフォルトスタイルを定義すれば、そのスタイルをレポートに適用できます)。

Report Builder でスタイル名を割り当てると、いつでも CSS ファイル内のスタイル定義を更新し、cfreport タグと cfreportparam タグを使用して実行時に適用できます。レポートにサブレポートが含まれている場合は、マスターレポートとすべてのサブレポートにデフォルトスタイルが適用されます。デフォルトスタイル以外の CSS スタイルがマスターレポートで使用されている場合は、明示的に指定しない限り、その CSS スタイルはサブレポートに適用されません。

次のコードは、3 つの異なるスタイルシートを実行時にメインレポートと 2 つのサブレポートに適用する方法を示しています。

```
<cfreport template="myreport.cfr" style="mystyle.css" format="PDF">
    <cfreportParam subreport="subreport1" style="subreport1-style.css">
    <cfreportParam subreport="subreport2" style="subreport2-style.css">
</cfreport>
```

次のコードは、style 属性の値として CSS スタイルを適用する方法を示しています。

```
<cfreport template="myreport.cfr" style='mystyle { defaultStyle: true; font-family:"Comic Sans MS"; color:
##00FF00; }' format="FlashPaper">
</cfreport>
```

次のコードは、**myStyle** という変数を作成して、style 属性の値として使用する方法を示しています。

```
<cfset mystyle='mystyle { defaultStyle: false; font-family: "Comic Sans MS"; }'>
<cfreport template="myreport.cfr" style="#mystyle#" format="HTML">
</cfreport>
```

style 属性のシンタックス

スタイルファイルまたは文字列は、有効な CSS シンタックスで記述する必要があります。詳細については、http://www.w3.org/Style/CSS/ を参照してください。スタイルには少なくとも 1 つのルールセットが含まれている必要があります。各ルールセットは単純なセレクタ (Report Builder のスタイル名) と、それに続く宣言ブロック (一連の宣言をセミコロンで区切ったブロック) で構成されます。宣言はプロパティと値のペアで構成されます。

セレクタに無効なシンタックスが含まれている場合、そのセレクタと宣言ブロックは無視されます。また、この機能でサポートされていないセレクタとプロパティも無視されます。CSS の制御下にない部分 (フォント名など) を除き、スタイルの大文字と小文字は区別されません。

次の例は、デフォルトスタイルの CSS 定義を示しています。

```
DefaultStyle
    {
        default-style: true;
        color: black;
        font-family: Arial, "Comic Sans MS";
        font-size: 16;
        text-decoration: underline;
    }
```

次の例は、PurpleBoldItalicText というカスタムスタイルの CSS 定義を示しています。

```
PurpleBoldItalicText
    {
        color: purple;
        font: italic bold 20px 30px Arial;
    }
```

スタイルの識別子は CSS2 に準拠している必要があります。たとえば、CSS1 では識別子の中で '_' を使用できますが、CSS2 では使用できません。

CSS2 の場合、セレクタの要素名、クラス、ID などの識別子の中で使用できる文字は、A ～ Z、a ～ z、0 ～ 9 のみです。また、ISO 10646 の 161 以降の文字やハイフン文字 (-) も使用できます。ただし、識別子の先頭にハイフンや数字を使用することはできません。エスケープ文字や ISO 10646 文字は数字コードに置き換えて使用できます。たとえば、「B&W?」という識別子を指定するには、「B¥&W¥?」または「B¥26 W¥3F」と入力します。

スタイルプロパティ

次の表は、Report Builder によってエクスポートされるスタイルプロパティを示しています。

プロパティ名	Report Builder のみ	有効な値	説明
background-color	いいえ	「有効なカラー値」を参照	指定されたレポート要素の背景色です (要素が透明でない場合のみ)。デフォルトの背景色は白です。
border	いいえ	[border-width] [border-style] [border-color]	要素内のすべてのボーダーに対して border-width、border-style、border-color の各プロパティを指定する略式のプロパティです。
border-color	いいえ	「有効なカラー値」を参照	テキスト、イメージ、およびチャートのボーダーの色です。ボーダーの各側面をカスタマイズできます。デフォルトの色は白です。
border-style	いいえ	solid dashed none	border-top-style、border-right-style、border-bottom-style、border-left-style の各プロパティを指定する略式のプロパティです (この順序で値を指定し、それぞれの値をカンマで区切る必要があります)。値が指定されていない場合は、反対側の値が使用されます。値が 1 つしか指定されていない場合は、すべての側面にその値が適用されます。 none を指定すると border-width の値が無視されます。その他の値 (hidden、dotted、groove、ridge、inset、outset、double など) はすべて実線として表示されます。
border-top-color border-left-color border-bottom-color border-right-color	いいえ	「有効なカラー値」を参照	要素の上、左、下、右のボーダーの色です。詳細については、「ボーダーとマージンのスタイル」を参照してください。 border-color プロパティが指定されていない場合は、color プロパティの値が代わりに使用されます。
border-top-style border-left-style border-bottom-style border-right-style	いいえ	solid dashed	要素の上、左、下、右のボーダーの線スタイルです。詳細については、「ボーダーとマージンのスタイル」を参照してください。 dashed 以外の値はすべて実線として表示されます。
border-top-width border-left-width border-bottom-width border-right-width	いいえ	thin medium thick 1px 2px 4px	要素の上、左、下、右のボーダーの太さです。負の値は無効です。詳細については、「ボーダーとマージンのスタイル」を参照してください。 • thin = 1/2 pt • medium = 2px • thick = 4px
border-width	いいえ	thin medium thick 1px 2px 4px	border-top-width、border-right-width、border-bottom-width の各プロパティを単一のプロパティと値の表記で指定する略式のプロパティです (この順序で値を指定し、それぞれの値をカンマで区切る必要があります)。値が指定されていない場合は、反対側の値が使用されます。値が 1 つしか指定されていない場合は、すべての側面にその値が適用されます。 • thin = 1/2 pt • medium = 2px • thick = 4px

プロパティ名	Report Builder のみ	有効な値	説明
clip	いいえ	auto stretch ratio	イメージのサイズを変更する方法を指定します。 • auto: ソースイメージのサイズがレポート内の要素と異なる場合、要素の境界内に収まるようにイメージをトリミングします。イメージのサイズは変更されません。境界内に収まるイメージ部分のみが表示されます。 • stretch: ソースイメージのサイズがレポート内の要素と異なる場合、要素の境界内に収まるようにイメージのサイズを変更します。縦横比が異なる場合は、イメージが歪みます。 • ratio: ソースイメージのサイズがレポート内の要素と異なる場合、要素の境界内に収まるようにイメージのサイズを変更します。ただし、イメージが歪まないようにソースの縦横比を維持します。
color	いいえ	「有効なカラー値」を参照	テキスト要素内のテキストとグラフィック要素内のボーダーの色です。デフォルトの色は黒です。
corner-radius	はい	integer	矩形の角の描画に使用する円弧の半径です。デフォルト値は 0 (直角) です。0 未満の値は 0 と解釈されます。
default-style	はい	true false	明示的にスタイルが適用されていない要素に対して使用するデフォルトスタイルです。サブレポートに固有のデフォルトスタイルがない場合は、親のデフォルトスタイルが継承されます。
embed-pdf-font	はい	true false	PDF ドキュメントにフォントを埋め込むかどうかを指定します。埋め込みフォントを使用すると、レポートを表示するシステムにそのフォントがインストールされていない場合でも、正しく表示されます。
empty-cells	いいえ	show hide	テキスト式の null 値を表示するか非表示にするかを指定します。 • show: テキスト式が null 値を返した場合は、"null" という文字列を表示します。 • hide: テキスト式が null 値を返した場合は、null 値を空の文字列に置き換えます。これがデフォルトの設定です。
font	いいえ	[font-style] [font-weight] [font-size] [line-height] [font-family]	フォントの特性を指定します。これを使用すると、複数のプロパティ値を簡単に指定できます。例 : font: italic 20px Arial; このプロパティのデフォルト値は、個々のプロパティのデフォルト値と同じです。個々のプロパティのデフォルト値は、font プロパティで指定されていない値の代わりとして適用されます。
font-family	いいえ	フォント名のカンマ区切りリスト	要素に適用するフォントのグループです。リストの最初で指定されているフォントが要素に適用されます。デフォルト値は次のとおりです。 font-family: Arial, Helvetica, sansserif; フォント名にスペースが含まれる場合は、名前を引用符で囲む必要があります。例 : font-family: Courier, "Courier New", Arial;

プロパティ名	Report Builder のみ	有効な値	説明
font-size	いいえ	[length]	フォントサイズです (単位 : ポイントまたはピクセル)。負の値は無効です。デフォルト値は 10 ポイントです。ポイント数またはピクセル数で指定できます。1 ピクセルは 0.75 ポイントに相当します。このプロパティは、font プロパティのコンポーネントでもあります。 標準の CSS では、Report Builder でサポートされていないタイプの値もサポートされます。
font-style	いいえ	normal italic oblique	フォントのスタイルです。italic と oblique の効果はほぼ同じです。デフォルト値は normal です。このプロパティは、font プロパティのコンポーネントでもあります。
font-weight	いいえ	normal bold bolder lighter 100 200 300 400 500 600 700 800 900	フォントの太さです。Report Builder ではフォントの太さを細かく指定できないため、normal と lighter は normal として表示され、他の値はすべて bold として表示されます。デフォルト値は normal です。このプロパティは、font プロパティのコンポーネントでもあります。
line-height	いいえ	normal [number] [length] [percentage]	連続するテキストの行間です。 • normal: 行の高さをシングルスペース (行間なし) に設定します (デフォルト)。 • number: 要素のフォントサイズの係数として行の高さを決定するための乗数を指定します。この数値から行の高さを求めるには、現在の要素のフォントサイズにこの数値を乗算します。負の値は無効です。 • length: 行の高さを指定した高さに設定します。ポイント数 ("20" など) またはピクセル数 ("20px" など) を指定できます。1 ピクセルは 0.75 ポイントに相当します。負の値は無効です。標準の CSS では、Report Builder でサポートされていない長さの単位も使用できます。 • percentage: 行の高さをパーセントで指定します。パーセント記号が必要です (例 : 150%)。負の値は無効です。
line-size	はい	none thin 1px 2px 4px dashed	グラフィック要素を囲むボーダーのタイプ、または線要素のタイプと太さです。デフォルトでは、線および矩形のボーダーは 1 ピクセルに設定されます。thin は 0.05 ピクセルです。

プロパティ名	Report Builder のみ	有効な値	説明
margin	いいえ	[top-integer] [right-integer] [bottom-integer] [left-integer]	要素の境界ボックス内の余白です。このプロパティは、margin-top、margin-right、margin-bottom、margin-left の各プロパティを単一のプロパティと値の表記で指定する略式のプロパティです (この順序で値を指定し、それぞれの値をカンマで区切る必要があります)。値が指定されていない場合は、反対側の値が使用されます。値が 1 つしか指定されていない場合は、すべての側面にその値が適用されます。詳細については、「ボーダーとマージンのスタイル」を参照してください。 CSS のマージンは透明なので、この部分には親要素の背景が表示されます。負の値も有効です (したがって、テキストを重ねて表示することもできます)。 例： margin: 10,20,30,40; margin: 10; margin: 10,20,30;
margin-top margin-left margin-bottom margin-right	いいえ	integer	margin を参照してください。
parent-style	はい	styleName	このスタイルに一部またはすべてのプロパティを継承する親レポートスタイルの名前です。このスタイル名は、レポート内で定義するか、CSS ファイルまたは CSS テキスト内でこのスタイル定義が出現する前に定義する必要があります。
text-align	いいえ	left center right justify	水平軸上におけるテキストおよびイメージの整列方法です。デフォルト値は left です。
text-decoration	いいえ	underline line-through underline line-through	font-style プロパティおよび font-weight プロパティで定義されないテキストの特性です。text-decoration の色は要素の color プロパティによって決定されます。不明な値は無視されます。
text-rotation	はい	none left right	テキスト要素を回転させるかどうかを指定します。テキストの向きを変えるには、このプロパティを使用してテキストを右または左に 90 度回転させます。

プロパティ名	Report Builder のみ	有効な値	説明
transparency-mode	はい	opaque transparent	要素を透明にするかどうかを指定します。矩形や線などのグラフィック要素はデフォルトで opaque (不透明) ですが、イメージは transparent (透明) です。サブレポートの要素、スタティックテキスト、およびテキストフィールドは、デフォルトで transparent (透明) です。
vertical-align	いいえ	top middle bottom	垂直軸上におけるテキストおよびイメージの整列方法です。デフォルト値は top です。
xhtml-formatted-text	はい	true false	XHTML と互換性を持つ命令がテキスト要素に含まれているかどうかを指定します。 • true: xhtml と互換性を持つ命令 (My Text Label など) がテキスト要素に含まれています。この例の場合、タグ内のテキスト (**My Text Label**) は太字で表示され、タグ ( ) は表示されません。 • false: xhtml と互換性を持つ命令がテキスト要素に含まれていません。したがって、テキスト要素内のすべてのテキストが表示されます。これがデフォルトの設定です。

Report Builder でサポートされていないスタイルや値がレポートに含まれている場合は無視されます。その場合、デフォルトスタイルが定義されていれば、デフォルトスタイルがその要素に適用されます。

有効なカラー値

色は #RRGGBB 形式で指定できます。これは 16 進数値のペアを 3 つ使用して色を表現したものです。この値は赤、緑、青の成分を表しています。各成分の値の範囲は 16 進数の 00 ～ FF です。また、140 個の X11 カラー名から 1 つを選択して使用することもできます (http://www.blooberry.com/indexdot/color/x11makerFrameNS.htm を参照してください)。カラー色では大文字と小文字が区別されません。カラー名のセットでは、特定の RGB 値に名前が割り当てられています。また、#RGB、rgb(r,g,b)、rgb(r%,g%,b%) のいずれかを使用してカラー名を指定することもできます。シンタックスの詳細については、http://www.blooberry.com/indexdot/css/syntax/units/color.htm の「Color Units in CSS」を参照してください。UI Name はサポートされていません。

次の例は、ライム色をさまざまな表現で指定する方法を示しています。

```
color:lime
color:#00FF00
color:#0F0
color:rgb(0,255,0)
color:rgb(0%,50%,0%)
```

cfreport タグの style 属性で 16 進数形式を使用して色を指定する場合は、##00FF00 の形式を使用します。たとえば、次のようになります。

```
<cfreport template="myreport.cfr" style='mystyle { defaultStyle: true;
    font-family:"Comic Sans MS"; color: ##00FF00; }' format="HTML"/>
```

ボーダーとマージンのスタイル

要素の 4 つの側面にすべて同じ値を使用する場合は、border-width、border-style、border-color、margin の各プロパティを使用します。これらのプロパティでは 1 ～ 4 つのパラメータを指定できます。

パラメータの数	例	結果
1	border-width: thick	要素のボーダーの 4 つの側面すべてにパラメータが適用されます。
2	border-width: thick, thin	最初のパラメータ (thick) は上と下の側面に、2 番目のパラメータ (thin) は左と右の側面に適用されます。
3	border-width: thick, thin, medium	最初のパラメータ (thick) は上の側面に、2 番目のパラメータ (thin) は左と右の側面に、3 番目のパラメータ (medium) は下の側面に適用されます。
4	border-width: thick, thin, medium, thick	最初のパラメータ (thick) は上の側面に、2 番目のパラメータ (thin) は右の側面に、3 番目のパラメータ (medium) は下の側面に、4 番目のパラメータ (thick) は左の側面に適用されます。

ボーダーの各側面のプロパティで指定されたスタイルは、border-width、border-style、border-color、margin の各プロパティで指定されたスタイルよりも優先されます。

例

例 1: この例では、ColdFusion Report Builder での cfreport の使用方法を示しています。

```
<cfquery name="northwindemployees" datasource="localnorthwind">
    SELECT EmployeeID, LastName, FirstName, Title, City, Region, Country
    FROM Employees
    ORDER BY Country, City
</cfquery>

<CFREPORT format="PDF" template="FifthReport.cfr"
    query="#northwindemployees#"/>
```

例 2: この参照専用の例は、Crystal Reports での cfreport の使用方法を示しています。

```
<h3>cfreport Tag<h3>
<p>cfreport lets reports from the Crystal Reports Professional report writer display through a ColdFusion
interface. To run, the tag requires the name of the report. cfreport can also pass information to the report
file displayed, to change the output conditions.</p>
<p>This example would run a report called "monthlysales.rpt " and pass it an
    optional filter condition to show only the information for a subset
    of the report.</p>

<cfreport report = '/reports/monthlysales.rpt'>
    {Departments.Department} = 'International'
</cfreport>

<p>Substitute your report files and filters for this code. cfreport can put Crystal Reports into web
pages.</p>
```

cfreportparam

説明

cfreportparam タグを使用すると、次のタスクを実行できます。

- ColdFusion Report Builder レポート定義に入力パラメータを渡す。
- Report Builder レポート内で定義されているサブレポートとチャートのクエリーデータを上書きする。
- Report Builder サブレポート内で定義されているスタイルを置き換える。

cfreportparam タグは、常に 578 ページの「cfreport」タグの子タグになります。

カテゴリ

データ出力タグ

シンタックス

```
<cfreport template = ...>
    <cfreportparam
        chart = "name of the chart contained in the report or subreport"
        name = "data name"
        query = "query value passed to the chart or subreport"
        series = "ordinal number of a chart series"
        style = "CSS style definition or CSS file pathname"
        subreport = "name of the subreport"
        value = "data value">
</cfreport>
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfreport、『ColdFusion アプリケーションの開発』の Creating Reports with Report Builder、Report Builder オンラインヘルプ

履歴

ColdFusion 8: chart、query、series、subreport、style の各属性が追加されました。

ColdFusion MX 7: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
chart	オプション		レポートまたはサブレポートに含まれるチャートの名前です。この属性の値は、Report Builder レポート内で定義されているチャートの Name プロパティと一致している必要があります。chart 属性を指定する場合は、subreport または name 属性を指定できません。
name	オプション		渡されるデータの変数名です。この属性の値は、Report Builder レポート内で定義されている入力パラメータの名前と一致している必要があります。name 属性を指定する場合は、chart または subreport 属性を指定できません。
query	オプション		サブレポートまたはチャートに渡すクエリー値です。ColdFusion クエリーには、少なくとも Report Builder クエリー内の列がすべて含まれている必要があります。チャートおよびサブレポートを使用する場合、この属性は必須です。
series	オプション	1	クエリーで使用するチャート系列の序数です。この属性は、chart 属性が指定されている場合にのみ有効です。
subreport	オプション		サブレポートの名前です。この属性の値は、Report Builder 内のサブレポートの Name プロパティと一致している必要があります。レポート内のサブレポート名は一意である必要があります。subreport 属性を指定する場合は、chart または name 属性を指定できません。
style	オプション		サブレポートの CSS 形式のスタイルです。CSS ファイルの絶対パス、レポートを基準とした CSS ファイルの相対パス、または有効な CSS 形式で記述された文字列を指定できます。スタイルを有効にするには、Report Builder レポート内の要素に割り当てられている Style Name 属性とこのスタイル名が一致している必要があります。CSS ファイルは Report Builder で生成してエクスポートすることも、テキストエディタで作成することもできます。サポートされている CSS スタイルのリストについては、「スタイルプロパティ」を参照してください。
value	オプション (「説明」を参照)		送信されるデータの値です。value 属性を指定する場合は、name 属性も指定します。chart または subreport 属性を指定する場合は、この属性を指定できません。この属性では、文字列または変数を指定できます。

使用方法

cfreportparam タグでは、次の属性のいずれか 1 つのみを指定できます。

- name
- subreport
- chart

query、subreport、および chart 属性を使用すると、Report Builder のクエリーおよびチャートの情報を実行時に上書きできます。これにより、レポートに埋め込まれたクエリーを変更せずに、CFM ページのサブレポートとチャートのデータをカスタマイズできます。

たとえば、Report Builder で複数のサブレポートを含むマスターレポートを作成し、異なるクエリーを使用して各サブレポートにデータを渡すことができます。Report Builder でクエリーを修正する代わりに、修正したクエリーを ColdFusion 呼び出しページで作成することによって、レポートをカスタマイズできます。ColdFusion クエリーには、少なくとも Report Builder クエリー内の列がすべて含まれている必要があります。

注意：マスターレポートからの引数に依存するサブレポートクエリーは指定できません。代わりに、マスターレポートから引数が渡された場合にサブレポートクエリーを返す CFML 関数や CFC メソッドを定義できます。これらのコードは、ColdFusion がサブレポートを実行するときに呼び出されます。

呼び出し元の CFM ページでは、Report Builder レポート内のサブレポートとチャートに対して cfreportparam タグを指定できます。subreport または chart 属性の値は、Report Builder レポート内で定義されているサブレポートまたはチャートの Name プロパティと一致している必要があります (チャートはサブレポートと同様に扱われます)。

次のコードは、2 つのサブレポートと、2 つのチャート系列で構成される 1 つのチャートを含むマスターレポートを示しています。

```
<cfreport template="myreport.cfr" query="master" format="RTF">
    <cfreportParam subreport="subreport1" query="subquery1">
    <cfreportParam subreport="subreport2" query="subquery2">
    <cfreportParam chart="chart1" series="1" query="chartquery1">
    <cfreportParam chart="chart1" series="2" query="chartquery2">
    <cfreportParam name="ReportDate" value="#DateFormat(Now())#, #TimeFormat(Now())#">
</cfreport>
```

cfreportparam タグは、Report Builder でサブレポートに割り当てられた CSS スタイルを上書きする目的にも使用できます。style 属性は subreport 属性とともに使用します。subreport 属性の値は、Report Builder で定義されたサブレポートの名前と一致している必要があります。次のコードは、マスターレポートに 1 つのスタイルシートを適用し、サブレポートに 2 つの異なるスタイルシートを適用する方法を示しています。

```
<cfreport template="myreport.cfr" style="myStyle.css" format="PDF">
    <cfreportParam subreport="subreport1" style="subreport-style.css">
    <cfreportParam subreport="subreport2" style="subreport-style.css">
</cfreport>
```

詳細については、「CSS (Cascading Style Sheet) の使用」を参照してください。

例

```
 <!--- The following example shows how to override a query in a Report Builder report from the CFM page. The
cfreportparam tag adds the current date and time to the report.--->
<cfquery name="coursedept" datasource="cfdocexamples">
    SELECT Departments.Dept_ID as dDept_ID, Departments.Dept_Name,
     CourseList.Course_ID, CourseList.Dept_ID as cDept_ID,
     CourseList.CorNumber, CourseList.CorName,
     CourseList.CorLevel
    FROM Departments, CourseList
    WHERE Departments.Dept_ID = CourseList.Dept_ID
    ORDER BY CourseList.Dept_ID
</cfquery>

<cfreport format="PDF" template="FourthReport.cfr" query="#coursedept#" overwrite="yes">
<cfreportparam name="ReportTime" value="#DateFormat(Now())#, #TimeFormat(Now())#">
</cfreport>
```

cfrethrow

説明

現在アクティブな例外を再び返します。例外の cfcatch.type および cfcatch.tagContext 変数の値を保持します。

カテゴリ

例外処理タグ、拡張タグ

シンタックス

```
 <cfrethrow>
```

関連項目

cferror、cfthrow、cftry、『ColdFusion アプリケーションの開発』の Handling runtime exceptions with ColdFusion tags

使用方法

このタグは、cfcatch ブロック内で使用します。このタグは、エラーハンドラが検出したエラーをうまく処理できないときに、エラー処理コードとして役立ちます。たとえば、cfcatch type = "any" で DATABASE 例外エラーが検出され、コードが CFX 例外のみを処理するように記述されている場合、詳細を保持したまま例外がハンドラによって再び返され、より高次のハンドラでエラー情報を処理できるようになります。cfthrow タグを使用すると、元の例外のタイプと詳細は失われます。

例

```
 <h3>cfrethrow Example</h3>
<!--- Rethrow a DATABASE exception. --->
<cftry>
    <cftry>
        <cfquery name = "GetMessages" dataSource = "cfdocexamples">
            SELECT*
            FROM Messages
        </cfquery>
    <cfcatch type = "DATABASE">
        <!--- If database signalled a 50555 error, ignore; otherwise, rethrow
            exception. --->
        <cfif cfcatch.sqlstate neq 50555>
            <cfrethrow>
        </cfif>
    </cfcatch>
    </cftry>
<cfcatch>
    <h3>Sorry, this request can't be completed</h3>
    <h4>Catch variables</h4>
    <cfoutput>
        <cfloop collection = #cfcatch# item = "c">
            <br>
            <cfif IsSimpleValue(cfcatch[c])>#c# = #cfcatch[c]#
            </cfif>
        </cfloop>
    </cfoutput>
</cfcatch>
</cftry>
```

cfreturn

説明

コンポーネントメソッドから結果の値を返します。関数の結果として返される式が含められます。

戻り値

式。このタグが呼び出される関数の結果です。

カテゴリ

拡張タグ

シンタックス

```
 <cfreturn
    expr>
```

関連項目

cfargument、cfcomponent、cffunction、cfinvoke、cfinvokeargument、cfobject、cfproperty、『ColdFusion アプリケーションの開発』の Building and Using ColdFusion Components

履歴

ColdFusion MX: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
expr	必須		関数の結果です。任意の型の値です。

使用方法

このタグは、cfscript タグ内の return ステートメントに相当します。戻り値の引数を 1 つ受け入れます。複数の値が返されるようにするには、名前と値のペアを構造体に挿入し、その構造体がこのタグとともに返されるようにします。

このタグの結果の値にアクセスするには、cfinvoke タグの returnVariable 属性の値である変数スコープを使用します。

1 つの関数には、cfreturn タグを 1 つしかコーディングできません。

コード例については、『ColdFusion アプリケーションの開発』の Building and Using ColdFusion Components を参照してください。

例

```
<cfcomponent>
    <cffunction name="getEmp">
         <cfquery name="empQuery" datasource="ExampleApps" >
            SELECT FIRSTNAME, LASTNAME, EMAIL
            FROM tblEmployees
         </cfquery>
         <cfreturn empQuery>
    </cffunction>
    <cffunction name="getDept">
        <cfquery name="deptQuery" datasource="ExampleApps" >
            SELECT *
            FROM tblDepartments
         </cfquery>
         <cfreturn deptQuery>
    </cffunction>
</cfcomponent>
```

cfsavecontent

説明

cfsavecontent タグの生成内容を保存します。保存される内容には、指定した変数での式の評価やカスタムタグの実行の結果が含まれます。

カテゴリ

変数操作タグ

シンタックス

```
<cfsavecontent
    variable = "variable name">
    the content
</cfsavecontent>
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目
『ColdFusion アプリケーションの開発』の Caching parts of ColdFusion pages

属性

属性	必須 / オプション	デフォルト	説明
variable	必須		生成されたタグの内容を保存する変数の名前です。

使用方法
このタグには終了タグが必要です。

このタグを使用して、タブライブラリからの出力を停止することはできません。

例
次の例では、カスタムタグを使用してレポートを生成し、そのレポートを変数 CONTENT に保存します。"report" という単語のインスタンスすべてを "MyCompany Quarterly Report" という句に置き換え、結果を出力します。

```
<cfsavecontent variable="content">
    <CF_OutputBigReport>
</cfsavecontent>
<cfoutput>
    #replace(content, "report", "MyCompany Quarterly Report", "all")#
</cfoutput>
```

cfschedule

説明
ColdFusion スケジューリングエンジンへのプログラムインターフェイスを提供します。ページ出力をスタティックな HTML ページに書き出すオプションとともに、スケジューリングした間隔で CFML ページを実行できます。この機能を利用すると、データベーストランザクションが実行されてページにデータが挿入されるのを待つことなく、レポートなどのデータをパブリッシュするページをスケジューリングすることができます。

カテゴリ
変数操作タグ

シンタックス

```
<cfschedule
    action = "run|update|pause|resume|delete|pauseall|resumeall|list"
    task = "task name"
    endDate = "date"
    endTime = "time"
    file = "filename"
    interval = "seconds"
    operation = "HTTPRequest"
    password = "password"
    path = "path to file"
    port = "port number"
    proxyPassword = "password"
    proxyPort = "port number"
    proxyServer = "host name"
    proxyUser = "user name"
    publish = "yes|no"
    requestTimeOut = "seconds"
    resolveURL = "yes|no"
    isDaily = "yes|no"
    overwrite = "yes|no"
    startDate = "date"
    startTime = "time"
    url = "URL"
    username = "user name">
    group="group1"
    oncomplete="how to handle exception"
    eventhandler="path_to_event_handler"
    onException="refire|pause|invokeHandler"
    cronTime="time"
    repeat="number"
    priority="integer"
    exclude="date|date_range|comma-separated_dates"
    onMisfire = ""
    cluster="yes|no
    mode="server|application"
    retryCount="number"

OR
<cfschedule
    action = "delete"
    task = "task name">

OR

<cfschedule
    action = "run"
```

```
    task = "task name">

OR

<cfschedule
    action = "pauseAll"
    mode = "server|application">

OR

<cfschedule
    action = "resumeAll"
    mode = "server|application">

OR

<cfschedule
    action = "list"
    mode = "server|application"
    result = "res">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfcookie、cfparam、cfregistry、cfsavecontent、cfset

履歴

ColdFusion 10：list、pauseall および resumeall の各アクションが追加されました。また、group、onComplete、eventHandler、onException、cronTime、repeat、priority、exclude、onMisfire、cluster、mode、isDaily、overwrite および retryCount の各属性が追加されました。

ColdFusion MX 6.1: 間隔の計算方法が変更されました。日の長さに標準時と夏時間の切り替えが反映されるようになりました。月の長さは、4 週間ではなくカレンダーの月の長さになりました。スケジューラで閏年を正しく処理できるようになりました。

属性

属性	必須 / オプション	デフォルト	説明
action	必須		• delete：指定されたタスクを削除します。 • update：既存のタスクを更新します。task 属性で指定された名前のタスクが存在しない場合は、タスクを作成します。 • run：指定されたタスクを実行します。 • pause：指定されたタスクを一時停止します。 • resume：指定されたタスクの実行を再開します。 • list：スケジューリングされているすべてのタスクをリスト出力します。 • pauseAll：スケジューリングされている、特定のアプリケーションのすべてのタスクを一時停止します。 • resumeAll：スケジューリングされている、特定のアプリケーションのすべてのタスクを再開します。
task	必須		タスクの名前です。
endDate	オプション		スケジューリングしたタスクが終了する日付です。デフォルトの日付形式は mm/dd/yy です。
endTime	オプション		スケジューリングしたタスクが終了する時刻です (秒)。
file	publish="Yes" の場合は必須		スケジューリングしたタスクのパブリッシュされた出力を保管するファイルの名前です。
interval	action="update" の場合は必須		タスクをスケジューリングする間隔です。 • number of seconds (秒数) • once (1 回) • daily (日に 1 回) • weekly (週に 1 回) • monthly (月に 1 回)
operation	action="update" の場合は必須		スケジューラが実行するオペレーションです。HTTPRequest を指定する必要があります。
overwrite		true	false の場合、タスクが実行されるたびに新しい出力ファイルが作成されます。true の場合、新しい出力ファイルは作成されずに、既存のファイルが上書きされます。
isDaily		true	デフォルトでは、タスクは毎日、指定された時間間隔（例えば、毎日午後 2 時から 3 時の間など）で実行されます。false の場合、タスクはスケジュールに基づいて実行されます。
password	オプション		URL が保護されている場合のパスワードです。
path	publish = "Yes" の場合は必須		パブリッシュされたファイルを入れるディレクトリへのパスです。

属性	必須 / オプション	デフォルト	説明
port	オプション	80	url パラメータで指定したサーバーで使用するポートです。resolveURL="yes" の場合、ポート番号を指定する取得ドキュメントの URL が自動的に変換され、取得ドキュメント内のリンクは保持されます。url 属性のポート値は、この値よりも優先されます。
proxyPassword	オプション		プロキシサーバーに提供するパスワードです。
proxyPort	オプション	80	プロキシサーバー上で使用するポート番号です。
proxyServer	オプション		プロキシサーバーのホスト名または IP アドレスです。
proxyUser	オプション		プロキシサーバーに提供するユーザー名です。
publish	オプション	no	• yes: 結果をファイルに保存します。 • no
requestTimeOut	オプション		デフォルトのタイムアウト時間を延長する場合に使用できます。
resolveURL	オプション	no	• yes: 出力ページ内のリンクを絶対参照に変換します。 • no
startDate	action="update" の場合は必須		スケジューリングしたタスクを最初に実行する日付です。デフォルトの日付形式は mm/dd/yy です。
startTime	action="update" の場合は必須		スケジューリングしたタスクを実行する時刻です。
url	action="update" の場合は必須		実行するページの URL です。
username	オプション		URL が保護されている場合のユーザー名です。
group	オプション	default	スケジューリングしたタスクが属するグループです。
onComplete	オプション	invokeHandler	現在のタスクの完了後に実行されるアクションです。
eventHandler	オプション		タスクの実行中に各種イベントで呼び出される定義済みのメソッドが含まれる CFC ファイルです。CFC には、ITaskEventhandler が実装されている必要があります。 指定するパスは、webroot の相対パスである必要があります。例：schedulerdemo.eventhandler
onException	オプション	invokeHandler	タスクでエラーが発生した場合にどうするかを指定します。オプションは次のとおりです。 • refire：直ちにタスクの実行を試行します。 • pause：タスクのそれ以上の実行を中止します。 • invokeHandler：ユーザーによって定義された eventhandler の onError メソッドを呼び出します。
cronTime	オプション		タスクのスケジュール時刻を cron ジョブシンタックスで指定します。
repeat	オプション	-1	タスクを繰り返す回数です。
priority	オプション	5	優先度を示す整数です。
exclude	オプション		スケジュール期間内で除外する日付または日付範囲のカンマ区切りリストです。

属性	必須 / オプション	デフォルト	説明
onMisfire	オプション		スケジュールされたタスクが誤実行された場合にサーバーがどうするかを指定します。
cluster	オプション	no	yes の場合、タスクをクラスタ設定で実行できます。
mode	オプション	server	タスクがサーバー固有であるか、またはアプリケーション固有であるかです。
retryCount	オプション	3	タスクが失敗した場合の再試行回数です。この値は、0 以上で 3 以下である必要があります。

使用方法

ColdFusion のタスクをスケジューリングするには、このタグと ColdFusion Administrator の [スケジュールされたタスク] ページを使用します。このタグを使用して追加または変更したタスクは Administrator で表示できます。Administrator の [サンドボックス / リソースセキュリティ] ページで、このタグを無効にすることができます。このタグの成功または失敗のステータスは、<ColdFusion のルートディレクトリ >/logs ディレクトリ (マルチサーバー設定および J2EE 設定では <ColdFusion Web アプリケーションのルートディレクトリ >/WEB-INF/cfusion/logs) にある schedule.log ファイルに書き込まれます。

タスクを作成するときに、実行する ColdFusion ページの URL、実行の日付、時刻、頻度、およびタスク出力を HTML ファイルにパブリッシュするかどうかを指定します。出力をパブリッシュする場合は、出力ファイルのパスとファイルを指定します。

毎月 28 日～ 31 日の間のいずれかの日に実行するジョブをスケジュールすると、スケジューラは次のように処理を実行します。

- 毎月のジョブを月の最終日に実行するようにスケジュールすると、スケジュールしたジョブは各月の最後の日に実行されます。たとえば、毎月のジョブを 1 月 31 日から開始するようにスケジュールした場合、ジョブは 1 月 31 日、2 月 28 日または 29 日、3 月 31 日、4 月 30 日という順序で実行されます。
- 毎月のジョブを 29 日または 30 日に実行するようにスケジュールすると、30 日または 31 日まである月では指定した日付に、2 月は最終日にジョブが実行されます。たとえば、毎月のジョブを 1 月 30 日から開始するようにスケジュールした場合、ジョブは 1 月 30 日、2 月 28 日または 29 日、3 月 30 日、4 月 30 日という順序で実行されます。

開始時刻に過去の時刻を指定して、ジョブを 1 回実行するようにスケジュールした場合、そのタスクがまだ実行されていないときは、直ちに実行されます。開始時刻に過去の時刻を指定して、繰り返し発生するジョブをスケジュールすると、ジョブは次に訪れる最も近い指定時刻に実行されます。

スケジューラー設定ファイル（**cf_root**¥lib¥neo-cron.xml）には、スケジュールされたすべてのイベント（クラスタ化されたタスクは除く）が個別のエントリとして書き込まれます。

例

```
 <h3>cfschedule Example</h3>
<!--- This read-only example schedules a task.
        To run the example, remove the comments around the code
        and change the startDate, startTime, url, file, and path attributes
        to appropriate values. --->
<!---
<cfschedule action = "update"
    task = "TaskName"
    operation = "HTTPRequest"
    url = "http://127.0.0.1/playpen/history.cfm"
    startDate = "8/17/09"
    startTime = "12:25 PM"
    interval = "3600"
    resolveURL = "Yes"
    publish = "Yes"
    file = "sample.html"
    path = "c:\inetpub\wwwroot\playpen"
    requestTimeOut = "600">
--->
```

cfscript

説明

cfscript ステートメントを含んでいるコードブロックを囲みます。

カテゴリ

アプリケーションフレームワークタグ、その他のタグ

シンタックス

```
 <cfscript>
    cfscript code here
</cfscript>
```

関連項目

cfinvoke、cfmessagebox、CreateObject、『ColdFusion アプリケーションの開発』の Extending ColdFusion Pages with CFML Scripting

履歴

ColdFusion MX:

- コンポーネントメソッドを呼び出す方法が変更されました。このタグで、CreateObject 関数を使用してコンポーネントメソッドを呼び出せるようになりました。
- 予約語の使い方が変更されました。このタグ内では ColdFusion の予約語は使用できません。
- try ステートメントと catch ステートメントが追加されました。

使用方法

CFScript で処理を実行します。このタグでは、ColdFusion 関数、式、および演算子が使用されます。このタグ内で ColdFusion 変数の読み書きを行うことができます。

CFScript ステートメントや CFML タグに相当する CFScript などに関する CFScript スクリプト言語の詳細については、『ColdFusion アプリケーションの開発』の Extending ColdFusion Pages with CFML Scripting を参照してください。

cfscript タグを使用して、一連の代入ステートメントを囲むことができます。cfscript を使用しない場合は、cfset ステートメントが必要です。

重要：例外の Java クラス名を使用して、このタグ内で cftry/cfcatch ブロックをコーディングする場合は、完全修飾クラス名を指定します。

このタグ内では、一部の ColdFusion 予約語を使用できません。このタグ内に、次の文字列のいずれかで始まる名前のユーザー関数を挿入することはできません。

- cf
- cf_
- _cf
- coldfusion
- coldfusion_
- _coldfusion

elseif 構文を cfscript タグ内で使用することはできません。次のようなコードを使用できます。

```
else if ( condition )
{
...
}
```

キーワード

次の単語はキーワードとして扱われます。

- import
- finally
- component
- interface
- pageencoding

cfscript タグでの例外処理

このタグで例外を処理するには、try および catch ステートメントを使用します。これらは、cftry および cfcatch タグと同じ役割を果たします。各 try ステートメントについて、catch ステートメントが必要です。catch ブロックでは、変数 exceptionVariable に例外タイプが含まれます。この変数は、cfcatch タグのビルトイン変数 cfcatch.Type と同じ役割を果たします。詳細については、『ColdFusion アプリケーションの開発』の Extending ColdFusion Pages with CFML Scripting を参照してください。

cfscript タグを使用した ColdFusion コンポーネントの呼び出し

CFScript では、CreateObject 関数を使用してコンポーネントメソッドを呼び出します。

次の例では、cfscript タグを使用したコンポーネントオブジェクトの呼び出し方法を示します。この例では、指定された引数を使用します。

```
<cfscript>
quote = CreateObject( "component", "nasdaq.quote" ) ;
<!--- Invocation using ordered arguments. --->
res = quote.getLastTradePrice( "macr" ) ;
</cfscript>
```

次の例では、コンポーネントオブジェクトの呼び出し時に cfscript タグ内で属性コレクションを使用してパラメータを渡す方法を示します。属性コレクションは構造体です。この構造体では、各キーはパラメータ名に対応し、各値は対応するキーに渡されるパラメータの値となります。

```
<cfscript>
    stArgs = structNew();
    stArgs.zipcode = "55987";
</cfscript>
...
<cfinvoke
    webservice = "http://www.xmethods.net/sd/2001/TemperatureService.wsdl"
    method = "getTemp"
    argumentCollection = "#stArgs#"
    returnVariable = "aTemp">
<cfoutput>The temperature at zip code 55987 is #aTemp#</cfoutput>
```

この例では、構造体は cfscript ブロックとして作成されますが、この構造体は任意の ColdFusion メソッドを使用して作成できます。

cfscript タグを使用した Web サービスの利用

次の例では、cfscript タグを使用した Web サービスの利用方法を示します。Web サービスに接続するには、CreateObject 関数を使用します。

```
<cfscript>
    ws = CreateObject("webservice",
                      "http://www.xmethods.net/sd/2001/TemperatureService.wsdl");
    xlatstring = ws.getTemp("55987");
    writeoutput(xlatstring);
</cfscript>
```

詳細については、『ColdFusion アプリケーションの開発』の Using Web Services を参照してください。

例

```
<p>This simple example shows variable declaration and manipulation.
<cfif IsDefined("form.myValue")>
    <cfif IsNumeric(form.myValue)>
        <cfset x = form.myValue>
        <cfscript>
            y = x;
            z = 2 * y;
            StringVar = form.myString;
        </cfscript>
    <cfoutput><p>twice #x# is #z#.
        <p>Your string value was: <b><I>#StringVar#</i></b></cfoutput>
<cfelse>
```

cfsearch

説明

Solr コレクションを検索します。

cfsearch タグによって検索結果が返されるようにするには、コレクションを作成し、これにインデックスを作成しておく必要があります。

コレクションは、次の方法で作成できます。

- 87 ページの「cfcollection」タグを使用します。
- ColdFusion Administrator を使用します。

コレクションには、次の方法でインデックスを作成できます。

- ColdFusion で、353 ページの「cfindex」タグを使用します。
- ColdFusion Administrator で、cfindex タグを呼び出します。

詳細については、『ColdFusion アプリケーションの開発』の Building a Search Interface を参照してください。

カテゴリ

拡張タグ

シンタックス

```
cfsearch supports script style syntax:

new search().search(name="<search_name>", collection="<collection_name>");


cfsearch supports script style syntax:

 <cfsearch
     collection = "collection name"
     name = "search name"
     category = "category[,category2,...]"
     categoryTree = "tree location"
     contextBytes = "number of bytes"
     contextHighlightBegin = "HTML string"
     contextHighlightEnd = "HTML string"
     contextPassages = "number of passages"
     criteria = "search expression"
     language = "language"
     maxRows = "number"
     orderBy = "rank_order"
     previousCriteria = "criteria"
     startRow = "row number"
     status = ""
     suggestions = "suggestion option"
     type = "criteria">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfcollection、cfexecute、cfindex、cfobject、cfreport、cfwddx

履歴

ColdFusion 10：新しい orderBy 属性

ColdFusion 9: Solr 検索エンジンを使用できるようになりました。

ColdFusion MX 7:

- category、categoryTree、status、suggestions、contextPassages、contextBytes、contextHighlightBegin、contextHighlightEnd、previousCriteria の各属性が追加されました。
- 結果列として、author、category、categoryTree、context、rank、size、recordsSearched、type が追加されました。
- ステータス構造体およびその構造体に関連付けられているキーに関する情報が追加されました。
- 個々の K2 サーバーおよび k2server.ini ファイルへの参照が削除されました。
- 未登録のコレクションへの参照が削除されました。

- 外部コレクションへの参照が削除されました。ColdFusion MX では、Verity 検索サービスを使用してすべてのコレクションを管理します。
- cflock の推奨事項が変更されました。cfsearch タグを cflock タグで囲むことは推奨事項ではなくなりました。

ColdFusion MX:

- external 属性は非推奨になりました。今後のリリースではこのタグは機能せずエラーを引き起こす可能性があります。各コレクションが内部コレクションか外部コレクションであるかは自動的に検出され、この情報は保管されます。このタグでは、絶対 (完全修飾) コレクションパス名およびマップ済みコレクション名がサポートされます。
- クエリー結果の動作が変更されました。cfindex タグでは、cfsearch オペレーションのクエリー結果にインデックスを付けることができます。
- Verity オペレーションの動作が変更されました。ColdFusion で、Acrobat PDF ファイルに対する Verity オペレーションがサポートされるようになりました。
- 複数のコレクションの動作が変更されました。このタグを使用して、複数のコレクションを検索できます。複数のコレクションの検索では、K2 サーバーで登録されたコレクションと他の方法で登録されたコレクションを同時に検索することはできません。
- コレクションのネーミング規則が変更され、スペースを含むコレクション名が使用できるようになりました。
- 次のサポートが変更されました。このタグは、Verity 2.6.1、LinguistX、および ICU ロケールをサポートしています。
- 返す例外が変更されました。このタグでは SEARCHENGINE 例外を返すことができます。

属性

属性	必須 / オプション	デフォルト	説明
name	必須		検索クエリーの名前です。
collection	必須		コレクション名です。カテゴリ検索を実行 (category または categoryTree を指定) する場合以外は、複数のコレクションを指定できます。
category	オプション		検索の条件として使用する、カンマで区切られたカテゴリのリストです。このレベルを指定しても、コレクションでカテゴリが有効でない場合、ColdFusion は例外を返します。
categoryTree	オプション		カテゴリの階層ツリー上の、検索を開始する位置です。ColdFusion では、このレベル以下を対象に検索が行われます。このレベルを指定しても、コレクションでカテゴリが有効でない場合、ColdFusion は例外を返します。category 属性と併用できます。
criteria	オプション		検索条件です。type 属性のシンタックスルールに従います。criteria 属性で大文字と小文字が混在したエントリを渡すと、検索では大文字と小文字が区別されます。大文字だけまたは小文字だけのエントリを渡すと、大文字と小文字は区別されません。Solr のシンタックスおよび区切り文字のルールに従ってください。『ColdFusion アプリケーションの開発』の Solr search support を参照してください。
contextBytes	オプション Solr	300	コンテキスト要約に返される最大バイト数です。
contextHighlightBegin	オプション Solr		コンテキスト要約で検索語の先頭に追加する HTML です。この属性を contextHighlightEnd とともに使用して、コンテキスト要約で検索語を強調表示します。
contextHighlightEnd	オプション Solr		コンテキスト要約で検索語の末尾に追加する HTML です。この属性を contextHighlightBegin とともに使用して、コンテキスト要約で検索語を強調表示します。
contextPassages	オプション Solr	0	Solr がコンテキスト要約（結果の context 列）に返す句および文の数です。デフォルトは 0 で、コンテキスト要約は無効です。

属性	必須 / オプション	デフォルト	説明
language	オプション	english	非推奨です。この属性は、無視されるようになりました。コレクションの言語は、検索を実行するために使用されます。
maxRows	オプション	all	クエリー結果に返される行の最大数です。
orderBy	オプション		カスタムフィールド列のランク順をソートします。デフォルトでは、昇順でソートされます。
previousCriteria	オプション		検索結果の既存セットの結果セット名です。検索エンジンは、前の検索スコアやランクに関係なく、criteria の結果セットを検索します。この属性を使用して、結果セット内で検索を実装します。
startRow	オプション	1	取得する最初の行の行番号です。
status	オプション		ColdFusion での検索情報の配置先となる構造体変数の名前を指定します。検索情報には、代替検索条件の提案 (スペルの修正) などが含まれます。この構造体のキーのリストについては、「ステータス構造体キー」を参照してください。
suggestions	オプション	never	スペルの誤りがある単語について検索エンジンがスペル提案を返すかどうかを指定します。次のいずれかのオプションを使用します。 • Always: 常にスペル提案を返します。 • Never: スペル提案を返しません。 • 正の整数 : 検索で取得されたドキュメントの数が指定値以下の場合、スペル提案を返します。 提案データを取得する際には、パフォーマンスが若干低下します。
type	オプション	simple	エンジンで条件を処理する際に使用するパーサーを指定するために使用します。 • simple: STEM および MANY 演算子を暗黙的に使用します。 • explicit: 演算子を明示的に呼び出す必要があります。Bool_Plus とも呼ばれます。 • internet: 大部分が WYSIWYG (what-you-see-is-what-you-get) ドキュメントであるドキュメントの場合に指定します。Internet_advanced とも呼ばれます。 • internet_basic: 検索時間を最小限にします。 • natural: QBE (Query By Example) パーサーを指定します。FreeText とも呼ばれます。

使用方法

cfsearch タグは、cfoutput タグ内で参照可能な列を持つクエリーオブジェクトを返します。たとえば、次のコードでは、"filming" または "filmed" に完全一致する用語の検索が指定されます。

```
<cfsearch
    name = "mySearch"
    collection = "myCollection"
    criteria = '<WILDCARD>`film{ing,ed}`'
    type="explicit"
    startrow=1
    maxrows = "100">
<cfdump var = "#mySearch#>
```

この例では、一重引用符（'）およびバックティック（`）が区切り文字として使用されています。

検索のパフォーマンスを最適化するには、常に maxrows 属性を指定し、使用するアプリケーションの要件に応じた値に設定してください。値を 300 未満に設定すると、最適なパフォーマンスが得られます。

406 ページの「cflock」タグをこのタグとともに使用しないでください。Solr にはロック機能が用意されています。406 ページの「cflock」タグを使用すると、検索のパフォーマンスが低下します。

cfsearch タグの結果列

変数	説明
context	デフォルトでボールドで強調表示された検索語を含むコンテキスト要約です。これは、contextpassages 属性を 0 よりも大きい数値に設定した場合に有効になります。
url	コレクションへの挿入に使用される cfindex タグ内の URLpath 属性の値です。
key	コレクションへの挿入に使用される cfindex タグ内の key 属性の値です。
title	PDF ドキュメントや Office ドキュメントのタイトルなど、コレクションへの挿入に使用される cfindex タグ内の title 属性の値です。タイトルがドキュメントから抽出されない場合は、cfindex の title 属性の値が各行に使用されます。
score	検索条件に基づいた、ドキュメントの関連性の値です。
custom1、custom2、custom3、custom4	コレクションへの挿入に使用される cfindex タグ内のカスタムフィールドの値です。
size	インデックスドキュメントのバイト数です。
rank	検索結果におけるこのドキュメントのランクです。
author	可能な場合に HTML、Office、および PDF ドキュメントから抽出されます。
type	ドキュメントの MIME タイプです。
category	このドキュメントのインデックス作成時に指定されたカテゴリのリストです。
categoryTree	このドキュメントのインデックス作成時に指定された、カテゴリの階層ツリーまたはカテゴリのシリアルリストです。単一のツリーだけが返されます。
summary	cfindex によって作成される自動要約の内容です。
recordCount	レコードセットで返されるレコードの数です。
currentRow	cfoutput により処理されている現在の行です。
columnList	レコードセット内の列名のリストです。
recordsSearched	検索されたレコードの数です。これは、レコードセット内の各行について同じ値です。ステータス構造体内の searched キーに対応します。

ステータス構造体キー

変数	説明
found	検索条件に一致するテキストが含まれているドキュメントの数
searched	検索されたドキュメントの数。検索結果の recordsSearched 列に対応します。
time	検索に要した時間（ミリ秒単位）です。
suggestedQuery	より適切な結果が得られる可能性がある代替クエリです。検索語のスペル修正などが行われます。suggestions というタグ属性で指定した条件が満たされた場合にのみ提示されます。
keywords	基本設定の順序に最大 5 つの代替用語を設定可能な配列へのキーワードとなる検索語を含む構造体です。suggestions 属性の条件が満たされる場合にのみ存在します。
keywordScore	keywords の場合と同じ形式の構造体です。

例

```
 <!--- #1 (TYPE=SIMPLE) ------------------------------>
<cfsearch
    name="name"
    collection="snippets,syntax,snippets"
    criteria="example"
    maxrows = "100">
<p>
<cfoutput>Search Result total =#name.RecordCount# </cfoutput><br>
<cfoutput>
    url=#name.url#<br>
    key=#name.key#<br>
    title=#name.title#<br>
    score=#name.score#<br>
    custom1=#name.custom1#<br>
    custom2=#name.custom2#<br>
    summary=#name.summary#<br>
    recordcount=#name.recordcount#<br>
    currentrow=#name.currentrow#<br>
    columnlist=#name.columnlist#<br>
    recordssearched=#name.recordssearched#<br>
</cfoutput>
<cfdump var = #name#>
<br>

<!--- #2 (TYPE=EXPLICIT) ------------------------------>
<cfsearch
    name = "snippets"
    collection = "snippets"
    criteria = '<wildcard>`film{ing,ed}`'
    type="explicit"
    startrow=1
    maxrows = "100">
<cfoutput
    query="snippets">
    url=#url#<br>
    key=#key#<br>
    title=#title#<br>
    score=#score#<br>
    custom1=#custom1#<br>
    custom2=#custom2#<br>
    summary=#summary#<br>
    recordcount=#recordcount#<br>
    currentrow=#currentrow#<br>
    columnlist=#columnlist#<br>
    recordssearched=#recordssearched#<br>
</cfoutput>
<cfdump var = #snippets#>
<br>
```

```
<!--- #3 (search by CF key) ----------------------------->
<cfsearch
    name = "book"
    collection = "custom_book"
    criteria = "cf_key=bookid2"
    maxrows = "100">
<cfoutput>
    url=#book.url#<br>
    key=#book.key#<br>
    title=#book.titleE#<br>
    score=#book.score#<br>
    custom1=#book.custom1#<br>
    custom2=#book.custom2#<br>
    summary=#book.summary#<br>
    recordcount=#book.recordcount#<br>
    currentrow=#book.currentrow#<br>
    columnlist=#book.columnlist#<br>
    recordssearched=#book.recordssearched#<br>
</cfoutput>
<cfdump var = #book#>
```

cfselect

説明

ドロップダウンリストボックスのフォームコントロールを構築します。cfform タグ内で使用します。ドロップダウンリストのエントリは、クエリーまたは HTML option タグを使用して渡すことができます。

カテゴリ

フォームタグ

シンタックス

```
<cfselect
    name="name" bind="bind expression"
    bindAttribute="attribute name"
    bindOnLoad="yes|no"
    display="text" editable="yes|no"
    enabled="yes|no"
    group="query column name" height="number of pixels"
    id="HTML id"
    label="label" message="text" multiple="yes|no"
    onBindError="JavaScript function name"
    onChange="JavaScript or ActionScript"
    onClick="JavaScript function name"
    onError="JavaScript"
    onKeyDown="JavaScript or ActionScript"
    onKeyUp="JavaScript or ActionScript"
    onMouseDown="JavaScript or ActionScript"
    onMouseUp="JavaScript or ActionScript"
    query="query name" queryPosition="above|below"
    required="yes|no"
    selected="value or list" size="integer" sourceForTooltip="URL"
    style="style specification" tooltip="text"
    value="text" visible="yes|no"
    width="number of pixels">

zero or more HTML option tags
```

</cfselect>

一部の属性は、特定の表示形式のみに適用されます。詳細については、「属性」の表を参照してください。

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfapplet、cfcalendar、cfform、cfformgroup、cfformitem、cfgrid、cfinput、cfslider、cftextarea、cftree、『ColdFusion アプリケーションの開発』の Introduction to Retrieving and Formatting Data および Using Ajax User Interface Components and Features

履歴

ColdFusion 8:

- HTML 形式フォームでバインディングがサポートされるようになりました (bind、bindAttribute、bindOnLoad、onBindError 属性など)。
- HTML 形式フォームでツールヒントがサポートされるようになりました (sourceForTooltip 属性など)。

ColdFusion MX 7:

- データベースフィールドに値ではなくスペースが含まれている場合、クエリーを使用して cfselect タグにデータを渡すと、ColdFusion MX 6.1 の場合と同様に、cfform タグの必須フィールドのエラー処理が実行されません。
- selected 属性に対して複数の値を指定できるようになりました。
- passthrough 属性は非推奨になりました。HTML select タグのすべての属性が直接サポートされるようになりました。
- 接頭辞 on が付いた属性が追加されました。
- enabled、group、height、label、queryPosition、tooltip、visible、および width の各属性が追加されました。

属性

次の表は、ColdFusion が直接使用する属性のリストです。このリストに記載されていない HTML select タグの属性もすべてサポートされ、ブラウザに直接渡されます。

注意："Flash の場合のみ " と記載されている属性は、ColdFusion に付属するスキンでは処理されません。ただし、これらの属性は生成された XML に含まれます。

属性	必須 / オプション、 形式	デフォルト	説明
name	必須、 すべて		選択フォーム要素の名前です。
bind	オプション、 HTML		コントロールの属性をダイナミックに設定するバインド式です。詳細については、「使用方法」を参照してください。
bindAttribute	オプション、 HTML	option の値を設定	bind 属性を使用して値を設定する HTML タグ属性を指定します。ColdFusion 固有の属性ではなく、ブラウザの HTML DOM ツリーに含まれる属性のみを指定できます。 bind 属性がない場合は無視されます。
bindOnLoad	オプション、 HTML	no	フォームを最初にロードするときに bind 属性の式を実行するかどうかを指定するブール値です。bind 属性がない場合は無視されます。

属性	必須 / オプション、 形式	デフォルト	説明
display	オプション、 すべて	value 属性の値	各リスト要素の表示ラベルに使用するクエリー列です。query 属性とともに使用します。
editable	オプション、 Flash	no	コントロールの内容を編集可能にするかどうかを指定するブール値です。
enabled	オプション、 Flash	yes	コントロールを有効にするかどうかを指定するブール値です。無効なコントロールはライトグレーで表示されます。disabled 属性の逆です。
group	オプション、 HTML および XML		ドロップダウンリスト内の項目を 2 つのレベルで構成された階層型のリストにグループ化するために使用するクエリー列です。
height	オプション、 Flash		コントロールの高さをピクセル単位で指定します。
id	オプション、 HTML	name 属性の値	コントロールの HTML ID です。
label	オプション、 Flash および XML		Flash または XML 形式のフォーム内でコントロールの横に配置するラベルです。
message	オプション、 すべて		required="yes" で、何も選択されていないときに表示されるメッセージです。
multiple	オプション、 すべて	no	• yes: ドロップダウンリスト内の複数の要素を選択できます。 • no
onBindError	オプション、 HTML	「説明」を参照	バインド式の評価でエラーが発生した場合に実行する JavaScript 関数の名前です。この関数は 2 つの属性 (HTTP ステータスコードとメッセージ) を取る必要があります。発生したエラーが HTTP エラーでない場合、ステータスコードは -1 です。 この属性を省略し、(1447 ページの「ColdFusion.setGlobalErrorHandler」関数を使用して) グローバルエラーハンドラを指定すると、エラーメッセージが表示されます。それ以外の場合は、デフォルトのエラーポップアップが表示されます。
onChange	オプション、 すべて		ユーザーのアクションに応じてコントロールに変化があったときに実行される JavaScript (HTML/XML の場合) または ActionScript (Flash の場合) です。
onClick	オプション、 HTML および XML		ユーザーがコントロールをクリックしたときに実行される JavaScript です。
onError	オプション、 HTML および XML		検証に失敗した場合に実行するカスタム JavaScript 関数です。
onKeyDown	オプション、 すべて		ユーザーがコントロール内でキーボードのキーを押したときに実行される JavaScript (HTML/XML の場合) または ActionScript (Flash の場合) です。
onKeyUp	オプション、 すべて		ユーザーがコントロール内でキーボードのキーを離したときに実行される JavaScript (HTML/XML の場合) または ActionScript (Flash の場合) です。
onMouseDown	オプション、 すべて		ユーザーがコントロール内でマウスボタンを離したときに実行される JavaScript (HTML/XML の場合) または ActionScript (Flash の場合) です。

属性	必須 / オプション、 形式	デフォルト	説明
onMouseUp	オプション、 すべて		ユーザーがコントロール内でマウスボタンを押したときに実行される JavaScript (HTML/XML の場合) または ActionScript (Flash の場合) です。
query	オプション、 すべて		ドロップダウンリストにエントリを渡すクエリーの名前です。
queryPosition	オプション、 すべて	above	クエリーを使用してオプションリストにエントリを渡し、HTML option 子タグを使用して追加エントリを指定する場合に、option タグのエントリを基準としてクエリーのエントリをどこに配置するかをこの属性で指定します。 • above: クエリー項目を options 項目の上に配置します。 • below: クエリー項目を options 項目の下に配置します。
required	オプション、 すべて	no	メモ : size 属性を省略した場合や、値を 1 に設定した場合は、表示されている項目が常にブラウザで送信されるので、この属性の設定は無効になります。この問題は、最初の option タグで value="" (引用符の間に空白文字があることに注意してください) を指定することで回避できます。この解決策は Flash フォームの場合にのみ適用されます。HTML フォームまたは XML フォームの場合には適用されません。 • yes: いずれかのリスト要素が選択されていないとフォームを送信できません。 • no
selected	オプション、 すべて		選択リストであらかじめ選択しておくオプション値です (複数指定可)。複数の値を指定するには、カンマ区切りのリストを使用します。この属性は、選択リスト項目がクエリーから生成される場合のみ適用されます。cfform タグの preserveData 属性でこの値を上書きできます。
size	オプション、 すべて	1	一度に表示するエントリの数です。デフォルト値では、ドロップダウンリストが表示されます。それ以外の値を指定すると、size の数のエントリを一度に表示するリストボックスが表示されます。
sourceForTooltip	オプション、 HTML		ツールヒントとして表示するページの URL です。このページにはヒントの内容と書式を制御するための CFML および HTML マークアップを含めることができ、ヒントにはイメージを含めることができます。 この属性を指定すると、ヒントのロード中にアニメーションのアイコンと「ロード中 ...」というテキストが表示されます。
style	オプション、 すべて		HTML または XML 形式フォームの場合、style 属性はブラウザまたは XML に渡されます。 Flash 形式では、対応する Flash 要素について、Flex で使用するものと同じシンタックスおよびコンテンツを使用する CSS 形式のスタイル仕様でなければなりません。
tooltip	オプション、 Flash、HTML		マウスポインタをコントロールの上に置いたときに表示されるテキストです。このテキストには HTML マークアップを含めることもできます。 sourceForTooltip 属性が指定されている場合は無視されます。
value	オプション、 すべて		各リスト要素の値に使用するクエリー列です。query 属性とともに使用します。
visible	オプション、 Flash	yes	コントロールを表示するかどうかを指定するブール値です。非表示のコントロールには、空の表示領域が割り当てられます。
width	オプション、 Flash		コントロールの幅をピクセル単位で指定します。

使用方法

このタグを正しく動作させるには、JavaScript 対応のブラウザを使用する必要があります。

このタグには終了タグが必要で、HTML option 子タグおよび optgroup 子タグを含めることができます。

ポストバックが繰り返されてもリストボックス内で選択されている項目が変更されないようにするには、クエリーから生成されるリストとともに cfform タグの preserveData 属性を使用します (この方法は、クエリーから渡されるデータに対してのみ使用できます)。

cfform の preserveData 属性が yes で、フォームが同じページにポストバックされ、コントロールのエントリがクエリーから渡される場合は、cfselect コントロールで選択されて送信されたエントリが Selected 属性の代わりに使用されます。通常の HTML option タグによってエントリが渡されるコントロールの場合は、Selected 属性を適切な option タグに動的に追加する必要があります。

group オプションを指定すると、SQL GROUP BY シンタックスを使用してクエリーが生成され、group 列のフィールド値の下にあるインデントされたリストに各グループの value 列エントリが配置されます。このオプションを指定すると、group 列のエントリごとに HTML optgroup タグが生成されます。

cfselect タグ本文内の各 HTML option タグは、</option> 終了タグで閉じる必要があります。このタグを閉じないで queryPosition="below" を指定した場合、クエリーの最初の項目がリストに表示されないことがあります。

bind 属性を使用すると、cfselect 属性をダイナミックに設定できます。多くの場合は、ユーザーがフォームに入力した値に基づいてオプションリストをダイナミックに作成するために使用します。たとえば、bind 属性を使用すると、州名の cfselect コントロールからユーザーが選択した項目に基づいて都市名のオプションリストを作成できます。

bind 属性を使用して選択リストに値を渡す場合は、次のいずれかの形式で選択値を返す関数または URL を指定します。

- 2 次元配列 (各配列行の最初の要素はオプション値で、2 番目の要素はオプションリストに表示されるテキストになります)。
- bind 属性で CFC 関数を指定する場合はクエリー、URL を指定する場合はクエリーの JSON 表現。このクエリーには、cfselect タグの value 属性および display 属性の値と一致する名前を持つ列が含まれている必要があります。その他の列を返すこともできますが、クライアントサイドのコードではそれらの列を使用できません。CFC 関数を呼び出す場合は、クエリーを JSON 形式に変換する必要はありません。ColdFusion によって自動的に変換されます。

  この形式を使用するには、value 属性を指定します。display 属性を省略した場合、クエリー内で値を含むことができる列は 1 つのみです。これらの値は表示テキストとしても使用されます。

バインディングの詳細については、『ColdFusion アプリケーションの開発』の Binding data to form fields を参照してください。

詳細については、250 ページの「cfform」タグのエントリを参照してください。

例

例 1: データバインディングなし

次の例では、部門別にグループ化された全従業員のリストから従業員名を選択します。選択した名前とその従業員の電子メールアドレスが表示されます。全従業員のデータを取得するオプションもあります。

```
 <!--- Get the employee names from the database. --->
<!--- Use SQL to create a Name field with first and last names. --->
<cfquery name = "GetAllEmployees" dataSource = "cfdocexamples"
        cachedwithin="#createTimeSpan(0,1,0,0)#">
    SELECT Emp_ID, EMail, Phone, Department, FirstName, LastName,
        FirstName ||' '
        ||lastName as Name
    FROM Employees
    GROUP BY Department, Emp_ID, EMail, Phone, FirstName, LastName, FirstName
</cfquery>

<h2>cfselect Example</h2>
<!-- The cfif statement is true if the form was submitted.
    Show the selected names. --->
<cfif IsDefined("form.employeeid")>
    <!--- The form was submitted. --->
    <h3>You Selected the following employees</h3>
    <cfif form.employeeid IS "">
        <!--- Select All option was selected. Show all employees. --->
        <cfoutput query="GetAllEmployees">
            #name#<br>
            Email: #email#<br><br>
        </cfoutput>
    <cfelse>
        <!---
            Use a query of queries to get the data for the selected users.
            Form.employeeid is a comma-delimited list of selected employee IDs.
        --->
        <cfquery name = "GetSelectedEmployees" dbtype="query">
            SELECT Emp_ID, EMail, Phone, Department, FirstName, LastName,
                FirstName
                ||' ' ||lastName as Name
            FROM GetAllEmployees
            WHERE Emp_ID in (#form.employeeid#)
        </cfquery>
        <!--- Display the names and e-mail addresses from the query. --->
        <cfoutput query="GetSelectedEmployees">
            #firstName# #lastName#<br>
            Email: #email#<br>
            <br>
        </cfoutput>
    </cfif>
</cfif>

<!--- The cfform tag posts back to the current page. --->
<h3>Select one or more employees</h3>
<cfform action = "#CGI.SCRIPT_NAME#">
    <!---
```

```
            Use cfselect to present the query's LastName column,
            grouped by department.
            Allow Multiple selections.
        --->
    <cfselect
        name = "employeeid"
        size = "15"
        multiple="yes"
        required = "Yes"
        message = "Select one or more employee names"
        query = "GetAllEmployees"
        group="Department"
        display ="name"
        value ="emp_id"
        queryPosition="Below">
        <!--- Add an option to select all employees. --->
        <option value = "">Select All</option>
    </cfselect><br><br>
    <input type="Submit">
</cfform>
```

例 2: データバインディングあり

次の例では、ユーザーが州名を選択した後、バインディングを使用して、都市名コントロールのオプションリストに値を渡します。この例では、California 州と New Jersey 州の都市名データのみを使用します。

この例の中で最も簡単な部分は CFML ページです。このページは次の行で構成されています。

```
 <html>
<head>
</head>
<body>
<cfform name="mycfform">
    <!---
        The States selector.
        The bindonload attribute is required to fill the selector.
    --->
    <cfselect name="state" bind="cfc:bindFcns.getstates()" bindonload="true">
        <option name="0">--state--</option>
    </cfselect>
    <cfselect name="city" bind="cfc:bindFcns.getcities({state})">
        <option name="0">--city--</option>
    </cfselect>
</cfform>
</body>
</html>
```

BinFcns CFC には、州名を取得する getstates 関数、都市名を取得する getcities 関数、XML ファイルを解析して州と都市の情報を取得する getXmlData 内部関数があります。この CFC の内容は次のとおりです。

```
<cfcomponent>
    <cffunction name="getXmlData" output="true">
        <cfset var xmlData = "">
        <cffile action="read" file="#expandpath('.')#\states.xml"
            variable="xmlData">
        <cfset xmlData = XmlParse(xmlData)>
        <cfreturn xmlData>
    </cffunction>

    <cffunction name="getstates" access="remote">
        <cfset state = arraynew(2)>
        <cfset xmlData = getXmlData()>
        <cfset numStates = 0>
        <cfset state[1][1] = "0">
        <cfset state[1][2] = "--state--">
        <cfset numStates = ArrayLen(xmlData.states.XmlChildren)>
        <cfloop from="1" to="#numStates#" index="j">
            <cfset state[j+1][1] =
                ltrim(xmlData.states.state[j].XmlAttributes.abr)>
            <cfset state[j+1][2] = ltrim(xmlData.states.state[j].name.xmlText)>
        </cfloop>
        <cfreturn state>
    </cffunction>

    <cffunction name="getcities" access="remote">
        <cfargument name="state" required="yes">
        <cfset var city = arraynew(2)>
        <cfset var xmlData = getXmlData()>
        <cfset var numStates = 0>
        <cfset var numCities = 0>
<cflog text="In getcities">
        <cfset city[1][1] = "0">
        <cfset city[1][2] = "--city--">
        <cftry>
            <cfset numStates = ArrayLen(xmlData.states.XmlChildren)>
            <cfloop from="1" to="#numStates#" index="j">
                <cfif xmlData.states.state[j].XmlAttributes.abr eq state>
                    <cfset numCities =
                        ArrayLen(xmlData.states.state[j].cities.XmlChildren)>
                    <cfloop from="1" to="#numCities#" index="k">
                        <cfset city[k+1][1] =
ltrim(xmlData.states.state[j].cities.city[k].XmlAttributes.name)>
                        <cfset city[k+1][2] =
ltrim(xmlData.states.state[j].cities.city[k].XmlAttributes.name)>
                    </cfloop>
                    <cfbreak>
                </cfif>
            </cfloop>
        <cfcatch type="any">
            <!--- Do nothing. --->
        </cfcatch>
        </cftry>
        <cfreturn city>
    </cffunction>

</cfcomponent>
```

states.xml ファイルには次のコードが含まれています。コードを短くするために、都市を持つ州は 2 つのみとし、合計 4 つの州のみがリストされています。

```
<states>
    <state abr="NJ">
        <name>New Jersey</name>
        <cities>
            <city name="Edison" />
            <city name="Rahway" />
            <city name="Atlantic City" />
            <city name="Hoboken" />
            <city name="Jersey City" />
            <city name="Newark" />
            <city name="Trenton" />
            <city name="Union City" />
        </cities>
    </state>
    <state abr="CA">
        <name>California</name>
        <cities>
            <city name="Anaheim" />
            <city name="Beverly Hills" />
            <city name="Elk Grove" />
            <city name="Fairfield" />
            <city name="Fremont" />
            <city name="Indian Wells" />
            <city name="Long Beach" />
        </cities>
    </state>
    <state abr="ME">
        <name>Maine</name>
    </state>
    <state abr="MA">
        <name>Massachusetts</name>
    </state>
</states>
```

cfservlet

説明
このタグは非推奨となっています。このタグは JRun エンジンで Java サーブレットを実行します。

ColdFusion と同じサーバー上で動作するサーブレットにアクセスするには、次のようなコードを使用します。このコードの **path** で、サーブレットや JSP などを指定します。

```
GetPageContext().include(path)
GetPageContext().forward(path)
```

詳細については、『JSP PageContext API』または『Servlet RequestDispatcher API』を参照してください。

履歴
ColdFusion MX: このタグは非推奨になりました。今後のリリースでは、動作せずにエラーとなる可能性があります。

cfservletparam

説明
このタグは非推奨となっています。

cfservlet タグの子タグです。データをサーブレットに渡します。cfservlet ブロック内の各 cfservletparam タグによって、データの各項目がサーブレットに渡されます。

ColdFusion と同じサーバー上で動作するサーブレットにアクセスするには、次のようなコードを使用します。このコードの **path** で、サーブレットや JSP などを指定します。

```
GetPageContext().include(path)
GetPageContext().forward(path)
```

詳細については、『JSP PageContext API』または『Servlet RequestDispatcher API』を参照してください。

履歴

ColdFusion MX: このタグは非推奨になりました。今後のリリースでは、動作せずにエラーとなる可能性があります。

cfset

説明

ColdFusion の値を設定します。変数が存在しない場合は変数を作成し、値を割り当てます。また、関数の呼び出しにも使用されます。

カテゴリ

変数操作タグ

シンタックス

```
<cfset
    var variable_name = expression>
```

関連項目

cfcookie、cfparam、cfregistry、cfsavecontent、cfschedule、『ColdFusion アプリケーションの開発』の Elements of CFML

属性

属性	必須 / オプション	デフォルト	説明
variable_name	必須		変数です。
var	オプション		キーワードです。値は取りません。変数が関数に対してローカルであることを示します。変数は、関数の現在の呼び出しの間だけ存在します。

使用方法

アプリケーションでは cfset タグをさまざまな方法で使用できます。

関数の呼び出し

cfset タグを使用して関数を呼び出すときに、その関数が値を返さない場合や、関数から返される値を使用する必要がない場合は、関数の戻り値を変数に代入する必要はありません。たとえば、次の例は Application スコープから MyVariable 変数を削除する、有効な ColdFusion cfset タグです。

```
<cfset StructDelete(Application, "MyVariable")>
```

配列

次の例では、新しい配列を変数 months に割り当てます。

```
<cfset months = ArrayNew(1)>
```

次の例では、配列 Scores の長さに解決される変数 Array_Length を作成します。

```
<cfset Array_Length = ArrayLen(Scores)>
```

次の例では、配列 months 内のインデックス位置 2 に値 February を割り当てます。

```
<cfset months[2] = "February">
```

ダイナミックな変数名

次の例では、変数名自体が変数となっています。

```
<cfset myvariable = "current_value">
<cfset "#myvariable#" = 5>
```

関数のローカル変数

var キーワードは、その変数が、cffunction タグで定義された関数の本文内でのみ利用可能であることを指定します。関数の 1 つの呼び出しで設定された変数の値は、関数の他の呼び出しでは利用できません。var キーワードは、CFScript の var ステートメントに相当します。var キーワードを使用する場合は、次のルールが適用されます。

- var キーワードを使用するすべての cfset タグを、cffunction タグの本文内に置く必要があります。cffunction タグの外にある cfset タグ内で var キーワードを使用すると、ColdFusion のエラーメッセージが表示されます。
- var キーワードを使用するすべての cfset タグを、cffunction タグ本文の最初、つまり、他のあらゆる ColdFusion タグの前に置きます。

次の例では、この新しいキーワードの使用方法を示します。

```
<cffunction name="myFunct">
    <cfset var myVar = "This is a test">
    <cfreturn myVar & " Message.">
</cffunction>
<cfoutput>#myFunct()#</cfoutput>
```

この例では、関数 myFunct が実行されるときにのみ変数 myVar が存在し、ColdFusion ページの他の場所ではこの変数を利用できません。

COM オブジェクト

次の例では、COM オブジェクトが作成されます。cfset タグを使用して、COM オブジェクトのインターフェイス内のメソッドまたはプロパティごとに値を定義します。最後の cfset では、COM オブジェクトの SendMail メソッドからの戻り値を保管する変数を作成します。

```
<cfobject action = "Create"
    name = "Mailer"
    class = "SMTPsvg.Mailer">

<cfset MAILER.FromName = form.fromname>
<cfset MAILER.RemoteHost = RemoteHost>
<cfset MAILER.FromAddress = form.fromemail>
<cfset MAILER.AddRecipient("form.fromname", "form.fromemail")>
<cfset MAILER.Subject = "Testing cfobject">
<cfset MAILER.BodyText = "form.msgbody">
<cfset Mailer.SMTPLog = "logfile">
<cfset success = MAILER.SendMail()>
<cfoutput> #success# </cfoutput>
```

例

```
 <!--- This example shows how to use cfset. --->
<cfquery name = "GetMessages" dataSource = "cfdocexamples">
    SELECT *
    FROM Messages
</cfquery>

<h3>cfset Example</h3>
<p>cfset sets and reassigns values to local or global variables within a page.

<cfset NumRecords = GetMessages.recordCount>
<p>For example, the variable NumRecords has been declared on this
    page to hold the number of records returned from query
    (<cfoutput>#NumRecords#</cfoutput>).

<p>In addition, cfset can be used to pass variables from other pages,
        such as this example, which takes the url parameter Test from this
        link: <a href = "cfset.cfm?test = <cfoutput>
        #URLEncodedFormat("hey, you, get off of my cloud")#</cfoutput>
        ">click here</A>) to display a message:
<p>
<cfif IsDefined ("url.test") is "True">
    <cfoutput><b><I>#url.test#</i></b></cfoutput>
<cfelse>
    <h3>The variable url.test has not been passed from another page.</h3>
</cfif>

<p>cfset can also be used to collect environmental variables, such as the
    time, the IP address of the user, or another function or expression.

<cfset the_date = #DateFormat(Now())# & " " & #TimeFormat(Now())#>
<cfset user_ip = CGI.REMOTE_ADDR>
<cfset complex_expr = (23 MOD 12) * 3>
<cfset str_example = Reverse(Left(GetMessages.body, 35))>

<cfoutput>
    <ul>
        <li>The date: #the_date#
        <li>User IP Address: #user_ip#
        <li>Complex Expression ((23 MOD 12) * 3): #complex_expr#
        <li>String Manipulation (the first 35 characters of
                the body of the first message in our query)
                <br><b>Reversed</b>: #str_example#
                <br><b>Normal</b>: #Reverse(str_example)#
    </ul>
</cfoutput>
```

cfsetting

説明

ページ内での HTML コードの出力など、ページ処理に関する機能を制御します。また、cfsetting タグは、スクリプト形式でも使用できます。

カテゴリ

ページ処理タグ、変数操作タグ

シンタックス

```
<cfsetting
    enableCFoutputOnly = "yes|no"
    requestTimeOut = "value in seconds"
    showDebugOutput = "yes|no" >
```

スクリプト形式の cfsetting のシンタックス：

```
setting enablecfoutputonly="true" requesttimeout="0" showdebugoutput="yes";
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfcache、cfflush、cfheader、cfhtmlhead、cfinclude、cfprocessingdirective、cfsilent、『ColdFusion アプリケーションの開発』の Controlling debugging output with the cfsetting tag

履歴

ColdFusion MX 6.1: 動作が変更されました。タグに本文がある場合は、その内容が実行されます。

ColdFusion MX:

- requestTimeOut 属性が追加されました。
- catchExceptionsByPattern 属性は廃止されました。この属性は ColdFusion 5 より後のリリースでは機能せず、エラーが発生します。
- 例外処理が変更されました。構造化例外マネージャによって、最適な cfcatch ハンドラが検索されます (以前のリリースでは、例外はそのタイプの例外を処理できる最初の cfcatch ブロックによって処理されていました)。

属性

属性	必須 / オプション	デフォルト	説明
enableCFoutputOnly	オプション		• yes: cfoutput タグの外側にある HTML の出力をブロックします。 • no: cfoutput タグの外側にある HTML を表示します。
requestTimeout	オプション		• 秒数 (整数)。ページが無応答のスレッドとして処理されるまでのタイムリミットです。ColdFusion Administrator で設定されたタイムアウトより優先されます。 値に 0 を指定した場合、リクエストのタイムアウトは無効になります。
showDebugOutput	オプション	yes	• yes: Administrator でデバッグが有効になっている場合は、デバッグ情報を表示します。 • no: 生成ページの最後に表示されるデバッグ情報を表示しません。

使用方法

cfsetting の requestTimeout 属性は、URL 内の requestTimeOut の代わりに使用されます。ページのタイムアウトを適用するには、URL 変数を検出し、次のようなコードを使用してページのタイムアウトを変更します。

```
<cfsetting RequestTimeout = "#URL.RequestTimeout#">
```

このタグを使用すると ColdFusion 出力ページの空白を処理することができます。

cfsetting タグをネストする場合、HTML 出力が表示されるようにするには、各 enableCFoutputOnly = "Yes" ステートメントを enableCFoutputOnly = "No" ステートメントに対応させます。たとえば、5 つの enableCFoutputOnly = "Yes" ステートメントの後には、HTML 出力を有効にするため、対応する enableCFoutputOnly = "No" ステートメントが 5 つ必要です。

HTML 出力が有効である場合は (処理済みの enableCFoutputOnly = "No" ステートメントの数に関係なく)、最初の enableCFoutputOnly = "Yes" ステートメントによって出力がブロックされます。

デバッグサービスが有効で、showDebugOutput =" Yes" の場合、IsDebugMode 関数は Yes を返します。そうでない場合は No を返します。

注意：ColdFusion MX より新しいリリースでは </cfsetting> 終了タグを使用できますが、この終了タグは処理には影響しません。cfsetting 属性は、cfsetting タグ本文の内側と外側のコードに影響します。ColdFusion MX では、cfsetting の開始タグと終了タグの間のコードが無視されていました。

例

```
<p>CFSETTING is used to control the output of HTML code in ColdFusion pages.
    This tag can be used to minimize the amount of generated whitespace.

<cfsetting enableCFoutputOnly = "Yes">
    This text is not shown
<cfsetting enableCFoutputOnly = "No">
    <p>This text is shown
<cfsetting enableCFoutputOnly = "Yes">
    <cfoutput>
        <p>Text within cfoutput is always shown
    </cfoutput>
<cfsetting enableCFoutputOnly = "No">
    <cfoutput>
        <p>Text within cfoutput is always shown
    </cfoutput>
```

cfsharepoint

説明

ドキュメントワークスペースの getdwsdata アクションなど、SharePoint が Web サービスアクションとして表示される機能を呼び出します。

カテゴリ

拡張タグ、MS Office の統合のタグ

シンタックス

```
<cfsharepoint
    action="webservice action"
    params="parameter structure"
    domain="domain name"
    name ="result variable name"
    password="connection password"
    userName="user ID"
    wsdl="WSDL file path">
or

<cfsharepoint
    action="webservice action"
    params="parameter structure"
    login = "credentials structure"
    name ="result variable name"
    wsdl="WSDL file path">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

履歴
ColdFusion 9: タグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
action	必須		Web サービスアクションの名前です。指定できるサービスアクションのリストについては、「使用方法」を参照してください。
domain	オプション		SharePoint サーバーへの接続に必要なドメイン名です。login 属性を指定しない場合は必須です。
login	オプション		サービスに渡されるユーザー、パスワード、およびドメインログイン資格情報を含む構造体です。domain、password、および userName 属性を指定しない場合は、domain、password、および userName エントリを持つ login 構造体を指定します。
name	オプション		SharePoint サービスによって返されるデータを配置する結果変数の名前です。
params	オプション		サービスに渡されるパラメータの名前および値を含む構造体です。 パラメータを必要とするサービスの場合、この属性は必須です。
password	オプション		SharePoint サーバーへの接続に必要なパスワードです。login 属性を指定しない場合は必須です。
userName	オプション		SharePoint サーバーへの接続に必要なユーザー名です。login 属性を指定しない場合は必須です。
wsdl	オプション		サービス wsdl ファイルのパスです。サポート対象のアクションのリストにないアクションを呼び出す場合に必要となります。詳細については、「使用方法」を参照してください。

使用方法
cfsharepoint タグは、Microsoft SharePoint Web サービスを呼び出します。数多くの SharePoint Web サービスアクションを呼び出すには、action 属性にアクション名を指定し、params 属性の Web サービスパラメータを渡します。cfsharepoint タグで直接サポートされていないサービスおよびメソッドにアクセスするには、wsdl 属性にサービス WSDL URL を指定します。

注意：cfsharepoint タグは、基本認証のみを使用するサービスとともに使用できます。

サービスおよびアクションを要求するには、次の表にリストされている action 属性値を使用します。通常、これらの属性値は、SharePoint アクション名と同じです。「注意」に、複数のサービスが同じアクション名を持ち、アクション名と属性値が異なる場合が示されています。

注意：Web サービスアクションパラメーターについては、http://msdn.microsoft.com/en-us/library/dd878586(v=office.12).aspx を参照してください。また、http://server_name/_vti_bin/WebServiceName?wsdl の Web サービス WSDL からパラメータを判別することもできます。

cfsharepoint タグが SharePoint サーバーからの結果を受け取り、終了したときに、name 属性で指定されている構造体に応答が含まれます。この構造体には、値 Success または Failure を含む ResultFlag エントリも含まれます。Axis Fault がないか、または応答でエラーが戻された場合、エントリの値は Success になります。その他の場合、この値は Failure になります。

ドキュメントワークスペース

cancreatedwsurl	deletedwsfolder	renamedws

createdws	finddwsdoc	updatedwsdata
createdwsfolder	getdwsdata	
deletedws	removedwsuser	

注意：createdwsfolder および deletedwsfolder アクション属性値は、ドキュメントワークスペースサービスの createfolder および deletefolder アクションに対応しています。

イメージング

delete	getitemsbyids	upload
download	listpicturelibrary	
getimaginglistitems	rename	

注意：getimaginglistitems アクション属性値は、イメージングサービスの getlistitems アクションに対応しています。

リスト

addattachment	getattachmentcollection	updatelist
addlist	getlist	updatelistitems
deleteattachment	getlistcollection	
deletelist	getlistitems	

search または spsearch

注意：spsearch および search は、Windows Sharepoint Services 2.0 にはありません。

Windows SharePoint Services 3.0 では、アクション属性で次のいずれかのアクションが指定されている場合、spsearch.asmx Web サービスを使用して、検索が行われます。Microsoft Office SharePoint Portal Server 2003 または Microsoft Office SharePoint Server 2007 では、search.asmx を使用して検索が行われます。Windows Sharepoint Services 2.0 では、例外がスローされます。

query	registration
queryex	status

UserGroup

addgrouptorole	getgroupcollection	removerole
addrole	getrolecollection	removeusercollectionfromgroup
addusercollectiontogroup	getusercollectionfromrole	removeuserfromgroup
addusercollectiontorole	getusercollectionfromrole	
addusertogroup	getuserinfo	

ビュー

addview	getview	updateview
deleteview	getviewcollection	

データ型の変換

一部の Web サービスアクションには、ColdFusion データ型とは直接対応していない Microsoft データ型のパラメータが必要となります。cfsharepoint タグは、Microsoft データ型と ColdFusion で内部的に使用されている最適な Java データ型を自動的に変換します。次の表に、変換をリストし、対応する ColdFusion のデータ型を示します。

SharePoint のデータ型	ColdFusion の Java データ型
XmlNode	XMLNodeList: たとえば、XML 文字列を XmlParse 関数に渡して作成された ColdFusion XML オブジェクトに対応しています。
ArrayOfString	文字列配列 : 文字列データを含む ColdFusion 配列に対応しています。
UnsignedInt	int: 整数値の ColdFusion 番号に対応しています。
ArrayOfUnsignedInt	int 配列 : 文字列データを含む ColdFusion 配列に対応しています。

例

次の例では、リストとビューの操作方法を示します。ここでは指定されていない、SharePoint サーバーのリソースが必要となります。

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<head>
    <title>cfsharepoint Views Example</title>
</head>

<body>
<cfoutput>

Getting the list collection<br />
<!--- All login information is defined using variables in the Application.cfc file. --->
<cfsharepoint action="getlistcollection" login="#login#" name="result"/>

result.ResultFlag: #result.ResultFlag#<br><br>

Deleting mycustomlist from the collection, if it exists.<br>
<cfloop array=#result.lists# index="list">
    <cfif list.Title EQ "mycustomlist">
        <cfsharepoint action="deletelist" login="#login#"
         name="result1" params="#{listname="mycustomlist"}#"/>
    </cfif>
</cfloop>

    Was anything deleted?
<cfif IsDefined("result1")>
    <b>YES.</b> The result is:</b><br>
    <cfdump var="#result1#"><br>
<cfelse>
    <b>NO</b>
</cfif>

Adding a mycustomlist list<br />
<cfsharepoint action="addlist" login="#login#"  name="result1"
    params="#{listname="mycustomlist",
        description="Adding a list via cfsharepoint",
        templateid=100}#"/>

addlist result.ResultFlag: #result1.ResultFlag#<br><br>


<cfset viewFields = xmlparse("<ViewFields>
```

```
                                <FieldRef Name='Title'/>
                                <FieldRef Name='ID'/>
                                </ViewFields>")>

<cfset query = xmlparse("<Query>
                            <Where>
                            <Lt>
                            <FieldRef Name='ID'/>
                            <Value Type='Counter'>10</Value>
                            </Lt>
                           </Where>
                           <OrderBy>
                           <FieldRef Name='ID'/>
                            </OrderBy>
                            </Query>")>

<cfset rowlimit = xmlparse("<RowLimit Paged='True'>50</RowLimit>")>

Adding a myview1 view for the mycustomlist list<br />
<cfsharepoint action="addview" login="#login#" name="result"
    params="#{listName="mycustomlist",viewname="myview1",
    viewFields="#viewFields#",query="#query#",rowlimit="#rowlimit#",
    type="grid",makeViewDefault=false}#"/>

addview result.ResultFlag: #result.ResultFlag#<br><br>


Adding a myview3 view for the mycustomlist list<br />
<cfsharepoint action="addview" login="#login#" name="result"
params="#{listName="mycustomlist",viewname="myview3",viewFields="#viewFields#",
    query="#query#",rowlimit="#rowlimit#",type="grid",makeViewDefault=false}#"/>

addview result.ResultFlag: #result.ResultFlag#<br><br>

Getting the updated mycustomlist view collection<br>
<cfsharepoint action="getviewcollection" login="#login#" name="result"
    params="#{listName="mycustomlist"}#"/>

<b>getviewcollection result</b><br>
<cfdump var="#result#"><br />

The names of the collection's views:<br>
<cfloop array=#result.views# index=v>
<cfoutput>#v.displayname#<br></cfoutput>
</cfloop>
<br>

Deleting the list<br>
<cfsharepoint action="deletelist" login="#login#" name="result1" params="#{listname="mycustomlist"}#"/>

deletelist result.ResultFlag: #result1.ResultFlag#
</cfoutput>

</body>
</html>
```

cfslider

説明

一定の範囲から数値を選択するためのスライダコントロールを ColdFusion フォーム内に配置します。このスライダは、スライダの溝に沿って移動します。ユーザーがスライダを動かすと、現在の値が表示されます。HTML およびアプレット形式のフォームの 250 ページの「cfform」タグ内で使用されます。Flash フォームではサポートされません。

HTML フォームでは、調整可能値を固定インクリメントで変更できる、視覚的に豊かなスライドを作成できます。最大値、最小値、およびインクリメント値を指定すると、複雑な結果をすばやくフィルタ処理できます。

スライダは、スライダコントロールに基づいて分類されます。使用可能なスライダコントロールは、次のとおりです。

Vertical 上下に調整できる縦方向コントロールを持つスライダとなります。

Horizontal 左右に調整できる横方向のコントロールを持つスライダとなります。

Tip スライダにデータヒントとして値が表示されます。

Snapping スライダは、インクリメント値で移動します。

カテゴリ

フォームタグ

シンタックス

HTML の場合

```
<cfslider
    name = "name"
    clickToChange = "true|false"
    format = "html"
    height = "integer"
    increment = "Unit increment value"
    max = "maximum value for the slider"
    min = "minimum value for the slider"
    onChange = "JavaScript function name"
    onDrag = "JavaScript function name"
    tip = "true|false"
    value = "integer"
    vertical = "true|false"
    width = "integer">
```

シンタックス

アプレットの場合

```
<cfslider
    name = "name"
    align = "top|left|bottom|baseline|texttop|absbottom|
    middle|absmiddle|right"
    bgColor = "color"
    bold = "yes|no"
    font = "font name"
    fontSize = "integer"
    height = "integer"
    hSpace = "integer"
    italic = "yes|no"
    label = "text"
    lookAndFeel = "motif|windows|metal"
    message = "text"
    notSupported = "text"
    onError = "text"
    onValidate = "script name"
    range = "minimum value, maximum value"
    scale = "integer"
    textColor = "color"
    value = "integer"
    vertical = "yes|no"
    vSpace = "integer"
    width = "integer">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfapplet、cfcalendar、cfform、cfformgroup、cfformitem、cfgrid、cfinput、cfselect、cftextarea、cftree、『ColdFusion アプリケーションの開発』の Introduction to Retrieving and Formatting Data および Building Dynamic Forms with cfform Tags

履歴

ColdFusion MX: img、imgStyle、grooveColor、refreshLabel、tickmarklabels、tickmarkmajor、tickmarkminor、tickmarkimages の各属性が非推奨になりました。これらのクラスおよびメンバーは動作せず、今後のリリースではエラーが発生する可能性があります。

属性

属性	必須 / オプション	デフォルト	説明
name	必須		cfslider コントロールの名前です。
align	オプション		スライダの配置です。 • top • left • bottom • baseline • texttop • absbottom • middle • absmiddle • right
bgColor	オプション		スライダラベルの背景色です。 16 進数の値を入力するには、bgColor="##xxxxxx" という形式を使用します。ここで、x は 0 から 9 または A から F です。シャープ記号 (#) は 2 つ使用するか、または使用しないでください。 • 16 進数形式の任意の色 • black • red • blue • magenta • cyan • orange • darkgray • pink • gray • white • lightgray • yellow
bold	オプション	no	• yes: ラベルテキストをボールドで表示します。 • no: 標準テキストです。

属性	必須 / オプション	デフォルト	説明
clickToChange	オプション HTML		スライダをクリックすると、ポインタの値が変更されるかどうかを指定します。 • true • false
font	オプション		ラベルテキストのフォント名です。
fontSize	オプション		ラベルテキストのフォントサイズです (単位 : ポイント)。
format	オプション	applet	次のいずれかの形式を指定します。 • html • applet
height	オプション	アプレットの場合は 40 HTML の場合は 100	スライダコントロールの高さです (単位 : ピクセル)。
hSpace	オプション		スライダの左右に取る水平方向の余白です (単位 : ピクセル)。
italic	オプション	no	• yes: ラベルテキストをイタリックで表示します。 • no: 標準テキストです。
increment	オプション HTML		スナップスライダのユニットインクリメント値です。
label	オプション		コントロールとともに表示されるラベルです。たとえば、"Volume" というラベルを指定すると、"Volume %value%" と表示されます。 値を参照するには、"%value%" を使用します。%% を省略すると、ラベルの直後にスライダ値が表示されます。
lookAndFeel	オプション	Windows	• motif: Motif スタイルを使用してスライダを表示します。 • windows: Windows スタイルを使用してスライダを表示します。 • metal: Java Swing スタイルを使用してスライダを表示します。 プラットフォームでこの選択肢がサポートされていない場合は、プラットフォームのデフォルトのスタイルが適用されます。
max	オプション HTML		スライダの最大値です。
min	オプション HTML		スライダの最小値です。
message	オプション アプレット		検証に失敗した場合に表示されるメッセージテキストです。

属性	必須 / オプション	デフォルト	説明
notSupported	オプション		Java アプレットベースの cfform コントロールが含まれているページを、Java がサポートされていない (または Java のサポートが無効になっている) ブラウザで開いた場合に表示するテキストです。次に例を示します。 "ColdFusion Java アプレットを表示するには、ブラウザで Java がサポートされていなければなりません。" デフォルトのメッセージ : ColdFusion Java アプレットを表示するには、 ブラウザが Java をサポートしている必要があります。
onChange	オプション HTML		スライダ値が変更された場合に実行するカスタム JavaScript 関数です。 関数名のみを指定します。
onDrag	オプション HTML		スライダをドラッグした場合に実行するカスタム JavaScript 関数です。 関数名のみを指定します。
onError	オプション		検証に失敗した場合に実行するカスタム JavaScript 関数です。 関数名のみを指定します。
onValidate	オプション		ユーザー入力 (この場合は、デフォルトのスライダ値への変更) を検証するカスタムの JavaScript 関数です。 関数名のみを指定します。
range	オプション	"0,100"	スライダの範囲の数値です。 値をカンマで区切ってください。
scale	オプション		符号なし整数です。range 内でスライダのスケールを定義します。たとえば、range="0,1000" および scale="100" の場合、表示の値は次のようになります。 0, 100, 200, 300, ... ColdFusion の符号付きおよび符号なしの整数は、-2,147,483,648 ～ 2,147,483,647 の範囲です。
textColor	オプション		オプション : bgcolor 属性の場合と同じです。
tip	オプション HTML	true	データ値をデータヒントとして表示する必要があるかどうかを指定します。 • true • false
value	オプション	範囲内の最小値	スライダの開始値です。range 属性の値の範囲内の値にする必要があります。
vertical	オプション	no (アプレットフォームの場合)、false (HTML フォームの場合)	アプレットフォームの場合 : • yes: スライダをブラウザに垂直に表示します。width 属性および height 属性を設定する必要があります。ColdFusion では、width および height の値は自動的には交換されません。 • no: スライダを水平に表示します。 HTML フォームの場合 : • true: スライダをブラウザに垂直に表示します。width 属性および height 属性を設定する必要があります。ColdFusion では、width および height の値は自動的には交換されません。 • false: スライダを水平に表示します。
vSpace	オプション		スライダの上下に取る垂直方向の余白です (単位 : ピクセル)。
width	オプション	HTML の場合は 200	スライダコントロールの幅です (単位 : ピクセル)。

使用方法

このタグを使用する場合は、クライアントで Java アプレットをダウンロードする必要があります。このタグを使用すると、同じ情報を表示するのに HTML の form 要素を使用する場合よりも処理速度が少し遅くなる可能性があります。また、クライアントに最新の Java プラグインがインストールされていない場合、タグを表示するために最新の Java プラグインをダウンロードしなければならないこともあります。

このタグを正しく動作させるには、JavaScript 対応のブラウザを使用する必要があります。

次の条件を満たす場合、このタグのオプションとして挿入されるユーザーが選択したクエリーデータからの値は、ユーザーがフォームを送信した後も継続して表示されます。

- cfform の preserveData 属性が "Yes" に設定されている場合。
- cfform の action 属性がフォーム自身と同じページ (デフォルト) に送信される場合、または、ユーザー入力フォーム上のコントロールと同じ名前のコントロールを持つフォームがアクションページ上にある場合

詳細については、250 ページの「cfform」タグのエントリを参照してください。

例

```
<!--- This example shows how to use cfslider</h3>
    <br/ >
    <cfform name="form01">
    <cfslider name="slider1"
    format="HTML"
    vertical="false"
    width="350"
    value="100"
    min="0"
    max="200"
    increment="10"
    tip="true"/>
</cfform>
```

cfsilent

説明

タグのスコープ内で CFML によって生成される出力を抑制します。

カテゴリ

データ出力タグ、ページ処理タグ

シンタックス

```
<cfsilent>
    ...
</cfsilent>
```

関連項目

cfcache、cfflush、cfheader、cfhtmlhead、cfinclude、cfsetting、『ColdFusion アプリケーションの開発』の Writing and Calling User-Defined Functions

使用方法

このタグには終了タグが必要です。

例

```
 <h3>cfsilent</h3>

<cfsilent>
<cfset a = 100>
<cfset b = 99>
<cfset c = b-a>
<cfoutput>Inside cfsilent block<br>
b-a = #c#</cfoutput><br>
</cfsilent>

<p>Even information within cfoutput tags does not display within a
cfsilent block.<br>
<cfoutput>
b-a = #c#
</cfoutput>
</p>
```

cfspreadsheet

説明

Excel スプレッドシートファイルを管理します。

- シートファイルからシートを読み取り、ColdFusion スプレッドシートオブジェクト、クエリー、CSV 文字列、または HTML 文字列に保管します。
- クエリー、ColdFusion スプレッドシートオブジェクト、または CSV 文字列変数から単一のシートを新規 XLS ファイルに書き込みます。
- 既存の XLS ファイルにシートを追加します。

カテゴリ

拡張タグ

シンタックス

タグのシンタックスは action 属性の値によって異なります。

Read

```
<cfspreadsheet
    action="read"
    src = "filepath"
    columns = "range"
    columnnames = "comma-delimited list"
    excludeHeaderRow = "true | false"
    format = "CSV|HTML"
    headerrow = "row number"
    name = "text"
    query = "query name"
    rows = "range"
    sheet = "number"
    sheetname = "text">
```

Update

```
<cfspreadsheet
    action="update"
    filename = "filepath"
    format = "csv"
    name = "text"
    password = "password"
    query = "query name"
    sheetname = "text" >
```

Write

```
<cfspreadsheet
    action="write"
    filename = "filepath"
    format = "csv"
    name = "text"
    overwrite = "true | false"
    password = "password"
    query = "queryname"
    sheetname = "text" >
```

関連項目

スプレッドシート関数

履歴

ColdFusion 9.0.1：excludeHeaderRow 属性が追加されました。

ColdFusion 9: このタグが追加されました。

属性

属性	アクション	必須 / オプション	デフォルト	説明
action	すべて	必須		次のいずれかです。 • read: XLS 形式のファイルの内容を読み取ります。 • update: 既存の XLS ファイルに新しいシートを追加します。uppdate アクションを使用して、ファイル内の既存のシートを変更することはできません。詳細については、「使用方法」を参照してください。 • write: 新しい XLS 形式のファイルを書き込むか、または既存のファイルを上書きします。
filename	update、writer	必須		書き込まれるファイルのパス名です。
excludeHeaderRow	read	オプション	false	true に設定すると、クエリ結果にヘッダー行が含まれなくなります。この属性は、Excel をクエリとして読み込む場合に役立ちます。headerrow 属性を指定すると、列名はヘッダー行から取得されます。ですが、列名は、クエリーの最初の行にも含まれてしまいます。ヘッダ行が含まれないようにするには、前述の属性値を true に設定します。
name	すべて	name または query が必須です。		• read アクション : スプレッドシートファイルデータを保管する変数です。name または query を指定します。 • write および update アクション : 書き込むデータが含まれている CSV フォーマットデータまたは ColdFusion スプレッドシートオブジェクトを含む変数です。name または query を指定します。
query	すべて	name または query が必須です。		• read アクション : 変換されたスプレッドシートファイルを保管するクエリーです。format、name、または query を指定します。 • write および update アクション : 書き込むデータが含まれているクエリー変数です。name または query を指定します。
src	read	必須		読み取るファイルのパス名です。
columns	read	オプション		列番号または列の範囲です。単一の番号、ハイフン区切りの列範囲、カンマ区切りリスト、またはこれらの任意の組み合わせ (1,3-6,9 など) を指定します。
columnnames	read	オプション		カンマ区切りの列名です。
format	すべて	オプション	read: スプレッドシートオブジェクトとして保存します。 update および write: スプレッドシートオブジェクトを保存します。	name 変数で表されるデータの形式です。 • すべて : csv: 読み取り時に、XLS ファイルを CSV 変数に変換します。 • update または write の際、CSV 変数を XLS ファイルに保存します。 • 読み取り専用 : html: XLS ファイルを HTML 変数に変換します。 cfspreadsheet タグは、常に、スプレッドシートデータを XLS ファイルとして書き込みます。HTML 変数または CSV 変数を HTML ファイルまたは CSV ファイルとして書き込むには、cffile タグを使用します。
headerrow	read	オプション		列名を含む行番号です。
overwrite	write	オプション	false	既存のファイルを上書きするかどうかを指定するブール値です。
password	update write	オプション		シートを変更するためのパスワードを設定します。 **メモ** : 空白文字列のパスワードを設定しても、パスワードの保護が完全に設定解除されるわけではありません。シートを変更しようとすると、パスワードの入力を求められます。

属性	アクション	必須 / オプション	デフォルト	説明
rows	read	オプション		読み取る行の範囲です。単一の番号、ハイフン区切りの行範囲、カンマ区切りリスト、またはこれらの任意の組み合わせ (1,3-6,9 など) を指定します。
sheet	read	オプション		シートの番号です。read アクションの場合、sheet または sheetname を指定できます。
sheetname	すべて	オプション		シートの名前 : read アクションの場合、sheet または sheetname を指定できます。 write および update アクションの場合は、sheetname で指定した値に従って、指定されたシートの名前が変更されます。

使用方法

各 ColdFusion スプレッドシートオブジェクトは、Excel シートを表しています。

- 複数のシートを持つ Excel ファイルを読み取るには、read オプションで複数の cfspreadsheet タグを使用し、シートごとに異なる name および sheet または sheetname 属性を指定します。
- 単一ファイルに複数のシートを書き込むには、write アクションを使用してファイルを作成し、最初のシートを保存します。update アクションを使用して、各追加シートを追加します。
- 既存のファイルを更新するには、ファイルのすべてのシートを読み取り、複数のシートを変更して、その内容を使用します。write アクションおよび Update アクション (複数のシートファイルの場合) を使用して、ファイル全体を再度書き込みます。

cfspreadsheet タグは、XLS 形式のファイルのみを書き込みます。CSV ファイルを書き込むには、データを CSV 形式の文字列変数に配置し、cffile タグを使用して、ファイルにこの変数の内容を書き込みます。

1267 ページの「SpreadsheetNew」や 1236 ページの「SpreadsheetAddColumn」などの ColdFusion Spreadsheet* 関数を使用して、新しい ColdFusion スプレッドシートオブジェクトを作成し、スプレッドシートの内容を変更します。

例

次の例では、cfspreadsheet タグを使用して、さまざまな形式を使用した Excel スプレッドシートの読み取りおよび書き込みを行います。また、ColdFusion Spreadsheet 関数を使用してシートを変更する簡単な方法も示します。

```
 <!--- Read data from two datasource tables. --->
<cfquery
       name="courses" datasource="cfdocexamples"
       cachedwithin="#CreateTimeSpan(0, 6, 0, 0)#">
       SELECT CORNUMBER,DEPT_ID,COURSE_ID,CORNAME
       FROM COURSELIST
</cfquery>

<cfquery
       name="centers" datasource="cfdocexamples"
       cachedwithin="#CreateTimeSpan(0, 6, 0, 0)#">
       SELECT *
       FROM CENTERS
</cfquery>

<cfscript>
    //Use an absolute path for the files. --->
       theDir=GetDirectoryFromPath(GetCurrentTemplatePath());
    theFile=theDir & "courses.xls";
    //Create two empty ColdFusion spreadsheet objects. --->
    theSheet = SpreadsheetNew("CourseData");
    theSecondSheet = SpreadsheetNew("CentersData");
    //Populate each object with a query. --->
    SpreadsheetAddRows(theSheet,courses);
    SpreadsheetAddRows(theSecondSheet,centers);
```

```
</cfscript>

<!--- Write the two sheets to a single file --->
<cfspreadsheet action="write" filename="#theFile#" name="theSheet"
    sheetname="courses" overwrite=true>
<cfspreadsheet action="update" filename="#theFile#" name="theSecondSheet"
    sheetname="centers">

<!--- Read all or part of the file into a spreadsheet object, CSV string,
      HTML string, and query. --->
<cfspreadsheet action="read" src="#theFile#" sheetname="courses" name="spreadsheetData">
<cfspreadsheet action="read" src="#theFile#" sheet=1 rows="3,4" format="csv" name="csvData">
<cfspreadsheet action="read" src="#theFile#" format="html" rows="5-10" name="htmlData">
<cfspreadsheet action="read" src="#theFile#" sheetname="centers" query="queryData">

<h3>First sheet row 3 read as a CSV variable</h3>
<cfdump var="#csvData#">

<h3>Second sheet rows 5-10 read as an HTML variable</h3>
<cfdump var="#htmlData#">

<h3>Second sheet read as a query variable</h3>
<cfdump var="#queryData#">

<!--- Modify the courses sheet. --->
<cfscript>
    SpreadsheetAddRow(spreadsheetData,"03,ENGL,230,Poetry 1",8,1);
    SpreadsheetAddColumn(spreadsheetData,
    "Basic,Intermediate,Advanced,Basic,Intermediate,Advanced,Basic,Intermediate,Advanced",
    3,2,true);
</cfscript>

<!--- Write the updated Courses sheet to a new XLS file --->
<cfspreadsheet action="write" filename="#theDir#updatedFile.xls" name="spreadsheetData"
    sheetname="courses" overwrite=true>
<!--- Write an XLS file containing the data in the CSV variable. --->
<cfspreadsheet action="write" filename="#theDir#dataFromCSV.xls" name="CSVData"
    format="csv" sheetname="courses" overwrite=true>
```

cfsprydataset

説明

バインド式の結果から Spry XML または JSON データセットを作成します。

カテゴリ

インターネットプロトコルタグ

シンタックス

```
<cfsprydataset
    bind = "bind expression"
    name = "data set name"
    onBindError = "JavaScript function name"
    options = "Spry options object"
    type = "xml|json"
    xpath = "XPath expression">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfajaximport、『ColdFusion アプリケーションの開発』の Using Spry with ColdFusion

履歴

ColdFusion 8: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
bind	必須		Spry データセットに渡す XML 形式または JSON 形式の文字列を返すバインド式です。このバインド式では CFC 関数または URL を指定し、ColdFusion コントロールの値を表すバインドパラメータを含めます。 バインド式の詳細については、『ColdFusion アプリケーションの開発』の Binding data to form fields を参照してください。
name	必須		Spry データセットの名前です。
onBindError	オプション、 HTML	「説明」を参照	バインド式でエラーが発生した場合に実行する JavaScript 関数の名前です。この関数は 2 つの属性 (HTTP ステータスコードとメッセージ) を取る必要があります。 この属性を省略し、(ColdFusion.setGlobalErrorHandler 関数を使用して) グローバルエラーハンドラを指定すると、エラーメッセージが表示されます。それ以外の場合は、デフォルトのエラーポップアップが表示されます。
options	オプション		データセットのコンストラクタオプションを含んでいる JavaScript オブジェクトです。たとえば、HTTP POST メソッドを使用してデータをリクエストするには、次の属性を指定します。 options="{method: 'POST'}". Spry オプションの詳細については、www.adobe.com/go/learn_spry_framework_jp の Spry のドキュメントを参照してください。
type	オプション	xml	バインド式によって返されるデータの形式に対応するデータセットタイプを指定します。使用できる値は次のとおりです。 • json • xml
xpath	xml タイプの場合は必須 JSON の場合は無効		バインド式によって返される XML からデータを抽出する XPath 式です。データセットには、この XPath 式に一致するデータのみが格納されます。

使用方法

このタグでは、バインド式を使用して、ColdFusion コントロールや別の Spry データセットの値に基づき、Spry XML データセットまたは JSON データセットの内容をダイナミックに作成します。バインド式を使用せずに Spry データセットを作成するには、JavaScript 関数 Spry.Data.JSONDataSet() および Spry.Data.XMLDataSet() を使用します。

このタグでは、Spry HTML データセットは作成できません。

フィルタを使用して JSON 式から JSON データセットの内容を選択するには、options 属性で path または subpath オプションを指定します。たとえば、items.item 要素のみを使用して JSON データから Spry JSON データセットを作成するには、次のようなタグを使用します。

```
<cfsprydataset name="theItems" type="json"
    bind="CFC:dataMgr.getdetails(prodname={myform:mygrid.TITLE})"
    options="{path: 'items.item.'}">
```

例

次の cfsprydataset タグは、bookgrid コントロールで選択されている行が変わるたびに、CFC 関数 GridDataManager.getProductDetails を呼び出して dsProduct Spry XML データセットを更新します。このタグは、選択された行の TITLE フィールドを prodname パラメータとして CFC 関数に渡します。このタグの詳しい使用例については、『ColdFusion アプリケーションの開発』の Using Spry with ColdFusion を参照してください。

```
<cfsprydataset
    name="dsProduct"
    type="xml"
    bind="CFC:GridDataManager.getProductDetails(prodname=
        {bookform:bookgrid.TITLE})"
    xpath="products/product"
    options="{method: 'POST'}"
    onBindError="errorHandler">
```

cfstoredproc

説明

サーバーデータベースのストアドプロシージャを実行します。このタグを使用すると、データベース接続情報やストアドプロシージャを指定できます。

カテゴリ

データベース操作タグ

シンタックス

```
<cfstoredproc
    dataSource = "data source name"
    procedure = "procedure name"
    cachedAfter = "date"
    cachedWithin = "time span"
    debug = "yes|no"
    timeOut = "timeout interval"
    fetchClientInfo = "yes|no"
    blockFactor = "block size"
    password = "password"
    result = "result name"
    returnCode = "yes|no"
    username = "user name">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfinsert、cfqueryparam、cfprocparam、cfprocresult、cftransaction、cfquery、cfupdate、『ColdFusion アプリケーションの開発』の Optimizing database use

履歴

ColdFusion 10：timeOut、fetchClientInfo および clientInfo の各属性が追加されました。

ColdFusion MX 7: result 属性が追加されました。

ColdFusion MX: connectString、dbName、dbServer、dbtype、provider、および providerDSN 属性は非推奨になりました。ColdFusion 5 以降のリリースでは、これらは機能せず、エラーを引き起こす可能性があります (ColdFusion MX 以降のリリースでは、Type 4 JDBC ドライバが使用されます)。

属性

属性	必須 / オプション	デフォルト	説明
dataSource	必須		ストアドプロシージャが含まれているデータベースをポイントするデータソースの名前です。
procedure	必須		データベースサーバー上のストアドプロシージャの名前です。
blockFactor	オプション	1	サーバーから一度に取得する行の最大数です。値の範囲は 1 ～ 100 です。
cachedAfter	オプション		日付の値 (たとえば、April 16, 2008、4-16-2008)。元のクエリーの日付がこの日付よりも後の場合は、ColdFusion ではキャッシュされているクエリーデータが使用されます。キャッシュされているデータを使用するには、現在のクエリーで、同じ SQL 文、データソース、クエリー名、ユーザー名、およびパスワードを使用する必要があります。 日付時刻オブジェクトの範囲は、西暦 100 ～ 9999 年です。 日付の値を文字列として指定するときは、文字列を引用符で囲みます。
cachedWithin	オプション		CreateTimeSpan 関数を使用して作成された期間。元のクエリーの日付がこの期間内の場合、キャッシュされているクエリーデータが使用されます。CreateTimeSpan では、現在からさかのぼって期間を定義します。クエリーのキャッシュ機能が Administrator 内で有効になっている場合のみ機能します。 キャッシュされているデータを使用するには、現在のクエリーで、同じ SQL 文、データソース、クエリー名、ユーザー名、およびパスワードを使用する必要があります。
clientInfo	オプション		データベース接続で設定するクライアントのプロパティが含まれる構造体です。
debug	オプション	no	• yes: 各ステートメントのデバッグ情報をリストします。 • no
fetchClientInfo	オプション	no	yes に設定した場合、最後のクエリによって渡された、キーと値のペアを持つ構造体が返されます。
password	オプション		データソースセットアップのパスワードよりも優先されます。
result	オプション		cfstoredproc から statusCode 変数および ExecutionTime 変数を返す構造体の名前を指定します。これを指定すると、これらの変数にアクセスするときに使用する接頭辞として cfstoredproc の代わりにこの値が使用されます。詳細については、「使用方法」を参照してください。
returnCode	オプション	no	• yes: ストアドプロシージャによって返されるステータスコードを cfstoredproc.statusCode に挿入します。 • no
timeOut	オプション		各アクションの実行を許可する秒数です。この時間を過ぎると、エラーが返されます。 累積時間はこの値を超える可能性があります。JDBC ステートメントの場合は、この属性が設定されます。その他のドライバについては、各ドライバのマニュアルを参照してください。
username	オプション		データソースセットアップのユーザー名よりも優先されます。

使用方法

このタグは、データベースのストアドプロシージャを呼び出すために使用します。このタグ内で、次のように 554 ページの「cfprocresult」タグおよび 548 ページの「cfprocparam」タグをコーディングします。

- 554 ページの「cfprocresult」: ストアドプロシージャから結果セットが返される場合は、結果セットごとに 1 つの cfprocresult タグをコーディングします。
- 548 ページの「cfprocparam」: ストアドプロシージャで入力または出力パラメータが使用されている場合は、パラメータごとに 1 つの cfprocparam タグをコーディングし、ストアドプロシージャ定義内の各パラメータを含めます。

returnCode = "Yes" に設定すると、このタグによりストアドプロシージャのステータスコードを保持する変数 **prefix.**statusCode が設定されます。ステータスコードの値は DBMS によって異なります。コード値の意味については、DBMS のドキュメントを参照してください。

このタグでは、変数 **prefix.**ExecutionTime が設定されます。この変数には、ストアドプロシージャの実行時間 (単位 : ミリ秒) が格納されます。

prefix の値は、cfstoredproc、または (設定されている場合は) result 属性で指定された値のいずれかになります。result 属性を使用すると、複数のページから同時にストアドプロシージャを呼び出す場合に、一方の結果によって他方の結果が上書きされることを防止できます。たとえば、result 属性を myResult に設定した場合は、myResult.ExecutionTime として ExecutionTime にアクセスします。この属性を設定していない場合は、cfstoredproc.ExecutionTime としてアクセスします。

このタグは、ストアドプロシージャとその使用方法について十分に理解してから実装してください。

次の例では、Sybase ストアドプロシージャを使用しています。Oracle 8 または 9 のストアドプロシージャの例については、548 ページの「cfprocparam」を参照してください。

例

```
 <cfset ds = "sqltst">

<!---
If submitting a new book,
insert the record and display
confirmation --->
<cfif isDefined("form.title")>

<cfstoredproc procedure="Insert_Book" datasource="#ds#">

<cfprocparam
    cfsqltype="cf_sql_varchar"
    value="#form.title#">

<cfprocparam
    cfsqltype="cf_sql_numeric"
    value="#form.price#">

<cfprocparam
    cfsqltype="cf_sql_date"
    value="#form.publishDate#">

<cfprocparam
    cfsqltype="cf_sql_numeric"
    type="out"
    variable="bookId">

</cfstoredproc>

<cfoutput>
<h3>'#form.title#' inserted into database.The ID is #bookId#.</h3>
</cfoutput>

</cfif>

<cfform action="#CGI.SCRIPT_NAME#" method="post">
<h3>Insert a new book</h3>

Title:
<cfinput type="text" size="20" required="yes" name="title"/>
<br/>

Price:
```

```
<cfinput type="text" size="20" required="yes" name="price" validate="float" />
<br/>

Publish Date:
<cfinput type="text" size="5" required="yes" name="publishDate" validate="date" />
<br/>

<input type="submit" value="Insert Book"/>

</cfform>

<!---
    This view-only example executes a Sybase stored procedure that
    returns three result sets, two of which we want. The stored
    procedure returns the status code and one output parameter,
    which we display. We use named notation for the parameters.
--->
<!---
<cfstoredproc procedure = "foo_proc"
    dataSource = "MY_SYBASE_TEST" username = "sa"
    password = "" dbServer = "scup" dbName = "pubs2"
    returnCode = "Yes" debug = "Yes">
    <cfprocresult name = RS1>
    <cfprocresult name = RS3 resultSet = 3>

    <cfprocparam type = "IN" CFSQLType = CF_SQL_INTEGER
            value = "1" dbVarName = @param1>
    <cfprocparam type = "OUT" CFSQLType = CF_SQL_DATE
        variable = FOO dbVarName = @param2>
</cfstoredproc>
--->
<!---
<cfoutput>The output param value: '#foo#'<br></cfoutput>
<h3>The Results Information</h3>
<cfoutput query = RS1>#name#,#DATE_COL#<br></cfoutput><p>
<cfoutput>
    <hr>
    <p>Record Count: #RS1.recordCount# >p>Columns: #RS1.columnList#<hr>
</cfoutput>
<cfoutput query = RS3>#col1#,#col2#,#col3#<br>
</cfoutput><p>
<cfoutput>
    <hr>
    <p>Record Count: #RS3.recordCount# <p>Columns: #RS3.columnList#<hr>
    The return code for the stored procedure is: '#cfstoredproc.statusCode#'<br>
</cfoutput>
--->
```

cfswitch

説明

渡された式を評価し、式の結果にマッチする cfcase タグに制御権を渡します。オプションで、cfdefaultcase タグをコーディングすることもできます。一致する cfcase タグの値がなければ、このタグに制御権が渡されます。

カテゴリ

フロー制御タグ

シンタックス

```
<cfswitch
    expression = "expression">
        one or more cfcase tags
        zero or one cfdefaultcase tags
</cfswitch>
```

関連項目

cfcase、cfdefaultcase、cfabort、cfloop、cfbreak、cfexecute、cfexit、cfif、cflocation、cfrethrow、cfthrow、cftry、『ColdFusion アプリケーションの開発』の cfswitch, cfcase, and cfdefaultcase

履歴

ColdFusion 8: <cfcase> の値を解析する方法が変更されました。以前は、数値の日付が設定された <cfcase> タグからは期待どおりの結果が返されませんでした。たとえば、<cfcase value="00"> と <cfcase value="0A"> は両方とも 0 と評価されていました。"0A" という値は、日付として処理されて、12/30/1899 から 0 日に変換されていました。"00" という値も 0 という値に評価されていました。このため、「CFCASE タグ用のコンテキスト検証エラーです。CFSWITCH には、値 "0.0" 用の重複した CFCASE があります。」という例外が発生していました。現在、<cfswitch> タグは期待どおりの結果を返します。

ColdFusion MX: cfdefaultcase タグの配置の必要条件が変更されました。cfdefaultcase タグを cfswitch ステートメント内の任意の場所に配置できるようになりました。最後の項目にする必要はありません。

属性

属性	必須 / オプション	デフォルト	説明
expression	必須		スカラー値を生成する ColdFusion 式です。ColdFusion により、整数、実数、ブール値、および日付が数値に変換されます。たとえば、true、1、1.0 はすべて同じです。

使用方法

このタグには終了タグが必要です。このタグ内のすべてのコードは cfcase または cfdefaultcase タグで囲む必要があります。そうでない場合、エラーが発生します。

このタグの後には、cfcase タグを少なくとも 1 つ指定します。オプションで、cfdefaultcase タグを指定することもできます。このタグを使用すると、cfcase タグおよび cfdefaultcase タグから一致する代替タグを選択してそのタグにジャンプし、cfcase の開始タグと終了タグの間にあるコードを実行できます。

cfswitch タグを使用すれば、一連の cfif タグや cfelseif タグを使用するよりもパフォーマンスが良くなり、コードも読みやすくなります。

例

```
<!---
    This example shows the use of cfswitch and cfcase to
    exercise a case statement in CFML.
--->
<cfquery name = "GetEmployees" dataSource = "cfdocexamples">
    SELECT Emp_ID, FirstName, LastName, EMail, Phone, Department
    FROM Employees
</cfquery>

<h3>cfswitch Example</h3>
<!--- By outputting the query and using cfswitch, we classify the
    output without using a cfloop construct. --->
<p>Each time the case is fulfilled, the specific information is printed;
if the case is not fulfilled, the default case is output </p>
<cfoutput query="GetEmployees">
<cfswitch expression="#Trim(Department)#">
    <cfcase value="Sales">
        #FirstName# #LastName# is in <b>sales</b><br><br>
    </cfcase>
    <cfcase value="Accounting">
        #FirstName# #LastName# is in <b>accounting</b><br><br>
    </cfcase> <cfcase value="Administration">
        #FirstName# #LastName# is in <b>administration</b><br><br>
    </cfcase>
    <cfdefaultcase>
        #FirstName# #LastName# is not in Sales, Accounting, or
            Administration.<br><br>
    </cfdefaultcase>
</cfswitch>
</cfoutput>
```

タグ t

cftable

説明

ColdFusion ページ内にテーブルを構築します。このタグは、書式付きテキストとして、または HTMLTable 属性を使用して、HTML テーブルにデータを表示します。HTML の table タグコードを記述したくない場合や、書式付きテキストとしてデータを表示できる場合に、このタグを使用します。

HTML で <pre> タグおよび </pre> タグを使用して定義された、書式付きテキストは、固定幅のフォントで表示されます。空白および改行は、pre タグ内の記述に従って表示されます。詳細については、HTML のリファレンスガイドを参照してください。

テーブルの列および行の仕様を定義するには、このタグ内で cfcol タグを使用します。

カテゴリ

データ出力タグ

シンタックス

```
<cftable
    query = "query name"
    border
    colHeaders
    colSpacing = "number of spaces"
    headerLines = "number of lines"
    HTMLTable
    maxRows = "maxrows table"
    startRow = "row number">
    ...
</cftable>
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfcol、cfcontent、cflog、cfoutput、cfprocessingdirective、『ColdFusion アプリケーションの開発』の Retrieving data

属性

属性	必須 / オプション	デフォルト	説明
query	必須		データを取り出す cfquery の名前です。
border	オプション		テーブルの周囲に枠を表示します。 値にかかわらず、この属性を使用するとテーブルの周囲に枠が表示されます。 HTMLTable 属性を使用する場合のみ使用します。
colHeaders	オプション		列のヘッダを表示します。この属性を使用する場合、cfcol タグの header 属性を同時に使用してヘッダを定義します。 値にかかわらず、この属性を使用すると列のヘッダが表示されます。
colSpacing	オプション	2	列間のスペースの数です。
headerLines	オプション	2	テーブルのヘッダに使用する行の数です (デフォルトでは、テーブルのヘッダと最初の行の間を 1 行空けます)。
HTMLTable	オプション		HTML 3.0 テーブルでデータを表示します。 値にかかわらず、この属性を使用すると、データが HTML テーブルで表示されます。
maxRows	オプション		テーブルに表示する行の最大数です。
startRow	オプション	1	テーブルの 1 行目に配置するクエリー結果の行です。

使用方法

このタグでは、テーブルデータの配置、列幅の設定、および列のヘッダの定義が行われます。

このタグには、少なくとも 1 つの cfcol タグが必要です。また、cfcol タグおよび cftable タグはページ内で隣接させて配置します。このタグ内には、cfcol タグだけをネストできます。cftable タグはネストできません。

cfcol の header で指定したテキストを表示するには、cfcol の header 属性と cftable の colHeader 属性を指定します。いずれか一方の属性だけを指定しても、ヘッダは表示されず、エラーも発生しません。

例

```
<!--- This example shows the use of cfcol and cftable to align information
    returned from a query. --->
<!--- This query selects employee information from cfdocexamples datasource. --->
<cfquery name = "GetEmployees" dataSource = "cfdocexamples">
    SELECT Emp_ID, FirstName, LastName, EMail, Phone, Department
    FROM Employees
</cfquery>

<html>
<body>
<h3>cftable Example</h3>

<!--- Note use of HTMLTable attribute to display cftable as an HTML table,
    rather than as PRE formatted information. --->
<cftable query = "GetEmployees"
    startRow = "1" colSpacing = "3" HTMLTable>
<!--- Each cfcol tag sets width of a column in table, and specifies header
    information and text/CFML with which to fill cell. --->
    <cfcol header = "<b>ID</b>"
        align = "Left"
        width = 2
        text= "#Emp_ID#">

    <cfcol header = "<b>Name/Email</b>"
        align = "Left"
        width = 15
        text= "<a href = 'mailto:#Email#'>#FirstName# #LastName#</A>">

    <cfcol header = "<b>Phone Number</b>"
        align = "Center"
        width = 15
        text= "#Phone#">
</cftable>
</body>
</html>
```

cftextarea

説明

cfform タグ内に複数行のテキスト入力ボックスを配置し、表示仕様を制御します。さらにオプションで、HTML テキストの書式を制御するための設定可能なコントロールを含むリッチテキストエディタを表示します。

カテゴリ

フォームタグ

シンタックス

```
<cftextarea
    name="name"
    basepath="path"
    bind="bind expression"
    bindAttribute="attribute name"
    bindOnLoad="false|true"
    disabled="true|false" or no attribute value
    enabled="yes|no"
    fontFormats="comma separated list"
    fontNames="comma separated list"
    fontSizes="comma separated list"
    height="number of pixels"
    html="no|yes"
    label="text"
    maxlength="number"
    message="text"
    onBindError = "JavaScript function name"
    onChange="JavaScript or ActionScript"
    onClick="JavaScript or ActionScript"
    onError="script name"
    onKeyDown="JavaScript or ActionScript"
    onKeyUp="JavaScript or ActionScript"
    onMouseDown="JavaScript or ActionScript"
    onMouseUp="JavaScript or ActionScript"
    onValidate="script name"
    pattern="regexp"
    range="minimum value, maximum value"
    required="yes|no"
    richtext="no|yes"
    secureUpload="true|false"
    skin="default|silver|office2003|custom skin"
    sourceForToolTip="URL"
    style="style specification"
    stylesXML="path"
    templatesXML"path"
    toolbar="Default|Basic|custom toolbar"
    toolbarOnFocus"no|yes"
    tooltip="tip text"
    validate="data type"
    validateAt= one or more of "onBlur, onServer, onSubmit"
    value="text"
    visible="yes|no"
    width="number of pixels"
    wrap="off|hard|soft|virtual|physical">

        text

</cftextarea>
```

一部の属性は、特定の表示形式のみに適用されます。詳細については、「属性」の表を参照してください。

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfajaximport、cfapplet、cfcalendar、cfform、cfformgroup、cfformitem、cfgrid、cfinput、cfselect、cfslider、cftree、『ColdFusion アプリケーションの開発』の Introduction to Retrieving and Formatting Data および Using Ajax form controls and features

履歴
ColdFusion 8:

- HTML 形式フォームで bind 属性を使用できるようになりました (bindAttribute、bindOnLoad、onBindError 属性を含む)。
- HTML 形式でツールヒントを使用できるようになりました (sourceForTooltip 属性など)。
- HTML 形式でリッチテキストエディタを使用できるようになりました (richtext、basepath、fontFormats、fontNames、fontSizes、skin、stylesXML、templatesXML、toolbar、toolbarOnFocus 属性など)。height および width 属性のサポートが追加されました。

ColdFusion MX 7: このタグが追加されました。

属性
次の表は、ColdFusion が直接使用する属性のリストです。HTML 形式では、このリストに記載されていない HTML textarea タグ属性もすべてサポートされ、ブラウザに直接渡されます。

注意：「すべて」または「XML」と記載されてない属性は、ColdFusion に同梱されているスキンでは処理されません。ただし、これらの属性は生成された XML に含まれます。

属性	必須 / オプション、 形式	デフォルト	説明
name	必須、 すべて		cftextinput コントロールの名前です。
basepath	オプション、 HTML	/CFIDE/scripts/ajax/FCKEditor	リッチテキストエディタが存在するディレクトリのパスです。エディタの設定ファイルは、このディレクトリのトップレベルにあります。 richText 属性が true の場合にのみ効果を持ちます。
bind	オプション、 Flash、 HTML		コントロールの属性をダイナミックに設定するバインド式です。詳細については、「使用方法」を参照してください。
bindAttribute	オプション、 HTML	value	bind 属性を使用して値を設定する HTML タグ属性を指定します。ColdFusion 固有の属性ではなく、ブラウザの HTML DOM ツリーに含まれる属性のみを指定できます。 bind 属性がない場合は無視されます。
bindOnLoad	オプション、 HTML	no	フォームを最初にロードするときに bind 属性の式を実行するかどうかを指定するブール値です。bind 属性がない場合は無視されます。
disabled	オプション、 すべて	no	ユーザー入力を無効にし、コントロールを読み取り専用にします。入力を無効にするには、属性を省略して disabled を指定するか、disabled="yes" (または ColdFusion で使用する true などの真を表すブール値) を指定します。入力を有効にするには、属性を省略するか、disabled="no" (または ColdFusion で使用する false などの偽を表すブール値) を指定します。
enabled	オプション、 Flash	yes	コントロールを有効にするかどうかを指定するブール値です。無効なコントロールはライトグレーで表示されます。disabled 属性の逆です。

属性	必須 / オプション、 形式	デフォルト	説明
fontFormats	オプション、HTML	すべての有効な形式	リッチテキストエディタの [形式] セレクタに表示されるフォント名のカンマ区切りリストです。形式を選択すると、入力または選択されたテキストに適用する HTML タグが決定されます。p、div、pre、address、h1、h2、h3、h4、h5、h6 の中から任意のタグを指定できます。
fontNames	オプション、HTML	すべての有効なフォント名	リッチテキストエディタの [フォント] セレクタに表示されるフォント名のカンマ区切りリストです。Arial、Comic Sans MS、Courier New、Tahoma、Times New Roman、Verdana の中から任意のフォントを指定できます。
FontSizes	オプション、HTML	「説明」を参照	リッチテキストエディタの [サイズ] セレクタに表示されるフォントサイズのカンマ区切りリストです。このリストのエントリは、「< フォントサイズの値 >/< 説明テキスト >」という形式で指定する必要があります。たとえば、フォントサイズ 1 を示すために "small font" というテキストを表示するには、"1/small font" と指定します。デフォルトでは、1/xx-small、2/x-small、3/small、4/medium、5/large、6/x-large、および 7/xx-large という値がセレクタに表示されます。
height	オプション、Flash、HTML		Flash フォームの場合は、コントロールの高さをピクセル単位で指定します。 HTML フォームの場合は、richtext="true" が指定されている場合にのみ効果を持ち、その場合はコントロール (コントロールバーやテキストボックスなど) の高さをピクセル単位で指定します。
html	オプション、Flash	no	テキスト領域に HTML が含まれるかどうかを指定するブール値です。
label	オプション、Flash および XML		フォーム内のコントロールの横に配置するラベルです。
maxLength	オプション、すべて		入力可能なテキストの最大長です。この属性は、validate 属性で maxlength が指定されている場合にのみ効果を持ちます。
message	オプション、すべて		検証に失敗した場合に表示されるメッセージテキストです。
onBindError	オプション、HTML	「説明」を参照	バインド式の評価でエラーが発生した場合に実行する JavaScript 関数の名前です。この関数は 2 つの属性 (HTTP ステータスコードとメッセージ) を取る必要があります。発生したエラーが HTTP エラーでない場合、ステータスコードは -1 です。 この属性を省略し、(ColdFusion.setGlobalErrorHandler 関数を使用して) グローバルエラーハンドラを指定すると、エラーメッセージが表示されます。それ以外の場合は、デフォルトのエラーポップアップが表示されます。
onChange	オプション、すべて		ユーザーのアクションに応じてコントロールに変化があったときに実行される JavaScript (HTML/XML の場合) または ActionScript (Flash の場合) です。
onClick	オプション、HTML および XML		ユーザーがコントロールをクリックしたときに実行される JavaScript です。

属性	必須 / オプション、 形式	デフォルト	説明
onError	オプション、 HTML および XML		検証に失敗した場合に実行するカスタム JavaScript 関数です。
onKeyDown	オプション、 すべて		ユーザーがコントロールでキーボードのキーを押したときに実行される JavaScript (HTML/XML の場合) または ActionScript (Flash の場合) です。
onKeyUp	オプション、 すべて		ユーザーがコントロール内でキーボードのキーを離したときに実行される JavaScript (HTML/XML の場合) または ActionScript (Flash の場合) です。
onMouseDown	オプション、 すべて		ユーザーがコントロール内でマウスボタンを離したときに実行される JavaScript (HTML/XML の場合) または ActionScript (Flash の場合) です。
onMouseUp	オプション、 すべて		ユーザーがコントロール内でマウスボタンを押したときに実行される JavaScript (HTML/XML の場合) または ActionScript (Flash の場合) です。
onValidate	オプション、 HTML および XML		ユーザー入力を検証するためのカスタム JavaScript です。JavaScript DOM フォームオブジェクト、入力オブジェクト、および入力オブジェクト値が関数に渡されます。検証に成功すると true が返されます。検証に失敗すると false が返されます。この属性を指定した場合、ColdFusion は validate 属性を無視します。
pattern	validate = "regular_expression" の場合は必須 HTML および XML		入力を検証するための、JavaScript 正規表現のパターンです。先頭および末尾のスラッシュ記号は削除されます。この属性は、validate 属性で **regex** が指定されている場合にのみ効果を持ちます。例とシンタックスについては、『ColdFusion アプリケーションの開発』の Building Dynamic Forms with cfform Tags を参照してください。
range	オプション、 すべて		指定可能な数値の最大値と最小値です。この属性は、validate 属性で **range** が指定されている場合にのみ効果を持ちます。 単一の値を指定するか、単一の値の後にカンマを付けて指定した場合、その値は最小値として処理されます。最大値はありません。カンマの後に単一の値を指定した場合、その値は最大値として設定されます。最小値はありません。
required	オプション、 すべて	no	• yes: フィールドにはテキストが必要です。 • no: フィールドは空白のままでもかまいません。
richText	オプション、 HTML	no	このコントロールが、テキストの書式を制御するためのツールバーを含むリッチテキストエディタであるかどうかを指定するブール値です。リッチテキストエディタの使用方法の詳細については、『ColdFusion アプリケーションの開発』の Using the rich text editor を参照してください。
skin	オプション、 HTML	default	リッチテキストエディタで使用するスキンを指定します。デフォルトで有効な値は、default、silver、および office2003 です。カスタムスキンを作成して、そのスキンをこの属性で指定することもできます。詳細については、『ColdFusion アプリケーションの開発』の Using the rich text editor を参照してください。

属性	必須 / オプション、 形式	デフォルト	説明
sourceForTooltip	オプション、HTML		ツールヒントとして表示するページの URL です。このページには内容と書式を制御するための CFML および HTML を含めることができ、ヒントにはイメージを含めることができます。 この属性を指定すると、ヒントのロード中にアニメーションのアイコンと「ロード中 ...」というテキストが表示されます。
style	オプション、 すべて		HTML または XML 形式フォームの場合、style 属性はブラウザまたは XML に渡されます。 Flash 形式フォームでは、対応する Flash 要素について、Flex で使用するものと同じシンタックスおよびコンテンツを使用する CSS 形式のスタイル仕様でなければなりません。
stylesXML	オプション、HTML	/CFIDE/scripts/ajax/FCKEditor/fckstyles.xml	リッチテキストエディタの [スタイル] セレクタに表示されるスタイルを定義したファイルのパスです。fckeditor.html ファイルが存在するディレクトリ (通常は <ColdFusion の Web ルート >/CFIDE/scripts/ajax/FCKeditor/editor) からの相対パスです。/myApps/RTEeditor.mystyles.xml のように Web ルートから始まる絶対パスを指定できます。スタイルの設定については、『ColdFusion アプリケーションの開発』の Using the rich text editor を参照してください。
templatesXML	オプション、HTML	/CFIDE/scripts/ajax/FCKEditor/fcktemplates.xml	リッチテキストエディタの [テンプレート] アイコンをクリックしたときに表示されるテンプレートを定義するファイルのパスです。パスの指定方法については、stylesXML を参照してください。テンプレートの設定については、『ColdFusion アプリケーションの開発』の Using the rich text editor を参照してください。
toolbar	オプション、HTML	Default	リッチテキストエディタのツールバーを指定します。デフォルトで有効な値は、Default (完全なコントロールのセット) と Basic (最小限の設定) です。設定は追加することができます。詳細については、『ColdFusion アプリケーションの開発』の Using the rich text editor を参照してください。 **メモ :** この属性では大文字と小文字が区別されます。
toolbarOnFocus	オプション、HTML	no	リッチテキストエディタがフォーカスを持っているときにのみリッチテキストエディタツールバーのコントロールを展開して表示するかどうかを指定するブール値です。
tooltip	オプション、Flash、HTML		マウスポインタをコントロールの上に置いたときに表示されるテキストです。HTML 書式設定も使用できます。 sourceForTooltip 属性が指定されている場合は無視されます。
validate	オプション、 すべて		実行する検証のタイプです。使用可能な検証のタイプとアルゴリズムは、形式によって異なります。詳細については、cfinput タグリファレンスの「使用方法」を参照してください。
validateAt	オプション、 HTML および XML	onSubmit	検証の実行方法です。次の中から 1 つ以上を指定します。 • onSubmit • onServer • onBlur Flash 形式フォームの場合、onSubmit と onBlur は同一です。検証はいずれもフォームの送信時に実行されます。複数の値を指定する場合は、カンマ区切りのリストを使用します。 詳細については、cfinput タグリファレンスの「使用方法」を参照してください。

属性	必須 / オプション、 形式	デフォルト	説明
value	オプション、 すべて		テキストコントロール内に表示される初期値です。初期値は属性またはタグ本文を使って指定できますが、これら両方の場所で指定することはできません。属性として値を指定する場合は、cftextarea 開始タグの直後に (間にスペースや改行をはさまずに) cftextarea 終了タグを置くか、cftextarea 開始タグの末尾に終了のスラッシュを置きます。例 : <cftextareaname="description"value="Enteradescription."/>
visible	オプション、 Flash	yes	コントロールを表示するかどうかを指定するブール値です。表示されないコントロールが使用するスペースは空白です。
width	オプション、 Flash、 HTML		コントロールの幅をピクセル単位で指定します。 HTML フォームの場合は、richtext="true" が指定されている場合にのみ効果を持ちます。
wrap	オプション すべて		• hard: 長い行を折り返し、復帰文字をサーバーに送信します。 • off: 長い行を折り返しません。 • physical: 長い行を折り返し、すべての折り返し点でテキストをサーバーに伝送します。 • soft: 長い行を折り返しますが、復帰文字をサーバーに送信しません。 • virtual: 長い行を折り返しますが、復帰文字をサーバーに送信しません。

使用方法

HTML 形式でこのタグを正しく動作させるには、JavaScript 対応のブラウザを使用する必要があります。

テキストをタグ本文に配置すると、コントロールでは cftextarea の開始タグと終了タグの間にあるすべてのテキスト文字が表示されます。したがって、改行や空白を使ってソーステキストの形式を設定した場合、それらはコントロールに表示されます。

cfform の preserveData 属性が "yes" で、フォームが同じページに送信される場合は、value 属性の値ではなく cftextinput コントロールの送信された値が使用されます。

検証

validation 属性の詳細、および ColdFusion がサポートする検証のタイプの詳細については、cfinput タグリファレンスの「使用方法」を参照してください。ColdFusion の検証方法の詳細については、『ColdFusion アプリケーションの開発』の Validating Data を参照してください。

Flash フォームデータのバインディング

bind 属性を使用すると、フォームフィールドに他のフォームフィールドの内容を挿入できます。cftextarea の bind 属性で他の Flash フォームフィールドのテキストを指定するには、次の形式を使用します。

{**sourceTagName**.text}

たとえば、次の例では、ユーザーが userName フィールドに入力したテキスト値が、コメントテキストボックスの挨拶で使用されます。ユーザーはこのメッセージを変更できます。

```
<cfform format="flash" height="300">
    <cfformitem type="text">
        Enter your name here</cfformitem>
    <cftextarea name="userName" height="20" Width="500"/>
    <cftextarea name="comment" height="100" Width="500"
        bind="Hello {userName.text}! Enter your comments here." />
</cfform>
```

HTML フォームデータのバインディング

bind 属性を使用すると、別のフォームコントロールの値を使用したり、CFC 関数または JavaScript 関数を呼び出すことによって、cftextarea の属性をダイナミックに設定することができます。デフォルトでは、コントロールの value 属性が設定されますが、bindAttribute 属性を使用することで、他の属性を設定することもできます。バインディングの詳細については、『ColdFusion アプリケーションの開発』の Binding data to form fields を参照してください。

例

この例には 2 つの cftextarea コントロールがあります。フォームが送信されると、1 番目のコントロールから 2 番目のコントロールにテキストがコピーされます。onBlur maxlength 検証により、100 文字を超えて入力しないように制限されます。cftextarea 開始タグを閉じる > 文字、タグ本文内のテキスト、および cftextarea 終了タグは、目的のテキストのみが表示されるように 1 行に収められますが、この例では、形式設定を行うために行が分割されています。

```
<h3>cftextarea Example</h3>
<cfparam name="text2" default="">
<cfif isdefined("form.textarea1") AND (form.textarea1 NEQ "")>
    <cfset text2=form.textarea1>
</cfif>

<cfform name="form1">
    <cftextarea name="textarea1" wrap="virtual" rows="5" cols="25"
    validate="maxlength" validateAt="onBlur" maxlength="100"
    >Replace this text. Maximum length is 100 Characters, and this text is
    currently 99 characters long.</cftextarea>
    <cftextarea name="textarea2" wrap="virtual" rows="5" cols="50" value="#text2#" /><br><br>
    <input type="submit" value="submit field"><br>
</cfform>
```

cftextinput

説明

cfform タグ内に 1 行のテキスト入力ボックスを配置し、表示仕様を制御します。

このタグは非推奨となっています。XML 形式のフォーマットではサポートされません。代わりに、cfinput または cftextarea タグを使用し、CSS (カスケーディングスタイルシート) を使用してテキストスタイル特性を指定する必要があります。

履歴

ColdFusion MX 7: このタグは非推奨となっています。今後のリリースではこのタグは機能せず、エラーを引き起こす可能性があります。

ColdFusion MX 6.1: validate = "creditcard" オプションの必要条件が変更されました。13 ～ 16 桁のテキスト入力が必要です。

cfthread

説明

cfthread タグを使用すると、コード実行から独立したストリーム (スレッド) を ColdFusion アプリケーションで作成できます。このタグでは、スレッドの実行と終了、スレッド実行の一時停止、複数のスレッドの結合などを実行できます。

カテゴリ

アプリケーションフレームワークタグ

シンタックス

```
join
    <cfthread
    required
    name="thread name[,thread name]..."
    optional
    action="join"
    timeout="milliseconds"/>

run
    <cfthread
    required
    name="thread name"
    optional
    action="run"
    priority="NORMAL|HIGH|LOW"
    zero or more application-specific attributes>

        Thread code

    </cfthread>

sleep
    <cfthread
    required
    action="sleep"
    duration="milliseconds"/>

terminate
    <cfthread
    required
    action="terminate"
    name="thread name"/>
```

run アクションの場合を除き、cfthread タグの本文には何も記述せず、開始タグの直後に終了タグ </cfthread> を配置する必要があります。あるいは、終了タグを使用しないで、タグを閉じる前にスラッシュを配置する必要があります。例 : <cfthread action="sleep" duration="1000"/>

注意 : このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

Sleep、『ColdFusion アプリケーションの開発』の Using ColdFusion Threads

履歴

ColdFusion 8: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	適用対象	説明
action	オプション	run	すべて	実行するアクションです。次のいずれかの値を指定します。 • join: name 属性で指定したスレッドの処理が完了するか、timeout 属性で指定した時間が経過するまで待機してから、現在のスレッドの処理を再開します。タイムアウトを指定しないと、結合対象のスレッドが完了するまで、現在のスレッドの処理も完了できません。 • run: スレッドを作成して処理を開始します。cfthread タグ本文のコードはページレベルのコードや他の cfthread タグのコードと同時に実行されますが、各コードは相互に独立して実行されます。 • sleep: duration 属性で指定されている時間が経過するまで現在のスレッドの処理を中断します。Sleep 関数と同等です。 • terminate: name 属性で指定されているスレッドの処理を停止します。スレッドを終了すると、終了に関する情報を含む ERROR メタデータ構造体がスレッドスコープに追加されます。
duration	必須		sleep	スレッドの処理を中断する時間です (単位 : ミリ秒)。
name	オプション、action = "join" または "terminate" の場合は必須		join run terminate	アクションを適用するスレッドの名前です。 • terminate: 停止するスレッドの名前を指定します。 • join: 現在のスレッドに結合するスレッドの名前を指定します。複数のスレッドを指定するには、カンマ区切りリストを使用します。 • run: 作成するスレッドを識別するための名前を指定します。
priority	オプション	NORMAL	run	スレッドを実行するときの優先度レベルです。使用できる値は次のとおりです。 • HIGH • LOW • NORMAL 優先度レベルが高いスレッドほど多くの処理時間が割り当てられます。ページレベルのコード (cfthread タグの外にあるコード) の優先度レベルは常に NORMAL になります。
timeout	オプション	0	join	結合対象のスレッドが完了するまで現在のスレッドを中断する時間です (単位 : ミリ秒)。指定した時間内に完了しないスレッドがあった場合、現在のスレッドが再開されます。 この属性の値が 0 の場合は、次の処理が行われます。 • 結合対象のスレッドがすべて完了するまで現在のスレッドが中断されます。 • 現在のスレッドがページスレッドの場合は、ページタイムアウトが指定されている場合でも、スレッドの結合が完了するまでページの処理が中断されます。

使用方法

ページレベルのコード (cfthread タグの外にあるコード) は、ページスレッドと呼ばれる独自のスレッドで実行されます。スレッドを作成できるのはページスレッドのみです。ユーザーによって作成されたスレッドは子スレッドを作成できません。

注意 : スレッドの処理が完了しない (ハングした) 場合、そのスレッドはシステムリソースを占有し続けます。ColdFusion Sever Monitor を使用すると、ハングしたスレッドがないかどうかを確認して終了させることができます。

ColdFusion は、(ページスレッドを含め) 各スレッド内の変数値の独立性を保つために、すべての属性変数を完全にコピーしてから各スレッドに属性値を渡します。したがって、他のスレッドによって属性値が変更される可能性がなくなるので、各スレッドに渡される値はスレッドセーフになります。

スレッドのスコープ

各スレッドには、次の 3 つの特殊なスコープがあります。

- スレッドローカルスコープは、現在のスレッドに対してのみ使用できる変数を含む暗黙のスコープであり、現在のスレッドが存続している間だけ存在します。
- Thread スコープは、現在のページだけでなく、現在のページから開始されたすべてのスレッドで使用できます。このスコープのデータは、現在のページと現在のページから開始されたすべてのスレッドが完了するまで保持されます (それらのスレッドが完了する前に現在のページが完了した場合でも保持されます)。
- Attributes スコープには、そのスコープに渡された属性が格納されます。Attributes スコープは現在のスレッド内でのみ使用でき、現在のスレッドが存続している間だけ存在します。

  スレッド内で ColdFusion スコープを使用する方法の詳細については、『ColdFusion アプリケーションの開発』の Using ColdFusion Threads を参照してください。

  同じページ内のすべてのスレッドは 1 つの Variables スコープを共有するので、すべてのスレッドに共通のデータがある場合は、このスコープを使用すると便利です。ただし、スレッド間のデッドロックや競合状態を防止するために、必要に応じて変数へのアクセスをロックする必要があります。

  注意：ColdFusion がコネクタを使用して Web サーバーにアクセスする場合、ページの完了後には、cfthread タグを使用して作成したスレッドから CGI スコープおよび Request スコープにアクセスできなくなります。ただし、この制限は、統合型の Web サーバーを使用している場合、または J2EE アプリケーションとして ColdFusion を実行している場合には適用されません。

メタデータ変数

スレッドスコープには、スレッドに関する情報 (メタデータ) を提供する次の変数が含まれます。

変数	説明
ElapsedTime	スレッドの処理に使用された合計プロセッサ時間です。
Error	スレッドの実行中にエラーが発生した場合に生成される構造体です。この構造体には、cfcatch タグでアクセスできるキーが含まれます。 スレッド内でエラーが発生しても、ページレベルの処理は影響を受けず、エラーメッセージは生成されません。エラーが発生したスレッドは終了されます。Error フィールドのエラー情報をページレベルのコードまたは他のスレッドで取得して、エラーを適切に処理することもできます。詳細については、『ColdFusion アプリケーションの開発』の Handling ColdFusion thread errors を参照してください。
Name	スレッド名。
Output	スレッドによって生成される出力です。スレッドは出力を表示できません。スレッドの結果を表示するには、ページレベルのコードでこの変数を使用する必要があります。詳細については、『ColdFusion アプリケーションの開発』の Handling thread output を参照してください。

変数	説明
Priority	cfthread の priority 属性で指定されたスレッド処理の優先度です。使用できる値は次のとおりです。 • HIGH • LOW • *NORMAL*
Starttime	スレッドの処理が開始された時刻です (ColdFusion の日付時刻形式)。
Status	スレッドの現在のステータスです。次のいずれかの値が入ります。 • NOT_STARTED: このスレッドはキューに格納されていますが、まだ処理されていません。 • RUNNING: このスレッドは正常に実行されています。 • TERMINATED: terminate アクションが設定された cfthread タグ、エラー、または管理者の操作によって、このスレッドの実行が停止されました。 • COMPLETED: このスレッドは正常に終了しました。 • WAITING: このスレッドで action="join" に設定された cfthread タグの実行を開始しましたが、結合対象のスレッドがまだ完了していません。

例

次の例では、3 つのスレッドを使用して 3 つの RSS フィードの結果を取得します。ユーザーは、指定された 3 つのフィードをすべて使用してフォームを送信する必要があります。アプリケーションでは、6 秒間のタイムアウトを指定してスレッドを結合し、フィードのタイトルと、個々の項目タイトルをリンクとして表示します。

```
<!--- Run this code if the feed URL form has been submitted. --->
<cfif isDefined("Form.submit")>
    <cfloop index="i" from="1" to="3">
        <!--- Use array notation and string concatenation to create a variable
                 for this feed. --->
        <cfset theFeed = Form["Feed"&i]>
        <cfif theFeed NEQ "">
            <!--- Use a separate thread to get each of the feeds. --->
            <cfthread action="run" name="t#i#" feed="#theFeed#">
                <cffeed source="#feed#"
                    properties="thread.myProps"
                    query="thread.myQuery">
            </cfthread>
        <cfelse>
            <!--- If the user didn't fill all fields, show an error message. --->
            <h3>This example requires three feeds.<br />
            Click the Back button and try again.</h3>
            <cfabort>
        </cfif>
    </cfloop>

    <!--- Join the three threads. Use a 6 second time-out. --->
    <cfthread action="join" name="t1,t2,t3" timeout="6000" />

    <!--- Use a loop to display the results from the feeds. --->
    <cfloop index="i" from="1" to="3">
        <!--- Use the cfthread scope and associative array notation to get the
                Thread scope dynamically. --->
        <cfset feedResult=cfthread["t#i#"]>
        <!--- Display feed information only if you got items,
                 for example the feed must complete before the join. --->
        <cfif isDefined("feedResult.myQuery")>
```

```
            <cfoutput><h2>#feedResult.myProps.title#</h2></cfoutput>
            <cfoutput query="feedResult.myQuery">
                <p><a href="#RSSLINK#">#TITLE#</a></p>
            </cfoutput>
        </cfif>
    </cfloop>

</cfif>

<!--- The form for entering the feeds to aggregate. --->
<cfform>
    <h3>Enter three RSS Feeds</h3>
    <cfinput type="text" size="100" name="Feed1" validate="url"
        value="http://rss.adobe.com/events.rss?locale=en"><br />
    <cfinput type="text" size="100" name="Feed2" validate="url"
        value="http://weblogs.macromedia.com/dev_center/index.rdf"><br />
    <cfinput type="text" size="100" name="Feed3" validate="url"
        value="http://rss.adobe.com/studio.rss?locale=en"><br />
    <cfinput type="submit" name="submit">
</cfform>
```

cfthrow

説明

開発者指定の例外を返します。この例外は、cfcatch タグで次のいずれかの type 属性オプションを指定して検出できます。

- type = "**custom_type**"
- type = "Application"
- type = "Any"

カテゴリ

例外処理タグ、フロー制御タグ

シンタックス

```
<cfthrow
    message = "message"
    type = "exception type"
    detail = "detail description"
    errorCode = "error code"
    extendedInfo = "additional information"
    object = "java except object">

OR

<cfthrow
    object = #object_name#>
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cferror、cfrethrow、cftry、onError、『ColdFusion アプリケーションの開発』の Handling Errors

履歴

ColdFusion MX: 返される例外が変更されました。このタグは、ColdFusion コンポーネントメソッドの例外を発生させることができます。

属性

属性	必須 / オプション	デフォルト	説明
detail	オプション		イベントの説明です。ColdFusion によって、説明の後にエラー位置が追加されます。コードによってエラーが検出されない場合は、サーバーがこのパラメータを使用します。
errorCode	オプション		ユーザーが指定するカスタムエラーコードです。
extendedInfo	オプション		ユーザーが指定するカスタムエラーコードです。
message	オプション		例外的なイベントを示すメッセージです。
object	オプション		cfobject タグの name 属性の値が必要です。 CFML タグからの Java 例外を返します。 この属性は、このタグの他の属性すべてと排他の関係にあります。
type	オプション	Application	• カスタムタイプ • Application 他の定義済みのタイプは ColdFusion アプリケーションでは生成されないため、入力しないでください。Application を指定する場合は、cfcatch のタイプを指定する必要はありません。

使用方法

エラーを返すには、このタグを cftry ブロック内で使用します。cfcatch ブロックを使用すると、次のような付随情報にアクセスできます。

- メッセージ (cfcatch.message を使用した場合)
- 詳細 (cfcatch.detail を使用した場合)
- エラーコード (cfcatch.errorcode を使用した場合)

  さらに情報を取得するには、cfcatch.tagContext を使用してください。この配列は、タグスタック内でページからページに制御が移行する位置を示します (例 : cfinclude、cfmodule)。

  tagContext 変数の情報を表示するには、ColdFusion Administrator の [デバッグとロギング]-[デバッグ出力の設定] ページで [Robust 例外情報の有効化] オプションを選択します。

  このタグを object パラメータとともに使用するには、まず有効な Java 例外クラスを指定する cfobject タグを使用します。たとえば、次の cfobject タグでは、例外クラス myException (これは Java で作成する必要があります) のオブジェクト obj を定義しています。

```
<cfobject
    type="java"
    action="create"
    class="myException"
    name="obj">
```

  例外クラスにメッセージなどのパラメータを取るコンストラクタがある場合は、次のように特殊な init メソッドを使用してコンストラクタを呼び出すことができます。コンストラクタの属性を指定する必要がない場合は、この手順を省略できます。

```
<cfset obj.init("You must save your work before preceding")>
```

次に cfthrow ステートメントを使用して、次のように例外を返すことができます。

```
<cfthrow object=#obj#>
```

ColdFusion での Java オブジェクトの使用方法の詳細については、『ColdFusion アプリケーションの開発』の Integrating J2EE and Java Elements in CFML Applications を参照してください。

例

```
<h3>cfthrow Example</h3>
<!--- Open a cftry block. --->
<cftry>
<!--- Define a condition upon which to throw the error. --->
<cfif NOT IsDefined("URL.myID")>
    <!--- throw the error --->
    <cfthrow message = "ID is not defined">
</cfif>
<!--- Perform the error catch. --->
<cfcatch type = "application">
<!--- Display your message. --->
    <h3>You've Thrown an <b>Error</b></h3>
<cfoutput>
    <!--- And the diagnostic feedback from the application server. --->
<p>#cfcatch.message#</p>
    <p>The contents of the tag stack are:</p>
    <cfloop
        index = i
        from = 1 to = #ArrayLen(cfcatch.tagContext)#>
                <cfset sCurrent = #cfcatch.tagContext[i]#>
                    <br>#i# #sCurrent["ID"]#
                (#sCurrent["LINE"]#,#sCurrent["COLUMN"]#)
                #sCurrent["TEMPLATE"]#
    </cfloop>
</cfoutput>
</cfcatch>
</cftry>
```

次の例では、コンポーネントメソッドから例外を返す方法を示します。

```
<cfcomponent>
    <cffunction name="getEmp">
    <cfargument name="lastName" required="yes">
         <cfquery name="empQuery" datasource="cfdocexamples" >
            SELECT LASTNAME, FIRSTNAME, EMAIL
            FROM tblEmployees
            WHERE LASTNAME LIKE '#arguments.lastName#'
         </cfquery>
         <cfif empQuery.recordcount LT 1>
         <cfthrow type="noQueryResult"
                message="No results were found. Please try again.">
            <cfelse>
                <cfreturn empQuery>
        </cfif>
    </cffunction>
</cfcomponent>
```

この例の説明と、さらに詳しい情報については、『ColdFusion アプリケーションの開発』の Building and Using ColdFusion Components を参照してください。

cftimer

説明
CFML コードの指定されたセクションの実行時間を表示します。ColdFusion では、時間コードによって生成されたすべての出力とともに時間情報が表示されます。

注意：このタグを実行できるようにするには、ColdFusion Administrator の [デバッグの設定] ページで [デバッグの有効化] および [タイマー情報] オプションを有効にします。また、リモートマシンからリクエストが送信された場合は、ColdFusion が実行されているマシンの IP アドレスも [デバッグする IP アドレス] ページのデバッグ IP アドレスのリストに追加する必要があります。リクエストがローカルホストから送信された場合は、IP アドレス 127.0.0.1 がデバッグ IP アドレスのリストに存在する必要があります。

カテゴリ
デバッグタグ

シンタックス
```
<cftimer
    label= "text"
    type = "inline|outline|comment|debug" >

    CFML statement(s)

</cftimer>
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目
cfdump、cftrace、『ColdFusion アプリケーションの開発』の Debugging and Troubleshooting Applications

履歴
ColdFusion MX 7: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
label	オプション	" "	時間情報とともに表示するラベルです。
type	オプション	debug	• inline: 結果 HTML に続いて、時間情報をインラインで表示します。 • outline: 時間情報を表示し、時間コードによって作成された出力の周囲の行も表示します。ブラウザが FIELDSET タグをサポートしてアウトラインを表示できる必要があります。 • comment: <!-- <ラベル>: <経過時間> ms --> の形式で HTML コメント内に時間情報を表示します。デフォルトのラベルは cftimer です。 • debug: CFTimer Times ヘッダの下のデバッグ出力に時間情報を表示します。

使用方法
このタグを使用して、コードブロックの実行時間を確認することができます。cftimer タグをネストできます。

このタグは、アプリケーション開発時の CFML コードのデバッグに役立ちます。本番環境では、ColdFusion Administrator でデバッグオプションを無効にしている限り、cftimer タグをコード内にそのまま残すことができます。

例

```
...
<!--- type="inline"> --->
    <cftimer label="Query and Loop Time Inline" type="inline">
        <cfquery name="empquery" datasource="cfdocexamples">
            SELECT *
            FROM Employees
        </cfquery>

        <cfloop query="empquery">
        <cfoutput>#lastname#, #firstname#</cfoutput><br>
        </cfloop>
    </cftimer>
<hr><br>

<!--- type="outline" --->
    <cftimer label="Query and CFOUTPUT Time with Outline" type="outline">
        <cfquery name="coursequery" datasource="cfdocexamples">
            SELECT *
            FROM CourseList
        </cfquery>
    <table border="1" width="100%">
        <cfoutput query="coursequery">
        <tr>
        <td>#Course_ID#</td>
        <td>#CorName#</td>
        <td>#CorLevel#</td>
        </tr>
        </cfoutput>
        </table>
    </cftimer>
<hr><br>

<!--- type="comment" --->
    <cftimer label="Query and CFOUTPUT Time in Comment" type="comment">
        <cfquery name="parkquery" datasource="cfdocexamples">
                SELECT *
                FROM Parks
        </cfquery>
<p>Select View &gt; Source to see timing information</p>
    <table border="1" width="100%">
        <cfoutput query="parkquery">
        <tr>
        <td>#Parkname#</td>
        </tr>
        </cfoutput>
        </table>
```

```
    </cftimer>
<hr><br>

<!--- type="debug" --->
    <cftimer label="Query and CFOUTPUT Time in Debug Output" type="debug">
        <cfquery name="deptquery" datasource="cfdocexamples">
                SELECT *
                FROM Departments
        </cfquery>
<p>Scroll down to CFTimer Times heading to see timing information</p>
    <table border="1" width="100%">
        <cfoutput query="deptquery">
        <tr>
        <td>#Dept_ID#</td>
        <td>#Dept_Name#</td>
        </tr>
        </cfoutput>
        </table>
    </cftimer>
...
```

cftooltip

説明
ユーザーがタグ本文の要素の上にマウスポインタを置いたときに表示されるツールヒントテキストを指定します。このタグはフォームがなくても使用でき、Flash フォーム内では使用されません。

カテゴリ
表示管理タグ

シンタックス

```
<cftooltip
    autoDismissDelay="5000"
    hideDelay="250"
    preventOverlap="true|false"
    showDelay="200"
    sourceForTooltip="URL"
    style="CSS style specification"
    tooltip="text">

    Display tags

</cftooltip>
```

このタグには終了タグが必要です。

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目
cfajaximport、『ColdFusion アプリケーションの開発』の Using Ajax User Interface Components and Features

履歴
ColdFusion 8: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
autoDismissDelay	オプション	5000	ユーザーがコンポーネントの上にマウスポインタを置いたままにしてからツールヒントを隠すまでの時間をミリ秒単位で指定します。
hideDelay	オプション	250	ユーザーがコンポーネントの上からマウスポインタを移動してからツールヒントを隠すまでの時間をミリ秒単位で指定します。
preventOverlap	オプション	true	ツールヒントがコンポーネントの上に重なって表示されないようにするかどうかを指定するブール値です。
showDelay	オプション	200	ユーザーがコンポーネントの上にマウスポインタを置いてからツールヒントを表示するまでの時間をミリ秒単位で指定します。
sourceForTooltip	オプション		ツールヒントの内容を含むページの URL です。このページには書式を制御するための HTML マークアップを含めることができ、ヒントにはイメージを含めることができます。 この属性を指定すると、ヒントのロード中にアニメーションのアイコンと「ロード中 ...」というテキストが表示されます。
style	オプション		ツールヒントの CSS スタイル指定です。この属性を使用して、幅、テキストカラー、背景色、パディングなどのスタイルプロパティを設定します。
tooltip	オプション		表示するヒントテキスト。テキストには HTML 書式設定も使用できます。 sourceForTooltip 属性が指定されている場合は無視されます。

使用方法

tooltip 属性または sourceForTooltip 属性が指定されていない場合、このタグは無視されます。

sourceForTooltip 属性で CFML ページのパスを指定した場合は、ColdFusion によってそのページが処理され、その出力がヒントテキストで使用されます。したがって、ヒントテキストの内容と外観の制御には、HTML 書式設定だけでなく CFML プログラミングも使用できます。

ウィンドウ、ポッド、レイアウト領域などの複雑な Ajax コンポーネントではなく、テキストやイメージなどの単純なコンポーネントを利用する場合は、cftooltip タグを使用する必要があります。複雑なコンポーネントに対して cftooltip タグを使用した場合は、予期しない動作が発生する可能性があります。たとえば、preventoverlap 属性を指定しても、ツールヒントがウィンドウの内容の上に重なって表示される場合があります。

ツールヒントは、cfinput、cfgrid、および cfform タグ内にネストできます。ただし、これにより、複数のツールヒントがわかりにくくなることがあります。

例

次の簡単な例では、メインの CFML ページの theItem 変数の値に基づいて、異なるツールヒントテキストをダイナミックに表示できます。

メインの CFML ページ :

```
<!--- These variables could be set dynamically --->
<cfset theItem="left-handed & other specialty wrenches">
<cfset theImage="lhbwrench.jpg">

<!--- The theItem string has an ampersand, so you must URL-encode it. --->
<cftooltip sourceForTooltip="tiptext.cfm?itemid=#URLEncodedFormat(theItem)#">
    <cfoutput>
        <b>Try this one!</b>
        <img src="#theImage#" />
    </cfoutput>
</cftooltip>
```

tiptext.cfm ページには CFML タグが 1 つだけ含まれています。

```
<cfoutput><b> Click to find more about #URL.itemid# </b></cfoutput>
```

cftrace

説明

cftrace タグ実行時のアプリケーションの状態について、デバッグデータを表示し、ロギングを行います。実行時のロジックフロー、変数値、および実行時間を追跡します。リクエストの最後、またはリクエストの最後のデバッグセクションに出力を表示します。または、Dreamweaver MX 以降では、[Results] ウィンドウの [Server Debug] タブに出力を表示します。

ColdFusion では、ColdFusion インストールディレクトリ内のファイル logs¥cftrace.log に cftrace の出力がロギングされます。

注意：このタグを実行可能にするには、ColdFusion Administrator でデバッグを有効にします。オプションとして、トレースの要約を報告するには、[トレース] セクションを有効にします。

カテゴリ

デバッグタグ、変数操作タグ

シンタックス

```
<cftrace
    var = "variable name"
    text = "string"
    type = "format"
    category = "string"
    inline = "yes|no"
    abort = "yes|no">
</cftrace>
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfdump、cferror、cfrethrow、cftimer、cftry、『ColdFusion アプリケーションの開発』の Debugging and Troubleshooting Applications

履歴

ColdFusion MX: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
abort	オプション	no	• yes: このタグを実行するときに、cfabort タグを呼び出します。 • no
category	オプション		トレースグループを識別するための、ユーザー定義の文字列です。
inline	オプション	no	• yes: デバッグ情報の出力に加えて、ページの cftrace タグの位置にトレースコードをインラインで表示します。 • no

属性	必須 / オプション	デフォルト	説明
text	オプション		ユーザー定義の文字列です。単純変数は使用できますが、配列などの複合変数は使用できません。cflog の text 属性に出力します。
type	オプション	Information	cflog の type 属性に相当します。適切なアイコンを表示します。 • Information • Warning • Error • Fatal Information
var	オプション		表示する単純変数または複合変数の名前です。 一時的な値、または、CFM ページ上に表示されない値の表示に役立ちます。

使用方法

このタグ内にはアプリケーションコードを指定できません。これは、デバッグを無効にした場合に問題が発生するのを回避するためです。

このタグは、アプリケーション開発時の CFML コードのデバッグに役立ちます。

cftrace タグの出力は、次の方法で表示できます。

- デバッグ出力内のセクションとして表示します。
- アプリケーションページにおいてインラインで、デバッグ出力内のセクションとして表示します。インライントレースを指定すると、ColdFusion により cftrace タグまでの出力がすべて消去され、タグ検出時にトレース出力が表示されます。

  ログファイルのエントリの例は、次のとおりです。

```
 "Information","web-4","04/08/02","23:21:30", ,"[30 ms (1st trace)]
 [C:\CFusion\wwwroot\generic.cfm @ line: 9] -
     [thisPage = /generic.cfm]"
 "Information","web-0","04/08/02","23:58:58", ,"[5187 ms (10)]
 [C:\CFusion\wwwroot\generic.cfm @ line: 14] - [category]
     [thisPage = /generic.cfm] [ABORTED] thisPage "
```

  複合変数の場合、ColdFusion では変数名およびオブジェクト内の要素数がリスト表示されます。変数の内容はロギングされません。

例

次の例では、cfif ブロックで評価を行う FORM 変数をトレースします。

```
<cftrace var="FORM.variable"
    text="doing equivalency check for FORM.variable"
    category="form_vars"
    inline="true">
<cfif isDefined("FORM.variable") AND #FORM.variable# EQ 1>
    <h1>Congratulations, you're a winner!</h1>
<cfelse>
    <h1>Sorry, you lost!</h1>
</cfif>
```

cftransaction

説明

トランザクション処理をサポートするエンタープライズデータベース管理システムの場合、そのデータベース管理システムに対し、複数のデータベースオペレーションを単一のトランザクションとして処理するように指示します。データベースのコミットおよびロールバック処理を実行できます。使用するデータベース管理システムが SQL トランザクション処理をサポートするかどうかについては、そのシステムのマニュアルを参照してください。

カテゴリ

データベース操作タグ

シンタックス

```
<cftransaction
    action = "begin|commit|rollback|setsavepoint"
    isolation = "read_uncommitted|read_committed|repeatable_read"
    savepoint = "savepoint name">
</cftransaction>
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfinsert、cfprocparam、cfprocresult、cfquery、cfqueryparam、cfstoredproc、cfupdate、『ColdFusion アプリケーションの開発』の Commits, rollbacks, and transactions および Tags as functions and operators

履歴

ColdFusion 8: setsavepoint 値が action 属性に追加されました。savepoint 属性が追加されました。

属性

属性	必須 / オプション	デフォルト	説明
action	オプション	begin	• begin: 実行するコードのブロックの先頭です。 • commit: 未処理のトランザクションをコミットします。 • rollback: 未処理のトランザクションをロールバックします。 • setsavepoint: トランザクション内の特定の状態を保存します。
isolation	オプション		複数の SQL トランザクションを同時に実行する場合の読み取り方法を指定する分離レベルです。読み取りアクションには、ダーティリード、反復不可能読み取り、ファントムという 3 つの分離レベルがあります。ダーティリードにおいては、最初の SQL トランザクションが COMMIT を実行する前に、次の SQL トランザクションが行を読み取ることができます。反復不可能読み取りにおいては、最初の SQL トランザクションが行を読み取った後に、次の SQL トランザクションがその行を変更または削除して COMMIT を実行します。ファントムにおいては、最初の SQL トランザクションが検索条件に一致する行を読み取り、次の SQL トランザクションが最初のトランザクションの検索条件に一致する行を少なくとも 1 つ生成した後、最初のトランザクションが検索を繰り返すので、異なる結果セットが返されます。 • read_uncommitted: ダーティリード、反復不可能読み取り、およびファントムが許可されます。 • read_committed: 反復不可能読み取りとファントムが許可されます。ダーティリードは許可されません。 • repeatable_read: ファントムが許可されます。ダーティリードと反復不可能読み取りは許可されません。 • serializable: ダーティリード、反復不可能読み取り、およびファントムは許可されません。
savepoint	オプション		トランザクション内のセーブポイントの名前です。セーブポイントを設定すると、トランザクションの一部をロールバックできます。たとえば、トランザクションに挿入、更新、削除のアクションが含まれていて、更新の後にセーブポイントを設定した場合は、削除を実行する前の状態までトランザクションをロールバックできます。
nested	オプション	true	この属性は、cftransaction タグを別の cftransaction タグ内にネストできるかどうかを指定します。この属性値が false の場合、親の cftransaction タグが存在するとエラーが発生します。

使用方法

action 属性の値を指定しない場合、自動トランザクション処理は次のように進行します。

- トランザクションブロック内の cfquery オペレーションがエラーなしで完了した場合、このトランザクションはコミットされます。
- cfquery タグが cftransaction ブロック内でエラーを生成した場合、このトランザクション内のすべての cfquery オペレーションがロールバックされます。

  isolation 属性の値を指定しない場合、ColdFusion は関連付けられたデータベースに対してデフォルトの分離レベルを使用します。

  ただし、CFML エラー処理と action 属性を使用すると、データベースクエリーの成否に基づいて、トランザクションをコミットするかロールバックするかを明示的に制御できます。トランザクションブロック内では、次の操作が可能です。

- ブロック内に <cftransaction action = "commit"/> タグをネストすることによって、データベーストランザクションをコミットします。
- ブロック内に <cftransaction action = "rollback"/> タグをネストすることによって、トランザクションをロールバックします。

  これらの例では、終了タグに代わりにスラッシュを使用しています。

1 つのトランザクションブロック内に、複数のデータベースへのクエリーを記述することができます。ただし、別のデータベースへのクエリーを記述する前に、1 つのデータベースへのトランザクションをコミットするか、ロールバックする必要があります。

データベースエンジンによるトランザクション中のロックの方法を制御するには、isolation 属性を使用します。

cftransaction タグ内で cfthread タグを使用してクエリー呼び出しを行った場合、cftransaction タグは期待どおりに動作しません。

- cftransaction タグをネストできるようになりました。通常、ColdFusion 9 ではトランザクションのネストがサポートされていませんが、1 つの cftransaction タグを別のタグ内に埋め込むことができます。これらのタグをネストした場合は、最も外側の cftransaction タグのみが有効になります。

  この機能を使用すると、cftransaction タグ内にあるコードによって関数が呼び出されるかどうかを検討することなく、トランザクション内で実行する必要がある関数を記述できます。関数内で cftransaction タグを使用します。呼び出し元のコードがトランザクション内にある場合、このタグに効果はありません。呼び出し元のコードがトランザクション内にない場合、このタグによりトランザクションが開始されます。

  次のコードは、ネストされたトランザクションタグを示しています。

```
<cftransaction>
<cfquery name="iquery" datasource="dsn">
insert into region(regionid, regiondescription) values('111', 'YPR')
</cfquery>
<cftransaction>
<cfquery name="iquery" datasource="dsn">
update region set regiondescription = 'new' where regionid='111'
</cfquery>
</cftransaction>
</cftransaction>
```

  注意：実際の状況では、2 つ目の cftransaction および cfquery を CFC に記述できます。1 つ目の cftransaction および cfquery が regionid 値を渡すことによって、これらを順番に呼び出します。

例

```
 <p>The cftransaction tag can be used to group multiple queries that use
the cfquery tag into one business event. Changes to data that is requested
by the queries are not committed to the datasource until all actions within
the transaction block have executed successfully.
<p>This a view-only example.
<!---
<cftransaction>
    <cfquery name='makeNewCourse' datasource='Snippets'>
    INSERT INTO Courses
        (Number, Descript)
    VALUES
        ('#myNumber#', '#myDescription#')
    </cfquery>

    <cfquery name='insertNewCourseToList' datasource='Snippets'>
    INSERT INTO CourseList
        (CorNumber, CorDesc, Dept_ID,
        CorName, CorLevel, LastUpdate)
    VALUES
        ('#myNumber#', '#myDescription#', '#myDepartment#',
        '#myDescription#', '#myCorLevel#', #Now()#)
    </cfquery>
</cftransaction>
--->
```

セーブポイントは、トランザクションの挿入、更新、および削除のアクションが完了した時点で設定できます。その場合は、エラー処理ロジックを使用して、前のセーブポイントにロールバックする必要があるかどうかを判別できます。

例

```
<!--- This example performs batch processing of withdrawals --->
<!--- from a bank account. The withdrawal amounts are stored --->
<!--- in an array. --->
<!--- There is a CFC named bank.cfc whose contains appear --->
<!--- after the example. --->

<cftransaction>
    <!--- Get the account balance. --->
    <cfinvoke component="bank" method="getBalance"
        returnvariable="getacctbalance" accountnum=1>

<cfloop index="withdrawnum" from="1" to="#ArrayLen(withdrawals)#">
    <!--- Set a savepoint before making the withdrawal. --->
    <cfset noxfer = "point" & #withdrawnum#>
    <cftransaction action = "setsavepoint" savepoint = "#noxfer#"/>

    <!--- Make the withdrawal. --->
    <cfinvoke component="bank" method="makewithdrawal"
        returnvariable="getacctbalance" accountnum=1
        withdrawamount="#withdrawals[withdrawnum]#">

    <!--- Get the account balance. --->
    <cfinvoke component="bank" method="getBalance"
        returnvariable="getacctbalance" accountnum=1>

    <!--- If the balance is negative, roll back the transaction. --->
    <cfif getacctbalance.balance lt 0>
        <cftransaction action="rollback" savepoint="#noxfer#" />
    </cfif>
    </cfloop>
</cftransaction>

<!--- The bank.cfc contains the following:

cfcomponent>
    <cffunction name="getBalance" access="public" returntype="query">
        <cfargument name="accountnum" type="numeric" required="yes">
            <cfquery name="getacctbalance" datasource="testsqlserver">
                SELECT * FROM dbo.mybank
                WHERE accountid = #accountnum#
            </cfquery>
        <cfreturn getacctbalance>
    </cffunction>

    <cffunction name="makewithdrawal" access="public" returntype="query">
        <cfargument name="accountnum" type="numeric" required="yes">
        <cfargument name="withdrawamount" type="numeric" required="yes">
            <cfquery name="withdrawfromacct" datasource="testsqlserver">
                UPDATE dbo.mybank SET balance = balance - #withdrawamount#
                WHERE accountid = 1
            </cfquery>
            <cfinvoke method="getBalance" returnvariable="getacctbalance"
        accountnum=1>
        <cfreturn getacctbalance>
    </cffunction>
</cfcomponent>
--->
```

cftree

説明

フォーム内にツリーコントロールを挿入します。また、ユーザーの選択を検証します。cfform タグブロック内で使用します。ツリーにデータを渡すには、ColdFusion クエリーを使用します。

カテゴリ

フォームタグ

シンタックス

```
<cftree
    name="name"
    align="top|left|bottom|baseline|texttop|absbottom|
        middle|absmiddle|right"
    appendKey="yes|no"
    bold="yes|no"
    border="yes|no"
    cache="yes|no"
    completePath="yes|no"
    delimiter="delimiter"
    enabled="yes|no"
    font="font"
    fontSize="size"
    format="applet|flash|html|object|xml"
    height="integer"
    highlightHref="yes|no"
    hScroll="yes|no"
    hSpace="integer"
    italic="yes|no"
    lookAndFeel="motif|windows|metal"
    message="text"
    notSupported="text">
    onBlur="ActionScript to invoke"
    onChange="ActionScript to invoke"
    onError="text"
    onFocus="Actionscript to invoke"
    onValidate="script name"
    required="yes|no"
    style= "style specification"
    tooltip="text"
    visible="yes|no"
    vScroll="yes|no"
    vSpace="integer"
    width="integer">
</cftree>
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfajaximport、cfapplet、cfcalendar、cfform、cfformgroup、cfformitem、cfgrid、cfinput、cfselect、cfslider、cftextarea、cftreeitem、『ColdFusion アプリケーションの開発』の Working with action pages、Building tree controls with the cftree tag、および Using HTML trees

履歴

ColdFusion 8: Ajax ベースの HTML ツリーがサポートされるようになりました (cache 属性、format 属性の html 値など)。

ColdFusion MX 7.01: onBlur イベントと onFocus イベントのサポートが追加されました。

ColdFusion MX 7:

- format 属性が追加され、Flash、XML、およびオブジェクト出力を生成できるようになりました。
- enabled、onChange、style、tooltip、および visible の各属性が追加されました (Flash 形式の場合のみ)。

  ColdFusion MX: 動作が変更されました。treeitem が含まれているかどうかにかかわらず、ツリーコントロールが表示されるようになりました。

属性

注意 : XML 形式では、ColdFusion はすべての属性を XML に渡します。用意された XSLT スキンでは XML 形式のツリーは処理または表示されませんが、アプレットおよび Flash 形式のツリーは表示されます。

属性	**必須 / オプション** **形式**	**デフォルト**	**説明**
name	必須、 すべて		ツリーコントロールの名前です。
align	オプション、 アプレット、オブジェクト		• top • left • bottom • baseline • texttop • absbottom • middle • absmiddle • right
appendKey	オプション、 すべて	yes	• yes: cftreeitem の href 属性が指定されている場合、ColdFusion は、選択されたツリー項目の値を持つ CFTREEITEMKEY クエリー文字列変数を cfform アクション URL に追加します。 • no: ツリー項目の値を URL に追加しません。
bold	オプション、 アプレット、Flash、HTML	no	• yes: ツリーコントロールのテキストがボールドで表示されます。 • no
border	オプション、 アプレット、オブジェクト	yes	• yes: ツリーコントロールの周囲にボーダーを表示します。 • no

属性	必須 / オプション 形式	デフォルト	説明
cache	オプション、 HTML	yes	ツリーの treeitem 子タグでバインド式が使用されている場合にのみ適用されます。 ユーザーがツリーノードを展開するたびに新しいデータを取得するかどうかを指定するブール値です。 • yes: ノードが最初に展開されたときにノードの子アイテムを一度だけ取得します。 • no: ノードが展開されるたびに子アイテムを取得します。
completePath	オプション、 アプレット、HTML、オブジェクト	no	• yes: Form.**treename**.path 変数の値が、cftree が送信されたときのツリーパスのルートから始まります。 • no: Form.**treename**.path 変数でルートレベルを省略します。この場合の値は、ツリー内の最初の子ノードから始まります。 cfform の preserveData 属性がツリーに対して機能するためには、この属性を yes に設定します。 クエリーによって挿入されるツリー項目については、cftreeitemqueryasroot 属性を使用してルート名を指定した場合、その値が返されます。ルート名を指定しない場合は、クエリー名が返されます。
delimiter	オプション、 すべて	¥¥	アクションページの Forms.**treename**.path 変数内で要素を区切る文字です。
enabled	オプション、 Flash	yes	Flash 形式の場合のみ : コントロールを有効にするかどうかを指定するブール値です。無効なコントロールはライトグレーで表示されます。
font	オプション、 アプレット、HTML		ツリーコントロール内のテキストのフォント名です。
fontSize	オプション、 アプレット、Flash、HTML		ツリーコントロール内のテキストのフォントサイズです (単位 : ピクセル)。
format	オプション、 すべて	applet	• applet: ブラウザで Java アプレットを使用してツリーを表示します。 • flash: Flash コントロールを使用してツリーを表示します。 • html: Ajax ベースの HTML を使用してツリーを表示します。 • object: name 属性で指定された名前を持つ ColdFusion 構造体としてツリーを返します。構造体の内容の詳細については、「オブジェクト形式」を参照してください。 • xml: ツリーの XML 表現を生成します。XML 形式フォームでは、生成された XML がフォーム内に含まれます。HTML 形式フォームでは、name 属性で指定された名前を持つ文字列変数に XML が格納されます。
height	オプション、 アプレット、Flash	320(applet の場合のみ)	ツリーコントロールの高さです (単位 : ピクセル)。Flash 形式でこの属性を省略した場合、ツリーは自動的にサイズ設定されます。
highlightHref	オプション、 アプレット、オブジェクト	yes	• yes: href 属性が指定された cftreeitem タグによって表示される値をリンクとして強調表示します。 • no: 強調表示は無効になります。

属性	必須 / オプション 形式	デフォルト	説明
hScroll	オプション、 アプレット、オブジェクト	yes	• yes: 水平方向にスクロールできます。 • no
hSpace	オプション、 アプレット		ツリーコントロールの左右に取る水平方向の余白です (単位 : ピクセル)。
italic	オプション、 アプレット、Flash、HTML	no	• yes: ツリーコントロールのテキストがイタリックで表示されます。 • no
label	オプション、 HTML、		
lookAndFeel	オプション、 アプレット、オブジェクト	windows	• motif: Motif スタイルでツリーを表示します。 • windows: Windows スタイルでツリーを表示します。 • metal: Java Swing スタイルでツリーを表示します。 指定したスタイルオプションがプラットフォームでサポートされていない場合は、プラットフォームのデフォルトのスタイルが使用されます。
message	オプション、 アプレット、HTML		検証に失敗した場合に表示されるメッセージです。
notSupported	オプション、 アプレット	「説明」を参照	Java アプレットベースの cfform コントロールが含まれているページを、Java がサポートされていない (または Java のサポートが無効になっている) ブラウザで開いた場合に表示するテキストです。例 : "ColdFusion Java アプレットを表示するには、ブラウザで Java がサポートされていなければなりません。" デフォルトのメッセージ : ColdFusion Java アプレットを表示するには、 ブラウザが Java をサポートしている必要があります。
onBlur	オプション、 Flash		ツリーがフォーカスを失ったときに実行される ActionScript です。
onChange	オプション、 Flash		ユーザーのアクションに応じてコントロールに変化があったときに実行される ActionScript です。 onChange イベントハンドラを指定した場合、ColdFusion アクションページの Form スコープでは選択された項目に関する情報は自動的に取得されません。ActionScript の onChange イベントハンドラで、すべての変更と選択を処理する必要があります。
onError	オプション、 アプレット、HTML		検証に失敗した場合に実行する JavaScript 関数です。
onFocus	オプション、 Flash		ツリーがフォーカスを取得したときに実行される ActionScript です。JavaScript の DOM フォームオブジェクト、name 属性の値、検証でエラーが発生した値、および message 属性で指定されているエラーテキストがメソッドに渡されます。

属性	必須 / オプション 形式	デフォルト	説明
onValidate	オプション、 アプレット、HTML		ユーザー入力を検証するための JavaScript 関数です。JavaScript DOM フォームオブジェクト、入力オブジェクト、および入力オブジェクト値が、指定されているルーチンに渡されます。検証に成功すると true が返されます。検証に失敗すると false が返されます。
required	オプション、 アプレット、Flash、HTML	no	• yes: ユーザーはツリーコントロール内の項目を選択する必要があります。 • no
style	オプション、 Flash、HTML		CSS 形式のスタイル指定でなければなりません。HTML 形式の場合、この属性は HTML の style 属性の値に対応します。Flash 形式の場合は、対応する Flash 要素に対して、Flex で使用するものと同じシンタックスとコンテンツを使用します。
tooltip	オプション、 Flash		Flash 形式の場合のみ : マウスポインタをコントロールの上に置いたときに表示されるテキストです。
value	オプション、 HTML、		
visible	オプション、 Flash	yes	Flash 形式の場合のみ : コントロールを表示するかどうかを指定するブール値です。表示されないコントロールが使用するスペースは空白です。
vScroll	オプション、 アプレット、オブジェクト	yes	• yes: 垂直方向にスクロールできます。 • no
vSpace	オプション、 アプレット		ツリーコントロールの上下に取る垂直方向の余白です (単位 : ピクセル)。
width	オプション、 アプレット、Flash	200 (applet の場合のみ)	ツリーコントロールの幅です (単位 : ピクセル)。Flash 形式でこの属性を省略した場合、ツリーは自動的にサイズ設定されます。

注意 : すべての属性は XML 形式で生成された XML に渡されますが、cftree XML を解釈する ColdFusion スキンはありません。

使用方法

このタグは、cfform タグブロックの内部で使用する必要があります。

アプレット形式のツリーでは、クライアントが Java アプレットをダウンロードする必要があります。また、クライアントに最新の Java プラグインがインストールされていない場合、アプレット形式のツリーを表示するために最新の Java プラグインをダウンロードしなければならないこともあります。Flash 形式のツリーでは Flash コントロールを使用します。このツリーは、HTML 形式の cfform タグに埋め込むことができます。このタグを Flash 形式、HTML 形式、またはアプレット形式で正しく動作させるには、JavaScript 対応のブラウザを使用する必要があります。

注意 : HTML 形式フォームでこのタグを使用して Flash 形式を指定し、height 属性と width 属性を指定しない場合、Flash 形式での表示は画面上の表示可能領域を超えるサイズになります。ツリーの後に他の出力 (フォームコントロールなど) が続く場合、それを表示するにはスクロールする必要があります。したがって、HTML フォームで Flash ツリーの後に他の出力を続ける場合は、height と width の値を指定してください。

次の条件を満たす場合、このタグのオプションとして挿入されるユーザーが選択したクエリーデータからの値は、ユーザーがフォームを送信した後も継続して表示されます。

- cfform の preserveData 属性が "yes" に設定されている場合
- cfform の action 属性がフォーム自身と同じページ (デフォルト) に送信される場合、または、ユーザー入力フォーム上のコントロールと同じ名前のコントロールを持つフォームがアクションページ上にある場合

  詳細については、cfform タグのエントリを参照してください。

フォーム変数

ツリー項目を選択し、ツリーを含むフォームを送信すると、アクションページの Form スコープで 2 つの変数を持つ構造体が作成されます。構造体名はツリー名です。次の表に、フィールドを示します。

フィールド	値
path	選択したノードに至るまでのツリー上のパスです。[< ルート >¥]< ノード 1>¥< ノード 2>¥... という形式を取ります。アプレット形式では、completePath 属性が true の場合に限り、このパスにはルートノードが含まれます。Flash 形式では、このパスには常にルートノードが含まれます。
node	選択したツリーノードの値です。

オブジェクト形式

format 属性で object を指定した場合、ツリーは ColdFusion 構造体として返され、ブラウザには送信されません。たとえば、構造体のループによるメニューの挿入、ページを移動するための "breadcrumb" リンクの生成、DHTML ツリーの作成などが可能です。

注意：XML 形式のフォームでオブジェクト形式のツリーを指定しても、ColdFusion はこのツリーを生成しません。

構造体変数名は、cftreename 属性で指定します。構造体の最上位レベルには、次の 2 種類のエントリがあります。

- 属性設定
- 子配列

属性設定

構造体には、次の cftree 属性の値を持つ最上位レベルのエントリがあります。

align	completePath	highlightHref	lookAndFeel
appendKey	delimiter	hScroll	name
bold	fontWeight	italic	vscroll
border			

子配列

最上位レベルの子エントリは、項目エントリの配列です。各項目には次のエントリがあります。

フィールド	値
children	この項目の子項目であり、項目の構造体の配列です。
display	cftreeitem の display 属性で定義された、ツリー項目のラベルです。
expand	項目を展開して子を表示するかどうかを示します。cftreeitemexpand 属性の値で指定します。
href	ユーザーが項目を選択したときのリンク先の URL です。cftreeitemhref 属性の値で指定します。

フィールド	値
img	ツリー項目のアイコンとして表示する、ツリーのアイコンイメージです。cftreeitemimg 属性の値で指定します。img 属性を使用してカスタムアイコンを表示できるのはアプレットバージョンの場合のみです。Flash バージョンでは表示できません。
imgOpen	ツリー項目が開かれた (展開された) ときに表示するイメージです。cftreeitemimgopen 属性の値で指定します。
parent	ツリーにおけるこの項目の親項目の値です。
path	ツリールートから現在の要素へのノードパスです。
queryAsRoot	クエリーがこの項目のルートであるかどうかを示します。cftreeitemqueryAsRoot 属性の値で指定します。
target	_blank などのリンクターゲットです。項目の cftreeitemtarget 属性の値で指定します。
value	cftreeitemvalue 属性で指定された、項目の値です。

例

次の例では、cfdocexamples データベースの CourseList テーブルの中から利用可能なコースを表示するツリーを作成し、部門ごとのコースをフォルダに配置します。この例は Flash で表示され、Departments リストを使用して部門名を取得します。

```
<cfquery name="getCourses" datasource="cfdocexamples">
    SELECT d.dept_name, c.course_id, c.CorName, c.CorLevel, c.corName ||' ( ' ||c.corLevel ||' )' AS
corLabel
    FROM CourseList c, Departments d
    WHERE d.Dept_ID = c.Dept_ID
    ORDER BY d.dept_Name, c.corName, c.corLevel
</cfquery>

<cfform name="studentForm" format="flash" width="400" height="450">
    <cftree name="courseTree" width="350" height="400">
        <cftreeitem
            query="getCourses"
            value="dept_name,Course_id"
            display="dept_name,CorLabel" queryasroot="NO" expand="yes,no">
    </cftree>
</cfform>
```

次の例では、部門別に構成された、組織内の全従業員に関する基本情報を示すツリーを作成します。部門を展開すると、全従業員が表示されます。プラス記号 (+) をクリックすると、追加情報が表示されます。従業員名をクリックすると、同じページに戻り、選択対象の Path およびノードの値が表示されます。

```
<!--- Query the datasource to get employee information. --->
<!--- Group the output by Department.
    (All fields are required in Group By clause.) --->
<cfquery name = "GetEmployees" dataSource = "cfdocexamples">
    SELECT Emp_ID, FirstName, LastName, EMail, Phone, Department
    FROM Employees
    GROUP BY Department, Emp_ID, FirstName, LastName, EMail, Phone
</cfquery>
<html>
<body>
<h3>cftree Example</h3>

<!--- The following runs if the user clicked on a link in the tree.
    A complete application would use the ID for additional processing. --->

<cfif isdefined("Form.fieldnames")>
<b>Selected item information</b><br>
<cfoutput>
<b>Path: </b>#form.Employees.Path#<br>
<b>node: </b>#form.Employees.node#<br>
<br>
</cfoutput>
</cfif>

<!--- Display the tree. The cftree tag must be in a cfform. --->
<cfform action="#cgi.script_name#" preservedata="Yes" format="Flash">
    <cftree name = "Employees" height = "400" width = "400"
        font = "Arial Narrow" italic="yes" highlighthref="No" HScroll="no" VScroll="no"
        completepath="no" lookandfeel="windows" border="No" required="yes">
        <!--- cfoutput tag with a group attribute loops over the departments. --->
        <cfoutput group="Department" query = "GetEmployees">
            <cftreeitem value="#Department#" parent="Employees" expand="yes">
            <!--- This cfoutput tag loops over the records for the department.
                    The cfoutput tag does not need any attributes. --->
            <cfoutput>
                <!--- Create an item for each employee in the department.
                    Do not expand children. Each employee name links to this page,
                    and sends the employee ID in the query string.--->
                <cftreeitem value = "#LastName#, #FirstName#"
                    parent = "#Department#" expand="false" img="cd"
                    href="#cgi.script_name#?user_id=#emp_id#">
                <!--- Each Employee entry has Id, and contact info children. --->
                <cftreeitem value = "#Emp_ID#" display = "Employee ID: #Emp_ID#"
                    parent = "#LastName#, #FirstName#" img="remote">
                <!--- Each node must be unique value, so use Emp_ID om value. --->
                <cftreeitem value = "#Emp_ID#_ContactInfo" img="computer"
                    display = "Contact Information"
                    parent = "#LastName#, #FirstName#" expand = "false">
                <!--- ContacInfo has two children --->
                <cftreeitem value = "#Phone#" parent = "#Emp_ID#_ContactInfo">
                <cftreeitem value = "#Email#" parent = "#Emp_ID#_ContactInfo">
            </cfoutput>
        </cfoutput>
    </cftree>
    <cfinput type="Submit" name="submitit" value="Submit" width="100">
</cfform>
```

cftreeitem

説明
cftree タグによって作成されたフォームツリーコントロールに要素を挿入します。

カテゴリ
フォームタグ

シンタックス
```
<cftreeitem
    value = "text"
    bind = "bind expression"
    display = "text"
    expand = "yes|no"
    href = "URL"
    img = "filename"
    imgopen = "filename"
    parent = "parent name"
    query = "queryname"
    queryAsRoot = "yes|no"
    target = "URL target">

OR

<cftreeitem
    bind = "bind expression">
    onBindError = "JavaScript function name"
```

注意： このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

履歴
ColdFusion 8: bind 属性と onBindError 属性が追加されました。

関連項目
cfapplet、cfform、cfformgroup、cfformitem、cfgrid、cfinput、cfselect、cfslider、cftextarea、cftree、『ColdFusion アプリケーションの開発』の Building tree controls with the cftree tag および Using HTML trees

属性
注意： XML 形式では、ColdFusion はすべての属性を XML に渡します。用意された XSLT スキンでは XML 形式のツリーは処理または表示されませんが、アプレットおよび Flash 形式のツリーは表示されます。

属性	必須 / オプション、 形式	デフォルト	説明
value	アプレット、Flash、XML の場合は必須。 HTML の場合は、value または bind が必須。		ツリーを含むフォームが送信されたときに渡される値です。cfquery からのデータをツリーに挿入する場合は、使用する列を区切りリストで複数指定できます。たとえば、value="dept_id,emp_id" のように指定します。この場合、各列では、リスト内でその列の前にある列の子となる項目が生成されます。
bind	HTML の場合は、value または bind が必須		すべてのツリーノードをダイナミックに取得する CFC 関数、JavaScript 関数、または URL を指定するバインド式です。この属性はツリーの最上位レベルでのみ使用できます。この属性を使用する場合、ツリーに含めることができる要素は cftreeitem タグのみです。 bind 属性とともに使用できるその他の属性は、onBindError のみです。 バインディングを使用するツリーの作成方法については、『ColdFusion アプリケーションの開発』の Using HTML trees を参照してください。
display	オプション、 すべて	value	ツリー項目のラベルです。クエリーからのデータをツリーに挿入する場合は、名前を区切りリストで指定します。例 : display = "dept_name,emp_name"
expand	オプション、 すべて	yes	• yes: ツリーが展開されてツリー項目の子項目が表示されます。 • no: ツリー項目が閉じた状態で表示されます。
href	オプション、 すべて		ユーザーがツリー項目をクリックしたときのリンク先の URL です。query 属性を使用する場合は、URL を含むクエリー列を href 属性で指定できます。href がクエリー列でない場合、属性のテキストは URL または URL のリストでなければなりません。 クエリーからのデータをツリーに挿入する場合は、カンマ区切りリストで URL を指定します。例 : href = "http://dept_svr,http://emp_svr"

属性	必須 / オプション、 形式	デフォルト	説明
img	オプション、 アプレット、HTML、オブジェクト	folder	ツリー項目のアイコンの、イメージ名、ファイル名、またはファイルの URL です。 次の値があります。 • cd • computer • document • element • folder • floppy • fixed • remote カスタムイメージを指定することもできます。パスおよびファイル拡張子を含めて指定します。例 : img = "../images/page1.gif" Web ルートを基準にした相対パスを指定することもできます。 カスタムイメージは、Flash 形式の場合はサポートされません。 1 つのツリー内で複数のイメージを指定する場合、または 2 番目以降のレベルのイメージを指定する場合は、カンマを使用して名前を区切ります (項目の順番は各レベルに対応します)。例 : img="folder,document" img = ",document" (2 番目のレベルのイメージを指定する場合)
imgopen	オプション、 アプレット、HTML、オブジェクト		img 属性の説明にある、開かれたツリー項目で表示されるアイコンです。
onBindError	オプション、 HTML	「説明」を参照	バインド式の評価でエラーが発生した場合に実行する JavaScript 関数の名前です。この関数は 2 つの属性 (HTTP ステータスコードとメッセージ) を取る必要があります。 この属性を省略し、(ColdFusion.setGlobalErrorHandler 関数を使用して) グローバルエラーハンドラを指定すると、エラーメッセージが表示されます。それ以外の場合は、デフォルトのエラーポップアップが表示されます。
parent	オプション、 すべて		ツリー項目の親の値です。この属性により、ツリー階層におけるその項目の位置が決定されます。省略した場合、その項目はツリーのルートレベルに配置されます。または、queryAsRoot 属性が true の場合、クエリーの直下に配置されます。

属性	必須 / オプション、 形式	デフォルト	説明
query	オプション、 すべて		ツリー項目にデータを挿入するために使用するクエリーの名前です。value 属性で指定されたクエリー列のリストに含まれるフィールド値ごとに項目が生成されます。各行のフィールドは、最初の列に階層形式でリンクされます。
queryAsRoot	オプション、 すべて	yes	query 属性を指定した場合に限り適用されます。このタグによって生成されるすべての項目のルートレベルとしてクエリーを使用します。この属性を使用することにより、親 cftreeitem を作成しなくても済むようになります。 • yes: このタグによって生成される他のすべての項目の親 (ルート) 項目を生成し、クエリー名を親項目の名前として使用します。parent 属性が指定されている場合は、指定されている親の子がルート項目になります。 • no: このタグによって生成されるすべての項目の直接の親として parent 属性で指定された項目を使用します。parent 属性がない場合は、クエリーを親として使用します。 • 任意の文字列 : ルート項目を作成し、指定された文字列を項目名として使用します。parent 属性が指定されている場合は、指定されている親の子がルート項目になります。
target	オプション、 すべて		href URL のターゲット属性です。クエリーからのデータをツリーに挿入する場合は、カンマ区切りリストでターゲットを指定します。例 : target = "FRAME_BODY,_blank"

使用方法

このタグを正しく動作させるには、JavaScript 対応のブラウザを使用する必要があります。このタグは、cftree タグの子である必要があります。

cftreeitem タグには、次の 3 つの基本的な形式があります。

- query 属性または bind 属性を使用してこのタグにデータを挿入しない場合は、ツリー項目が 1 つだけ作成されます。
- query を使用する場合は、複数の項目が作成されます。クエリーの各行では、階層的にネストされた項目が各列に 1 つずつ作成されます。
- bind 属性を使用する場合は、クライアントサイドのツリーがツリー項目の直下にある子のデータをダイナミックに取得し、ツリー項目が展開されたときに子項目を作成します。bind 属性を使用して HTML 形式のツリーにデータを渡す方法の詳細については、『ColdFusion アプリケーションの開発』の Using HTML trees を参照してください。

例

次の例では、単一の cftreeitem タグおよびクエリーを使用してシンプルなツリーを作成します。

```
<cfform action = "#cgi.script_name#">
    <cftree name = "Employees" height = "400" width = "200">
        <cftreeitem value="LastName, FirstName, Emp_ID" query="getEmployees"
         queryAsRoot="False">
    </cftree>
</cfform>
```

次の例では、部門別に構成された、組織内の全従業員に関する基本情報を示すツリーを作成します。部門を展開すると、全従業員が表示されます。プラス記号 (+) をクリックすると、追加情報が表示されます。従業員名をクリックすると、同じページに戻り、選択した従業員の ID が表示されます。

```
<!--- Query the datasource to get employee information.--->
<!--- Group the output by Department.
    (All fields are required in Group By clause.) --->
<cfquery name = "GetEmployees" dataSource = "cfdocexamples">
    SELECT Emp_ID, FirstName, LastName, EMail, Phone, Department
    FROM Employees
    GROUP BY Department, Emp_ID, FirstName, LastName, EMail, Phone
</cfquery>
<html>
<body>
<h3>cftreeitem Example</h3>

<!--- The following runs if the user clicked on a link in the tree.
    A complete application would use the ID for additional processing. --->
<cfif isdefined("URL.user_ID")>
    <cfoutput>
        <!--- URL.cftreeitemkey is the selected tree item's value attribute. --->
        You Requested information on #URL.cftreeitemKey#; User ID #URL.user_ID#
    </cfoutput>
    <br><br>
</cfif>
<!--- Display the tree. The cftree tag must be in a cfform. --->
<cfform>
    <cftree name = "Employees" height = "400" width = "200"
        font = "Arial Narrow" highlighthref="No" hscroll="No">
        <!--- cfoutput tag with a group attribute loops over the departments. --->
        <cfoutput group="Department" query = "GetEmployees">
            <cftreeitem value="#Department#" parent="Employees" expand="yes">
            <!--- This cfoutput tag loops over the records for the department.
                    The cfoutput tag does not need any attributes. --->
            <cfoutput>
                <!--- Create an item for each employee in the department.
                    Do not expand children. Each employee name links to this page,
                    and sends the employee ID in the query string.--->
                <cftreeitem value = "#LastName#, #FirstName#"
                    display = "#LastName#, #FirstName#"
                    parent = "#Department#" expand="no"
                    href="#cgi.script_name#?user_id=#emp_id#">
                <!--- Each Employee entry has ID and ContactInfo children. --->
                <cftreeitem value = "#Emp_ID#" display = "Employee ID: #Emp_ID#"
                    parent = "#LastName#, #FirstName#">
                <!--- Each node must be unique value, so use Emp_ID om value. --->
                <cftreeitem value = "#Emp_ID#_ContactInfo"
                    display = "Contact Information"
                    parent = "#LastName#, #FirstName#" expand = "No">
                <!--- ContactInfo has two children. --->
                <cftreeitem value = "#Phone#" parent = "#Emp_ID#_ContactInfo">
                <cftreeitem value = "#Email#" parent = "#Emp_ID#_ContactInfo">
            </cfoutput>
        </cfoutput>
    </cftree>
</cfform>
```

cftry

説明

cfcatch タグとともに使用します。これらを併用すると、ColdFusion ページで発生する例外の検出や処理を実行できます。例外とは、データベースオペレーションの失敗、インクルードファイルの欠如、開発者によって指定されたイベントなど、ColdFusion ページ内での正常な命令の流れを妨げるイベントです。

カテゴリ

例外処理タグ

シンタックス

```
<cftry>
    Code that might throw an exception
    One or more cfcatch blocks
</cftry>
```

関連項目

cfcatch、cffinally、cferror、cfrethrow、cfthrow、onError、『ColdFusion アプリケーションの開発』の Handling Errors

履歴

ColdFusion MX: cfscript が変更され、cftry および cfcatch タグに相当する try および catch ステートメントが含まれるようになりました。

使用方法

cftry ブロック内に、例外を返す可能性があるコードを配置します。その後ろに、例外を検出して処理する cfcatch タグを配置します。このタグには終了タグが必要です。

例

```
<!--- cftry example, using TagContext to display the tag stack. --->
<h3>cftry Example</h3>
<!--- Open a cftry block. --->
<cftry>
    <!--- Note misspelled tablename "employees" as "employeeas". --->
    <cfquery name = "TestQuery" dataSource = "cfdocexamples">
        SELECT *
        FROM EMPLOYEES
    </cfquery>

    <!--- <p>... other processing goes here --->
    <!--- specify the type of error for which we search --->
    <cfcatch type = "Database">
    <!--- the message to display --->
        <h3>You've Thrown a Database <b>Error</b></h3>
        <cfoutput>
            <!--- and the diagnostic message from the ColdFusion server --->
            <p>#cfcatch.message#</p>
            <p>Caught an exception, type = #CFCATCH.TYPE# </p>
            <p>The contents of the tag stack are:</p>
            <cfloop index = i from = 1
                    to = #ArrayLen(CFCATCH.TAGCONTEXT)#>
                <cfset sCurrent = #CFCATCH.TAGCONTEXT[i]#>
                <br>#i# #sCurrent["ID"]#
                    (#sCurrent["LINE"]#,#sCurrent["COLUMN"]#)
                    #sCurrent["TEMPLATE"]#
            </cfloop>
        </cfoutput>
    </cfcatch>
</cftry>
```

タグ u ～ z

cfupdate

説明
ColdFusion フォームまたは Form スコープ内のデータからデータソース内のレコードを更新します。

カテゴリ
データベース操作タグ

シンタックス
```
<cfupdate
    dataSource = "ds_name"
    tableName = "table_name"
    formFields = "field_names"
    password = "password"
    tableOwner = "name"
    tableQualifier = "qualifier"
    username = "username">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目
cfinsert、cfprocparam、cfprocresult、cfquery、cfqueryparam、cfstoredproc、cftransaction、『ColdFusion アプリケーションの開発』の Creating an update action page with cfupdate

履歴
ColdFusion 10：clientInfo 属性が追加されました。

ColdFusion MX: connectString、dbName、dbServer、dbtype、provider、および providerDSN 属性は非推奨になりました。ColdFusion 5 以降のリリースでは、これらは機能せず、エラーを引き起こす可能性があります。

属性

属性	必須 / オプション	デフォルト	説明
clientInfo	オプション		データベース接続で設定するクライアントのプロパティが含まれる構造体です。
dataSource	必須		テーブルが含まれているデータソースの名前です。
tableName	必須		更新するテーブルの名前です。 • Oracle ドライバの場合、大文字で指定する必要があります。 • Sybase ドライバの場合、大文字と小文字の区別があるため、テーブルの作成時に使用した名前と大文字小文字を同じにする必要があります。
formFields	オプション	(キーを除くフォーム上の全フィールド)	更新するフォームフィールドのカンマ区切りリストです。 フォームフィールドがデータベース内の列名と一致しない場合は、エラーが発生します。 formFields のリストには、データベーステーブルのプライマリキーフィールドを含める必要があります。このフィールドはフォームに必ず存在します。このフィールドは非表示にできます。

属性	必須 / オプション	デフォルト	説明
password	オプション		ODBC セットアップで指定されている password の値よりも優先されます。
tableOwner	オプション		テーブル所有権がサポートされているデータソース (SQL Server、Oracle、Sybase SQL Anywhere など) の場合は、テーブルの所有者です。
tableQualifier	オプション		テーブル修飾子がサポートされているデータソースに使用します。テーブル修飾子の内容は次のとおりです。 • SQL Server および Oracle の場合 : テーブルが含まれているデータベースの名前 • Intersolv dBASE ドライバの場合 : DBF ファイルのディレクトリ
username	オプション		ODBC セットアップで指定されている username の値よりも優先されます。

例

```
<!--- This example lets you update a person's telephone number in the employee table. --->
<cfif isDefined("form.phone")>
    <cfupdate datasource="cfdocexamples" tablename="EMPLOYEES">
</cfif>

<cfquery name="empTable" datasource="cfdocexamples">
    SELECT * FROM EMPLOYEES
</cfquery>

<!--- This code shows the contents of the employee table and allows you to choose a row for updating. --->
<table border="1">
<cfoutput query="empTable">
    <tr>
        <td>#firstName#</td>
        <td>#lastName#</td>
        <td>#phone#</td>
        <td><a href="cfupdate.cfm?id=#emp_id#">Edit</a></td>
    </tr>
</cfoutput>
</table>

<cfif isDefined("url.id")>
    <cfquery name="phoneQuery" datasource="cfdocexamples">
        SELECT * FROM employees WHERE emp_id=#url.id#
    </cfquery>
<!--- This code displays the row to edit for update. --->
    <cfoutput query="phoneQuery">
        <form action="cfupdate.cfm" method="post">
        #phoneQuery.firstName# #phoneQuery.lastName#
        <input name="phone" type="text" value="#phone#" size="12">
        <input type="submit" value="Update">
        <input name="emp_id" type="hidden" value="#emp_id#">
        <!--- The emp_id is passed as a hidden field to be used as a primary
            key in the CFUPDATE. --->
        </form>
    </cfoutput>
</cfif>
```

注意：cfupdate タグは、パラメータ化されたクエリーを内部的に使用します。

cfwddx

説明

CFML データ構造体の、XML ベースの WDDX 形式へのシリアル化およびシリアル化解除を行います。WDDX とは、標準の汎用的な方法で複雑なデータ構造体を記述するための XML の言語の一部です。WDDX を実装すると、アプリケーションサーバープラットフォーム、アプリケーションサーバー、およびブラウザなどの情報に対して HTTP プロトコルを使用できるようになります。

このタグでは、JavaScript ステートメントを生成して、WDDX パケットや CFML データ構造体の内容に相当する JavaScript オブジェクトのインスタンスを作成します。このタグは Unicode に対応しています。

カテゴリ

拡張タグ

シンタックス

```
<cfwddx
    action = "cfml2wddx|wddx2cfml|cfml2js|wddx2js"
    input = "inputdata"
    output = "result variable name"
    topLevelVariable = "top-level variable name for JavaScript"
    useTimeZoneInfo = "yes|no"
    validate = "yes|no" >
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfcollection、cfdump、cfexecute、cfindex、cfobject、cfreport、cfsearch、ToScript、『ColdFusion アプリケーションの開発』の Using XML and WDDX

履歴

ColdFusion MX

- 列名の大文字と小文字の区別の動作が変更されました。列名の大文字と小文字の区別が JavaScript 内で保持されます。以前のリリースでは、クエリー列の名前は小文字に変換されていました。
- エンコーディング形式のサポートが変更されました。このタグでは、複数のエンコード形式がサポートされます。デフォルトのエンコード形式は UTF-8 です。このタグは Unicode に対応しています。

属性

属性	必須 / オプション	デフォルト	説明
action	必須		• cfml2wddx: CFML をシリアル化して WDDX にします。 • wddx2cfml: WDDX のシリアル化を解除して、CFML にします。 • cfml2js: CFML をシリアル化して JavaScript にします。 • wddx2js: WDDX のシリアル化を解除して、JavaScript にします。
input	必須		処理する値です。
output	action="wddx2cfml" の場合は必須		出力する変数の名前です。action="WDDX2JS" または "CFML2JS" の場合、この属性を省略すると、結果は HTML ストリーム内に出力されます。

属性	必須 / オプション	デフォルト	説明
topLevelVariable	action="wddx2js" または "cfml2js" の場合は必須		シリアル化解除で作成したトップレベルの JavaScript オブジェクトの名前です。このオブジェクトは、WddxRecordset オブジェクトのインスタンスです。
useTimeZoneInfo	オプション	yes	CFML を WDDX にシリアル化する場合に、タイムゾーン情報を出力するかどうかを示します。 • yes: 時間と分のオフセットが ISO8601 形式で出力されます。 • no: ローカルな時刻が出力されます。
validate	オプション	no	action="wddx2cfml" または "wddx2js" の場合に適用されます。 • yes: WDDX DTD を使用して、WDDX 入力を XML パーサーで検証します。パーサーでエラーが発生することなく入力が処理されると、パケットのシリアル化は解除されます。そうでない場合は、エラーが発生します。 • no: 入力検証は行われません。

使用方法

列名の大文字と小文字の区別は JavaScript 内で保持されます。

RecordSet java オブジェクトが検出されると、wddx2js および cfml2js アクションによって、WddxRecordset javascript オブジェクトが作成されます。シリアル化された JavaScript コードには、wddx.js ファイルが必要です。

このタグで実行できる変換は、次のとおりです。

変換前	変換後
CFML	WDDX
CFML	JavaScript
WDDX	CFML
WDDX	JavaScript

詳細、および XML ドキュメントオブジェクトと関数の管理に使用できる ColdFusion の配列関数と構造体関数のリストについては、『ColdFusion アプリケーションの開発』の Using XML and WDDX を参照してください。

注意：CFC またはユーザー定義関数 (UDF) をシリアル化しようとすると、cfwddx タグで例外が発生します。

例

```
<!--- This example shows basic use of the cfwddx tag. --->
<html>
<body>
<!--- Create a simple query. --->
<cfquery name = "q" dataSource = "cfdocexamples">
    SELECT Message_Id, Thread_id, Username FROM messages
</cfquery>

The recordset data is:...<p>
<cfoutput query = q>
    #Message_ID# #Thread_ID# #Username#<br>
</cfoutput><p>

<!--- Serialize data to WDDX format. --->
Serializing CFML data...<p>
<cfwddx action = "cfml2wddx" input = #q# output = "wddxText">

<!--- Display WDDX XML packet. --->
Resulting WDDX packet is:
<xmp><cfoutput>#wddxText#</cfoutput></xmp>

<!--- Deserialize to a variable named wddxResult. --->
Deserializing WDDX packet...<p>
<cfwddx action = "wddx2cfml" input = #wddxText# output = "qnew">

The recordset data is:...<p>
<cfoutput query = qnew>
    #Message_ID# #Thread_ID# #Username#<br>
</cfoutput><p>
```

cfwebsocket

説明

CFM テンプレートで WebSocket オブジェクトを作成できます。このタグは、WebSocket の JavaScript オブジェクトへの参照をクライアントサイドに作成します。

カテゴリ

インターネットプロトコルタグ

シンタックス

```
<cfwebsocket
    name="websocketName"
    onMessage="JavaScript function name"
    onOpen="JavaScript function name"
    onClose="JavaScript function name"
    onError="JavaScript function name"
    useCfAuth=true|false
    subscribeTo="channel_list">
```

属性

属性	必須 / オプション	説明
name	必須	WebSocket オブジェクトの名前です。 これは、JavaScript オブジェクトへの参照で、WebSocket の JavaScript 関数の呼び出しに使用します。
onMessage	必須	WebSocket がサーバーからメッセージを受信したときに呼び出される JavaScript 関数です。 次の 2 つの引数がサポートされています。 • aEvent：サーバーから送信される WebSocket イベントです。 • aToken：サーバーから受信したデータを取得するために使用します。
onOpen	オプション	WebSocket が接続を確立したときに呼び出される JavaScript 関数です。
onClose	オプション	WebSocket が接続を閉じたときに呼び出される JavaScript 関数です。
onError	オプション	WebSocket 接続でのアクションの実行時にエラーが発生した場合に呼び出される JavaScript 関数です。
usecfAuth	オプション	true（デフォルト）に設定した場合、ユーザーは WebSocket 接続の認証を行う必要はありません（ただし、ユーザーが既にアプリケーションにログインしている場合）。これはデフォルト値です。 false の場合、ユーザーは WebSocket 接続のための資格情報を指定する必要があります。
subscribeTo	オプション	サブスクライブするチャネルのカンマ区切りリストです。Application.cfc で設定されたチャネルのすべてまたは任意のチャネルを指定できます。

履歴

ColdFusion 10: このタグが追加されました。

例

次の例では、

- ユーザーは、自動的に stocks チャネルにサブスクライブされます。
- Application.cfc で、このチャネル名を指定しています。
- この例では、デフォルトのチャネルリスナーを使用します。
- Index.cfm で、（subscribeTo 属性を使用して）ユーザーが自動的にサブスクライブ可能なチャネルを指定し、さらにメッセージハンドラーを定義します。

Application.cfc

```
component
{
	this.name="websocketsampleapp1";
	this.wschannels=[{name="stocks"}];
}
```

Index.cfm

```
<script type="text/javascript">
       function mymessagehandler(aevent, atoken)
       {
              var message = ColdFusion.JSON.encode(atoken);
           var txt=document.getElementById("myDiv");
           txt.innerHTML +=message  +"<br>";
       }
</script>
<cfwebsocket name="mycfwebsocketobject"  onmessage="mymessagehandler" subscribeto="stocks" >
<cfdiv id="myDiv"></cfdiv>
```

このコードでは、mycfwebsocketobject という名前の JavaScript の WebSocket オブジェクトを作成します。サーバーからメッセージが送信されると、mymessagehandler 関数が呼び出されます。

cfwebsocket タグで subscribeTo を指定したので、WebSocket オブジェクトにより、ユーザーは自動的に stocks チャネルにサブスクライブされます。

cfwindow

説明
ブラウザ内にポップアップウィンドウを作成します。ただし、独立したブラウザポップアップインスタンスは作成できません。

カテゴリ
表示管理タグ

シンタックス

```
<cfwindow
    bodyStyle = "CSS style specification"
    center="true|false"
    closable="true|false"
    destroyOnClose ="true|false"
    draggable="true|false"
    headerStyle = "CSS style specification"
    height="number of pixels"
    initShow="false|true"
    minHeight="number of pixels"
    minWidth="number of pixels"
    modal="true|false"
    name="string"
    onBindError = "JavaScript function name"
    refreshOnShow = "false|true"
    resizable="true|false"
    source="path"
    title="string"
    width="number of pixels"
    x="numeric pixel coordinate"
    y="numeric pixel coordinate">

        window contents

</cfwindow>
```

source 属性が指定されている場合、タグ本文の内容は無視されます。タグ本文を指定せず、</window> 終了タグを省略する場合は、cfwindow タグの最後を /> で閉じます。

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfajaximport、cfdiv、cflayout、cfpod、ColdFusion.Window.create、『ColdFusion アプリケーションの開発』の Using pop-up windows

履歴

ColdFusion 8: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
bodyStyle	オプション		ウィンドウ本体の CSS スタイル指定です。原則として、この属性は色とフォントスタイルを設定する目的に使用します。たとえば、この属性を使用して高さや幅を設定すると、予期しない結果になる場合があります。
center	オプション	false	このウィンドウをブラウザウィンドウの中央に配置するかどうかを指定するブール値です。 • true の場合、x および y 属性の値は無視されます。 • この属性が false に設定されていて、x および y 属性が指定されていない場合、ウィンドウは中央に配置されます。
closable	オプション	true	ユーザーがウィンドウを閉じることを許可するかどうかを指定するブール値です。true の場合は、ウィンドウ上に X 印のウィンドウを閉じるアイコンが表示されます。
destroyOnClose	オプション	false	true の場合、ウィンドウは閉じたときに廃棄されます。
draggable	オプション	true	ユーザーがウィンドウをドラッグすることを許可するかどうかを指定するブール値です。ウィンドウをドラッグするには、タイトルバーの上でマウスをクリックして、ボタンを押したままドラッグします。ウィンドウにタイトルがない場合はドラッグできません。
headerStyle	オプション		ウィンドウヘッダの CSS スタイル指定です。原則として、この属性は色とフォントスタイルを設定する目的に使用します。たとえば、この属性を使用して高さや幅を設定すると、予期しない結果になる場合があります。
height	オプション	300	ウィンドウの高さです (単位 : ピクセル)。使用可能なスペースを超える値を指定すると、使用可能なスペースがウィンドウによってすべて占有され、サイズを変更するためのハンドルが表示されません。
initShow	オプション	false	このウィンドウを含むページが最初に表示されたときに、このウィンドウを表示するかどうかを指定するブール値です。この値が false の場合は、ColdFusion.Window.show JavaScript 関数を使用してウィンドウを表示します。
minHeight	オプション	0	ユーザーがウィンドウのサイズを変更した場合の最小の高さです (単位 : ピクセル)。
minWidth	オプション	0	ユーザーがウィンドウのサイズを変更した場合の最小の幅です (単位 : ピクセル)。
modal	オプション	false	このウィンドウがモーダルであるかどうか (つまり、このウィンドウが表示されている間、ユーザーがメインウィンドウを操作できないようにするかどうか) を指定するブール値です。true の場合、ユーザーはメインウィンドウを操作できません。
name	オプション		ウィンドウの名前です。ページ上で一意である必要があります。ウィンドウを操作する必要がある場合 (ウィンドウをダイナミックに表示したり隠したりする必要がある場合)、この属性は必須です。
onBindError	オプション	「説明」を参照	バインド式の評価でエラーが発生した場合に実行する JavaScript 関数の名前です。この関数は 2 つの属性 (HTTP ステータスコードとメッセージ) を取る必要があります。 この属性を省略し、(ColdFusion.setGlobalErrorHandler 関数を使用して) グローバルエラーハンドラを指定すると、エラーメッセージが表示されます。それ以外の場合は、デフォルトのエラーポップアップが表示されます。

属性	必須 / オプション	デフォルト	説明
overflow	オプション	auto	子コンテンツのサイズが大きいためにコントロールがウィンドウの境界を超えてしまう場合に、子コンテンツをどのように表示するかを指定します。使用できる値は次のとおりです。 • auto: 必要に応じてスクロールバーを表示します。 • hidden: 境界からはみ出したコンテンツにアクセスできないようにします。 • scroll: 必要がない場合でも常に水平および垂直方向のスクロールバーを表示します。 • visible: ウィンドウ境界の外側にコンテンツを表示します。 **メモ :** Internet Explorer では、ウィンドウを visible に設定した場合、レイアウト領域を超えてコンテンツを表示するのではなく、コンテンツのサイズに合わせてウィンドウが拡大されます。
refreshOnShow	オプション	false	• true バインドイベントの発生時に加え、JavaScript 関数 ColdFusion.Window.show を呼び出すなどの方法でウィンドウが表示されるたびに、source のバインド式を実行してウィンドウのコンテンツを更新します。 • false バインドイベントによってバインド式が実行された場合にのみウィンドウを更新します。 この属性を使用する場合は、source 属性も指定する必要があります。
resizable	オプション	true	ユーザーがウィンドウのサイズを変更することを許可するかどうかを指定するブール値です。
source	オプション		ウィンドウのコンテンツを返す URL です。この属性で URL パラメータを使用すると、ページにデータを渡すことができます。ColdFusion は標準のページパス解決ルールを使用してページを探します。 この属性ではバインド式を使用できます。詳しくは、「使用方法」を参照してください。 **メモ :** この属性で指定した CFML ページに Ajax 機能を使用するタグ (cfform、cfgrid、cfpod など) が含まれている場合は、そのページで cfwindow タグとともに cfajaximport タグを使用する必要があります。詳細については、cfajaximport を参照してください。
title	オプション		ウィンドウのタイトルバーに表示するテキストです。HTML マークアップを使用するとタイトルの外観を制御できます (たとえば、赤いイタリックフォントでテキストを表示することができます)。
width	オプション	500	ウィンドウの幅です (単位 : ピクセル)。使用可能なスペースを超える値を指定すると、使用可能なスペースがウィンドウによってすべて占有され、サイズを変更するためのハンドルが表示されません。
x	オプション		ブラウザウィンドウを基準とした、ウィンドウの左上隅の X (水平) 座標です。 center 属性の値が true で、y 属性の値が設定されていない場合、この属性は無視されます。
y	オプション		ブラウザウィンドウを基準とした、ウィンドウの左上隅の Y (垂直) 座標です。 center 属性の値が true で、x 属性の値が設定されていない場合、この属性は無視されます。

使用方法

このタグをフォーム内で使用したり、cflayout または cflayoutarea タグの子として使用することはできません。

cfwindow タグは、ウィンドウを表示するページ (または cfinclude タグを使用してインクルードされるページ) で定義します。したがって、cfmenuitem タグの http 属性または cfdiv タグの bind 属性で指定されているページや、cflayoutarea または cfpod タグの source 属性で指定されているページで cfwindow タグを使用することはできません。たとえば、ユーザーがメ

ニューアイテムをクリックしたときにウィンドウを表示するには、メニューと同じページでウィンドウを定義し、cfmenuitem タグの http 属性で JavaScript 関数を使用してウィンドウの show 関数を呼び出します。cfwindow タグでは、source 属性を使用して別のページからコンテンツを取得します。

ウィンドウのコンテンツを指定するには、source 属性またはタグ本文を使用します。両方とも指定されている場合は、source 属性で指定されたコンテンツが使用され、タグ本文は無視されます。source 属性を使用すると、コンテンツの取得中にアニメーションのアイコンと「ロード中 ...」というテキストが表示されます。

JavaScript 関数の定義を含むページを source 属性で指定する場合、そのページの関数定義は次の形式に準拠している必要があります。

```
functionName = function(arguments) {function body}
```

次のような関数定義は、正常に動作しない可能性があります。

```
function functionName (arguments) {function body}
```

ただし、カスタム JavaScript すべてを外部の JavaScript ファイルにインクルードして、アプリケーションのメインページにインポートすることをお勧めします。source 属性を使用して取得するコードには、カスタム JavaScript をインラインで記述しないでください。インポートするページには、関数定義に関するこのような制限はありません。

source 属性を使用する場合は、バインド式を使用してフォームフィールドの値やその他のフォームコントロール属性をソース指定の一部として含めることができます。バインド可能なフォームコントロールは、HTML 形式のフォームコントロールのみです。バインド式の使用方法の詳細については、『ColdFusion アプリケーションの開発』の Binding data to form fields を参照してください。

JavaScript 関数

次の JavaScript 関数を使用すると、HTML 形式のウィンドウを管理できます。

関数	説明
ColdFusion.Window.create	cfwindow タグを使用せずにウィンドウを作成します。
ColdFusion.Window.getWindowObject	指定された HTML 形式の cfwindow コントロールの基盤となる Ext JS - JavaScript Library オブジェクトを取得します。
ColdFusion.Window.hide	ウィンドウを閉じます。
ColdFusion.Window.onHide	特定のウィンドウが閉じられるたびに実行する関数を指定します。
ColdFusion.Window.onShow	特定のウィンドウが開かれるたびに実行する関数を指定します。
ColdFusion.Window.show	ウィンドウを開きます。

例

次の例は、cfwindow タグのいくつかの機能と、cfwindow タグの source 属性をダイナミックにフォームコントロールにバインドする方法を示しています。また、x 属性と y 属性を使用してウィンドウの位置を指定する方法や、closable、resizable などの属性を使用してウィンドウの外観を制御する方法も示しています。さらに、フォームコントロールの値が変更されたときにバインド式を使用してウィンドウのコンテンツをダイナミックに更新する方法も示しています。

```
<html>
<head>
</head>

<body>
<cfform name="myform">
    <cfinput type="hidden" name="hiddentext"
        value="This is hidden text from the main page">

    Click the mouse on the control to show its text in window 1.
    <cfinput name="text1"><br />

    Click the button to show the input control text in window 2.
    <cfinput name="text2">
    <cfinput type="button" name="mybutton" value="Show Text"
        onclick="javascript:ColdFusion.Window.show('mywindow2')"><br />

    Click the Checkbox to change and show its status in window 3
    <cfinput name="check1" type="checkbox"><br />

    Click the button to open a window containing the page
        specified by the input control.
    <cfinput name="text3" value="windowsource.cfm">
    <cfinput type="button" name="mybutton3" value="Open Window"
        onclick="javascript:ColdFusion.Window.show('mywindow4')">
</cfform>

<!--- This window shows initially and cannot be closed, dragged, or resized.
        The value of the text URL parameter, and therefore, the contents of the
        text displayed in the window changes each time the user clicks the
        mouse over the text1 control. --->
<cfwindow x="0" y="100" width="200" height="150"
        name="mywindow" title="My First Window"
        closable="false" draggable="false" resizable="false" initshow="true"
        source="windowsource.cfm?text={myform:text1@mousedown}" />

<!--- This window shows initially and cannot be dragged, or resized, but can
        be closed.
        The text URL parameter represents the text2 input control value. --->
<cfwindow x="0" y="250" width="200" height="150"
        name="mywindow2" title="My Second Window"
        initshow="true" draggable="false" resizable="false"
        source="windowsource.cfm?text={myform:text2}" />

<!--- This window shows initially and cannot be resized, but can be dragged
        or closed.
        The value of the text URL parameter, and therefore, Boolean value
        displayed in the window changes each time the user clicks the mouse
```

```
        in the check1 control to change the check box status.
        The bind expression binds to the check box checked attribute;
        it specifies a click event because Internet Explorer does not
        generate a change event when the user clicks the box.--->
<cfwindow x="0" y="400" width="200" height="150"
        name="mywindow3" title="My Third Window"
        initshow="true" resizable="false"
        source="windowsource.cfm?text=<b>Checkbox: </b>{myform:check1.checked@click}" />

<!--- This window does not display initially.
        The Open Window button control opens it.
        It can be closed, dragged, and resized.
        The value text URL parameter represents the value of a hidden text
        field. --->
<cfwindow x="210" y="100" width="500" height="480" name="mywindow4"
        minHeight="400" minWidth="400"
        title="My Fourth Window" initshow="false"
        source="{myform:text3}?text={myform:hiddentext}" />
</body>
</html>
```

cfwindow タグの source 属性でウィンドウ内に表示するように指定されている windowsource.cfm ページには、次のコードが含まれています。

```
<h3>Main page input:</h3>
<cfoutput>
    #url.text#
</cfoutput>
```

cfxml

説明

タグ本文内のマークアップを含む ColdFusion XML ドキュメントを作成します。このタグには、XML タグおよび CFML タグを含めることができます。ColdFusion では、タグの本文に含まれている CFML コードを処理し、結果のテキストを XML ドキュメントオブジェクト変数に割り当てます。これは、常に Unicode で保管されます。

カテゴリ

拡張タグ

シンタックス

```
<cfxml
    variable="xmlVarName"
    caseSensitive="yes|no">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

IsXmlDoc、IsXmlElem、IsXmlRoot、ToString、XmlChildPos、XmlNew、XmlParse、XmlSearch、XmlTransform、『ColdFusion アプリケーションの開発』の Using XML and WDDX

履歴

ColdFusion MX 7: テキストの先頭で XML 宣言を使用できるようになりました。

ColdFusion MX: このタグが追加されました。

属性

属性	必須 / オプション	デフォルト	説明
variable			ドキュメントオブジェクトの名前です。
caseSensitive	オプション	no	• yes: ドキュメント要素と属性の大文字と小文字の区別を保持します。 • no

使用方法

大文字と小文字が区別される XML オブジェクトの要素名や属性名は、ドット表記法では参照できません。連想配列 (括弧) 表記法の名前を使用するか、大文字と小文字が区別される名前を使用せずに参照する (xmlChildren[1] などのようにする) 必要があります。次のコードで、最初の行は大文字と小文字を区別する XML オブジェクトに対して機能します。2 番目と 3 番目の行ではエラーが発生します。

```
MyDoc.xmlRoot.XmlAttributes["Version"] = "12b";
MyDoc.xmlRoot.XmlAttributes.Version = "12b";
MyDoc.MyRoot.XmlAttributes["Version"] = "12b";
```

XmlFormat 関数を使用して、&、>、< などの特殊文字をエスケープします。

XML ドキュメントオブジェクトを文字列に戻すには、ColdFusion で <?xml version="1.0" encoding="UTF-8" ?> XML 宣言が自動的に先頭に挿入されるときに ToString 関数を使用します。

宣言を変更して、別のエンコーディングを指定するには、Replace 関数を使用します。ブラウザや他のアプリケーションに返されるテキストのエンコーディングを指定するには、cfcontent タグを使用します。

次の例は、この処理を示しています。

```
<cfprocessingdirective suppresswhitespace="Yes">
<cfcontent type="text/xml; charset=utf-16">
<cfxml variable="xmlobject">
<breakfast_menu>
    <food>
        <name quantity="50">Belgian Waffles</name>
        <description>Our famous Belgian Waffles</description>
    </food>
</breakfast_menu>
</cfxml>

<!--- <cfdump var="#xmlobject#">--->

<cfset myvar=toString(xmlobject)>
<cfset mynewvar=replace(myvar, "UTF-8", "utf-16")>

<!---<cfdump var="#mynewvar#">--->

<cfoutput>#mynewvar#</cfoutput>
</cfprocessingdirective>
```

cfprocessingdirective タグを使用することにより、ColdFusion が XML 宣言の前に空白文字を置かないようにすることができます。

例

次の例では、ルート要素が MyDoc であるドキュメントオブジェクトを作成します。このオブジェクトには、ColdFusion 変数 testVar の値を示すテキストが含まれています。コード例では、4 つのネストされた子要素を作成します。これらはインデックスが付けられた cfloop タグを使用して生成できます。cfdump タグを使用して、XML ドキュメントオブジェクトを表示します。

```
<cfset testVar = True>
<cfxml variable="MyDoc">
    <?xml version='1.0' encoding='utf-8' ?>
    <MyDoc>
        <cfif testVar IS True>
            <cfoutput>The value of testVar is True.</cfoutput>
        <cfelse>
            <cfoutput>The value of testVar is False.</cfoutput>
        </cfif>
        <cfloop index = "LoopCount" from = "1" to = "4">
            <childNode>
                This is Child node <cfoutput>#LoopCount#.</cfoutput>
            </childNode>
        </cfloop>
    </MyDoc>
</cfxml>
<cfdump var=#MyDoc#>
```

cfzip

説明

ZIP ファイルと Java Archive (JAR) ファイルを操作します。cfzip タグを使用すると、基本的な圧縮 / 展開機能に加え、アーカイブからのエントリの削除、ファイルの抽出、バイナリ形式でのファイルの読み取り、アーカイブのコンテンツのリスト表示、実行可能 JAR ファイルで使用するエントリパスの指定などを行えます。

履歴

ColdFusion 8: このタグが追加されました。

カテゴリ

ファイル管理タグ

シンタックス

```
delete
<cfzip
    required
    action = "delete"
    file = "absolute pathname"
    optional
    entrypath = "full pathname"
    filter = "file filter"
    recurse = "yes|no">

list
<cfzip
    required
    action = "list"
    file = "absolute pathname"
    name = "recordset name"
    optional
    filter = "file filter"
    recurse = "yes|no"
    showDirectory= "yes|no">

read
<cfzip
    required
    action = "read"
```

```
    entrypath = "full pathname"
    file = "absolute pathname"
    variable = "variable name"
    optional
    charset = "encoding type">

readBinary
<cfzip
    required
    action = "readBinary"
    entrypath = "full pathname"
    file = "absolute pathname"
    variable = "variable name">

unzip
<cfzip
    required
    action = "unzip"
    destination = "destination directory"
    file = "absolute pathname"
    optional
    entrypath = "full pathname"
    filter = "file filter"
    overwrite = "yes|no"
    recurse = "yes|no"
    storePath = "yes|no">

zip
<cfzip
    required
    file = "absolute pathname"
    One of the following:
    source = "source directory"
    <cfzipparam source = "source directory" ...>
    optional
    action = "zip"
    filter = "file filter"
    overwrite = "yes|no"
    prefix = "string"
    recurse = "yes|no"
    storePath = "yes|no">
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfzipparam

属性

属性	アクション	必須 / オプション	デフォルト	説明
action	N/A	オプション	zip	実行するアクションです。次のいずれかになります。 • delete • list • read • readBinary • unzip • zip アクションを指定しなかった場合は、デフォルトのアクション zip が適用されます。
charset	read	オプション	ホストマシンのデフォルトのエンコード	ZIP または JAR エントリをテキスト文字列に変換するときに使用する文字セットです。たとえば、次の文字セットを指定できます。 • JIS • RFC1345 • UTF-16
destination	unzip	必須		ZIP または JAR ファイルの展開先ディレクトリです。
entryPath	delete read readBinary unzip	オプション		アクションを実行するパスです。
file	delete list read readBinary unzip zip	必須		アクションを実行する絶対パスです。たとえば、ZIP ファイルのフルパスを指定します (例 : c:¥temp¥log.zip)。 フルパスを指定しなかった場合は (例 : file="log.zip")、テンポラリディレクトリにファイルが作成されます。この ZIP または JAR ファイルにアクセスするには、GetTempDirectory 関数を使用します。
filter	delete list unzip zip	オプション		アクションに適用するファイルフィルタです。指定されたパスに存在するファイルの中で、フィルタに一致するファイルのみにアクションが適用されます。

属性	アクション	必須 / オプション	デフォルト	説明
name	list	必須		list アクションの結果を格納するレコードセット名です。レコードセットには次の列があります。 • name: JAR ファイルのエントリのファイル名。たとえば、エントリが help/docs/index.htm の場合、名前は index.htm です。 • directory: エントリを含むディレクトリ。上の例の場合、ディレクトリは help/docs です。ディレクトリと名前を結合すると完全なエントリ名を取得できます。エントリがルートレベルにある場合、ディレクトリは空 (" ") です。 • size: 展開後のエントリのサイズ (単位 : バイト)。 • compressedSize: 圧縮後のエントリのサイズ (単位 : バイト)。 • type: エントリのタイプ (ディレクトリまたはファイル)。 • dateLastModified: エントリの最終修正日 (cfdate オブジェクト)。 • comment: エントリに関するコメント (存在する場合)。 • crc: 展開後のエントリデータの Crc-32 チェックサム。
overwrite	unzip zip	オプション	no	unzip: 展開したファイルを上書きするかどうかを指定します。 • yes: 指定された展開先に同じ名前のファイルが存在する場合、ファイルは上書きされます。 • no: 指定された展開先に同じ名前のファイルが存在する場合、ファイルは上書きされず、そのエントリは展開されません。残りのエントリは展開されます。 zip: ZIP または JAR ファイルのコンテンツを上書きするかどうかを指定します。 • yes: 存在する場合は、ZIP または JAR ファイルのすべてのコンテンツを上書きします。 • no: 存在する場合は、既存のエントリを更新して、新しいエントリを ZIP または JAR ファイルに追加します。
prefix	zip	オプション		ZIP または JAR エントリに接頭辞として追加する文字列です。この文字列は、エントリを追加するサブディレクトリの名前です。
recurse	delete list unzip zip	オプション	yes	アクションをサブディレクトリに適用するかどうかを指定します。 • yes: サブディレクトリを対象に含めます。 • no: サブディレクトリを対象に含めません。
showDirectory	list	オプション	no	ディレクトリ構造を表示するかどうかを指定します。 • yes: ディレクトリをリスト表示します。 • no: ディレクトリをリスト表示しません。

属性	アクション	必須 / オプション	デフォルト	説明
source	zip	必須 (「説明」を参照)。		圧縮するソースディレクトリです。cfzipparam タグが指定されている場合は不要です。
storePath	unzip zip	オプション	yes	unzip: ファイルをエントリパスに展開するかどうかを指定します。 • yes: ファイルはエントリパスに展開されます。 • no: エントリパスは無視され、すべてのファイルがルートレベルに展開されます。 zip: ZIP または JAR ファイル内でパス名を保持するかどうかを指定します。 • yes: ZIP または JAR ファイル内でエントリのパス名が保持されます。 • no: ZIP または JAR ファイル内でエントリのパス名は保持されません。すべてのファイルがルートレベルに配置されます。同じ名前のファイルが存在する場合は、それらの中で最後のファイルが追加されます。
variable	read readBinary	必須		コンテンツが格納される変数です。

使用方法

cfzip タグを使用すると、ファイルの圧縮 / 展開を実行したり、ColdFusion 内の既存の ZIP または JAR ファイルを操作できます。cfzip タグは単独で使用することもできますが、cfzipparam タグと組み合わせて複数のファイルまたはディレクトリを操作することもできます。cfzip タグは、cfzipparam タグの親タグです。

ZIP 形式は、ファイルのアーカイブや圧縮に使用される標準形式です。JAR 形式は ZIP 形式をベースとしています。JAR ファイルはプラットフォームに依存しません。

注意：cfzip タグは、ディレクトリを作成しません。存在しないディレクトリを指定した場合も、エラーは生成されません。

次のシンタックスを使用して、パスを取る任意の属性でメモリ内のファイルまたはディレクトリを指定します。メモリ内のファイルはディスクに書き込まれないため、一時的データの処理が早くなります。

```
ram:///filepath
```

ファイルパスには、複数のディレクトリを含めることができます (例 : ram:///petStore/images/dogImages.zip)。ファイルを指定する前に、パスのディレクトリを作成しておく必要があります。メモリ内のファイルの使用方法について詳しくは、『ColdFusion アプリケーションの開発』の Working with in-memory files を参照してください。

delete アクション

ZIP または JAR ファイルからエントリを削除するには、delete アクションを使用します。

```
<!--- This example shows how to delete all the properties in a JAR file.
    --->
<cfzip file="e:\work\soure.jar" action="delete" filter="*.properties, *.props">
<!--- This example shows how to delete all of the entries in a ZIP file with a JPG, GIF, or PNG extension,
including entries in subdirectories. --->
<cfzip file="c:\myApp\images.zip" action="delete" filter="*.jpg, *.gif, *.png" recurse="yes">
<!--- This example shows how to delete the "images" subdirectory (and its contents) from the "myApp.zip"
file. --->
<cfzip action="delete" file="c:\myApp.zip" entrypath="images">
<!--- This example shows how to delete all Java source entries in the "work/source" directory and images
(*.gif, *.jpg, *.jpeg) from a JAR file. --->
<cfzip file="C:\downloads\source.jar" action="delete">
    <cfzipparam entrypath="work/source" filter="*.java">
    <cfzipparam filter="*.gif,*.jpg,*.jpeg">
</cfzip>
```

list アクション

ZIP または JAR ファイルのエントリをリスト表示するには、list アクションを使用します。次の表に、アーカイブ内のエントリに関して取得できる情報の種類を示します。

フィールド	説明
comment	エントリソースとともに保存されたテキスト文字列の説明です。
compressedSize	圧縮後のエントリのサイズです (単位 : バイト)。
crc	エントリソースのチェックサムです。
dateLastModified	ソースが最後に修正された日付と時刻です。
directory	エントリが保存されているディレクトリの名前です。
name	エントリのパス名です。
size	展開後のエントリソースのサイズです (単位 : バイト)。
type	エントリソースのタイプです (**file** など)。

cfdump タグを使用すると、ZIP または JAR ファイル内のすべての情報をリスト表示できます。次の例を参照してください。

```
<cfzip file="c:/myApp.jar" action="list" name="entry">
<cfdump var="#entry#">
```

cfoutput タグを使用すると、アーカイブ内のエントリの個々のフィールドをリスト表示できます。次の例を参照してください。

```
<cfzip file="c:\zipTest\Test.zip" action="list" name="entry">
<table>
    <cfoutput>
    <tr>
        <td><b>Entry Name:</b> #entry.name#</td>
        <td><b>Last Modified Date:</b>
#dateFormat(entry.dateLastModified)#,#timeFormat(entry.dateLastModified)#</td>
        <td><b>Size (uncompressed):</b> #numberFormat(entry.size/1000)# KB
    <br></td>
    </cfoutput>
    </tr>
</table>
```

read アクション

ZIP または JAR ファイルのエントリのコンテンツを人間が読み取れる形式で読み込むには、read アクションを使用します。read アクションでは、charset の値を使用して文字列が作成されます。

```
<!--- This example shows how to read a text file in a JAR file. --->
<cfzip action="read" file="/home/sam/work/util.jar" entrypath="info.txt" variable="text">
```

readBinary アクション

ZIP または JAR ファイルのコンテンツをバイナリ形式で読み込むには、readBinary アクションを使用します。

```
<!--- This example shows how to use the readBinary action to copy a ZIP entry from one ZIP file to another
ZIP file. --->
<cfzip file="c:\work\instr.zip" action="readBinary"
    entryPath="com/test/abc.jpg" variable="xyz">
<cfzip file="c:\work\copy_instr.zip">
    <cfzipparam entryPath="com/test/xyz.jpg" content="#xyz#">
</cfzip>
```

unzip アクション
ZIP または JAR ファイルからエントリを展開するには、unzip アクションを使用します。

```
<!--- This example shows how to extract the class files of a JAR file and save the files to a local drive.
--->
<cfzip file="e:\work\tools.jar" action="unzip" filter="*.class" destination="c:\temp\tools\classes"/>
<!--- This example shows how to extract files from a JAR file in multiple directories. --->
<cfzip file="e:\work\images.jar" action="unzip" destination="c:\images">
    <cfzipparam entryPath="toWork\small">
    <cfzipparam entryPath="final\large">
</cfzip>
```

zip アクション
ZIP または JAR ファイルを作成または更新するには、zip アクションを使用します。これはデフォルトのアクションです。明示的に指定する必要はありません。存在しない ZIP または JAR ファイルを指定すると、ColdFusion はそのファイルを作成します。既存の ZIP または JAR ファイルがある場合、ColdFusion はソースから新規のエントリを追加し、変更されている場合は既存のエントリを更新します。overwrite 属性を yes に設定した場合は、ZIP または JAR ファイル内のすべてのエントリが新しいコンテンツで上書きされます。

```
<!--- This example shows how to zip the directory "c:\temp" into the ZIP file "e:\work\abc.zip". --->
<cfzip file="e:\work\abc.zip" source="c:\temp">
<!--- This example shows how to zip all the class files in a directory and add a subdirectory named "classes"
to the JAR file entry name. --->
<cfzip file="e:\work\util.jar" action="zip" source="c:\src\util\" prefix="classes" filter="*.class">
<!---This example shows how to zip all of the log files in the ColdFusion directory and create a subdirectory
called exception where zipped files are archived.
<cfzip file="c:\zipTest\log2.zip" action="zip" source="c:\ColdFusion\" prefix="exception" filter="*.log">
<!--- This example shows how to overwrite all of the content of a ZIP file with the entries specified in
the source. --->
<cfzip file="c:\currentApp.zip" source="c:\myApp\work" overwrite="yes">
```

例
次の例は、フォームから選択されたイメージファイルを ZIP 形式で圧縮し、イメージを要求したユーザーにその ZIP ファイルを電子メールで送信する方法を示しています。

最初の ColdFusion ページによって、データベースクエリーから生成されたアーティストの名前がポップアップメニューに挿入されます。

```
<!--- Create a query to extract artist names from the cfartgallery database. --->
<cfquery name="artist" datasource="cfartgallery">
    SELECT FIRSTNAME || ' ' || LASTNAME AS FULLNAME,ARTISTS.ARTISTID
    FROM ARTISTS
</cfquery>

<!--- Create a form that lists the artists generated by the query. --->
<cfform action="zipArt_action.cfm" method="post">
<h3>Choose an Artist</h3>
<p>Please choose an artist:</p>
<cfselect name="artistName" query="artist" display="FULLNAME" value="ARTISTID" required="yes" multiple="no"
size="8">
</cfselect>
<br/><cfinput type="submit" name="submit" value="OK">
</cfform>
```

最初のアクションページでは、選択されたアーティストによる作品のイメージが表示され、それらのファイルが ZIP 形式で圧縮されて、テンポラリディレクトリに書き込まれます。このページには ZIP ファイルを電子メールで送信するためのフォームも含まれています。

```
<!--- Create a query to extract artwork for the selected artist from the cfartgallery database. --->
<cfquery name="artwork" datasource="cfartgallery">
    SELECT FIRSTNAME, LASTNAME, LARGEIMAGE
    FROM ARTISTS, ART
    WHERE ARTISTS.ARTISTID = ART.ARTISTID
    AND ARTISTS.ARTISTID=<cfqueryparam value="#form.artistName#">
    ORDER BY ARTNAME
</cfquery>

<cfoutput>
<p>You have chosen the work of #artwork.FirstName# #artwork.LastName#.</p>
<cfset thisDir = ExpandPath(".")>
<cfset imgDir = ExpandPath("..")>
</cfoutput>
<cfset xctr = 1>
<table border="0" cellpadding="15" cellspacing="0" bgcolor="#FFFFFF">
<cfoutput query="artwork">
    <cfif xctr mod 3 eq 1>
    <tr>
        </cfif>
        <!--- Use the IsImageFile function to verify that the image files
            extracted from the database are valid. Use the ImageNew function to
                create a ColdFusion image from valid image files. --->
        <cfif IsImageFile("#imgdir#/cfdocs/images/artgallery/
                #artwork.largeImage#")>
        <cfset myImage=ImageNew("#imgdir#/cfdocs/images/artgallery/
                #artwork.largeImage#")>
            <td valign="top" align="center" width="200">
            <cfset xctr = xctr + 1>
        <img src="#imgdir#/cfdocs/images/artgallery/#artwork.largeImage#"/>
            </td>

<!---Zip the files by the specified artist. --->
        <cfzip source="#imgDir#/cfdocs/images/artgallery/#artwork.LARGEIMAGE#"
            action="zip" file="#thisDir#/#artwork.lastname#.zip">
            </cfif>
                </cfoutput>
                </tr>
                </table>
<h3>Mail the ZIP File</h3>
<p>Please enter your e-mail address so we can send you the ZIP file as an attachment.</p>
<cfform action = "zipArt_action2.cfm" method="post">
Your e-mail address: <cfinput type = "Text" name = "MailTo">
<!--- Specify the required field. --->
    <cfinput type = "hidden" name = "MailTo_required" value = "You must enter
            your email address">
    <cfinput type="hidden" name="zipPath"
            value="#thisDir#/#artwork.lastname#.zip">
    <p><cfinput type = "Submit" name = "OK" label="Mail">
</cfform>
```

2 番目のアクションページでは、ZIP ファイルをメールに添付して送信します。

```
<h3>Mail the ZIP file</h3>
<p>Your file has been mailed to you.</p>
<cfset eMail="#form.MailTo#">
<cfset zipPath="#form.zipPath#">
<cfmail from="coldfusion@adobe.com" to="#eMail#" subject="see zipped attachment">
    The images you requested are enclosed in a ZIP file.
            <cfmailparam file="#zipPath#">
</cfmail>
```

cfzipparam

説明

cfzip タグに追加情報を提供します。

cfzipparam タグは、常に cfzip タグの子タグです。

履歴

ColdFusion 8: このタグが追加されました。

カテゴリ

ファイル管理タグ

シンタックス

```
<cfzip ..>
    <cfzipparam
        charset = "encoding type"
        content = "variable name"
        entryPath = "full pathname"
        filter = "file filter"
        prefix = "string"
        recurse = "yes|no"
        source = "source directory">
</cfzip>
```

注意：このタグの属性は attributeCollection 属性で指定でき、その値は構造体になります。attributeCollection 属性で構造体の名前を指定し、タグの属性名を構造体のキーとして使用します。

関連項目

cfzip

属性

属性	必須 / オプション	デフォルト	説明
charset	オプション	ホストマシンのデフォルトのエンコード	文字列コンテンツを ZIP または JAR ファイルに書き込む前にバイナリデータに変換します。cfzip action="zip" が指定されていて、cfzipparam のコンテンツが文字列である場合にのみ使用されます。 たとえば、次の文字セットを指定できます。 • JIS • RFC1345 • UTF-16
content	オプション		ZIP または JAR エントリに書き込むコンテンツです。cfzip action="zip" が指定されている場合にのみ使用されます。 有効なコンテンツデータのタイプは binary および string です。content 属性を指定する場合は、entrypath 属性を指定します。
entryPath	オプション		使用するパス名です。 • cfzip action="zip" の場合は、使用するエントリパスを指定します。これは、source がファイルの場合にのみ有効です。エントリパスを指定すると、ZIP または JAR ファイル内にサブディレクトリが作成されます。 • cfzip action="unzip" の場合は、展開先のパス名を指定します。 • cfzip action="delete" の場合は、ZIP または JAR ファイルから削除するパス名を指定します。
filter	オプション		アクションに適用するファイルフィルタです。たとえば、zip アクションの場合は、フィルタに一致する source ディレクトリ内のファイルがすべて圧縮されます。
prefix	オプション		ZIP または JAR エントリに接頭辞として追加する文字列です。cfzip action="zip" が指定されている場合にのみ使用されます。
recurse	オプション	yes	親の cfzip タグで指定されたアクション (圧縮、展開、または削除) をこのディレクトリに適用するかどうかを指定します。
source	オプション		ソースディレクトリまたはファイルです。cfzip action="zip" が指定されている場合にのみ使用されます。 指定されたファイルは、ZIP または JAR ファイルに追加されます。 • cfzip タグで source 属性が指定されている場合、cfzipparam source は相対パスで指定します。 • cfzip タグで source 属性が指定されていない場合、cfzipparam source は絶対パス名で指定する必要があります。

使用方法

複数のファイルまたはディレクトリを圧縮、展開、削除するには、cfzip タグと cfzipparam タグを組み合わせて使用します。たとえば、複数のディレクトリを圧縮するには、ソースディレクトリごとに cfzipparam タグを指定します。

例

例 1

```
<!--- This example shows how to zip class files from one subdirectory and class and property files from another directory into a JAR file. --->
<cfzip file="c:\util.jar" source="c:\myproj\classes">
    <cfzipparam source="com\abc\util" filter="*.class">
    <cfzipparam source="com\abc\io" filter="*.class, *.properties">
</cfzip>
```

例 2

```
<!--- This example shows how to update a ZIP file with files from multiple locations, each with a different
filter. --->
<cfzip file="e:\work\test.jar" action="zip">
    <cfzipparam source="c:\temp\abc.txt" prefix="com\abc">
    <cfzipparam source="c:\src\classes" recurse="yes"
            filter="*.class,*.properties" prefix="classes">
    <cfzipparam source="c:\src\Manifest.MF" entrypath="META-INF\MANIFEST">
</cfzip>
```

例 3

```
<!--- This example shows how to insert the string format for a programmatically generated XML file into a
ZIP file. --->
<!--- Create a variable that specifies a time-out period. --->
<cfset myDoc="<system-config><timeout>1500</timeout><pool-max-size>30
            </pool-max-size></system-config>">
<!--- Zip the file. --->
<cfzip file="e:\work\test.zip" action="zip">
    <cfzipparam source="c:\src\Manifest.MF" entrypath="META-INF\MANIFEST">
    <cfzipparam content="#myDoc#" entrypath="system-config.xml">
</cfzip>
```

例 4

```
<!--- This example shows how to update a JAR file with a new version of the file and add some new files to
the JAR file. --->
<cfzip file="e:\work\admin.jar">
    <cfzipparam source="c:\src\classes" recurse="yes"
            filter="*.class,*.properties">
    <cfzipparam source="c:\src\Manifest.MF" entrypath="META-INF\MANIFEST">
</cfzip>
```

例 5

次の例は、フォームから選択された複数のイメージファイルを ZIP 形式で圧縮し、イメージを要求したユーザーにその ZIP ファイルを電子メールで送信する方法を示しています。

最初の ColdFusion ページによって、データベースクエリーから生成されたアーティストの名前がポップアップメニューに挿入されます。

```
<!--- The following code creates a form for selecting images. --->
<h3>Select the images</h3>
<p>Please choose the images you would like sent to you.</p>
<!--- Create the ColdFusion form to select the images. --->
<table>
<cfform action="zip2_action.cfm" method="post"
    enctype="multipart/form-data">
    <tr>
        <td><img src="../cfdocs/images/artgallery/elecia01.jpg"/><br/>
        <cfinput type="checkbox" name="ck1" Value=1>Cube</td>
        <td><img src="../cfdocs/images/artgallery/elecia02.jpg"/><br/>
        <cfinput type="checkbox" name="ck2" Value=1>Pentagon</td>
        <td><img src="../cfdocs/images/artgallery/elecia03.jpg"/><br/>
        <cfinput type="checkbox" name="ck3" Value=1>Surfer Dude</td>
        <td><img src="../cfdocs/images/artgallery/elecia04.jpg"/><br/>
        <cfinput type="checkbox" name="ck4" Value=1>Surfer Girl</td></tr>
        <tr><td><cfinput type = "Submit" name = "OK" label="OK"></td></tr>
</table>
</cfform>
```

最初のアクションページでは、フォームから選択されたファイルを圧縮し、ZIP ファイルをハードドライブに書き込みます。このページには ZIP ファイルを電子メールで送信するためのフォームも含まれています。

```
<!--- Determine the absolute pathname on the server. --->
<cfset thisDir = ExpandPath(".")>

<!--- Create a ZIP file based on the selections from the form. Use the cfzipparam tag to specify the source
for each check box selection. --->
<cfzip file="c:\images.zip" source="#thisDir#" action="zip" overwrite="yes">
    <cfif IsDefined("form.ck1")>
        <cfzipparam source="../cfdocs/images/artgallery/elecia01.jpg">
        </cfif>
    <cfif IsDefined("form.ck2")>
        <cfzipparam source="../cfdocs/images/artgallery/elecia02.jpg">
        </cfif>
    <cfif IsDefined("form.ck3")>
        <cfzipparam source="../cfdocs/images/artgallery/elecia03.jpg">
        </cfif>
    <cfif IsDefined("form.ck4")>
        <cfzipparam source="../cfdocs/images/artgallery/elecia04.jpg">
        </cfif>
</cfzip>
<h3>Mail the ZIP File</h3>
<p>Please enter your e-mail address so we can send you the ZIP file as an attachment.</p>
<cfif IsDefined("form.mailto")>
    <cfif form.mailto is not "" >
        <cfoutput>
            <cfmail from="coldfusion@adobe.com" to="#form.mailto#"
                    subject="see zipped attachment">
    The images you requested are enclosed in a ZIP file.
            <cfmailparam file="#thisDir#/images.zip">
            </cfmail>
        </cfoutput>
        </cfif>
    </cfif>
<cfform action = "zipArt_action2.cfm" method="post">
    Your e-mail address: <cfinput type = "Text" name = "MailTo">
    <!--- Specify the required field. --->
    <cfinput type = "hidden" name = "MailTo_required"
    value = "You must enter your email address">
    <cfinput type = "hidden" name="zipPath" value = "c:\images.zip">
    <p><cfinput type = "Submit" name = "OK" label="Mail">
</cfform>
```

2 番目のアクションページでは、ZIP ファイルをメールに添付して送信します。

```
<h3>Mail the ZIP file</h3>
<p>Your file has been mailed to you.</p>
<cfset eMail="#form.MailTo#">
<cfset zipPath="#form.zipPath#">
<cfmail from="coldfusion@adobe.com" to="#eMail#"
    subject="see zipped attachment">
    The images you requested are enclosed in a ZIP file.
    <cfmailparam file="#zipPath#">
</cfmail>
```

第 4 章：ColdFusion 関数

以下の表に、CFML (ColdFusion Markup Language) 関数のカテゴリ別一覧を示します。

ColdFusion 10 の新規関数

ArraySlice
ArrayEach
ArrayFilter
ArrayFindAll
ArrayFindAllNoCase
ArrayFindNoCase

CacheIdExists
CacheRegionNew
CacheRegionRemove
CacheRemoveAll
Canonicalize
CacheRegionExists

CallStackDump
CallStackGet
CSRFGenerateToken
CSRFVerifyToken
DateTimeFormat
DecodeForHTML

DecodeFromURL
DirectoryCopy
EncodeForHTML
EncodeForCSS
EncodeForHTMLAttribute
EncodeForJavaScript

EncodeForURL
EncodeForXML
FileUploadAll
FileGetMimeType
GetApplicationMetadata
GetCPUUsage

GetFreeSpace
ImageMakeTranslucent
Invoke
IsClosure
ListFilter
LSDateTimeFormat

ListRemoveDuplicates
onWSAuthenticate
ORMIndex
ORMIndexPurge
ORMSearch
ORMSearchOffline

ReEscape
RestInitApplication
RemoveCachedQuery
RestDeleteApplication
RestSetResponse
SessionRotate

sessionGetMetaData
SessionInvalidate
StructEach
StructFilter
WsGetAllChannels
WsGetSubscribers

WsPublish
WSSendMessage

GetTotalSpace

GetSystemFreeMemory

GetSystemTotalMemory

HMac

ImageCreateCaptcha

ImageMakeColorTransparent

カテゴリ別の関数

以下の表に、関数の一覧をカテゴリまたは用途別に示します。

配列関数

ArrayAppend
ArrayAvg
ArrayClear
ArrayDeleteAt
ArrayInsertAt
ArrayContains
ArrayEach
ArraySlice
ArrayIsDefined
ArrayIsEmpty
ArrayLen
ArrayMax
ArrayMin
ArrayDelete
ArrayFilter
ArrayNew
ArrayPrepend
ArrayResize
ArraySet
ArraySort
ArrayFind
ArrayFindAll
ArraySum
ArraySwap
ArrayToList
IsArray
ListToArray
ArrayFindNoCase
ArrayFindAllNoCase

キャッシュ関数

CacheGet
CacheGetAllIds
CacheRegionNew
CacheRegionExists
CacheGetMetadata
CacheGetProperties
CacheRegionRemove
CachePut
CacheRemove
CacheRemoveAll
CacheSetProperties
CacheIdExists
Canonicalize

変換関数

ArrayToList
BinaryDecode
BinaryEncode
CharsetDecode
CharsetEncode
DeserializeJSON
DotNetToCFType
Hash
LCase
ListToArray
SerializeJSON
ToBase64
ToBinary
ToScript
ToString
URLDecode
URLEncodedFormat
Val
XmlFormat
XmlParse
XmlTransform

日付および時刻関数

CreateDate
CreateDateTime
CreateODBCDate
CreateODBCDateTime
CreateODBCTime
CreateTime
CreateTimeSpan
DateAdd
DateCompare
DateConvert
DateDiff
DateFormat
DatePart
Day
DayOfWeek
DayOfWeekAsString
DayOfYear
DaysInMonth
DaysInYear
FirstDayOfMonth
GetHttpTimeString
GetTickCount
GetTimeZoneInfo
Hour
IsDate
IsLeapYear
IsNumericDate
LSDateFormat
LSIsDate
LSParseDateTime
LSTimeFormat
Minute
Month
MonthAsString
Now
ParseDateTime
Quarter
Second
TimeFormat
Week
Year
DateTimeFormat

データ出力関数

WriteDump
WriteLog
WriteOutput

デバッグ関数

Trace

決定関数

DirectoryExists
FileExists
FileIsEOF
Iif
IsArray
IsBinary
IsBoolean
IsCustomFunction
IsDate
IsDebugMode
IsDDX
IsDefined
IsInstanceOf
IsJSON
IsK2ServerABroker
IsK2ServerDocCountExceeded
IsK2ServerOnline
IsLeapYear
IsNumeric
IsNumericDate
IsObject
IsNull
IsPDFFile
IsPDFObject
IsQuery
IsSimpleValue
IsStruct
IsUserInAnyRole
IsValid
IsWDDX
IsXML
IsXmlAttribute
IsXmlDoc
IsXmlElem
IsXmlNode
IsXmlRoot
LSIsCurrency
LSIsDate
LSIsNumeric
StructIsEmpty
StructKeyExists
YesNoFormat

表示および書式制御関数

AjaxLink
AjaxOnLoad
CJustify
DateFormat
DecimalFormat
DollarFormat
FormatBaseN
GetLocale
GetLocaleDisplayName
HTMLCodeFormat
HTMLEditFormat
LJustify
LSCurrencyFormat
LSDateFormat
LSEuroCurrencyFormat
LSIsCurrency
LSIsDate
LSNumberFormat
LSParseCurrency
LSParseDateTime
LSParseEuroCurrency
LSParseNumber
LSTimeFormat
NumberFormat
ParagraphFormat
RJustify
StripCR
TimeFormat
YesNoFormat

ダイナミック評価関数

DE
Evaluate
IIf
PrecisionEvaluate
SetVariable

例外処理関数

Throw

拡張関数

CreateObject
DotNetToCFType
GetComponentMetaData
GetGatewayHelper
IsInstanceOf
ReleaseComObject
SendGatewayMessage
ToScript

フロー制御関数

Throw
Location

全文検索関数

履歴
ColdFusion MX 6.1: これらの関数は非推奨になりました。今後のリリースでは、これらは機能せずエラーが発生する可能性があります。

GetK2ServerDocCountLimit
IsK2ServerABroker
IsK2ServerDocCountExceeded
IsK2ServerOnline

Image 関数

ImageAddBorder
ImageBlur
ImageClearRect
ImageCopy
ImageCrop
ImageDrawArc
ImageDrawBeveledRect
ImageDrawCubicCurve
ImageDrawLine
ImageDrawLines
ImageDrawOval
ImageDrawPoint
ImageDrawQuadraticCurve
ImageDrawRect
ImageDrawRoundRect
ImageDrawText
ImageFlip
ImageGetBlob
ImageGetBufferedImage
ImageGetEXIFTag
ImageGetHeight
ImageGetIPTCTag
ImageGetWidth
ImageGrayscale
ImageInfo
ImageNegative
ImageNew
ImageOverlay
ImagePaste
ImageRead
ImageReadBase64
ImageResize
ImageRotate
ImageRotateDrawingAxis
ImageScaleToFit
ImageSetAntialiasing
ImageSetBackgroundColor
ImageSetDrawingColor
ImageSetDrawingStroke
ImageSetDrawingTransparency
ImageSharpen
ImageShear
ImageShearDrawingAxis
ImageTranslate
ImageTranslateDrawingAxis
ImageWrite
ImageWriteBase64
ImageXORDrawingMode
IsImage
IsImageFile

各国語対応関数

DateConvert
GetEncoding
GetHttpTimeString
GetLocale
GetLocaleDisplayName
GetTimeZoneInfo
LSIsCurrency
LSCurrencyFormat
LSDateFormat
LSEuroCurrencyFormat
LSIsDate
LSParseDateTime
LSIsNumeric
LSNumberFormat
LSParseCurrency
LSParseEuroCurrency
LSParseNumber
LSTimeFormat
SetLocale

リスト関数

ArraySort
ArrayToList
Asc
Chr
CJustify
Compare
CompareNoCase
Decrypt
Encrypt
Find
FindNoCase
FindOneOf
FormatBaseN
GetClientVariablesList
LCase
Left
Len
ListAppend
ListChangeDelims
ListContains
ListContainsNoCase
ListDeleteAt
ListFind
ListFindNoCase
ListFirst
ListGetAt
ListInsertAt
ListLast
ListLen
ListPrepend
ListQualify
ListRest
ListSetAt
ListSort
ListToArray
ListValueCount
ListValueCountNoCase
ReplaceList
ValueList

算術関数

Abs
ACos
ArrayAvg
ArraySum
ASin
Atn
BitAnd
BitMaskClear
BitMaskRead
BitMaskSet
BitNot
BitOr
BitSHLN
BitSHRN
BitXor
Ceiling
Cos
DecrementValue
Exp
Fix
FormatBaseN
IncrementValue
InputBaseN
Int
Log
Log10
Max
Min
Pi
PrecisionEvaluate
Rand
Randomize
RandRange
Round
Sgn
Sin
Sqr
Tan

Microsoft Office の統合関数

IsSpreadsheetFile
IsSpreadsheetObject
SpreadsheetAddColumn
SpreadsheetAddFreezePane
SpreadsheetAddImage
SpreadsheetAddInfo
SpreadsheetAddRow
SpreadsheetAddRows
SpreadsheetAddSplitPane
SpreadsheetCreateSheet
SpreadsheetDeleteColumn
SpreadsheetDeleteColumns
SpreadsheetDeleteRow
SpreadsheetDeleteRows
SpreadsheetFormatCell
SpreadsheetFormatCellRange
SpreadsheetFormatColumn
SpreadsheetFormatColumns
SpreadsheetFormatRow
SpreadsheetFormatRows
SpreadsheetGetCellComment
SpreadsheetGetCellFormula
SpreadsheetGetCellValue
SpreadsheetRead
SpreadsheetRemoveSheet
SpreadsheetInfo
SpreadsheetMergeCells
SpreadsheetNew
SpreadsheetReadBinary
SpreadsheetSetActiveSheetNumber
SpreadsheetSetCellComment
SpreadsheetSetCellFormula
SpreadsheetSetCellValue
SpreadsheetSetActiveSheet
SpreadsheetSetFooter
SpreadsheetSetHeader
SpreadsheetSetColumnWidth
SpreadsheetShiftColumns
SpreadsheetShiftRows
SpreadsheetSetRowHeight
SpreadsheetWrite
Trace

ORM 関数

EntityDelete
EntityLoad
EntityLoadByExample
EntityLoadByPK
EntityMerge
EntityNew
EntityReload
EntitySave
EntitytoQuery
ORMClearSession
ORMCloseSession
ORMEvictCollection
ORMEvictEntity
ORMEvictQueries
ORMExecuteQuery
ORMFlush
ORMGetSession
ORMGetSessionFactory
ORMReload

その他の関数

ApplicationStop
CreateUUID
DeleteClientVariable
GetBaseTagData
GetBaseTagList
GetBaseTemplatePath
GetClientVariablesList
GetLocalHostIP
IsLocalHost
ObjectEquals
ObjectLoad
ObjectSave
PreserveSingleQuotes
URLSessionFormat
WriteDump
WriteLog
WriteOutput

クエリー関数

IsQuery
QueryAddColumn
QueryAddRow
QueryConvertForGrid
QueryNew
QuerySetCell
QuotedValueList
ValueList

セキュリティ関数

Decrypt
DecryptBinary
Encrypt
EncryptBinary
GetAuthUser
GenerateSecretKey
GetTempDirectory
GetTempFile
GetUserRoles
Hash
IsUserInAnyRole
IsUserInRole
IsUserLoggedIn
VerifyClient

スプレッドシート関数

SpreadsheetAddColumn
SpreadsheetAddImage
SpreadsheetAddFreezePane
SpreadsheetAddInfo
SpreadsheetAddRow
SpreadsheetAddRows
SpreadsheetAddSplitPane
SpreadsheetCreateSheet
SpreadsheetDeleteColumn
SpreadsheetDeleteColumns
SpreadsheetDeleteRow
SpreadsheetDeleteRows1249 ページの「SpreadsheetDeleteRows」
SpreadsheetFormatCell
SpreadsheetFormatColumn
SpreadsheetFormatCellRange
SpreadsheetFormatColumns
SpreadsheetFormatRow1258 ページの「SpreadsheetFormatRow」
SpreadsheetFormatRows1259 ページの「SpreadsheetFormatRows」
SpreadsheetGetCellComment
SpreadsheetGetCellFormula
SpreadsheetGetCellValue
SpreadsheetInfo
SpreadsheetMergeCells
SpreadsheetNew
SpreadsheetRead
SpreadsheetReadBinary
SpreadsheetRemoveSheet
SpreadsheetSetActiveSheet
SpreadsheetSetActiveSheetNumber
SpreadsheetSetCellComment
SpreadsheetSetCellFormula
SpreadsheetSetCellValue
SpreadsheetSetColumnWidth
SpreadsheetSetFooter
SpreadsheetSetHeader
SpreadsheetSetRowHeight
SpreadsheetShiftColumns
SpreadsheetShiftRows
SpreadsheetWrite

文字列関数

履歴
ColdFusion MX: Unicode 文字値 0 ～ 65535 の Java UCS-2 表現がサポートされるようになりました (以前のリリースでは ASCII 値がサポートされていました)。

文字列処理関数では、ASCII 0 (nul) 文字を含むこれらの文字が処理され、後続の文字が継続してカウントされます (これまでのリリースでは、一部の文字列処理関数において、ASCII 0 (NUL) 文字の処理は行われましたが、文字列内の後続の文字が処理されませんでした)。

Asc
BinaryDecode
BinaryEncode
CharsetDecode
CharsetEncode
Chr
CJustify
Compare
CompareNoCase
DayOfWeekAsString
Decrypt
Encrypt
Find
FindNoCase
FindOneOf
FormatBaseN
GenerateSecretKey
GetToken
Hash
HTMLCodeFormat
HTMLEditFormat
Insert
JavaCast
JSStringFormat
LCase
Left
Len
LJustify
ListValueCount
LSParseNumber
LTrim
ListValueCountNoCase
LSIsDate
LSIsNumeric
LSParseCurrency
LSIsCurrency
LSParseDateTime
LSParseEuroCurrency
Mid
MonthAsString
ParagraphFormat
ParseDateTime
REFind
REFindNoCase
REMatch
REMatchNoCase
RemoveChars
RepeatString
Replace
RTrim
SpanExcluding
ReplaceNoCase
REReplace
REReplaceNoCase
ReplaceList
Reverse
Right
RJustify
SpanIncluding
StripCR
ToBase64
ToBinary
ToString
Trim
UCase
URLDecode
URLEncodedFormat
Val
Wrap
XmlFormat

711 ページの「変換関数」も参照してください。

構造体関数

Duplicate
IsStruct
StructAppend
StructClear
StructCopy
StructCount
StructDelete
StructFind
StructFindKey
StructFindValue
StructGet
StructInsert
StructIsEmpty
StructKeyArray
StructKeyExists
StructKeyList
StructNew
StructSort
StructUpdate

システム関数

DirectoryExists
Duplicate
ExpandPath
FileClose
FileCopy
FileDelete
FileExists
FileIsEOF
FileMove
FileOpen
FileRead
FileReadBinary
FileReadLine
FileSetAccessMode
FileSetAttribute
FileSetLastModified
DirectoryDelete
FileSeek
FileWrite
GetBaseTemplatePath
GetContextRoot
GetCurrentTemplatePath
GetDirectoryFromPath
GetDirectoryFromPath
GetEncoding
GetException
GetFileFromPath
GetFileInfo
GetFunctionList
GetHttpRequestData
GetLocale
GetLocaleDisplayName
GetMetaData
GetMetricData
DirectoryList
FileSkipBytes
GetPageContext
GetPrinterInfo
GetProfileSections
GetProfileString
GetReadableImageFormats
GetTempDirectory
GetTempFile
GetTemplatePath
GetTickCount
GetWriteableImageFormats
SetLocale
SetProfileString
Sleep
WriteOutput
DirectoryExists
DirectoryCreate
DirectoryRename
GetFunctionCalledName

トランザクション関数

TransactionCommit
TransactionRollback
TransactionSetSavePoint

XML 関数

AddSOAPRequestHeader
AddSOAPResponseHeader
GetSOAPRequest
GetSOAPRequestHeader
GetSOAPResponse
GetSOAPResponseHeader
IsSOAPRequest
IsXML
IsXmlAttribute
IsXmlDoc
IsXmlElem
IsXmlNode
IsXmlRoot
IsWDDX
ToString
XmlChildPos
XmlElemNew
XmlFormat
XmlGetNodeType
XmlNew
XmlParse
XmlSearch
XmlTransform
XmlValidate

ColdFusion 5 以降に変更された関数

以下の表に、ColdFusion 5 以降に変更された関数、パラメータ、および値を示します。また、その変更がどのリリースで加えられたかも併せて示します。

新規の関数、パラメータ、および値

関数	パラメータまたは値	追加された ColdFusion リリース
AjaxLink	すべて	ColdFusion 8
AjaxOnLoad	すべて	ColdFusion 8
ArrayIsDefined	すべて	ColdFusion 8

関数	パラメータまたは値	追加された ColdFusion リリース
BinaryDecode	すべて	ColdFusion MX 7
BinaryEncode	すべて	ColdFusion MX 7
CharsetDecode	すべて	ColdFusion MX 7
CharsetEncode	すべて	ColdFusion MX 7
CreateObject	type パラメータの .net 値と、関連する assembly、server、port、protocol、secure パラメータ Web サービスオブジェクト用の WSDL2Java パラメータと argStruct パラメータ	ColdFusion 8
	portName パラメータ	ColdFusion MX 7
	すべて	ColdFusion MX
DateAdd	datepart パラメータの l キー	ColdFusion MX 6.1
DatePart	datepart パラメータの l キー	ColdFusion MX 6.1
Decrypt	IVorSalt パラメータと iterations パラメータ	ColdFusion MX 7.0.1
	algorithm パラメータと encoding パラメータ	ColdFusion MX 7
DecryptBinary	すべて	ColdFusion MX 7.0.1
DeserializeJSON	すべて	ColdFusion 8
DotNetToCFType	すべて	ColdFusion 8
Encrypt	IVorSalt パラメータと iterations パラメータ	ColdFusion MX 7.0.1
	algorithm パラメータと encoding パラメータ	ColdFusion MX 7
EncryptBinary	すべて	ColdFusion MX 7.0.1
FileClose	すべて	ColdFusion 8
FileCopy	すべて	ColdFusion 8
FileDelete	すべて	ColdFusion 8
FileIsEOF	すべて	ColdFusion 8
FileMove	すべて	ColdFusion 8
FileOpen	すべて	ColdFusion 8
FileRead	すべて	ColdFusion 8
FileReadBinary	すべて	ColdFusion 8
FileReadLine	すべて	ColdFusion 8
FileSetAccessMode	すべて	ColdFusion 8
FileSetAttribute	すべて	ColdFusion 8
FileSetLastModified	すべて	ColdFusion 8
FileWrite	すべて	ColdFusion 8
GenerateSecretKey	すべて	ColdFusion MX 7

関数	パラメータまたは値	追加された ColdFusion リリース
GetGatewayHelper	すべて	ColdFusion MX 7
GetAuthUser	すべて	ColdFusion MX
GetComponentMetaData	すべて	ColdFusion 8
GetContextRoot	すべて	ColdFusion MX 7
GetEncoding	すべて	ColdFusion MX
GetFileInfo	すべて	ColdFusion 8
GetLocaleDisplayName	すべて	ColdFusion MX 7
GetLocalHostIP	すべて	ColdFusion MX 7.0.1
GetMetaData	すべて	ColdFusion MX
GetPageContext	すべて	ColdFusion MX
GetPrinterInfo	すべて	ColdFusion 8
GetProfileSections	すべて	ColdFusion MX
GetReadableImageFormats	すべて	ColdFusion 8
GetSOAPRequest	すべて	ColdFusion MX 7
GetSOAPRequestHeader	すべて	ColdFusion MX 7
GetSOAPResponse	すべて	ColdFusion MX 7
GetSOAPResponseHeader	すべて	ColdFusion MX 7
GetUserRoles	すべて	ColdFusion 8
GetWriteableImageFormats	すべて	ColdFusion 8
Hash	algorithm パラメータと encoding パラメータ	ColdFusion MX 7
ImageAddBorder	すべて	ColdFusion 8
ImageBlur	すべて	ColdFusion 8
ImageClearRect	すべて	ColdFusion 8
ImageCopy	すべて	ColdFusion 8
ImageCrop	すべて	ColdFusion 8
ImageDrawArc	すべて	ColdFusion 8
ImageDrawBeveledRect	すべて	ColdFusion 8
ImageDrawCubicCurve	すべて	ColdFusion 8
ImageDrawPoint	すべて	ColdFusion 8
ImageDrawLine	すべて	ColdFusion 8
ImageDrawLines	すべて	ColdFusion 8
ImageDrawOval	すべて	ColdFusion 8
ImageDrawQuadraticCurve	すべて	ColdFusion 8
ImageDrawRect	すべて	ColdFusion 8

関数	パラメータまたは値	追加された ColdFusion リリース
ImageDrawRoundRect	すべて	ColdFusion 8
ImageDrawText	すべて	ColdFusion 8
ImageFlip	すべて	ColdFusion 8
ImageGetBlob	すべて	ColdFusion 8
ImageGetBufferedImage	すべて	ColdFusion 8
ImageGetEXIFMetadata	すべて	ColdFusion 8
ImageGetEXIFTag	すべて	ColdFusion 8
ImageGetHeight	すべて	ColdFusion 8
ImageGetIPTCMetadata	すべて	ColdFusion 8
ImageGetIPTCTag	すべて	ColdFusion 8
ImageGetWidth	すべて	ColdFusion 8
ImageGrayscale	すべて	ColdFusion 8
ImageInfo	すべて	ColdFusion 8
ImageNegative	すべて	ColdFusion 8
ImageNew	すべて	ColdFusion 8
ImageOverlay	すべて	ColdFusion 8
ImagePaste	すべて	ColdFusion 8
ImageRead	すべて	ColdFusion 8
ImageReadBase64	すべて	ColdFusion 8
ImageResize	すべて	ColdFusion 8
ImageRotate	すべて	ColdFusion 8
ImageRotateDrawingAxis	すべて	ColdFusion 8
ImageScaleToFit	すべて	ColdFusion 8
ImageSetAntialiasing	すべて	ColdFusion 8
ImageSetBackgroundColor	すべて	ColdFusion 8
ImageSetDrawingColor	すべて	ColdFusion 8
ImageSetDrawingStroke	すべて	ColdFusion 8
ImageSetDrawingTransparency	すべて	ColdFusion 8
ImageSharpen	すべて	ColdFusion 8
ImageShear	すべて	ColdFusion 8
ImageShearDrawingAxis	すべて	ColdFusion 8
ImageTranslate	すべて	ColdFusion 8
ImageTranslateDrawingAxis	すべて	ColdFusion 8
ImageWrite	すべて	ColdFusion 8

関数	パラメータまたは値	追加された ColdFusion リリース
ImageWriteBase64	すべて	ColdFusion 8
ImageXORDrawingMode	すべて	ColdFusion 8
IsDDX	すべて	ColdFusion 8
IsImage	すべて	ColdFusion 8
IsImageFile	すべて	ColdFusion 8
IsInstanceOf	すべて	ColdFusion 8
IsJSON	すべて	ColdFusion 8
IsLocalHost	すべて	ColdFusion MX 7.0.1
IsObject	すべて	ColdFusion MX
IsPDFFile	すべて	ColdFusion 8
IsPDFObject	すべて	ColdFusion 8
IsSOAPRequest	すべて	ColdFusion MX 7
IsUserInAnyRole	すべて	ColdFusion 8
IsUserInRole	すべて	ColdFusion MX
IsUserLoggedIn	すべて	ColdFusion 8
IsValid	すべて	ColdFusion MX 7
IsXML	すべて	ColdFusion MX 7
IsXmlAttribute	すべて	ColdFusion MX 7
IsXmlDoc	すべて	ColdFusion MX
IsXmlElem	すべて	ColdFusion MX
IsXmlNode	すべて	ColdFusion MX 7
IsXmlRoot	すべて	ColdFusion MX
LSTimeFormat	mask パラメータの l キー	ColdFusion MX 6.1
QueryAddColumn	datatype パラメータ	ColdFusion MX 7
QueryConvertForGrid	すべて	ColdFusion 8
QueryNew	columntypelist パラメータ	ColdFusion MX 7
PrecisionEvaluate	すべて	ColdFusion 8
Rand	algorithm パラメータ	ColdFusion MX 7
Randomize	algorithm パラメータ	ColdFusion MX 7
RandRange	algorithm パラメータ	ColdFusion MX 7
ReleaseComObject	すべて	ColdFusion MX 6.1
REMatch	すべて	ColdFusion 8
REMatchNoCase	すべて	ColdFusion 8
SerializeJSON	すべて	ColdFusion 8

関数	パラメータまたは値	追加された ColdFusion リリース
SendGatewayMessage	すべて	ColdFusion MX 7
Sleep	すべて	ColdFusion 8
TimeFormat	mask パラメータの l キー	ColdFusion MX 6.1
ToScript	すべて	ColdFusion MX 7
URLDecode	charset パラメータ	ColdFusion MX
URLEncodedFormat	charset パラメータ	ColdFusion MX
URLSessionFormat	すべて	ColdFusion MX
VerifyClient	すべて	ColdFusion 8
Wrap	すべて	ColdFusion MX 6.1
XmlChildPos	すべて	ColdFusion MX
XmlElemNew	すべて	ColdFusion MX
XmlElemNew	namespace パラメータ	ColdFusion MX 7
XmlGetNodeType	すべて	ColdFusion MX 7
XmlNew	すべて	ColdFusion MX
XmlParse	すべて	ColdFusion MX
XmlParse	validator パラメータ	ColdFusion MX 7
XmlSearch	すべて	ColdFusion MX
XmlTransform	すべて	ColdFusion MX
XmlTransform	parameters パラメータ	ColdFusion MX 7
XmlValidate	すべて	ColdFusion MX 7

非推奨の関数、パラメータ、および値

次の関数、パラメータ、および値は非推奨になっています。これらを ColdFusion のアプリケーションで使用しないでください。ColdFusion MX よりも新しいリリースでは、これらは動作せずエラーが発生する可能性があります。

関数	パラメータまたは値	非推奨となった ColdFusion リリース
GetMetricData	cachepops パラメータ	ColdFusion MX
GetK2ServerDocCount	すべて	ColdFusion MX 6.1
GetK2ServerDocCountLimit	すべて	ColdFusion MX 6.1
GetTemplatePath	すべて	ColdFusion MX
IsK2ServerABroker	すべて	ColdFusion MX 6.1
IsK2ServerDocCountExceeded	すべて	ColdFusion MX 6.1

関数	パラメータまたは値	非推奨となった ColdFusion リリース
IsK2ServerOnLine	すべて	ColdFusion MX 6.1
ParameterExists	すべて	ColdFusion MX。IsDefined 関数を使用してください。
SetLocale	locale = "Spanish (Mexican)" 値	ColdFusion MX。Spanish (Standard) を使用してください。

廃止された関数、パラメータ、および値

次の関数、パラメータ、および値は廃止されました。これらを ColdFusion のアプリケーションで使用しないでください。これらは ColdFusion 5 よりも新しいリリースでは動作しません。

関数	パラメータまたは値	これらが廃止となった ColdFusion リリース
AuthenticatedContext	すべて	ColdFusion MX
AuthenticatedUser	すべて	ColdFusion MX
isAuthenticated	すべて	ColdFusion MX
isAuthorized	すべて	ColdFusion MX
isProtected	すべて	ColdFusion MX

関数 a ～ b

Abs

説明

絶対値関数です。数値の絶対値は、その正負符号を除いた数値です。

戻り値

数値の絶対値

カテゴリ

算術関数

関数のシンタックス

```
Abs(number)
```

関連項目

Sgn

パラメータ

パラメータ	説明
number	数値です。

例

```
 <h3>Abs Example</h3>
<p>The absolute value of the following numbers:
1,3,-4,-3.2,6 is
<cfoutput>
#Abs(1)#,#Abs(3)#,#Abs(-4)#,#Abs(-3.2)#,#Abs(6)#
</cfoutput>

<p>The absolute value of a number is the number without its sign.</p>
```

ACos

説明

アークコサイン関数です。アークコサインはコサインが **number** のときの角度です。

戻り値

アークコサイン値 (単位 : ラジアン)

カテゴリ

算術関数

関数のシンタックス

```
ACos(number)
```

関連項目

Cos、Sin、ASin、Tan、Atn、Pi

パラメータ

パラメータ	説明
number	角度のコサインです。この値の範囲は -1.0 ～ 1.0 でなければなりません (-1 と 1 を含む)。

使用方法

結果の範囲は 0 ～ p です。

度数をラジアンに変換するには、度数に p/180 を乗算します。ラジアンを度数に変換するには、ラジアンに 180/p を乗算します。

例

```
 <h3>ACos Example</h3>
<!--- Output the arccosine value. --->
<cfif IsDefined("FORM.CosNum")>
    <cfif IsNumeric(FORM.CosNum)>
        <cfif Abs(FORM.CosNum) LESS THAN OR EQUAL TO 1>
            <cfoutput>ACos(#FORM.CosNum#) = #ACos(FORM.cosNum)# Radians </cfoutput>
            <br>or<br>
            <cfoutput>ACos(#FORM.CosNum#) = #ACos(FORM.cosNum) * 180/PI()#</cfoutput>
        <cfelse>
            <!--- If it is empty, output an error message. --->
            <h4>Enter a number between -1 and 1</h4>
        </cfif>
    </cfif>
</cfif>

<form method="post" action = "acos.cfm">
    <p>Enter a number to get its arccosine in Radians and Degrees.
    <br><input type = "Text" name = "cosNum" size = "25">
    <p><input type = "Submit" name = ""> <input type = "RESET">
</form>
```

AddSOAPRequestHeader

説明

リクエストする前に、SOAP ヘッダを Web サービスリクエストに追加します。

戻り値

なし

カテゴリ

XML 関数

関数のシンタックス

```
AddSOAPRequestHeader(webservice, namespace, name, value [, mustunderstand])
```

関連項目

AddSOAPResponseHeader、GetSOAPRequest、GetSOAPRequestHeader、GetSOAPResponse、GetSOAPResponseHeader、IsSOAPRequest、『ColdFusion アプリケーションの開発』の Basic web service concepts

履歴

ColdFusion MX 7: この関数が追加されました。

パラメータ

パラメータ	説明
webservice	cfobject タグまたは CreateObject 関数から返される Web サービスオブジェクトです。
namespace	ヘッダの名前空間を含む文字列です。

パラメータ	説明
name	リクエスト内の SOAP ヘッダの名前を含む文字列です。
value	SOAP ヘッダの値です。この値は CFML XML 値でもかまいません。
mustunderstand	オプション。値は true または false (デフォルト) です。このヘッダの SOAP mustunderstand 値を設定します。

使用方法

Web サービスのコンシューマの CFML コード内で、Web サービスを呼び出す前に使用されます。

value パラメータで XML を渡す場合、ColdFusion は namespace パラメータと name パラメータを無視します。名前空間が必要な場合は、XML 内で定義してください。

例

この例は 2 つの部分で構成されています。最初の部分は Web サービス CFC です。この関数およびその他の ColdFusion SOAP 関数は、この Web サービス CFC を使用して、Web サービスとのやり取りを示します。この関数で利用される Web サービスを実装する方法については、AddSOAPResponseHeader の例を参照してください。

クライアントとして次の例を実行し、AddSOAPRequestHeader 関数の動作を確認します。

```
 <!--- Note that you might need to modify the URL in the CreateObject function
    to match your server and the location of the headerservice.cfc file if it is
    different than shown here. Likewise for the cfinvoke tag at the end. --->

<h3>AddSOAPRequestHeader Example</h3>
<cfscript>
    // Create the web service object.
    ws = CreateObject("webservice", "http://localhost/soapheaders/headerservice.cfc?WSDL");

    // Set the username header as a string.
    addSOAPRequestHeader(ws, "http://mynamespace/", "username", "tom", false);

    // Set the password header as a CFML XML object.
    doc = XmlNew();
    doc.password = XmlElemNew(doc, "http://mynamespace/", "password");
    doc.password.XmlText = "My Voice is my Password";
    doc.password.XmlAttributes["xsi:type"] = "xsd:string";
    addSOAPRequestHeader(ws, "ignoredNameSpace", "ignoredName", doc);

    // Invoke the web service operation.
    ret = ws.echo_me("argument");

    // Get the first header as an object (string) and as XML.
```

```
    header = getSOAPResponseHeader(ws, "http://www.tomj.org/myns", "returnheader");
    XMLheader = getSOAPResponseHeader(ws, "http://www.tomj.org/myns", "returnheader", true);

    // Get the second header as an object (string) and as XML.
    header2 =getSOAPResponseHeader(ws, "http://www.tomj.org/myns", "returnheader2");
    XMLheader2 = getSOAPResponseHeader(ws, "http://www.tomj.org/myns", "returnheader2", true);
</cfscript>
<hr>
<cfoutput>
    Soap Header value: #HTMLCodeFormat(header)#<br>
    Soap Header XML value: #HTMLCodeFormat(XMLheader)#<br>
    Soap Header 2 value: #HTMLCodeFormat(header2)#<br>
    Soap Header 2 XML value: #HTMLCodeFormat(XMLheader2)#<br>
    Return value: #HTMLCodeFormat(ret)#<br>
</cfoutput>
<hr>

<cfinvoke component="soapheaders.headerservice" method="echo_me" returnvariable="ret" in_here="hi">
</cfinvoke>
<cfoutput>The cfinvoke tag returned: #ret#</cfoutput>
```

AddSOAPResponseHeader

説明

SOAP レスポンスヘッダを Web サービスレスポンスに追加します。リクエストを SOAP サービスとして処理する CFC Web サービス関数内からのみ呼び出します。

戻り値

なし

カテゴリ

XML 関数

関数のシンタックス

```
AddSOAPResponseHeader(namespace, name, value[, mustunderstand])
```

関連項目

AddSOAPRequestHeader、GetSOAPRequest、GetSOAPRequestHeader、GetSOAPResponse、GetSOAPResponseHeader、IsSOAPRequest、『ColdFusion アプリケーションの開発』の Basic web service concepts

履歴

ColdFusion MX 7: この関数が追加されました。

パラメータ

パラメータ	説明
namespace	ヘッダの名前空間を含む文字列です。
name	リクエスト内の SOAP ヘッダの名前を含む文字列です。
value	SOAP ヘッダの値です。この値は CFML XML 値でもかまいません。
mustunderstand	オプション。値は true または false (デフォルト) です。このヘッダの SOAP mustunderstand 値を設定します。

使用方法

この関数は、CFC Web サービス関数内からのみ呼び出します。Web サービスリクエストではないコンテキストで呼び出された場合は、エラーが返されます。

value パラメータで XML を渡す場合、ColdFusion は namespace パラメータと name パラメータを無視します。名前空間が必要な場合は、XML 内で定義してください。

IsSOAPRequest 関数を使用して、CFC が Web サービスとして稼動しているかどうかを判別します。

例

この例では、AddSOAPResponseHeader 関数のオペレーションを示す CFC Web サービスを作成します。また、その他の ColdFusion SOAP 関数のオペレーションを示す Web サービスも提供します。

次のコードを、Web ルート下の "soapheaders" というフォルダに "headerservice.cfc" として保存します。この Web サービスを起動する例を実行することにより、そのオペレーション、特に AddSOAPResponseHeader 関数のオペレーションをテストします。たとえば、AddSOAPRequestHeader の例を参照してください。

```
 <h3>AddSOAPResponseHeader Example</h3>
<!--- The headerservice.cfc CFC Web Service.--->
<cfcomponent displayName="tester" hint="Test for SOAP headers">
<cffunction name="echo_me"
            access="remote"
            output="false"
            returntype="string"
            displayname="Echo Test" hint="Header test">

<cfargument name="in_here" required="true" type="string">

<cfset isSOAP = isSOAPRequest()>
<cfif isSOAP>
    <!--- Get the first header as a string and as XML. --->
    <cfset username = getSOAPRequestHeader("http://mynamespace/", "username")>
    <cfset return = "The service saw username: " & username>
    <cfset xmlusername = getSOAPRequestHeader("http://mynamespace/", "username", "TRUE")>
    <cfset return = return & "<br> as XML: " & xmlusername>

    <!--- Get the second header as a string and as XML. --->
    <cfset password = getSOAPRequestHeader("http://mynamespace/", "password")>
    <cfset return = return & "The service saw password: " & password>
    <cfset xmlpassword = getSOAPRequestHeader("http://mynamespace/", "password", "TRUE")>
    <cfset return = return & "<br> as XML: " & xmlpassword>

    <!--- Add a header as a string. --->
```

```
    <cfset addSOAPResponseHeader("http://www.tomj.org/myns",
        "returnheader", "AUTHORIZED VALUE", false)>

    <!--- Add a second header using a CFML XML value. --->
    <cfset doc = XmlNew()>
    <cfset x = XmlElemNew(doc, "http://www.tomj.org/myns", "returnheader2")>
    <cfset x.XmlText = "hey man, here I am in XML">
    <cfsetx.XmlAttributes["xsi:type"] = "xsd:string">
    <cfset tmp = addSOAPResponseHeader("ignoredNameSpace", "ignoredName", x)>
<cfelse>
    <!--- Add a header as a string - Must generate error!
    <cfset addSOAPResponseHeader("http://www.tomj.org/myns",
        "returnheader", "AUTHORIZED VALUE", false)>
    --->
    <cfset return = "Not invoked as a web service">
</cfif>

<cfreturn return>
</cffunction>
</cfcomponent>
```

AjaxLink

説明
HTML の href 属性によって、リンク先が現在の Ajax コンテナに表示されます。この関数で指定されたリンクがブラウザでクリックされると、HTTP レスポンスによって現在のページが置き換えられるのではなく、cfdiv、cflayoutarea、cfpod、または cfwindow コントロールにリンク先の URL が渡されます。

戻り値
コントロール内にリンク先のページを表示するためのコード

カテゴリ
表示および書式制御関数

関数のシンタックス
```
AjaxLink(URL)
```

関連項目
cfdiv、cflayoutarea、cfpod、cfwindow、『ColdFusion アプリケーションの開発』の Using Ajax User Interface Components and Features

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
URL	リンクの URL です。

使用方法
この関数は、cfdiv、cflayoutarea、cfpod、または cfwindow コントロール内に HTML の a タグがあり、その href 属性の URL がこの関数で指定されている場合にのみ効果を持ちます。そうでない場合、リンクは通常どおりに機能します。

クロスサイトスクリプティングを回避するために、リモートの Web ページはロードされません。

例

```
<cfpod height="600" width="600" name="podTest">
    <a href="<cfoutput>#AjaxLink('HelloWorld.cfm')#</cfoutput>">Click me</a>
</cfpod>
```

AjaxOnLoad

説明

ページをロードするときに、指定された JavaScript 関数を実行します。

戻り値

この関数は値を返しません。

カテゴリ

表示および書式制御関数

関数のシンタックス

```
AjaxOnLoad(functionName)
```

関連項目

cfdiv、cflayoutarea、cfpod、cfwindow、『ColdFusion アプリケーションの開発』の Using Ajax User Interface Components and Features

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
functionName	ページをロードするときに実行する関数の名前です。この関数はパラメータを取りません。

使用方法

この関数を使用すると、ブラウザにページをロードするときに JavaScript 関数が実行されます。JavaScript 関数を使用することで、ページを正しく機能させるために必要な初期化アクションなどを実行できます。たとえば、ログイン状態を確認して必要な場合にのみログインウィンドウを開く JavaScript 関数を AjaxOnLoad 関数で指定すると、ユーザーがまだログインしていない場合にログインウィンドウを表示できます。

この関数は、トップレベルのページ、または cfpod タグおよび cfwindow タグの source 属性によってアプリケーションにダイナミックに挿入されるページで使用できます。

例

次の例では、AjaxOnLoad 関数を使用して、ページがロードされるたびに init 関数を呼び出します。init 関数が呼び出されると、ログインウィンドウが表示されます。

```
 <html>
<head>
<title>Enter Mail Login Details</title>

<script>
init = function() {
    ColdFusion.Window.show('loginwindow');
}
</script>
</head>

<body>
<cfwindow name="loginwindow" title="Enter Login Details"
    draggable="false" closable="false" resizable="false"
    width="450" height="200">
<cfoutput>
<form action="#cgi.script_name#" method="post" name="loginform">
    <table width="400" class="loginTable" cellpadding="5">
        <tr>
            <td style="text-align: right">mail server:</td>
            <td><input type="text" name="server"></td>
        </tr>
        <tr>
            <td style="text-align: right">username:</td>
            <td><input type="text" name="username"></td>
        </tr>
        <tr>
            <td style="text-align: right">password:</td>
            <td><input type="password" name="password"></td>
        </tr>
        <tr>
            <td> </td>
            <td><input type="submit" name="login" value="Login"></td>
        </tr>
    </table>
</form>
</cfoutput>
</cfwindow>

<cfset AjaxOnLoad("init")>
</body>
</html>
```

ApplicationStop

説明
現在のアプリケーションを停止またはリセットします。停止またはリセットされたアプリケーションは、次回のリクエスト時に再起動されます。

戻り値
void

カテゴリ
715 ページの「その他の関数」

関数のシンタックス

```
applicationstop()
```

履歴
ColdFusion 9 で追加されました。

例
```
<!---- get artist by id ---->
<cftry>
...
....
<!---stops the application to reset the artistid, if the artworks for the artist do not exist--->
<cfset applicationstop()/>
<cfset artist = EntityLoad( "artist", url.artistid, True ) />
<cfoutput>
<h2>#artist.getFullName()#</h2>
<table width="400">
    <tr>
        <th>Item</th>
        <th>Price</th>
        <th>Sold</th>
    </tr>
    <cfloop array="#artist.getArt()#" index="index">
        <tr>
            <td>#index.getArtName()#</td>
            <td>#index.getPrice()#</td>
            <td>#index.getIsSold()#</td>
        </tr>
    </cfloop>
</table>
<p><a href="index.cfm">View list</a></p>
</cfoutput>
    <cfcatch type = "any">
        <cfoutput>Artworks for the selected artists are not available</cfoutput>
</cfcatch>
</cftry>
```

ArrayAppend

説明
配列要素を任意の配列に追加します。

戻り値
正常に完了した場合は true

カテゴリ
配列関数

関数のシンタックス
```
ArrayAppend(array, value [,merge])
```

関連項目
ArrayPrepend、『ColdFusion アプリケーションの開発』の Adding elements to an array

履歴
ColdFusion MX: 動作が変更されました。この関数を XML オブジェクトに対して使用できるようになりました。

パラメータ

パラメータ	説明
array	配列の名前です。
arrayAppend	配列要素がソース配列の最後に 1 つの要素として追加されるかどうかを決定するブール値です。デフォルトでは false になるため、以前の動作が維持されます。
value	配列の最後に追加する値です。
merge	true に設定した場合は、ソース配列に、配列要素が個別に追加されます。false（デフォルト）の場合は、ソース配列の最後に、配列全体が 1 つの要素として追加されます。

例

```
 <h3>ArrayAppend Example</h3>
<cfquery name = "GetEmployeeNames" datasource = "cfdocexamples">
    SELECT FirstName, LastName FROM Employees
</cfquery>
<!--- Create an array> --->
<cfset myArray = ArrayNew(1)>
<!--- Set element one to show where we are. --->
<cfset myArray[1] = "Test Value">
<!--- Loop through the query; append these names successively to the last
    element. --->
<cfloop query = "GetEmployeeNames">
    <cfoutput>#ArrayAppend(myArray, "#FirstName# #LastName#")#
    </cfoutput>, Array was appended<br>
</cfloop>
<!--- Show the resulting array as a list. --->
<cfset myList = ArrayToList(myArray, ",")>
<!--- Output the array as a list. --->
<cfoutput>
    <p>The contents of the array are as follows:</p>
    <p>#myList#</p>
</cfoutput>
```

ArrayAvg

説明

任意の配列内にある値の平均を計算します。

戻り値

数値。array パラメータ値が空の配列の場合はゼロを返します。

カテゴリ

配列関数、算術関数

関数のシンタックス

```
ArrayAvg(array)
```

関連項目

ArraySum、『ColdFusion アプリケーションの開発』の Basic array techniques

パラメータ

パラメータ	説明
array	配列の名前です。

使用方法

次の例では、フォームのアクションページで使用可能な ColdFusion のビルトイン変数 Form.fieldNames を使用します。この変数には、フォーム上のフィールドの名前のカンマ区切りリストが含まれます。

例

```
 <!--- This example shows the use of ArrayAvg. --->
<!-- The following lines of code keep track of the form fields that can
be dynamically generated on the screen. It uses the Fieldnames variable
with the ListLen function to determine the number of fields on the form. --->
<cfset FormElem = 2>
    <cfif Isdefined("Form.Submit")>
        <cfif Form.Submit is "More">
            <cfset FormElem = ListLen(Form.Fieldnames)>
        </cfif>
    </cfif>

<html>
<head>
<title>ArrayAvg Example</title>
</head>
<body>
<h3>ArrayAvg Example</h3>
<p> This example uses ArrayAvg to find the average of the numbers that you enter
    into an array.<br>
To enter more than two numbers click the <b>more</b> button.
</p>
<form action = "arrayavg.cfm">
<!--- The following code initially creates two fields. It adds fields if the
    user presses MORE. FormElem is initialized to two at the beginning of this
    code to show that the form has two fields. ----->
<input type = "submit" name = "submit" value = "more">
<table cellspacing = "2" cellpadding = "2" border = "0">
<cfloop index = "LoopItem" from = "1" to = "#FormElem#">
    <tr>
        <cfoutput>
        <th align = "left">Number #LoopItem#</th>
        <td><input type = "text" name = "number#LoopItem#"></td>
        </cfoutput>
    </tr>
</cfloop>
</table>
<input type = "submit" name = "submit" value = "get the average">
</form>

<!--- Create an array. --->
<cfif IsDefined("FORM.submit")>
```

```
    <cfset myNumberArray = ArrayNew(1)>
    <cfset Count = 1>
    <cfloop index = "ListItem" list = "#Form.Fieldnames#">
            <cfif Left(ListItem,3) is "Num">
            <cfset myNumberArray[Count] = Val("number#Count#")>
            <cfset count = IncrementValue(Count)>
        </cfif>
    </cfloop>

    <cfif Form.Submit is "get the average">
        <!--- use ArrayAvg to get the average of the two numbers --->
        <p>The average of the numbers that you entered is
        <cfoutput>#ArrayAvg(myNumberArray)#.</cfoutput>
    <cfelse>
        <cfoutput>Try again. You must enter at least two numeric values.
        </cfoutput>
    </cfif>
</cfif>
</body>
</html>
```

ArrayClear

説明
配列内のデータを削除します。

戻り値
正常に完了した場合は true

カテゴリ
配列関数

関数のシンタックス
```
ArrayClear(array)
```

関連項目
ArrayDeleteAt、『ColdFusion アプリケーションの開発』の Functions for XML object management

履歴
ColdFusion MX: 動作が変更されました。この関数を XML オブジェクトに対して使用できるようになりました。

パラメータ

パラメータ	説明
array	配列の名前です。

例

```
 <h3>ArrayClear Example</h3>
<!--- Create a new array. --->
<cfset MyArray = ArrayNew(1)>
<!--- Populate an element or two. --->
<cfset MyArray[1] = "Test">
<cfset MyArray[2] = "Other Test">
<!--- Output the contents of the array. --->
<p>Your array contents are:
<cfoutput>#ArrayToList(MyArray)#</cfoutput>
<!--- Check to see if the array is empty. --->
<p>Is the array empty?:
<cfoutput>#ArrayIsEmpty(MyArray)#</cfoutput>
<p>Now, clear the array:
<!--- Now clear the array. --->
<cfset Temp = ArrayClear(MyArray)>
<!--- Check to see if the array is empty. --->
<p>Is the array empty?:
<cfoutput>#ArrayIsEmpty(MyArray)#</cfoutput>
```

ArrayContains

説明

指定したオブジェクトが存在するかどうかを配列内で検索します。この関数は、文字列や数値などの単純なオブジェクト、または構造体などの複雑なオブジェクトを検索します。文字列の検索では、大文字と小文字が区別されます。この関数では、COM オブジェクトおよび CORBA オブジェクトの検索はサポートされません。

戻り値

指定したオブジェクトが配列内に存在する場合は、Yes が返されます。

カテゴリ

配列関数

関数のシンタックス

```
#ArrayContains(array,object)
```

関連項目

ArrayFind、ArrayFindNoCase

パラメータ

パラメータ	説明
array	配列の名前です。
object	検索するオブジェクトです。

例

```
<h3>ArrayContains Example</h3>
<h3>2-dimensional array example</h3>
<!--Creating a 2-dimensional array- - >

<cfset dayarray = ArrayNew(2)>
<cfset dayarray[1][1] = "Sunday">
<cfset dayarray[1][2] = "Monday">
<cfset dayarray[1][3] = "Tuesday">
<cfset dayarray[2][1] = "Wednesday">
<cfset dayarray[2][2] = "Thursday">
<cfset dayarray[2][3] = "Friday">

<cfoutput>
    <p>Array contains</p>
    #dayarray[1][1]#, #dayarray[1][2]#, #dayarray[1][3]#, #dayarray[2][1]#,
    #dayarray[2][2]#, #dayarray[2][3]#
    <p>Checking value in the array</p>
    #ArrayContains(dayarray[1], "Tuesday")#</cfoutput>
<!--Creating a one-dimensional array-->
<h3>1-dimensional array example</h3>
<cfset montharray = ArrayNew(1)>
<cfset montharray[1] = "April">
<cfset montharray[2] = "July">
<cfset montharray[3] = "October">
<cfset montharray[4] = "December">
<p>Checking if value exists</p>
<cfoutput>
    #ArrayContains(montharray, "December")#
</cfoutput>
```

ArrayDelete

説明

配列から要素を削除します。COM オブジェクトおよび CORBA オブジェクトはサポートされません。

戻り値

配列のエレメントが正常に削除された場合、Yes が返されます。

カテゴリ

配列関数

関数のシンタックス

```
ArrayDelete(array,object)
```

関連項目

ArrayDeleteAt、ArrayClear

パラメータ

パラメータ	説明
array	配列の名前です。
object	削除対象のオブジェクトです。

例

```
<cfset MyNewArray=ArrayNew(1)>
<cfloop index="i" from="1" to="20" step="1">
<cfset MyNewArray[i] = i*5>
</cfloop>
<cfloop index="i" from="1" to="20" step="1">
<cfoutput># MyNewArray[i] # ,</cfoutput>
</cfloop>
<cfoutput>
<br />
Checkvalue 25:#ArrayContains(MyNewArray,"25")#<br />
Delete value 25:#ArrayDelete(MyNewArray,"25")#<br />
Checkvalue 25:#ArrayContains(MyNewArray,"25")#<br />
</cfoutput>
```

ArrayDeleteAt

説明

配列の指定された位置からエレメントを削除します。

要素が削除されると、ColdFusion によってインデックス位置が再計算されます。たとえば、1 年の月が含まれている配列から 5 番目の要素を削除すると、5 月のエントリが削除されます。この後で 6 月のエントリを削除するには、6 番目ではなく 5 番目の要素を削除することになります。

戻り値

正常に完了した場合は true

カテゴリ

配列関数

関数のシンタックス

```
ArrayDeleteAt(array, position)
```

関連項目

ArrayInsertAt、『ColdFusion アプリケーションの開発』の Functions for XML object management

履歴

ColdFusion MX:

- 動作が変更されました。この関数を XML オブジェクトに対して使用できるようになりました。
- 返す例外が変更されました。InvalidArrayIndexException エラーが返されるようになりました。

パラメータ

パラメータ	説明
array	配列の名前です。
position	配列の位置です。

戻り値

この関数で、位置 0 の要素を削除しようとした場合、または array のサイズよりも大きい値を position に指定した場合、InvalidArrayIndexException エラーが返されます。

例

```
 <h3>ArrayDeleteAt Example</h3><p>
<!--- Create an array. --->
<cfset DaysArray = ArrayNew(2)>
<!--- Populate an element or two. --->
<cfset DaysArray[1][1] = "Monday">
<cfset DaysArray[2][1] = "Tuesday">
<cfset DaysArray[3][1] = "Wednesday">
<cfset DaysArray[1][2] = "April 12">
<cfset DaysArray[2][2] = "April 13">
<cfset DaysArray[3][2] = "April 14">
<p>This is what the array looks like before delete:<br>
<cfoutput>
#DaysArray[1][1]#  #DaysArray[1][2]#<br>
#DaysArray[2][1]#  #DaysArray[2][2]#<br>
#DaysArray[3][1]#  #DaysArray[3][2]#<br>
</cfoutput>

<cfoutput>
We delete this element of the array:<br>
#ArrayDeleteAt(DaysArray,2)#<br>
</cfoutput>
<!--- The formerly third element, "Wednesday" is now the second element. --->
<p>This is what the array looks like after delete:<br>
<cfoutput>
#DaysArray[1][1]#  #DaysArray[1][2]#<br>
#DaysArray[2][1]#  #DaysArray[2][2]#<br>
</cfoutput>
```

ArrayEach

説明
配列をループする場合に使用します。

戻り値
なし

カテゴリ
クロージャ関数

シンタックス
arrayEach(array,function(any currentObj) {});

関連項目
その他のクロージャ関数

履歴
ColdFusion 10: この関数が追加されました。

パラメータ

パラメータ	説明
array	配列オブジェクトの名前です。
function	配列内の各要素に対して実行されるインライン関数です。

ArrayFilter

説明
配列の要素をフィルタリングする場合に使用します。

戻り値
新しい配列

カテゴリ
クロージャ関数

シンタックス
arrayFilter(array,function(***arrayElement***){return true|false;});

関連項目
その他のクロージャ関数

履歴
ColdFusion 10: この関数が追加されました。

パラメータ

パラメータ	説明
array	配列オブジェクトの名前です。
function	配列内の各要素に対して実行されるインライン関数です。結果の配列に配列要素が含まれる場合は true が返されます。
arrayElement	アクセスする配列要素です。

ArrayFind

説明
この関数は、指定されたオブジェクトを配列内で検索します。検索時には大文字と小文字が区別されます。この関数では、文字列や数値などの単純オブジェクト、および構造体などの複合オブジェクトを検索できます。COM オブジェクトおよび CORBA オブジェクトはサポートされません。

戻り値
最初に一致した要素の配列内のインデックス。一致する要素がない場合は 0 が返されます。

カテゴリ
配列関数

関数のシンタックス
```
ArrayFind(array, function(currentObj), object)
```

関連項目
ArrayFindNoCase

パラメータ

パラメータ	説明
array	配列の名前です。
function	配列内の各要素に対して実行されるインライン関数です。配列要素が検索条件に一致する場合は true が返されます。
object	検索するオブジェクトです。

例
```
<cfoutput>
    #ArrayFind(MyArray, 2)#
</cfoutput>
```

ArrayFindAll

説明
配列内の要素を検索する場合に使用します。

戻り値
要素が見つかったインデックス。

カテゴリ
クロージャ関数

シンタックス
ArrayFindAll(array,function(currentObj) {return true|false;});

関連項目
その他のクロージャ関数

履歴
ColdFusion 10: この関数が追加されました。

パラメータ

パラメータ	説明
array	配列オブジェクトの名前です。
function	配列内の各要素に対して実行されるインライン関数です。配列要素が検索条件に一致する場合は true が返されます。
arrayElement	アクセスする配列要素です。

ArrayFindAllNoCase

説明
この関数は、配列内の要素の検索で大文字と小文字を区別した検索を行います。

戻り値
要素が見つかったインデックス。

カテゴリ
クロージャ関数

シンタックス
ArrayFindAllNoCase(array,function(currentObj) {return true|false;});

関連項目
その他のクロージャ関数

履歴
ColdFusion 10: この関数が追加されました。

パラメータ

パラメータ	説明
array	配列オブジェクトの名前です。
function	配列内の各要素に対して実行されるインライン関数です。配列要素が検索条件に一致する場合は true が返されます。
arrayElement	アクセスする配列要素です。

ArrayFindNoCase

説明
この関数は、指定されたオブジェクトを配列内で検索します。検索時には大文字と小文字が区別されません。この関数では、文字列や数値などの単純オブジェクト、および構造体などの複合オブジェクトを検索できます。COM オブジェクトおよび CORBA オブジェクトはサポートされません。

戻り値
最初に一致した要素の配列内のインデックス。一致する要素がない場合は 0 が返されます。

カテゴリ
配列関数

関数のシンタックス
```
ArrayFindNoCase(array, object)
```

関連項目
ArrayFind

パラメータ

パラメータ	説明
array	配列の名前です。
object	検索するオブジェクトです。

例

```
<cfoutput>
    #ArrayFindNoCase(MyArray, 2)#
</cfoutput>
```

ArrayInsertAt

説明

配列に値を挿入します。新しいデータの位置と同じインデックス値、または新しいデータより大きなインデックス値を持つ配列要素のインデックス値は 1 ずつ増加します。配列の長さは 1 増加します。

戻り値

正常に完了した場合は true

カテゴリ

配列関数

関数のシンタックス

```
ArrayInsertAt(array, position, value)
```

関連項目

ArrayDeleteAt、『ColdFusion アプリケーションの開発』の Functions for XML object management

履歴

ColdFusion MX:

- 動作が変更されました。この関数を XML オブジェクトに対して使用できるようになりました。
- 返す例外が変更されました。InvalidArrayIndexException エラーが返されるようになりました。

パラメータ

パラメータ	説明
array	配列の名前です。
position	値を挿入するインデックス位置です。
value	挿入する値です。

使用方法

ArrayInsertAt() 関数を多次元配列に適用するには、array パラメータの最後のインデックスを除くすべてのインデックスを指定する必要があります。次のコード例は myarray[2][4] に要素を挿入します。

```
<cfset ArrayInsertAt(myarray[2], 4, "test")>
```

戻り値
この関数で、位置 0 に要素を挿入しようとした場合、または array のサイズよりも大きい値を position に指定した場合、InvalidArrayIndexException エラーが返されます。

例
```
 <h3>ArrayInsertAt Example</h3><p>
<!--- Create a new array. --->
<cfset DaysArray = ArrayNew(1)>
<!--- Populate an element or two. --->
<cfset DaysArray[1] = "Monday">
<cfset DaysArray[2] = "Tuesday">
<cfset DaysArray[3] = "Thursday">
<!--- Add an element before position 3. --->
<p>Add an element before position 3:
    <cfoutput>#ArrayInsertAt(DaysArray,3,"Wednesday")#</cfoutput>
<p>Now output the array as a list:
<cfoutput>#ArrayToList(DaysArray)#</cfoutput>
<!--- The array now has four elements. Element 3, "Thursday", has become element four. --->
```

ArrayIsDefined

説明
配列要素が定義されているかどうかを調べます。

戻り値
配列要素が定義されている (存在する) 場合は true、定義されていない場合は false

カテゴリ
配列関数

関数のシンタックス
```
ArrayIsDefined(array, elementIndex)
```

関連項目
ArrayIsEmpty

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
array	1 次元配列の場合は、配列名です。多次元配列の場合は、配列名と上位次元のインデックスです。
elementIndex	1 次元配列の場合は、要素のインデックスです。多次元配列の場合は、最終次元の要素のインデックスです。

使用方法
要素のインデックスの値は、配列の長さより小さくなければなりません。

多次元配列の要素が存在するかどうかを調べるには、最初のパラメータで、配列の最後の次元を除くすべての次元を指定します。たとえば、次のコードは要素 MyArray[2][4][1] が存在するかどうかを調べます。

```
ArrayIsDefined(MyArray[2][4], 1)
```

例

```
<h3>ArrayIsDefined Example</h3>
<!--- Create a sparse new array. --->
<cfset MyArray = ArrayNew(1)>
<!--- Populate an element or two. --->
<cfset MyArray[1] = "Test">
<cfset MyArray[3] = "Other Test">

<cfoutput>
    <!--- Display the contents of the array. --->
    <p>Your array contents are:
    <cfdump var="#MyArray#"></p>

    <!--- Check if an existing element is defined. --->
    <p>Does element 3 exist?: 
    #ArrayIsDefined(MyArray, 3)#</p>

    <!--- Check if a non-existent element is defined. --->
    <p>Does element 2 exist? 
    #ArrayIsDefined(MyArray, 2)#
</cfoutput>
```

ArrayIsEmpty

説明

配列のデータ要素が空かどうかを調べます。

戻り値

配列が空の場合は true、空でない場合は false。

カテゴリ

配列関数

関数のシンタックス

```
ArrayIsEmpty(array)
```

関連項目

ArrayIsDefined、ArrayLen、『ColdFusion アプリケーションの開発』の Functions for XML object management

履歴

ColdFusion MX: 動作が変更されました。この関数を XML オブジェクトに対して使用できるようになりました。

パラメータ

パラメータ	説明
array	配列の名前です。

使用方法

多次元配列の上位次元の要素を ArrayIsEmpty 関数で指定すると、その要素が空かどうかを調べることができます。2 次元配列 MyArray の最初の行が空かどうかを調べるには、次の関数を使用します。

```
ArrayIsEmpty(MyArray2[1])
```

例

```
<h3>ArrayIsEmpty Example</h3>
<!--- Create a new array. --->
<cfset MyArray = ArrayNew(1)>
<!--- Populate an element or two. --->
<cfset MyArray[1] = "Test">
<cfset MyArray[2] = "Other Test">
<!--- Output the contents of the array. --->
<p>Your array contents are:
<cfoutput>#ArrayToList(MyArray)#</cfoutput>
<!--- Check to see if the array is empty. --->
<p>Is the array empty?:
<cfoutput>#ArrayIsEmpty(MyArray)#</cfoutput>
<p>Now, clear the array:
<!--- Now clear the array. --->
<cfset Temp = ArrayClear(MyArray)>
<!--- Check to see if the array is empty. --->
<p>Is the array empty?:
<cfoutput>#ArrayIsEmpty(MyArray)#</cfoutput>
```

ArrayLen

説明

配列内の要素の数を調べます。

戻り値

配列内の要素の数

カテゴリ

配列関数

関数のシンタックス

```
ArrayLen(array)
```

関連項目

ArrayIsEmpty、『ColdFusion アプリケーションの開発』の Functions for XML object management

履歴

ColdFusion MX: 動作が変更されました。この関数を子 XML オブジェクトに対して使用できるようになりました。

パラメータ

パラメータ	説明
array	配列の名前です。

例

```
 <h3>ArrayLen Example</h3>
<cfquery name = "GetEmployeeNames" datasource = "cfdocexamples">
    SELECT FirstName, LastName FROM Employees
</cfquery>
<!--- Create an array. --->
<cfset myArray = ArrayNew(1)>
<!--- Set element one to show where we are. --->
<cfset myArray[1] = "Test Value">
<!--- Loop through the query and append these names
    successively to the last element. --->
<cfloop query = "GetEmployeeNames">
    <cfset temp = ArrayAppend(myArray, "#FirstName# #LastName#")>
</cfloop>
<!--- Show the resulting array as a list. --->
<cfset myList = ArrayToList(myArray, ",")>
<!--- Output the array as a list. --->
<cfoutput>
    <p>The contents of the array are as follows:</p>
    <p>#myList#</p>
    <p>This array has #ArrayLen(MyArray)# elements.</p>
</cfoutput>
```

ArrayMax

説明

配列の最大値を得る関数です。

戻り値

配列内の最大の数値。array パラメータ値が空の配列の場合はゼロを返します。

カテゴリ

配列関数

関数のシンタックス

```
ArrayMax(array)
```

パラメータ

パラメータ	説明
array	配列の名前です。

例

```
 <h3>ArrayMax Example</h3>
<p>This example uses ArrayMax to find the largest number in an array.<br> </p>
<!--- After checking whether the form has been submitted, the code creates an array
    and assigns the form fields to the first two elements in the array. ----->
<cfif IsDefined("FORM.submit")>
    <cfset myNumberArray = ArrayNew(1)>
    <cfset myNumberArray[1] = number1>
    <cfset myNumberArray[2] = number2>
    <cfif Form.Submit is "Maximum Value">
        <!--- Use ArrayMax to find the largest number in the array. --->
        <p>The largest number that you entered is
        <cfoutput>#ArrayMax(myNumberArray)#.</cfoutput>
    </cfif>
</cfif>
<!---- The following form provides two numeric fields that are compared when the
    form is submitted. --->
<form action = "arraymax.cfm">
    <input type = "hidden" name = "number1_Float">
    <input type = "hidden" name = "number2_Float">
    <input type = "text" name = "number1"><br>
    <input type = "text" name = "number2"><br>
    <input type = "submit" name = "submit" value = "Maximum Value">
</form>
```

ArrayMin

説明

配列の最小値を得る関数です。

戻り値

配列内の最小の数値。array パラメータ値が空の配列の場合はゼロを返します。

カテゴリ

配列関数

関数のシンタックス

```
ArrayMin(array)
```

パラメータ

パラメータ	説明
array	配列の名前です。

例

```
 <h3>ArrayMin Example</h3>
<p>This example uses ArrayMin to find the smallest number in an array.<br></p>
<!--- After checking whether the form has been submitted, this code creates an
    array and assigns the form fields to the first two elements. ----->
<cfif IsDefined("FORM.submit")>
    <cfset myNumberArray = ArrayNew(1)>
    <cfset myNumberArray[1] = FORM.number1>
    <cfset myNumberArray[2] = FORM.number2>
    <cfif Form.Submit is "Minimum Value">
        <!--- Use ArrayMin to find the smallest number in the array. --->
        <p>The smallest number that you entered is
        <cfoutput>#ArrayMin(myNumberArray)#.</cfoutput>
    </cfif>
</cfif>
<!---- The following form provides two numeric fields that are compared when the form is
    submitted. ----->
<form action = "arraymin.cfm">
    <input type = "hidden" name = "number1_Float">
    <input type = "hidden" name = "number2_Float">
    <input type = "text" name = "number1"><br>
    <input type = "text" name = "number2"><br>
    <input type = "submit" name = "submit" value = "Minimum Value">
</form>
```

ArrayNew

説明

1 ～ 3 次元の配列を作成します。配列要素のインデックス付けには角括弧 ([ ]) を使用します。

ColdFusion 配列は、データが追加されるにしたがってダイナミックに拡大されます。

戻り値

配列

カテゴリ

配列関数

関数のシンタックス

```
ArrayNew(dimension)
```

パラメータ

パラメータ	説明
dimension	新しい配列の次元の数です。1、2、3 のいずれかを指定します。

例

```
 <h3>ArrayNew Example</h3>
<!--- Create an array. --->
<cfset MyNewArray = ArrayNew(1)>
<!--- ArrayToList does not function properly if the Array is not initialized with
    ArraySet --->
<cfset temp = ArraySet(MyNewArray, 1,6, "")>

<!--- Set some elements. --->
<cfset MyNewArray[1] = "Sample Value">
<cfset MyNewArray[3] = "43">
<cfset MyNewArray[6] = "Another Value">

<!--- Is it an array? --->
<cfoutput>
    <p>Is this an array? #IsArray(MyNewArray)#</p>
    <p>It has #ArrayLen(MyNewArray)# elements.</p>
    <p>Contents: #ArrayToList(MyNewArray)#</p>
<!--- The array has expanded dynamically to six elements with the use of ArraySet,
    even though we only set three values. --->
</cfoutput>
```

ArrayPrepend

説明

配列の先頭に配列要素を挿入します。

戻り値

正常に完了した場合は true

カテゴリ

配列関数

関数のシンタックス

```
ArrayPrepend(array, value)
```

関連項目

ArrayAppend、『ColdFusion アプリケーションの開発』の Adding elements to an array

パラメータ

パラメータ	説明
array	配列の名前です。
value	配列の先頭に挿入する値です。

例

```
 <h3>ArrayPrepend Example</h3>
<cfquery name = "GetEmployeeNames" datasource = "cfdocexamples">
    SELECT FirstName, LastName FROM Employees
</cfquery>
<!--- Create an array. --->
<cfset myArray = ArrayNew(1)>
<!--- Set element one to show where we are. --->
<cfset myArray[1] = "Test Value">
<!--- Loop through query. Append names successively before last element.
    (The list reverses itself from the standard queried output, because it keeps
    prepending the array entry.) --->
<cfloop query = "GetEmployeeNames">
    <cfoutput>#ArrayPrepend(myArray, "#FirstName# #LastName#")#
    </cfoutput>, Array was prepended<br>
</cfloop>
<!--- Show the resulting array as a list. --->
<cfset myList = ArrayToList(myArray, ",")>
<!--- Output the array as a list. --->
<cfoutput>
    <p>The contents of the array are as follows:
    <p>#myList#
</cfoutput>
```

ArrayResize

説明

指定された最小要素数に配列を設定し直します。予想される最大の大きさに配列のサイズを設定し直すと、パフォーマンスが向上する場合がありします。要素が 500 個を超える場合には、ArrayNew タグの直後に ArrayResize を使用してください。

ColdFusion 配列は、データが追加されるにしたがってダイナミックに拡大されます。

戻り値

正常に完了した場合は true

カテゴリ

配列関数

関数のシンタックス

```
ArrayResize(array, minimum_size)
```

パラメータ

パラメータ	説明
array	配列の名前です。
minimum_size	配列の最小サイズです。

例

```
 <h3>ArrayResize Example</h3>
<!--- Perform a query to get the list. --->
<cfquery name = "GetCourses" datasource = "cfdocexamples">
    SELECT * FROM Courses
</cfquery>
<!--- Create a new array. --->
<cfset MyArray = ArrayNew(1)>
<!--- Resize that array to the number of records in the query. --->
<cfset temp = ArrayResize(MyArray, GetCourses.RecordCount)>
<cfoutput>
    The array is now #ArrayLen(MyArray)# elements, to match
    the query of #GetCourses.RecordCount# records.
</cfoutput>
```

ArraySet

説明

1 次元配列において、指定されたインデックス範囲内の要素を、指定された値に設定します。ArrayNew を呼び出した後で配列を初期化するために利用できます。

戻り値

正常に完了した場合は true

カテゴリ

配列関数

関数のシンタックス

```
ArraySet(array, start_pos, end_pos, value)
```

関連項目

ArrayNew、『ColdFusion アプリケーションの開発』の Populating arrays with data

履歴

ColdFusion MX: 動作が変更されました。この関数を XML オブジェクトに対して使用できるようになりました。

パラメータ

パラメータ	説明
array	配列の名前です。
start_pos	設定範囲の開始インデックス位置です。
end_pos	設定範囲の終了インデックス位置です。この値が配列の長さを超える場合は、ColdFusion によって配列に要素が追加されます。
value	範囲内の各要素に設定する値です。

例

```
 <h3>ArraySet Example</h3>

<!--- Create an array. --->
<cfset MyNewArray = ArrayNew(1)>
<!--- ArrayToList does not function properly if the Array has not been initialized
    with ArraySet. --->
<cfset temp = ArraySet(MyNewArray, 1,6, "Initial Value")>

<!--- Set some elements. --->
<cfset MyNewArray[1] = "Sample Value">
<cfset MyNewArray[3] = "43">
<cfset MyNewArray[6] = "Another Value">
...
```

ArraySlice

説明

必要な要素のみを持つ配列の一部を返します。

戻り値

offset と length の設定に基づいた配列の一部。

シンタックス

arraySlice(array,offset,[length])

プロパティ

パラメータ	説明
array	抽出を行う配列の名前です。
offset	配列の抽出を行う開始位置を指定します。負の値は、配列の最後から数えた順番を使用して配列の抽出を行うことを示します。
length	offset からの要素の数です。

例

```
<cfscript>
      array = [1, 2, 3, 4, 5, 6, 7, 8];
      newArray = arraySlice(array, 2, 3);//returns 2,3,4
      newArray = arraySlice(array, 4);//returns 4,5,6, 7, 8
      newArray = arraySlice(array, -5, 3);//returns 4,5,6
</cfscript>
```

ArraySort

説明

数字またはアルファベットの順番に配列の要素をソートします。

戻り値

ソートが正常に完了した場合は true、失敗した場合は false

カテゴリ

配列関数、リスト関数

関数のシンタックス

```
ArraySort(array, sort_type[, sort_order])
```

履歴

ColdFusion 10：localeSensitive 属性が追加されました。

ColdFusion MX:

- 返す例外が変更されました。ArraySortSimpleValueException エラーおよび ValueNotNumeric エラーが返されるようになりました。
- ソートした要素を返す順序が変更されました。textnocase の降順でソートした場合、以前のリリースと異なるソート順で要素を返すことがあります。sort_type = "textnocase"、および sort_order = "desc" である場合、大文字と小文字の違いだけがある要素については、ColdFusion と以前のリリースでは次のように処理が異なります。
  - ColdFusion では要素の元の順番が逆になります。
  - ColdFusion MX より前のリリースでは、要素の元の順番は変更されません。

  たとえば、d,a,a,b,A を textnocase および desc の指定でソートすると、次のようになります。
  - ColdFusion MX 以降では、d,b,A,a,a が返されます。
  - ColdFusion MX より前のリリースでは、d,b,a,a,A が返されます。

パラメータ

パラメータ	説明
array	配列の名前です。
localeSensitive	ロケールを考慮したソートを実行するかどうかを指定します。デフォルト値は false です。
sort_type	• numeric: 数値をソートします。 • text: テキストを、大文字と小文字を区別してアルファベット順にソートします。大文字と小文字は分けられ、次のようにソートされます。 - sort_order = "asc" (昇順) の場合は、aabzABZ となります。 - sort_order = "desc" (降順) の場合は、ZBAzbaa となります。 • textnocase: テキストを、大文字と小文字を区別せずにアルファベット順にソートします。この場合、大文字と小文字に関係なく、次のようにアルファベット順にソートされます。 - 昇順では、aAaBbBzzZ のように、同じアルファベットが複数ある場合はそれらの元の順序が保持されます。 - 降順では、ZzzBbBaAa のように、同じアルファベットでの元の順番は逆になります。
sort_order	• asc - 昇順のソートです (デフォルト)。 - 文字の場合、sort_type の値に応じて aabzABZ または aAaBbBzzZ のようにソートされます。 - 数値の場合、小さい数から大きい数の順にソートされます。 • desc - 降順のソートです。 - 文字の場合、sort_type の値に応じて ZBAzbaa または ZzzBbBaAa のようにソートされます。 - 数値の場合、大きい数から小さい数の順にソートされます。

戻り値
配列要素が単純な要素でない場合、この関数では ArraySortSimpleValueException エラーが返されます。配列要素が数値でない場合に sort_type に numeric を指定すると、ValueNotNumeric エラーが返されます。

使用方法
ColdFusion 10 では、すべての Java でサポートされているロケール固有の文字のサポート（ウムラウト文字のサポートなど）が追加されました。このサポートのフラグは、sorttype = "text" または sorttype = "textnocase" に対して追加されました。

例
```
 <!--- This example shows ArraySort. --->
<cfquery name = "GetEmployeeNames" datasource = "cfdocexamples">
    SELECT FirstName, LastName FROM Employees
</cfquery>
<!--- Create an array. --->
<cfset myArray = ArrayNew(1)>
<!--- Loop through the query and append these names successively to the last element. --->
<cfloop query = "GetEmployeeNames">
    <cfset temp = ArrayAppend(myArray, "#FirstName# #LastName#")>
</cfloop>
<!--- Show the resulting array as a list. --->
<cfset myList = ArrayToList(myArray, ",")>
<!--- Sort that array in descending order alphabetically. --->
<cfset isSuccessful = ArraySort(myArray, "textnocase", "desc")>
...
```

ArraySum

説明
配列の SUM 関数です。

戻り値
配列内の値の合計。array パラメータ値が空の配列の場合はゼロを返します。

カテゴリ
配列関数、算術関数

関数のシンタックス
```
ArraySum(array)
```

パラメータ

パラメータ	説明
array	配列の名前です。

例

```
 <h3>ArraySum Example</h3>
<p>This example uses ArraySum to add two numbers.<br> </p>
<!--- After checking whether the form has been submitted, the code creates
    an array and assigns the form fields to the first two elements in the array. --->
<cfif IsDefined("FORM.submit")>
    <cfset myNumberArray = ArrayNew(1)>
    <cfset myNumberArray[1] = number1>
    <cfset myNumberArray[2] = number2>

    <cfif Form.Submit is "Add">
        <!--- Use ArraySum to add the number in the array. --->
        <p>The sum of the numbers is
        <cfoutput>#ArraySum(myNumberArray)#.</cfoutput>
    </cfif>
</cfif>
<!--- This form provides two numeric fields that are added when the form is submitted. --->
<form action = "arraysum.cfm" method="post">
    <input type = "hidden" name = "number1_Float">
    <input type = "hidden" name = "number2_Float">
    <input type = "text" name = "number1">
    <br>
    <input type = "text" name = "number2">
    <br>
    <input type = "submit" name = "submit" value = "Add">
</form>
```

ArraySwap

説明

配列内で、指定された位置にある値を交換します。この関数は、cfset タグを何度も使用するよりも効率的です。

戻り値

正常に完了した場合は true

カテゴリ

配列関数

関数のシンタックス

```
ArraySwap(array, position1, position2)
```

関連項目

『ColdFusion アプリケーションの開発』の Functions for XML object management

パラメータ

パラメータ	説明
array	配列の名前です。
position1	交換する最初の要素の位置です。
position2	交換する 2 番目の要素の位置です。

例

```
 <h3>ArraySwap Example</h3>

<cfset month = ArrayNew(1)>
<cfset month[1] = "February">
<cfset month[2] = "January">
<cfset temp = ArraySwap(month, 1, 2)>
<cfset temp = ArrayToList(month)>

<p>Show the results: <cfoutput>#temp#</cfoutput>
```

ArrayToList

説明

1 次元配列をリストに変換します。

戻り値

文字列の区切りリスト

カテゴリ

配列関数、変換関数、リスト関数

関数のシンタックス

```
 ArrayToList(array[, delimiter])
```

パラメータ

パラメータ	説明
array	配列の名前です。
delimiter	リストの要素を区切る単一の文字または複数の文字からなる文字列です。デフォルト値はカンマ (,) です。

例

```
 <h3>ArrayToList Example</h3>

<cfquery name = "GetEmployeeNames" datasource = "cfdocexamples">
    SELECT FirstName, LastName FROM Employees
</cfquery>

<!--- Create an array. --->
<cfset myArray = ArrayNew(1)>

<!--- Loop through the query, append names successively to the last element. --->
<cfloop query = "GetEmployeeNames">
    <cfset temp = ArrayAppend(myArray, "#FirstName# #LastName#")>
</cfloop>

<!--- Show the resulting array as a list. --->
<cfset myList = ArrayToList(myArray, ",")>

<!--- Sort that array in descending order alphabetically. --->
<cfset myAlphaArray = ArraySort(myArray, "textnocase", "desc")>

<!--- Show the resulting alphabetized array as a list. --->
<cfset myAlphaList = ArrayToList(myArray, ",")>

<!--- Output the array as a list. --->
<cfoutput>
    <p>The contents of the array are as follows:
    <p>#myList#
    <p>This array, alphabetized by first name (descending):
    <p>#myAlphaList#
    <p>This array has #ArrayLen(MyArray)# elements.
</cfoutput>
```

Asc

説明

文字の値を調べます。

戻り値

文字列の最初の文字の値。文字列が空の場合はゼロを返します。

カテゴリ

文字列関数

関数のシンタックス

```
Asc(string)
```

関連項目

Chr

履歴

ColdFusion MX: Unicode サポートが変更されました。Unicode 文字の Java UCS-2 表現の値が 65536 までサポートされるようになりました (以前のリリースでは 1 ～ 255 がサポートされていました)。

パラメータ

パラメータ	説明
string	任意の文字列です。

例

```
 <h3>Asc Example</h3>
<!--- If the character string is not empty, output its ASCII value. --->
<cfif IsDefined("FORM.charVals")>

    <cfif FORM.charVals is not "">
        <cfoutput>#Left(FORM.charVals,1)# =
    #Asc(FORM.charVals)#</cfoutput>
    <cfelse>
<!--- If it is empty, output an error message. --->
        <h4>Enter a character</h4>
    </cfif>
</cfif>

<form action = "asc.cfm" method=post>
    <p>Enter a character to see its ASCII value
    <br><input type = "Text" name = "CharVals" size = "1" maxlength = "1"></p>
    <p><input type = "Submit" name = ""> <input type = "RESET"></p>
</form>
```

ASin

説明

数値のアークサインを求めます。アークサインはサインが **number** のときの角度です。

戻り値

アークサイン値 (単位 : ラジアン)

カテゴリ

算術関数

関数のシンタックス

```
ASin(number)
```

関連項目

Sin、Cos、ACos、Tan、Atn、Pi

パラメータ

パラメータ	説明
number	角度のサインです。この値の範囲は -1 ～ 1 でなければなりません (-1 と 1 を含む)。

使用方法

結果の範囲は、-p/2 ～ p/2 ラジアンです。度数をラジアンに変換するには、度数に p/180 を乗算します。ラジアンを度数に変換するには、ラジアンに 180/p を乗算します。

例

```
 <h3>ASin Example</h3>
<!--- Output its arcsine value. --->
<cfif IsDefined("FORM.SinNum")>
    <cfif IsNumeric(FORM.SinNum)>
        <cfif FORM.SinNum LESS THAN OR EQUAL TO 1>
            <cfif FORM.SinNum GREATER THAN OR EQUAL TO -1>
                ASin(<cfoutput>#FORM.SinNum#</cfoutput>) =
                 <cfoutput>#ASin(FORM.sinNum)# Radians</cfoutput>
                 <br> or <br>ASin(<cfoutput>#FORM.SinNum#</cfoutput>) =
                <cfoutput>
                    #ASin(FORM.sinNum) * 180/Pi()# Degrees
                </cfoutput>
            <cfelse>
<!--- If it is less than negative one, output an error message. --->
                <h4>Enter the sine of the angle to calculate, in degrees and radians.
                The value must be between 1 and -1, inclusive.</h4>
            </cfif>
        <cfelse>
<!--- If it is greater than one, output an error message. --->
        <h4>Enter the sine of the angle to calculate, in degrees and radians. The
value must be between 1 and -1, inclusive.</h4>
        </cfif>
    <cfelse>
<!--- If it is empty, output an error message. --->
        <h4>Enter the sine of the angle to calculate, in degrees and radians. The
value must be between 1 and -1,inclusive.</h4>
    </cfif>
</cfif>
<form action = "./evaltest.cfm" method="post">
<p>Enter a number to get its arcsine in Radians and Degrees.
<br><input type = "Text" name = "SinNum" size = "25">
<p><input type = "Submit" name = ""> <input type = "RESET">
</form>
```

Atn

説明

アークタンジェント関数です。アークタンジェントはタンジェントが **number** のときの角度です。

戻り値

アークタンジェント値 (単位 : ラジアン)

カテゴリ

算術関数

関数のシンタックス

```
Atn(number)
```

関連項目

Sin、ASin、Cos、ACos、Pi

パラメータ

パラメータ	説明
number	角度のタンジェントです。

使用方法

結果の範囲は、-p/2 ～ p/2 ラジアンです。度数をラジアンに変換するには、度数に p/180 を乗算します。ラジアンを度数に変換するには、ラジアンに 180/p を乗算します。

例

```
 <h3>Atn Example</h3>
<!--- Output its Atn value. --->
<cfif IsDefined("FORM.AtnNum")>
    <cfif IsNumeric(FORM.AtnNum)>
        Atn(<cfoutput>#FORM.AtnNum#</cfoutput>) is
    <cfoutput>#Atn(FORM.AtnNum)# radians is #Atn(FORM.AtnNum * 180/PI())# Degrees</cfoutput>
    <cfelse>
<!--- If it is empty, output an error message. --->
        <h4>Enter a number</h4>
    </cfif>
</cfif>
<form action = "evaltest.cfm" method="post">
    <p>Enter a number to get its arctangent in Radians and Degrees
    <br><input type = "Text" name = "atnNum" size = "25"></p>
    <p><input type = "Submit" name = ""> <input type = "RESET"></p>
</form>
```

AuthenticatedContext

説明

この関数は廃止されました。新しいセキュリティツールを使用してください。詳細については、711 ページの「変換関数」および『ColdFusion アプリケーションの開発』の Securing Applications を参照してください。

履歴

ColdFusion MX: この関数は廃止されました。この関数は ColdFusion MX および ColdFusion の今後のリリースでは動作しません。

AuthenticatedUser

説明

この関数は廃止されました。新しいセキュリティツールを使用してください。詳細については、711 ページの「変換関数」および『ColdFusion アプリケーションの開発』の Securing Applications を参照してください。

履歴

ColdFusion MX: この関数は廃止されました。この関数は ColdFusion MX および ColdFusion の今後のリリースでは動作しません。

BinaryDecode

説明
文字列をバイナリオブジェクトに変換します。文字列形式にエンコードされたバイナリデータをバイナリオブジェクトに戻すときに使用します。

戻り値
バイナリオブジェクト

カテゴリ
変換関数、文字列関数

関数のシンタックス
```
BinaryDecode(string, binaryencoding)
```

関連項目
BinaryEncode、CharsetEncode、CharsetDecode

履歴
ColdFusion MX 7: この関数が追加されました。

パラメータ

パラメータ	説明
string	エンコードされたバイナリデータを含む文字列です。
binaryencoding	元のバイナリデータを文字列にエンコードするときに使用したアルゴリズムを指定する文字列です。次のいずれかでなければなりません。 • Hex: 0 ～ 9 および A ～ F の文字で、バイトごとに 16 進数の値を表します。たとえば、3A です。 • UU: データは UNIX UUencode アルゴリズムを使用してエンコードされています。 • Base64: データは www.ietf.org/rfc/rfc2045.txt の IETF RFC 2045 に従い、Base 64 アルゴリズムを使用してエンコードされています。

使用方法
この関数を使用して、バイナリ形式でエンコードされた文字列表記のバイナリデータを、バイナリオブジェクトに変換し、アプリケーションで使用できるようにします。バイナリデータは、HTTP や SMTP などの多くのインターネットプロトコルで転送することやデータベースで保管することが可能な文字列としてエンコードされることがよくあります。

新しく開発するアプリケーションで Base64 エンコードデータをバイナリデータに変換する場合は、ToBinary(**base64data**) 関数ではなく、BinaryDecode 関数を使用することをお勧めします。

バイナリデータの処理の詳細については、次の各ページを参照してください。

- ファイル内のバイナリデータをロードする方法または読み込む方法については、cffile を参照してください。
- バイナリデータをシリアル化またはシリアル化解除する方法については、cfwddx を参照してください。
- 変数がバイナリ形式かどうかをチェックする方法については、IsBinary を参照してください。
- バイナリオブジェクトの長さを調べる方法については、Len を参照してください。

例
次の例では、GIF ファイルをバイナリデータとして読み込み、バイナリ形式でエンコードされた文字列に変換した後に、このエンコードされたデータをバイナリデータに戻し、結果をファイルに書き込みます。エンコードされた文字列およびイメージを出力ファイルに表示します。

```
 <h3>Binary Encoding Conversion Example</h3>

<!--- Do the following if the form has been submitted. --->
<cfif IsDefined("Form.binEncoding")>

    <!--- Read in a binary data file. --->
    <cffile action="readbinary" file="C:\images\help.gif" variable="binimage">

    <!--- Convert the read data to binary encoding and back to binary data. --->
    <cfscript>
        binencode=BinaryEncode(binimage, Form.binEncoding);
        bindecode=BinaryDecode(binencode, Form.binEncoding);
    </cfscript>

    <!---Write the converted results to a file. --->
    <cffile action="write" file="C:\temp\help.gif" output="#bindecode#" addnewline="No" >

    <!--- Display the results. --->
    <cfoutput>
        <p><b>The binary encoding:</b> #Form.binEncoding#</p>

        <p><b>The image converted into a binary-encoded string by BinaryEncode
            </b><br>
            #binencode#</p>
        <p><b>The image as written back to a file after converting back to binary
            using BinaryDecode</b><br>
        <img src="C:\temp\help.gif"><br>
    </cfoutput>
</cfif>

<!--- The input form. --->
<form action="#CGI.SCRIPT_NAME#" method="post">
    <b>Select binary encoding</b><br>
    <select size="1" name="binEncoding" >
        <option selected>UU</option>
        <option>Base64</option>
        <option>Hex</option>
    </select><br>
    <br>
    <input type = "Submit" value = "convert my data">
</form>
```

BinaryEncode

説明
バイナリデータを文字列に変換します。

戻り値
バイナリデータを表すエンコードされた文字列

カテゴリ
変換関数、文字列関数

関数のシンタックス

```
BinaryEncode(binarydata, encoding)
```

関連項目

BinaryDecode、CharsetEncode、CharsetDecode

履歴

ColdFusion MX 7: この関数が追加されました。

パラメータ

パラメータ	説明
binarydata	エンコードするバイナリデータを含む変数です。
encoding	データを表すためのエンコード方法を指定する文字列です。次のいずれかです。 • Hex: 0 ～ 9 および A ～ F の文字を使用して、バイトごとに 16 進数の値を表します。たとえば、3A です。 • UU: UNIX UUencode アルゴリズムを使用してデータを変換します。 • Base64: www.ietf.org/rfc/rfc2045.txt の IETF RFC 2045 に従い、Base 64 アルゴリズムを使用してデータを変換します。

使用方法

バイナリオブジェクトや一部の 8 ビット文字は、HTTP や SMTP などの多くのインターネットプロトコルでは転送することができません。また、一部のデータベースシステムではサポートされないことがあります。データをバイナリエンコードすることにより、任意のインターネットプロトコルで転送することや、文字データとしてデータベースに保管することが可能な形式にデータを変換します。データをバイナリ形式に戻すには、BinaryDecode 関数を使用します。

新しく開発するアプリケーションでバイナリデータを Base64 データに変換する場合は、ToBase64(**binarydata**) 関数ではなく、BinaryEncode 関数を使用することをお勧めします。

この関数は、ToBase64(**binarydata**) 関数の機能を含んでいます。

バイナリデータの処理の詳細については、次の各ページを参照してください。

- バイナリデータをロードする方法または読み込む方法については、cffile を参照してください。
- バイナリデータをシリアル化またはシリアル化解除する方法については、cfwddx を参照してください。
- 変数がバイナリ形式かどうかをチェックする方法については、IsBinary を参照してください。
- バイナリオブジェクトの長さを調べる方法については、Len を参照してください。

例

次の例では、GIF ファイルをバイナリデータとして読み込み、バイナリ形式でエンコードされた文字列に変換した後に、このエンコードされたデータをバイナリデータに戻し、結果をファイルに書き込みます。エンコードされた文字列およびイメージを出力ファイルに表示します。

```
 <h3>Binary Encoding Conversion Example</h3>

<!--- Do the following if the form has been submitted. --->
<cfif IsDefined("Form.binEncoding")>

    <!--- Read in a binary data file. --->
    <cffile action="readbinary"
        file="C:\images\help.gif"
        variable="binimage">

    <!--- Convert the read data to binary encoding and back to binary data. --->
    <cfscript>
        binencode=BinaryEncode(binimage, Form.binEncoding);
        bindecode=BinaryDecode(binencode, Form.binEncoding);
    </cfscript>

    <!---Write the converted results to a file. --->
    <cffile action="write" file="C:\temp\help.gif" output="#bindecode#" addnewline="No" >

    <!--- Display the results. --->
    <cfoutput>
        <p><b>The binary encoding:</b> #Form.binEncoding#</p>

        <p><b>The image converted into a binary-encoded string by BinaryEncode
            </b><br>
            #binencode#</p>
        <p><b>The image as written back to a file after converting back to binary
            using BinaryDecode</b><br>
        <img src="C:\temp\help.gif"><br></p>
    </cfoutput>
</cfif>

<!--- The input form. --->
<form action="#CGI.SCRIPT_NAME#" method="post">
    <b>Select binary encoding</b><br>
    <select size="1" name="binEncoding" >
        <option selected>UU</option>
        <option>Base64</option>
        <option>Hex</option>
    </select><br>
    <br>
    <input type = "Submit" value = "convert my data">
</form>
```

BitAnd

説明

ビットごとの論理積 (AND) 演算を実行します。

戻り値

2 つの整数をビットごとに AND 演算した結果

カテゴリ

算術関数

関数のシンタックス

```
BitAnd(number1, number2)
```

関連項目
BitNot、BitOr、BitXor

パラメータ

パラメータ	説明
number1	32 ビットの符号付き整数です。
number2	32 ビットの符号付き整数です。

使用方法
Bit 関数は、32 ビットの符号付き整数に対して -2147483648 ～ 2147483647 の範囲で演算を行います。

例
```
 <h3>BitAnd Example</h3>

<p>Returns the bitwise AND of two long integers.</p>
<p>BitAnd(5,255): <cfoutput>#BitAnd(5,255)#</cfoutput></p>
<p>BitAnd(5,0): <cfoutput>#BitAnd(5,0)#</cfoutput></p>
<p>BitAnd(128,128): <cfoutput>#BitAnd(128,128)#</cfoutput></p>
```

BitMaskClear

説明
ビットのマスククリア演算を実行します。

戻り値
start から始まる **length** ビットをクリアした数値

カテゴリ
算術関数

関数のシンタックス
```
 BitMaskClear(number, start, length)
```

関連項目
BitMaskRead、BitMaskSet

パラメータ

パラメータ	説明
number	32 ビットの符号付き整数です。
start	0 ～ 31 の範囲の整数です (0 および 31 を含む)。マスクの開始ビットです。
length	0 ～ 31 の範囲の整数です (0 および 31 を含む)。マスクの長さです。

使用方法
Bit 関数は、32 ビットの符号付き整数に対して -2147483648 ～ 2147483647 の範囲で演算を行います。

例

```
 <h3>BitMaskClear Example</h3>

<p>Returns number bitwise cleared with length bits beginning from start.</p>

<p>BitMaskClear(255, 4, 4): <cfoutput>#BitMaskClear(255, 4, 4)#</cfoutput></p>
<p>BitMaskClear(255, 0, 4): <cfoutput>#BitMaskClear(255, 0, 4)#</cfoutput></p>
<p>BitMaskClear(128, 0, 7): <cfoutput>#BitMaskClear(128, 0, 7)#</cfoutput></p>
```

BitMaskRead

説明

ビットのマスク読み込み演算を実行します。

戻り値

number の、**start** から **length** ビット分の値から生成された整数

カテゴリ

算術関数

関数のシンタックス

```
BitMaskRead(number, start, length)
```

関連項目

BitMaskClear、BitMaskSet

パラメータ

パラメータ	説明
number	マスクする 32 ビットの符号付き整数です。
start	0 ～ 31 の範囲の整数です (0 および 31 を含む)。読み込みの開始ビットです。
length	0 ～ 31 の範囲の整数です (0 および 31 を含む)。マスクの長さです。

使用方法

Bit 関数は、32 ビットの符号付き整数に対して -2147483648 ～ 2147483647 の範囲で演算を行います。

例

```
 <h3>BitMaskRead Example</h3>
<p>Returns integer created from <em>length</em> bits of <em>number</em>, beginning
    with <em>start</em>.</p>

<p>BitMaskRead(255, 4, 4): <cfoutput>#BitMaskRead(255, 4, 4)#</cfoutput></p>
<p>BitMaskRead(255, 0, 4): <cfoutput>#BitMaskRead(255, 0, 4)#</cfoutput></p>
<p>BitMaskRead(128, 0, 7): <cfoutput>#BitMaskRead(128, 0, 7)#</cfoutput></p>
```

BitMaskSet

説明

ビットのマスク設定演算を実行します。

戻り値
mask の **length** ビットを使用して **start** からビットをマスクした数値

カテゴリ
算術関数

関数のシンタックス
```
BitMaskSet(number, mask, start, length)
```

関連項目
BitMaskClear、BitMaskRead

パラメータ

パラメータ	説明
number	32 ビットの符号付き整数です。
mask	マスクする 32 ビットの符号付き整数です。
start	0 ～ 31 の範囲の整数です (0 および 31 を含む)。マスクの開始ビットです。
length	0 ～ 31 の範囲の整数です (0 および 31 を含む)。マスクの長さです。

使用方法
Bit 関数は、32 ビットの符号付き整数に対して -2147483648 ～ 2147483647 の範囲で演算を行います。

例
```
 <h3>BitMaskSet Example</h3>
<p>Returns number bitwise masked with length bits of mask beginning at start.</p>

<p>BitMaskSet(255, 255, 4, 4): <cfoutput>#BitMaskSet(255, 255, 4, 4)#</cfoutput></p>
<p>BitMaskSet(255, 0, 4, 4): <cfoutput>#BitMaskSet(255, 0, 4, 4)#</cfoutput></p>
<p>BitMaskSet(0, 15, 4, 4): <cfoutput>#BitMaskSet(0, 15, 4, 4)#</cfoutput></p>
```

BitNot

説明
ビットごとの論理 NOT 演算を実行します。

戻り値
整数のビットごとの NOT 演算の結果

カテゴリ
算術関数

関数のシンタックス
```
BitNot(number)
```

関連項目
BitAnd、BitOr、BitXor

パラメータ

パラメータ	説明
number	32 ビットの符号付き整数です。

使用方法
Bit 関数は、32 ビットの符号付き整数に対して -2147483648 ～ 2147483647 の範囲で演算を行います。

例
```
 <h3>BitNot Example</h3>
<p>Returns the bitwise NOT of a long integer.</p>

<p>BitNot(0): <cfoutput>#BitNot(0)#</cfoutput></p>
<p>BitNot(255): <cfoutput>#BitNot(255)#</cfoutput></p>
```

BitOr

説明
ビットごとの論理和 (OR) 演算を実行します。

戻り値
2 つの整数をビットごとに OR 演算した数値

カテゴリ
算術関数

関数のシンタックス
```
BitOr(number1, number2)
```

関連項目
BitAnd、BitNot、BitXor

パラメータ

パラメータ	説明
number1	32 ビットの符号付き整数です。
number2	32 ビットの符号付き整数です。

使用方法
Bit 関数は、32 ビットの符号付き整数に対して -2147483648 ～ 2147483647 の範囲で演算を行います。

例
```
 <h3>BitOr Example</h3>
<p>Returns the bitwise OR of two long integers.</p>

<p>BitOr(5,255): <cfoutput>#BitOr(5,255)#</cfoutput></p>
<p>BitOr(5,0): <cfoutput>#BitOr(5,0)#</cfoutput></p>
<p>BitOr(7,8): <cfoutput>#BitOr(7,8)#</cfoutput></p>
```

BitSHLN

説明
ビットの左シフト演算を循環なしで実行します。

戻り値
count で指定されたビット分左へ循環なしでシフトした整数

カテゴリ
算術関数

関数のシンタックス
```
BitSHLN(number, count)
```

関連項目
BitSHRN

パラメータ

パラメータ	説明
number	32 ビットの符号付き整数です。
count	0 ～ 31 の整数です (0 および 31 を含む)。シフトするビット数を指定します。

使用方法
Bit 関数は、32 ビットの符号付き整数に対して -2147483648 ～ 2147483647 の範囲で演算を行います。

例
```
 <h3>BitSHLN Example</h3>
<p>Returns the number, bitwise shifted, without rotation, to the left by count bits.</p>

<p>BitSHLN(1,1): <cfoutput>#BitSHLN(1,1)#</cfoutput></p>
<p>BitSHLN(1,30): <cfoutput>#BitSHLN(1,30)#</cfoutput></p>
<p>BitSHLN(1,31): <cfoutput>#BitSHLN(1,31)#</cfoutput></p>
<p>BitSHLN(2,31): <cfoutput>#BitSHLN(2,31)#</cfoutput></p>
```

BitSHRN

説明
ビットの右シフト演算を循環なしで実行します。

戻り値
count で指定されたビット分右へ循環なしでシフトした整数

カテゴリ
算術関数

関数のシンタックス
```
BitSHRN(number, count)
```

関連項目
BitSHLN

パラメータ

パラメータ	説明
number	32 ビットの符号付き整数です。
count	0 ～ 31 の範囲の整数です (0 および 31 を含む)。シフトするビット数を指定します。

使用方法
Bit 関数は、32 ビットの符号付き整数に対して -2147483648 ～ 2147483647 の範囲で演算を行います。

例
```
 <h3>BitSHRN Example</h3>
<p>Returns a number, bitwise shifted, without rotation, to the right, by count bits.</p>

<p>BitSHRN(1,1): <cfoutput>#BitSHRN(1,1)#</cfoutput></p>
<p>BitSHRN(255,7): <cfoutput>#BitSHRN(255,7)#</cfoutput></p>
<p>BitSHRN(-2147483548,1): <cfoutput>#BitSHRN(-2147483548,1)#</cfoutput></p>
```

BitXor

説明
ビットごとの論理 XOR 演算を実行します。

戻り値
2 つの整数をビットごとに XOR 演算した値

カテゴリ
算術関数

関数のシンタックス
```
BitXor(number1, number2)
```

関連項目
BitAnd、BitNot、BitOr

パラメータ

パラメータ	説明
number1	32 ビットの符号付き整数です。
number2	32 ビットの符号付き整数です。

使用方法
Bit 関数は、32 ビットの符号付き整数に対して -2147483648 ～ 2147483647 の範囲で演算を行います。

例

```
 <h3>BitXOr Example</h3>
<p>Returns the bitwise XOR of two long integers.</p>

<p>BitXOr(5,255): <cfoutput>#BitXOr(5,255)#</cfoutput></p>
<p>BitXOr(5,0): <cfoutput>#BitXOr(5,0)#</cfoutput></p>
<p>BitXOr(128,128): <cfoutput>#BitXOr(128,128)#</cfoutput></p>
```

関数 c ～ d

CacheIdExists

説明

キャッシュ領域にキャッシュされたオブジェクトが存在するかどうかを調べるために使用します。領域は、（サーバーレベルまたはアプリケーションレベルの）デフォルトのキャッシュ領域、または指定したカスタム領域です。

戻り値

指定したキャッシュ領域にキャッシュされたオブジェクトが存在する場合は true

シンタックス

cacheIdExists(***id*** [, ***region***])

プロパティ

パラメータ	説明
id	（必須）キャッシュされたオブジェクトの ID です。
region	（オプション）キャッシュされたオブジェクトをチェックするキャッシュ領域です。

例：ユーザー定義キャッシュ領域にキャッシュオブジェクトが存在するかどうかをチェックします。

```
 <!--- Creating a new object '
        <cfset obj1 = structNew()>
        <cfset obj1.name = "xyz">

            <!---Defining the time to live and time to Idle parameters --'
        <cfset timeToLive=createtimespan(0,0,0,30)>
        <cfset timeToIdle=createtimespan(0,0,0,30)>

        <cfoutput>Starting to write to cache..</cfoutput>
           <cfset cachePut("obj1",obj1,timeToLive,timeToIdle,"customcache")>
           <br/>
        <cfoutput>Done!!</cfoutput>

       <cfoutput>Trying to check if the cached item is present...</cfoutput>
              <cfoutput>#cacheIdExists("obj1","customcache")#</cfoutput>
```

例：デフォルトのキャッシュ領域にキャッシュオブジェクトが存在するかどうかをチェックします。

```
<cfset obj2 = structNew()>
        <cfset obj2.name = "xyz">
        <cfoutput>Starting to write to cache..</cfoutput>
        <cfset cachePut("obj2",obj2)>
        <br/>
        <cfoutput>Done!!</cfoutput>

        <cfoutput>Trying to fetch cached item...</cfoutput>
        <cfset obj = cacheGet("obj2")>
        <cfoutput>#cacheIdExists("obj2","OBJECT")#</cfoutput>
```

CacheGet

説明
キャッシュに保管されているオブジェクトを取得します。

戻り値
キャッシュに保管されているオブジェクト

カテゴリ
キャッシュ関数

関数のシンタックス
```
CacheGet(id, [region])
```

関連項目
cfcache、CachePut、CacheGetAllIds、CacheGetMetadata、CacheGetProperties、CacheRemove、CacheSetProperties

履歴
ColdFusion 10：region パラメーターが追加されました。

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
id	キャッシュオブジェクトの作成時に割り当てられた ID です。
region	（オプション）キャッシュオブジェクトを配置可能なキャッシュ領域を指定します。

CacheGetAllIds

説明
キャッシュに保管されているすべてのオブジェクトの ID を取得します。

戻り値
キャッシュに保管されているすべてのオブジェクトの ID が格納された配列

カテゴリ
キャッシュ関数

関数のシンタックス
```
CacheGetAllIds()
```

パラメータ

パラメータ	説明
region	（オプション）キャッシュ領域の名前です。

関連項目
cfcache、CachePut、CacheGet、CacheGetMetadata、CacheGetProperties、CacheRemove、CacheSetProperties

履歴
ColdFusion 10：region 属性が追加されました。

ColdFusion 9: この関数が追加されました。

CacheGetMetadata

説明
キャッシュされたオブジェクトおよびテンプレートキャッシュのメタデータの値を取得します。

戻り値
キャッシュされたオブジェクトのメタデータが格納された構造体。この構造体には次のフィールドがあります。

構造体の要素	説明
cache_hitcount	キャッシュエントリが使用された回数です。
cache_misscount	オブジェクトがリクエストされたものの、そのオブジェクトがキャッシュ内に見つからないか、またはキャッシュされているオブジェクトが最新ではなかった (古かった) 回数です。
createdtime	要素またはオブジェクトがキャッシュされた時刻です。
hitcount	キャッシュされたオブジェクトが使用された回数です。
idletime	キャッシュされたオブジェクトが使用されないままあと何秒が経過すると消去されるかを示す秒数です。
lasthit	キャッシュされたオブジェクトが最後に使用された時刻を示す日付時刻オブジェクト、または空の文字列です。
lastupdated	キャッシュされたオブジェクトが最後に変更された時刻を示す日付時刻オブジェクト、または空の文字列です。
size	オブジェクトをシリアル化するときに必要なバイト数です。
timespan	キャッシュされたオブジェクトが有効である残り時間を示す秒数です。

カテゴリ
キャッシュ関数

関数のシンタックス
```
CacheGetMetadata(id, template)
```

関連項目

cfcache、CachePut、CacheGet、CacheGetAllIds、CacheGetProperties、CacheRemove、CacheSetProperties

履歴

ColdFusion 10：region 属性が追加されました。

ColdFusion 9.0.1：template パラメーターが追加されました。

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
id	キャッシュされたオブジェクトの ID です。
region	（オプション）キャッシュ領域の名前です。
template	テンプレートキャッシュのメタデータを取得します。

例

cacheTemp.cfm

```
<cfcache action="cache" timespan="#CreateTimeSpan(0,0,10,0)#" useQueryString="true"
metadata="cachemetadata">
<cfoutput>-This date/time IS cached: #Now()#-<br /></cfoutput>
</cfcache>
```

getMetaDataTemplate.cfm

```
<cfcache action="flush">
<!--- contruct the templatecacheid to pass to the getcachemetadata function --->
<cfset baseurl =
"http://#cgi.server_name#:#cgi.server_port##getDirectoryFromPath(cgi.script_name)#cacheTemp.cfm">
<cfset pageid = hash(expandpath('cacheTemp.cfm'))>
<cfset lineno = 1>
<cfset templateCacheId = trim(ucase("#baseurl#_PAGEID:#pageid#_LINE:#lineno#"))>
<!--- cache the template --->
<cfhttp url="#baseurl#" />
<cfdump var="#getAllTemplateCacheIds()#"><br>
<!--- work with the returned metadata --->
<cfset templateMetaData = cachegetmetadata(templatecacheid,'Template')>
<cfdump var="#templateMetaData#">
```

CacheGetProperties

説明

オブジェクトキャッシュ、ページキャッシュ、またはその両方のキャッシュプロパティを取得します。取得される情報はアプリケーション固有です。

戻り値

キャッシュプロパティが格納された構造体の配列。各構造体には、キャッシュタイプ (オブジェクトまたはページ) ごとのプロパティが格納されます。パラメータでいずれかのタイプを指定した場合は、配列には構造体のエントリが 1 つだけ含まれます。各構造体には次のフィールドがあります。

構造体の要素	説明
diskpersistent	JVM を再起動した場合でも、ディスク上に保存されたキャッシュを継続して使用するかどうかを指定するブール値です。
eternal	タイムアウトやアイドル時間を適用しないかどうかを指定するブール値です。この値が true の場合は、期間を指定しないでオブジェクトまたはページがキャッシュされることを示します。
maxelementsinmemory	メモリにキャッシュ可能なオブジェクトの最大数です。オブジェクトの数がこの数よりも多くなり、overflowtodisk が false の場合は、memoryevictionpolicy エントリに指定されたアルゴリズムを使用して、古い要素が新たなオブジェクトに置換されます。
maxelementsondisk	overfllowtodisk が true の場合にディスク上に保存可能なオブジェクトの最大数です。
memoryevictionpolicy	LRU (Least Recently Used: 使用時期が最も古い) や LFU (Least Frequently Used: 使用頻度が最も低い) など、オブジェクトの数が最大数に達した場合に古いエントリを消去するために使用されるアルゴリズムです。
objecttype	キャッシュタイプで、object と template のいずれかです。
overflowtodisk	メモリ内に保存できる要素の最大数に達したときに、memoryevictionpolicy の値に従ってオブジェクトをディスクに移動するかどうかを指定するブール値です。
statistics	true は、Ehcache の統計の収集が有効になっていることを示します。
timetoidoleconds	アイドル時間を示す秒数です。cfcache タグに idleTime 属性が指定されていない場合に使用されます。
timetolivesecond	タイムアウト時間を示す秒数です。cfcache タグに timespan 属性が指定されていない場合に使用されます。

cacheGetProperties 関数では、ehCache 構造体の次の要素もサポートされます。

- cacheloadertimeoutmillis
- clearonflush
- copyonread
- copyonwrite
- diskaccessstripes
- diskexpirythreadintervalseconds
- diskpersistent
- diskspoolbuffersizemb
- eternal
- logging
- maxbyteslocaldisk
- maxbyteslocaldiskasstring
- maxbyteslocaldiskpercentageset
- maxbyteslocalheap
- maxbyteslocalheapasstring
- maxbyteslocalheappercentageset
- maxbyteslocaloffheap
- maxbyteslocaloffheapasstring
- maxbyteslocaloffheappercentageset
- maxelementsinmemory
- maxelementsondisk

- maxentrieslocaldisk
- maxentrieslocalheap
- maxmemoryoffheap
- maxmemoryoffheapinbytes
- memoryevictionpolicy
- name
- objecttype
- overflowtodisk
- overflowtooffheap
- overflowtooffheapset
- statistics
- timetoidleseconds
- timetoliveseconds

前述の構造体の要素について詳しくは、EhCache Documentation を参照してください。

カテゴリ

キャッシュ関数

関数のシンタックス

```
CacheGetProperties([type])
```

関連項目

cfcache、CachePut、CacheGet、CacheGetAllIds、CacheGetMetadata、CacheRemove、CacheSetProperties

履歴

ColdFusion 10：region 属性が追加されました。

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
type	(オプション) キャッシュタイプです。 • template - ページキャッシュのプロパティを取得します。このページキャッシュにはキャッシュされたページおよびページセグメントが含まれます。 • object - オブジェクトキャッシュのプロパティを取得します。 • パラメータなし - 両方のキャッシュタイプのプロパティを取得します。
region	（オプション）キャッシュ領域の名前です。

cacheGetSession

説明
基盤となるキャッシュオブジェクトを取得し、cfcache タグでは実装されていない追加のキャッシュ機能にアクセスできます。

注意：cacheGetSession 関数を使用すると、セキュリティの脆弱性が生じる可能性があることに注意してください。この関数の使用を無効にする場合は、サンドボックスセキュリティにこの関数を追加します。詳しくは、Using sandbox security を参照してください。

戻り値
基盤となるキャッシュオブジェクト

シンタックス
cacheGetSession()

パラメータ

パラメータ	説明
objectType	次のいずれかの値です。 • object • template • ユーザー定義キャッシュの名前
isKey	objectType がユーザー定義キャッシュの場合は、true に設定します。デフォルト値は false です。

カテゴリ
キャッシュ関数

関連項目
cfcache、CachePut、CacheGet、CacheGetAllIds、CacheGetMetadata、CacheRemove、CacheSetProperties

履歴
ColdFusion 9.0.1: この関数が追加されました。

例 1
次の例は、ehCache.xml にエントリを追加して、ユーザー定義キャッシュを作成する方法を示しています。

```
<cache
name="customcache"
maxElementsInMemory="1000"
eternal="false"
timeToIdleSeconds="720"
timeToLiveSeconds="720"
overflowToDisk="true"
diskSpoolBufferSizeMB="10"
maxElementsOnDisk="100000"
diskPersistent="true"
diskExpiryThreadIntervalSeconds="3600"
memoryStoreEvictionPolicy="LRU"/>
```

ehCache.xml に詳細を指定した後は、ユーザー定義キャッシュを使用できます。次に例を示します。

```
<!--- put an object into user-defined object cache --->
     <cfset cachePut("cache1","hello",15,15,customCache)>

     <!--- get underlying user-defined object cache --->
     <cfset objectCache = cachegetsession(customCache,true)>

     <!--- get/print user-defined object cache properties --->
     <cfset config = objectCache.getCacheConfiguration()>
     <cfoutput>
           getMaxElementsInMemory() :: #config.getMaxElementsInMemory()#<br>
           isEternal() ::    #config.isEternal()#<br>
           getTimeToIdleSeconds() :: #config.getTimeToIdleSeconds()#<br>
           getTimeToLiveSeconds() :: #config.getTimeToLiveSeconds()#<br>
           isOverflowToDisk() :: #config.isOverflowToDisk()#<br>
           getDiskSpoolBufferSizeMB() :: #config.getDiskSpoolBufferSizeMB()#<br>
           getMaxElementsOnDisk() :: #config.getMaxElementsOnDisk()#<br>
           isDiskPersistent() :: #config.isDiskPersistent()#<br>
           getDiskExpiryThreadIntervalSeconds() :: #config.getDiskExpiryThreadIntervalSeconds()#<br>
           getMemoryStoreEvictionPolicy() :: #config.getMemoryStoreEvictionPolicy()#<br>
           isClearOnFlush() :: #config.isClearOnFlush()#<br>
     </cfoutput>
```

例 2
次の例は、cacheGetSession 関数を使用して、デフォルトのキャッシュで処理を行う方法を示しています。

```
<!--- put an object into user-defined object cache --->
     <cfset cachePut("cache1","hello",15,15)>

     <!--- get underlying user-defined object cache --->
     <cfset objectCache = cachegetsession("object",true)>

     <!--- get/print user-defined object cache properties --->
     <cfset config = objectCache.getCacheConfiguration()>
```

CachePut

説明
オブジェクトをキャッシュに保存します。

戻り値
なし

カテゴリ
キャッシュ関数

関数のシンタックス

```
CachePut(id, value, [timeSpan], [idleTime], [region], [throwOnError])
```

関連項目
cfcache、CacheGet、CacheGetAllIds、CacheGetProperties、CacheRemove、CacheSetProperties

履歴
ColdFusion 10：region パラメーターと throwOnError パラメーターが追加されました。

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
id	キャッシュオブジェクトの ID です。
value	オブジェクトの値です。ColdFusion によってサポートされている任意のデータ型を指定できます。
timeSpan	(オプション) オブジェクトがキャッシュから消去されるまでの期間を示す、小数で表された日数です。CreateTimeSpan 関数からの戻り値を使用してこの値を設定できます。デフォルトでは、オブジェクトのタイムアウトは設定されません。
idleTime	(オプション) オブジェクトへのアクセスが一定期間ない場合にそのオブジェクトをキャッシュから消去するための期間を示す、小数で表された日数です。CreateTimeSpan 関数からの戻り値を使用してこの値を設定できます。
region	オプション。キャッシュオブジェクトを配置可能なキャッシュ領域を指定します。
throwOnError	オプション。True の場合、regen が存在しない場合はエラーが発生します。

CacheRegionExists

説明
キャッシュ領域が存在するかどうかをチェックします。

戻り値
キャッシュ領域が存在する場合は true

シンタックス
cacheRegionExists(***region***)

プロパティ

パラメータ	説明
region	キャッシュ領域の名前です。

例

```
<!--- Checking if the region is present in the Cache --->
       <cfif #cacheRegionExists("testregion")# EQ "YES">
              <cfset cacheRegionRemove("testregion")>
              <cfif #cacheRegionExists("testregion")# EQ "NO">
                     <cfoutput>Region is deleted<br></cfoutput>
              </cfif>
        <cfelse>
              <cfset cacheRegionNew("testregion")>
                    <cfset cacheRegionRemove("testregion")>
             <cfif #cacheRegionExists("testregion")# EQ "NO">
                    <cfoutput>Region is deleted<br></cfoutput>
            </cfif>
        </cfif>
```

CacheRegionNew

説明
新しいカスタムキャッシュ領域を作成します（キャッシュ領域が存在しない場合）。

戻り値
throwOnError パラメーターを true に設定した場合に、キャッシュ領域が既に存在する場合はエラー

シンタックス
cacheRegionNew(***region***,[properties],[throwOnError])

プロパティ

パラメータ	説明
region	作成する新しいキャッシュ領域の名前です。
properties	オプション。キャッシュ領域のプロパティが含まれる構造体です。
throwOnError	オプション。指定したキャッシュ領域名が既に存在する場合に例外を返すかどうかを指定するブール値です。デフォルト値は true です。

例
```
<!--- Defining properties for the struct --->
            <cfset defaultCacheProps = StructNew()>
              <cfset defaultCacheProps.CLEARONFLUSH = "true">
              <cfset defaultCacheProps.DISKEXPIRYTHREADINTERVALSECONDS = "3600">
              <cfset defaultCacheProps.DISKPERSISTENT = "false">
              <cfset defaultCacheProps.DISKSPOOLBUFFERSIZEMB = "30">
              <cfset defaultCacheProps.ETERNAL = "false">
              <cfset defaultCacheProps.MAXELEMENTSINMEMORY = "5">
              <cfset defaultCacheProps.MAXELEMENTSONDISK = "10">
              <cfset defaultCacheProps.MEMORYEVICTIONPOLICY = "LRU">
              <cfset defaultCacheProps.OBJECTTYPE = "OBJECT">
              <cfset defaultCacheProps.OVERFLOWTODISK = "true">
              <cfset defaultCacheProps.TIMETOLIVESECONDS = "5">
<cfset defaultCacheProps.TIMETOIDLESECONDS = "30">
        <cfset cacheRegionNew("testregion",#defaultCacheProps#,false)>
        <!--- Defining a struct object --->
        <cfset obj1 = structNew()>
        <cfset obj1.name = "xyz">
              <cfset timeToLive = CreateTimeSpan(0,0,5,0)>
        <cfset timeToIdle = CreateTimeSpan(0,0,10,0)>
        <!--- Putting Cache in the USD specific cache --->
        <cfoutput>Starting to write to cache..</cfoutput>
        <cfset cachePut("obj1",obj1,timetoLive,timeToIdle,"testregion")>

        <cfoutput>Trying to fetch cached item...</cfoutput>
        <cfset obj = cacheGet("obj1","testregion")>
              <br/>
        <cfoutput>Done!!<br></cfoutput>
         <cfoutput>#obj.name#</cfoutput>
```

CacheRegionRemove

説明
指定したキャッシュ領域を削除します。

戻り値
なし

シンタックス
cacheRegionRemove(***region***)

プロパティ

パラメータ	説明
region	削除するキャッシュ領域の名前です。

例
781 ページの「CacheRegionExists」の例を参照してください。

CacheRemove

説明
オブジェクトをキャッシュから削除します。

戻り値
なし

カテゴリ
キャッシュ関数

関数のシンタックス
```
CacheRemove(ids, [throwOnError])
```

関連項目
cfcache、CacheGet、CachePut、CacheGetAllIds、CacheGetProperties、CacheGetMetadata、CacheSetProperties

履歴
ColdFusion 10：region 属性と exact 属性が追加されました。

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
exact	（オプション）true の場合、検索対象を（削除の）ID に完全に一致する値に限定します。デフォルト値は true です。
ids	削除するキャッシュされたオブジェクトの ID を指定するカンマ区切りの文字列です。
region	（オプション）削除対象のキャッシュされたオブジェクトが存在するキャッシュ領域の名前です。
throwOnError	指定された ID がキャッシュされた要素の ID でない場合に例外を返すかどうかを指定するブール値です。デフォルト値は false です。

exact パラメーターを true に設定する場合の例

```
 <!--- clear all object caches  --->
             <cfif ArrayLen(cacheGetAllIds()) gt 0>
                     <cfset cacheRemove(ArrayToList(cacheGetAllIds()))>
              </cfif>

              <!--- create few caches --->
              <cfloop from="1" to="10" index="i">
              <cfset id = "cache_#i#">
            <cfset timeToLive = CreateTimeSpan(0,0,30,0)>
            <cfset timeToIdle = CreateTimeSpan(0,0,30,0)>
                     <cfset cachePut(id,createQryObj(i),timeToLive,timeToIdle)>
            </cfloop>
              <cfoutput>Before cacheRemove() :: Number of objects in the cache:
#ArrayLen(cacheGetAllIds())#<br></cfoutput>

              <!--- clear all objects from the cache --->
              <cfset cacheRemove("cache_1",true,"object",true)>

              <cfoutput>(List of cache Ids -
#ListSort(ArrayToList(cacheGetAllIds()),"textnocase","ASC")#)</cfoutput><br>
              <cfoutput>After cacheRemove() :: Number of objects in the cache:
#ArrayLen(cacheGetAllIds())#<br><br></cfoutput>
```

exact パラメーターを false に設定する場合の例

```
 <!--- create few caches --->
              <cfloop from="1" to="10" index="i">
              <cfset id = "cache_#i#">
              <cfset timeToLive = CreateTimeSpan(0,0,30,0)>
              <cfset timeToIdle = CreateTimeSpan(0,0,30,0)>
                     <cfset cachePut(id,createQryObj(i),timeToLive,timeToIdle)>
              </cfloop>
              <cfoutput>Before cacheRemove() :: Number of objects in the cache:
#ArrayLen(cacheGetAllIds())#<br></cfoutput>

              <!--- clear all objects from the cache --->
              <cfset cacheRemove("cache",true,"object",false)>

              <cfoutput>(List of cache Ids -
#ListSort(ArrayToList(cacheGetAllIds()),"textnocase","ASC")#)</cfoutput><br>
<cfoutput>After cacheRemove() :: Number of objects in the cache:
#ArrayLen(cacheGetAllIds())#<br><br></cfoutput>
```

CacheRemoveAll

説明

キャッシュ領域に保管されているすべてのオブジェクトを削除します。キャッシュ領域を指定しない場合は、デフォルトの領域のオブジェクトが削除されます。

戻り値

なし

シンタックス

cacheRemoveAll(***region***)

プロパティ

パラメータ	説明
region	（オプション）削除対象の保管されているオブジェクトが存在するキャッシュ領域を指定します。値を指定しない場合、デフォルトのキャッシュ領域がデフォルトで対象と見なされます。

例

```
<!--- clear all object caches  --->
              <cfif ArrayLen(cacheGetAllIds()) gt 0>
                     <cfset cacheRemove(ArrayToList(cacheGetAllIds()))>
              </cfif>

              <!--- create few caches --->
              <cfloop from="1" to="10" index="i">
              <cfset id = "cache_#i#">
            <cfset timeToLive = CreateTimeSpan(0,0,30,0)>
            <cfset timeToIdle = CreateTimeSpan(0,0,30,0)>
                     <cfset cachePut(id,createQryObj(i),timeToLive,timeToIdle)>
             </cfloop>
              <cfoutput>Before cacheRemove() :: Number of objects in the cache:
#ArrayLen(cacheGetAllIds())#<br></cfoutput>

              <!--- clear all objects from the cache --->
              <cfset cacheRemoveAll()>

              <cfoutput>After cacheRemove() :: Number of objects in the cache:
#ArrayLen(cacheGetAllIds())#<br><br></cfoutput>
```

CacheSetProperties

説明

オブジェクトキャッシュ、ページキャッシュ、またはその両方のキャッシュプロパティを設定します。キャッシュがまだ存在しない場合には、キャッシュを作成します。キャッシュおよびプロパティはアプリケーション固有です。

戻り値

なし

カテゴリ

キャッシュ関数

関数のシンタックス

```
CacheSetProperties(propStruct)
```

関連項目

cfcache、CacheGet、CachePut、CacheGetAllIds、CacheGetProperties、CacheGetMetadata、CacheRemove

履歴

ColdFusion 10：region 属性が追加されました。

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
propsSruct	キャッシュプロパティと識別情報を指定する構造体です。
region	（オプション）キャッシュ領域の名前です。

propsStruct 構造体では、次のフィールドの一部またはすべてを指定できます。

構造体の要素	説明
diskstore	ディスクストアです。
diskpersistent	JVM を再起動した場合でも、ディスク上に保存されたキャッシュを継続して使用するかどうかを指定するブール値です。
eternal	タイムアウトやアイドル時間を適用しないかどうかを指定するブール値です。この値が true の場合は、期間を指定しないでオブジェクトまたはページがキャッシュされることを示します。
maxelementsinmemory	メモリにキャッシュ可能なオブジェクトの最大数です。オブジェクトの数がこの数よりも多くなり、overflowtodisk が false の場合は、memoryevictionpolicy エントリに指定されたアルゴリズムを使用して、古い要素が新たなオブジェクトに置換されます。
maxelementsondisk	overfllowtodisk が true の場合にディスク上に保存可能なオブジェクトの最大数です。
memoryevictionpolicy	LRU (Least Recently Used: 使用時期が最も古い) や LFU (Least Frequently Used: 使用頻度が最も低い) など、オブジェクトの数が最大数に達した場合に古いエントリを消去するために使用されるアルゴリズムです。
objecttype	キャッシュタイプで、object と template のいずれかです。
overflowtodisk	メモリ内に保存できる要素の最大数に達したときに、memoryevictionpolicy の値に従ってオブジェクトをディスクに移動するかどうかを指定するブール値です。
statistics	true は、Ehcache の統計の収集が有効になっていることを示します。
timetoidoleconds	アイドル時間を示す秒数です。cfcache タグに idleTime 属性が指定されていない場合に使用されます。
timetolivesecond	タイムアウト時間を示す秒数です。cfcache タグに timespan 属性が指定されていない場合に使用されます。
objecttype	キャッシュのタイプで、次のいずれかです。 • template - ページキャッシュのプロパティを設定します。このページキャッシュにはキャッシュされたページおよびページセグメントが含まれます。 • object - オブジェクトキャッシュのプロパティを設定します。 • all - 両方のキャッシュタイプのプロパティを設定します。

CallStackGet

説明
構造体の配列を返します。各構造体には、テンプレート名、行番号、および適用可能な場合は関数名が含まれます。

シンタックス
callStackGet()

使用方法
呼び出しスタックは、すべての関数呼び出しのスナップショットです。ColdFusion アプリケーションでは、呼び出しスタックにより、テンプレート名、行番号、および適用可能な場合は関数名が提供されます。

この機能は、作成した再帰呼び出しを追跡する場合に役立ちます。

次に例を示します。

- a.cfm では、b.cfm のインクルードを実行しています。
- b.cfm では、15 行目で callStackGet 関数を呼び出しています。

この場合、呼び出しスタックで、テンプレート名、行番号および関数名を含む構造体が返されます。

例

この例では、数値の階乗を計算します。

ここでは、

1 テンプレート fact.cfm の function factorial の 9 行目で、callStackGet 関数を呼び出します。

2 同様に、テンプレート fact.cfm の 14 行目から呼び出します。

3 同様に、テンプレート callfact.cfm（2 行目）から呼び出します。

4 n のその他の値についても同様です。

callfact.cfm

```
<cftry>
    <cfinclude template="fact.cfm">
<cfcatch type="any">
    <cfoutput>
        #cfcatch.message#
        <br>#cfcatch.detail#
        <br>
    </cfoutput>
</cfcatch>
</cftry>
```

fact.cfm

```
<cfscript>
    numeric function factorial(n)
    {
        if(n == 1)
        {
            writedump(callStackGet());
            writeoutput("<br>");
            return 1;
        }
        else
        {
            writedump(callStackGet());
            writeoutput("<br>");
            return n * factorial(n - 1);
        }
    }
    factorial(5);
</cfscript>
```

CallStackDump

説明

callStackGet 関数に似ていますが、呼び出しスタックの文字列表現が返される点が異なります。

シンタックス
callStackDump(destination)

パラメータ

パラメータ	説明
destination	オプションのパラメーターで、次のいずれかの値を使用します。 • console • browser：これは、デフォルトの出力先です。 • file：ファイルの完全パスを指定しない場合は、getTempDirectory 関数によって決定されるテンポラリディレクトリにファイルが書き込まれます。

使用方法
呼び出しスタックは、すべての関数呼び出しのスナップショットです。ColdFusion アプリケーションでは、呼び出しスタックにより、テンプレート名、行番号、および適用可能な場合は関数名が提供されます。

この機能は、作成した再帰呼び出しを追跡する場合に役立ちます。

例
この例では、数値の階乗を計算します。この例は、CallStackGet の例に似ていますが、ここで使用されている関数が callStackDump である点が異なります。

callfact.cfm

```
<cftry>
    <cfinclude template="fact.cfm">
<cfcatch type="any">
    <cfoutput>
        #cfcatch.message#
        <br>#cfcatch.detail#
        <br>
    </cfoutput>
</cfcatch>
</cftry>
```

fact.cfm

```
<cffunction name="factorial" hint="returns the factorial of a number" output="true">
    <cfargument name="n" required="yes" type="numeric" hint="The number for which the factorial is
returned"/>
    <cfif n eq 1>
        <Cfset callStackDump()>
        <cfreturn 1>
    <cfelse>
        <Cfset callStackDump()>
        <cfreturn n * factorial(n - 1)>
    </cfif>
</cffunction>
<cfoutput> Factorial of 5 - #factorial(5)#</cfoutput>
```

Canonicalize

説明
入力文字列を標準化またはデコードします。

戻り値
入力文字列をデコードした形式

カテゴリ
表示および書式制御関数

シンタックス
canonicalize(inputString, restrictMultiple, restrictMixed)

履歴
ColdFusion 10: この関数が追加されました。

関連項目
EncodeForHTML、EncodeForHTMLAttribute、EncodeForJavaScript、EncodeForCSS、EncodeForURL

パラメータ

パラメータ	説明
inputString	必須です。エンコードする文字列です。
restrictMultiple	必須です。true に設定すると、複数のエンコードは制限されます。
restrictMixed	必須です。true に設定すると、混合エンコードは制限されます。

例
```
<cfoutput>#canonicalize("&lt;",false,false)#</cfoutput><br/>
<cfoutput>#canonicalize("%26lt; %26lt; %2526lt%253B %2526lt%253B
%2526lt%253B",false,false)#</cfoutput><br/>
<cfoutput>#canonicalize("&##X25;3c",false,false)#</cfoutput><br/>
<cfoutput>#canonicalize("&##x25;3c",false,false)#</cfoutput><br/>
```

Ceiling

説明
指定された数値を切り上げて整数にします。

戻り値
指定された数値を切り上げて整数にした値

カテゴリ
算術関数

関数のシンタックス
```
Ceiling(number)
```

関連項目
Int、Fix、Round

パラメータ

パラメータ	説明
number	実数値です。

例

```
<h3>Ceiling Example</h3>

<cfoutput>
    <p>The ceiling of 3.4 is #ceiling(3.4)#</p>
    <p>The ceiling of 3 is #ceiling(3)#</p>
    <p>The ceiling of 3.8 is #ceiling(3.8)#</p>
    <p>The ceiling of -4.2 is #ceiling(-4.2)#</p>
</cfoutput>
```

CharsetDecode

説明

文字列をバイナリ形式に変換します。

戻り値

文字列を表すバイナリオブジェクト

カテゴリ

変換関数、文字列関数

関数のシンタックス

```
CharsetDecode(string, encoding)
```

関連項目

BinaryDecode、BinaryEncode、CharsetEncode、『ColdFusion アプリケーションの開発』の Determining the page encoding of server output

履歴

ColdFusion MX 7: この関数が追加されました。

パラメータ

パラメータ	説明
string	バイナリ形式にエンコードするデータを含む文字列です。
encoding	入力データのエンコードを指定する文字列です。Java ランタイムで認識される文字エンコード名でなければなりません。一般的に使用される値を次に示します。 • utf-8 • iso-8859-1 • windows-1252 • us-ascii • shift_jis • iso-2022-jp • euc-jp • euc-kr • big5 • euc-cn • utf-16 Sun Java ランタイムによってサポートされているすべての文字エンコード名のリストについては、http://java.sun.com/j2se/1.3/docs/guide/intl/encoding.doc.html および http://java.sun.com/j2se/1.4/docs/guide/intl/encoding.doc.html を参照してください。

使用方法

この関数は、文字列をバイナリオブジェクトに直接変換します。ColdFusion の ColdFusion MX 6.1 リリースまでは、ToBase64 関数を使用して文字列を Base64 に変換した後、ToBinary 関数を使用して文字列をバイナリデータに変換する必要がありました。

例

次の例では、CharsetDecode 関数を使用して文字列をフォームからバイナリオブジェクトに変換し、CharsetEncode 関数を使用して元の形式の値に戻します。変換に使用する文字エンコードは、変更することができます。ただし、アジア言語のエンコードを選択した場合、指定した文字セットに含まれていない文字は正常に変換されません。

```
<h3>Character Encoding Conversion Example</h3>
<!--- Do the following if the form has been submitted. --->
<cfif IsDefined("Form.myString")>

    <!--- Do the conversions. --->
    <cfscript>
        chardecode=CharsetDecode(Form.myString, Form.charEncoding);
        charencode=CharsetEncode(chardecode, Form.charEncoding);
    </cfscript>

    <!--- Display the input values and results. --->
    <cfoutput>
        <h3>Parameter Settings</h3>
        <p><b>The string:</b><br>
             #Form.myString#</p>
        <p><b>The character encoding:</b> #Form.charEncoding#</p>

        <h3>Results of the operations:</h3>
        <p><b>Dump of the string converted to a binary object by CharsetDecode:
            </b><br>
            <cfdump var="#chardecode#"></p>
        <p><b>The binary object converted back to a string by CharsetEncode:
            </b><br>
        #charencode#</p>
    </cfoutput>
</cfif>

<!--- The input form. --->
<form action="#CGI.SCRIPT_NAME#" method="post">
    <b>Select the character encoding</b><br>
    <!--- This is a subset, additional encodings are available. --->
    <select size="1" name="charEncoding" >
        <option selected>UTF-8</option>
        <option>ASCII</option>
        <option>ISO8859_1</option>
        <option>CP1252</option>
        <option>SJIS</option>
        <option>MS932</option>
        <option>EUC_CN</option>
        <option>Big5</option>
    </select><br>
    <br>
    <b>Enter a string</b><br>
    <textarea name = "myString" cols = "40" rows = "5" WRAP = "VIRTUAL">
    The following four characters are not in all character encodings: ½âç÷
    </textarea><br>
    <br>
    <input type = "Submit" value = "convert my data">
</form>
```

CharsetEncode

説明

指定されたエンコードを使用して、バイナリデータを文字列に変換します。

戻り値

バイナリオブジェクトの文字列表現

カテゴリ
変換関数、文字列関数

関数のシンタックス
```
CharsetEncode(binaryobject, encoding)
```

関連項目
BinaryDecode、BinaryEncode、CharsetDecode、『ColdFusion アプリケーションの開発』の Determining the page encoding of server output

履歴
ColdFusion MX 7: この関数が追加されました。

パラメータ

パラメータ	説明
binaryobject	テキストにデコードするバイナリデータを含む変数です。
encoding	文字列をバイナリ形式にエンコードするために使用した文字エンコードです。Java ランタイムで認識される文字エンコード名でなければなりません。一般的に使用される値を次に示します。 • utf-8 • iso-8859-1 • windows-1252 • us-ascii • shift_jis • iso-2022-jp • euc-jp • euc-kr • big5 • euc-cn • utf-16 Sun Java ランタイムによってサポートされているすべての文字エンコード名のリストについては、http://java.sun.com/javase/6/docs/technotes/guides/intl/encoding.doc.html を参照してください。

使用方法
新しく開発するアプリケーションでバイナリデータを文字列に変換する場合は、ToString 関数ではなく、この関数を使用することをお勧めします。

例
次の例では、CharsetDecode 関数を使用して文字列をフォームからバイナリオブジェクトに変換し、CharsetEncode 関数を使用して元の形式の値に戻します。変換に使用する文字エンコードは、変更することができます。ただし、アジア言語のエンコードを選択した場合、指定した文字セットに含まれていない文字は正常に変換されません。

```
<h3>Character Encoding Conversion Example</h3>
<!--- Do the following if the form has been submitted. --->
<cfif IsDefined("Form.myString")>

    <!--- Do the conversions. --->
    <cfscript>
        chardecode=CharsetDecode(Form.myString, Form.charEncoding);
        charencode=CharsetEncode(chardecode, Form.charEncoding);
    </cfscript>

    <!--- Display the input values and results. --->
    <cfoutput>
        <h3>Parameter Settings</h3>
        <p><b>The string:</b><br>
             #Form.myString#</p>
        <p><b>The character encoding:</b> #Form.charEncoding#</p>

        <h3>Results of the operations:</h3>
        <p><b>Dump of the string converted to a binary object by CharsetDecode:
            </b><br>
            <cfdump var="#chardecode#"></p>
        <p><b>The binary object converted back to a string by CharsetEncode:
            </b><br>
        #charencode#</p>
    </cfoutput>
</cfif>

<!--- The input form. --->
<form action="#CGI.SCRIPT_NAME#" method="post">
    <b>Select the character encoding</b><br>
    <!--- This is a subset, additional encodings are available. --->
    <select size="1" name="charEncoding" >
        <option selected>UTF-8</option>
        <option>ASCII</option>
        <option>ISO8859_1</option>
        <option>CP1252</option>
        <option>SJIS</option>
        <option>MS932</option>
        <option>EUC_CN</option>
        <option>Big5</option>
    </select><br>
    <br>
    <b>Enter a string</b><br>
    <textArea name = "myString" cols = "40" rows = "5" WRAP = "VIRTUAL">
    The following four characters are not in all character encodings: ½âç÷
    </textArea><br>
    <br>
    <input type = "Submit" value = "convert my data">
</form>
```

Chr

説明

数値を UCS-2 文字に変換します。

戻り値

指定された UCS-2 コードの文字

カテゴリ
文字列関数

関数のシンタックス
```
Chr(number)
```

関連項目
Asc

履歴
ColdFusion MX: Unicode サポートが変更されました。Unicode 文字の Java UCS-2 表現の値が 65535 までサポートされるようになりました (以前のリリースでは 1 ～ 255 がサポートされていました)。

パラメータ

パラメータ	説明
number	値です (0 ～ 65535 の範囲内の数値を指定します)。

使用方法
0 ～ -31 の値は標準の非印刷文字のコードです。たとえば、次のようになります。

- Chr(10) は改行文字を返します。
- Chr(13) は復帰文字を返します。
- Chr(13) と Chr(10) の 2 文字の文字列は、Windows の改行文字を返します。

注意：Unicode 文字およびそのコードの詳細なリストについては、www.unicode.org/charts/ を参照してください。

例
```
<!--- If the character string is not empty, output its Chr value. --->
<cfif IsDefined("form.charVals")>
        <cfoutput>#form.charVals# = #Chr(form.charVals)#</cfoutput>
</cfif>

<cfform action="#CGI.script_name#" method="POST">
    <p>Type an integer character code from 1 to 65535<br>
        to see its corresponding character.<br>
    <cfinput type="Text"
        name="CharVals"
        range="1,65535"
        message="Enter an integer from 1 to 65535"
        validate="integer"
        required="Yes"
        size="5"
        maxlength="5"
        >
<p><input type="Submit" name=""> <input type="RESET">
</cfform>
```

CJustify

説明
任意の長さのフィールド内で文字列を中央に揃えます。

戻り値
文字列。入力パラメータの前または後にスペースが付加された中央揃えの文字列が返されます。**length** が入力パラメータ文字列の長さよりも短い場合は、元のままの文字列が返されます。

カテゴリ
表示および書式制御関数、文字列関数

関数のシンタックス
```
Cjustify(string, length)
```

関連項目
LJustify、RJustify

パラメータ

パラメータ	説明
string	文字列、または文字列を含んでいる変数です。空の文字列を指定することもできます。数値として定義された変数の場合は、文字列として処理されます。
length	正の整数、または正の整数を含んでいる変数です。フィールドの長さを指定します。 次のようにコードを記述することができます。 • 数値。例：6 • 数値を表す文字列。例："6" その他の値を使用するとエラーが発生します。

例
```
<!--- This example shows how to use CJustify. --->
<CFPARAM name = "jstring" DEFAULT = "">

<cfif IsDefined("FORM.submit")>
<cfdump var="#Form#">
    <cfset jstring = Cjustify("#FORM.justifyString#", 35)>
</cfif>
<html>
<head>
<title>CJustify Example</title>
</head>
<body>
<h3>CJustify</h3>
<p>Enter a string; it will be center-justified within the sample field.
<form action = "cjustify.cfm" method="post">
<p><input type = "Text" value = "<cfoutput>#jString#</cfoutput>"
        size = 35 name = "justifyString">
<p><input type = "Submit" name = "submit">
<input type = "RESET">
</form>
</body>
</html>
```

Compare

説明
大文字と小文字を区別して、2 つの文字列の比較を行います。

戻り値
- **string1** が **string2** より小さい場合は -1
- **string1** が **string2** と等しい場合は 0
- **string1** が **string2** より大きい場合は 1

カテゴリ
文字列関数

関数のシンタックス
```
Compare(string1, string2)
```

関連項目
CompareNoCase、Find

パラメータ

パラメータ	説明
string1	文字列、または文字列を含んでいる変数です。
string2	文字列、または文字列を含んでいる変数です。

使用方法
string1 および **string2** に対応する文字の値を比較します。

例

```
<h3>Compare Example</h3>
<p>The compare function performs a <I>case-sensitive</I> comparison of two strings.</p>

<cfif IsDefined("FORM.string1")>
    <cfset comparison = Compare(FORM.string1, FORM.string2)>
    <!--- Switch on the variable to give various responses. --->
    <cfswitch expression = #comparison#>
    <cfcase value = "-1">
        <h3>String 1 is less than String 2</h3>
        <I>The strings are not equal</I>
    </cfcase>
    <cfcase value = "0">
        <h3>String 1 is equal to String 2</h3>
        <I>The strings are equal!</I>
    </cfcase>
    <cfcase value = "1">
        <h3>String 1 is greater than String 2</h3>
        <I>The strings are not equal</I>
    </cfcase>
    <cfdefaultcase>
        <h3>This is the default case</h3>
    </cfdefaultcase>
    </cfswitch>
</cfif>
<form action = "compare.cfm" method="post">
<p>String 1
<br><input type = "Text" name = "string1">
<p>String 2
<br><input type = "Text" name = "string2">
<p><input type = "Submit" value = "Compare these Strings" name = "">
    <input type = "RESET">
</form>
```

CompareNoCase

説明

大文字と小文字を区別しないで、2 つの文字列の比較を行います。

戻り値

2 つの文字列の相違を表すインジケータ

- **string1** が **string2** より小さい場合は負の数
- **string1** が **string2** と等しい場合は 0
- **string1** が **string2** より大きい場合は正の数

カテゴリ

文字列関数

関数のシンタックス

```
CompareNoCase(string1, string2)
```

関連項目

Compare、FindNoCase、『ColdFusion アプリケーションの開発』の Evaluation and type conversion issues

パラメータ

パラメータ	説明
string1	文字列、または文字列を含んでいる変数です。
string2	文字列、または文字列を含んでいる変数です。

例

```
<H3>CompareNoCase Example</H3>
<P>This function performs a <I>case-insensitive</I> comparison of two strings.
<cfif IsDefined("form.string1")>
<cfset comparison = Comparenocase(form.string1, form.string2)>
<!--- switch on the variable to give various responses --->
<cfswitch expression=#comparison#>
    <cfcase value="-1">
        <H3>String 1 is less than String 2</H3>
        <I>The strings are not equal</I>
    </cfcase>
    <cfcase value="0">
        <H3>String 1 is equal to String 2</H3>
        <I>The strings are equal!</I>
    </cfcase>
    <cfcase value="1">
        <H3>String 1 is greater than String 2</H3>
        <I>The strings are not equal</I>
    </cfcase>
    <cfdefaultcase>
        <H3>This is the default case</H3>
    </cfdefaultcase>
</cfswitch>
</cfif>
<form action="comparenocase.cfm" method="POST">
<P>String 1
<BR><input type="Text" name="string1">
<P>String 2
<BR><input type="Text" name="string2">
<P><input type="Submit" value="Compare these Strings" name="">
    <input type="RESET">
</form>
```

Cos

説明

ラジアン単位で入力された角度のコサインを計算します。

戻り値

角度のコサインを表す数値

カテゴリ

算術関数

関数のシンタックス

```
Cos(number)
```

関連項目
ACos、Sin、ASin、Tan、Atn、Pi

パラメータ

パラメータ	説明
number	コサインを計算する角度です (単位 : ラジアン)。

使用方法
結果の範囲は -1 ～ 1 です。

度数をラジアンに変換するには、度数に p/180 を乗算します。ラジアンを度数に変換するには、ラジアンに 180/p を乗算します。

注意 : 浮動小数点演算の誤差により、本来は戻り値が 0 となるような角度を指定した場合、実際には小さい値 (たとえば 6.12323399574E-017) が返されます。したがって、結果が 0 かどうかを調べる場合は、戻り値が 0.0000000000001 より小さいかどうかで判定してください。

例
```
<h3>Cos Example</h3>
<!--- Calculate cosine if form has been submitted. --->
<cfif IsDefined("FORM.cosNum")>
<!--- Make sure input is a number. --->
    <cfif IsNumeric(#FORM.cosNum#)>
<!--- Convert degrees to radians, call the Cos function. --->
<cfset cosValue=#Cos((Form.cosNum * PI()) / 180)#>
<!--- 0.0000000000001 is the function's precision limit. If the absolute value
    of the returned cosine value is less, set result to 0 .--->
<cfif Abs(cosValue) LT 0.0000000000001>
    <cfset cosValue=0>
</cfif>
<cfoutput>Cos(#FORM.cosNum#) = #cosValue#<br><br></cfoutput>
    <cfelse>
<!--- If input is not a number, show an error message. --->
        <h4>You must enter a numeric angle in degrees.</h4>
    </cfif>
</cfif>
<form action = "#CGI.script_name#" method="post">
Enter an angle in degrees to get its cosine:
<br><input type = "Text" name = "cosNum" size = "15"><br><br>
<input type = "Submit" name = "">  
<input type = "RESET"
</form>
```

CreateDate

説明
日付時刻オブジェクトを作成します。

戻り値
日付時刻値

カテゴリ
日付および時刻関数

関数のシンタックス

```
CreateDate(year, month, day)
```

関連項目

CreateDateTime、CreateODBCDateTime、『ColdFusion アプリケーションの開発』の Evaluation and type conversion issues

パラメータ

パラメータ	説明
year	0 ～ 9999 の範囲の整数です。0 ～ 29 の範囲の整数は 2000 ～ 2029 に変換されます。30 ～ 99 の範囲の整数は 1930 ～ 1999 に変換されます。西暦 100 年より前の日付は指定できません。
month	1 (1 月) ～ 12 (12 月) の範囲の整数です。
day	1 ～ 31 の範囲の整数です。

使用方法

CreateDate は CreateDateTime のサブセットです。

返されたオブジェクトの時刻は 00:00:00 に設定されます。

例

```
<h3>CreateDate Example</h3>
<cfif IsDefined("form.year")>
<p>Your date value, generated with CreateDate:</p>
<cfset yourDate = CreateDate(form.year, form.month, form.day)>
<cfoutput>
<ul>
    <li>Formatted with CreateDate: #CreateDate(form.year, form.month, form.day)#</li>
    <li>Formatted with CreateDateTime: #CreateDateTime(form.year, form.month,
        form.day, 12,13,0)#</li>
    <li>Formatted with CreateODBCDate: #CreateODBCDate(yourDate)#</li>
    <li>Formatted with CreateODBCDateTime: #CreateODBCDateTime(yourDate)#</li>
</ul>

<p>The same value can be formatted with DateFormat:
<ul>
    <li>Formatted with CreateDate and DateFormat:
        #DateFormat(CreateDate(form.year, form.month, form.day), "mmm-dd-yyyy")#</li>
    <li>Formatted with CreateDateTime and DateFormat:
        #DateFormat(CreateDateTime(form.year, form.month, form.day, 12,13,0))#</li>
    <li>Formatted with CreateODBCDate and DateFormat:
        #DateFormat(CreateODBCDate(yourDate), "mmmm d, yyyy")#</li>
    <li>Formatted with CreateODBCDateTime and DateFormat:
        #DateFormat(CreateODBCDateTime(yourDate), "d/m/yy")#</li>
</ul>
</cfoutput>
</cfif>
<cfform action="createdate.cfm" METHOD="POST">
<p>Enter the year, month, and day, as integers:
<pre>
Year<cfinput type="Text" name="year" value="1998" validate="integer" required="Yes">
Month<cfinput type="Text" name="month" value="6" validate="integer" required="Yes">
Day<cfinput type="Text" name="day" value="8" validate="integer" required="Yes">
</pre>
<p><input type="Submit" name=""> <input type="RESET">
</cfform>
```

CreateDateTime

説明
日付時刻オブジェクトを作成します。

戻り値
日付時刻値

カテゴリ
日付および時刻関数

関数のシンタックス
```
CreateDateTime(year, month, day, hour, minute, second)
```

関連項目
CreateDate、CreateTime、CreateODBCDateTime、Now、『ColdFusion アプリケーションの開発』の Evaluation and type conversion issues

パラメータ

パラメータ	説明
year	0 ～ 9999 の範囲の整数です。0 ～ 29 の範囲の整数は 2000 ～ 2029 に変換されます。30 ～ 99 の範囲の整数は 1930 ～ 1999 に変換されます。西暦 100 年より前の日付は指定できません。
month	1 (1 月) ～ 12 (12 月) の範囲の整数です。
day	1 ～ 31 の範囲の整数です。
hour	0 ～ 23 の範囲の整数です。
minute	0 ～ 59 の範囲の整数です。
second	0 ～ 59 の範囲の整数です。

例
```
<h3>CreateDateTime Example</h3>

<cfif IsDefined("form.year")>
Your date value, generated with CreateDateTime:
<cfset yourDate = CreateDateTime(form.year, form.month, form.day,
    form.hour, form.minute, form.second)>

<cfoutput>
<ul>
    <li>Formatted with CreateDate: #CreateDate(form.year, form.month, form.day)#</li>
    <li>Formatted with CreateDateTime: #CreateDateTime(form.year, form.month,
    form.day, form.hour, form.minute, form.second)#</li>
    <li>Formatted with CreateODBCDate: #CreateODBCDate(yourDate)#</li>
    <li>Formatted with CreateODBCDateTime: #CreateODBCDateTime(yourDate)#</li>
</ul>

<p>The same value can be formatted with DateFormat:
<ul>
    <li>Formatted with CreateDate and DateFormat:
        #DateFormat(CreateDate(form.year, form.month, form.day), "mmm-dd-yyyy")#</li>
    <li>Formatted with CreateDateTime and DateFormat:
```

```
        #DateFormat(CreateDateTime(form.year, form.month, form.day,
    form.hour, form.minute, form.second))#</li>
    <li>Formatted with CreateODBCDate and DateFormat:
        #DateFormat(CreateODBCDate(yourDate), "mmmm d, yyyy")#</li>
    <li>Formatted with CreateODBCDateTime and DateFormat:
        #DateFormat(CreateODBCDateTime(yourDate), "d/m/yy")#</li>
</ul>
</cfoutput>
</cfif>

<CFFORM ACTION="createdatetime.cfm" METHOD="POST">
<p>Please enter the year, month, and day, in integer format, for a date to view:
<PRE>
Year<CFINPUT TYPE="Text" NAME="year" VALUE="1998" VALIDATE="integer"
        REQUIRED="Yes">
Month<CFINPUT TYPE="Text" NAME="month" VALUE="6" RANGE="1,12"
        MESSAGE="Please enter a month (1-12)" VALIDATE="integer"
        REQUIRED="Yes">
Day <CFINPUT TYPE="Text" NAME="day" VALUE="8" RANGE="1,31"
        MESSAGE="Please enter a day of the month (1-31)" VALIDATE="integer"
        REQUIRED="Yes">
Hour<CFINPUT TYPE="Text" NAME="hour" VALUE="16" RANGE="0,23"
        MESSAGE="You must enter an hour (0-23)" VALIDATE="integer"
        REQUIRED="Yes">
Minute<CFINPUT TYPE="Text" NAME="minute" VALUE="12" RANGE="0,59"
        MESSAGE="You must enter a minute value (0-59)" VALIDATE="integer"
        REQUIRED="Yes">
Second<CFINPUT TYPE="Text" NAME="second" VALUE="0" RANGE="0,59"
        MESSAGE="You must enter a value for seconds (0-59)" VALIDATE="integer"
        REQUIRED="Yes">
</PRE>
<p><INPUT TYPE="Submit" NAME=""> <INPUT TYPE="RESET">
</cfform>
```

CreateObject

説明

指定されたタイプの ColdFusion オブジェクトを作成します。

戻り値

指定されたタイプのオブジェクト

カテゴリ

拡張関数

履歴

ColdFusion 10：wsVersion パラメーターが追加されました。

ColdFusion 9:

- ColdFusion 9 では、type 引数は必須ではありません。

ColdFusion 8:

- .NET/dotnet タイプが追加されました。
- Web サービスオブジェクトの場合 : WSDL2Java パラメータと argstruct パラメータが追加されました。

ColdFusion MX 7: Web サービスオブジェクトの場合 : WSDL の service 要素で名前が付けられたポートを指定する、portName パラメータが追加されました。

ColdFusion MX:

1 インスタンス作成時の動作が変更されました。この関数、および cfobject タグにより、ColdFusion コンポーネントと Web サービスのインスタンスを作成できます。CFC オブジェクトでオペレーションを実行すると、CFC ファイル内の CFC のメソッドを実装する CFML コードが実行されます。

詳細については、『ColdFusion アプリケーションの開発』を参照してください。

2 CORBA オブジェクトの場合 : アドレスのネーミングサービスセパレータの形式が、ピリオドからスラッシュに変更されました。たとえば、あるクラスに "context=NameService" が指定されている場合、class パラメータには、次のいずれかの形式を使用します。

- "/Eng/CF"
- ".current/Eng.current/CF"

以前のリリースでは、形式は ".Eng.CF" でした。

3 CORBA オブジェクトの場合 : locale パラメータが変更されました。このパラメータでは、プロパティファイルを含む Java 設定を指定します。

CreateObject のオブジェクトの種類

この関数の使用方法の詳細については、次のセクションを参照してください。

- CreateObject: .NET オブジェクト
- CreateObject: COM オブジェクト
- CreateObject: コンポーネントオブジェクト
- CreateObject: CORBA オブジェクト
- CreateObject: Java または EJB オブジェクト
- CreateObject: Web サービスオブジェクト

注意：UNIX 上では、この関数は COM オブジェクトをサポートしません。

CreateObject: .NET オブジェクト

説明

.NET オブジェクトを作成します。このオブジェクトは、ローカルまたはリモートの .NET アセンブリに存在するクラスにアクセスするための ColdFusion プロキシになります。

戻り値

.NET オブジェクト (ローカルまたはリモートの .NET アセンブリへの ColdFusion 参照)

関数のシンタックス

```
CreateObject(type, class, assembly[, server, port, protocol, secure])
```

関連項目

cfobject: .NET オブジェクト、DotNetToCFType、『ColdFusion アプリケーションの開発』の Using Microsoft .NET Assemblies

パラメータ

属性	デフォルト	説明
type	component	オブジェクトのタイプです。.NET オブジェクトの場合は .NET または dotnet に設定する必要があります。
class		オブジェクトとして表現する .NET クラスの名前です。
assembly	.NET コアクラスを含む mscorlib.dll	**ローカル .NET アセンブリの場合** : .NET クラスとそのサポートクラスにアクセスするために使用するアセンブリ (.exe または .dll ファイル) の絶対パスです (複数指定可)。アセンブリ内のクラスが他のアセンブリ内のサポートクラスを必要とする場合は、それらのアセンブリも指定します。ただし、次のタイプのサポートクラスについては、サポートアセンブリの指定を省略できます。 • .NET コアクラス (mscorlib.dll に含まれるクラス) • GAC (Global Assembly Cache) に存在するアセンブリ内のクラス 複数のアセンブリを指定するには、カンマ区切りのリストを使用します。 **リモート .NET アセンブリの場合** : アセンブリの代わりとして使用するローカルプロキシ JAR ファイルの絶対パスを指定します (複数指定可)。 ローカルに .NET がインストールされていない場合にこのパラメータを省略すると、この関数は機能せず、エラーも生成されません。ローカルに .NET がインストールされている場合、このパラメータを省略して、指定したクラスが .NET コアクラスに存在しないときは、エラーが生成されます。
server	localhost	.NET サイドのエージェントが実行されているサーバーのホスト名または IP アドレスです。次のいずれかの形式で指定します。 • サーバー名 (例 : myserver) • IP アドレス (例 : 127.0.0.1) リモートサーバー上の .NET コンポーネントにアクセスする場合は、この属性を指定します。
port	6086	.NET サイドのエージェントがリスンするポート番号です。
protocol	tcp	ColdFusion と .NET の通信に使用するプロトコルです。有効な値は次のとおりです。 • http: HTTP/SOAP 通信プロトコルを使用します。tcp より遅くなりますが、ファイアウォール経由でアクセスする場合に必要になることがあります。 • tcp: バイナリの TCP/IP プロトコルを使用します。HTTP よりも効率的です。
secure	false	.NET サイドのエージェントとの通信を暗号化するかどうかを指定します。true を指定すると、.NET との通信に SSL が使用されます。

使用方法

CreateObject 関数と cfobject タグはシンタックスのみ異なります。ColdFusion .NET オブジェクトの作成の詳細については、461 ページの「cfobject: .NET オブジェクト」を参照してください。ColdFusion での .NET アセンブリの使用の詳細については、『ColdFusion アプリケーションの開発』の Using Microsoft .NET Assemblies を参照してください。

CreateObject: COM オブジェクト

説明

CreateObject 関数で、COM (Component Object Model) オブジェクトを作成することができます。

COM オブジェクトを作成するには、次の情報を指定します。

- オブジェクトのプログラム ID またはファイル名
- IDispatch インターフェイスを介して COM オブジェクトに使用できるメソッドおよびプロパティ
- オブジェクトのメソッドの引数および戻り値のタイプ

ほとんどのオブジェクトについて、これらの情報は、OLEView ユーティリティを使用して得ることができます。

注意：UNIX 上では、この関数は COM オブジェクトをサポートしません。

戻り値
COM オブジェクト

関数のシンタックス
```
CreateObject(type, class, context, serverName)
```

関連項目
ReleaseComObject、cfobject、『ColdFusion アプリケーションの開発』の Integrating COM and CORBA Objects in CFML Applications

パラメータ

パラメータ	説明
type	作成するオブジェクトタイプです。 • com • corba • java • component • webservice type のデフォルト値は component です。
class	呼び出すオブジェクトのコンポーネントプログラム ID です。
context	• InProc • Local • Remote
serverName	サーバー名です。UNC (Universal Naming Convention) または DNS (Domain Name Server) 表記規則を使用して、次のいずれかの形式で指定します。 • ¥¥lanserver • lanserver • http://www.servername.com • www.servername.com • 127.0.0.1 context = "remote" の場合、このパラメータは必須です。

使用方法
次の例では、メールを送信する NTS NewMail オブジェクトの Windows CDO (Collaborative Data Object) を作成します。このコードは cfscript タグ内で使用します。

```
Mailer = CreateObject("COM", "CDONTS.NewMail");
```

CreateObject: コンポーネントオブジェクト

説明
CreateObject 関数で、ColdFusion コンポーネント (CFC) オブジェクトのインスタンスを作成することができます。

戻り値
コンポーネントオブジェクト

関数のシンタックス
```
CreateObject(type, component-name)
```

関連項目
『ColdFusion アプリケーションの開発』の Building and Using ColdFusion Components

パラメータ

パラメータ	説明
type	作成するオブジェクトタイプです。 • com • corba • java • component • webservice type のデフォルト値は component です。
component-name	CFC 名です。コンポーネントを定義するファイルの名前と同じです。たとえば、"engineComp.cfc" ファイルで定義されているコンポーネントを指定するには、engineComp という名前を使用します。

使用方法
UNIX システムでは、まず指定のコンポーネント名と同じ名前 (ただしすべて小文字) のファイルが検索されます。これに該当するファイルが見つからない場合は、大文字小文字の違いも含めて、指定されたコンポーネント名とまったく同じ名前のファイルが次に検索されます。

次の例では、CFScript ステートメントで CreateObject 関数を使用して、tellTimeCFC 変数を tellTime コンポーネントに割り当てます。ここで CreateObject 関数は、別のディレクトリ内のコンポーネントを参照します。コンポーネントメソッドを呼び出すには、関数シンタックスを使用します。

```
<b>Server's Local Time:</b>
<cfscript>
    tellTimeCFC=CreateObject("component","appResources.components.
        tellTime");
    tellTimeCFC.getLocalTime();
</cfscript>
<br>
<b>Calculated UTC Time:</b>
<cfscript>
    tellTimeCFC.getUTCTime();
</cfscript>
```

CreateObject: CORBA オブジェクト

説明
CreateObject 関数で、CORBA オブジェクトのメソッドを呼び出すことができます。このオブジェクトは、使用する前にその定義と登録を行っておく必要があります。

戻り値
CORBA インターフェイスへのハンドル

関数のシンタックス
```
CreateObject(type, context, class, locale)
```

関連項目
『ColdFusion アプリケーションの開発』の Integrating COM and CORBA Objects in CFML Applications

履歴
CreateObject 関数のメインページの「履歴」を参照してください。

パラメータ

パラメータ	説明
type	作成するオブジェクトタイプです。 • com • corba • java • component • webservice type のデフォルト値は component です。
context	• IOR: ColdFusion では IOR を使用して CORBA サーバーにアクセスします。 • NameService: ColdFusion ではネーミングサービスを使用してサーバーにアクセスします。VisiBroker Orb の InitialContext の使用時のみ有効です。
class	• context = "ior" の場合 : 文字列形式の IOR (Interoperable Object Reference) が含まれているファイルの絶対パスです。このファイルは常に、ColdFusion で読み取り可能である必要があります。つまり、このファイルは、ColdFusion サーバーのローカルファイルであるか、またはネットワーク上のアクセス可能な場所に置かれている必要があります。 • context = "nameservice" の場合 : スラッシュで区切った、ネーミングサービスに対するネーミングコンテキストです。例 : Allaire//Doc/empobject
locale	プロパティファイルを含む Java 設定の名前です。詳細については、『ColdFusion 設定と管理』を参照してください。

使用方法
"context=NameService" の場合、**class** パラメータでは、文字列の最初の部分のセパレータとしてピリオドを使用します。次のいずれかの形式を使用します。

- "/Eng/CF"
- ".current/Eng.current/CF"

ColdFusion では、DII (Dynamic Invocation Interface) で CORBA がサポートされています。CORBA オブジェクトとともにこの関数を使用するには、文字列形式の IOR が含まれているファイル名、またはネーミングサービス内のオブジェクトのネーミングコンテキストのいずれかを指定します。また、オブジェクトの属性、メソッド名、およびメソッドのシグネチャも指定します。

この関数は、ユーザー定義のタイプ (構造体、配列、およびシーケンス) をサポートします。

例

```
myobj = CreateObject("corba", "d:\temp\tester.ior", "ior",
    "visibroker") //    uses IOR

myobj = CreateObject("corba", "/Eng/CF", "nameservice",
    "visibroker")    // uses nameservice

myobj = CreateObject("corba", "d:\temp\tester.ior", "nameservice")
       // uses nameservice and default configuration
```

CreateObject: Java または EJB オブジェクト

説明

CreateObject 関数で、Java オブジェクト、および拡張して EJB オブジェクトを作成することができます。

戻り値

Java オブジェクトです。

関数のシンタックス

```
CreateObject(type, class)
```

パラメータ

パラメータ	説明
type	作成するオブジェクトタイプです。 • com • corba • java • component • webservice type のデフォルト値は component です。
class	Java クラス名です。

使用方法

ColdFusion Administrator で指定されたクラスパスで使用可能な Java クラスは、CreateObject 関数を使用して ColdFusion からロードして使用できます。

Java メソッドとフィールドにアクセスするには :

1 CreateObject 関数または cfobject タグを呼び出してクラスをロードします。

2 init メソッドを適切な引数とともに使用して、クラスのインスタンスを呼び出します。たとえば、次のようになります。

```
<cfset ret = myObj.init(arg1, arg2)>
```

最初に "init" メソッドを呼び出さずにオブジェクト上のパブリックメソッドを呼び出すと、スタティックメソッドが呼び出されます。引数および戻り値は、任意の Java タイプ (simple、array、object) になります。ColdFusion では、文字列が引数として渡された場合は適切に変換されますが、文字列を戻り値として受け取った場合は変換は行われません。

オーバーロードされたメソッドは、引数の数が異なる場合はサポートされます。将来の拡張では、メソッドのシグネチャをより正確に作成できる cast 関数を使用できるようになります。

CreateObject: Web サービスオブジェクト

説明

この関数で、web サービスオブジェクトを作成できます。

戻り値

web サービスオブジェクト

関数のシンタックス

```
CreateObject(type, urltowsdl[, portname, wsdl2JavaArgs])

OR

CreateObject(type, urltowsdl, argStruct)
```

パラメータ

パラメータ	説明
type	作成するオブジェクトタイプです。 • com • corba • java • component • webservice type のデフォルト値は component です。
urltowsdl	Web サービスの WSDL ファイルの URL を指定します。次のいずれかです。 • Web サービスの絶対 URL • ColdFusion Administrator で Web サービスに割り当てられた名前 (文字列)

パラメータ	説明
portname	Web サービスのポート名です。この値は、service 要素内の port 要素の name 属性に対応します。大文字と小文字が区別されます。Web サービスに複数のポートが含まれている場合、このパラメータを指定します。ポートを指定しない場合、ColdFusion は WSDL の最初に検出されたポートを使用します。
wsdl2JavaArgs	WSDL2Java ツールに渡す引数のスペース区切りリストを含む文字列です。WSDL2Java ツールは Web サービス用の Java スタブを作成するためのツールです。使用可能な引数は次のとおりです。 • -W または --noWrapped: wrapped document/literal スタイルオペレーションの特別な処理を無効にします。 • -a または --all: 参照されないものも含め、WSDL 内の全要素のコードを生成します。 • -w または --wrapArrays: ラップされた XML 配列型に対してストレートな配列を作成するのではなく bean を作成します。このスイッチは Axis のマニュアルには記載されていません。 有効な引数の詳細については、『Apache Axis WSDL2Java Reference』を参照してください。
argStruct	Web サービスの設定引数を含む構造体です。詳細については、「使用方法」を参照してください。

使用方法

この CreateObject 関数を使用して、Web サービスを作成できます。

argStruct 構造体には、次の値を任意の組み合わせで格納できます。

名前	デフォルト	説明
password	存在する場合には、Administrator で設定されたパスワード	Web サービスにアクセスするためのパスワードです。Administrator で設定された Web サービス名が webservice 属性で指定されている場合は、Administrator のエントリで指定されたユーザー名ではなく、ここで指定されたユーザー名が使用されます。
port		シンタックスパラメータ表の portname を参照してください。
proxyPassword	存在する場合には、http.proxyPassword システムプロパティ	プロキシサーバー上で使用するユーザーのパスワードです。
proxyPort	存在する場合には、http.proxyPort システムプロパティ	プロキシサーバー上で使用するポートです。
proxyServer	存在する場合には、http.proxyHost システムプロパティ	webservice の URL にアクセスするために必要なプロキシサーバーです。
proxyUser	存在する場合には、http.proxyUser システムプロパティ	プロキシサーバーに送信するユーザー ID です。
refreshWSDL	no	• yes: WSDL ファイルをリロードし、Web サービスを利用するために必要な生成結果を再生成します。 • no
saveJava	no	• yes: Web サービス用の Java スタブを生成する WSDL2Java コンバータによって生成された Java を保存します。このコードはエラーのデバッグに役立ちます。 • no
timeout	0 (タイムアウトなし)	Web サービスの WSDL を取得するまでのタイムアウトです (単位 : 秒)。
username	存在する場合には、Administrator で設定されたユーザー名	Web サービスにアクセスするためのユーザー名です。Administrator で設定された Web サービス名が webservice 属性で指定されている場合は、Administrator のエントリで指定されたユーザー名ではなく、ここで指定されたユーザー名が使用されます。
wsdl2javaArgs		シンタックスパラメータ表を参照してください。

例

```
<cfscript>
    ws = CreateObject("webservice",
            "http://www.xmethods.net/sd/2001/TemperatureService.wsdl");
    xlatstring = ws.getTemp(zipcode = "55987");
    writeoutput("The temperature at 55987 is " & xlatstring);
</cfscript>
```

CreateODBCDate

説明

ODBC の日付オブジェクトを作成します。

戻り値

ODBC の標準日付形式の日付オブジェクト

カテゴリ

日付および時刻関数

関数のシンタックス

```
CreateODBCDate(date)
```

関連項目

CreateDate、CreateODBCDateTime

パラメータ

パラメータ	説明
date	日付または日付時刻オブジェクトです。値の範囲は西暦 100 ～ 9999 年です。

使用方法

この関数は、値の解析や検証は行いません。日付を確実に正しく入力し処理するため (たとえば、日 / 月 / 年と月 / 日 / 年とで混乱しないようにするため)、DateFormat 関数で mm-dd-yyyy のマスクを使用して、入力した日付を 3 つの要素に分けて解析することをお勧めします。ここでは適切な範囲内の値が使用されるようにしてください。たとえば、月の値を確認するには、属性として validate = "integer" および range = "1,12" を使用します。

例

```
<h3>CreateODBCDate Example</h3>
<cfif IsDefined("form.year")>
<p>Your date value, generated with CreateDateTime:</p>
<cfset yourDate = CreateDateTime(form.year, form.month, form.day, form.hour,
    form.minute, form.second)>
<cfoutput>
<ul>
    <li>Formatted with CreateDate: #CreateDate(form.year, form.month, form.day)#</li>
    <li>Formatted with CreateDateTime:
        #CreateDateTime(form.year, form.month, form.day, form.hour, form.minute,
        form.second)#</li>
    <li>Formatted with CreateODBCDate: #CreateODBCDate(yourDate)#</li>
    <li>Formatted with CreateODBCDateTime: #CreateODBCDateTime(yourDate)#</li>
</ul>
<p>The same value can be formatted with DateFormat:
<ul>
    <li>Formatted with CreateDate and DateFormat:
        #DateFormat(CreateDate(form.year,form.month, form.day), "mmm-dd-yyyy")#</li>
    <li>Formatted with CreateDateTime and DateFormat:
        #DateFormat(CreateDateTime(form.year, form.month, form.day, form.hour,
        form.minute, form.second))#</li>
    <li>Formatted with CreateODBCDate and DateFormat:
        #DateFormat(CreateODBCDate(yourDate), "mmmm d, yyyy")#</li>
    <li>Formatted with CreateODBCDateTime and DateFormat:
        #DateFormat(CreateODBCDateTime(yourDate), "d/m/yy")#</li>
</ul>
</cfoutput>
</cfif>
<cfform action="createodbcdate.cfm" method="POST">
<p>Enter the year, month and day, as integers:
<pre>
Year    <cfinput type="Text" name="year" value="1998" validate="integer"
        required="Yes">
Month   <cfinput type="Text" name="month" value="6" range="1,12"
        message="Please enter a month (1-12)" validate="integer"
        required="Yes">
Day     <cfinput type="Text" name="day" value="8" range="1,31"
        message="Please enter a day of the month (1-31)" validate="integer"
        required="Yes">
Hour    <cfinput type="Text" NAME="hour" value="16" range="0,23"
        message="You must enter an hour (0-23)" validate="integer"
        required="Yes">
Minute  <cfinput type="Text" name="minute" value="12" range="0,59"
        message="You must enter a minute value (0-59)" validate="integer"
        required="Yes">
Second  <cfinput type="Text" name="second" value="0" range="0,59"
        message="You must enter a value for seconds (0-59)" validate="integer"
        required="Yes">
</pre>
<p><input type="Submit" name=""> <input type="Reset">
</cfform>
```

CreateODBCDateTime

説明

ODBC 日付時刻オブジェクトを作成します。

戻り値
ODBC タイムスタンプ形式の日付時刻オブジェクト

カテゴリ
日付および時刻関数

関数のシンタックス
```
CreateODBCDateTime(date)
```

関連項目
CreateDateTime、CreateODBCDate、CreateODBCTime、Now、『ColdFusion アプリケーションの開発』の Evaluation and type conversion issues

パラメータ

パラメータ	説明
date	日付時刻オブジェクトです。値の範囲は西暦 100 ～ 9999 年です。

使用方法
日付時刻値を文字列として渡すときは、その値を引用符で囲みます。引用符で囲まない場合、その値は日付時刻オブジェクトの数値表現として解釈されます。

例
```
<!--- This example shows how to use CreateDate, CreateDateTime, CreateODBCDate, and CreateODBCDateTime --->
<h3>CreateODBCDateTime Example</h3>

<cfif IsDefined("form.year")>
Your date value, generated using CreateDateTime:
<cfset yourDate = CreateDateTime (form.year, form.month, form.day,
    form.hour,form.minute, form.second)>
<cfoutput>
<ul>
    <li>Formatted with CreateDate: #CreateDate(form.year, form.month,form.day)#
    <li>Formatted with CreateDateTime: #CreateDateTime(form.year,form.month,
        form.day,form.hour,form.minute,form.second)#
    <li>Formatted with CreateODBCDate: #CreateODBCDate(yourDate)#
    <li>Formatted with CreateODBCDateTime: #CreateODBCDateTime(yourDate)#
</ul>
<p>The same value can be formatted with DateFormat:
<ul>
    <li>Formatted with CreateDate and DateFormat:
        #DateFormat(CreateDate(form.year,form.month,form.day), "mmm-dd-yyyy")#
    <li>Formatted with CreateDateTime and DateFormat:
        #DateFormat(CreateDateTime(form.year,form.month,form.day,
        form.hour,form.minute,form.second))#
    <li>Formatted with CreateODBCDate and DateFormat:
        #DateFormat(CreateODBCDate(yourDate), "mmmm d, yyyy")#
    <li>Formatted with CreateODBCDateTime and DateFormat:
        #DateFormat(CreateODBCDateTime(yourDate), "d/m/yy")#
</ul>
</cfoutput>
</cfif>
<CFFORM ACTION="createodbcdatetime.cfm" METHOD="POST">
<p>Enter a year, month and day, as integers:
<PRE>
```

```
Year    <CFINPUT
            TYPE="Text"NAME="year" VALUE="1998" VALIDATE="integer" REQUIRED="Yes">

Month   <CFINPUT
            TYPE="Text" NAME="month"VALUE="6"RANGE="1,12"
            MESSAGE="Enter a month (1-12)" VALIDATE="integer" REQUIRED="Yes">

Day     <CFINPUT TYPE="Text" NAME="day" VALUE="8" RANGE="1,31"
            MESSAGE="Enter a day of the month (1-31)" VALIDATE="integer" REQUIRED="Yes">

Hour    <CFINPUT TYPE="Text" NAME="hour" VALUE="16" RANGE="0,23"
            MESSAGE="You must enter an hour (0-23)" VALIDATE="integer" REQUIRED="Yes">

Minute  <CFINPUT TYPE="Text" NAME="minute" VALUE="12" RANGE="0,59"
            MESSAGE="You must enter a minute value (0-59)" VALIDATE="integer" REQUIRED="Yes">

Second  <CFINPUT TYPE="Text" NAME="second" VALUE="0" RANGE="0,59"
            MESSAGE="You must enter a seconds value (0-59)" VALIDATE="integer" REQUIRED="Yes">
</PRE>
<p><INPUT TYPE="Submit" NAME=""> <INPUT TYPE="RESET">
</cfform>
```

CreateODBCTime

説明
ODBC 時刻オブジェクトを作成します。

戻り値
ODBC タイムスタンプ形式の時刻オブジェクト

カテゴリ
日付および時刻関数

関数のシンタックス
```
CreateODBCTime(date)
```

関連項目
CreateODBCDateTime、CreateTime、『ColdFusion アプリケーションの開発』の Evaluation and type conversion issues

パラメータ

パラメータ	説明
date	日付時刻オブジェクトです。値の範囲は西暦 100 ～ 9999 年です。

使用方法
日付時刻値を文字列として渡すときは、その値を引用符で囲みます。引用符で囲まない場合、その値は日付時刻オブジェクトの数値表現として解釈されます。

例

```
<h3>CreateODBCTime Example</h3>
<CFIF IsDefined("form.hour")>
Your time value, created with CreateTime...
<CFSET yourTime = CreateTime(form.hour, form.minute, form.second)>
<cfoutput>
<ul>
    <li>Formatted with CreateODBCTime: #CreateODBCTime(yourTime)#
    <li>Formatted with TimeFormat: #TimeFormat(yourTime)#
</ul></cfoutput>
</CFIF>
<CFFORM action="createodbctime.cfm" METHOD="post">
<PRE>
Hour    <CFINPUT TYPE="Text" NAME="hour" VALUE="16" RANGE="0,23" MESSAGE="You must
    enter an hour (0-23)" VALIDATE="integer" REQUIRED="Yes">
Minute  <CFINPUT TYPE="Text" NAME="minute" VALUE="12" RANGE="0,59" MESSAGE="You must
    enter a minute value (0-59)" VALIDATE="integer" REQUIRED="Yes">
Second  <CFINPUT TYPE="Text" NAME="second" VALUE="0" RANGE="0,59" MESSAGE="You must
    enter a value for seconds (0-59)" VALIDATE="integer" REQUIRED="Yes">
</PRE>
<p><INPUT TYPE="Submit" NAME=""> <INPUT TYPE="RESET">
</cfform>
```

CreateTime

説明

時刻変数を作成します。

戻り値

時刻変数

カテゴリ

日付および時刻関数

関数のシンタックス

```
CreateTime(hour, minute, second)
```

関連項目

CreateODBCTime、CreateDateTime、『ColdFusion アプリケーションの開発』の Evaluation and type conversion issues

パラメータ

パラメータ	説明
hour	0 ～ 23 の範囲の数値です。
minute	0 ～ 59 の範囲の数値です。
second	0 ～ 59 の範囲の数値です。

使用方法

CreateTime は CreateDateTime のサブセットです。

時刻変数は、日付時刻変数の特殊ケースです。時刻変数の日付の部分は、1899 年 12 月 30 日に設定されます。

例

```
<h3>CreateTime Example</h3>
<cfif IsDefined("FORM.hour")>
Your time value, presented using CreateTime time function:
<cfset yourTime = CreateTime(FORM.hour, FORM.minute, FORM.second)>
<cfoutput><ul>
    <li>Formatted with timeFormat: #TimeFormat(yourTime)#</li>
    <li>Formatted with timeFormat and hh:mm:ss: #TimeFormat(yourTime, 'hh:mm:ss')#</li>
</ul></cfoutput>
</cfif>
<CFFORM action="createtime.cfm" METHOD="post">
<PRE>
Hour   <CFINPUT TYPE="Text" NAME="hour" VALUE="16" RANGE="0,23"
    MESSAGE="You must enter an hour (0-23)" VALIDATE="integer" REQUIRED="Yes">
Minute <CFINPUT TYPE="Text" NAME="minute" VALUE="12" RANGE="0,59"
    MESSAGE="You must enter a minute value (0-59)" VALIDATE="integer"
    REQUIRED="Yes">
Second <CFINPUT TYPE="Text" NAME="second" VALUE="0" RANGE="0,59"
    MESSAGE="You must enter a value for seconds (0-59)" VALIDATE="integer"
    REQUIRED="Yes">
</PRE>
<p><INPUT TYPE="Submit" NAME=""> <INPUT TYPE="RESET"></p>
</cfform>
```

CreateTimeSpan

説明

期間を定義する日付時刻オブジェクトを作成します。このオブジェクトは、他の日付時刻オブジェクトへの加算または減算が可能であり、cfquery の cachedWithin 属性とともに使用できます。

戻り値

日付時刻オブジェクト（Java の double 型）

カテゴリ

日付および時刻関数

関数のシンタックス

```
CreateTimeSpan(days, hours, minutes, seconds)
```

関連項目

CreateDateTime、DateAdd、DateConvert、『ColdFusion アプリケーションの開発』の Defining application-level settings and variables

パラメータ

パラメータ	説明
days	期間の日数を示す 0 ～ 32768 の範囲の整数です。
hours	期間の時数を示す数値です。
minutes	期間の分数を示す数値です。
seconds	期間の秒数を示す数値です。

使用方法

特殊な日付時刻オブジェクトを作成します。このオブジェクトは、他の日付時刻オブジェクトに対して加算または減算を行う場合や、cfquery の cachedWithin 属性とともに使用する場合にのみ使用できます。

cfquery の cachedWithin 属性を使用する場合、元のクエリーの日付がユーザーが定義した期間内であれば、キャッシュされたクエリーデータが使用されます。この場合、CreateTimeSpan 関数は現在からさかのぼって期間を定義するために使用します。cachedWithin 属性は、ColdFusion Administrator のクエリーのキャッシュ機能が有効な場合にのみ使用できます。詳細については、cfquery を参照してください。

例

```
<!--- This example shows the use of CreateTimeSpan with cfquery --->
<h3>CreateTimeSpan Example</h3>
<!--- define startrow and maxrows to facilitate 'next N' style browsing --->
<CFPARAM name = "MaxRows" default = "10">
<CFPARAM name = "StartRow" default = "1">
<!--- Query database for information, if cached database information has not been updated in the last six
hours. -------->
<cfoutput>
<cfquery name = "GetParks" datasource = "cfdocexamples"
    cachedWithin = "#CreateTimeSpan(0, 6, 0, 0)#">
    SELECT PARKNAME, REGION, STATE
    FROM Parks
    ORDER by ParkName, State
</cfquery>
</cfoutput>
<!--- build HTML table to display query --->
<TABLE cellpadding = 1 cellspacing = 1>
<TR>
    <TD colspan = 2 bgcolor = f0f0f0>
    <B><I>Park Name</I></B>
    </TD>
    <TD bgcolor = f0f0f0>
    <B><i>Region</I></B>
    </TD>
    <TD bgcolor = f0f0f0>
    <B><I>State</I></B>
    </TD>
</TR>
<!--- Output query, define startrow and maxrows. Use query variable CurrentCount to track the row you are
displaying. --->
<cfoutput query = "GetParks" StartRow = "#StartRow#"
    MAXROWS = "#MaxRows#">
<TR>
    <TD valign = top bgcolor = ffffed>
    <B>#GetParks.CurrentRow#</B>
    </TD>
    <TD valign = top>
    <FONT SIZE = "-1">#ParkName#</FONT>
    </TD>
```

```
    <TD valign = top>
    <FONT SIZE = "-1">#Region#</FONT>
    </TD>
    <TD valign = top>
    <FONT SIZE = "-1">#State#</FONT>
    </TD>
</TR>
</cfoutput>
<!--- If number of records is less than or equal to number of rows, offer link to same page, with startrow
value incremented by maxrows (in this example,
    incremented by 10). --->
<TR>
    <TD colspan = 4>
    <cfif (StartRow + MaxRows) LTE GetParks.RecordCount>
        <a href = "cfquery.cfm?startrow = <cfoutput>#StartRow + MaxRows#
        </cfoutput>">See next <cfoutput>#MaxRows#</cfoutput> rows</A>
    </cfif>
    </TD>
</TR>
</TABLE>
```

CreateUUID

説明
UUID (Universally Unique Identifier) を作成します。UUID は、固有の 128 ビット整数で表される 35 文字の文字列です。

戻り値
ColdFusion 形式の UUID。xxxxxxxx-xxxx-xxxx-xxxxxxxxxxxxxxxx という形式で、x は 16 進数 (0 ～ 9 または A ～ F) です (文字グループは 8-4-4-16 です)。

カテゴリ
その他の関数

関数のシンタックス
```
CreateUUID()
```

使用方法
ColdFusion の UUID 生成アルゴリズムでは、固有の time-of-day 値、IEEE 802 の Host ID、および暗号作成上強力な乱数ジェネレータを使用して、IEEE RFC の "**UUIDs and GUIDs**" のドラフトで示された原則に準拠した UUID を生成します。

ColdFusion の UUID 形式は次のとおりです。

xxxxxxxx-xxxx-xxxx-xxxxxxxxxxxxxxxx (8-4-4-16)

これは、次のような Microsoft/DCE 標準には準拠していません。

xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx (8-4-4-4-12)

UUID テストツールおよびユーザー定義関数 CreateGUID は、Web サイト www.cflib.org から入手可能です。CreateGUID は、CFML UUID を UUID/Microsoft GUID 形式に変換する関数です。

この関数は、分散環境において永続的な識別子を生成するために使用します。この関数では、確実性を高めるため、固有の値が返されます。同じシステムまたは他のどのシステムから呼び出された場合でも、同じ値は返されません。

UUID は、DCE/RPC、COM+、CORBA などの分散型コンピューティングフレームワークで使用されます。ColdFusion では、データが複数の共有データベース上に保管されるアプリケーションに対して、UUID をプライマリテーブルキーとして使用できます。このような場合に数値のキーを使用すると、テーブルをマージするときにプライマリキーの制約違反が起こる可能性があります。UUID を使用することによって、これらの違反が起こらないようにできます。

例

```
<h3>CreateUUID Example</h3>
<p> This example uses CreateUUID to generate a UUID when you submit the form.
    You can submit the form more than once. </p>
<!--- Checks whether the form was submitted; if so, creates UUID. --->
<cfif IsDefined("Form.CreateUUID") Is True>
    <hr>
    <p>Your new UUID is: <cfoutput>#CreateUUID()#</cfoutput></p>
</cfif>
<form action = "createuuid.cfm">
<p><input type = "Submit" name = "CreateUUID"> </p>
</form>
```

CSRFGenerateToken

説明

ランダムトークンを指定し、それをセッションに格納します。また、セッションに格納するための特定のキーを指定することもできます。

戻り値

トークン

カテゴリ

表示および書式制御関数

シンタックス

CSRFGenerateToken([key] [,forceNew])

関連項目

CSRFVerifyToken

履歴

ColdFusion 10: この関数が追加されました。

パラメータ

パラメータ	必須／オプション	説明
key	オプション	指定したキーに基づいてランダムトークンが生成されます。このキーはセッションに格納されます。
forceNew	オプション	true に設定すると、メソッドが呼び出されるたびに新しいトークンが生成されます。false の場合、そのキーのトークンが存在する場合は同じキーが返されます。

使用方法

この関数は、ランダムトークンを作成して、セッションに格納するために使用します。

例

```
<cfset csrfToken=CSRFGenerateToken() />
<cfform method="post" action="sayHello.cfm">
    <cfinput name="userName" type="text" >
    <cfinput name="token" value="#csrfToken#" type="hidden" >
    <cfinput name="submit"  value="Say Hello!!" type="submit" >
</cfform>
```

CSRFVerifyToken

説明

指定されたトークンを、セッションに格納されている特定のキーの同じトークンと照合して検証します。

戻り値

ブール値です。true は正常な終了を示します。

カテゴリ

表示および書式制御関数

シンタックス

CSRFVerifyToken(token [,key])

関連項目

CSRFGenerateToken

履歴

ColdFusion 10: この関数が追加されました。

パラメータ

パラメータ	必須／オプション	説明
token	必須	セッションに格納されているトークンとの検証を行うトークンです。
key	オプション	トークンの検索に使用するキーです。

使用方法

この関数は、指定されたトークンを、セッションに格納されている特定のキーの同じトークンと照合して検証するために使用します。

例

```
<cfset token=form.token>
    <cfset validate = CSRFverifyToken(token)>
<cfoutput >#validate#</cfoutput>
```

DateAdd

説明

時刻の単位を日付に追加します。

戻り値
日付時刻オブジェクト

カテゴリ
日付および時刻関数

関数のシンタックス
```
DateAdd("datepart", number, "date")
```

関連項目
DateConvert、DatePart、CreateTimeSpan

履歴
ColdFusion MX 6.1: ミリ秒を表す文字 L および l が **datepart** に追加されました。

パラメータ

パラメータ	説明
datepart	文字列です。 • yyyy: 年 • q: 四半期 • m: 月 • y: 年間通算日 • d: 日 • w: 曜日 • ww: 週 • h: 時 • n: 分 • s: 秒 • l: ミリ秒
number	**date** に加算する **datepart** の単位の数値です (将来の日付を得るには正の数値を指定し、過去の日付を得るには負の数値を指定します)。数値は整数でなければなりません。
date	日付時刻オブジェクトです。値の範囲は西暦 100 ～ 9999 年です。

使用方法
datepart の値 y、d、および w を使用すると、日付に日数が加算されます。

日付時刻オブジェクトを文字列として渡すときは、そのオブジェクトを引用符で囲みます。引用符で囲まない場合、そのオブジェクトは日付時刻オブジェクトの数値表示として解釈されます。

例

```
<cfset date="{ts '2433-09-01 23:59:59'}">
    <cfoutput>#date#</cfoutput>
        <cfset diff=30>
        <cfset posdateresult=DateAdd("s",diff,date1)>
    <cfoutput>#posdateresult#</cfoutput>
```

例

```
<!--- This example shows the use of DateAdd --->
<cfparam name="value" default="70">
<cfparam name="type" default="m">

<!--- If numbers passed, then use those. --->
<cfif IsDefined("form.value")>
    <cfset value = form.value>
</cfif>
<cfif IsDefined("form.type")>
    <cfset type = form.type>
</cfif>


<cfquery name="GetMessages" datasource="cfdocexamples">
    SELECT UserName, Subject, Posted
    FROM Messages
</cfquery>

<p>This example uses DateAdd to determine when a message in
the database will expire. Currently, messages older
than <cfoutput>#value#</cfoutput>

<cfswitch expression="#type#">
    <cfcase value="yyyy">years</cfcase>
    <cfcase value="q">quarters</cfcase>
    <cfcase value="m">months</cfcase>
    <cfcase value="y">days of year</cfcase>
    <cfcase value="w">weekdays</cfcase>
    <cfcase value="ww">weeks</cfcase>
    <cfcase value="h">hours</cfcase>
    <cfcase value="n">minutes</cfcase>
    <cfcase value="s">seconds</cfcase>
    <cfdefaultcase>years</cfdefaultcase>
</cfswitch>
    are expired.

<table>
<tr>
    <td>UserName</td>
    <td>Subject</td>
    <td>Posted</td>
</tr>
<cfoutput query="GetMessages">
<tr>
    <td>#UserName#</td>
    <td>#Subject#</td>
    <td>#Posted# <cfif DateAdd(type, value, posted) LT Now()><font color="red">EXPIRED</font></cfif></td>
</tr>
```

```
</cfoutput>
</table>

<cfform action="#CGI.Script_Name#" method="post">

Select an expiration value:
<cfinput type="Text" name="value" value="#value#" message="Please enter whole numbers only"
validate="integer" required="Yes">
<select name="type">
    <option value="yyyy">years
    <option value="m" selected>months
    <option value="d">days
    <option value="ww">weeks
    <option value="h">hours
    <option value="n">minutes
    <option value="s">seconds
</select>

<input type="Submit" value="Submit">
</cfform>
```

DateCompare

説明

2 つの日付の日付時刻の詳細比較を行います。

戻り値

- **date1** が **date2** より前の日付である場合は -1
- **date1** が **date2** と等しい場合は 0
- **date1** が **date2** より後の日付である場合は 1

カテゴリ

日付および時刻関数

関数のシンタックス

```
DateCompare("date1", "date2" [, "datePart"])
```

関連項目

CreateDateTime、DatePart

パラメータ

パラメータ	説明
date1	日付時刻オブジェクトです。値の範囲は西暦 100 ～ 9999 年です。
date2	日付時刻オブジェクトです。値の範囲は西暦 100 ～ 9999 年です。
datePart	オプション。文字列。比較の精度を表します。 • s 秒の精度 (デフォルト) • n 分の精度 • h 時の精度 • d 日の精度 • m 月の精度 • yyyy 年の精度

使用方法

日付時刻オブジェクトを文字列として渡すときは、そのオブジェクトを引用符で囲みます。引用符で囲まない場合、そのオブジェクトは日付時刻オブジェクトの数値表示として解釈されます。

例

```
<h3>DateCompare Example</h3>
<p>The DateCompare function compares two date/time values.
<cfif IsDefined("FORM.date1")>
    <cfif IsDate(FORM.date1) and IsDate(FORM.date2)>
        <cfset comparison = DateCompare(FORM.date1, FORM.date2, FORM.precision)>

<!--- Switch on the variable to give various responses. --->
        <cfswitch expression = #comparison#>
            <cfcase value = "-1">
                <h3><cfoutput>#DateFormat(FORM.date1)#
                #TimeFormat(FORM.date1)#</cfoutput> (Date 1) is
                earlier than <cfoutput>#DateFormat(FORM.date2)#
                #TimeFormat(FORM.date2)#</cfoutput> (Date 2)</h3>
                <I>The dates are not equal</I>
            </cfcase>
            <cfcase value = "0">
                <h3><cfoutput>#DateFormat(FORM.date1)#
                #TimeFormat(FORM.date1)#</cfoutput> (Date 1) is equal
                to <cfoutput>#DateFormat(FORM.date2)#
                #TimeFormat(FORM.date2)#</cfoutput> (Date 2)</h3>
                <I>The dates are equal!</I>
            </cfcase>
            <cfcase value = "1">
                <h3><cfoutput>#DateFormat(FORM.date1)#
                #TimeFormat(FORM.date1)#</cfoutput> (Date 1) is later
                than <cfoutput>#DateFormat(FORM.date2)#
                #TimeFormat(FORM.date2)#</cfoutput> (Date 2)</h3>
                <I>The dates are not equal</I>
            </cfcase>
            <CFDEFAULTCASE>
                <h3>This is the default case</h3>
            </CFDEFAULTCASE>
        </cfswitch>
    <cfelse>
        <h3>Enter two valid date values</h3>
```

```
    </cfif>
</cfif>

<form action = "datecompare.cfm" method="post">
<hr size = "2" color = "#0000A0">
<p>Date 1
<br><input type = "Text" name = "date1"
        value = "<cfoutput>#DateFormat(Now())# #TimeFormat(Now())#
</cfoutput>">
<p>Date 2
<br><input type = "Text" name = "date2"
        value = "<cfoutput>#DateFormat(Now())# #TimeFormat(Now())#
</cfoutput>">
<p>Specify precision to the:
<br><select name = "precision">
            <option value = "s">
                Second
            </option>
            <option value = "n">
                Minute
            </option>
            <option value = "h">
                Hour
            </option>
            <option value = "d">
                Day
            </option>
            <option value = "m">
                Month
            </option>
            <option value = "yyyy">
                Year
            </option>
    </select>
<p><input type = "Submit" value = "Compare these dates" name = "">
<input type = "reset">
</form>
```

DateConvert

説明

現地時刻から世界標準時 (UTC: Universal Coordinated Time) への変換、または UTC から現地時刻への変換を行います。この関数は必要に応じて、実行コンピュータの夏時間設定を使用して夏時間を計算します。

戻り値

UTC または現地時刻の形式が設定された時刻オブジェクト

カテゴリ

日付および時刻関数

関数のシンタックス

```
DateConvert("conversion-type", "date")
```

関連項目

GetTimeZoneInfo、CreateDateTime、DatePart

パラメータ

パラメータ	説明
conversion-type	• local2Utc: 現地時刻を UTC 時間に変換します。 • utc2Local: UTC 時間を現地時刻に変換します。
date	日付と時刻の文字列、または日付と時刻の文字列を含む変数です。 これを作成するには、CreateDateTime を使用します。

使用方法

日付時刻オブジェクトを文字列として渡すときは、そのオブジェクトを引用符で囲みます。引用符で囲まない場合、そのオブジェクトは日付時刻オブジェクトの数値表示として解釈されます。

注意：CreateDate 関数または Now 関数をこの関数の date パラメータとして渡すことができます。例 : #DateConvert(CreateDate(2007, 4, 10))#

例

```
<h3>DateConvert Example</h3>
<!--- This shows conversion of current date - time to UTC and back. --->
<cfset curDate = Now()>
<p><cfoutput>The current date and time: #curDate#. </cfoutput></p>
<cfset utcDate = DateConvert("local2utc", curDate)>
<cfoutput>
    <p>The current date and time converted to UTC time: #utcDate#.</p>
</cfoutput>
<!--- This code checks whether form was submitted. If so, the code generates
    the CFML date with the CreateDateTime function. --->
<cfif IsDefined("FORM.submit")>
    <cfset yourDate = CreateDateTime(FORM.year, FORM.month, FORM.day,
        FORM.hour,FORM.minute, FORM.second)>
    <p><cfoutput>Your date value, presented as a ColdFusion date/time
        string:#yourdate#.</cfoutput></p>
<cfset yourUTC = DateConvert("local2utc", yourDate)>
    <p><cfoutput>Your date and time value, converted to Coordinated Universal Time
    (UTC): #yourUTC#.</cfoutput></p>
    <p><cfoutput>Your UTC date and time, converted back to local date and time:
    #DateConvert("utc2local", yourUTC)#.
    </cfoutput></p>
<cfelse>
    Type the date and time, and press Enter to see the conversion.
</cfif>
<Hr size = "2" color = "#0000A0">
<form action = "dateconvert.cfm" method="post">
<p>Enter year, month and day in integer format for date value to view:
<table cellspacing = "2" cellpadding = "2" border = "0">
<tr>
<td>Year</td>
<td><input type = "Text" name = "year" value = "1998"
        validate = "integer" required = "Yes"></td></tr>
<tr>
<td>Month</td>
<td><input type = "Text" name = "month" value = "6"
        range = "1,12" message = "Enter a month (1-12)"
        validate = "integer" required = "Yes"></td></tr>
<tr>
<td>Day</td>
<td><input type = "Text" name = "day" value = "8"
        range = "1,31"
```

```
        message = "Enter a day of the month (1-31)"
        validate = "integer" required = "Yes"></td></tr>
<tr>
<td>Hour</td>
<td><input type = "Text" name = "hour" value = "16"
        range = "0,23"
        message = "You must enter an hour (0-23)"
        validate = "integer" required = "Yes"></td></tr>
<tr>
<td>Minute</td>
<td><input type = "Text" name = "minute" value = "12"
        range = "0,59"
        message = "You must enter a minute value (0-59)"
        validate = "integer" required = "Yes"></td></tr>
<tr>
<td>Second</td>
<td><input type = "Text" name = "second" value = "0"
        range = "0,59"
        message = "You must enter a value for seconds (0-59)"
        validate = "integer" required = "Yes"></td></tr>
<tr>
<td><input type = "Submit" name = "submit" value = "Submit"></td>
<td><input type = "RESET"></td></tr>
</table>
```

DateDiff

説明
date2 から **date1** を引いた値の単位数を整数で取得します。

戻り値
タイプ **datepart** の単位数

カテゴリ
日付および時刻関数

関数のシンタックス

```
DateDiff("datepart", "date1", "date2")
```

関連項目
DateAdd、DatePart、CreateTimeSpan

履歴
ColdFusion MX:

- 日付の差が負の数の場合の計算方法が変更されました。この関数では、日付の差が負の数になった場合も正しく計算できるようになりました。出力は以前のリリースとは異なる場合があります。
- w マスクおよび ww マスクが変更されました。2 つの日付の間が丸何週間かを求めます。

パラメータ

パラメータ	説明
datepart	どの単位で数えるかを指定する文字列です。たとえば、yyyy と指定すると、日付の差が丸何年かを求めます。 • yyyy: 年 • q: 四半期 • m: 月 • y: 年間通算日 (d と同じ) • d: 日 • w: 曜日 (ww と同じ) • ww: 週 • h: 時 • n: 分 • s: 秒
date1	日付時刻オブジェクトです。値の範囲は西暦 100 ～ 9999 年です。
date2	日付時刻オブジェクトです。値の範囲は西暦 100 ～ 9999 年です。

使用方法

DateDiff 関数は、2 つの日付の間に **datepart** 単位でどれだけの差があるかを求めます。たとえば、**datepart** パラメータが "m" で、日付の差が 55 日の場合、この関数は 1 を返します。

日付の文字列定数は引用符で囲みます。数値のみのテキスト (たとえば 1932) を引用符で囲まずに指定した場合は、日付時刻オブジェクトとして解釈されるため、結果は誤った値となります。

例

```
<cfif IsDefined("form.value")>
    <cfset value = form.value>
</cfif>
<cfif IsDefined("form.type")>
    <cfset type = form.type>
</cfif>

<cfif IsDefined("form.date1") and IsDefined("form.date2")>

    <cfif IsDate(form.date1) and IsDate(form.date2)>

        <p>This example uses DateDiff to determine the difference
        in
        <cfswitch expression = "#form.type#">
            <cfcase value="yyyy">years</cfcase>
            <cfcase value="q">quarters</cfcase>
            <cfcase value="m">months</cfcase>
            <cfcase value="y">days</cfcase>
            <cfcase value="d">days</cfcase>
            <cfcase value="w">weekdays</cfcase>
            <cfcase value="ww">weeks</cfcase>
            <cfcase value="h">hours</cfcase>
            <cfcase value="n">minutes</cfcase>
            <cfcase value="s">seconds</cfcase>
```

```
            <cfdefaultcase>years</cfdefaultcase>
        </cfswitch>
             dateparts between date1 and date2.

        <cfif DateCompare("#form.date1#","#form.date2#") is not 0>
        <p>The difference is <cfoutput>#Abs(DateDiff(type, form.date2, form.date1))#</cfoutput>
        <cfswitch expression = "#form.type#">
            <cfcase value="yyyy">years</cfcase>
            <cfcase value="q">quarters</cfcase>
            <cfcase value="m">months</cfcase>
            <cfcase value="y">days</cfcase>
            <cfcase value="d">days</cfcase>
            <cfcase value="w">weekdays</cfcase>
            <cfcase value="ww">weeks</cfcase>
            <cfcase value="h">hours</cfcase>
            <cfcase value="n">minutes</cfcase>
            <cfcase value="s">seconds</cfcase>
            <cfdefaultcase>years</cfdefaultcase>
        </cfswitch>.
        <cfelse>
        <p>The two dates are equal!Try changing one of the values ...
        </cfif>

    <cfelse>
    <p>Please enter two valid date/time values, formatted like this:
    <cfoutput>#DateFormat(Now())#</cfoutput>
    </cfif>

</cfif>
<form action="index.cfm" method="post">

<pre>
Date 1
<input type="Text" name="date1" value="<cfoutput>#DateFormat(Now())#</cfoutput>">
Date 2
<input type="Text" name="date2" value="<cfoutput>#DateFormat(Now())#</cfoutput>">
What kind of unit to show difference?
    <select name="type">
        <option value="yyyy" selected>years
        <option value="q">quarters
        <option value="m">months
        <option value="y">days of year
        <option value="d">days
        <option value="w">weekdays
        <option value="ww">weeks
        <option value="h">hours
        <option value="n">minutes
        <option value="s">seconds
    </select>
</pre>

<input type="Submit" name=""><input type="Reset">
</form>
```

DateFormat

説明

米国の日付書式を使用して日付値を形式設定します。国際的な日付書式を使用する場合は、LSDateFormat を使用してください。

戻り値
指定したマスクに応じて形式設定された、日付を表す文字列。マスクを指定しないと、**dd-mmm-yy** の形式の値が返されます。

カテゴリ
日付および時刻関数

関数のシンタックス
```
DateFormat("date" [, "mask" ])
```

関連項目
Now、CreateDate、LSDateFormat、LSParseDateTime、LSTimeFormat、TimeFormat、ParseDateTime

履歴
ColdFusion MX: mask パラメータオプションの short、medium、long、および full がサポートされるようになりました。

パラメータ

パラメータ	説明
date	日付時刻オブジェクトです。値の範囲は西暦 100 ～ 9999 年です。
mask	ColdFusion で日付表示に使用される文字です。 • d: 数字で表した日です。1 桁の場合は先頭に 0 を付けません。 • dd: 数字で表した日です。1 桁の場合は先頭に 0 を付けます。 • ddd: 3 文字に短縮して表した曜日です。 • dddd: つづりを短縮しないで表した曜日です。 • m: 数字で表した月です。1 桁の場合は先頭に 0 を付けません。 • mm: 数字で表した月です。1 桁の場合は先頭に 0 を付けます。 • mmm: 3 文字に短縮して表した月です。 • mmmm: つづりを短縮しないで表した月です。 • yy: 下 2 桁の数字で表した年です。1 桁の場合は先頭に 0 を付けます。 • yyyy: 4 桁の数字で表した年です。 • gg: 時代および紀元を表す文字列です。無視されます。この値は予約されています。以下のマスクは、日付全体の形式設定を指定するものです。他のマスクと組み合わせることはできません。 • short: m/d/y と同等です。 • medium: mmm d, yyyy と同等です。 • long: mmmm d, yyyy と同等です。 • full: dddd, mmmm d, yyyy と同等です。

使用方法
日付時刻オブジェクトを文字列として渡すときは、そのオブジェクトを引用符で囲みます。引用符で囲まない場合、そのオブジェクトは日付時刻オブジェクトの数値表示として解釈されます。

注意：CreateDate 関数または Now 関数を、この関数の date パラメータとして渡すことができます。例 : #DateFormat(CreateDate(2001, 3, 3))#

日付および時刻の形式を設定する関数を使用しないと、データベースクエリーの結果の日付時刻値は、その順序と形式がばらばらになる可能性があります。表示された日付および時刻をアプリケーションのユーザーが正しく理解できるようにするため、この関数と、LSDateFormat、TimeFormat、および LSTimeFormat 関数を使用して、結果セットの値を形式設定することをお勧めします。詳細と例については、Web サイト go.adobe.com/kb/ts_tn_18070_en-us のテクニカルノート「**ColdFusion Server (5 and 4.5.x) with Oracle: Formatting Date and Time Query Results**」を参照してください。

注意：DateFormat 関数は、入力形式ではなく出力形式を設定する場合に適した関数です。入力形式を設定する場合は、日付時刻の作成関数 (CreateDate など) を使用してください。

例

```
<cfset todayDate = Now()>
<body>
<h3>DateFormat Example</h3>
<p>Today's date is <cfoutput>#todayDate#</cfoutput>.
<p>Using DateFormat, we can display that date in different ways:
<cfoutput>
<ul>
    <li>#DateFormat(todayDate)#
    <li>#DateFormat(todayDate, "mmm-dd-yyyy")#
    <li>#DateFormat(todayDate, "mmmm d, yyyy")#
    <li>#DateFormat(todayDate, "mm/dd/yyyy")#
    <li>#DateFormat(todayDate, "d-mmm-yyyy")#
    <li>#DateFormat(todayDate, "ddd, mmmm dd, yyyy")#
    <li>#DateFormat(todayDate, "short")#
    <li>#DateFormat(todayDate, "medium")#
    <li>#DateFormat(todayDate, "long")#
    <li>#DateFormat(todayDate, "full")#
</ul>
</cfoutput>
```

DatePart

説明

日付値から一部を抽出します。

戻り値

日付の一部を表す整数値

カテゴリ

日付および時刻関数

関数のシンタックス

```
DatePart("datepart", "date")
```

関連項目

DateAdd、DateConvert

履歴

ColdFusion MX 6.1: ミリ秒を表す文字 L および l が **datepart** に追加されました。

パラメータ

パラメータ	説明
datepart	文字列です。 • yyyy: 年 • q: 四半期 • m: 月 • y: 年間通算日 • d: 日 • w: 曜日 • ww: 週 • h: 時 • n: 分 • s: 秒 • l: ミリ秒
date	日付時刻オブジェクトです。値の範囲は西暦 100 ～ 9999 年です。

使用方法

日付時刻オブジェクトを文字列として渡すときは、そのオブジェクトを引用符で囲みます。引用符で囲まない場合、そのオブジェクトは日付時刻オブジェクトの数値表示として解釈されます。

例

```
<!--- This example shows information available from DatePart --->
<cfset todayDate = Now()>
<h3>DatePart Example</h3>
<p>Today's date is <cfoutput>#todayDate#</cfoutput>.
<p>Using datepart, we extract an integer representing the dateparts from that value
<cfoutput>
<ul>
    <li>year: #DatePart("yyyy", todayDate)#</li>
    <li>quarter: #DatePart("q", todayDate)#</li>
    <li>month: #DatePart("m", todayDate)#</li>
    <li>day of year: #DatePart("y", todayDate)#</li>
    <li>day: #DatePart("d", todayDate)#</li>
    <li>weekday: #DatePart("w", todayDate)#</li>
    <li>week: #DatePart("ww", todayDate)#</li>
    <li>hour: #DatePart("h", todayDate)#</li>
    <li>minute: #DatePart("n", todayDate)#</li>
    <li>second: #DatePart("s", todayDate)#</li>
</ul>
</cfoutput>
```

DateTimeFormat

説明

日付と時刻の形式設定規則を使用して、日付時刻値を形式設定します。

戻り値
形式設定された日付時刻値。

シンタックス
dateTimeFormat (***date***)

dateTimeFormat (***date [, mask***])

dateTimeFormat (***date [***, mask, ***timeZone***])

プロパティ

パラメータ	説明
date	必須です。日付時刻オブジェクトです。値の範囲は西暦 100 ～ 9999 年です。
mask	オプション。形式設定に使用するマスクです。
timeZone	タイムゾーン情報です。次のいずれかの形式で指定できます。 • GMT や PST などの省略形 • Europe/Dublin などの完全な名前 デフォルトでは、これは、システムに準拠したタイムゾーンです。

例
```
<cfset todayDateTime = Now()>
<body>
<h3>DateTimeFormat Example</h3>
<p>Today's date and time are <cfoutput>#todayDateTime#</cfoutput>.
<p>Using DateTimeFormat, we can display that date and time in different ways:
<cfoutput>
<ul>
    <li>#DateTimeFormat(todayDateTime)#
    <li>#DateTimeFormat(todayDateTime, "yyyy.MM.dd G 'at' HH:nn:ss z")#
    <li>#DateTimeFormat(todayDateTime, "EEE, MMM d, ''yy")#
    <li>#DateTimeFormat(todayDateTime, "h:nn a")#
    <li>#DateTimeFormat(todayDateTime, "hh 'o''clock' a, zzzz")#
    <li>#DateTimeFormat(todayDateTime, "K:nn a, z")#
    <li>#DateTimeFormat(todayDateTime, "yyyyy.MMMMM.dd GGG hh:nn aaa")#
    <li>#DateTimeFormat(todayDateTime, "EEE, d MMM yyyy HH:nn:ss Z")#
    <li>#DateTimeFormat(todayDateTime, "yyMMddHHnnssZ", "GMT")#
</ul>
</cfoutput>
```

Day

説明
月の何日目の日付であるかを調べます。

戻り値
月の第何日目かを返します。範囲は 1 ～ 31 です。

カテゴリ
日付および時刻関数

関数のシンタックス

```
Day("date")
```

関連項目

DayOfWeek、DayOfWeekAsString、DayOfYear、DaysInMonth、DaysInYear、FirstDayOfMonth

パラメータ

パラメータ	説明
date	日付時刻オブジェクトです。値の範囲は西暦 100 ～ 9999 年です。

使用方法

日付時刻オブジェクトを文字列として渡すときは、そのオブジェクトを引用符で囲みます。引用符で囲まない場合、そのオブジェクトは日付時刻オブジェクトの数値表示として解釈されます。

注意： CreateDate 関数または Now 関数をこの関数の date パラメータとして渡すことができます。例 :
#Day(CreateDate(2001, 3, 3))#

例

```
<h3>Day Example</h3>
<cfif IsDefined("FORM.year")>
    <p>More information about your date:
    <cfset yourDate = CreateDate(FORM.year, FORM.month, FORM.day)>
    <cfoutput>
        <p>Date: #DateFormat(yourDate)#. <br>
    It is #DayofWeekAsString(DayOfWeek(yourDate))#,
    day #DayOfWeek(yourDate)# in the week.
        <br>This is day #Day(YourDate)#
    in the month of #MonthAsString(Month(yourDate))#,
    which has #DaysInMonth(yourDate)# days.
        <br>We are in week #Week(yourDate)# of #Year(YourDate)#
        (day #DayofYear(yourDate)# of #DaysinYear(yourDate)#).
        <br><cfif IsLeapYear(Year(yourDate))>This is a leap year
                <cfelse>This is not a leap year</cfif>
    </cfoutput>
</cfif>
```

DayOfWeek

説明

日付の曜日を調べます。

戻り値

曜日を表す 1 (日曜日) ～ 7 (土曜日) の範囲の整数

カテゴリ

日付および時刻関数

関数のシンタックス

```
DayOfWeek("date")
```

関連項目

Day、DayOfWeekAsString、DayOfYear、DaysInMonth、DaysInYear、FirstDayOfMonth

パラメータ

パラメータ	説明
date	日付時刻オブジェクトです。値の範囲は西暦 100 ～ 9999 年です。

使用方法

日付時刻オブジェクトを文字列として渡すときは、そのオブジェクトを引用符で囲みます。引用符で囲まない場合、そのオブジェクトは日付時刻オブジェクトの数値表示として解釈されます。

注意：CreateDate 関数または Now 関数をこの関数の date パラメータとして渡すことができます。例 :
#DayOfWeek(CreateDate(2001, 3, 3))#

例

```
<h3>DayOfWeek Example</h3>
<cfif IsDefined("FORM.year")>
    More information about your date:
    <cfset yourDate = CreateDate(FORM.year, FORM.month, FORM.day)>
    <cfoutput>
    <p>Your date, #DateFormat(yourDate)#.
    <br>It is #DayofWeekAsString(DayOfWeek(yourDate))#, day
         #DayOfWeek(yourDate)# in the week.
    <br>This is day #Day(YourDate)# in the month of
         #MonthAsString(Month(yourDate))#, which has
     #DaysInMonth(yourDate)# days.
    <br>We are in week #Week(yourDate)# of #Year(YourDate)# (day
     #DayofYear(yourDate)# of #DaysinYear(yourDate)#).
    <br><cfif IsLeapYear(Year(yourDate))>This is a leap year
        <cfelse>This is not a leap year</cfif>
    </cfoutput>
</cfif>
```

DayOfWeekAsString

説明

日付の曜日を文字列関数として調べます。

戻り値

day_of_week に対応する、現在のロケールでの曜日の文字列

カテゴリ

日付および時刻関数、文字列関数

関数のシンタックス

```
DayOfWeekAsString(day_of_week [, locale])
```

関連項目

Day、DayOfWeek、DayOfYear、DaysInMonth、DaysInYear、FirstDayOfMonth

履歴
ColdFusion 8: locale パラメータが追加されました。

ColdFusion MX 7: 動作が変更されました。返される文字列は、現在のロケールの言語で表記されるようになりました。

パラメータ

パラメータ	説明
day_of_week	1 (日曜) ～ 7 (土曜) の範囲の整数です。
locale	関数を処理するときに、ページのロケールの代わりに使用するロケールです。

例
次の例では、DayOfWeekAsString 関数の使い方を示します。年、月、および日のフィールドを送信するフォームのアクションページです。

```
<h3>DayOfWeekAsString Example</h3>
<cfif IsDefined("FORM.year")>
More information about your date:
<cfset yourDate = CreateDate(FORM.year, FORM.month, FORM.day)>

<cfoutput>
<p>Your date, #DateFormat(yourDate)#.
<br>It is #DayofWeekAsString(DayOfWeek(yourDate))#, day
     #DayOfWeek(yourDate)# in the week.
<br>This is day #Day(YourDate)# in the month of
    #MonthAsString(Month(yourDate))#, which has
         #DaysInMonth(yourDate)# days.
<br>We are in week #Week(yourDate)# of #Year(YourDate)# (day #DayofYear(yourDate)#
    of #DaysinYear(yourDate)#).
<br><cfif IsLeapYear(Year(yourDate))>This is a leap year
    <cfelse>This is not a leap year</cfif>
</cfoutput>
</cfif>
```

DayOfYear

説明
日付が 1 年の第何日目かを調べます。

戻り値
年内の日を序数で表す整数値

カテゴリ
日付および時刻関数

関数のシンタックス
```
DayOfYear("date")
```

関連項目
Day、DayOfWeek、DayOfWeekAsString、DaysInMonth、DaysInYear、FirstDayOfMonth

パラメータ

パラメータ	説明
date	日付時刻オブジェクトです。値の範囲は西暦 100 ～ 9999 年です。

使用方法

この関数では閏年が識別されます。

日付時刻オブジェクトを文字列として渡すときは、そのオブジェクトを引用符で囲みます。引用符で囲まない場合、そのオブジェクトは日付時刻オブジェクトの数値表示として解釈されます。

注意： CreateDate 関数または Now 関数をこの関数の date パラメータとして渡すことができます。例：
#DayOfYear(CreateDate(2001, 3, 3))#

例

```
<h3>Day7OfYear Example</h3>
<cfif IsDefined("FORM.year")>
    More information about your date:
    <cfset yourDate = CreateDate(FORM.year, FORM.month, FORM.day)>
    <cfoutput>
    <p>Your date, #DateFormat(yourDate)#.
    <br>It is #DayofWeekAsString(DayOfWeek(yourDate))#,
         day #DayOfWeek(yourDate)# in the week.
    <br>This is day #Day(yourDate)# in the month of
     #MonthAsString(Month(yourDate))#, which has
     #DaysInMonth(yourDate)# days.
    <br>We are in week #Week(yourDate)# of #Year(yourDate)#
    (day #DayofYear(yourDate)# of #DaysinYear(yourDate)#).
    <br><cfif IsLeapYear(Year(yourDate))>This is a leap year
        <cfelse>This is not a leap year</cfif>
    </cfoutput>
</cfif>
```

DaysInMonth

説明

月の日数を調べます。

戻り値

date の月の日数

カテゴリ

日付および時刻関数

関数のシンタックス

```
DaysInMonth("date")
```

関連項目

Day、DayOfWeek、DayOfWeekAsString、DayOfYear、DaysInYear、FirstDayOfMonth

パラメータ

パラメータ	説明
date	日付時刻オブジェクトです。値の範囲は西暦 100 ～ 9999 年です。

使用方法

日付時刻オブジェクトを文字列として渡すときは、そのオブジェクトを引用符で囲みます。引用符で囲まない場合、そのオブジェクトは日付時刻オブジェクトの数値表示として解釈されます。

注意： Now 関数または CreateDate 関数をこの関数の date パラメータとして渡すことができます。例 : #DaysInMonth(CreateDate(2001, 3, 3))#

例

```
<h3>DaysInMonth Example</h3>
<cfif IsDefined("FORM.year")>
    More information about your date:
    <cfset yourDate = CreateDate(FORM.year, FORM.month, FORM.day)>
    <cfoutput>
    <p>Your date, #DateFormat(yourDate)#.
    <br>It is #DayofWeekAsString(DayOfWeek(yourDate))#, day
         #DayOfWeek(yourDate)# in the week.
    <br>This is day #Day(YourDate)# in the month of
         #MonthAsString(Month(yourDate))#, which has
     #DaysInMonth(yourDate)# days.
    <br>We are in week #Week(yourDate)# of #Year(YourDate)#
     (day #DayofYear(yourDate)# of #DaysinYear(yourDate)#).
    <br><cfif IsLeapYear(Year(yourDate))>This is a leap year
        <cfelse>This is not a leap year</cfif>
    </cfoutput>
</cfif>
```

DaysInYear

説明

年の日数を調べます。

戻り値

年の日数

カテゴリ

日付および時刻関数

関数のシンタックス

```
DaysInYear("date")
```

関連項目

Day、DayOfWeek、DayOfWeekAsString、DayOfYear、DaysInMonth、FirstDayOfMonth、IsLeapYear

パラメータ

パラメータ	説明
date	日付時刻オブジェクトです。値の範囲は西暦 100 ～ 9999 年です。

使用方法
DaysInYear では閏年が識別されます。

日付時刻オブジェクトを文字列として渡すときは、そのオブジェクトを引用符で囲みます。引用符で囲まない場合、そのオブジェクトは日付時刻オブジェクトの数値表示として解釈されます。

注意：CreateDate 関数または Now 関数を、この関数の date パラメータとして渡すことができます。例 : #DaysInYear(CreateDate(2001, 3, 3))#

例
```
<h3>DaysInYear Example</h3>
<cfif IsDefined("FORM.year")>
    More information about your date:
    <cfset yourDate = CreateDate(FORM.year, FORM.month, FORM.day)>
    <cfoutput>
    <p>Your date, #DateFormat(yourDate)#.
    <br>It is #DayofWeekAsString(DayOfWeek(yourDate))#, day
         #DayOfWeek(yourDate)# in the week.
    <br>This is day #Day(YourDate)# in the month of
         #MonthAsString(Month(yourDate))#, which has
     #DaysInMonth(yourDate)# days.
    <br>We are in week #Week(yourDate)# of #Year(yourDate)# (day
     #DayofYear(yourDate)# of #DaysinYear(yourDate)#).
</cfif>
```

DE

説明
パラメータ内の二重引用符 (") をエスケープし、結果を二重引用符で囲みます。

戻り値
二重引用符で囲まれたパラメータ。内部の二重引用符はエスケープされます。

カテゴリ
ダイナミック評価関数

関数のシンタックス
```
DE(string)
```

関連項目
Evaluate、IIf、PrecisionEvaluate、『ColdFusion アプリケーションの開発』の Using Expressions and Number Signs

パラメータ

パラメータ	説明
string	遅延の後に評価する文字列です。

使用方法
DE 関数は、IIf 関数、Evaluate 関数、または PrecisionEvaluate 関数にパラメータとして渡される文字列の評価を延期します。

通常は 2 番目と 3 番目のパラメータが式として自動的に評価される IIf 関数とともにこの関数を使用すると特に便利です。DE 関数を使用すると、変数として出力し、式として処理してはならない文字列パラメータの評価を避けることができます。次の例でその使い方を示します。IIF を使用して、テーブル行の背景色として白とグレーを交互に設定します。また、DE 関数を使用して、ColdFusion で色の文字列が評価されないようにします。

```
<cfoutput>
<table border="1" cellpadding="3">
<cfloop index="i" from="1" to="10">
    <tr bgcolor="#IIF( i mod 2 eq 0, DE("white"), DE("gray") )#">
        <td>
            hello #i#
        </td>
    </tr>
</cfloop>
</table>
</cfoutput>
```

DE 関数は、シャープ記号 (#) で囲まれた変数名の評価は遅らせません。ColdFusion 関数は、DE 関数が存在するかどうかに関係なく、変数を評価します。

次の例では、DE 関数とシャープ記号を一緒に使用する方法、およびこの関数を IIF 関数とともに使用する場合の動作を示します。

```
<cfoutput>
<cfset var1="Blue">
<cfset var2="Green">
<cfset myresult=IIf( 1 eq 2, DE(#Var1#), DE(#Var2#))>
The expression is #myresult#
</cfoutput>
```

ColdFusion は、このコードを次のように処理します。

1. ColdFusion は、変数 var1 と var2 が文字列 Blue と Green になるように設定します。
2. 4 行目で、ColdFusion はシャープ記号で囲まれた変数を最初に評価し、これらの変数の値を文字列 Blue と Green に置き換えます。
3. IIF 関数は、テスト式を評価して false であることを判別し、次に 3 番目のパラメータを評価します。
4. 3 番目のパラメータは DE 関数です。この関数は文字列 Green を取り、それを引用符で囲みます。
5. IIF 関数は文字列 "Green" (引用符を含む) を返します。
6. cfset タグは、式の結果として "Green" を取得し、myresult 変数の値を文字列 Green に設定します。
7. ColdFusion は、出力テキスト内の #myresult# を評価し、その値を文字列 Green に置き換え、その結果を表示します。

例

```
<!--- This example shows the use of DE and Evaluate --->
<h3>DE Example</h3>
<cfif IsDefined("FORM.myExpression")>
    <cftry>
        <!--- Show the expression and the results of evaluating it --->
        <cfoutput>
            <h3>Evaluate the Expression #FORM.MyExpression#</h3>
        </cfoutput>
        The code:<br>
        #Evaluate(FORM.myExpression)#
        <br><br>
        The result:<br>
        <cfoutput>
            #Evaluate(FORM.myExpression)#
        </cfoutput>

        <h3>Use DE to prevent the Evaluate function from evaluating</h3>
        The code:<br>
        #Evaluate(DE(FORM.MyExpression))#<br><br>
        The result:<br>
        <cfoutput>
            #Evaluate(DE(FORM.MyExpression))#
        </cfoutput>
        <!--- Error handling code for bad expressions and any other error.--->
        <cfcatch type = "Any">
        <!--- the message to display --->
            <h3>Sorry, there's been an <B>Error</B>.
            Try a simple expression, such as "2+2".</h3>
            <cfoutput>
            <!--- Display the diagnostic message from ColdFusion. --->
                <p>#cfcatch.message#
            </cfoutput>
        </cfcatch>
    </cftry>
</cfif>

<h3>Enter any valid ColdFusion expression</h3>
<cfform>
    <cfinput name="myExpression" type="Text" size="40">
    <cfinput type="submit" name="submitit">
</cfform>
```

DecimalFormat

説明

数値を 10 進数形式の文字列に変換します。

戻り値

number を、小数点以下 2 桁 (小数点以下 3 桁を四捨五入)、1000 単位のセパレータを使用して形式設定した文字列

カテゴリ

表示および書式制御関数

関数のシンタックス

```
DecimalFormat(number)
```

関連項目

DollarFormat、NumberFormat

パラメータ

パラメータ	説明
number	形式設定する数値です。

例

```
<h3>DecimalFormat Function</h3>
<p>Returns a number to two decimal places.
<p>
<cfloop FROM = 1 TO = 20 INDEX = "counter">
    <cfoutput>
    #counter# * Square Root of 2:
        #DecimalFormat(counter * sqr(2))#
    </cfoutput>
    <br>
</cfloop>
```

DecodeForHTML

説明

HTML エンコードされた文字列をデコードします。

戻り値

デコードされた HTML 文字列。

カテゴリ

表示および書式制御関数

関数のシンタックス

```
DecodeforHTML(String encodedinput)
```

関連項目

Canonicalize、EncodeForHTMLAttribute、EncodeForJavaScript、EncodeForCSS、EncodeForURL

履歴

ColdFusion 10: この関数が追加されました。

パラメータ

パラメータ	説明
inputString	必須です。デコードするエンコード文字列です。

例

```
<cfif isDefined("form.submit")>
    <b>
    Output:<cfoutput >#DecodeForHTML(form.encodedUserName)#</cfoutput>
    </b>
<cfelse>
    <cfset form.username=""/>
</cfif>
<cfform>
    <cfinput name="encodedUserName" type="text" value="#form.encodedUserName#">
    <cfinput name="submit" type="submit" value="Submit">
</cfform>
```

DecodeFromURL

説明

エンコードされた HTML URL 文字列をデコードします。

戻り値

デコードされた HTML URL 文字列。

カテゴリ

表示および書式制御関数

関数のシンタックス

```
DecodeFromURL(String encodedinput)
```

関連項目

Canonicalize、EncodeForHTMLAttribute、EncodeForJavaScript、EncodeForCSS、EncodeForURL

履歴

ColdFusion 10: この関数が追加されました。

パラメータ

パラメータ	説明
inputString	必須です。デコードするエンコード URL 文字列です。

例

```
<cfset string = "http://www.adobe.com/">
<cfset urlencoded = encodeforURL(string)>
<cfset urldecoded = decodefromUrl(urlEncoded)>
<cfoutput>
    String: #string# <br/>
    URL Encoded: #urlencoded#<br/>
    URL Decoded: #urldecoded#<br/>
</cfoutput>
```

DecrementValue

説明
数値の整数部をデクリメントします。

戻り値
number の整数部。1 だけデクリメントされます。

カテゴリ
算術関数

関数のシンタックス
```
DecrementValue(number)
```

関連項目
IncrementValue

パラメータ

パラメータ	説明
number	デクリメントする数値です。

例
```
<h3>DecrementValue Example</h3>
<p>Returns the integer part of a number decremented by one.</p>
<p>DecrementValue(0):
    <cfoutput>#DecrementValue(0)#</cfoutput></p>
<p>DecrementValue("1"):
    <cfoutput>#DecrementValue("1")#</cfoutput></p>
<p>DecrementValue(123.35):
    <cfoutput>#DecrementValue(123.35)#</cfoutput></p>
```

Decrypt

説明
Encrypt 関数によって暗号化された文字列など、標準の暗号化技術を使用して暗号化された文字列を復号します。

戻り値
復号された文字列

カテゴリ
セキュリティ関数、文字列関数

関数のシンタックス
```
Decrypt(encrypted_string, key[, algorithm, encoding, IVorSalt, iterations])
```

関連項目
Duplicate、Encrypt

履歴

ColdFusion 8: エンタープライズ版で RSA BSafe Crypto-J ライブラリを使用した暗号化がサポートされるようになりました。

ColdFusion MX 7.01: **IVorSalt** パラメータと **iterations** パラメータが追加されました。

ColdFusion MX 7: **algorithm** パラメータと **encoding** パラメータが追加されました。

パラメータ

パラメータ	説明
encrypted_string	復号する文字列です。
key	文字列。CFMX_COMPAT アルゴリズムの場合、これは文字列の暗号化に使用されたシードです。その他のアルゴリズムの場合は、generateSecretKey() メソッドで生成された文字列です。
algorithm	(オプション) エンタープライズ版の ColdFusion では、RSA BSafe Crypto-J ライブラリがインストールされます。このライブラリを使用すると、FIPS-140 に準拠した強力な暗号化を利用できます。アルゴリズムのリストについては、Encrypt 関数を参照してください。 標準版の ColdFusion では、次のアルゴリズムを使用する暗号ライブラリがインストールされます。 • CFMX_COMPAT: ColdFusion MX およびそれ以前のリリースで使用されるアルゴリズムです。このアルゴリズムは、最も安全性が低いオプションです (デフォルト)。 • AES: NIST (National Institute of Standards and Technology: 米国標準技術局) FIPS-197 で定義された Advanced Encryption Standard です。 • BLOWFISH: Bruce Schneier 氏が定義した Blowfish アルゴリズムです。 • DES: NIST FIPS-46-3 で定義された Data Encryption Standard アルゴリズムです。 • DESEDE: NIST FIPS-46-3 で定義された Triple DES アルゴリズムです。 これら以外の暗号アルゴリズムを使用するセキュリティプロバイダをインストールした場合は、その暗号化 / 復号アルゴリズムを指定することもできます。
encoding	(オプション) このパラメータを指定する場合は、algorithm パラメータも指定します。これは、データを文字列として表すために使用するバイナリエンコードです。文字列を暗号化するときに使用したものと同じアルゴリズムでなければなりません。 • Base64: IETF RFC 2045 で定義された Base64 アルゴリズムです。 • Hex: バイト値を 16 進数で表す、A ～ F および 0 ～ 9 の文字です。 • UU: UNIX 標準の UUEncode アルゴリズムです (デフォルト)。
IVorSalt	(オプション) このパラメータは、他の暗号化ソフトウェアの動作と一致するように ColdFusion の暗号化動作を調整する場合に使用します。このパラメータを指定する場合は、algorithm パラメータも指定します。 • ブロック暗号化アルゴリズムの場合 : アルゴリズムで使用するバイナリ初期化ベクター値です。このアルゴリズムには、ECB 以外のフィードバックモードが含まれている必要があります。これは、アルゴリズムのブロックサイズとまったく同じサイズのバイナリ値である必要があります。 • パスワードベースの暗号化アルゴリズムの場合 : パスワードをキーに変換するためのバイナリのソルト値です。
iterations	(オプション) パスワードをバイナリキーに変換するための繰り返し回数です。このパラメータは、他の暗号化ソフトウェアの動作と一致するように ColdFusion の暗号化動作を調整する場合に使用します。このパラメータを指定した場合は、algorithm パラメータで PBE (Password Based Encryption : パスワードベースの暗号化) アルゴリズムを指定します。ブロック暗号化アルゴリズムの場合は、このパラメータを指定しないでください。データの暗号化と復号には同じ値を使用します。

使用方法

この関数では、文字列の暗号化と復号に同じキーを使用する、シンメトリカルなキーベースのアルゴリズムが使用されます。パラメータ値は、文字列をエンコードするときに使用した値と一致する必要があります。暗号化された文字列のセキュリティは、キーの機密性によって異なります。

ColdFusion では、JCE (Java Cryptography Extension) を使用し、Sun JCE デフォルトセキュリティプロバイダを含む Sun Java ランタイムがインストールされます。このプロバイダには、前の「パラメータ」に示したアルゴリズムが含まれています。JCE フレームワークには、他のプロバイダの実装を使用するための機能も含まれています。ただし、Adobe ではサードパーティのセキュリティプロバイダに対するテクニカルサポートは提供していません。

例

```
<h3>Decrypt Example</h3>

<!--- Do the following if the form has been submitted. --->
<cfif IsDefined("Form.myString")>
    <cfscript>
        /* GenerateSecretKey does not generate key for the CFMX_COMPAT algorithm,
        so use the key from the form.
        */
        if (Form.myAlgorithm EQ "CFMX_COMPAT")
            theKey=Form.MyKey;
        // For all other encryption techniques, generate a secret key.
        else
            theKey=generateSecretKey(Form.myAlgorithm);
        //Encrypt the string
        encrypted=encrypt(Form.myString, theKey, Form.myAlgorithm,
            Form.myEncoding);
        //Decrypt it
        decrypted=decrypt(encrypted, theKey, Form.myAlgorithm, Form.myEncoding);
    </cfscript>

    <!--- Display the values used for encryption and decryption,
            and the results. --->
    <cfoutput>
        <b>The algorithm:</b> #Form.myAlgorithm#<br>
        <b>The key:</B> #theKey#<br>
        <br>
        <b>The string:</b> #Form.myString# <br>
        <br>
        <b>Encrypted:</b> #encrypted#<br>
        <br>
        <b>Decrypted:</b> #decrypted#<br>
    </cfoutput>
</cfif>

<!--- The input form.--->
<form action="#CGI.SCRIPT_NAME#" method="post">
    <b>Select the encoding</b><br>
    <select size="1" name="myEncoding">
        <option selected>UU</option>
        <option>Base64</option>
```

```
        <option>Hex</option>
    </select><br>
    <br>
    <b>Select the algorithm</b><br>
    <select size="1" name="myAlgorithm">
        <option selected>CFMX_COMPAT</option>
        <option>AES</option>
        <option>DES</option>
        <option>DESEDE</option>
    </select><br>
    <br>
    <b>Input your key</b> (used for CFMX_COMPAT encryption only)<br>
    <input type = "Text" name = "myKey" value = "MyKey"><br>
    <br>
    <b>Enter string to encrypt</b><br>
    <textArea name = "myString" cols = "40" rows = "5" WRAP = "VIRTUAL">
        This string is encrypted (you can replace it with more typing).
    </textArea><br>
    <input type = "Submit" value = "Encrypt my String">
</form>
```

DecryptBinary

説明

指定されたキー、値、アルゴリズム、ソルト、および繰り返しを使用して、暗号化されたバイナリデータを復号します。

戻り値

復号されたバイナリデータ

カテゴリ

セキュリティ関数、文字列関数

関数のシンタックス

```
DecryptBinary(bytes, key[, algorithm, IVorSalt, iterations])
```

関連項目

Duplicate、Encrypt、Decrypt

履歴

ColdFusion 8: エンタープライズ版で RSA BSafe Crypto-J ライブラリを使用した暗号化がサポートされるようになりました。

ColdFusion MX 7.01: この関数が追加されました。

パラメータ

パラメータ	説明
bytes	復号するバイナリデータです。
key	文字列。CFMX_COMPAT アルゴリズムの場合、これはバイナリデータの暗号化に使用されたシードです。その他のアルゴリズムの場合は、generateSecretKey() メソッドで生成された文字列です。
algorithm	(オプション) エンタープライズ版の ColdFusion では、RSA BSafe Crypto-J ライブラリがインストールされます。このライブラリを使用すると、FIPS-140 に準拠した強力な暗号化を利用できます。アルゴリズムのリストについては、Encrypt 関数を参照してください。 標準版の ColdFusion では、次のアルゴリズムを使用する暗号ライブラリがインストールされます。 • CFMX_COMPAT: ColdFusion MX およびそれ以前のリリースで使用されるアルゴリズムです。このアルゴリズムは、最も安全性が低いオプションです (デフォルト)。 • AES: NIST (National Institute of Standards and Technology: 米国標準技術局) FIPS-197 で定義された Advanced Encryption Standard です。 • BLOWFISH: Bruce Schneier 氏が定義した Blowfish アルゴリズムです。 • DES: NIST FIPS-46-3 で定義された Data Encryption Standard アルゴリズムです。 • DESEDE: NIST FIPS-46-3 で定義された Triple DES アルゴリズムです。 これら以外の暗号アルゴリズムを使用するセキュリティプロバイダをインストールした場合は、その暗号化 / 復号アルゴリズムを指定することもできます。
IVorSalt	(オプション) このパラメータは、他の暗号化ソフトウェアの動作と一致するように ColdFusion の暗号化動作を調整する場合に使用します。このパラメータを指定する場合は、algorithm パラメータも指定します。 • ブロック暗号化アルゴリズムの場合 : アルゴリズムで使用するバイナリ初期化ベクター値です。このアルゴリズムには、ECB 以外のフィードバックモードが含まれている必要があります。これは、アルゴリズムのブロックサイズとまったく同じサイズのバイナリ値である必要があります。 • パスワードベースの暗号化アルゴリズムの場合 : パスワードをキーに変換するためのバイナリのソルト値です。
iterations	(オプション) パスワードをバイナリキーに変換するための繰り返し回数です。このパラメータは、他の暗号化ソフトウェアの動作と一致するように ColdFusion の暗号化動作を調整する場合に使用します。このパラメータを指定した場合は、algorithm パラメータで PBE (Password Based Encryption : パスワードベースの暗号化) アルゴリズムを指定します。ブロック暗号化アルゴリズムの場合は、このパラメータを指定しないでください。データの暗号化と復号には同じ値を使用します。

使用方法

この関数では、データの暗号化と復号に同じキーを使用する、シンメトリカルなキーベースのアルゴリズムが使用されます。パラメータ値は、文字列をエンコードするときに使用した値と一致する必要があります。暗号化された文字列のセキュリティは、キーの機密性によって異なります。

ColdFusion では、JCE (Java Cryptography Extension) を使用し、Sun JCE デフォルトセキュリティプロバイダを含む Sun Java ランタイムがインストールされます。このプロバイダには、前の「パラメータ」に示したアルゴリズムが含まれています。JCE フレームワークには、他のプロバイダの実装を使用するための機能も含まれています。ただし、Adobe ではサードパーティのセキュリティプロバイダに対するテクニカルサポートは提供していません。

例

```
<h3>DecryptBinary Example</h3>
< !--- Do the following if the form has been submitted. --->
<cfif IsDefined("Form.myfile")>
<cffile file="#Form.myfile#" action="readBinary" variable="myData">
<cfscript>
/* GenerateSecretKey does not generate key for the CFMX_COMPAT algorithm,
so use the key from the form.
*/
if (Form.myAlgorithm EQ "CFMX_COMPAT")
theKey=Form.MyKey;
// For all other encryption techniques, generate a secret key.
else
theKey=generateSecretKey(Form.myAlgorithm);
//Encrypt the string
encrypted=encryptBinary(myData, theKey, Form.myAlgorithm);
//Decrypt it
decrypted=decryptBinary(encrypted, theKey, Form.myAlgorithm);
</cfscript>
<cfset encfile="#Form.myfile#" & "_enc">
<cfset decfile="#Form.myfile#" & "_dec">
<cffile file="#encfile#" action="write" output="#encrypted#">
<cffile file="#decfile#" action="write" output="#decrypted#">
<!--- Display the values used for encryption and decryption,
and the results. --->
<cfoutput>
<b>The algorithm:</b> #Form.myAlgorithm#<br>
<b>The key:</B> #theKey#<br>
<br>
<b>The InputFile:</b> #Form.myfile# <br>
<br>
<b>Encrypted:</b> #encfile#<br>
<br>
<b>Decrypted:</b> #decfile#<br>
</cfoutput>
</cfif>
<!--- The input form. --->
<form action="#CGI.SCRIPT_NAME#" method="post">
<b>Select the algorithm</b><br>
    <select size="1" name="myAlgorithm">
        <option selected>CFMX_COMPAT</option>
        <option>AES</option>
        <option>DES</option>
        <option>DESEDE</option>
</select><br>
<br>
<b>Input your key</b> (used for CFMX_COMPAT encryption only)<br>
<input type = "Text" name = "myKey" value = "MyKey"><br>
<br>
<b>Enter filename to encrypt</b><br>
<input type="text" name="myfile" value="Enter the path of the file to encrypt"><br>
<input type = "Submit" value = "Encrypt file ">
</form>
```

DeleteClientVariable

説明

クライアント変数を削除します (変数の存在の有無をテストするには、IsDefined を使用します)。

戻り値
変数を削除できた場合は true、削除できなかった場合は false

カテゴリ
その他の関数

関数のシンタックス
```
DeleteClientVariable("name")
```

関連項目
GetClientVariablesList

履歴
ColdFusion MX: 動作が変更されました。変数が存在しない場合、この関数は false を返すようになりました (以前のリリースでは、エラーを返していました)。

パラメータ

パラメータ	説明
name	削除するクライアント変数の名前です。二重引用符 (") で囲みます。

例
```
<!--- This view-only example shows DeleteClientVariable --->
<h3>DeleteClientVariable Example</h3>

<p>This view-only example deletes a client variable called "User_ID", if it
    exists in the list of client variables returned by GetClientVariablesList.
<p>This example requires the existence of an Application.cfm file and client
    management to be in effect.
<!---
<cfset client.somevar = "">
<cfset client.user_id = "">
<p>Client variable list:<cfoutput>#GetClientVariablesList()#</cfoutput>
<cfif ListFindNoCase(GetClientVariablesList(), "User_ID") is not 0>

        <cfset temp = DeleteClientVariable("User_ID")>
        <p>Was variable "User_ID" Deleted? <cfoutput>#temp#</cfoutput>
</cfif>
<p>Amended Client variable list:<cfoutput>#GetClientVariablesList()#
</cfoutput>
--->
```

DeserializeJSON

説明
JSON (JavaScript Object Notation) 文字列データ表現を CFML データ (CFML 構造体や CFML 配列など) に変換します。

戻り値
ColdFusion 形式のデータ値 : 構造体、配列、クエリー、単純値

カテゴリ
変換関数

シンタックス

```
DeserializeJSON(JSONVar[, strictMapping])
```

関連項目

IsJSON、SerializeJSON、cfajaxproxy、『ColdFusion アプリケーションの開発』の Using Ajax Data and Development Features、http://www.json.org

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
JSONVar	有効な JSON 構造を含む文字列、または JSON 構造を表す変数です。
strictMapping	JSON を厳密に変換するかどうかを指定するブール値です。 • true: (デフォルト) JSON 文字列を、その JSON データ型に直接対応する ColdFusion データ型に変換します。 • false: JSON 文字列に ColdFusion クエリーの表現が含まれているかどうかを判断し、含まれている場合はそれらの表現をクエリーに変換します。

使用方法

この関数は、ColdFusion ページでデータを JSON 文字列として受け取る場合に使用すると便利です。クライアントブラウザ上のデータ表現に Ajax を使用する ColdFusion アプリケーションでこの関数を使用すると、クライアントサイドの Ajax JavaScript からの JSON 形式データをサーバーで利用できます。また、この関数をページで使用すると、JSON パラメータを含む JavaScript 関数呼び出しとしてデータを提供するサービスからデータを取得できます。次の例は、このケースを示しています。

DeserializeJSON 関数は、次のように JSON データ型を ColdFusion データ型に直接変換します。

- strictMapping パラメータが true (デフォルト) の場合、JSON オブジェクトはすべて CFML 構造体に変換されます。
- strictMapping パラメータが false の場合は、JSON オブジェクトがクエリーを表しているかどうかを判断し、クエリーが表されている場合はそれらを ColdFusion クエリーオブジェクトに変換します。その他すべての JSON オブジェクトは ColdFusion 構造体に変換されます。SerializeJSON のリファレンスで説明されている 2 種類のクエリー表現のいずれかが JSON 構造体で使用されている場合、DeserializeJSON 関数はその構造体をクエリーとして認識して正しく変換します。
- JSON の配列、文字列、数値は、それぞれ ColdFusion の配列、文字列、数値に変換されます。
- JSON の null 値は、文字列 null に変換されます。
- 日付と時刻の JSON 文字列表現は文字列のまま変換されませんが、ColdFusion の日付時刻処理コードでは、それらを日付と時刻を示すデータとして認識できます。

例

次の例は、SerializeJSON 関数の例で生成された JSON 形式のデータフィードを基に天気情報を表示します。このようなコードでは、JavaScript ページとしてエクスポートされるデータを利用する場合もあります。このフィードは JavaScript 関数呼び出しの形式を持ち、フィードのデータを含む JSON 文字列がパラメータとして指定されています。この例では、次の処理を行います。

1. cfhttp タグを使用してフィードを取得します (cfhttp.fileContent 変数)。
2. テキストから関数呼び出しラッパーを取り除きます。
3. IsJSON 関数を使用して、前の処理の結果が有効な JSON 形式文字列であるかどうかを確認します。有効でない場合は、メッセージを表示して処理を停止します。

4 文字列が有効な JSON テキストである場合は、DeserializeJSON 関数を使用して文字列を ColdFusion 変数に変換します。この例では、ColdFusion クエリーを表す 2 つの配列を含む構造体に変換されます。最初の配列にはクエリー列名、2 つ目の配列にはクエリーデータが格納されます。

5 オブジェクトを解析し、配列の内容を表示します。

この例を実行するときは、このファイルと SerializeJSON 関数の例を ColdFusion の Web ルート下の適切な場所に置き、URL をシリアル化の例の URL に置き換えてから、このページを実行してください。

```
<!--- Get the JSON Feed --->
<cfhttp url="http://localhost:8500/My_Stuff/Ajax/Books/CreateJSON_NEW.cfm">

<!--- JSON data is sometimes distributed as a JavaScript function.
     The following REReplace functions strip the function wrapper. --->
<cfset theData=REReplace(cfhttp.FileContent,
        "^\s*[[:word:]]*\s*\(\s*",")>
<cfset theData=REReplace(theData, "\s*\)\s*$", ")>

<!--- Test to make sure you have JSON data. --->
<cfif !IsJSON(theData)>
    <h3>The URL you requested does not provide valid JSON</h3>
    <cfdump var="#theData#">

<!--- If the data is in JSON format, deserialize it. --->
<cfelse>
    <cfset cfData=DeserializeJSON(theData)>
    <!--- Parse the resulting array or structure and display the data.
             In this case, the data represents a ColdFusion query that has been
             serialized by the SerializeJSON function into a JSON structure with
             two arrays: an array column names, and an array of arrays,
             where the outer array rows correspond to the query rows, and the
             inner array entries correspond to the column fields in the row. --->
    <!--- First, find the positions of the columns in the data array. --->
    <cfset colList=ArrayToList(cfData.COLUMNS)>
    <cfset cityIdx=ListFind(colList, "City")>
    <cfset tempIdx=ListFind(colList, "Temp")>
    <cfset fcstIdx=ListFind(colList, "Forecasts")>
    <!--- Now iterate through the DATA array and display the data. --->
    <cfoutput>
        <cfloop index="i" from="1" to="#ArrayLen(cfData.DATA)#">
            <h3>#cfData.DATA[i][cityIdx]#</h3>
            Current Temperature: #cfData.DATA[i][tempIdx]#<br><br>
            <b>Forecasts</b><br><br>
            <cfloop index="j" from="1" to="#ArrayLen(cfData.DATA[i][fcstIdx])#">
                <b>Day #j#</b><br>
                Outlook: #cfData.DATA[i][fcstIdx][j].WEATHER#<br>
                High: #cfData.DATA[i][fcstIdx][j].HIGH#<br>
                Low: #cfData.DATA[i][fcstIdx][j].LOW#<br><br>
            </cfloop>
        </cfloop>
    </cfoutput>
</cfif>
```

DirectoryCopy

説明

ディレクトリの内容を宛先のディレクトリにコピーします。

戻り値
なし

シンタックス
directoryCopy (***source***, ***destination***[, recurse][, ***filter***])

プロパティ

パラメータ	説明
source	内容をコピーするソースディレクトリの絶対パス名です。
destination	宛先のディレクトリのパスです。絶対パスを指定しない場合は、ソースディレクトリを基準とした相対パスを指定します。
recurse	デフォルトは false です。true の場合はサブディレクトリをコピーします。
filter	適用するファイル拡張子のフィルターです（例：*.cfm）。

例
```
directoryCopy(sourceDirExists,destDirExists,true,"*.cfm")
```

DirectoryCreate

説明
ディスク上またはメモリ内にディレクトリを作成します。

カテゴリ
システム関数

関数のシンタックス
DirectoryCreate(path)

関連項目
DirectoryDelete、DirectoryExists、DirectoryList、DirectoryRename

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
path	作成するディレクトリの絶対パスです。あるいは、DirectoryCreate("//12.3.123.123/c_drive/test"); のように IP アドレスを指定することもできます。

使用方法
この関数の実行に必要な権限があることを確認してください。

例
次のコードは、ディレクトリの作成方法を示しています。

```
<h2>DirectoryCreate Example</h2>
 <h3>Enter a directory to create.</h3>
 <cfform action = "directorycreate.cfm" method="post" preservedata="true" >
    <cfinput type = "text" required="true"  name = "createDirectory">
    <br>
    <cfinput type = "submit" value="submit" name = "submit">
 </cfform>

 <cfif IsDefined("FORM.createDirectory")>
    <cfif FORM.createDirectory is not "">
    <cfset createDirectory = FORM.createDirectory>
        <cftry>
        <cfset DirectoryCreate(createDirectory)>
            <cfoutput><b>Directory #createDirectory# successfully created.</b></cfoutput>
         <cfcatch>
            <b>Error Message:</b><cfoutput>#cfcatch.message#</cfoutput><br/>
            <b>Error Detail:</b><cfoutput>#cfcatch.Detail#</cfoutput>
         </cfcatch>
        </cftry>
    </cfif>
 </cfif>
```

DirectoryDelete

説明
ディスク上またはメモリ内のディレクトリを削除します。

カテゴリ
システム関数

関数のシンタックス
DirectoryDelete(path [, recurse])

関連項目
DirectoryCreate、DirectoryExists、DirectoryList、DirectoryRename

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
path	削除するディレクトリの絶対パスです。あるいは、DirectoryDelete("//12.3.123.123/c_drive/test"); のように IP アドレスを指定することもできます。
recurse	このパラメータはオプションで、デフォルト値は false です。 true の場合は、指定したディレクトリとそのサブディレクトリが削除されます。 削除するディレクトリにサブディレクトリがある場合、recurse を false に設定すると、例外が発生します。

使用方法
この関数の実行に必要な権限があることを確認してください。

例

```
<h2>DirectoryDelete Example</h2>
 <h3>Enter a directory to delete.</h3>
 <form action = "directoryDelete.cfm" method="post">
    <label for="delDirectory">Directory Path: </label><input type = "text" id="delDirectory" name =
"delDirectory">
    <br>
    <label for="recurse">Recurse: </label><input id="recurse" type="checkbox" value="recurse"
name="recurse">
    <br>
    <input type = "submit" value="submit" name = "submit" onclick='return confirm("Are you sure !!!");'>
 </form>
 <cfif IsDefined("FORM.delDirectory")>
    <cfif FORM.delDirectory is not "">
    <cfset delDirectory = FORM.delDirectory>
    <cfset recurse = false>
    <cfif isDefined("FORM.recurse")>
        <cfset recurse = true>
    </cfif>
        <cftry>
        <cfset DirectoryDelete(delDirectory,recurse)>
            <cfoutput><p>Directory <b>#delDirectory#</b> has been deleted.</cfoutput>
         <cfcatch>
            <b>Error Message:</b><cfoutput>#cfcatch.message#</cfoutput><br/>
            <b>Error Detail:</b><cfoutput>#cfcatch.Detail#</cfoutput>
         </cfcatch>
        </cftry>
    </cfif>
 </cfif>
```

DirectoryExists

説明

ディスク上またはメモリ内のディレクトリが存在するかどうかを判別します。

戻り値

指定されたディレクトリが存在する場合は YES、存在しない場合は NO。

カテゴリ

システム関数

関数のシンタックス

```
DirectoryExists(absolute_path)
```

関連項目

DirectoryCreate、DirectoryDelete、DirectoryList、DirectoryRename

パラメータ

パラメータ	説明
absolute_path	ディスク上またはメモリ内の絶対パスです。あるいは、DirectoryExists("//12.3.123.123/c_drive/test"); のように IP アドレスを指定することもできます。

例

```
<h3>DirectoryExists Example</h3>
<h3>Enter a directory to check for existence.</h2>
<form action = "directoryexists.cfm" method="post">
    <input type = "text" name = "yourDirectory">
    <br>
    <input type = "submit" name = "submit">
</form>

<cfif IsDefined("FORM.yourDirectory")>
    <cfif FORM.yourDirectory is not "">
    <cfset yourDirectory = FORM.yourDirectory>
        <cfif DirectoryExists(yourDirectory)>
            <cfoutput>
            <p>Your directory exists. Directory name: #yourDirectory#
            </cfoutput>
        <cfelse>
            <p>Your directory does not exist.</p>
        </cfif>
    </cfif>
</cfif>
```

DirectoryList

説明

ディスク上またはメモリ内のディレクトリの内容を出力します。recurse が true に設定されている場合は、サブディレクトリの内容も出力します。

戻り値

listInfo パラメータに基づいて、次のようにディレクトリの内容を出力します。

- listInfo="query" の場合は、クエリーオブジェクト
- listInfo="name" の場合は、名前の配列
- listInfo="path" の場合は、パスの配列

カテゴリ

システム関数

関数のシンタックス

DirectoryList(path [,recurse] [,listInfo] [,filter] [,sort])

関連項目

DirectoryCreate、DirectoryDelete、DirectoryExists、DirectoryRename

パラメータ

パラメータ	説明
path	内容を出力するディレクトリの絶対パスです。あるいは、DirectoryList("//12.3.123.123/c_drive/test"); のように IP アドレスを指定することもできます。
recurse	ColdFusion でサブディレクトリ上のアクションを実行するかどうかを指定します。 true の場合は、すべてのサブディレクトリの内容も出力されます。
listInfo	• name : ファイルとディレクトリの名前の配列を返します。 • path : ファイルとディレクトリのパスの配列を返します。 • query : クエリーを返します。
filter	返される名前に適用するファイル拡張子のフィルタです。たとえば、*.cfm などです。適用できるフィルタは 1 つだけです。
sort	ディレクトリリストのソートに使用するクエリー列です。クエリー出力に含まれる列をカンマ区切りのリストで指定します。 列に条件を付ける場合は、次のいずれかの値を使用します。 • asc: 昇順 (a ～ z) のソートです。 • dec : 降順 (z ～ a) のソートです。 たとえば、次のようになります。 sort = "directory ASC, size DESC, datelastmodified"

使用方法

この関数の実行に必要な権限があることを確認してください。

例

次のコードは、ディレクトリの内容をダンプします。

```
<h2>DirectoryList Example</h2>
<h3>Enter a directory for Listing.</h3>
 <cfform action = "directoryList.cfm" method="post" preservedata="true" >
    <label for="listDirectory">Directory Path: </label><cfinput type = "text" id="listDirectory" name =
"listDirectory">
    <br />
    <label for="recurse">Recurse: </label><cfinput id="recurse" type="checkbox" value="recurse"
name="recurse">
    <br />
    <label for="listInfo">List Info: </label>
    <cfselect name="listInfo" id="listInfo">
      <option value="name">name</option>
      <option value="path">path</option>
      <option value="query">query</option>
    </cfselect>
    <br />
    <label for="filter">Filter: </label><cfinput id="filter" type="text" value="" name="filter">
    <br/>
    <input type = "submit" value="submit" name = "submit">
 </cfform>
```

```
<cfif IsDefined("FORM.listDirectory")>
   <cfif FORM.listDirectory is not "">
   <cfset listDirectory = FORM.listDirectory>
   <cfset recurse = false>
   <cfif isDefined("FORM.recurse")>
       <cfset recurse = true>
   </cfif>
   <cfset listInfo = FORM.listInfo>
   <cfset filter = FORM.filter>
       <cftry>
       <cfset res=  DirectoryList(listDirectory,recurse,listInfo,filter)>
           <cfoutput><b>Content of Directory #listDirectory#: </b></cfoutput>
           <cfdump var="#res#">
        <cfcatch>
           <b>Error Message:</b><cfoutput>#cfcatch.message#</cfoutput><br/>
           <b>Error Detail:</b><cfoutput>#cfcatch.Detail#</cfoutput>
        </cfcatch>
       </cftry>
   </cfif>
</cfif>
```

DirectoryRename

説明
ディスク上またはメモリ内のディレクトリの名前を変更します。

カテゴリ
システム関数

関数のシンタックス
DirectoryRename(currentName,newName)

関連項目
DirectoryCreate、DirectoryDelete、DirectoryExists、DirectoryList

使用方法
この関数の実行に必要な権限があることを確認してください。

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
currentName	名前を変更するディレクトリの絶対パスです。あるいは、DirectoryRename("//12.3.123.123/c_drive/test"); のように IP アドレスを指定することもできます。
newName	変更後の新しいディレクトリ名です。

例

```
<h2>DirectoryRename Example</h2>
 <h3>Enter a directory to rename.</h3>
 <cfform action = "directoryRename.cfm" method="post" preservedata="true" >
    <label for="renameDirectory">Directory to rename: </label><cfinput required="true" type = "text"
id="renameDirectory" name = "renameDirectory">
    <br/>
    <label for="newName">New Name: </label><cfinput required="true" type="text" id="newName" name =
"newName">
    <br/>
    <input type = "submit" value="submit" name="submit">
 </cfform>

 <cfif IsDefined("FORM.renameDirectory") and IsDefined("FORM.newName") >
    <cfif FORM.renameDirectory is not "" and FORM.newName is not "">
    <cfset renameDirectory = FORM.renameDirectory>
    <cfset newName = FORM.newName>
        <cftry>
        <cfset DirectoryRename(renameDirectory,newName)>
            <cfoutput><b>Directory #renameDirectory# renamed to newName</b></cfoutput>
         <cfcatch>
            <b>Error Message:</b><cfoutput>#cfcatch.message#</cfoutput><br/>
            <b>Error Detail:</b><cfoutput>#cfcatch.Detail#</cfoutput>
         </cfcatch>
        </cftry>
    </cfif>
 </cfif>
```

DollarFormat

説明

文字列を米国通貨形式に設定します (米国通貨以外については、LSCurrencyFormat または LSEuroCurrencyFormat を使用します)。

戻り値

小数点以下 2 桁で、1000 単位のセパレータ、およびドル記号を使用して形式設定した数値の文字列。**number** が負の数値の場合、戻り値は括弧で囲まれます。**number** が空の文字列の場合はゼロが返されます。

カテゴリ

表示および書式制御関数

関数のシンタックス

```
DollarFormat(number)
```

関連項目

DecimalFormat、NumberFormat

パラメータ

パラメータ	説明
number	形式設定する数値です。

例

```
<!--- This example shows the use of DollarFormat --->
...
<h3>DollarFormat Example</h3>
<cfloop from = 8 to = 50 index = counter>
    <cfset full = counter>
    <cfset quarter = counter + (1/4)>
    <cfset half = counter + (1/2)>
    <cfset threefourth = counter + (3/4)>
    <cfoutput>
    <pre>
    bill#DollarFormat(full)##DollarFormat(quarter)#
     #DollarFormat(half)# #DollarFormat(threefourth)#
    18% tip#DollarFormat(full * (18/100))#
     #DollarFormat(quarter * (18/100))#
     #DollarFormat(half * (18/100))#
     #DollarFormat(threefourth * (18/100))#
    </pre>
    </cfoutput>
</cfloop>
...
```

DotNetToCFType

説明

.NET メソッドから返された値を、その型に対応する ColdFusion データ型に明示的に変換します。

戻り値

ColdFusion データ値

カテゴリ

構造体関数、システム関数

関数のシンタックス

```
DotNetToCFType(variable_name)
```

関連項目

CreateObject: .NET オブジェクト、cfobject: .NET オブジェクト、『ColdFusion アプリケーションの開発』の Converting between .NET and ColdFusion data types

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
variable_name	変換する .NET 変数の名前です。

使用方法

この関数の用途と目的については、『ColdFusion アプリケーションの開発』の Working with complex .NET data types を参照してください。

例
次の例は、.NET の System.Data.DataTable オブジェクトを作成して、ColdFusion クエリーに変換します。

```
<!---Create a SQL Command Object--->
<cfobject action="create" name="sqlCommandObject"
    class="System.Data.SqlClient.SqlCommand" type=".Net"
    assembly="#assemblyList#">

<cfset sqlCommandObject.init("SELECT [ID], [FriendlyName] FROM [Batch]",
        sqlConnectionObject)>

<cfset sqlDataReaderObject = sqlCommandObject.ExecuteReader()>

<cfset dataTable = createObject(".net", "System.Data.DataTable",
        assemblyList)>
<!--- populate the datatable --->
<cfset dataTable.load(sqlDataReaderObject)>

<!--- convert to cfquery --->
<cfset myquery=DotNetToCFType(dataTable)>
```

Duplicate

説明
変数のクローン (ディープコピー) を返します。元の変数への参照はありません。

戻り値
変数のクローン

カテゴリ
構造体関数、システム関数

関数のシンタックス
```
Duplicate(variable_name)
```

関連項目
StructCopy、その他の構造体関数、『ColdFusion アプリケーションの開発』の Modifying a ColdFusion XML object

履歴
ColdFusion 8: 動作が変更されました。この関数で CFC をコピーできるようになりました。

ColdFusion MX: 動作が変更されました。この関数を XML オブジェクトに対して使用できるようになりました。

パラメータ

パラメータ	説明
variable_name	コピーを作成する変数の名前です。

使用方法
この関数を使用すると、ネストされた構造体やクエリーなどの複雑な構造体をコピーできます。

CFC インスタンスをコピーすると、Duplicate 関数を呼び出した時点で This スコープに存在していた変数の値を含め、CFC の内容全体がコピーされます。その後、2 つの CFC インスタンスは独立して存在するため、関数を呼び出して一方のコピーを変更しても、他方のコピーには影響しません。

注意：この関数では、cfobject タグや CreateObject 関数から返された COM、CORBA、JAVA のオブジェクトはコピーできません。配列要素や構造体フィールドが COM、CORBA、JAVA オブジェクトの場合、その配列や構造体はコピーできません。

例

```
<h3>Duplicate Example</h3>
<cfset s1 = StructNew()>
<cfset s1.nested = StructNew()>
<cfset s1.nested.item = "original">
<cfset copy = StructCopy(s1)>
<cfset clone = Duplicate(s1)>
<!--- modify the original --->
<cfset s1.nested.item = "modified">
<cfoutput>
<p>The copy contains the modified value: #copy.nested.item#</p>
<p>The duplicate contains the original value: #clone.nested.item#</p>
</cfoutput>
```

関数 e ～ g

EncodeForCSS

説明
入力文字列を CSS 用にエンコードします。

戻り値
エンコードされた文字列

カテゴリ
表示および書式制御関数

シンタックス
encodeForCSS(inputString [,strict])

関連項目
EncodeForHTML、EncodeForHTMLAttribute、EncodeForURL、Canonicalize

履歴
ColdFusion 10: この関数が追加されました。

パラメータ

パラメータ	説明
inputString	必須です。エンコードする文字列です。
strict	オプション。true に設定すると、複数のエンコードおよび混合エンコードは制限されます。

例

```
<cfif not isDefined ("form.bgcolor")>
    <cfset form.bgcolor = 'red'>
</cfif>
<cfoutput>
<style>
    .myDiv
    {
        background-color : #encodeForCSS(form.bgcolor)#;
        /* Encode the input to avoid any malicious code execution.*/
    }
</style>
</cfoutput>
<hr/>
<cfoutput>
    <div class="myDiv">
        This div element is styled!!!!
    </div>
</cfoutput>
<hr/>
<cfform action="#cgi.SCRIPT_NAME#" method="post" >
    Background Color : <cfinput name="bgcolor" type="text" value="#form.bgcolor#"> <br/><cfinput
name="submit" type="submit" value="Style the div!!!">
</cfform>
```

EncodeForHTML

説明

入力文字列を HTML 用にエンコードします。

戻り値

エンコードされた文字列

カテゴリ

表示および書式制御関数

シンタックス

encodeForHTML(inputString [,strict])

関連項目

Canonicalize、EncodeForHTMLAttribute、EncodeForJavaScript、EncodeForCSS、EncodeForURL

履歴

ColdFusion 10: この関数が追加されました。

パラメータ

パラメータ	説明
inputString	必須です。エンコードする文字列です。
strict	オプション。true に設定すると、複数のエンコードおよび混合エンコードは制限されます。

例

```
<cfif isDefined("form.submit")>
    <b>
    Output:<cfoutput >#encodeForHTML(form.userName)#</cfoutput>
    </b>
<cfelse>
    <cfset form.username=""/>
</cfif>
<cfform>
    <cfinput name="userName" type="text" value="#form.userName#">
    <cfinput name="submit" type="submit" value="Submit">
</cfform>
```

EncodeForHTMLAttribute

説明

入力文字列を HTML 属性用（table width、image height など）にエンコードします。

戻り値

エンコードされた文字列

カテゴリ

表示および書式制御関数

シンタックス

encodeForHTMLAttribute(inputString [,strict])

関連項目

Canonicalize、EncodeForJavaScript、EncodeForCSS、EncodeForURL、EncodeForHTML

履歴

ColdFusion 10: この関数が追加されました。

パラメータ

パラメータ	説明
inputString	必須です。エンコードする文字列です。
strict	オプション。true に設定すると、複数のエンコードおよび混合エンコードは制限されます。

例

```
<cfif not isDefined ("form.submit")>
    <cfset form.width = 200 />
</cfif>
<cfoutput >
    <table width="#encodeForHTMLAttribute(form.width)#" border="1" bgcolor="RED">
        <tr>
            <td>
                Enter the value in the below text field.
            </td>
        </tr>
    </table>
</cfoutput>
<cfform>
        <cfinput name="width" type="text" value="#form.width#">
        <cfinput name="submit" type="submit" value="Submit">
</cfform>
```

EncodeForJavaScript

説明

入力文字列を JavaScript 用にエンコードします。

戻り値

エンコードされた文字列

カテゴリ

表示および書式制御関数

シンタックス

encodeForJavaScript(inputString [,strict])

関連項目

Canonicalize、EncodeForHTMLAttribute、EncodeForHTML、EncodeForCSS、EncodeForURL

履歴

ColdFusion 10: この関数が追加されました。

パラメータ

パラメータ	説明
inputString	必須です。エンコードする文字列です。
strict	オプション。true に設定すると、複数のエンコードおよび混合エンコードは制限されます。

例

```
<cfif isDefined ("form.submit")>
<!--- If the user submits the form, show him/her a JavaScript alert. --->
        <cfoutput >
        <script type="text/javascript">
            alert('Hello #encodeForJavascript(form.userName)# !!!');
            // For security purpose, encode the user-generated input, so that it does not execute malicious
codes.
        </script>
        </cfoutput>
<cfelse >
    <cfset form.username = "" />
</cfif>
<cfform action="#cgi.SCRIPT_NAME#" method="post" >
    <cfinput name="userName" type="text" value="#form.userName#">
    <cfinput name="submit" type="submit" value="SayHello!!!">
</cfform>
```

EncodeForURL

説明

入力文字列を URL 用にエンコードします。

戻り値

エンコードされた文字列

カテゴリ

表示および書式制御関数

シンタックス

encodeForURL(inputString [,strict])

関連項目

Canonicalize、EncodeForHTMLAttribute、EncodeForHTML、EncodeForCSS、EncodeForJavaScript

履歴

ColdFusion 10: この関数が追加されました。

パラメータ

パラメータ	説明
inputString	必須です。エンコードする文字列です。
strict	オプション。true に設定すると、複数のエンコードおよび混合エンコードは制限されます。

例

```
<cfif not isDefined ("form.url")>
    <cfset form.url = "www.adobe.com">
</cfif>
<cfform action="#cgi.SCRIPT_NAME#" method="post">
    <cfinput name="url" type="text" value="#form.url#">
    <cfinput name="submit" type="submit" value="Show link to this URL!!!">
</cfform>
<hr />
<cfoutput >
    <b>LINK to URL:</b> <a target="_blank"
href="http://#encodeForURL(form.url)#">#encodeForURL(form.url)#</a>
</cfoutput>
```

EncodeForXML

説明

文字列を XML 用にエンコードします。

戻り値

エンコードされた XML 文字列。

カテゴリ

表示および書式制御関数

関数のシンタックス

```
encodeforXML(Inputstring , [strict])
```

関連項目

Canonicalize、EncodeForHTMLAttribute、EncodeForJavaScript、EncodeForCSS、EncodeForURL

履歴

ColdFusion 10: この関数が追加されました。

パラメータ

パラメータ	説明
inputString	必須です。エンコードする文字列です。
strict	オプション。true に設定すると、複数のエンコードおよび混合エンコードは制限されます。

例

```
<cfif isDefined("form.submit")>
    MyDoc = XmlNew();
    MyDoc.xmlRoot = XmlElemNew(MyDoc,"MyRoot");
    MyDoc.MyRoot.XmlText = #encodeForXML(form.userName)#;
    <b>
    Output:<cfoutput >#encodeForXML(form.userName)#</cfoutput>
    </b>
<cfelse>
    <cfset form.username=""/>
</cfif>
<cfform>
    <cfinput name="userName" type="text" value="#form.userName#">
    <cfinput name="submit" type="submit" value="Submit">
</cfform>
```

Encrypt

説明

特定のアルゴリズムおよびエンコード方法を使用して、文字列を暗号化します。

戻り値

文字列。元の文字列よりも長くなる場合があります。

カテゴリ

セキュリティ関数、文字列関数

関数のシンタックス

```
Encrypt(string, key [, algorithm, encoding, IVorSalt, iterations])
```

関連項目

Decrypt、EncryptBinary、DecryptBinary

履歴

ColdFusion 8: エンタープライズ版で RSA BSafe Crypto-J ライブラリを使用した暗号化がサポートされるようになりました。

ColdFusion MX 7.01: **IVorSalt** パラメータと **iterations** パラメータが追加されました。

ColdFusion MX 7: **algorithm** パラメータと **encoding** パラメータが追加されました。

パラメータ

パラメータ	説明
string	暗号化する文字列です。
key	文字列。文字列を暗号化するために使用するキーまたはシードです。 • CFMX_COMPAT アルゴリズムの場合、任意の数の文字を任意に組み合わせて、32 ビットの暗号化キーを生成するためのシードとして使用します。 • その他のアルゴリズムの場合は、そのアルゴリズムで使用する形式のキーです。これらのアルゴリズムでは、GenerateSecretKey 関数を使用してキーを生成します。
algorithm	(オプション) 文字列を暗号化するために使用するアルゴリズムです。 エンタープライズ版の ColdFusion では、RSA BSafe Crypto-J ライブラリがインストールされます。このライブラリを使用すると、FIPS-140 に準拠した強力な暗号化を利用できます。このライブラリでは、次のアルゴリズムを使用できます。 • AES: NIST (National Institute of Standards and Technology: 米国標準技術局) FIPS-197 で定義された Advanced Encryption Standard です。 • DES: NIST FIPS-46-3 で定義された Data Encryption Standard アルゴリズムです。 • DES-EDE: NIST FIPS-46-3 で定義された Triple DES アルゴリズムです。 • DESX: DES 対称暗号化アルゴリズムの拡張版です。
	• RC2: RFC 2268 で定義された RC2 ブロック対称暗号化アルゴリズムです。 • RC4: RC4 対称暗号化アルゴリズムです。 • RC5: RC5 暗号化アルゴリズムです。 • PBE: PKCS #5 で定義されたパスワードベースの暗号化アルゴリズムです。
	これらのアルゴリズムに加えて、標準版の ColdFusion に用意されているアルゴリズムも使用できます。
	標準版の ColdFusion では、次のアルゴリズムを使用する暗号ライブラリがインストールされます。 • CFMX_COMPAT: ColdFusion MX およびそれ以前のリリースで使用されるアルゴリズムです。このアルゴリズムは、最も安全性が低いオプションです (デフォルト)。 • AES: NIST (National Institute of Standards and Technology: 米国標準技術局) FIPS-197 で定義された Advanced Encryption Standard です。 • BLOWFISH: Bruce Schneier 氏が定義した Blowfish アルゴリズムです。 • DES: NIST FIPS-46-3 で定義された Data Encryption Standard アルゴリズムです。 • DESEDE: NIST FIPS-46-3 で定義された Triple DES アルゴリズムです。 これら以外の暗号アルゴリズムを使用するセキュリティプロバイダをインストールした場合は、その暗号化 / 復号アルゴリズムを指定することもできます。

パラメータ	説明
encoding	(オプション) このパラメータを指定する場合は、algorithm パラメータも指定します。これは、データを文字列として表すために使用するバイナリエンコードです。 • Base64: IETF RFC 2045 で定義された Base64 アルゴリズムです。 • Hex: バイト値を 16 進数で表す、A ～ F および 0 ～ 9 の文字です。 • UU: UUEncode アルゴリズムです (デフォルト)。
IVorSalt	(オプション) このパラメータは、他の暗号化ソフトウェアの動作と一致するように ColdFusion の暗号化動作を調整する場合に使用します。このパラメータを指定する場合は、*algorithm* パラメータも指定します。 • ブロック暗号化アルゴリズムの場合 : アルゴリズムで使用するバイナリ初期化ベクター値です。このアルゴリズムには、ECB 以外のフィードバックモードが含まれている必要があります。これは、アルゴリズムのブロックサイズとまったく同じサイズのバイナリ値である必要があります。データを正しく復号するには、Decrypt 関数で同じ値を使用します。 • パスワードベースの暗号化アルゴリズムの場合 : パスワードをキーに変換するためのバイナリのソルト値です。データを復号するときには同じ値を使用する必要する必要があります。
iterations	(オプション) パスワードをバイナリキーに変換するための繰り返し回数です。このパラメータは、他の暗号化ソフトウェアの動作と一致するように ColdFusion の暗号化動作を調整する場合に使用します。このパラメータを指定した場合は、algorithm パラメータで PBE (Password Based Encryption : パスワードベースの暗号化) アルゴリズムを指定します。ブロック暗号化アルゴリズムの場合は、このパラメータを指定しないでください。データの暗号化と復号には同じ値を使用します。

使用方法

この関数では、文字列の暗号化と復号に同じキーを使用する、シンメトリカルなキーベースのアルゴリズムが使用されます。暗号化された文字列のセキュリティは、キーの機密性によって異なります。

ColdFusion で使用される、RSA BSafe Crypto-J ライブラリに含まれる FIPS-140 認定アルゴリズムには次のものがあります。一部のアルゴリズムは Encrypt 関数では使用されませんが、他の関数で使用されます。

- AES - ECB、CBC、CFB (128)、OFB (128) - [128、192、256 ビットのキーサイズ]
- AES - CTR
- Diffie-Hellman Key Agreement
- DSA
- FIPS 186-2 General Purpose [(x-Change Notice); (SHA-1)]
- FIPS 186-2 [(x-Change Notice)、(SHA-1)]
- HMAC-SHAx (x は 1、224、256、384、または 512)
- RSA PKCS#1 v1.5 (署名、検証) (SHA-1、SHA-224、SHA-256、SHA-384、SHA-512)
- Secure Hash Standard (SHA-1、SHA-224、SHA-256、SHA-384、SHA-512)
- Triple DES - ECB、CBC、CFB (64 ビット)、および OFB (64 ビット)

RSA BSafe Crypto-J ライブラリに含まれるアルゴリズムは、エンタープライズ版で使用できます。一部のアルゴリズムを無効したいという場合もあります。DESX、RC5、および MD5PRNG アルゴリズムを無効にするには、ColdFusion Administrator の [Java と JVM] ページで次のように JVM 引数を指定します。

```
-Dcoldfusion.enablefipscrypto=true
```

JBoss の WebSphere で ColdFusion を実行している場合は、FIPS-140 認定の暗号化方式を使用できません。

エンタープライズ版で IBM Lotus Sametime の Instant Messaging Gateway を使用する場合は、ColdFusion Administrator の [Java と JVM の設定] ページで次のように JVM 引数を指定して、FIPS-140 認定アルゴリズム以外の暗号化方式も使用できるようにします。

```
-Dcoldfusion.disablejsafe=true
```

標準版では、デフォルトのアルゴリズム以外のすべてのアルゴリズムに対して JCE (Java Cryptography Extension) が使用され、Sun JCE のデフォルトセキュリティプロバイダを含む Sun Java ランタイムがインストールされます。このプロバイダには、前の「パラメータ」に示したアルゴリズムが含まれています。JCE フレームワークには、他のプロバイダの実装を使用するための機能も含まれています。ただし、Adobe ではサードパーティのセキュリティプロバイダに対するテクニカルサポートは提供していません。

デフォルトのアルゴリズムは、ColdFusion 5 および ColdFusion MX で使用されていたものと同じです。ユーザーが関数のパラメータとして渡すシードに基づいた擬似 32 ビットキーを使用する、XOR ベースのアルゴリズムが使用されます。このアルゴリズムは、使用可能な他のアルゴリズムに比べて安全性が劣ります。

例

次の例では、テキスト文字列を暗号化および復号します。暗号化アルゴリズムとエンコード方式を指定する必要があります。CFMX_COMPAT アルゴリズムで使用するキーシード用のフィールドもあります。CFMX_COMPAT 以外のすべてのアルゴリズムでは、シークレットキーが生成されます。

```
<h3>Encrypt Example</h3>
<!--- Do the following if the form has been submitted. --->
<cfif IsDefined("Form.myString")>
    <cfscript>
        /* GenerateSecretKey does not generate key for the CFMX_COMPAT algorithm,
        so use the key from the form.
        */
        if (Form.myAlgorithm EQ "CFMX_COMPAT")
            theKey=Form.MyKey;
        // For all other encryption techniques, generate a secret key.
        else
            theKey=generateSecretKey(Form.myAlgorithm);
        //Encrypt the string
        encrypted=encrypt(Form.myString, theKey, Form.myAlgorithm,
            Form.myEncoding);
        //Decrypt it
        decrypted=decrypt(encrypted, theKey, Form.myAlgorithm, Form.myEncoding);
    </cfscript>

    <!--- Display the values used for encryption and decryption,
            and the results. --->
    <cfoutput>
        <b>The algorithm:</b> #Form.myAlgorithm#<br>
        <b>The key:</B> #theKey#<br>
        <br>
        <b>The string:</b> #Form.myString# <br>
        <br>
        <b>Encrypted:</b> #encrypted#<br>
        <br>
        <b>Decrypted:</b> #decrypted#<br>
    </cfoutput>
</cfif>

<!--- The input form.--->
<form action="#CGI.SCRIPT_NAME#" method="post">
    <b>Select the encoding</b><br>
    <select size="1" name="myEncoding">
        <option selected>UU</option>
        <option>Base64</option>
```

```
        <option>Hex</option>
    </select><br>
    <br>
    <b>Select the algorithm</b><br>
    <select size="1" name="myAlgorithm">
        <option selected>CFMX_COMPAT</option>
        <option>AES</option>
        <option>DES</option>
        <option>DESEDE</option>
    </select><br>
    <br>
    <b>Input your key</b> (used for CFMX_COMPAT encryption only)<br>
    <input type = "Text" name = "myKey" value = "MyKey"><br>
    <br>
    <b>Enter string to encrypt</b><br>
    <textArea name = "myString" cols = "40" rows = "5" WRAP = "VIRTUAL">This string will be encrypted (you
can replace it with more typing).
    </textArea><br>
    <input type = "Submit" value = "Encrypt my String">
</form>
```

EncryptBinary

説明

特定のアルゴリズムおよびエンコード方法を使用して、バイナリデータを暗号化します。

戻り値

バイナリデータ

カテゴリ

セキュリティ関数、文字列関数

関数のシンタックス

```
EncryptBinary(bytes, key [, algorithm, IVorSalt, iterations])
```

関連項目

Decrypt、DecryptBinary、Encrypt

履歴

ColdFusion 8: エンタープライズ版で RSA BSafe Crypto-J ライブラリを使用した暗号化がサポートされるようになりました。

ColdFusion MX 7.01: この関数が追加されました。

パラメータ

パラメータ	説明
bytes	暗号化するバイナリデータです。
key	文字列。文字列を暗号化するために使用するキーまたはシードです。 • CFMX_COMPAT アルゴリズムの場合、任意の数の文字を任意に組み合わせて、32 ビットの暗号化キーを生成するためのシードとして使用します。 • その他のアルゴリズムの場合は、そのアルゴリズムで使用する形式のキーです。これらのアルゴリズムでは、GenerateSecretKey 関数を使用してキーを生成します。
algorithm	(オプション) 文字列を復号するために使用するアルゴリズムです。 エンタープライズ版の ColdFusion では、RSA BSafe Crypto-J ライブラリがインストールされます。このライブラリを使用すると、FIPS-140 に準拠した強力な暗号化を利用できます。アルゴリズムのリストについては、Encrypt 関数を参照してください。 標準版の ColdFusion では、次のアルゴリズムを使用する暗号ライブラリがインストールされます。 • CFMX_COMPAT: ColdFusion MX およびそれ以前のリリースで使用されるアルゴリズムです。このアルゴリズムは、最も安全性が低いオプションです (デフォルト)。 • AES: NIST (National Institute of Standards and Technology: 米国標準技術局) FIPS-197 で定義された Advanced Encryption Standard です。 • BLOWFISH: Bruce Schneier 氏が定義した Blowfish アルゴリズムです。 • DES: NIST FIPS-46-3 で定義された Data Encryption Standard アルゴリズムです。 • DESEDE: NIST FIPS-46-3 で定義された Triple DES アルゴリズムです。 これら以外の暗号アルゴリズムを使用するセキュリティプロバイダをインストールした場合は、その暗号化 / 復号アルゴリズムを指定することもできます。
IVorSalt	(オプション) このパラメータは、他の暗号化ソフトウェアの動作と一致するように ColdFusion の暗号化動作を調整する場合に使用します。このパラメータを指定する場合は、*algorithm* パラメータも指定します。 • ブロック暗号化アルゴリズムの場合 : アルゴリズムで使用するバイナリ初期化ベクター値です。このアルゴリズムには、ECB 以外のフィードバックモードが含まれている必要があります。これは、アルゴリズムのブロックサイズとまったく同じサイズのバイナリ値である必要があります。データを正しく復号するには、Decrypt 関数で同じ値を使用します。 • パスワードベースの暗号化アルゴリズムの場合 : パスワードをキーに変換するためのバイナリのソルト値です。データを復号するときには同じ値を使用する必要する必要があります。
iterations	(オプション) パスワードをバイナリキーに変換するための繰り返し回数です。このパラメータは、他の暗号化ソフトウェアの動作と一致するように ColdFusion の暗号化動作を調整する場合に使用します。このパラメータを指定した場合は、*algorithm* パラメータで PBE (Password Based Encryption : パスワードベースの暗号化) アルゴリズムを指定します。ブロック暗号化アルゴリズムの場合は、このパラメータを指定しないでください。データの暗号化と復号には同じ値を使用します。

使用方法

この関数では、バイナリデータの暗号化と復号に同じキーを使用する、シンメトリカルなキーベースのアルゴリズムが使用されます。暗号化されたデータのセキュリティは、キーの機密性によって異なります。

デフォルトのアルゴリズム以外のすべてのアルゴリズムについて、ColdFusion では JCE (Java Cryptography Extension) を使用し、Sun JCE デフォルトセキュリティプロバイダを含む Sun Java ランタイムがインストールされます。このプロバイダには、前の「パラメータ」に示したアルゴリズムが含まれています。JCE フレームワークには、他のプロバイダの実装を使用するための機能も含まれています。ただし、Adobe ではサードパーティのセキュリティプロバイダに対するテクニカルサポートは提供していません。

デフォルトのアルゴリズムは、ColdFusion 5 および ColdFusion MX で使用されていたものと同じです。ユーザーが関数のパラメータとして渡すシードに基づいた擬似 32 ビットキーを使用する、XOR ベースのアルゴリズムが使用されます。このアルゴリズムは、使用可能な他のアルゴリズムに比べて安全性が劣ります。

例

次の例では、バイナリデータを暗号化および復号します。ファイルに含まれているバイナリデータを暗号化し、暗号化されたファイルを復号します。暗号化アルゴリズムとエンコード方式を指定する必要があります。CFMX_COMPAT アルゴリズムで使用するキーシード用のフィールドもあります。CFMX_COMPAT 以外のすべてのアルゴリズムでは、シークレットキーが生成されます。

```
<h3>EncryptBinary Example</h3>
<!--- Do the following if the form has been submitted. --->
<cfif IsDefined("Form.myfile")>

<cffile file="#Form.myfile#" action="readBinary" variable="myData">
<cfscript>
/* GenerateSecretKey does not generate key for the CFMX_COMPAT algorithm,
so use the key from the form.
*/
if (Form.myAlgorithm EQ "CFMX_COMPAT")
theKey=Form.MyKey;
// For all other encryption techniques, generate a secret key.
else
theKey=generateSecretKey(Form.myAlgorithm);
//Encrypt the string
encrypted=encryptBinary(myData, theKey, Form.myAlgorithm);
//Decrypt it
decrypted=decryptBinary(encrypted, theKey, Form.myAlgorithm);
</cfscript>
<cfset encfile="#Form.myfile#" & "_enc">
<cfset decfile="#Form.myfile#" & "_dec">
<cffile file="#encfile#" action="write" output="#encrypted#">
<cffile file="#decfile#" action="write" output="#decrypted#">

<!--- Display the values used for encryption and decryption,
and the results. --->
<cfoutput>
<b>The algorithm:</b> #Form.myAlgorithm#<br>
<b>The key:</B> #theKey#<br>
<br>
<b>The InputFile:</b> #Form.myfile# <br>
<br>
<b>Encrypted:</b> #encfile#<br>
<br>
<b>Decrypted:</b> #decfile#<br>
```

```
</cfoutput>
</cfif>

<!--- The input form. --->
<form action="#CGI.SCRIPT_NAME#" method="post">
<b>Select the algorithm</b><br>
<select size="1" name="myAlgorithm">
<option selected>CFMX_COMPAT</option>
<option>AES</option>
<option>DES</option>
<option>DESEDE</option>
</select><br>
<br>
<b>Input your key</b> (used for CFMX_COMPAT encryption only)<br>
<input type = "Text" name = "myKey" value = "MyKey"><br>
<br>
<b>Enter filename to encrypt</b><br>
<input type="text" name="myfile" value="Enter the path of the file to encrypt"><br>
<input type = "Submit" value = "Encrypt file ">
</form>
```

EntityDelete

説明
指定したエンティティに対応するレコードをデータベースから削除します。マッピングで指定されている cascade 属性によっては、そのエンティティに関連付けられているオブジェクトも削除されます。

カテゴリ
ORM 関数

関数のシンタックス
`EntityDelete(`**`entity`**`)`

関連項目
EntityLoad、EntitySave

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
entity	削除されるエンティティの名前です。

例
```
<cfset employee = EntityLoad('employee', 100, true)>
<cfset EntityDelete(employee)>
<cfset employee = CreateObject('component', 'employee')>
<cfset employee.setEmployeeID(100)>
<cfset EntityDelete(employee)>
```

EntityLoad

説明
指定されたエンティティ名を持つエンティティの配列をロードして返します。フィルタ条件とソート順を指定することもできます。すべての EntityLoad メソッドは、入力としてエンティティ名を取ります。

戻り値
配列 (unique=false の場合)

単一のエンティティ (unique=true の場合)

カテゴリ
ORM 関数

関数のシンタックス
```
EntityLoad (entityname)
EntityLoad (entityname, id [, unique])
EntityLoad (entityname, filtercriteria [,unique]
EntityLoad(entityname, filtercriteria, sortorder [, options])
```

関連項目
EntityLoadByExample、EntityReload、EntityDelete

パラメータ

パラメータ	説明
entity name	ロードされるエンティティの名前です。
ID	ロードする必要があるエンティティのプライマリキーの値です。 エンティティに複合キーがある場合は、キーと値のペア (ColdFusion 構造体) として ID を指定する必要があります。
unique	unique が true に設定されている場合は、エンティティが返されます。 この filtercriteria に一致するレコードが 1 件だけであることがわかっている場合は、unique=true を指定すると、配列ではなく単体のエンティティが返されます。 unique=true を設定した場合に、複数のレコードが返されると、例外が発生します。

パラメータ	説明
filtercriteria	プロパティ名と値のキーと値のペア (ColdFusion 構造体) です。キーと値のペアが複数ある場合は、AND 演算子が使用されます。このパラメータが指定されている場合は、filtercriteria に一致する名前を持つエンティティの配列がロードされて返されます。
sortorder	返されるエンティティのソート順を指定するための文字列です。このパラメータが指定されている場合は、sortorder に従ってソートされた filtercriteria に一致するエンティティの配列がロードされて返されます。
options	次のオプションを使用して出力をカスタマイズできます。 • ignorecase : true に設定した場合、ソート順の文字列の大文字と小文字が区別されなくなります。sortorder パラメータを指定する場合にのみ使用します。 • offset : オブジェクトを取得する位置を指定します。 • maxResults: 取得されるオブジェクトの最大数を指定します。 • cacheable : 結果をセカンダリキャッシュに格納する必要があるかどうかを指定します。デフォルト値は false です。 • cachename: セカンダリキャッシュ内のキャッシュの名前です。 • timeout : クエリーのタイムアウト値 (秒) を指定します。

履歴
ColdFusion 9: この関数が追加されました。

使用方法
ページ割りを行う場合は、次の例に示すように offset および maxResults オプションを使用できます。

```
EntityLoad('employee', {department='qa'} , {offset=21, maxResults=10})
```

この例では、qa 部門に属する従業員のオブジェクトを、オフセット 22 より後の位置から 10 個取得します。

例
エンティティ名のみを指定する場合は、次のようにします。

```
<cfset employees = EntityLoad('employee')>
```

EntityName と ID を指定し、unique を true に設定した例です。unique を true ではなく false に設定すると、エンティティを 1 つだけ含む配列が返されます。

```
<cfset employee = EntityLoad('employee', 100, true)>
Entity name, composite key.
<cfset orderDetail = EntityLoad('orderdetails', {OrderID=100, ProductID=1}, true)>
```

国が UK のオブジェクトを Department の昇順で並べ替え、その後に Age の降順で並べ替えて取得するには、次のようにします。

```
<cfset employeesInUKSorted = EntityLoad('employee',
{country="UK"}, "Department Asc, Age Desc")>
```

"UK" に居住するすべての従業員の詳細を取得するには、次のようにします。

```
<cfset employeesFromUK = EntityLoad('employee', {country="UK"}>
```

一意のオブジェクトを 1 つだけ取得するには、次のようにします。unique= "true" を指定し、複数のオブジェクトが条件を満たす場合は、例外が発生します。

```
<cfset employee = EntityLoad('employee', {firstname="Marcia", lastname="Em"}, "true")>
```

EntityLoadByExample

説明
sampleentity と一致するオブジェクトの配列をロードして返します。フィルタ条件は、sampleentity で指定された null 以外のプロパティを AND 演算して作成されます。

戻り値
オブジェクトの配列

カテゴリ
ORM 関数

関数のシンタックス
```
entityloadbyexample(sampleentity [, unique])
```

関連項目
EntityLoad、EntityReload

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
sampleentity	ロードする類似エンティティの検索およびフィルタ処理に使用するサンプルエンティティの名前です。

例
```
<cfset employee= CreateObject("component", "employee")>
<cfset employee.setDepartment("ColdFusion")>
<cfset employee.setCountry("USA")>
<cfset employee=EntityLoadByExample(employee)>
```

EntityLoadByPK

説明
指定されたプライマリキーのオブジェクトの配列をロードして返します。EntityLoad() 関数で使用する必要があるブール値パラメータを指定しないようにするには、この関数を使用します。

戻り値
オブジェクト

カテゴリ
ORM 関数

関数のシンタックス
```
entityLoadByPK( entityName ,id)
```

パラメータ

パラメータ	説明
entity name	ロードされるエンティティの名前です。
id	プライマリキー ID

関連項目

EntityLoad、EntityReload、EntityLoadByExample、EntityDelete、『ColdFusion アプリケーションの開発』の ColdFusion ORM

履歴

ColdFusion 9: この関数が追加されました。

例

```
<cfscript>
    art = EntityLoadByPK("Art", 1);
    writedump(art);
</cfscript>
```

EntityMerge

説明

指定されたエンティティを現在の ORM セッションに結合します。指定されたオブジェクトの状態を、同じ識別子を持つ永続オブジェクトにコピーし、その永続オブジェクトを返します。

セッションに現在関連付けられている永続インスタンスがない場合は、インスタンスがロードされます。このインスタンスはセッションに関連付けられていません。ユーザーは、このセッションから返されるオブジェクトを使用する必要があります。

戻り値

オブジェクト

カテゴリ

ORM 関数

関数のシンタックス

```
entityMerge( entity)
```

パラメータ

パラメータ	説明
entity	ORM セッションに結合する必要があるエンティティです。

関連項目

EntityLoad、EntityLoadByExample、EntityDelete、『ColdFusion アプリケーションの開発』の ColdFusion ORM

履歴

ColdFusion 9: この関数が追加されました。

EntityNew

説明
指定したエンティティ名を持つ永続 CFC のインスタンスを作成します。

戻り値
オブジェクト

カテゴリ
ORM 関数

関数のシンタックス
entityNew(entityName [,properties])

パラメータ

パラメータ	説明
entityName	永続 CFC のエンティティ名です。
properties	プロパティ名と値のキーと値のペア (ColdFusion 構造体) です。

関連項目
EntityLoad、EntityLoadByExample、EntityDelete、『ColdFusion アプリケーションの開発』の ColdFusion ORM

履歴
ColdFusion 9: この関数が追加されました。

使用方法
作成対象のオブジェクトをアプリケーションで初期化できます。properties は構造体で、プロパティ名のキーが含まれます。オブジェクトが作成されると、渡された構造体がすべてのプロパティに挿入されます。

次に例を示します。

```
cfset artistObj = entityNew("Artists",{FirstName="Tom",LastName="Ron"})>
```

例
```
newArtistObj = EntityNew("Artists");
newArtistObj.setfirstname("John");
newArtistObj.setlastname("Smith");
newArtistObj.setaddress("5 Newport lane");
newArtistObj.setcity("San Francisco");
newArtistObj.setstate("CA");
newArtistObj.setPostalCode("90012");
newArtistObj.setphone("612-832-2343");
newArtistObj.setfax("612-832-2344");
newArtistObj.setemail("jsmith@company.com");
newArtistObj.setThePassword("jsmith");
EntitySave(newartistobj);
ormflush();
```

EntityReload

説明
既にロードしたエンティティのデータを再ロードします。このメソッドを使用すると、データベースからデータが再取得され、エンティティに最新のデータが反映されます。

カテゴリ
ORM 関数

関数のシンタックス
```
entityreload()
```

関連項目
EntityLoad、EntityLoadByExample、EntityDelete

履歴
ColdFusion 9: この関数が追加されました。

EntitySave

説明
指定されたエンティティとそれに関連するすべてのエンティティのデータがデータベースに保存されます (既存のデータがある場合は更新されます)。指定されたエンティティに関して新しいレコードを挿入するか、既存のレコードを更新するかは、ColdFusion によって自動的に判断されます。forceinsert=true を設定した場合、ColdFusion は常に新しいレコードとしてエンティティを挿入しようと試みます。

戻り値
void

カテゴリ
ORM 関数

関数のシンタックス
```
EntitySave(entity, [forceinsert])
```

関連項目
EntityLoad、EntityLoadByExample、EntityDelete

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
entity	データベースに保存する必要があるエンティティの名前です。
forceinsert	true の場合、ColdFusion は常に新しいレポートしてエンティティを挿入しようとします。

例

エンティティを保存するには：

```
<cfset employee = createObject("Employee")>
<cfset employee.setFirstName("Marcia")>
<cfset employee.setlastName("Em")>
<cfset EntitySave(employee)>
```

エンティティを更新するには：

```
<cfset employee = EntityLoad('employee', 100, true)>
<cfset employee.setLastName("Em")>
<cfset EntitySave(employee)>
```

EntitytoQuery

説明

入力として指定されたエンティティオブジェクトまたはエンティティオブジェクトの配列を、クエリーオブジェクトに変換します。

戻り値

クエリー

カテゴリ

ORM 関数

関数のシンタックス

```
EntitytoQuery (orm_object, [entity_name])
EntitytoQuery (orm_object_array, [entity_name])
```

パラメータ

パラメータ	説明
orm_object	クエリーオブジェクトに変換する必要があるエンティティオブジェクトです。
orm_object_array	クエリーオブジェクトに変換する必要がある配列です。
entity_name	エンティティの名前です。このオプションパラメータは、継承マッピングの場合に、指定したエンティティのクエリーを返す必要があるとき使用します。

関連項目

EntityLoad、EntityDelete

履歴

ColdFusion 9: この関数が追加されました。

使用方法

この関数には次の条件があります。

- 入力として配列を使用する場合は、配列に含まれるすべてのオブジェクトが同じ型でなければなりません。
- 結果のクエリーには関係データは含まれません。

例

```
<cfset artists = EntityLoad("Artist")>
<cfset artistQuery = EntityToQuery(artists)>
```

Evaluate

説明

文字列式をダイナミックに左から右へ評価します (左部分の評価結果が右部分に対して意味を持つことがあります)。右端にある式の評価結果が返されます。

戻り値

評価結果のオブジェクト

カテゴリ

ダイナミック評価関数

関数のシンタックス

```
Evaluate(string_expression1 [, string_expression2 , … ])
```

関連項目

DE、IIf、PrecisionEvaluate、『ColdFusion アプリケーションの開発』の Using Expressions and Number Signs

パラメータ

パラメータ	説明
string_expression1、 string_expression2...	評価対象となる式です。

使用方法

複雑な文字列式でも評価できます。文字列式に引用符または二重引用符が含まれている場合は、その引用符をエスケープする必要があります。

この関数は、複数の変数から 1 つの変数を形成するのに役立ちます。たとえば、行全体にわたるインデックス値を使用して、var 変数でクエリー qNames の列を参照するには、次のコードを使用します。

```
<cfset var=Evaluate("qNames.#colname#[#index#]")>
```

例

```
<!--- This example shows the use of PrecisionEvaluate and DE functions.--->
<h3>Evaluate Example</h3>
<cfif IsDefined("FORM.myExpression")>
    <cftry>
        <!--- Evaluate the expression --->
        <cfset theExpression = Evaluate(Form.myExpression)>
        <cfoutput>
            <!--- The DE function prevents the Evaluate function from evaluating
                     the expression. --->
            The value of the expression #Evaluate(DE(FORM.MyExpression))#
            is #theExpression#.<br>
            <!--- The following line does not use the DE function. --->
            The value of the expression #FORM.MyExpression#
            is #theExpression#.<br>
        </cfoutput>

        <cfcatch type="application">
            <cfoutput>Could not evaluate the expression #Form.myExpression#.<br>
                    Make sure you enter a valid ColdFusion Expression.
            </cfoutput>
        </cfcatch>
    </cftry>
</cfif>

<cfform preservedata="yes">
    <h3>Enter a ColdFusion expression for evaluation</h3>
    <cfinput type="text" name="myExpression" size="60"><br />
    <br />
    <cfinput type="submit" name="submit">
</cfform>
```

Exp

説明

number を表す底 e の指数を計算します。定数 e は、2.71828182845904 で、自然対数の底となります。この関数は、**number** の自然対数を求める Log の逆関数です。

戻り値

定数 e の **number** 乗。

カテゴリ

算術関数

関数のシンタックス

```
Exp(number)
```

関連項目

Log、Log10

パラメータ

パラメータ	説明
number	底 e に適用される指数です。

使用方法
他の底の累乗を計算するには対数演算子 (^) を使用します。

例
```
<h3>Exp Example</h3>
<cfif IsDefined("FORM.Submit")>
    <cfoutput>
    <p>Your number, #FORM.number#
    <br>#FORM.number# raised to the E power: #exp(FORM.number)#
    <cfif FORM.number LTE 0>
        <br>You must enter a positive real number to see its natural logarithm
        <cfelse><br>
        The natural logarithm of #FORM.number#: #log(FORM.number)#
    </cfif>
    <cfif FORM.number LTE 0><br>
        You must enter a positive real number to see its logarithm to base 10
        <cfelse><br>
            The logarithm of #FORM.number# to base 10: #log10(FORM.number)#
</cfif>
</cfoutput>
</cfif>
<cfform action = "exp.cfm">
Enter a number to see its value raised to the E power, its natural logarithm,
    and the logarithm of number to base 10.
<cfinput type = "Text" name = "number" message = "You must enter a number"
    validate = "float" required = "No">
<input type = "Submit" name = "Submit">
</cfform>
```

ExpandPath

説明
relative_path の値と同等の、プラットフォームに応じた絶対パスを作成して、ベースパスに追加します。この関数では (名前に反して)、**relative_path** パラメータに絶対パスまたは相対パスを使用できます。

ベースパスとは、現在実行しているページのディレクトリパスです。ベースパスは pageContext.getServletContext() に格納されています。

戻り値
文字列です。相対パスの文字列の末尾にスラッシュ (または円記号) が含まれている場合は、戻り値の末尾にも同じ文字が含められます。

カテゴリ
システム関数

関数のシンタックス
```
ExpandPath(relative_path)
```

関連項目
FileExists、GetCurrentTemplatePath、GetFileFromPath

履歴

ColdFusion MX: **relative_path** パラメータの動作が変更されました。この関数の **relative_path** パラメータには、絶対パスと相対パスのどちらも使用できるようになりました。この関数ではパスの変換に、ColdFusion Administrator で定義された仮想マッピングが使用されます。この関数では、IIS、Apache、または他の Web サーバーで定義されている仮想マッピングを確実に実行できるわけではありません。

パラメータ

パラメータ	説明
relative_path	絶対パスに変換する、現在のディレクトリ (.¥ および ..¥) 内の相対または絶対ディレクトリのリファレンス名またはファイル名です。スラッシュまたは円記号を含めることができます。 また、カスタムタグディレクトリ内のファイルも解決されます。例えば、カスタムタグディレクトリ（C:¥ColdFusion10）に test.txt というファイルが存在する場合、（¥test.txt を指定した）この関数では C:¥ColdFusion10¥test.txt が返されます。

使用方法

このパラメータまたは返されたパスが無効である場合、エラーが発生します。この関数はメモリ内のファイルには使用できません。

次の例では、相対パスの有効な構成を示します。

- ExpandPath("*.*")
- ExpandPath("/")
- ExpandPath("¥")
- ExpandPath("/mycfpage.cfm")
- ExpandPath("mycfpage.cfm")
- ExpandPath("myDir/mycfpage.cfm")
- ExpandPath("/myDir/mycfpage.cfm")
- ExpandPath("../../mycfpage.cfm")

ColdFusion 10 での機能強化により、カスタムタグディレクトリ内のファイルも解決されるようになりました。例えば、カスタムタグディレクトリ（C:¥ColdFusion10）に test.txt というファイルが存在する場合、（¥test.txt を指定した）この関数では C:¥ColdFusion10¥test.txt が返されます。

例

```
<h3>ExpandPath Example</h3>

<cfset thisPath=ExpandPath("*.*")>
<cfset thisDirectory=GetDirectoryFromPath(thisPath)>
<cfoutput>
The current directory is: #GetDirectoryFromPath(thisPath)#

<cfif IsDefined("form.yourFile") AND form.yourFile is not "">
    <cfset yourFile = form.yourFile>
       <cfif FileExists(ExpandPath(yourfile))>
        <p>Your file exists in this directory. You entered
        the correct filename, #GetFileFromPath("#thisPath#/#yourfile#")#
       <cfelse>
        <p>Your file was not found in this directory:
        <br>Here is a list of the other files in this directory:
        <!--- use cfdirectory to give the contents of the
        snippets directory, order by name and size --->
        <cfdirectory directory="#thisDirectory#"
        name="myDirectory"
        sort="name ASC, size DESC">
        <!--- Output the contents of the cfdirectory as a cftable --->
        <cftable query="myDirectory">
        <cfcol header="NAME:"
               text="#Name#">
        <cfcol header="SIZE:"
               text="#Size#">
        </cftable>
       </cfif>
</cfif>
</cfoutput>

<form action="expandpath.cfm" method="post">
<h3>Enter the name of a file in this directory <i>
       <font size="-1">(try expandpath.cfm)</font></i></h3>
<input type="text" name="yourFile">
<input type="submit" name="">
</form>
```

FileClose

説明

開いているディスク上またはメモリ内のファイルを閉じます。FileOpen 関数を使用すると、ColdFusion からファイルオブジェクトのハンドルが返されます。ファイルを閉じてもそのハンドルは存在し続けますが、ファイルは閉じられたものとしてリストされます。

カテゴリ

システム関数

関数のシンタックス

```
FileClose(fileObj)
```

関連項目

FileCopy、FileIsEOF、FileOpen、FileRead、FileReadLine、FileWrite

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
fileobj	閉じるファイルです。

使用方法
開いたファイルは必ず閉じる必要があります。FileOpen 関数を使用してファイルを開くと、ファイルストリームが開かれます。内容の読み書きはそのファイルストリームに対して行われます。FileClose 関数はこのストリームを閉じます。ファイルを閉じないとストリームは開かれたままになります。この場合、オペレーティングシステムによってファイルがロックされることがあり、サーバーを再起動するまでそのファイルを使用できなくなることがあります。

例
次の例では、ファイルがまだ開いているかどうかを確認した後に、そのファイルを閉じます。

```
<cfscript>
myfile = FileOpen("c:\ColdFusion9\wwwroot\test1.txt", "read");
while(NOT FileIsEOF(myfile))
{
x = FileReadLine(myfile);
WriteOutput("#x# <br>");
}
</cfscript>
<!--- Additional code goes here. --->
<cfif #myfile.status# IS "open">
    <cfoutput>The file #myfile.filepath# is #myfile.status#</cfoutput><br>
    <cfscript>
    FileClose(myfile);
    </cfscript>
</cfif>
```

FileCopy

説明
指定されたディスク上またはメモリ内のコピー元ファイルを、指定されたコピー先ファイルにコピーします。

カテゴリ
システム関数

関数のシンタックス
```
FileCopy(source, destination)
```

関連項目
FileClose、FileIsEOF、FileOpen、FileRead、FileReadLine、FileWrite、cffile

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
source	コピーするディスク上またはメモリ内のファイルのパス名です。絶対パス (ドライブ文字とコロン、あるいはスラッシュまたは円記号から始まるパス) を指定しない場合は、GetTempDirectory 関数から返される ColdFusion テンポラリディレクトリを基準とする相対パスを指定します。
destination	ファイルのコピー先となる、Web サーバー上の、ディスク上またはメモリ内のディレクトリまたはファイルのパス名です。ディレクトリパスを明示しないでファイル名を指定した場合、ColdFusion は source で指定されたディレクトリの相対パスにファイルをコピーします。

使用方法

メモリ内のファイルまたはディレクトリを指定するには、次のシンタックスを使用します。この場合、ディスクには書き込まれません。メモリ内のファイルを使用すると、一時的データの処理が速くなります。

```
ram:///filepath
```

ファイルパスには、ram:///petStore/images/poodle.jpg のようにディレクトリを含めることができます。ファイルを指定する前に、パスに含まれるディレクトリを作成しておく必要があります。メモリ内のファイルの使用方法について詳しくは、『ColdFusion アプリケーションの開発』の Working with in-memory files を参照してください。

例

次の例では、Windows の c:¥testingdir¥ ディレクトリから c:¥productiondir¥ ディレクトリに test1.txt ファイルをコピーし、新しいコピーに test2.txt という名前を付けます。

```
<h3>FileCopy Example</h3>
<cfset sourcefile="c:\testingdir\test1.txt">
<cfset destinationfile="c:\productiondir\test2.txt">

<cfif FileExists(#sourcefile#)>
    <cfif FileExists(#destinationfile#)>
        <cfoutput>A copy of #destinationfile# already exists.</cfoutput>
    <cfelse>
        <cfscript>
        FileCopy(#sourcefile#, #destinationfile#);
        </cfscript>
        <cfoutput>Copied: #sourcefile# <br>
            To: #destinationfile#</cfoutput><br>
    </cfif>
<cfelse>
    <cfoutput>The source file does not exist.</cfoutput><br>
</cfif>
```

FileDelete

説明

サーバー上の、指定されたディスク上またはメモリ内のファイルを削除します。

カテゴリ

システム関数

関数のシンタックス

```
FileDelete(filepath)
```

関連項目

FileClose、FileIsEOF、FileOpen、FileRead、FileReadLine、FileWrite、cffile

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
filepath	削除するディスク上またはメモリ内のファイルのパス名です。絶対パス (ドライブ文字とコロン、あるいはスラッシュまたは円記号から始まるパス) を指定しない場合は、GetTempDirectory 関数から返される ColdFusion テンポラリディレクトリを基準とする相対パスを指定します。

使用方法

メモリ内のファイルによって使用されているメモリを解放するには、この関数を使用します。メモリ内のファイルの使用方法について詳しくは、『ColdFusion アプリケーションの開発』の Working with in-memory files を参照してください。

例

次の例では、ファイル c:¥productiondir¥test1.txt を削除した後、ファイル c:¥testdir¥test1.txt を移動します。

```
<h3>FileDelete Example</h3>

<cfset sourcefile="c:\testdir\test1.txt">
<cfset destinationfile="c:\productiondir\test1.txt">

<cfif FileExists(#sourcefile#)>
    <cfif FileExists(#destinationfile#)>
        <cfoutput>The destination file already exists.<br>
            Deleting previous copy of #destinationfile#.<br>
            Moving: #sourcefile# <br>
            To: <br> #destinationfile#.</cfoutput><br>
        <cfscript>
            FileDelete(#destinationfile#);
            FileMove(#sourcefile#, #destinationfile#);
        </cfscript>
    <cfelse>
        <cfscript>
            FileMove(#sourcefile#, #destinationfile#);
        </cfscript>
        <cfoutput>Moved: #sourcefile# <br>
            To: <br> #destinationfile#.</cfoutput><br>
    </cfif>
<cfelse>
    <cfoutput>The source file does not exist.</cfoutput><br>
</cfif>
```

FileExists

説明

ディスク上またはメモリ内のファイルが存在するかどうかを判別します。

戻り値

パラメータに指定されたファイルが存在する場合は YES、存在しない場合は NO

カテゴリ
システム関数、決定関数

関数のシンタックス
```
FileExists(absolute_path)
```

関連項目
DirectoryExists、ExpandPath、GetTemplatePath

パラメータ

パラメータ	説明
absolute_path	ディスク上またはメモリ内のファイルの絶対パスです。

使用方法
リモートシステム上のファイルにアクセスするには、ColdFusion を実行しているアカウント (Windows の場合) またはユーザー (UNIX および Linux の場合) に、そのファイル、ディレクトリ、およびリモートシステムへのアクセスが許可されている必要があります。たとえば、Windows サービスとしてサーバー設定で ColdFusion を実行する場合、デフォルトでは ColdFusion はローカルシステムアカウントで実行され、リモートシステムにアクセスできる権限がありません。ただし、この設定は [Services Properties] ダイアログボックスの [Log On] ページで変更できます。

例
```
<h3>FileExists Example</h3>

<cfset thisPath = ExpandPath("*.*")>
<cfset thisDirectory = GetDirectoryFromPath(thisPath)>
<cfoutput>
The current directory is: #GetDirectoryFromPath(thisPath)#
<cfif IsDefined("FORM.yourFile")>
<cfif FORM.yourFile is not "">
<cfset yourFile = FORM.yourFile>
    <cfif FileExists(ExpandPath(yourfile))>
    <p>Your file exists in this directory. You entered
    the correct filename, #GetFileFromPath("#thisPath#/#yourfile#")#</p>
    <cfelse>
```

FileGetMimeType

説明
指定したファイルパスまたはファイルオブジェクトの MIME タイプを取得します。

戻り値
MIME タイプを返します。

シンタックス
fileGetMimeType(path[, strict]) fileGetMimeType(fileObject[,strict])

パラメータ

パラメータ	説明
path	ディスク上のファイルのフルパスです（strict を true に設定する場合）。 フルパスを指定しない場合、ファイルは、getTempDirectory 関数で返されるテンポラリディレクトリに存在するものと見なされます。
fileObject	ファイルオブジェクトの名前です。
strict	false の場合は、ファイルタイプを拡張子で判別します。デフォルト値は true です。

例

例えば、/folder1 に test.pdf という名前のファイルが存在し、同じフォルダーに test.txt が存在していて、MIME タイプをチェックするとします。この場合の test.txt は、test.pdf をコピーして拡張子の名前を txt に変更したものです。

```
<cfscript>
    //Case 1.
    mime.mimeType1 = FilegetMimeType(expandPath('/folder1/test.pdf'));
    //Case 2.
    mime.mimeType2 = FilegetMimeType(expandPath('/folder1/test.pdf'),false);
    //Case 3.
    mime.mimeType3 = FilegetMimeType(expandPath('/folder1/test.txt'));
    //Case 4.
    mime.mimeType4 = FilegetMimeType(expandPath('/folder1/test.txt'),false);
    writeDump(mime);
</cfscript>
```

- **Case 1 および Case 2**：このファイルは初めから PDF ファイルであるので、strict = true であるか false であるかに関係なく、application/pdf が返されます。
- **Case 3：**デフォルトは strict = true であり、このファイルはもともとは PDF で名前が TXT に変更されたものであるので、application/pdf が返されます。
- **Case 4：**strict = false となっているので、text/plain が返されます。

FileIsEOF

説明

ディスク上またはメモリ内のファイルを読み込むときに、既にファイルの末尾に達しているかどうかを調べます。

戻り値

既にファイルの末尾に達している場合は Yes、達していない場合は No

カテゴリ

システム関数、決定関数

関数のシンタックス

```
FileIsEOF(fileObj)
```

関連項目

FileClose、FileOpen、FileRead、FileReadLine

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
fileObj	ファイルオブジェクトです。

例

次の例では、ファイルの末尾に達するまでファイルを読み込みます。

```
<h3>FileIsEOF Example</h3>

<cfscript>
myfile = FileOpen("c:\ColdFusion9\wwwroot\test1.txt", "read");
while(NOT FileIsEOF(myfile))
{
x = FileReadLine(myfile);
WriteOutput("#x# <br>");
}
FileClose(myfile);
</cfscript>
```

FileMove

説明

ディスク上またはメモリ内のファイルをサーバー上のある場所から別の場所に移動します。

カテゴリ

システム関数

履歴

ColdFusion 8: この関数が追加されました。

関数のシンタックス

```
FileMove(source, destination)
```

関連項目

FileClose、FileCopy、FileOpen、FileRead、FileReadLine、FileWrite、cffile

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
source	移動するディスク上またはメモリ内のファイルのパス名です。絶対パス (ドライブ文字とコロン、あるいはスラッシュまたは円記号から始まるパス) を指定しない場合は、GetTempDirectory 関数から返される ColdFusion テンポラリディレクトリを基準とする相対パスを指定します。
destination	移動先となる、ディスク上またはメモリ内のディレクトリやファイルのパス名です。絶対パスを指定しない場合は、ソースディレクトリを基準とした相対パスを指定します。

使用方法
メモリ内のファイルを指定するには、次のシンタックスを使用します。この場合、ディスクには書き込まれません。メモリ内のファイルを使用すると、一時的データの処理が速くなります。

```
ram:///filepath
```

ファイルパスには、ram:///petStore/images/poodle.jpg のようにディレクトリを含めることができます。ファイルを指定する前に、パスに含まれるディレクトリを作成しておく必要があります。メモリ内のファイルの使用方法について詳しくは、『ColdFusion アプリケーションの開発』の Working with in-memory files を参照してください。

例
次の例では、Windows の c:¥testingdir¥ ディレクトリから c:¥productiondir¥ ディレクトリに test1.txt ファイルを移動し、ファイルの名前を test2.txt に変更します。

```
<h3>FileMove Example</h3>
<cfset sourcefile="c:\testingdir\test1.txt">
<cfset destinationfile="c:\productiondir\test2.txt">

<cfif FileExists(#sourcefile#)>
    <cfif FileExists(#destinationfile#)>
        <cfoutput>The destination file already exists.</cfoutput>
    <cfelse>
        <cfscript>
        FileMove(#sourcefile#, #destinationfile#);
        </cfscript>
        <cfoutput>Moved: #sourcefile# <br>
            To: <br> #destinationfile#.</cfoutput><br>
    </cfif>
<cfelse>
    <cfoutput>The source file does not exist.</cfoutput><br>
</cfif>
```

FileOpen

説明
読み込み、書き込み、または追記のためにディスク上またはメモリ内のファイルを開きます。大きなファイルを読み込むときは、この関数を FileRead 関数と組み合わせて使用します。

戻り値
開いたファイルを表すファイルオブジェクトです。

カテゴリ
システム関数

関数のシンタックス
```
FileOpen(filepath, [mode, charset])
```

関連項目
FileClose、FileCopy、FileReadBinary、FileRead、FileReadLine、FileWrite、cffile

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
filepath	サーバー上の、ディスク上またはメモリ内のファイルの絶対パスです。
mode	ファイルに対して実行するアクションです。次のモードがあります。 • read • readBinary • write • append モードを指定しない場合は、read モードでファイルが開かれます。
charset	ファイルの文字セットです。

使用方法

開く前にファイルが存在していなくてもかまいません。新しいファイルを書き込むには、書き込み用にファイルを開き、それからそのファイルを書き込みます。

ファイルオブジェクトはファイルのハンドルです。このオブジェクトは、次の情報にアクセスするための構造体として使用できます。

- filename: 開いたファイルの名前
- filepath: 絶対パスとファイル名
- lastmodified: ファイルの最終変更日時
- mode: ファイルを開いた目的
- size: ファイルサイズ (バイト単位)
- status: ファイルオブジェクトの状態 (開いているか、閉じているか)

次のコードは、ファイルを開き、そのファイルの絶対パスとファイル名を表示します。

```
<cfscript>
myfile = FileOpen("c:\temp\test1.txt", "read");
</cfscript>
myfile refers to:
<cfdump var="#myfile.filepath#">
```

メモリ内のファイルを指定するには、次のシンタックスを使用します。この場合、ディスクには書き込まれません。メモリ内のファイルを使用すると、一時的データの処理が速くなります。

```
ram:///filepath
```

ファイルパスには、ram:///petStore/images/poodle.jpg のようにディレクトリを含めることができます。ファイルを指定する前に、パスに含まれるディレクトリを作成しておく必要があります。メモリ内のファイルの使用方法について詳しくは、『ColdFusion アプリケーションの開発』の Working with in-memory files を参照してください。

開いたファイルは必ず閉じる必要があります。FileOpen 関数を使用してファイルを開くと、ディスクからのファイルストリームが開かれます。内容の読み書きはそのファイルストリームに対して行われます。FileClose 関数はこのストリームを閉じます。ファイルを閉じないとストリームは開かれたままになります。この場合、オペレーティングシステムによってファイルがロックされることがあり、サーバーを再起動するまでそのファイルを使用できなくなることがあります。

例

次の例では、ファイルを開き、ファイルの各行を読み込んで出力した後、ファイルを閉じます。

```
<h3>FileOpen Example</h3>

<cfscript>
myfile = FileOpen("c:\temp\test1.txt", "read");
while(NOT FileIsEOF(myfile))
{
x = FileReadLine(myfile);
WriteOutput("#x# <br>"); }
FileClose(myfile);
</cfscript>
```

FileRead

説明

ディスク上またはメモリ内のテキストファイル、または FileOpen 関数で作成されたファイルオブジェクトを読み込みます。この関数は、cffile タグで action="read" 属性を使用する方法の代わりとして使用できます。また、FileRead はメモリにファイル全体を読み込まないため、この関数を使用することで大きなサイズのファイルを読み込むときのパフォーマンスを向上させることができます。

戻り値

ファイルパスを指定した場合は、そのファイルの全テキスト

ファイルオブジェクトを指定した場合は、指定したサイズの文字バッファーまたはバイトバッファー

ファイルが read モードで開かれている場合、FileRead は文字データ (文字列) を返します。それ以外のモードの場合はバイナリデータを返します。

カテゴリ

システム関数

関数のシンタックス

```
FileRead(filepath [, charset])

OR

FileRead(fileobj [, buffersize])
```

関連項目

FileClose、FileIsEOF、FileReadBinary、FileReadLine、FileWrite

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
filepath	サーバー上の、ディスク上またはメモリ内のテキストファイルの絶対パスです。
charset	ファイルコンテンツをエンコードする文字エンコードを指定します。一般的に使用される値を次に示します。 • utf-8 • iso-8859-1 • windows-1252 • us-ascii • shift_jis • iso-2022-jp • euc-jp • euc-kr • big5 • euc-cn • utf-16 ファイルが BOM (Byte Order Mark) で始まる場合に、この属性を別の文字エンコードに設定すると、エラーになります。
fileobj	読み込むファイルオブジェクトです。
buffersize	読み込む文字数です。

使用方法

FileRead 関数では、テキストファイルまたはファイルオブジェクトを読み込めます。テキストファイルの絶対パスを指定した場合は、そのファイルの全内容が読み込まれます。FileOpen 関数で作成されたファイルオブジェクトを指定した場合は、buffersize で指定した文字数が読み込まれます。

例

```
<h3>FileRead Example - Reading a file</h3>

<!--- This reads and outputs the entire file contents. --->
<cfscript>
myfile = FileRead("c:\temp\myfile.txt");
    WriteOutput("#myfile#");
</cfscript>

<!--- This reads and outputs the first 100 characters --->
<!--- from the same file. --->
<cfscript>
myfile = FileOpen("c:\temp\test1.txt", "read");
x = FileRead(myfile, 100);
WriteOutput("#x#");
FileClose(myfile);
</cfscript>
```

FileReadBinary

説明
ページ内で使用できるバイナリオブジェクトのパラメータに、サーバー上にあるディスク上またはメモリ内のバイナリファイル (実行可能ファイルやイメージファイルなど) を読み込みます。このファイルを HTTP や SMTP などの Web プロトコルを介して送信するか、またはデータベースに格納するには、ToBase64 関数を使用してファイルをまず Base64 に変換してください。

注意：このアクションによって、ファイルがローカル Variables スコープ内の変数に読み込まれます。サーバーがダウンするおそれがあるため、ログなどの大きなファイルには使用しないでください。

戻り値
バイナリファイルの全内容

カテゴリ
システム関数

関数のシンタックス
```
FileReadBinary(filepath)
```

関連項目
FileClose、FileIsEOF、FileRead、FileReadLine、FileWrite

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
filepath	サーバー上の、ディスク上またはメモリ内のバイナリファイルの絶対パスです。

使用方法
バイナリファイルを Base64 に変換して、それを別のサイトに転送します。ColdFusion 8 では、イメージファイルをバイナリとして読み込み、その結果を cfimage に渡すことができます。

例
次の例では、バイナリファイルを読み込みます。

```
<h3>FileReadBinary Example</h3>
<cfscript>
myfile = FileReadBinary("c:\testingdir\test3.jpg");
</cfscript>
```

FileReadLine

説明
ディスク上またはメモリ内のファイルから行を読み取ります。

戻り値
ファイルの行

カテゴリ
システム関数

関数のシンタックス
```
FileReadLine(fileObj)
```

関連項目
FileClose、FileIsEOF、FileRead、FileWrite

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
fileObj	ファイルオブジェクトです。

例
次の例では、ファイルを開き、各行を読み込んで出力した後、ファイルを閉じます。

```
<h3>FileReadLIne Example</h3>

<cfscript>
myfile = FileOpen("c:\ColdFusion9\wwwroot\test1.txt", "read");
while(NOT FileisEOF(myfile))
{
x = FileReadLine(myfile); // read line
WriteOutput("#x#");
}
FileClose(myfile);
</cfscript
```

FileSeek

説明
サーバーのディスク上またはメモリ内のファイルの読み取り操作または書き込み操作の位置をシークします。

戻り値
次のファイル操作が発生するストリーム内のファイル位置を返します。

カテゴリ
システム関数

関数のシンタックス
```
FileSeek(fileObj, pos)
```

関連項目

FileCopy、FileDelete、FileExists、FileMove、FileSetAccessMode、FileSetAttribute、FileSkipBytes

履歴

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
fileObj	ファイルオブジェクトです。
pos	次の読み取り操作および書き込み操作を実行する必要があるストリーム内のファイルの位置です。

例

```
<cfscript>
            NewFile = FileOpen(ExpandPath(".") & "\test.txt","write","",true);
            FileSeek(#NewFile#,0);
            FileWrite(#NewFile#,"Hello World.. This is for FileOpen, FileSeek, FileSkipBytes.");
            FileClose(#NewFile#);
            WriteOutput("<br>Opening in Read Mode.<br>");
            NewFile = FileOpen(ExpandPath(".") & "\test.txt","read","",true);
            ReadFile = FileRead(#NewFile#,100);
            WriteOutput("#ReadFile#<br>");
            FileClose(#NewFile#);
      WriteOutput("<br>Opening in Read-Write Mode.<br>");
      NewFile = FileOpen(ExpandPath(".") & "\test.txt","readwrite","",true);
      FileSeek(#NewFile#,2);
      FileSkipBytes(#NewFile#,4);
      FileWrite(#NewFile#,"Earth");
      ReadFile = FileRead(#NewFile#,100);
      WriteOutput("#ToString(ReadFile)#<br>");
      FileClose(#NewFile#);
</cfscript>
```

FileSetAccessMode

説明

UNIX または Linux 上での、ディスク上のファイルの属性を設定します。この関数はメモリ内のファイルには使用できません。

カテゴリ

システム関数

関数のシンタックス

```
FileSetAccessMode(filepath, mode)
```

関連項目

FileCopy、FileDelete、FileExists、FileMove

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
filepath	サーバー上のファイルの絶対パスです。
mode	3 桁の値です。それぞれの桁が、個人またはグループのファイルアクセス権限を指定します。 • 1 桁目は所有者を表します。 • 2 桁目はグループを表します。 • 3 桁目はすべてのユーザーを表します。 このコードの各桁で、個人またはグループの権限を設定します。 • 4 = 読み込み権限 • 2 = 書き込み権限 • 1 = 実行権限 権限の組み合わせを示すには、これらの数値の合計を使用します。 • 3 = 書き込み + 実行 • 5 = 読み込み + 実行 • 6 = 読み込み + 書き込み • 7 = 読み込み + 書き込み + 実行 たとえば、所有者のみにファイルの読み込みを許可する場合は 400、すべてのユーザーにファイルの読み込みを許可する場合は 004 を指定します。

例

次の例では、所有者のみが読み取れるようにファイルのアクセスモードを設定します。

```
<h3>FileSetAccessMode Example</h3>

<cfscript>
    FileSetAccessMode("test1.txt", "004");
</cfscript>
```

FileSetAttribute

説明

Windows 上での、ディスク上のファイルの属性を設定します。この関数はメモリ内のファイルには使用できません。

カテゴリ

システム関数

関数のシンタックス

```
FileSetAttribute(filepath, attribute)
```

関連項目

FileCopy、FileDelete、FileExists、FileMove

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
filepath	サーバー上のファイルの絶対パスです。
attribute	次のいずれかです。 • readOnly • hidden • normal ファイルの属性を normal に設定すると、読み取り専用の属性と非表示の属性が解除されます。

例
次の例では、ファイルのアクセスモードを読み取り専用に設定します。

```
<h3>FileSetAttribute Example</h3>

<cfscript>
    FileSetAttribute("c:\temp\test1.txt", "readOnly");
</cfscript>
```

FileSetLastModified

説明
ディスク上またはメモリ内のファイルの最終変更日を設定します。

カテゴリ
システム関数

関数のシンタックス
```
FileSetLastModified(filepath, date)
```

関連項目
FileCopy、FileDelete、FileExists、FileMove、FileSetAccessMode、FileSetAttribute

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
filepath	サーバー上の、ディスク上またはメモリ内のファイルの絶対パスです。
date	有効な ColdFusion の日付または日時

例

```
<cfscript>
    FileSetLastModified("c:\temp\test1.txt", "#Now()#");
    WriteOutput(#GetFileInfo("c:\temp\test1.txt").lastmodified#);
</cfscript>
```

FileSkipBytes

説明

サーバーのディスク上またはメモリ内のファイルの読み取り操作または書き込み操作の前に、データをスキップします。

カテゴリ

システム関数

関数のシンタックス

FileSkipBytes(fileObj, noOfBytesToSkip)

関連項目

FileCopy、FileDelete、FileExists、FileMove、FileSetAccessMode、FileSetAttribute、FileSeek

履歴

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
fileObj	ファイルオブジェクトです。
noOfBytesToSkip	次のファイル操作が発生する前にスキップする必要があるバイト数です。

使用方法

noOfBytesToSkip では、実際のバイト数より大きな値を指定すると、すべてのバイトがスキップされます。

例

関数 FileSeek の「例」を参照してください。

FileUpload

説明

サーバー上のディレクトリにファイルをアップロードします。

戻り値

ファイルのアップロードの結果 (ステータス) が含まれた構造体

構造体に含まれる情報について詳しくは、cffile action = "upload" の「使用方法」の節を参照してください。

関数のシンタックス

FileUpload(destination)FileUpload(destination, filefield)FileUpload(destination, filefield, accept)FileUpload(destination, filefield, accept, nameconflict)

履歴

ColdFusion 9.0.1: この関数が追加されました。

パラメータ

値	説明
destination	ファイルのアップロード先となるディレクトリのパスです。 絶対パス (ドライブ文字とコロン、あるいはスラッシュまたは円記号から始まるパス) を指定しない場合は、GetTempDirectory 関数から返される ColdFusion テンポラリディレクトリを基準とする相対パスを指定します。指定した宛先が存在しない場合は、指定された宛先の名前のファイルが作成されます。 たとえば、宛先 C:¥XYZ を指定した場合は、C: ドライブにファイル XYZ が作成されます。
accept	受け入れる MIME タイプを制限します。カンマ区切りリストです。 たとえば、次のコードでは、JPEG ファイルと Microsoft Word ファイルのアップロードが許可されます。 "image/jpg,application/msword" ブラウザは、ファイル名の拡張子に基づいてファイルタイプを判別します。
nameConflict	ファイル名がディレクトリ内のファイルと同じ場合に実行するアクションです。 • Error: ファイルは保存されません。ColdFusion によりページの処理が停止され、エラーが返されます。 • Skip: ファイルは保存されません。このオプションでは、ファイルプロパティに基づいて、動作をカスタマイズできます。 • Overwrite: ファイルを上書きします。 • MakeUnique: アップロード用に固有のファイル名を作成します。この名前は、file オブジェクト変数 serverFile に保管されます。
fileField	ファイルを選択するときに使用したフォームフィールドの名前です。 フィールド名を指定するときにシャープ記号 (#) を使用しないでください。

関連項目

FileUploadAll

使用方法

cffile action = "upload"

例

```
<cfif isdefined("form.fileData")>
      <cfscript>
            hello = FileUpload("<path>","<mime type>","unique");
      </cfscript>
      <cfdump var="#hello#">
<cfelse>
      <cfform name="myUpload" enctype="multipart/form-data">
            <cfinput type="file" name="fileData"><br>
            <cfinput type="submit" name="submit">
      </cfform>
</cfif>
```

FileUploadAll

説明
HTTP リクエストでページに送信されたすべてのファイルを、サーバー上のディレクトリにアップロードします。

戻り値
ファイルのアップロードステータスを含む構造体の配列

構造体に含まれる情報について詳しくは、cffile action = "uploadAll" の「使用方法」の節を参照してください。

関数のシンタックス
FileUploadAll(destination)

FileUploadAll(destination, accept)

FileUploadAll(destination, accept, nameConflict)

履歴
ColdFusion 9.0.1: この関数が追加されました。

パラメータ

値	説明
destination	ファイルのアップロード先となるディレクトリのパスです。 絶対パス (ドライブ文字とコロン、あるいはスラッシュまたは円記号から始まるパス) を指定しない場合は、GetTempDirectory 関数から返される ColdFusion テンポラリディレクトリを基準とする相対パスを指定します。 指定した宛先が存在しない場合は、指定された宛先の名前のファイルが作成されます。たとえば、宛先 C:¥XYZ を指定した場合は、C: ドライブにファイル XYZ が作成されます。
accept	受け入れる MIME タイプを制限します。カンマ区切りリストです。 たとえば、次のコードでは、JPEG ファイルと Microsoft Word ファイルのアップロードが許可されます。 "image/jpg, application/msword" ブラウザは、ファイル名の拡張子に基づいてファイルタイプを判別します。
nameConflict	ファイル名がディレクトリ内のファイルと同じ場合に実行するアクションです。 • Error: ファイルは保存されません。ColdFusion によりページの処理が停止され、エラーが返されます。 • Skip: ファイルは保存されません。ファイルプロパティに基づいて、動作をカスタマイズできます。 • Overwrite: ファイルを上書きします。 • MakeUnique: アップロード用に固有のファイル名を作成します。この名前は、file オブジェクト変数 serverFile に保管されます。

関連項目
FileUpload

使用方法
cffile action = "uploadAll"

FileWrite

説明
ファイルパスを指定した場合は、すべての内容を指定したディスク上またはメモリ内のファイルに書き込みます。ファイルオブジェクトを指定した場合は、テキストまたはバイナリデータをファイルオブジェクトに書き込みます。

カテゴリ
システム関数

関数のシンタックス
```
FileWrite(filepath, data [, charset])

OR

FileWrite(fileobj, data)
```

関連項目
FileCopy、FileDelete、FileExists、FileMove、cffile

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
charset	ファイルコンテンツをエンコードする文字エンコードを指定します。一般的に使用される値を次に示します。 • utf-8 • iso-8859-1 • windows-1252 • us-ascii • shift_jis • iso-2022-jp • euc-jp • euc-kr • big5 • euc-cn • utf-16 ファイルが BOM (Byte Order Mark) で始まる場合に、この属性を別の文字エンコードに設定すると、エラーになります。

パラメータ	説明
data	作成するファイルまたはファイルオブジェクトの内容です。
fileobj	書き込み先となるファイルオブジェクトの名前です。
filepath	書き込み先となる、ディスク上またはメモリ内のファイルのパス名です。 絶対パス (ドライブ文字とコロン、あるいはスラッシュまたは円記号から始まるパス) を指定しない場合は、GetTempDirectory 関数から返される ColdFusion テンポラリディレクトリを基準とする相対パスを指定します。

使用方法

メモリ内のファイルを指定するには、次のシンタックスを使用します。この場合、ディスクには書き込まれません。メモリ内のファイルを使用すると、一時的データの処理が速くなります。

```
ram:///filepath
```

ファイルパスには、ram:///petStore/images/poodle.jpg のようにディレクトリを含めることができます。ファイルを指定する前に、パスに含まれるディレクトリを作成しておく必要があります。メモリ内のファイルの使用方法について詳しくは、『ColdFusion アプリケーションの開発』の Working with in-memory files を参照してください。

例

```
<h3>FileWrite Example</h3>
<!--- This example gets the email addresses of employees, --->
<!--- creates a file object that contains the e-mail addresses, --->
<!--- read the file object, and then creates a text file with a --->
<!--- list of e-mail addresses. --->

<cfquery name="getemployees" datasource="cfdocexamples">
SELECT EMAIL
FROM Employees
</cfquery>

<cfset companymail = "">

<cfloop query = "getemployees">
    <cfset companymail = companymail & #EMAIL# & ";" & " ">
</cfloop>

<cfscript>
FileWrite("mail_list", "#companymail#");
mlist = FileRead("mail_list");
FileWrite("c:\temp\mail_list.txt", "#mlist#");
</cfscript>
```

FileWriteLine

説明

指定したテキストをファイルオブジェクトの末尾に追加します。

カテゴリ

システム関数

関数のシンタックス

```
FileWriteLine(fileobj, text)
```

関連項目
FileCopy、FileDelete、FileExists、FileMove、FileWrite、cffile

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
text	ファイルオブジェクトに追加する内容です。
fileobj	行の書き込み先となるファイルオブジェクトです。

例
```
<h3>FileWriteLine Example</h3>

<cfscript>
    myfile = FileOpen("c:\temp\test1.txt", "write");
    FileWriteLine(myfile,"This line is new.");
    FileClose(myfile);
</cfscript>
```

Find

説明
指定された開始位置から、文字列 **string** 内で最初に出現する検索文字列 **substring** を探します。この検索では大文字と小文字が区別されます。

戻り値
数値。**string** 内の **substring** の位置が返されます。**string** 内に **substring** がなかった場合は 0 が返されます。

カテゴリ
文字列関数

関数のシンタックス
```
Find(substring, string [, start ])
```

関連項目
FindNoCase、Compare、FindOneOf、REFind、Replace

パラメータ

パラメータ	説明
substring	文字列、または文字列を含んでいる変数です。検索する文字列です。
string	文字列、または文字列を含んでいる変数です。検索対象の文字列を指定します。
start	検索の開始位置です。

例

```
<cfoutput>
    <cfset stringToSearch = "The quick brown fox jumped over the lazy dog.">
    #find("the",stringToSearch)#<br>
    #find("the",stringToSearch,35)#<br>
    #find("no such substring",stringToSearch)#<br>
    <br>
    #findnocase("the",stringToSearch)#<br>
    #findnocase("the",stringToSearch,5)#<br>
    #findnocase("no such substring",stringToSearch)#<br>
    <br>
    #findoneof("aeiou",stringToSearch)#<br>
    #findoneof("aeiou",stringToSearch,4)#<br>
    #findoneof("@%^*()",stringToSearch)#<br>
</cfoutput>
```

FindNoCase

説明

指定された開始位置から、文字列 **string** 内で最初に出現する検索文字列 **substring** を探します。検索文字列 (**substring**) が文字列 (**string**) 内にない場合は 0 を返します。この検索では大文字と小文字は区別されません。

戻り値

string 内の **substring** の位置、または **string** 内に **substring** がなかった場合は 0。

カテゴリ

文字列関数

関数のシンタックス

```
FindNoCase(substring, string [, start ])
```

関連項目

Find、CompareNoCase、FindOneOf、REFind、Replace

パラメータ

パラメータ	説明
substring	文字列、または文字列を含んでいる変数です。検索する文字列です。
string	文字列、または文字列を含んでいる変数です。検索対象の文字列を指定します。
start	検索の開始位置です。

例

次の例では、検索対象の "the" が小文字であるため、Find 関数は検索文字が最初に出現した位置として 33 を返します。FindNoCase 関数は、大文字と小文字を区別しないので、検索文字が最初に出現した位置として 1 を返します。

```
<cfset stringToSearch = "The quick brown fox jumped over the lazy dog.">

stringToSearch = <cfoutput>#stringToSearch#</cfoutput><br>
<p>
Find Function:<br>
Find("the",stringToSearch) returns <cfoutput>#find("the",stringToSearch)#</cfoutput><br>
<p>
FindNoCase Function:<br>
FindNoCase("the",stringToSearch) returns <cfoutput>#FindNoCase("the",stringToSearch)#</cfoutput>
```

FindOneOf

説明

指定された開始位置以降で、文字列 **string** 内の文字セットのいずれかの文字で最初に出現するものを探します。この検索では大文字と小文字が区別されます。

戻り値

文字列 **string** 内で最初に見つかった **set** のメンバーの位置、または文字列 **string** 内で **set** のメンバーが見つからないときは 0。

カテゴリ

文字列関数

関数のシンタックス

```
FindOneOf(set, string [, start ])
```

関連項目

Find、Compare、REFind

パラメータ

パラメータ	説明
set	文字列、または文字列を含んでいる変数です。検索する文字を含んでいる文字列です。
string	文字列、または文字列を含んでいる変数です。検索対象の文字列を指定します。
start	検索の開始位置です。

例

```
<cfoutput>
<cfset stringToSearch = "The quick brown fox jumped over the lazy dog.">
#find("the",stringToSearch)#<br>
#find("the",stringToSearch,35)#<br>
#find("no such substring",stringToSearch)#<br>
<br>
#findnocase("the",stringToSearch)#<br>
#findnocase("the",stringToSearch,5)#<br>
#findnocase("no such substring",stringToSearch)#<br>
<br>
#findoneof("aeiou",stringToSearch)#<br>
#findoneof("aeiou",stringToSearch,4)#<br>
#findoneof("@$^*()",stringToSearch)#<br>
</cfoutput>
```

FirstDayOfMonth

説明
指定された月の最初の日が 1 年の何日目かを調べます。

戻り値
1 年の何日目かを示す数値

カテゴリ
日付および時刻関数

関数のシンタックス
```
FirstDayOfMonth(date)
```

関連項目
Day、DayOfWeek、DayOfWeekAsString、DayOfYear、DaysInMonth、DaysInYear

パラメータ

パラメータ	説明
date	日付時刻オブジェクトです。値の範囲は西暦 100 ～ 9999 年です。

使用方法
日付時刻値を文字列として渡すときは、その値を引用符で囲みます。引用符で囲まない場合、その値は日付時刻オブジェクトの数値表現として解釈されます。

例
```
<h3>FirstDayOfMonth Example</h3>

<cfoutput>
The first day of #MonthAsString(Month(Now()))#, #Year(Now())# was
day #FirstDayOfMonth(Now())# of the year.
</cfoutput>
```

Fix

説明
実数を整数に変換します。

戻り値
number が 0 以上の場合、**number** 以下で最も近い整数。

number が 0 より小さい場合、**number** 以上で最も近い整数。

カテゴリ
算術関数

関数のシンタックス
```
Fix(number)
```

関連項目
Ceiling、Int、Round

パラメータ

パラメータ	説明
number	数値です。

例
```
<h3>Fix Example</h3>
<p>Fix returns the closest integer less than the number if the number is
    greater than or equal to 0. Fix returns the closest integer greater than
    the number if number is less than 0.</p>
<cfoutput>
<p>The fix of 3.4 is #Fix(3.4)#</p>
<p>The fix of 3 is #Fix(3)#</p>
<p>The fix of 3.8 is #Fix(3.8)#</p>
<p>The fix of -4.2 is #Fix(-4.2)#</p>
</cfoutput>
```

FormatBaseN

説明
radix で指定された基数を使用して **number** を文字列に変換します。

戻り値
基数 **radix** で **number** を表す文字列

カテゴリ
表示および書式制御関数、算術関数、文字列関数

関数のシンタックス
```
FormatBaseN(number, radix)
```

関連項目
InputBaseN

パラメータ

パラメータ	説明
number	変換する数値です。
radix	結果の基数です。

例

```
<h3>FormatBaseN Example</h3>
<p>Converts a number to a string in the base specified by Radix.
<p>
<cfoutput>
<br>FormatBaseN(10,2): #FormatBaseN(10,2)#
<br>FormatBaseN(1024,16): #FormatBaseN(1024,16)#
<br>FormatBaseN(125,10): #FormatBaseN(125,10)#
<br>FormatBaseN(10.75,2): #FormatBaseN(10.75,2)#
</cfoutput>
<h3>InputBaseN Example</h3>
<p>InputBaseN returns the number obtained by converting a string,
    using base specified by Radix (an integer from 2 to 36).</p>

<cfoutput>
<br>InputBaseN("1010",2): #InputBaseN("1010",2)#
<br>InputBaseN("3ff",16): #InputBaseN("3ff",16)#
<br>InputBaseN("125",10): #InputBaseN("125",10)#
<br>InputBaseN(1010,2): #InputBaseN(1010,2)#
</cfoutput>
```

GenerateSecretKey

説明

Encrypt 関数で使用するための暗号キーの値を取得します。

戻り値

暗号化キーを含む文字列

カテゴリ

セキュリティ関数、文字列関数

関数のシンタックス

```
GenerateSecretKey(algorithm [,keysize])
```

関連項目

Decrypt、Encrypt

履歴

ColdFusion 8: keysize 属性が追加されました。

ColdFusion MX 7: この関数が追加されました。

パラメータ

パラメータ	説明
algorithm	キーを生成する暗号化アルゴリズムです。ColdFusion では、次のアルゴリズムを使用する暗号ライブラリがインストールされます。 • AES: NIST (National Institute of Standards and Technology: 米国標準技術局) FIPS-197 で定義された Advanced Encryption Standard です。 • BLOWFISH: Bruce Schneier 氏が定義した Blowfish アルゴリズムです。 • DES: NIST FIPS-46-3 で定義された Data Encryption Standard アルゴリズムです。 • DESEDE: NIST FIPS-46-3 で定義された Triple DES アルゴリズムです。
keysize	指定したアルゴリズムのキーで要求するビット数です。 JDK で許可されている場合は、このパラメータを使用して、より長いキーを要求することもできます。たとえば、Java Unlimited Strength Jurisdiction Policy Files がインストールされていない場合、AES アルゴリズムのキーは 128 ビットまでに制限されます。詳細については、http://java.sun.com/products/jce/index-14.html を参照してください。

使用方法

GenerateSecretKey 関数を使用して、Encrypt 関数および Decrypt 関数の ColdFusion デフォルト暗号化アルゴリズム (CFMX_COMPAT) のキーを生成することはできません。

ColdFusion では、JCE (Java Cryptography Extension) を使用し、Sun JCE デフォルトセキュリティプロバイダを含む Sun Java ランタイムがインストールされます。このプロバイダには、前の「パラメータ」に示したアルゴリズムが含まれています。JCE フレームワークには、他のプロバイダの実装を使用するための機能も含まれています。ただし、Adobe ではサードパーティのセキュリティプロバイダに対するテクニカルサポートは提供していません。

例

次の例では、テキスト文字列を暗号化および復号します。暗号化アルゴリズムとエンコード方式を指定する必要があります。CFMX_COMPAT アルゴリズムで使用するキーシード用のフィールドもあります。その他のアルゴリズムの場合は、GenerateSecretKey 関数を使用してシークレットキーを生成します。

```
<h3>Decrypt Example</h3>

<!--- Do the following if the form has been submitted. --->
<cfif IsDefined("Form.myString")>
    <cfscript>
        /* GenerateSecretKey does not generate keys for the CFMX_COMPAT algorithm,
        so we use a key from the form.
        */
        if (Form.myAlgorithm EQ "CFMX_COMPAT")
            theKey=Form.MyKey;
        // For all other encryption techniques, generate a secret key.
        else
            theKey=generateSecretKey(Form.myAlgorithm);
        //Encrypt the string.
        encrypted=encrypt(Form.myString, theKey, Form.myAlgorithm,
                Form.myEncoding);
        //Decrypt it.
        decrypted=decrypt(encrypted, theKey, Form.myAlgorithm, Form.myEncoding);
    </cfscript>

    <!--- Display the values used for encryption and decryption,
            and the results. --->
    <cfoutput>
        <b>The algorithm:</b> #Form.myAlgorithm#<br>
        <b>The key:</B> #theKey#<br>
```

```
        <br>
        <b>The string:</b> #Form.myString# <br>
        <br>
        <b>Encrypted:</b> #encrypted#<br>
        <br>
        <b>Decrypted:</b> #decrypted#<br>
    </cfoutput>
</cfif>

<!--- The input form. --->
<form action="#CGI.SCRIPT_NAME#" method="post">
    <b>Select the encoding</b><br>
    <select size="1" name="myEncoding" >
        <option selected>UU</option>
        <option>Base64</option>
        <option>Hex</option>
    </select><br>
    <br>
    <b>Select the algorithm</b><br>
    <select size="1" name="myAlgorithm" >
        <option selected>CFMX_COMPAT</option>
        <option>AES</option>
        <option>DES</option>
        <option>DESEDE</option>
    </select><br>
    <br>
    <b>Input your key</b> (used for CFMX_COMPAT encryption only)<br>
    <input type = "Text" name = "myKey" value = "foobar"><br>
    <br>
    <b>Enter string to encrypt</b><br>
    <textArea name = "myString" cols = "40" rows = "5" WRAP = "VIRTUAL">This string will be encrypted (you
can replace it with more typing).
    </textArea><br>
    <input type = "Submit" value = "Encrypt my String">
</form>
```

GetApplicationMetadata

説明

アプリケーション（Application.cfc または Application.cfm）で指定したアプリケーション設定を返します。詳しくは、『CFML リファレンス』の「アプリケーション変数」の節を参照してください。

注意：ColdFusion Administrator の「グローバルなスクリプト保護の有効化」（サーバーの設定／設定）をオンにしている場合、戻り値は、保護されているデフォルトのスコープ（フォーム、URL、CGI または Cookie）になります。無効な値を指定している場合や、値を指定していない場合、ColdFusion Administrator の「グローバルなスクリプト保護の有効化」がオフに設定されている場合は、値は返されません。

注意：ColdFusion Administrator で VFS のメモリ制限を指定していて、「メモリ内仮想ファイルシステムのアプリケーションごとのメモリ制限」（サーバーの設定／設定）の値が Application.cfc（this.inmemoryfilesystem.size）で指定した値よりも小さい場合、戻り値は、Application.cfc ではなく、ColdFusion Administrator で指定されている値になります。

注意：Application.cfc または Application.cfm が見つからない場合は、「Application.cfc および Application.cfm の検索順序」（ColdFusion Administrator ／設定／サーバーの設定）で設定した順序でアプリケーションが検索されます。アプリケーションが見つからない場合は、空の構造体が返されます。

戻り値

アプリケーション設定（name、sessionManagement、invokeImplicitAccessor など）が含まれる構造体

カテゴリ
その他の関数

シンタックス
getApplicationMetadata()

例
この例では、getApplicationMetadata 関数を使用して、CFM でアプリケーション設定にアクセスする方法を示しています。

Application.cfc

```
component
{
    this.name = "myapp";
    this.applicationtimeout = createtimespan(0, 0, 0, 10);
    this.sessiontimeout = createtimespan(0, 0, 0, 10);
    this.sessionmanagement = true;
    this.clientmanagement = true;
    this.datasource = "cfartgallery";
    this.ormenabled = true;
    this.secureJSON = true;
    this.secureJSONPrefix = "///";
    this.setClientCookies = true;
    this.setDomainCookies = true;
    this.setClientStorage = "Registry";
    this.setLoginStorage = "Cookies";
    this.scriptProtect = "all";
    this.mappings["mymapping"] = getdirectoryfrompath(cgi.cf_template_path);
    this.customTagPaths = "path1,path2,path3";
    this.invokeImplicitAccessor = true;
    this.inmemoryfilesystem.enabled = true;
    this.inmemoryfilesystem.size = 10;
}
```

AppMetaData.cfm

```
<cfscript>
    writedump(getApplicationMetadata());
</cfscript>
```

GetAuthUser

説明
認証ユーザーの名前を取得します。

戻り値
認証ユーザーの名前

カテゴリ
セキュリティ関数

関数のシンタックス

```
GetAuthUser()
```

関連項目
cflogin、cfloginuser、cflogout、GetUserRoles、IsUserInAnyRole、IsUserInRole、IsUserLoggedIn、『ColdFusion アプリケーションの開発』の Securing Applications

履歴
ColdFusion MX: この関数が追加されました。

使用方法
この関数は、cflogin 認証または Web サーバー認証で使用できます。ログインユーザーのチェックは次のように行われます。

1. まず、cfloginuser で行われたログインがあるかがチェックされます。
2. cfloginuser でログインしているユーザーがない場合には、Web サーバーログイン (cgi.remote_user) がチェックされます。

例
```
<H3>GetAuthUser Example</H3>

<P>Authenticated User: <cfoutput>#GetAuthUser()#</cfoutput>
```

GetBaseTagData

説明
カスタムタグ内で使用されます。呼び出しタグ (祖先タグ) を名前で検索し、そのデータにアクセスします。

戻り値
祖先タグからのデータ (変数、スコープなど) が含まれているオブジェクト。指定された名前の祖先タグがない場合や、その祖先タグからデータ (たとえば cfif など) を取得できない場合は、例外が返されます。

カテゴリ
その他の関数

関数のシンタックス
```
GetBaseTagData(tagname [, instancenumber ])
```

関連項目
GetBaseTagList、『ColdFusion アプリケーションの開発』の High-level data exchange

パラメータ

パラメータ	説明
tagname	(必須) データを返す祖先タグ名です。
instancenumber	(オプション) データを返す前にジャンプする祖先タグのレベル数です。デフォルト値は 1 (最も近い祖先) です。

例

```
<!--- This example shows the use of GetBaseTagData
    function. Typically used in custom tags.--->
...
<cfif trim(inCustomTag) neq "">
        <cfoutput>
            Running in the context of a custom
            tag named #inCustomTag#.<p>
        </cfoutput>
        <!--- Get the tag instance data --->
        <cfset tagData = GetBaseTagData(inCustomTag)>
        <!--- Find the tag's execution mode --->
        Located inside the
        <cfif tagData.thisTag.executionMode neq 'inactive'>
            template
        <cfelse>
            BODY
        </cfif>
```

GetBaseTagList

説明

親タグから開始して、祖先タグの名前を取得します。

戻り値

大文字の祖先タグの名前をカンマで区切ったリストの文字列。リストの最初の要素は現在のタグです。現在のタグがネストされている場合は、次の要素が親タグになります。トップレベルのタグに対してこの関数を呼び出すと、空の文字列が返されます。祖先タグからデータを取得できない場合 (GetBaseTagData を参照)、その名前は返されません。

カテゴリ

その他の関数

関数のシンタックス

```
GetBaseTagList()
```

関連項目

GetBaseTagData、『ColdFusion アプリケーションの開発』の High-level data exchange

使用方法

この関数は、次のタグおよび終了タグを祖先タグのリストに表示しません。

- cfif、cfelseif、cfelse
- cfswitch、cfcase、cfdefaultcase
- cftry、cfcatch

この関数は、次の条件でのみ次のタグを表示します。

- cfloop: query 属性を使用している場合
- cfoutput: 複雑な式を持つ子タグが 1 つでもある場合
- cfprocessingdirective: pageencoding 以外にその他の属性が 1 つでもある場合

例

```
<!--- This example shows the use of GetBaseTagList function.
Typically used in custom tags. --->
<cfif thisTag.executionMode is "start">
    <!--- Get the tag context stack
    The list will look something like "CFIF,MYTAGNAME..." --->
    <cfset ancestorList = GetBaseTagList()>
<br><br>Dump of GetBaseTagList output:<br>
    <cfdump var="#ancestorList#"><br><br>
    <!--- Output current tag name --->
    <cfoutput>This is custom tag#ListGetAt(ancestorList,1)#</cfoutput><br>
    <!--- Determine whether this is nested inside a loop --->
    <cfset inLoop = ListFindNoCase(ancestorList, "cfloop")>
    <cfif inLoop>
        Running in the context of a cfloop tag.<br>
    </cfif>
</cfif>
```

GetBaseTemplatePath

説明

アプリケーションのベースページの絶対パスを取得します。

戻り値

アプリケーションのベースページの絶対パスの文字列

カテゴリ

その他の関数、システム関数

関数のシンタックス

```
GetBaseTemplatePath()
```

関連項目

GetCurrentTemplatePath、FileExists、ExpandPath

例

```
<h3>GetBaseTemplatePath Example</h3>

<p>The template path of the current page is:
<cfoutput>#GetBaseTemplatePath()#</cfoutput>
```

GetClientVariablesList

説明

ページからの書き込みアクセスが可能なクライアント変数を検索します。

戻り値

読み取り専用でないクライアント変数のカンマ区切りリストを含む文字列

カテゴリ

リスト関数、その他の関数

関数のシンタックス

```
GetClientVariablesList()
```

関連項目

DeleteClientVariable

使用方法

この関数によって返される変数のリストは、ColdFusion のリスト関数で使用できます。

例

```
<h3>GetClientVariablesList Example</h3>

<!--- The following line enables client variables.
      You would normally do this in Application.cfc.--->
<cfapplication clientmanagement="yes">

<p>This example creates two client variables and deletes the User_ID client variable,
if it exists in the list of client variables returned by GetClientVariablesList().</p>

<cfset client.somevar = "">
<cfset client.User_ID = "">
<p>Client variable list: <cfoutput>#GetClientVariablesList()#</cfoutput></p>

<cfif ListFindNoCase(GetClientVariablesList(), "User_ID") is not 0>
    Delete client.User_ID variable.
       <cfset temp = DeleteClientVariable("User_ID")>
       <p>Was variable "User_ID" Deleted? <cfoutput>#temp#</cfoutput>
</cfif>

<p>Amended Client variable list: <cfoutput>#GetClientVariablesList()#
</cfoutput>
```

GetComponentMetaData

説明

CFC または ColdFusion インターフェイスのメタデータ (コンポーネントの関数や実装されるインターフェイスなど) を取得します。

戻り値

CFC またはインターフェイスのメタデータを含む構造体。構造体の内容については、GetMetaData の「使用方法」の表に記載されているコンポーネントの説明を参照してください。

カテゴリ

拡張関数

関数のシンタックス

```
GetComponentMetaData(path)
```

関連項目

GetMetaData

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
path	インターフェイスまたは CFC 定義のドット区切りのパスです。 現在のディレクトリまたは Web ルートからの相対パスを指定できます。たとえば、この関数を呼び出す CFM ページが <Web ルート >/my_apps/interfaces にあり、インターフェイスファイルが <Web ルート >/my_apps/interfaces/definitions にある場合、I2.cfc で定義されているインターフェイスのメタデータを取得するには、このパラメータで次のいずれかの値を指定します。 • definitions.I2 • my_apps.interface.definitions.I2.cfc

使用方法
この関数と getMetaData 関数は同じデータを返します。ただし、この関数は CFC またはインターフェイス定義ファイルへのパスをパラメータとして取ります。また、この関数はオブジェクトのインスタンスを使用したり、作成することはありません。この関数で取得できるのは CFC とインターフェイスに関するデータのみです。getMetaData 関数ではインターフェイスを指定できません。

GetContextRoot

説明
現在のリクエストの J2EE サーバーコンテキストルートへのパスを返します。

戻り値
Web ルートから現在のページのコンテキストルートへのパス。このパスの先頭にはスラッシュ (/) が付きますが、末尾には付きません。デフォルト (ルート) コンテキストのアプリケーションの場合は、空の文字列が返されます。

カテゴリ
システム関数

履歴
ColdFusion MX 7: この関数が追加されました。

関数のシンタックス
```
GetContextRoot()
```

関連項目
GetPageContext

使用方法
この関数は、GetPageContext().GetRequest().GetContextPath() を呼び出した場合と同じです。J2EE 設定では、この関数は、Web ルートから ColdFusion J2EE アプリケーションの J2EE コンテキストルートへのパスを返します。サーバー設定では、コンテキストルートは Web ルートにあるので、この関数は空の文字列を返します。

この関数は、複数のアプリケーションがさまざまな J2EE コンテキストルートに分散してインストールされている場合に便利です。

例
ColdFusion Administrator では、次の行を使用して administrator ディレクトリの場所を取得します。

```
<cfset request.thisURL = "#getContextRoot()#/CFIDE/administrator/">
```

Administrator では、次の行のように、イメージなどの Administrator リソースにアクセスするために URL を使う場所で戻り値を使用します。

```
<a href="index.cfm"><img src="#request.thisURL#images/back.gif" width="16"
            height="16" border="0" alt=" "></a>
```

GetCPUUsage

説明
デフォルトまたはカスタムのスナップショット間隔で CPU 使用率を取得します。デフォルトの間隔は 1000 ミリ秒です。

シンタックス
getCPUUsage()

getCPUUsage(long ms)

パラメータ

パラメータ	説明
long ms	ミリ秒単位の時間です。これは、2 つのスナップショットの時間間隔です。

GetCurrentTemplatePath

説明
この関数を呼び出すページのパスを取得します。

戻り値
この関数の呼び出しを含んでいるページの絶対パスの文字列

カテゴリ
システム関数

関数のシンタックス
```
GetCurrentTemplatePath()
```

関連項目
GetBaseTemplatePath、FileExists、ExpandPath

使用方法
cfinclude タグでインクルードされているページから関数が呼び出された場合、この関数はインクルードされたページのページパスを返します。インクルードされたページから呼び出された場合でもトップレベルのページのパスを返す GetBaseTemplatePath 関数とは対照的です。

例

```
<!--- This example uses GetCurrentTemplatePath to show the
    template path of the current page --->
<h3>GetCurrentTemplatePath Example</h3>

<p>The template path of the current page is:
<cfoutput>#GetCurrentTemplatePath()#</cfoutput>
```

GetDirectoryFromPath

説明

ディスク上またはメモリ内の絶対パスからディレクトリを抽出します。

戻り値

ファイル名を除いた絶対パス。最後の文字はスラッシュまたは円記号です (オペレーティングシステムによって異なります)。

カテゴリ

システム関数

関数のシンタックス

```
GetDirectoryFromPath(path)
```

関連項目

ExpandPath、GetFileFromPath

パラメータ

パラメータ	説明
path	ディスク上またはメモリ内の絶対パスです。

使用方法

例

```
<h3>GetDirectoryFromPath Example</h3>
<cfset thisPath = ExpandPath("*.*")>
<cfset thisDirectory = GetDirectoryFromPath(thisPath)>
<cfoutput>
The current directory is: #GetDirectoryFromPath(thisPath)#
<cfif IsDefined("FORM.yourFile")>
    <cfif FORM.yourFile is not "">
        <cfset yourFile = FORM.yourFile>
            <cfif FileExists(ExpandPath(yourfile))>
            <p>Your file exists in this directory. You entered the correct filename,
        #GetFileFromPath("#thisPath#/#yourfile#")#
    <cfelse>
        <p>Your file was not found in this directory:
        <br>Here is a list of the other files in this directory:
        <!--- use cfdirectory show directory, order by name & size --->
        <cfdirectory directory = "#thisDirectory#"
            name = "myDirectory" SORT = "name ASC, size DESC">
        <!--- Output the contents of the cfdirectory as a CFTABLE --->
        <cftable query = "myDirectory">
            <cfcol header = "NAME:" text = "#Name#">
        <cfcol header = "SIZE:" text = "#Size#">
        </cftable>
        </cfif>
    </cfif>
<cfelse>
    <H3>Please enter a filename</H3>
    </cfif>
    </cfoutput>
    <form action="getdirectoryfrompath.cfm" METHOD="post">
        <H3>Enter the name of a file in this directory <I><FONT SIZE="-1">
    (try expandpath.cfm)</FONT></I></H3>
        <input type="Text" NAME="yourFile">
        <input type="Submit" NAME="">
    </form> --->
```

GetEncoding

説明

Form スコープまたは URL スコープのエンコード (文字セット) を返します。

戻り値

文字列。指定したスコープの文字エンコードです。

カテゴリ

各国語対応関数、システム関数

関数のシンタックス

```
GetEncoding(scope_name)
```

関連項目

cfcontent、cfprocessingdirective、URLDecode、URLEncodedFormat

履歴
ColdFusion MX: この関数が追加されました。

パラメータ

パラメータ	説明
scope_name	• フォーム • URL

使用方法
この関数を使うと、URL クエリー文字列の文字エンコードや、現在のページに送信されたフォームのフィールドの文字エンコードを特定できます。明示的に設定されているエンコードがない場合のデフォルトのエンコードは UTF-8 です。

詳細については、www.iana.org/assignments/character-sets を参照してください。

例
```
<!--- This example sends the contents of two fields and interprets them as
        big5 encoded text. Note that the form fields are received as URL variables because the form uses
the GET method.--->
<cfcontent type="text/html; charset=big5">
<form action='#cgi.script_name#' method='get'>
<input name='xxx' type='text'>
<input name='yyy' type='text'>
<input type="Submit" value="Submit">
</form>

<cfif IsDefined("URL.xxx")>
<cfscript>
    SetEncoding("url", "big5");
    WriteOutput("URL.XXX is " & URL.xxx & "<br>");
    WriteOutput("URL.YYY is " & URL.yyy & "<br>");
theEncoding = GetEncoding("URL");
     WriteOutput("The URL variables were decoded using '" & theEncoding & "' encoding.");

WriteOutput("The encoding is " & theEncoding);
</cfscript>
</cfif>
```

GetException

説明
cftry タグおよび cfcatch タグとともに使用します。Java オブジェクトからの Java 例外オブジェクトを取得します。

戻り値
Java オブジェクトに対する以前のメソッド呼び出しによって生じた Java 例外オブジェクト

カテゴリ
システム関数

シンタックス
```
GetException(object)
```

パラメータ

パラメータ	説明
object	Java オブジェクトです。

使用方法

ColdFusion では、Java オブジェクトへの各メソッド呼び出しに対する例外オブジェクトが保管されます。次のメソッド呼び出しによって、この例外オブジェクトはリセットされます。現在の例外オブジェクトを取得するには、その Java オブジェクトに対して他のメソッドが呼び出される前に、GetException を呼び出します。

例

```
<!--- Create the Java object reference --->
<cfobject action = create type = java class = primativetype name = myObj>
<!--- Calls the object's constructor --->
<cfset void = myObj.init()>
<cftry>
<cfset void = myObj.DoException() >
<Cfcatch type = "Any">
    <cfset exception = GetException(myObj)>
<!--- User can call any valid method on the exception object.--->
    <cfset message = exception.toString()>
    <cfoutput>
        Error<br>
        I got exception <br>
        <br> The exception message is: #message# <br>
    </cfoutput>
</cfcatch>
</cftry>
```

GetFileFromPath

説明

ディスク上またはメモリ内の絶対パスからファイル名を抽出します。

戻り値

ファイル名の文字列

カテゴリ

システム関数

関数のシンタックス

```
GetFileFromPath(path)
```

関連項目

ExpandPath、GetCurrentTemplatePath

パラメータ

パラメータ	説明
path	ディスク上またはメモリ内の絶対パスです。

例

```
<h3>GetFileFromPath Example</h3>
<cfset thisPath = ExpandPath("*.*")>
<cfset thisDirectory = GetDirectoryFromPath(thisPath)>
<cfoutput>
The current directory is: #GetDirectoryFromPath(thisPath)#
<cfif IsDefined("FORM.yourFile")>
<cfif FORM.yourFile is not "">
<cfset yourFile = FORM.yourFile>
<cfif FileExists(ExpandPath(yourfile))>
        <p>Your file exists in this directory. You entered the correct file
    name, #GetFileFromPath("#thisPath#/#yourfile#")#
    <cfelse>
        <p>Your file was not found in this directory:
        <br>Here is a list of the other files in this directory:
        <!--- use CFDIRECTORY to give the contents of the snippets
    directory, order by name and size --->
        <CFDIRECTORY
            DIRECTORY = "#thisDirectory#"
            name = "myDirectory"
            SORT = "name ASC, size DESC">
        <!--- Output the contents of the CFDIRECTORY as a CFTABLE --->
        <CFTABLE QUERY = "myDirectory">
        <CFCOL HEADER = "NAME:" TEXT = "#Name#">
    <CFCOL HEADER = "SIZE:" TEXT = "#Size#">
...
```

GetFileInfo

説明

ディスク上またはメモリ内のファイルに関する情報を取得します。

戻り値

ファイル名、パス、親ディレクトリ、種類、サイズ、最終変更日時、アクセス許可 (読み込み、書き込み、非表示) が設定されているかどうか

カテゴリ

システム関数

関数のシンタックス

```
GetFileInfo(path)
```

関連項目

FileOpen

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
path	ディスク上またはメモリ内の絶対パスです。

使用方法
この関数は、次のキーを含む構造体を返します。

- Name: ファイルの名前
- Path: ファイルの絶対パス
- Parent: ファイルの親ディレクトリのパス
- Type: "directory" または "file"
- Size: ファイルサイズ (バイト単位)
- Lastmodified: ファイルの最後変更日時
- canRead: 読み込みが許可されているかどうか
- canWrite: 書き込みが許可されているかどうか
- isHidden: 非表示かどうか

例
```
<cfscript>
    FileSetLastModified("c:\temp\test1.txt", "#Now()#");
    WriteOutput(GetFileInfo("c:\temp\test1.txt").lastmodified);
</cfscript>
```

GetFreeSpace

説明
ハードディスクの空き容量またはメモリ内 VFS の空き容量に関する情報を取得します。

戻り値
空き容量（バイト単位）を返します。

シンタックス
getFreeSpace(path)

パラメータ

パラメータ	説明
path	次のパス • ハードディスクドライブ（例：C:（Windows の場合）、/（UNIX の場合））。Windows の場合は、C だけでなく、コロン（:）を明示的に使用する必要があることに注意してください。ハードディスクドライブ全体の容量のみが返されます。パス内にディレクトリを指定した場合（例：/opt/）、ディレクトリは無視されて / のみが考慮されます。 • メモリ内ファイルシステムの場合のパスは ram: です。

使用方法
GetTotalSpace の「使用方法」の節を参照してください。

例

次の例では、ColdFusion Administrator で、アプリケーション用のメモリ内ファイルシステムのメモリが 20 MB に設定されています。関数では 20 が返され、これは合計容量が 20 MB と見なされていることを意味します。これは、ColdFusion Administrator で指定されている値（「メモリ内仮想ファイルシステムのアプリケーションごとのメモリ制限」）が、Application.cfc で指定されている値より小さい（20 MB である）からです。

Application.cfc

```
<cfcomponent>
    <cfset this.name = "vfs_total_space">
    <cfset this.inmemoryfilesystem.size = 30>
</cfcomponent>
```

space.cfm

```
<cftry>
    <cfset totalRAMSpace = getTotalSpace("ram:")>
    <cfset freeRAMSpace = getFreeSpace("ram:")>
    <cfset totalDiskSpace = getTotalSpace("c:")>
    <cfset freeDiskSpace = getFreeSpace("c:")>
    <cfoutput>
        Total Hard Disk Space = #DecimalFormat(totalDiskSpace / (1024 * 1024 * 1024))# GB
        <br>Free Hard Disk Space = #DecimalFormat(freeDiskSpace / (1024 * 1024 * 1024))# GB
        <br>Total Application RAM Memory = #DecimalFormat(totalRAMSpace / (1024 * 1024))# MB
        <br>Free Application RAM Memory = #DecimalFormat(freeRAMSpace / (1024 * 1024))# MB
        <br>
    </cfoutput>
<cfcatch type="any">
    <cfoutput>
        #cfcatch.message#
        <br>#cfcatch.detail#
        <br>
    </cfoutput>
</cfcatch>
</cftry>
```

GetFunctionCalledName

説明

定義された関数を呼び出すために使用された変数の名前を返します。

戻り値

変数の名前。

カテゴリ

システム関数

関数のシンタックス

```
GetFunctionCalledName()
```

履歴

ColdFusion 9: この関数が追加されました。

使用方法

この関数は、getter と setter をシミュレートして CFC からデータを返す目的に使用できます。この方法を適用できるのは、ColdFusion に用意されている暗黙的な getter と setter を CFC で使用しない場合のみです。

例

次の例は、明示的に setter と getter を定義せずに、この関数を使用してデータを返す方法を示しています。

```
//callednamedemo.cfc
component
{
    variables.x1 = 1;
    variables.y1 = 2;
    function init()
    {
        return this;
    }
    function get()
    {
        var name = getFunctionCalledName();
        return variables[mid(name,4,len(name)-3)];
    }
    function set(value)
    {
        var name = getFunctionCalledName();
        variables[mid(name,4,len(name)-3)] = value;
    }
    this.getX1 = get;
    this.getY1 = get;
    this.setX1 = set;
    this.setY1 = set;
}
<!--- callednamecfm --->
<cfscript>
    function test()
    {
        return getFunctionCalledName();
    }
    WriteOutput(test() & "<br>"); // test
    a = test;
WriteOutput(variables.a() & "<br>"); // a
o = new callednamedemo();
// shows *real* methods get(), SetX1() and getY1(), etc.
    writeDump(o);
    o.setX1(10);
    o.setY1(20);
    WriteOutput(o.getX1() & "<br>"); // 10
    WriteOutput(o.getY1() & "<br>") ; // 20 </cfscript>
```

GetFunctionList

説明

ColdFusion で使用できる関数のリストを表示します。

戻り値

関数の構造体

カテゴリ
システム関数

関数のシンタックス
```
GetFunctionList()
```

例
```
<!----- This example shows the use of GetFunctionList. ---->
<cfset fList = GetFunctionList()>
<cfoutput>#StructCount(fList)# functions<br><br>
    </cfoutput>
<cfloop COLLECTION = "#fList#" ITEM = "key">
    <cfoutput>#key#<br>
    </cfoutput>
</cfloop>
```

GetGatewayHelper

説明
ColdFusion イベントゲートウェイに使用するためのメソッドとプロパティを提供する Java GatewayHelper オブジェクトを取得します。

戻り値
Java GatewayHelper オブジェクト

カテゴリ
拡張関数

関数のシンタックス
```
GetGatewayHelper(gatewayID)
```

関連項目
SendGatewayMessage

履歴
ColdFusion MX 7: この関数が追加されました。

パラメータ

パラメータ	説明
gatewayID	GatewayHelper オブジェクトを提供するゲートウェイの識別子です。ColdFusion Administrator の [イベントゲートウェイ] セクションの [ゲートウェイ] ページで設定された、いずれかの ColdFusion イベントゲートウェイインスタンスのゲートウェイ ID でなければなりません。

使用方法
ColdFusion GetGatewayHelper 関数は、イベントゲートウェイ固有のヘルパメソッドおよびプロパティを提供する Java GatewayHelper オブジェクトを返します。この関数を使用するには、GatewayHelper クラスを実装するクラスへのアクセスをイベントゲートウェイが提供する必要があります。たとえば、インスタントメッセージングイベントゲートウェイにより、バディリスト管理関数を GatewayHelper オブジェクトで使用できるようにすることができます。

イベントゲートウェイリスナー CFC は、着信メッセージの CFEvent 構造体から gatewayID の値を取得することができます。

標準の ColdFusion Java オブジェクトアクセステクニックを使用して、GatewayHelper オブジェクトのメソッドおよびプロパティにアクセスします。詳細については、『ColdFusion アプリケーションの開発』の The role of the GatewayHelper object を参照してください。

例

イベントゲートウェイのヘルパクラスに、単一の String パラメータを取る addBuddy メソッドが含まれている場合、次のコードを使用して、GatewayHelper オブジェクトを取得し、バディをバディリストに追加することができます。

```
<h3>GetGatewayHelper Example</h3>
<cfscript>
    myHelper = getGatewayHelper(myGatewayID);
    status = myHelper.addBuddy("jsmith");
</cfscript>
```

GetHttpRequestData

説明

HTTP リクエストヘッダおよび本文を CFML ページで使用できるようにします。HTTP ヘッダ内で送信される SOAP リクエストデータを取り込む際に便利です。

戻り値

ColdFusion 構造体です。

カテゴリ

システム関数

関数のシンタックス

```
GetHttpRequestData()
```

戻り値

この関数は、次のエントリを含む構造体を返します。

構造体の要素	説明
content	クライアントから送信されたフォームからの、文字列またはバイナリ形式の未処理のコンテンツです。文字列データと見なされるコンテンツでは、FORM のリクエストヘッダ "CONTENT_TYPE" は "text/" または "application/x-www-form-urlencoded" で始まる必要があります。他の型はバイナリオブジェクトとして保管されます。
headers	HTTP リクエストヘッダを値のペアとして含む構造体です。これには、SOAP リクエストなどのカスタムヘッダがあります。
method	CGI 変数 Request_Method を含んでいる文字列です。
protocol	CGI 変数 Server_Protocol を含んでいる文字列です。

使用方法

データがバイナリであるかどうかを確認するには、IsBinary(x.content) を使用します。データが文字列として表示可能な場合、データを文字列の値に変換するには ToString(x.content) を使用します。

例

次の例は、GetHttpRequestData が HTTP ヘッダ情報を返す方法を示しています。

```
<cfset x = GetHttpRequestData()>
<cfoutput>
<table cellpadding = "2" cellspacing = "2">
<tr>
     <td><b>HTTP Request item</b></td>
     <td><b>Value</b></td> </tr>
<cfloop collection = #x.headers# item = "http_item">
        <tr>
        <td>#http_item#</td>
        <td>#StructFind(x.headers, http_item)#</td></tr>
</cfloop>
<tr>
    <td>request_method</td>
    <td>#x.method#</td></tr>
<tr>
    <td>server_protocol</td>
    <td>#x.protocol#</td></tr>
</table>
<b>http_content --- #x.content#</b>
</cfoutput>
```

GetHttpTimeString

説明

現在の世界標準時 (UTC) を取得します。

戻り値

RFC 1123 およびその基盤となる RFC 822 に記述された HTTP 標準規格に準拠する時刻の文字列。通常、この形式は HTTP などのインターネットプロトコルに使用されます。

カテゴリ

日付および時刻関数、各国語対応関数

関数のシンタックス

```
GetHttpTimeString(date_time_object)
```

関連項目

GetLocale、GetTimeZoneInfo、SetLocale

パラメータ

パラメータ	説明
date_time_object	ColdFusion の日付時刻オブジェクトまたは Java Date オブジェクトです。

使用方法

返される文字列中の時刻は UTC であり、HTTP 標準規格に準拠しています。

例

```
<cfoutput>
    #GetHttpTimeString(now())#
</cfoutput>
```

GetK2ServerDocCountLimit

説明

この関数は非推奨となっています。

戻り値

K2 サーバーで許されるコレクションメタデータ項目の数 (整数)

カテゴリ

全文検索関数、クエリー関数

関数のシンタックス

```
GetK2ServerDocCountLimit()
```

履歴

ColdFusion MX 6.1: この関数が非推奨になりました。今後のリリースでは、動作せずにエラーとなる可能性があります。

ColdFusion MX: この関数が追加されました。

使用方法

検索したドキュメント数が制限を超えた場合、ColdFusion では、Administrator およびそのログファイルに警告メッセージを挿入します。

例

```
<cfoutput>GetK2ServerDocCountLimit =
    $*#GetK2ServerDocCountLimit()#*$</cfoutput>
```

GetLocale

説明

ColdFusion の現在の地理 / 言語ロケールの値を取得します。

ColdFusion アプリケーションセッションで、日付、時刻、数値、および通貨値のデフォルトの表示形式を設定するには、SetLocale 関数を使用します。

戻り値

現在のロケール値の文字列 (英語)。Java 名および ColdFusion MX 7 のリリース前に ColdFusion で使用されていた名前 (en_US および English (US) など) を持つロケールの場合、ColdFusion は ColdFusion 名 (English (US) など) を返します。

カテゴリ

表示および書式制御関数、各国語対応関数、システム関数

関数のシンタックス

```
GetLocale()
```

関連項目

GetLocaleDisplayName、SetLocale

履歴

ColdFusion MX 7: すべての Java ロケールおよびロケール名に対するサポートが追加されました。

ColdFusion MX:「使用方法」に記載されているように動作が変更されました。

使用方法

この関数は、ColdFusion での表記どおりのロケール名を返します。たとえば、Portuguese (Brazilian)、ca_ES などです。ロケールの言語で表記されたロケール名を取得するには、GetLocaleDisplayName 関数を使用します。この関数は、português (Brasil) や português (Espanya) などを返します。

この関数を使用すると、ColdFusion に対してロケール値が設定されているかどうかを確認できます。ロケール値を設定するには SetLocale 関数を使用します。

- ColdFusion ロケール値が存在する場合は、その値が返されます。
- ColdFusion ロケールが明示的に設定されていない場合は、ColdFusion サーバーコンピュータのオペレーティングシステムのデフォルトのロケール値が ColdFusion でサポートされるロケール値の中に含まれているかどうかが確認されます (デフォルトのロケールは、ユーザー環境変数 user.language および user.region に格納されています)。

  デフォルトのロケール値がサポートされていない場合、GetLocale は English (US) を返します。ColdFusion は JVM のロケールをこの値に設定します。この値は、サーバーを再起動するか、SetLocale 関数を使って値をリセットするまで保持されます。

この関数では、Web ブラウザの Accept-Language HTTP ヘッダ設定へのアクセスは行われません。

注意：ColdFusion の起動時には、サポートされているロケール値が変数 Server.ColdFusion.SupportedLocales に格納されます。ColdFusion は、Java ランタイム環境でサポートされるロケールをサポートします。SupportedLocales 値には、サポートされているロケールの Java 名、および ColdFusion MX 7 リリース前の ColdFusion が使用していた対応ロケール名がリストされます。

詳細については、『ColdFusion アプリケーションの開発』の Locales を参照してください。

例

```
<h3>Example: Using SetLocale and GetLocale</h3>
<cfoutput>
    <!--- For each new request, the locale gets reset to the JVM locale --->
    Initial locale's ColdFusion name: #GetLocale()#<br>
    <br>
    <!--- Do this only if the form was submitted. --->
    <cfif IsDefined("form.mylocale")>
        <b>Changing locale to #form.mylocale#</b><br>
        <br>
        <!--- Set the locale to the submitted value and save the old ColdFusion locale name.--->
        <cfset oldlocale=SetLocale("#form.mylocale#")>
        <!--- Get the current locale. It should have changed. --->
        New locale: #GetLocale()#<br>
    </cfif>

    <!--- Self-submitting form to select the new locale. --->
    <cfform>
        <h3>Please select the new locale:</h3>
        <cfselect name="mylocale">
            <!--- The server.coldfusion.supportedlocales variable is a
                    list of all supported locale names. Use a list cfloop tag
                    to create an HTML option tag for each name in the list. --->
            <cfloop index="i" list="#server.coldfusion.supportedlocales#">
                <option value="#i#">#i#</option>
            </cfloop>
        </cfselect><br>
        <br>
        <cfinput type="submit" name="submitit" value="Change Locale">
    </cfform>
</cfoutput>
```

GetLocaleDisplayName

説明

ロケール値を取得し、それぞれのロケールに適した方法でその名前を表示します。デフォルトでは、現在のロケールをそのロケールの言語で取得します。

戻り値

指定されたロケールの言語で表記されたロケールの表示名

カテゴリ

表示および書式制御関数、各国語対応関数、システム関数

関数のシンタックス

```
GetLocaleDisplayName([locale, inLocale])
```

関連項目

GetLocale、SetLocale

履歴

ColdFusion MX 7: この関数が追加されました。

パラメータ

パラメータ	説明
locale	名前を調べるロケールです。デフォルト値は、現在の ColdFusion のロケールです。ColdFusion のロケールが設定されていない場合は、JVM のロケールです。
inlocale	名前を返す際に使われるロケールです。デフォルト値は、現在の ColdFusion のロケールです。ColdFusion のロケールが設定されていない場合は、JVM のロケールです。

例

次の例では、GetLocale の例を展開し、GetLocaleDisplayName 関数を使用して現在のロケールまたはその他のロケールでロケール名を表示する方法を示しています。サポートされるすべてのロケールからロケールを選択することが可能です。ColdFusion ロケールは選択されたロケールに変更され、新旧のロケール名が表示されます。

```
<html>
<head>
    <title>Displaying locales</title>
</head>

<body>
<h3>Example: Changing and Displaying Locales</h3>
<cfoutput>
    <!--- For each new request, the locale gets reset to the JVM locale --->
    Initial locale's ColdFusion name: #GetLocale()#<br>
    Initial locale's display name: #GetLocaleDisplayName()#<br>
    <br>
    <!--- Do this only if the form was submitted. --->
    <cfif IsDefined("form.mylocale")>
        <b>Changing locale to #form.mylocale#</b><br>
        <br>
        <!--- Set the locale to the submitted value.
            SetLocale returns the old ColdFusion locale name.--->
        <cfset oldlocale=SetLocale("#form.mylocale#")>
        <!--- Get the current locale's ColdFusion name.
            It should have changed. --->
        <cfset newlocale=GetLocale()>
        New locale's ColdFusion name: #newlocale#<br>
        New locale's display name in current locale: #GetLocaleDisplayName()#<br>
        New locale's display name in old locale:
            #GetLocaleDisplayName(newlocale, oldlocale)#<br>
        New locale's display name in en_US:
            #GetLocaleDisplayName(newlocale, "en_US")#<br>
        <br>
        Old locale's display name in current locale:
            #GetLocaleDisplayName(oldlocale)#<br>
        Old locale's display name in en_US:
            #GetLocaleDisplayName(oldlocale, "en_US")#<br>
    </cfif>
```

```
    <!--- Self-submitting form to select the new locale. --->
    <cfform>
        <h3>Please select the new locale:</h3>
        <cfselect name="mylocale">
            <!--- The server.coldfusion.supportedlocales variable is a
                    list of all supported locale names. Use a list cfloop tag
                    to create an HTML option tag for each name in the list. --->
            <cfloop index="i" list="#server.coldfusion.supportedlocales#">
            <!--- In the select box, we use the US English display names for
                    the locales. You can change en_US to your prefered locale. --->
                <option value="#i#">#GetLocaleDisplayName(i, "en_US")#</option>
            </cfloop>
        </cfselect><br>
        <br>
        <cfinput type="submit" name="submitit" value="Change Locale">
    </cfform>
</cfoutput>

</body>
</html>
```

GetLocalHostIP

説明
ローカルホストの IP アドレスを返します。IPv4 アドレスの場合は 127.0.0.1、IPv6 アドレスの場合は ::1 です。

戻り値
ローカルホストの IP アドレス

カテゴリ
決定関数

関数のシンタックス
```
GetLocalHostIP()
```

関連項目
IsLocalHost

履歴
ColdFusion MX 7.01: この関数が追加されました。

例
```
<h3>GetLocalHostIP Example</h3>

<cfoutput>The localhost IP address is #GetLocalHostIP()#.</cfoutput>
```

GetMetaData

説明
ColdFusion サーバー上にデプロイされたオブジェクトに関連するメタデータ (コンポーネントのメソッド、プロパティ、パラメータなど) を取得します。

戻り値
構造化されたメタデータ情報 : ColdFusion コンポーネント (CFC) およびユーザー定義関数 (UDF) の場合は、構造体。クエリーオブジェクトの場合は、構造体の配列。

カテゴリ
システム関数

関数のシンタックス
```
GetMetaData(object)
```

関連項目
CreateObject、GetComponentMetaData、QueryAddColumn、QueryNew

履歴
ColdFusion MX 7: クエリーオブジェクトのメタデータを取得できるようになりました。

ColdFusion MX: この関数が追加されました。

パラメータ

パラメータ	説明
object	ColdFusion コンポーネント、ユーザー定義関数、またはクエリーオブジェクトです。CFC 内では、このパラメータを使用して This スコープを指定することもできます。

使用方法
この関数は、アプリケーションデータに関する情報を提供し、オブジェクトの構造体およびその使用方法をアプリケーションでダイナミックに確定できるようにします。この関数は、CFC およびクエリーオブジェクトに役立ちます。CFC のメタデータは、コンポーネントに関する情報、その関数、引数、およびプロパティに関する情報などです。getMetaData 関数は、CFC の一部ではないユーザー定義関数のメタデータも返します。まだインスタンス化していない CFC 定義、または ColdFusion インターフェイスに関する情報を取得するには、GetComponentMetaData 関数を使用します。

次の表に、この関数によって返されるデータを、サポートされるオブジェクトタイプ別に示します。

オブジェクト	フィールド	説明
コンポーネント		次のフィールドを含む構造体です。

オブジェクト	フィールド	説明
	displayname	存在する場合には、cfcomponent タグの displayname 属性の値です。
	extends	コンポーネントの祖先コンポーネントのメタデータです。他のコンポーネントを明示的に拡張しないコンポーネントは、WEB-INF.cftags.component を拡張します。
	fullname	ColdFusion の Web ルートからコンポーネントまでのドット区切りパスです。
	functions	コンポーネントの関数に関するメタデータ構造体の配列です。
	hint	存在する場合には、cfcomponent タグの displayname 属性の値です。
	implements	コンポーネントが実装するインターフェイスのメタデータを含む構造体です。構造体のキーでインターフェイス名を指定し、値でインターフェイスのメタデータを含む構造体を指定します。
	name	Web ルートディレクトリ、ColdFusion Administrator の [カスタムタグのパス] ページに指定されているディレクトリなどのコンポーネント検索ルートからのピリオド区切りのパスを含む、コンポーネント名です。
	output	存在する場合には、cfcomponent タグの output 属性の値です。
	path	コンポーネントの絶対パスです。
	properties	存在する場合には、コンポーネントの cfproperty タグで指定されるメタデータを含む構造体の配列です。
	type	常にコンポーネントです。
	userMetadata	cfcomponent タグ内のユーザー指定の属性です。
関数		次のフィールドを含む構造体です。
	access	存在する場合には、cffunction タグの access 属性の値です。
	displayname	存在する場合には、cffunction タグの displayname 属性の値です。
	hint	存在する場合には、cffunction タグの hint 属性の値です。
	name	関数名です。
	output	存在する場合には、cffunction タグの output 属性の値です。
	parameters	関数のパラメータに関するメタデータを含む構造体の配列です。
	returnformat	リモートの呼び出し元に返される値の形式です。この属性は、ローカルの呼び出し元に返される値の形式には影響しません。
	returntype	存在する場合には、cffunction タグの returntype 属性の値です。
	roles	存在する場合には、cffunction タグの output 属性の値です。
	userMetadata	cffunction タグ内のユーザー指定の属性です。
パラメータまたはプロパティ		次のフィールドを含む構造体です。
	default	存在する場合には、cfargument または cfproperty タグの default 属性の値です。
	displayname	存在する場合には、cfargument または cfproperty タグの displayname 属性の値です。
	hint	存在する場合には、cfargument または cfproperty タグの hint 属性の値です。
	name	関数のパラメータまたは CFC プロパティの名前です。
	required	存在する場合には、cfargument または cfproperty タグの required 属性の値です。
	type	存在する場合には、cfargument または cfproperty タグの type 属性の値です。
	userMetadata	argument タグ内のユーザー指定の属性です。

オブジェクト	フィールド	説明
クエリー		次の要素を含む構造体の配列です。
	IsCaseSensitive	文字データの大文字と小文字を正しく指定する必要があるかどうかを指定するブール値です。
	Name	列名です。
	TypeName	SQL データ型です。型を指定しないで QueryNew でクエリーオブジェクトが作成される場合は、省略されます。

注意：This スコープを使用して、CFC 内のコンポーネントのメタデータにアクセスします。This スコープは、実行時に、コンポーネント本体および CFC メソッドで使用できます。これを使用して、コンポーネントが有効な間に存在する変数の読み込みと書き込みを行います。

詳細については、『ColdFusion アプリケーションの開発』の Using introspection to get information about components を参照してください。

例

次の例では、cfdump タグを使用して、ColdFusion コンポーネントブラウザで使用するユーティリティ CFC のメタデータを表示します。cfdocexamples データベースの Employees テーブル内のフィールドの名前とデータ型も表示します。

```
<!--- Create an instance of the Component Explorer utilities CFC.
        and get its metadata --->
<cfscript>
componentutils = createObject("component", "cfide.componentutils.utils");
utilmetadata = getMetaData(componentutils);
</cfscript>

<h4>Metadata for the CFC component utilities</h4>
<cfdump var="#utilmetadata#">

<!--- use GetMetadata to get the names and data types of the fields in the
            cfdocexamples Employees table --->
<cfquery name="getemployees" datasource="cfdocexamples">
SELECT *
FROM Employees
</cfquery>
<cfset employeemeta=getMetaData(getemployees)>

<h4>The Employees table has the following columns</h4>
<cfloop index="i" from="1" to="#arrayLen(employeemeta)#">
    <cfoutput>
        #employeemeta[i].name# #employeemeta[i].TypeName# #employeemeta[i].isCaseSensitive#<br>
    </cfoutput>
</cfloop>
```

GetMetricData

説明

サーバーのパフォーマンスの測定値を取得します。

戻り値

mode 値に基づいた測定値を含む ColdFusion 構造体

カテゴリ

システム関数

関数のシンタックス

```
GetMetricData(mode)
```

履歴

ColdFusion MX: cachepops パラメータが非推奨になりました。今後のリリースでは、動作せずにエラーとなる可能性があります。

パラメータ

パラメータ	オプション	説明
mode	perf_monitor	内部データを構造体で返します。 データを受け取るには、この関数を実行する前に、ColdFusion Administrator のパフォーマンスモニタを有効にしておきます。 Windows の場合、ColdFusion Administrator のパフォーマンスモニタを有効にしていなければ、このデータは Windows の PerfMonitor に表示されます。
	simple_load	サーバーの内部キューの状態から計算された整数値を返します。サーバーの全体的な負荷を示します。
	prev_req_time	サーバーが前のリクエスト処理に要した時間をミリ秒単位で返します。
	avg_req_time	サーバーがリクエスト処理に要した平均時間をミリ秒単位で返します。 この設定を 0 に変更すると、サーバーで平均の計算が行われなくなるため、データ収集に関連するオーバーヘッドがなくなります。 デフォルト値は 120 秒です。

使用方法

mode="perf_monitor" の場合、この関数は次のデータフィールドを持つ構造体を返します。

フィールド	説明
InstanceName	ColdFusion サーバー名です。デフォルト値は cfserver です。
PageHits	ColdFusion が起動してから受け取った HTTP リクエストの数です。
ReqQueued	ステージングキュー内で処理待ちの HTTP リクエストの数です。
DBHits	サーバーが起動してからのデータベースリクエストの数です。
ReqRunning	現在実行中の HTTP リクエストの数です。 ColdFusion Administrator で、同時に実行するリクエストの最大数を設定できます。
ReqTimedOut	ステージングキューにあるときまたは実行中に、タイムアウトになった HTTP リクエストの数です。
BytesIn	ColdFusion に対する HTTP リクエストのバイト数です。
BytesOut	ColdFusion からの HTTP レスポンスのバイト数です。
AvgQueueTime	最後の 2 つの HTTP リクエスト (現在およびその前のリクエスト) の、ステージングキューでの平均待ち時間です。
AvgReqTime	最後の 2 つの HTTP リクエスト (現在およびその前のリクエスト) について、サーバーでのそのリクエストの処理にかかった平均時間です。
AvgDBTime	最後の 2 つの HTTP リクエスト (現在および前のリクエスト) について、サーバーでのそのリクエスト内の CFQueries の処理にかかった平均時間です。
cachepops	このパラメータは非推奨となりました。ColdFusion はこの値を自動的に -1 に設定します。

例

```
<!--- This example gets and displays metric data from Windows NT PerfMonitor --->
<cfset pmData = GetMetricData( "PERF_MONITOR" ) >
<cfoutput>
    Current PerfMonitor data is: <p>
    InstanceName:#pmData.InstanceName# <p>
    PageHits:#pmData.PageHits# <p>
    ReqQueued: #pmData.ReqQueued# <p>
    DBHits: #pmData.DBHits# <p>
    ReqRunning: #pmData.ReqRunning# <p>
    ReqTimedOut: #pmData.ReqTimedOut# <p>
    BytesIn: #pmData.BytesIn# <p>
    BytesOut: #pmData.BytesOut# <p>
    AvgQueueTime: #pmData.AvgQueueTime# <p>
    AvgReqTime: #pmData.AvgReqTime# <p>
    AvgDBTime: #pmData.AvgDBTime# <p>
</cfoutput>
```

GetPageContext

説明

ページの属性と設定、およびリクエストオブジェクトとレスポンスオブジェクトへのアクセスを提供する、現在の ColdFusion の PageContext オブジェクトを取得します。

戻り値

現在の ColdFusion の Java PageContext Java オブジェクト

カテゴリ

システム関数

関数のシンタックス

```
GetPageContext()
```

履歴

ColdFusion MX: この関数が追加されました。

使用方法

ColdFusion の PageContext クラスは、Java PageContext オブジェクトのラッパークラスです。このオブジェクトはスコープを変換し、大文字小文字の区別なしで変数のルックアップを実行します。

PageContext オブジェクトは、J2EE 統合で有用なフィールドとメソッドを提供します。これには、対応する標準 JSP タグに相当する include と forward のメソッドが含まれます。これらのメソッドを使用して、JSP ページおよびサーブレットを呼び出すことができます。たとえば、CFScript で次のコードを使用して、JSP ページの hello.jsp を含め、それを name パラメータに渡します。

```
GetPageContext().include("hello.jsp?name=Bobby"); ===
```

GetPageContext を使用して、WebLogic の CFML ページに JSP ページを含める場合、JSP ページを呼び出す前に、cfflush で CFML ページの出力をフラッシュしなければならないことがあります。そうしない場合、ColdFusion の出力は JSP の出力後に表示されます。

この関数で返される PageContext に対してサポートされるメソッドは、JSP 仕様で必須に指定されているメソッドのみです。名前でスコープを検索するには、次のように StructGet 関数を使用します。

```
<cfset myscope = "server">
<cfset myserver = StructGet(myscope)>
```

詳細については、JSP (Java Server Pages) のドキュメントを参照してください。

WebLogic では、JSP ページを呼び出す前に、cfflush を使用して CFML ページの出力をフラッシュしなければならない場合があります。これを行わないと、ColdFusion の出力は JSP の出力後に表示されます。

JBOSS では、この関数を使用して JSP ファイルをインクルードすると、この関数の前にある CFML タグが処理されるより前に、JSP ファイルの内容が処理されます。

例

```
<!--- This example shows using the page context to set a page
    variable and access the language of the current locale. --->
<cfset pc = GetPageContext()>

<cfset pc.setAttribute("name","John Doe")>
<cfoutput>name: #variables.name#<br></cfoutput>

<cfoutput>Language of the current locale is
    #pc.getRequest().getLocale().getDisplayLanguage()#</cfoutput>.
```

GetPrinterInfo

説明

選択されたプリンタでサポートされている印刷属性を調べます。

戻り値

プリンタでサポートされている属性を含む構造体。プリンタが指定されていない場合は、デフォルトのプリンタでサポートされている属性が構造体に格納されます (デフォルトのプリンタが存在する場合)。

カテゴリ

システム関数

関数のシンタックス

```
GetPrinterInfo("printer")
```

関連項目

cfpdf、cfprint

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
printer	プリンタの名前です。Windows の場合、"¥¥s1001prn02¥NTN-2W-HP_BW02" のように指定できます。デフォルト値は、ColdFusion サーバーを実行しているアカウントのデフォルトプリンタです。プリンタ名の大文字と小文字は区別されるため、ColdFusion Administrator の [システム情報] ページに表示されるとおりに正確に入力する必要があります。 GetPrinterInfo("") のように空の文字列を指定すると、エラーが発生します。デフォルトのプリンタに関する情報を取得するには、GetPrinterInfo() というコードを使用します。

使用方法
この関数を cfprint タグと組み合わせて使用すると、大規模な印刷ジョブを処理できます。すべてのプリンタが、cfprint タグで使用可能なすべての印刷属性をサポートしているわけではありません。選択したプリンタでサポートされていない印刷属性を指定した場合、その属性は無視されます。

Windows システムでは、ColdFusion サーバーを実行するアカウントに対して、使用するプリンタごとに PRINTER_ACCESS_USE アクセス権を設定する必要があります。ColdFusion を実行するアカウントに適切なアクセス許可を割り当てないと、システム上でローカルに設定されているプリンタを利用できません。

attributeStruct の詳細については、540 ページの「cfprint」を参照してください。

例
```
<!--- The following code returns information on the default printer. --->
<cfdump var="#GetPrinterInfo()#">

<!--- The following code returns information on the specified printer. --->
<cfdump var="#GetPrinterInfo('\\S1001prn02\NTN-2W-SHARP01')#">
```

GetPrinterList

説明
ColdFusion サーバーからアクセスできるプリンタのリストを取得します。

戻り値
アクセス可能なプリンタのリストです。

カテゴリ
システム関数

関数のシンタックス
```
getPrinterList()
```

関連項目
cfpdf、cfprint

履歴
ColdFusion 8: この関数が追加されました。

使用方法
目的のすべてのプリンタにアクセスするのに適切な権限を持つアカウントを使用して ColdFusion サーバーが実行されていることを確認します。

GetProfileSections

説明
初期化ファイルのすべてのセクションを取得します。

初期化ファイルは、システムの起動時、オペレーティングシステムの起動時、またはアプリケーションの起動時に設定される、エントリとも呼ばれる環境変数に値を割り当てます。初期化ファイルには、boot.ini、Win32.ini のように、接尾辞 ini が付けられています。

戻り値
次のような形式を持つ構造体としての初期化ファイル

- 各初期化ファイルのセクション名が構造体のキーとなります。
- 初期化ファイルのセクションの各エントリリストが、構造体の値となります。

値がない場合は、空の文字列が返されます。

カテゴリ
システム関数

関数のシンタックス
```
GetProfileSections(iniFile)
```

関連項目
GetProfileString、SetProfileString

履歴
ColdFusion MX: この関数が追加されました。

パラメータ

パラメータ	説明
iniFile	C:¥boot.ini のような初期化ファイルの絶対パスです (ドライブ、ディレクトリ、ファイル名、および拡張子)。

GetProfileString

説明
初期化ファイルのエントリを取得します。

初期化ファイルは、システムの起動時、オペレーティングシステムの起動時、またはアプリケーションの起動時に設定される、エントリとも呼ばれる環境変数に値を割り当てます。初期化ファイルには、boot.ini、Win32.ini のように、接尾辞 ini が付けられています。

戻り値
初期化ファイルのエントリの文字列。値がない場合は、空の文字列が返されます。

カテゴリ
システム関数

関数のシンタックス
```
GetProfileString(iniPath, section, entry)
```

関連項目
GetProfileSections、SetProfileString

パラメータ

パラメータ	説明
iniPath	C:¥boot.ini のような初期化ファイルの絶対パスです (ドライブ、ディレクトリ、ファイル名、および拡張子)。
section	情報の抽出する初期化ファイルのセクションです。
entry	取得する値の名前です。

例

```
<h3>GetProfileString Example</h3>
Uses GetProfileString to get the value of timeout in an initialization file. Enter
    the full path of your initialization file, and submit the form.
<!--- If the form was submitted, it gets the initialization path and timeout value specified in the form --->
<cfif Isdefined("Form.Submit")>
    <cfset IniPath = FORM.iniPath>
    <cfset Section = "boot loader">
    <cfset timeout = GetProfileString(IniPath, Section, "timeout")>
    <h4>Boot Loader</h4>
    <!--- If no entry in an initialization file, nothing displays --->
    <p>Timeout is set to: <cfoutput>#timeout#</cfoutput>.</p>
</cfif>
<form action = "getprofilestring.cfm">
<table cellspacing = "2" cellpadding = "2" border = "0">
    <tr>
         <td>Full Path of Init File</td>
         <td><input type = "Text" name = "IniPath" value = "C:\myboot.ini">
        </td>
    </tr>
    <tr>
     <td><input type = "Submit" name = "Submit" value = "Submit"></td>
     <td></td>
    </tr></table>
</form>
```

GetReadableImageFormats

説明

ColdFusion がデプロイされているオペレーティングシステム上で ColdFusion が読み取ることができるイメージ形式のリストを返します。

戻り値

イメージファイル形式のリスト

カテゴリ

システム関数

履歴

ColdFusion 8: この関数が追加されました。

関数のシンタックス

```
GetReadableImageFormats()
```

関連項目
GetWriteableImageFormats、cfimage (サポートされるイメージファイル形式について)

使用方法
この関数は、ColdFusion サーバー上でのイメージファイルの互換性を調べる場合に使用します。

例
```
<cfoutput>#GetReadableImageFormats()#</cfoutput>
```

GetSOAPRequest

説明
SOAP リクエスト全体を含む XML オブジェクトを返します。通常は、Web サービス CFC 内から呼び出されます。

戻り値
SOAP リクエスト全体を含む XML オブジェクト

カテゴリ
XML 関数

履歴
ColdFusion MX 7: この関数が追加されました。

関数のシンタックス
```
GetSOAPRequest()
```

関連項目
AddSOAPRequestHeader、AddSOAPResponseHeader、GetSOAPRequestHeader、GetSOAPResponse、GetSOAPResponseHeader、IsSOAPRequest、『ColdFusion アプリケーションの開発』の Basic web service concepts

パラメータ

パラメータ	説明
webservice	オプション。cfobject タグまたは createobject 関数から返される webservice オブジェクトです。 この関数がクライアントから呼び出される場合は必須です。

使用方法
Web サービスの起動後、この関数を呼び出して、WEB サービスリクエストオブジェクトを取得します。webservice パラメータがない外部の Web サービス CFC からこの関数を呼び出した場合、次の式エラーが返されます。

```
Unable to use getSOAPRequest: not processing a web service request.
```

Web サービス CFC 内からこの関数を呼び出した場合は、webservice 引数を省略できます。この関数は、現在処理しているリクエストに対して機能します。

CFML XML 関数を使用して、返された XML オブジェクトを調べることができます。

例

この例では、リクエストを行って、headerservice.cfc Web サービスの echo_me 関数を実行します。headerservice.cfc Web サービスの実装に関する詳細、echo_me 関数、および Web サービス CFC の内容については、AddSOAPResponseHeader 関数または GetSOAPRequestHeader 関数の例を参照してください。

```
<!--- Note that you might need to modify the URL in the CreateObject function
to match your server and the location of the headerservice.cfc file if it is
different than shown here.
Note, too, that getSOAPRequest is called from the client here, whereas often it
would be called from within the web service CFC. --->

<cfscript>
    ws = CreateObject("webservice",
"http://localhost/soapheaders/headerservice.cfc?WSDL");
    ws.echo_me("hello world");
    req = getSOAPRequest(ws);
</cfscript>
<cfdump var="#req#">
```

GetSOAPRequestHeader

説明

SOAP リクエストヘッダを取得します。リクエストを SOAP サービスとして処理する CFC Web サービス関数内からのみ呼び出します。

戻り値

SOAP リクエストヘッダ

カテゴリ

XML 関数

履歴

ColdFusion MX 7: この関数が追加されました。

関数のシンタックス

```
GetSOAPRequestHeader(namespace, name [, asXML])
```

関連項目

AddSOAPRequestHeader、AddSOAPResponseHeader、GetSOAPRequest、GetSOAPResponse、GetSOAPResponseHeader、IsSOAPRequest、『ColdFusion アプリケーションの開発』の Basic web service concepts

パラメータ

パラメータ	説明
namespace	ヘッダの名前空間を含む文字列です。
name	ヘッダの名前を含む文字列です。
asXML	true の場合、ヘッダは CFML XML オブジェクトとして返されます。false (デフォルト) の場合、ヘッダは Java オブジェクトとして返されます。

使用方法

asXML パラメータに対して false を指定した場合、ColdFusion は、ヘッダの xsi:type 属性で指定されたデータ型を使用してヘッダを最初に取得しようとします。xsi:type 属性を使用できない場合、ColdFusion はヘッダを文字列として取得しようとします。asXML パラメータに対して true を指定した場合、ColdFusion はヘッダを未処理の XML として取得します。

この関数は、Web サービスリクエストではないコンテキストで呼び出された場合、エラーを返します。IsSOAPRequest 関数を使用して、CFC が Web サービスとして稼動しているかどうかを判別します。

例

この例では、GetSOAPRequestHeader 関数のオペレーションを示す CFC Web サービスを作成するとともに、他の ColdFusion SOAP 関数のオペレーションを示す Web サービスも提供します。

次のコードを、Web ルート下の "soapheaders" というフォルダに "headerservice.cfc" として保存します。この Web サービスを起動する例を実行することにより、そのオペレーション、特に GetSOAPRequestHeader 関数のオペレーションをテストします。たとえば、AddSOAPRequestHeader の例を参照してください。

```
<h3>GetSOAPRequestHeader Example</h3>
<cfcomponent displayName="tester" hint="Test for SOAP headers">

<cffunction name="echo_me"
    access="remote"
    output="false"
    returntype="string"
    displayname="Echo Test" hint="Header test">

<cfargument name="in_here" required="true" type="string">

<cfset isSOAP = isSOAPRequest()>
<cfif isSOAP>

    <!--- Get the first header as a string and as XML --->
    <cfset username = getSOAPRequestHeader("http://mynamespace/", "username")>
    <cfset return = "The service saw username: " & username>
    <cfset xmlusername = getSOAPRequestHeader("http://mynamespace/", "username", "TRUE")>
    <cfset return = return & "<br> as XML: " & xmlusername>

    <!--- Get the second header as a string and as XML --->
    <cfset password = getSOAPRequestHeader("http://mynamespace/", "password")>
    <cfset return = return & "The service saw password: " & password>
    <cfset xmlpassword = getSOAPRequestHeader("http://mynamespace/", "password", "TRUE")>
    <cfset return = return & "<br> as XML: " & xmlpassword>

    <!--- Add a header as a string --->
<cfset addSOAPResponseHeader("http://www.tomj.org/myns", "returnheader", "AUTHORIZED VALUE", false)>
```

```
    <!--- Add a second header using a CFML XML value --->
    <cfset doc = XmlNew()>
    <cfset x = XmlElemNew(doc, "http://www.tomj.org/myns", "returnheader2")>
    <cfset x.XmlText = "hey man, here I am in XML">
    <cfsetx.XmlAttributes["xsi:type"] = "xsd:string">
    <cfset tmp = addSOAPResponseHeader("ignoredNameSpace", "ignoredName", x)>

<cfelse>
    <!--- Add a header as a string - Must generate error!
<cfset addSOAPResponseHeader("http://www.tomj.org/myns", "returnheader", "AUTHORIZED VALUE", false)>
    --->
<cfset return = "Not invoked as a web service">
</cfif>

<cfreturn return>

</cffunction>

</cfcomponent>
```

GetSOAPResponse

説明
Web サービスの起動後に、SOAP レスポンス全体を含む XML オブジェクトを返します。

戻り値
SOAP レスポンス全体を含む XML オブジェクト

カテゴリ
XML 関数

履歴
ColdFusion MX 7: この関数が追加されました。

関数のシンタックス
```
GetSOAPResponse(webservice)
```

関連項目
AddSOAPRequestHeader、AddSOAPResponseHeader、GetSOAPRequest、GetSOAPRequestHeader、GetSOAPResponseHeader、IsSOAPRequest、『ColdFusion アプリケーションの開発』の Basic web service concepts

パラメータ

パラメータ	説明
webservice	cfobject タグまたは createobject 関数から返される webservice オブジェクトです。

使用方法
レスポンスを取得しようとする前に、Web サービスを起動します。CFML XML 関数を使用して、XML レスポンスを調べることができます。

例

この例では、リクエストを行って、headerservice.cfc Web サービスの echo_me 関数を実行します。リクエストの後、この例では GetSOAPResponse 関数を呼び出して SOAP レスポンスを取得します。次に、cfdump を呼び出してその内容を表示します。

headerservice.cfc Web サービスの実装に関する詳細、echo_me 関数、および Web サービス CFC の内容については、AddSOAPResponseHeader 関数または GetSOAPRequestHeader 関数の例を参照してください。

```
<!--- Note that you might need to modify the URL in the CreateObject function
to match your server and the location of the headerservice.cfc file if it is
different than shown here. --->

<cfscript>
    ws = CreateObject("webservice",
"http://localhost/soapheaders/headerservice.cfc?WSDL");
    ws.echo_me("hello world");
    resp = getSOAPResponse(ws);
</cfscript>
<cfdump var="#resp#">
```

GetSOAPResponseHeader

説明

SOAP レスポンスヘッダを返します。Web サービスリクエストの実行後に、Web サービスを呼び出しているコード内からこの関数を呼び出します。

戻り値

SOAP レスポンスヘッダ

カテゴリ

XML 関数

履歴

ColdFusion MX 7: この関数が追加されました。

関数のシンタックス

```
GetSOAPResponseHeader(webservice, namespace, name [, asXML])
```

関連項目

AddSOAPRequestHeader、AddSOAPResponseHeader、GetSOAPRequest、GetSOAPRequestHeader、GetSOAPResponse、IsSOAPRequest、『ColdFusion アプリケーションの開発』の Basic web service concepts

パラメータ

パラメータ	説明
webservice	cfobject タグまたは createobject 関数から返される webservice オブジェクトです。
namespace	ヘッダの名前空間を含む文字列です。
name	SOAP ヘッダの名前を含む文字列です。
asXML	true の場合、ヘッダは CFML XML オブジェクトとして返されます。false (デフォルト) の場合、ヘッダは Java オブジェクトとして返されます。

使用方法

asXML パラメータに対して false を指定した場合、ColdFusion は、ヘッダの xsi:type 属性で指定されたデータ型を使用してヘッダを最初に取得しようとします。xsi:type 属性を使用できない場合、ColdFusion はヘッダを文字列として取得しようとします。asXML パラメータに対して true を指定した場合、ColdFusion はヘッダを未処理の XML として取得します。

cfinvoke を使用して Web サービスを呼び出した後に、Web サービスの利用者が CFML コード内で使用します。

例

この例は 2 つの部分で構成されています。最初の部分は Web サービス CFC です。この関数およびその他の ColdFusion SOAP 関数は、この Web サービス CFC を使用して、Web サービスとのやり取りを示します。この関数に対して Web サービスを実装するには、AddSOAPResponseHeader 関数または GetSOAPRequestHeader 関数の例を参照してください。

次の例を実行して、GetSOAPResponseHeader 関数の機能を確認してください。

```
<!--- Note that you might need to modify the URL in the CreateObject function
to match your server and the location of the headerservice.cfc file if it is
different than shown here. Likewise for the cfinvoke tag at the end --->

<h3>GetSOAPResponseHeader Example</h3>
<cfscript>
    // Create the web service object
    ws = CreateObject("webservice", "http://localhost/soapheaders/headerservice.cfc?WSDL");

    // Set the username header as a string
    addSOAPRequestHeader(ws, "http://mynamespace/", "username", "tom", false);

    // Set the password header as a CFML XML object
    doc = XmlNew();
    doc.password = XmlElemNew(doc, "http://mynamespace/", "password");
    doc.password.XmlText = "My Voice is my Password";
    doc.password.XmlAttributes["xsi:type"] = "xsd:string";
    addSOAPRequestHeader(ws, "ignoredNameSpace", "ignoredName", doc);

    // Invoke the web service operation
    ret = ws.echo_me("argument");

    // Get the first header as an object (string) and as XML
    header = getSOAPResponseHeader(ws, "http://www.tomj.org/myns", "returnheader");
    XMLheader = getSOAPResponseHeader(ws, "http://www.tomj.org/myns", "returnheader", true);

    // Get the second header as an object (string) and as XML
    header2 =getSOAPResponseHeader(ws, "http://www.tomj.org/myns", "returnheader2");
    XMLheader2 = getSOAPResponseHeader(ws, "http://www.tomj.org/myns", "returnheader2", true);
</cfscript>
<hr>
<cfoutput>
Soap Header value: #HTMLCodeFormat(header)#<br>
Soap Header XML value: #HTMLCodeFormat(XMLheader)#<br>
Soap Header 2 value: #HTMLCodeFormat(header2)#<br>
Soap Header 2 XML value: #HTMLCodeFormat(XMLheader2)#<br>
Return value: #HTMLCodeFormat(ret)#<br>
</cfoutput>
<hr>

<cfinvoke component="soapheaders.headerservice" method="echo_me" returnvariable="ret" in_here="hi">
</cfinvoke>
<cfoutput>Cfinvoke returned: #ret#</cfoutput>
```

GetSystemFreeMemory

説明
メモリの空き容量の詳細を取得します。

戻り値
現在使用可能なメモリ（バイト単位）

シンタックス
getSystemFreeMemory()

GetSystemTotalMemory

説明
オペレーティングシステムが使用可能なメモリの詳細（バイト単位）を取得します。

戻り値
使用可能なメモリ（バイト単位）

シンタックス
getSystemTotalMemory()

GetTempDirectory

説明
ColdFusion がテンポラリファイルに使用するディレクトリのパスを取得します。ディレクトリは、ColdFusion が実行されているアカウントやその他の要素によって異なります。アプリケーション内でこの関数を使用する前に、アカウントから返されるディレクトリを確認してください。

戻り値
最後に円記号 (¥) が付いた、ディレクトリの絶対パスの文字列

カテゴリ
システム関数

関数のシンタックス
```
GetTempDirectory()
```

関連項目
GetTempFile

履歴
ColdFusion MX: 動作が変更されました。Windows では、この関数は、組み込み型 Java アプリケーションサーバーのテンポラリディレクトリを返すようになりました。Windows 以外のプラットフォームでは、そのオペレーティングシステムのテンポラリディレクトリを返します。

例

```
<h3>GetTempDirectory Example</h3>

<p>The temporary directory for this ColdFusion server is
    <cfoutput>#GetTempDirectory()#</cfoutput>.</p>
<p>We have created a temporary file called:
<cfoutput>#GetTempFile(GetTempDirectory(),"testFile")#</cfoutput></p>
```

GetTempFile

説明

prefix の最初の 3 文字までを名前の先頭に使用して、ディレクトリ内にテンポラリファイルを作成します。

戻り値

テンポラリファイル名の文字列

カテゴリ

システム関数

関数のシンタックス

```
GetTempFile(dir, prefix)
```

関連項目

GetTempDirectory

パラメータ

パラメータ	説明
dir	ディレクトリ名です。
prefix	dir ディレクトリ内に作成するテンポラリファイルの接頭辞です。

例

```
<h3>GetTempFile Example</h3>
<p>The temporary directory for this ColdFusion Server is
    <cfoutput>#GetTempDirectory()#</cfoutput>.</p>
<p>We have created a temporary file called:
<cfoutput>#GetTempFile(GetTempDirectory(),"testFile")#</cfoutput></p>
```

GetTemplatePath

説明

この関数は非推奨となっています。代わりに GetBaseTemplatePath 関数を使用してください。

アプリケーションのベースページの絶対パスを取得します。

履歴

ColdFusion MX: この関数が非推奨になりました。今後のリリースでは、動作せずにエラーとなる可能性があります。

GetTickCount

説明
ミリ秒単位の内部タイマーの現在の値を返します。

戻り値
システム時間をミリ秒単位で表す文字列。

カテゴリ
日付および時刻関数、システム関数

関数のシンタックス
```
GetTickCount()
```

使用方法
この関数は、CFML コードのセグメントまたは他のページ処理要素のタイミングを取る場合に役立ちます。カウンタの 1 つの値そのものには意味はありません。タイミング値として有用な値を得るには、GetTickCount を 2 回呼び出した結果の差を求めます。

例
```
<!--- Setup timing test --->
<cfset iterationCount = 1000>
<!--- Time an empty loop with this many iterations --->
<cfset tickBegin = GetTickCount()>
<cfloop Index = i From = 1 To = #iterationCount#></cfloop>
<cfset tickEnd = GetTickCount()>
<cfset loopTime = tickEnd - tickBegin>

<!--- Report --->
<cfoutput>Loop time (#iterationCount# iterations) was: #loopTime#
     milliseconds</cfoutput>
```

GetTimeZoneInfo

説明
関数を呼び出したコンピュータの現地タイムゾーン情報を取得します。ここでは世界標準時 (UTC: Universal Time Coordinated) からの相対情報が示されます。UTC は、英国グリニッチの経線の基準太陽時で、全世界の標準時計算の基準として使用されています。

ColdFusion では、日付時刻値は標準時刻線上の実数値を持つ日付時刻オブジェクトとして保管されます。オブジェクトはその形式が設定されるとタイムゾーンに変換されます。タイムゾーンからの日付時刻値が解析されると、その値は実数値に変換されます。

戻り値
次の要素およびキーを含む構造体

- utcTotalOffset: UTC から現地時刻のオフセットです (単位 : 秒)。
  - プラス記号 (+) は、タイムゾーンが UTC の西側にあることを示します (北アメリカなど)。
  - マイナス (-) 記号は、タイムゾーンが UTC の東側にあることを示します (ドイツなど)。
- utcHourOffset: UTC から現地時刻のオフセットです (単位 : 時間)。

- utcMinuteOffset: 時単位のオフセットを考慮した後の分単位のオフセットです。北米の場合、この値は 0 です。時単位のオフセット上に位置していない国の場合、この値は 0 ～ 60 になります。たとえば、オーストラリアのアデレードの標準時は、UTC から 9 時間 30 分の時差があります。
- isDSTOn: ホストコンピュータで夏時間 (DST: Daylight Savings Time) が設定されている場合は true、設定されていない場合は false となります。

カテゴリ
日付および時刻関数、各国語対応関数

関数のシンタックス
```
GetTimeZoneInfo()
```

関連項目
DateConvert、CreateDateTime、DatePart

例
```
<h3>GetTimeZoneInfo Example</h3>
<!--- This example shows the use of GetTimeZoneInfo --->
<cfoutput>
The local date and time are #now()#.
</cfoutput>

<cfset info = GetTimeZoneInfo()>
<cfoutput>
<p>Total offset in seconds is #info.utcTotalOffset#.</p>
<p>Offset in hours is #info.utcHourOffset#.</p>
<p>Offset in minutes minus the offset in hours is
    #info.utcMinuteOffset#.</p>
<p>Is Daylight Savings Time in effect? #info.isDSTOn#.</p>
</cfoutput>
```

GetToken

説明
delimiters パラメータ内のリストのトークンが文字列内に存在するかどうかを調べます。

戻り値
文字列の **index** 位置で検出されたトークンの文字列。**index** の値が文字列内のトークンの数よりも大きい場合は空の文字列が返されます。

カテゴリ
文字列関数

関数のシンタックス
```
GetToken(string, index  [, delimiters ])
```

関連項目
Left、Right、Mid、SpanExcluding、SpanIncluding

パラメータ

パラメータ	説明
string	文字列、または文字列を含んでいる変数です。検索対象の文字列を指定します。
index	正の整数またはそれを含んでいる変数です。トークンの位置を表します。
delimiters	文字列、または文字列を含んでいる変数です。区切り文字の区切りリストです。要素は複数の文字で構成することができます。 区切り文字のデフォルトのリストは、空白文字、タブ文字、改行文字、またはそれらのコード "chr(32)"、"chr(9)"、"chr(10)" です。 リストのデフォルトの区切り文字は、カンマです。

使用方法

次の例は、この関数の動作を示しています。

例 1

次の例では、関数呼び出しで区切り文字 ":;" を使用して、文字列から 2 番目の要素をリクエストしています。

```
GetToken("red,blue:;red,black,tan:;red,pink,brown:;red,three", 2, ":;")
```

出力は次のようになります。

```
red,black,tan
```

例 2

```
<cfset mystring = "four,"
    & #chr(32)# & #chr(9)# & #chr(10)#
    & ",five, nine,zero:;"
    & #chr(10)#
    & "nine,ten:, eleven:;twelve:;thirteen,"
    & #chr(32)# & #chr(9)# & #chr(10)#
    & ",four">
<cfoutput>
    #HTMLCodeFormat(mystring)#<br><br>
</cfoutput>
```

出力は次のようになります。

```
four,
,five, nine,zero:;
nine,ten:, eleven:;twelve:;thirteen,
,four
```

GetToken 関数は、明示的な空白文字、タブ文字、または改行文字をパラメータの区切り文字として認識します (空白文字を指定するコードは chr(32)、タブ文字は chr(9)、改行文字は chr(10) です)。

この例での文字列 mystring には、次の文字があります。

- 部分文字列の "four," と ",five" の間に強制スペース
- "five," と "nine" の間にリテラルスペース
- "ten:," と "eleven," の間にリテラルスペース
- "thirteen," と ",four" の間に強制スペース

mystring に対する次の呼び出しでは、delimiters に空白は指定されていません (省略されています)。したがって、この関数では空白文字は string の区切り文字として使用されます。

```
<br>
<cfoutput>
    GetToken(mystring, 3) is : #GetToken(mystring, 3)#
</cfoutput><br>
```

このコードの出力は次のようになります。

```
GetToken(mystring, 3) is : nine,zero:;
```

ここでは関数によって 3 番目の区切り文字が検出され、その直前の、2 番目と 3 番目の区切り文字の間の部分文字列が返されます。この部分文字列は "nine,zero:;" です。

例 3

```
<cfset mystring2 = "four,"
    &#chr(9)# & #chr(10)#
    & ",five,nine,zero:;"
    & #chr(10)#
    & "nine,ten:,eleven:;twelve:;thirteen,"
    & #chr(9)# & #chr(10)# & ",four">
<cfoutput>
    #HTMLCodeFormat(mystring2)#<br>
</cfoutput>
```

出力は次のようになります。

```
four,
,five,nine,zero:;
nine,ten:,eleven:;twelve:;thirteen,
,four
```

次は mystring2 に対する呼び出しです。

```
<cfoutput>
    GetToken(mystring2, 2) is : #GetToken(mystring2, 2)#
</cfoutput>
```

出力は次のようになります。

```
GetToken(mystring2, 2) is : ,five,nine,zero:;
```

ここでは関数によって 2 番目の区切り文字が検出され、その直前の、最初の区切り文字と 2 番目の区切り文字の間の部分文字列が返されます。この部分文字列は ",five,nine,zero:;" です。

例

```
<h3>GetToken Example</h3>
<cfif IsDefined("FORM.yourString")>
<!--- set delimiter --->
<cfif FORM.yourDelimiter is not "">
        <cfset yourDelimiter = FORM.yourDelimiter>
    <cfelse>
        <cfset yourDelimiter = " ">
</cfif>
<!--- check whether number of elements in list is greater than or
    equal to the element sought to return --->
<cfif ListLen(FORM.yourString, yourDelimiter) GTE FORM.returnElement>
    <cfoutput>
    <p>Element #FORM.ReturnElement# in #FORM.yourString#,
    delimited by "#yourDelimiter#"
    <br>is:#GetToken(FORM.yourString, FORM.returnElement, yourDelimiter)#
    </cfoutput>
...
```

GetTotalSpace

説明
アプリケーションが使用可能なハードディスクまたはメモリ内の合計容量を返します。

戻り値
合計容量（バイト単位）

カテゴリ
システム関数

関連項目
getVFSMetadata

シンタックス
getTotalSpace(path)

パラメータ

パラメータ	説明
path	次のパス • ハードディスクドライブ（例：C:（Windows の場合）、/（UNIX の場合））。Windows の場合は、C だけでなく、コロン（:）を明示的に使用する必要があることに注意してください。ハードディスクドライブ全体の容量のみが返されます。パス内にディレクトリを指定した場合（例：/opt/）、ディレクトリは無視されて / のみが考慮されます。 • メモリ内ファイルシステムの場合のパスは ram: です。

使用方法
アプリケーションのメモリは、次の RAM の設定によって決まります。

- ColdFusion Administrator で、「メモリ内仮想ファイルシステムのアプリケーションごとのメモリ制限」（サーバーの設定／設定）に指定された値

  デフォルト値は 20 MB です。

- Application.cfc の this.inmemoryfilesystem.size で指定された値

両方の場所で値を設定した場合は、小さい方の値が考慮されます。

注意：Application.cfc で指定した値のほうが小さい場合も、ColdFusion Administrator で、サーバーの設定／設定／メモリ内ファイルシステムの有効化を確認してください。

例
GetFreeSpace の例を参照してください。

GetUserRoles

説明
現在のユーザーのロールのリストを取得します。この関数は、サーブレット API に対して設定されたロールセットは返さず、ColdFusion ロールのみを返します。

戻り値
現在のユーザーのロールのリスト

カテゴリ
セキュリティ関数

関数のシンタックス
```
GetUserRoles()
```

関連項目
cflogin、cfloginuser、cflogout、GetAuthUser、IsUserInAnyRole、IsUserInRole、IsUserLoggedIn、『ColdFusion アプリケーションの開発』の Securing Applications

履歴
ColdFusion 8: この関数が追加されました。

例
```
<cfloginuser name = "#cflogin.name#" password = "#cflogin.password#"
    roles = "#GetUserRoles()#" />
```

GetVFSMetaData

説明
メモリ内の仮想ファイルシステムのメタデータを取得します。

戻り値
メモリ内の仮想ファイルシステムに関する情報を含む構造体。

カテゴリ
システム関数

関数のシンタックス
getVFSMetaData (fileSystemType)

関連項目
『ColdFusion アプリケーションの開発』の「メモリ内ファイルの使用」

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
fileSystemType	メモリ内ファイルシステムのタイプです。ColdFusion は RAM のみをサポートしています。

使用方法
この関数は、次のキーを含む構造体を返します。

パラメータ	説明
Enabled	メモリ内仮想ファイルシステムのサポートが有効かどうかを示します。メモリ内仮想ファイルシステムのサポートが無効になっている場合は、このキーが構造体に含まれる唯一のキーになります。
Limt	メモリ内仮想ファイルシステムのメモリ制限 (バイト単位) です。
Used	指定されたメモリ制限のうち、現在使用されているメモリ量 (バイト単位) です。
Free	空きメモリ量 (バイト単位) です。

例

```
<cfset myroot = hash(getDirectoryFromPath(getCurrentTemplatePath())) >
<cfset name = "ram:///"&myroot&"/">
<cffile action="append" file="#name#" output="created at #now()#">
<cfset contents = fileRead(name)>
<cfdump var="#contents#">
<cfdump var="#getVFSMetaData("ram")#">
```

この例は、メモリ内仮想ファイルシステムが有効になっている場合にのみ機能します。

GetWriteableImageFormats

説明

ColdFusion がデプロイされているオペレーティングシステム上で ColdFusion が書き込むことができるイメージ形式のリストを返します。

戻り値

イメージファイル形式のリスト

カテゴリ

システム関数

履歴

ColdFusion 8: この関数が追加されました。

関数のシンタックス

```
GetWriteableImageFormats()
```

関連項目

GetReadableImageFormats、cfimage (サポートされるイメージファイル形式について)

使用方法

この関数は、ColdFusion サーバー上でのイメージファイルの互換性を調べる場合に使用します。

例

```
<cfoutput>#GetWriteableImageFormats()#</cfoutput>
```

関数 h ～ im

Hash

説明
可変長の文字列を、"fingerprint" として機能可能な固定長の文字列、または元の文字列に固有の識別子に変換します。ハッシュの結果をソースの文字列に逆変換することはできません。

戻り値
文字列です。

カテゴリ
変換関数、セキュリティ関数、文字列関数

関数のシンタックス
```
Hash(string [, algorithm [, encoding ]])
```

履歴
ColdFusion MX 7: **algorithm** パラメータと **encoding** パラメータが追加されました。

パラメータ

パラメータ	説明
string	ハッシュ対象の文字列です。
algorithm	(オプション) 文字列をハッシュするために使用するアルゴリズムです。ColdFusion では、次のアルゴリズムを使用する暗号ライブラリがインストールされます。 • CFMX_COMPAT : ColdFusion MX および ColdFusion MX 6.1 で生成されるものと同じハッシュ文字列を生成します (デフォルト)。これは単なるプレースホルダーのアルゴリズムで、ユーザーが algorithm に指定するオプションを持っていない場合に、CFMX と互換性があるアルゴリズムを使用することを ColdFusion に通知するものです。 • MD5 : (デフォルト) MD5 アルゴリズムを使用して、32 文字の 16 進数文字列を生成します。これは ColdFusion MX およびそれ以前のリリースで使用されるアルゴリズムです。 • SHA : NIST (Nation Institute of Standards and Technology : 米国標準技術局) FIPS-180-2 で定義された Secure Hash Standard の SHA-1 アルゴリズムを使用して、40 文字の文字列を生成します。 • SHA-256 : FIPS-180-2 で定義された SHA-256 アルゴリズムを使用して、44 文字の文字列を生成します。 • SHA-384 : FIPS-180-2 で定義された SHA-384 アルゴリズムを使用して、64 文字の文字列を生成します。 • SHA-512 : FIPS-180-2 で定義された SHA-1 アルゴリズムを使用して、128 文字の文字列を生成します。
	エンタープライズ版の ColdFusion では、RSA BSafe Crypto-J ライブラリがインストールされます。このライブラリを使用すると、FIPS-140 に準拠した強力な暗号化を利用できます。このライブラリでは、次のアルゴリズムを使用できます。 • MD2: RFC 1319 で定義された MD2 ハッシュアルゴリズムです。 • MD5 : RFC 1321 で定義されたアルゴリズムです。 • RIPEMD160: RACE Integrity Primitives Evaluation Message Digest 160 (ビット) メッセージダイジェストアルゴリズムおよび暗号化ハッシュ機能です。 • SHA-1: FIPS 180-2 および FIPS 198 で定義された 160 ビットセキュアハッシュアルゴリズムです。 • SHA-224: FIPS 180-2 および FIPS 198 で定義された 224 ビットセキュアハッシュアルゴリズムです。 • SHA-256: FIPS 180-2 および FIPS 198 で定義された 256 ビットセキュアハッシュアルゴリズムです。 • SHA-384: FIPS 180-2 および FIPS 198 で定義された 384 ビットセキュアハッシュアルゴリズムです。 • SHA-512: FIPS 180-2 および FIPS 198 で定義された 512 ビットセキュアハッシュアルゴリズムです。 これら以外の暗号アルゴリズムを使用するセキュリティプロバイダをインストールした場合は、そのハッシュアルゴリズムを指定することもできます。
encoding	(オプション) この属性を使用する場合は、**algorithm** パラメータも指定します。これは、文字列をハッシュアルゴリズムで使用されるバイトデータに変換するためのエンコードを指定する文字列です。Java ランタイムで認識される文字エンコード名でなければなりません。デフォルト値は、"neo-runtime.xml" ファイルの defaultCharset エントリで指定された値で、通常は UTF-8 です。この属性は、CFMX_COMPAT アルゴリズムを使用する場合は無視されます。

使用方法

この関数の結果は比較や検証を行うときに役立ちます。たとえば、パスワードを露出せずにパスワードのハッシュをデータベースに保管することができます。また、入力されたパスワードをハッシュし、その結果をデータベース内のハッシュされたパスワードと比較して、パスワードの有効性を確認できます。

ColdFusion では、JCE (Java Cryptography Extension) を使用し、Sun JCE デフォルトセキュリティプロバイダを含む Sun Java ランタイムがインストールされます。このプロバイダには、前の「パラメータ」に示したアルゴリズムが含まれています。JCE フレームワークには、他のプロバイダの実装を使用するための機能も含まれています。ただし、Adobe ではサードパーティのセキュリティプロバイダに対するテクニカルサポートは提供していません。

encoding 属性は、通常は必須ではありません。この属性は、デフォルトのエンコードが異なるシステムで同じハッシュ値を生成するためのメカニズムを提供します。"neo-runtime.xml" ファイルの defaultCharset エントリを変更しない限り、ColdFusion では UTF-8 がデフォルトのエンコードとして使用されます。

例

次の例では、パスワードを入力し、そのハッシュされたパスワードを、cfdocexamples データベースの SecureData テーブルに保存されているハッシュ値と比較します。このテーブルには、次のエントリがあります。

ユーザー ID	パスワード
blaw	blaw
dknob	dknob

```
<h3>Hash Example</h3>

<!--- Do the following if the form is submitted. --->
<cfif IsDefined("Form.UserID")>

    <!--- query the data base. --->
    <cfquery name = "CheckPerson" datasource = "cfdocexamples">
        SELECT PasswordHash
        FROM SecureData
        WHERE UserID = <cfqueryparam value = "#Form.userID#"
            cfsqltype = 'CF_SQL_VARCHAR'>
    </cfquery>

    <!--- Compare query PasswordHash field and the hashed form password
            and display the results. --->
    <cfoutput>
        <cfif Hash(Form.password, "SHA") is not checkperson.passwordHash>
            User ID #Form.userID# or password is not valid. Try again.
        <cfelse>
            Password is valid for User ID #Form.userID#.
        </cfif>
    </cfoutput>
</cfif>

<!--- Form for entering ID and password. --->
<form action="#CGI.SCRIPT_NAME#" method="post">
    <b>User ID: </b>
    <input type = "text" name="UserID" ><br>
    <b>Password: </b>
    <input type = "text" name="password" ><br><br>
    <input type = "Submit" value = "Encrypt my String">
</form>
```

HMac

説明

指定した文字列のハッシュベースメッセージ認証コードをアルゴリズムおよびエンコード方式に基づいて作成します。ハッシュベースメッセージ認証コード（HMAC）は、送信されたメッセージのデータの整合性と信憑性を検証するために使用されます。これには、暗号化ハッシュ関数と秘密鍵の組み合わせが必要です。暗号ハッシュ関数には、Message Digest 5（MD5）、Secure Hash Algorithm（SHA）などを使用できます。

戻り値
ブール値です。正常に終了した場合は true。

カテゴリ
表示および書式制御関数

シンタックス
HMac(message, key [,algorithm] [,encoding])

関連項目
SessionInvalidate、SessionRotate

履歴
ColdFusion 10: この関数が追加されました。

パラメータ

パラメータ	必須／オプション	説明
message	必須	送信するメッセージです。メッセージは文字列またはバイト配列です。
key	必須	HMAC を作成するための秘密鍵です。鍵は文字列またはバイト配列です。
algorithm	オプション	使用するアルゴリズムです。
encoding	オプション	使用するエンコードです。

使用方法
この関数は、指定した文字列のハッシュベースメッセージ認証コードをアルゴリズムおよびエンコード方式に基づいて作成するために使用します。

例

```
<h2>HMAC Test</h2>
<cfset x=hmac("Hi There","key1","HMACRIPEMD160")>
<cfoutput>#x#</cfoutput>
```

HQL メソッド

Hibernate Query Language (HQL) メソッドから返される値またはエンティティの配列は、HQL クエリーから返される内容に応じて、1 次元配列または多次元配列になります。指定したフィルタ条件に一致するレコードが 1 件だけであることがわかっている場合は、unique=true を指定すると、配列ではなく単体のエンティティが返されます。

unique=true の場合に複数のレコードが返されると、例外がスローされます。

注意：HQL で使用する entityname と properties では、大文字と小文字が区別されます。

使用可能な HQL メソッドは、次のとおりです。

- ORMExecuteQuery(hql, [,unique] [, queryoptions])
- ORMExecuteQuery(hql, params [,unique] [,queryOptions])
- ORMExecuteQuery(hql, namedparams [, unique] [, queryOptions])

ORMExecuteQuery(hql, [,unique] [, queryoptions])

説明

アプリケーションレベルで指定されているデフォルトのデータソースに対して HQL を実行します。queryOptions を使用すると、取得の動作を制御する次のオプションを指定できます。

- ignorecase: true に設定すると、ソート順の大文字と小文字が無視されます。このオプションは、sortorder パラメータを指定した場合にのみ使用します。
- maxResults: 取得されるオブジェクトの最大数を指定します。
- offset: 取得の開始場所となる、結果セットの開始索引を指定します。
- cacheable: このクエリーの結果がセカンダリキャッシュにキャッシュされるかどうかを指定します。デフォルトは false です。
- cachename: セカンダリキャッシュ内のキャッシュの名前です。
- timeout: クエリーのタイムアウト値 (秒単位) を指定します。

注意： maxresults および timeout はページ割りに使用されます。

カテゴリ

ORM 関数

例

```
<cfset employees = ORMExecuteQuery("from Employees")>
<cfset employees = ORMExecuteQuery("from Employees where age > 40")>
<cfset employeeObj = ORMExecuteQuery("from Employees where EmployeeID =1", true)>
<cfset firstNameArray = ORMExecuteQuery("select FirstName from Employees")>
<cfset numberOfEmps = ORMExecuteQuery("select count(*) from Employees")>
<cfset firstName = ORMExecuteQuery("select FirstName from Employees where EmployeeID = 1", true)>
<cfset employees = ORMExecuteQuery("from Employees", false, {offset=5, maxresults=10, timeout=5})>
```

ORMExecuteQuery(hql, params [,unique] [,queryOptions])

説明

このタイプの ORMExecuteQuery を使用すると、クエリーにパラメータを渡すことができます。'?' (疑問符) をパラメータのプレースホルダーとして使用します。パラメータの値は、配列として params に渡す必要があります。

また、queryOptions を使用して、取得動作を制御するための複数のオプションを指定できます。

- ignorecase: true に設定すると、ソート順の大文字と小文字が無視されます。このオプションは、sortorder パラメータを指定した場合にのみ使用します。
- maxResults: 取得されるオブジェクトの最大数を指定します。
- offset: 取得の開始場所となる、結果セットの開始索引を指定します。
- cacheable: このクエリーの結果がセカンダリキャッシュにキャッシュされるかどうかを指定します。デフォルトは false です。
- cachename: セカンダリキャッシュ内のキャッシュの名前です。
- timeout: クエリーのタイムアウト値 (秒単位) を指定します。

注意： maxresults および timeout はページ割りに使用されます。

カテゴリ
ORM 関数

例
```
<cfset employees = ORMExecuteQuery("from Employee where age > ?", [40])>
<cfset employeeObj = ORMExecuteQuery("from Employee where EmployeeID=?", [1], true)>
<cfset employees = ORMExecuteQuery("from Employee where age > ? and age < ?", [40, 80])>
```

ORMExecuteQuery(hql, namedparams [, unique] [, queryOptions])

説明
このタイプの ORMExecuteQuery を使用すると、クエリーに名前付きパラメータを渡すことができます。このパラメータのプレースホルダーは、":age" や ":id" のように ":" で始まる名前である必要があります。名前の値はキーと値のペアとして渡す必要があります。また、queryOptions を使用して、取得動作を制御するための複数のオプションを指定できます。

- ignorecase: true に設定すると、ソート順の大文字と小文字が無視されます。このオプションは、sortorder パラメータを指定した場合にのみ使用します。
- maxResults: 取得されるオブジェクトの最大数を指定します。
- offset: 取得の開始場所となる、結果セットの開始索引を指定します。
- cacheable: このクエリーの結果がセカンダリキャッシュにキャッシュされるかどうかを指定します。デフォルトは false です。
- cachename: セカンダリキャッシュ内のキャッシュの名前です。
- timeout: クエリーのタイムアウト値 (秒単位) を指定します。

注意： maxresults および timeout はページ割りに使用されます。

カテゴリ
ORM 関数

例
米国に居住し、かつ米国民であるすべての従業員の従業員詳細を取得するには :

```
<cfset USEmployees = ORMExecuteQuery("from Employee where country=:country and
citizenship=:country", {country='USA'})>
<cfset orderDetail = ORMExecuteQuery("from Orders where OrderID=:orderid and
ProductID=:productid", {orderid=1, productid=901}, true)>
```

Hour

説明
現在の時刻の時間部分を取得します。

戻り値
0 ～ 23 の範囲の時間の序数値

カテゴリ
日付および時刻関数

関数のシンタックス

```
Hour(date)
```

関連項目

DatePart、Minute、Second

パラメータ

パラメータ	説明
date	日付時刻オブジェクトです。値の範囲は西暦 100 ～ 9999 年です。

使用方法

日付時刻値を文字列として渡すときは、その値を引用符で囲みます。引用符で囲まない場合、その値は日付時刻オブジェクトの数値表現として解釈されます。

例

```
<!--- This example shows the use of Hour, Minute, and Second --->
<h3>Hour Example</h3>
<cfoutput>
The time is currently #TimeFormat(Now())#.
We are in hour #Hour(Now())#, Minute #Minute(Now())#
and Second #Second(Now())# of the day.
</cfoutput>
```

HTMLCodeFormat

説明

文字列内の特殊文字を、HTML のエスケープ表記に置き換え、文字列の最初と最後に <pre> タグと </pre> タグを挿入します。

戻り値

HTML のエスケープ表記の文字列 (**string**) を <pre> と </pre> タグで囲んだ文字列。復帰文字は削除されます。改行文字は保持されます。HTML で特別の意味を持つ文字は、> などの HTML 文字に変換されます。

カテゴリ

表示および書式制御関数

関数のシンタックス

```
HTMLCodeFormat(string [, version ])
```

関連項目

HTMLEditFormat

パラメータ

パラメータ	説明
string	文字列、または文字列を含んでいる変数です。
version	使用する HTML バージョンです。現在は無視されます。 • -1: 最新の HTML 実装 • 2.0: HTML 2.0 (デフォルト) • 3.2: HTML 3.2

使用方法

この関数は、次の文字を HTML 文字エンティティに変換します。

テキスト文字	エンコード
<	<
>	>
&	&
"	"

通常はこの関数では文字列が長くなるため、拡張された文字列に対して特定の関数 (たとえば、Left、Right、Mid など) を実行したときに、予想外の結果が生じる可能性があります。

この関数と HTMLEditFormat 関数との違いは、HTMLEditFormat 関数は文字列を HTML の pre タグで囲まないということだけです。

例

```
<!--- This example shows the effects of HTMLCodeFormat and
HTMLEditFormat. View it in your browser; then View it
using your browser's the View Source command. --->
<cfset testString="This is a test
& this is another
<This text is in angle brackets>

Previous line was blank!!!">

<cfoutput>
<h3>The text without processing</h3>
#testString#<br>
<h3>Using HTMLCodeFormat</h3>
#HTMLCodeFormat(testString)#
<h3>Using HTMLEditFormat</h3>
#HTMLEditFormat(testString)#
</cfoutput>
```

HTMLEditFormat

説明

文字列内の特殊文字を、HTML のエスケープ表記に置き換えます。

戻り値
HTML のエスケープ表記の文字列 (**string**)。復帰文字は削除されます。改行文字は保持されます。HTML で特別の意味を持つ文字は、> などの HTML 文字に変換されます。

カテゴリ
表示および書式制御関数

関数のシンタックス
```
HTMLEditFormat(string [, version ])
```

関連項目
HTMLCodeFormat、cfapplication

パラメータ

パラメータ	説明
string	文字列、または文字列を含んでいる変数です。
version	使用する HTML バージョンです。現在は無視されます。 • -1: 最新の HTML 実装 • 2.0: HTML 2.0 (デフォルト) • 3.2: HTML 3.2

使用方法
この関数は、次の文字を HTML 文字エンティティに変換します。

テキスト文字	エンコード
<	<
>	>
&	&
"	"

この関数は、ユーザーが入力したデータをクライアントブラウザに返す ColdFusion ページを、クロスサイトスクリプティング攻撃から保護する目的に役立ちます。ただし、ほとんどの場合、cfapplication タグの scriptprotect 属性、または Application.cfc 内の同様の This.scriptProtect 変数設定を使用するほうが適していると考えられます。それは、1 つのアプリケーションに対して 1 回指定すれば済むからです。

通常はこの関数では文字列が長くなるため、拡張された文字列に対して特定の関数 (たとえば、Left、Right、Mid など) を実行したときに、予想外の結果が生じる可能性があります。

この関数と HTMLCodeFormat 関数との違いは、HTMLCodeFormat 関数は文字列を HTML の pre タグで囲むということだけです。

例

```
<!--- This example shows the effects of HTMLCodeFormat and
HTMLEditFormat. View it in your browser, then View it
using your browser's the View Source command. --->
<cfset testString="This is a test
& this is another
<This text is in angle brackets>

Previous line was blank!!!">

<cfoutput>
<h3>The text without processing</h3>
#testString#<br>
<h3>Using HTMLCodeFormat</h3>
#HTMLCodeFormat(testString)#
<h3>Using HTMLEditFormat</h3>
#HTMLEditFormat(testString)#
</cfoutput>
```

IIf

説明

Boolean 条件付きのダイナミックな式を評価します。式が yes または no のどちらであるかによって、2 つの文字列式の 1 つをダイナミックに評価し、その結果を返します。この関数は、HTML に cfif タグをインラインで組み込むときに便利です。

通常の条件付きの処理については、cfif を参照してください。エラー処理については、cftry を参照してください。詳細については、『ColdFusion アプリケーションの開発』を参照してください。

戻り値

結果が yes の場合は、Evaluate(string_expression1) の値。結果が no の場合は、Evaluate(string_expression2) の値。

カテゴリ

決定関数、ダイナミック評価関数

関数のシンタックス

```
IIf(condition, string_expression1, string_expression2)
```

関連項目

DE、Evaluate

パラメータ

パラメータ	説明
condition	ブール値として評価できる任意の式です。
string_expression1	文字列、または文字列を含んでいる変数です。条件が yes の場合に評価して返す文字列式です。
string_expression2	文字列、または文字列を含んでいる変数です。条件が no の場合に評価して返す文字列式です。

使用方法

IIf 関数は、次のコードの省略形です。

```
<cfif condition>
    <cfset result = Evaluate(string_expression1)>
<cfelse>
    <cfset result = Evaluate(string_expression2)>
</cfif>
```

式 **string_expression1** と **string_expression2** は文字列として指定する必要があります。そうしないと、これらの式が IIf のパラメータとして直接評価されることになります。たとえば、次のようになります。

```
IIf(y is 0, DE("Error"), x/y)
```

y= 0 の場合は、3 番目の式が x/0 の値になり有効でないため、エラーが発生します。

ColdFusion は **string_expression1** と **string_expression2** を評価します。文字列自体を返すには、DE 関数を使用します。

注意：string_expression1 または string_expression2 でシャープ記号 (#) を使うと、シャープ記号で囲まれた部分が最初に評価されます。シャープ記号を誤って使用すると、IIf 関数で予期せぬ結果が発生する可能性があります。たとえば、string_expression1 の式全体をシャープ記号 (#) で囲んだ場合、string_expression1 の中に未定義の変数があると、"Error Resolving Parameter" というエラーが発生して関数は正しく実行されません。

変数が未定義の場合、ColdFusion では、この関数を処理する際にエラーが発生します。次の例で、この問題を示します。

```
#IIf(IsDefined("Form.Deliver"), DE(Form.Deliver), DE("no"))#
```

この行を実行すると、"Error resolving parameter FORM.DELIVER" というエラーが返されます。

この問題を避けるには、DE 関数、および Evaluate 関数をコードで次のように使用します。

```
#IIf(IsDefined("Form.Deliver"), Evaluate(DE("Form.Deliver")), DE("no"))#
```

この場合は "no" が返され、エラーは発生しません。

次の例では、LocalVar が未定義です。ただし、LocalVar をシャープ記号 (#) で囲まなければ、このコードは正しく機能します。

```
<cfoutput>
    #IIf(IsDefined("LocalVar"), "LocalVar",
        DE("The variable is not defined."))#
</cfoutput>
```

出力は次のようになります。

```
    The variable is not defined.
```

次のコード内のように LocalVar をシャープ記号 (#) で囲むとエラーが発生し、パラメータの変換エラーメッセージが出力されます。これは、ColdFusion により、最初の条件である IsDefined("LocalVar") が評価されないためです。

別の例を示します。

```
<cfoutput>
#IIf(IsDefined("LocalVar"), DE("#LocalVar#"), DE("The variable is not defined."))#
</cfoutput>
```

エラーメッセージは次のようになります。

```
    Error resolving parameter LOCALVAR
```

シャープ記号 (#) で囲まれている LocalVar が最初に評価されるため、DE 関数は LocalVar の評価に影響を与えません。

例

```
<h3>IIf Function Example</h3>
<p>IIf evaluates a condition, and does an Evaluate on string
    expression 1 or string expression 2 depending on the Boolean
    outcome <I>(yes: run expression 1; no: run expression 2)</I>.</p>
<p>The result of the expression
IIf( Hour(Now()) GTE 12,
    DE("It is afternoon or evening"),
    DE("It is morning"))
is:<br><b>
<cfoutput>
    #IIf( Hour(Now()) GTE 12,
    DE("It is afternoon or evening"),
    DE("It is morning"))#
</cfoutput>
</b>
```

ImageAddBorder

説明

ColdFusion イメージの外縁に矩形のボーダーを追加します。

戻り値

なし

カテゴリ

Image 関数

関数のシンタックス

```
ImageAddBorder(name, thickness [, color, borderType])
```

関連項目

cfimage、ImageDrawRect、IsImageFile

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。

パラメータ	説明
thickness	必須です。ボーダーの太さです (単位 : ピクセル)。デフォルト値は 1 です。ボーダーはイメージの外縁に追加されるため、指定する太さに応じてイメージ領域が広くなります。
color	オプション。ボーダーの色です。デフォルトのボーダー色は黒です。「使用方法」を参照 borderType が指定されていない場合、または borderType = "constant" が指定されている場合にのみ有効です。
borderType	オプション。ボーダーの種類です。 • zero : ボーダーの色を黒に設定します。 • constant : ボーダーを指定した色に設定します (デフォルト)。 • copy : サンプル値を、最も近い有効ピクセルのコピーに設定します。たとえば、有効な矩形の左側のピクセルは、同じ行にある有効なエッジピクセルの値を取ります。有効な矩形の上および左側のピクセルは、左上のピクセルの値を取ります。 • reflect : ソースイメージのボーダーを反映します。たとえば、有効な矩形の左端が x = 10 の位置にある場合、ピクセル (9, y) はピクセル (10, y) のコピーになり、ピクセル (6, y) はピクセル (13, y) のコピーになります。 • wrap : 平面上にソースイメージをタイル状に配置します。

使用方法

ボーダーの太さは、thickness パラメータを使用してピクセル単位で指定します。0 未満の太さは指定できません。

色の値には、16 進数値またはサポートされているカラー名を指定します。有効な HTML カラー名については、332 ページの「cfimage」のリストを参照してください。16 進数値を入力するには、"##xxxxxx" または "xxxxxx" という形式を使用します (x は 0 ～ 9 または A ～ F です)。シャープ記号 (#) は 2 つ使用するか、または使用しないでください。

例

例 1

```
<!--- This example shows how to create a 10-pixel-wide red border around an image with a 5-pixel-wide green
border around the red border.--->
<!--- Create a ColdFusion image from an existing JPEG file. --->
<cfimage source="../cfdocs/images/artgallery/jeff05.jpg" name="myImage">
<!--- Draw a red border around the outside edge of the image. --->
<cfset ImageAddBorder(myImage,10,"red")>
<!--- Draw a green border around the outside edge of the red border. --->
<cfset ImageAddBorder(myImage,5,"green")>
<!--- Save the modified ColdFusion image to a file. --->
<cfimage source="#myImage#" action="write" destination="test_myImage.jpeg" overwrite="yes">
<!--- Display the source image and the new image. --->
<img src="../cfdocs/images/artgallery/jeff05.jpg"/>
<img src="test_myImage.jpeg"/>
```

例 2

```
<!--- This example shows how to create a border from the tiled image. --->
<!--- Create a ColdFusion image from an existing JPEG file. --->
<cfimage source="../cfdocs/images/artgallery/lori05.jpg" name="myImage">
<!--- Add a 50-pixel-wide border to the outside edge of the image that is a tiled version of the image
itself. --->
<cfset ImageAddBorder(myImage,50,"","wrap")>
<!--- Save the modified ColdFusion image to a file. --->
<cfimage source="#myImage#" action="write" destination="test_myImage.jpeg" overwrite="yes">
<!--- Display the source image and the new image. --->
<img src="../cfdocs/images/artgallery/lori05.jpg"/>
<img src="test_myImage.jpeg"/>
```

例 3

```
<!--- This example shows how to create a 100-pixel-wide border that is a mirror of the source image. --->
<!--- Create a ColdFusion image from an existing JPEG file. --->
<cfimage source="../cfdocs/images/artgallery/maxwell01.jpg" name="myImage">
<!--- Create the border. --->
<cfset ImageAddBorder(myImage,100,"","reflect")>
<!--- Save the modified ColdFusion image to a file. --->
<cfimage source="#myImage#" action="write" destination="test_myImage.jpeg" overwrite="yes">
<!--- Display the source image and the new image. --->
<img src="../cfdocs/images/artgallery/maxwell01.jpg"/>
<img src="test_myImage.jpeg"/>
```

例 4

```
<!--- This example shows how to copy 100 pixels from the outer edge of the image and create a border from
it. --->
<!--- Create a ColdFusion image from an existing JPEG file. --->
<cfimage source="../cfdocs/images/artgallery/jeff05.jpg" name="myImage">
<cfset ImageAddBorder(myImage,100,"","copy")>
<!--- Save the modified ColdFusion image to a file. --->
<cfimage source="#myImage#" action="write" destination="test_myImage.jpeg" overwrite="yes">
<!--- Display the source image and the new image. --->
<img src="../cfdocs/images/artgallery/jeff05.jpg"/>
<img src="test_myImage.jpeg"/>
```

ImageBlur

説明
ColdFusion イメージをぼかします。

戻り値
なし

カテゴリ
Image 関数

関数のシンタックス
```
ImageBlur(name [, blurRadius])
```

関連項目
ImageSharpen、IsImageFile

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。
blurRadius	オプション。ぼかし半径のサイズです。 3 ～ 10 の値を指定する必要があります。デフォルト値は 3 です。

使用方法
blurRadius のオペレーションはパフォーマンスに影響します。blurRadius の値を上げると、パフォーマンスは低下します。

例

```
<!--- This example shows how to blur an image by a radius of 10. --->
<!--- Create a ColdFusion image from an existing JPEG file. --->
<cfimage source="../cfdocs/images/artgallery/jeff05.jpg" name="myImage">
<!--- Use the maximum blur radius to blur the image. --->
<cfset ImageBlur(myImage,10)>
<!--- Save the modified ColdFusion image to a JPEG file. --->
<cfimage source="#myImage#" action="write" destination="test_myImage.jpeg" overwrite="yes">
<!--- Display the source image and the new image. --->
<img src="../cfdocs/images/artgallery/jeff05.jpg"/>
<img src="test_myImage.jpeg"/>
```

ImageClearRect

説明

現在の描画面の背景色で塗りつぶすことによって、指定した矩形を消去します。

戻り値

なし

カテゴリ

Image 関数

関数のシンタックス

```
ImageClearRect(name, x, y, width, height)
```

関連項目

ImageSetBackgroundColor、IsImageFile

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。
x	必須です。消去する矩形の x 座標です。
y	必須です。消去する矩形の y 座標です。
width	必須です。消去する矩形の幅です。
height	必須です。消去する矩形の高さです。

使用方法

この関数は ImageSetBackgroundColor 関数とともに使用します。

例

```
<!--- This example shows how to set the background color to green and draws four rectangles in that color
on the image. --->
<!--- Create a ColdFusion image from an existing JPEG file. --->
<cfimage source="../cfdocs/images/artgallery/jeff05.jpg" name="myImage">
<!--- Set the background color to green. --->
<cfset ImageSetBackgroundColor(myImage,"green")>
<!--- Draw four rectangles in the background color. --->
<cfset ImageClearRect(myImage,10,25,50,50)>
<cfset ImageClearRect(myImage,100,25,50,50)>
<cfset ImageClearRect(myImage,10,100,50,50)>
<cfset ImageClearRect(myImage,100,100,50,50)>
<!--- Save the modified ColdFusion image to a file. --->
<cfimage source="#myImage#" action="write" destination="test_myImage.jpeg" overwrite="yes">
<!--- Display the source image and the new image. --->
<img src="../cfdocs/images/artgallery/jeff05.jpg"/>
<img src="test_myImage.jpeg"/>
```

ImageCopy

説明

イメージの矩形領域をコピーします。

戻り値

コピーされた領域の ColdFusion イメージ

カテゴリ

Image 関数

関数のシンタックス

```
ImageCopy(name, x, y, width, height [, dx, dy])
```

関連項目

ImageNew、ImagePaste、IsImageFile

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。
x	必須です。コピー元になる矩形の x 座標です。
y	必須です。コピー元になる矩形の y 座標です。
width	必須です。コピー元になる矩形の幅です。
height	必須です。コピー元になる矩形の高さです。
dx	オプション。コピー先になる矩形の x 座標です。
dy	オプション。コピー先になる矩形の y 座標です。

使用方法
(**x**,**y**,**width**,**height**) で矩形を指定します。この領域が、(**dx**,**dy**) で指定された座標を左上隅とする矩形にコピーされます。

例
例 1

```
<!--- This example shows how to copy a rectangular area of an image to a new image. --->
<!--- Create a ColdFusion image from an existing JPEG file. --->
<cfimage source="../cfdocs/images/artgallery/lori05.jpg" name="myImage">
<!--- Copy the rectangular area specified by the coordinates (25,25,50,50) in the image to the rectangle
beginning at (75,75), and return this copied rectangle as a new ColdFusion image. --->
<cfset dupArea = ImageCopy(myImage,25,25,50,50,75,75)>
<!--- Write the result to a PNG file. --->
<cfimage source=#myImage# action="write" destination="test_myImage.png" overwrite="yes">
<!--- Display the source image and the new image. --->
<img src="../cfdocs/images/artgallery/lori05.jpg">
<img src="test_myImage.png">
```

例 2

```
<!--- This example shows how to copy a rectangular area from one image and paste it over another image. --->
<!--- Create a ColdFusion image named "myImage1" from an existing JPEG file.--->
<cfimage source="../cfdocs/images/artgallery/lori05.jpg" name="myImage1">
<!--- Create the ColdFusion image "myImage2" from a different JPEG file.
    --->
<cfimage source="../cfdocs/images/artgallery/maxwell01.jpg" name="myImage2">
<!--- Copy a rectangular region of "myImage1" as a new image. --->
<cfset resImage = ImageCopy(myImage1,1,1,55,55)>
<!--- Paste the rectangular area on the second image. --->
<cfset ImagePaste(myImage2,resImage,50,75)>
<!--- Write the second ColdFusion image to a file. --->
<cfimage action="write" source="#myImage2#" destination="test_myImage.jpg" overwrite="yes">
<!--- Display the two source files and the new file. --->
<img src="../cfdocs/images/artgallery/lori05.jpg"/>
<img src="../cfdocs/images/artgallery/maxwell01.jpg"/>
<img src="test_myImage.jpg"/>
```

ImageCreateCaptcha

説明
スパムを防止するために CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart) イメージを作成する。CAPTCHA イメージとは、機械では読み取れないように (ただし、人間には読み取れるように) 歪められたテキストを含むイメージであり、本人確認などに使用されます。

戻り値
イメージオブジェクト

シンタックス
imageCreateCaptcha (***height***, ***width***, ***text***)

imageCreateCaptcha (***height***, ***width***, ***text [, difficulty***])

imageCreateCaptcha (***height***, ***width***, ***text [***, difficulty, ***fonts***, ***fontsize***)

プロパティ

パラメータ	説明
height	必須です。イメージの高さです (単位 : ピクセル)。
width	必須です。イメージの幅です (単位 : ピクセル)。
text	必須です。CAPTCHA イメージ内に表示するテキスト文字列です。読みやすいように大文字を使用してください。CAPTCHA イメージ内では見分けられないので、スペースは使用しないでください。
difficulty	オプション。CAPTCHA テキストの複雑度のレベルです。テキストの歪みのレベルを次のいずれかの値で指定します。 • low • medium • high
font	オプション。CAPTCHA テキストで使用する有効なフォントです。複数のフォントを指定する場合は、カンマで区切ります。ColdFusion では、JDK で認識可能なシステムフォントのみがサポートされます。たとえば、Windows のディレクトリの TTF フォントは Windows でサポートされます。
fontsize	オプション。CAPTCHA イメージ内のテキストのフォントサイズです。この値は整数で指定する必要があります。

例

```
<h1>ImageCreateCaptcha Method</h1>
<cfset funcimg1 = ImageCreateCaptcha(35,400,"loner")>
                <cfimage action="writetoBrowser" source="#funcimg1#">
<cfset funcimg2 = ImageCreateCaptcha(35,400,"loner","high")>
                <cfimage action="writetoBrowser" source="#funcimg2#">
<cfset funcimg3 = ImageCreateCaptcha(35,400,"loner","high","serif,sansserif", "24")>
                <cfimage action="writetoBrowser" source="#funcimg3#">
```

ImageCrop

説明

指定した矩形領域まで ColdFusion イメージをトリミングします。

戻り値

なし

カテゴリ

Image 関数

関数のシンタックス

```
ImageCrop(name, x, y, width, height)
```

関連項目

ImageFlip、ImageResize、IsImageFile

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。
x	必須です。トリミング領域の x 原点です。
y	必須です。トリミング領域の y 原点です。
width	必須です。トリミング領域の幅です。
height	必須です。トリミング領域の高さです。

使用方法
矩形領域を空にすることはできません。また、矩形領域は完全にソースイメージの境界内に含まれている必要があります。

例
```
<!--- This example shows how to crop an image. --->
<!--- Create a ColdFusion image from an existing JPEG file. --->
<cfimage source="../cfdocs/images/artgallery/lori05.jpg" name="myImage">
<!--- Crop myImage to 100x100 pixels starting at the coordinates (10,10).
--->
<cfset ImageCrop(myImage,10,10,100,100)>
<!--- Write the result to a file. --->
<cfimage source="#myImage#" action="write" destination="test_myImage.jpg" overwrite="yes">
<!--- Display the source image and the new image. --->
<img src="../cfdocs/images/artgallery/lori05.jpg"/>
<img src="test_myImage.jpg"/>
```

ImageDrawArc

説明
円または楕円の孤を描画します。

戻り値
なし

カテゴリ
Image 関数

関数のシンタックス
```
ImageDrawArc(name, x, y, width, height, startAngle, arcAngle [, filled])
```

関連項目
ImageDrawCubicCurve、ImageDrawOval、ImageDrawQuadraticCurve、ImageSetAntialiasing、ImageSetDrawingColor、ImageSetDrawingStroke、IsImageFile

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。
x	必須です。円弧の左上隅の x 座標です。
y	必須です。円弧の左上隅の y 座標です。
width	必須です。円弧の幅です。
height	必須です。円弧の高さです。
startAngle	必須です。開始角度です (単位 : 度数)。
arcAngle	必須です。開始角度を基準とした弧の角度です。
filled	オプション。円弧を塗りつぶすかどうかを指定します。 • yes : 指定した描画色で円弧が塗りつぶされます。 • no : 円弧の輪郭だけを描画します (デフォルト)。

使用方法

円弧は、startAngle の位置から始まり、arcAngle で指定した角度だけ描画されます。この角度は 3 時の位置が 0 になります。正の値は反時計回りの回転、負の値は時計回りの回転を表します。

円弧の中心は、原点が (x,y) となる矩形の中心です。この矩形のサイズは width パラメータと height パラメータで指定します。

角度は、楕円の中心から境界矩形の右上隅を結ぶ線が常に 45 度になるように、境界矩形の直角でない範囲を基準として指定します。境界矩形の縦軸と横軸の長さが明らかに違う場合は、長いほうの軸に沿って弧の開始点と終了点が傾斜します。

filled パラメータが yes に設定されている場合は、楕円内の領域が現在の描画色で塗りつぶされます。

弧の色と線の属性を指定するには、ImageSetDrawingColor 関数と ImageSetDrawingStroke 関数を使用します。レンダリングされるイメージの質を高めるには、ImageSetAntialiasing 関数を使用します。

例

```
<!--- This example shows how to use the ImageNew function to create a blank
ColdFusion image that is 250 pixels wide and 180 pixels high. --->
<cfset myImage=ImageNew("",250,320)>
<!--- Set the drawing color for the arc to orange. --->
<cfset ImageSetDrawingColor(myImage,"orange")>
<!--- Turn on antialiasing to improve image quality. --->
<cfset ImageSetAntialiasing(myImage,"on")>
<!--- Draw an enclosed orange arc starting at the coordinate (5,5). --->
<cfset ImageDrawArc(myImage,5,5,200,300,100,100,"yes")>
<!--- Display the image in a browser. --->
<cfimage action="writeToBrowser" source="#myImage#">
```

ImageDrawBeveledRect

説明

エッジをベベル加工した矩形を描画します。

戻り値

なし

カテゴリ
Image 関数

関数のシンタックス
```
ImageDrawBeveledRect(name, x, y, width, height, raised [, filled])
```

関連項目
ImageClearRect、ImageDrawLine、ImageDrawLines、ImageDrawRect、ImageDrawRoundRect、ImageSetAntialiasing、ImageSetDrawingColor、ImageSetDrawingStroke

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。
x	必須です。矩形の x 座標です。
y	必須です。矩形の y 座標です。
width	必須です。矩形の幅です。
height	必須です。矩形の高さです。
raised	必須です。矩形を表面から浮き上がらせて表示するか、沈めて表示するかを指定します。 • yes : 矩形を浮き上がらせます。 • no : 矩形を沈めます (デフォルト)。
filled	オプション。矩形を塗りつぶすかどうかを指定します。 • yes: 指定した描画色で矩形が塗りつぶされます。 • no: 矩形の輪郭だけを描画します (デフォルト)。

使用方法
矩形のエッジをベベル加工し、左上隅から光が当たっているように表示することで、矩形のエッジにハイライト効果を適用します。ハイライト効果に使用される色は現在の描画色により決定されます。

filled パラメータが yes に設定されている場合は、立方形の領域が現在の描画色で塗りつぶされます。

矩形の色と線の属性を指定するには、ImageSetDrawingColor 関数と ImageSetDrawingStroke 関数を使用します。レンダリングされるイメージの質を高めるには、ImageSetAntialiasing 関数を使用します。

例

```
<!--- This example shows how to create a bevel-edged rectangle. --->
<!--- Use the ImageNew function to create a 200x200-pixel image. --->
<cfset myImage=ImageNew("",200,200)>
<!--- Turn on antialiasing to improve image quality. --->
<cfset ImageSetAntialiasing(myImage,"on")>
<!--- Set the drawing color for the image to light gray. --->
<cfset ImageSetDrawingColor(myImage,"lightgray")>
<!--- Draw a 3D gray, filled, raised rectangle. --->
<cfset ImageDrawBeveledRect(myImage,50,50,100,75,"yes","yes")>
<!--- Display the image in a browser. --->
<cfimage source="#myImage#" action="writeToBrowser">
```

ImageDrawCubicCurve

説明

三次曲線を描画します。

戻り値

なし

カテゴリ

Image 関数

関数のシンタックス

```
ImageDrawCubicCurve(name, ctrlx1, ctrly1, ctrlx2, ctrly2, x1, y1, x2, y2)
```

関連項目

ImageDrawQuadraticCurve、ImageDrawRect、ImageDrawRoundRect、ImageSetAntialiasing、ImageSetDrawingColor、ImageSetDrawingStroke、IsImageFile

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。
ctrlx1	必須です。三次曲線の最初のコントロールポイントの x 座標です。
ctrly1	必須です。三次曲線の最初のコントロールポイントの y 座標です。
ctrlx2	必須です。三次曲線の 2 番目のコントロールポイントの x 座標です。
ctrly2	必須です。三次曲線の 2 番目のコントロールポイントの y 座標です。
x1	必須です。三次曲線の開始点の x 座標です。
y1	必須です。三次曲線の開始点の y 座標です。
x2	必須です。三次曲線の終了点の x 座標です。
y2	必須です。三次曲線の終了点の y 座標です。

使用方法

```
Coordinates can be integers or real numbers.
```

三次曲線の色と線の属性を指定するには、ImageSetDrawingColor 関数と ImageSetDrawingStroke 関数を使用します。レンダリングされるイメージの質を高めるには、ImageSetAntialiasing 関数を使用します。

例

```
<!--- This example shows how to draw a curved line that looks like a stylized 7. --->
<!--- Use the ImageNew function to create a blank ColdFusion image that is 200 pixels wide and 380 pixels
high. --->
<cfset myImage=ImageNew("",200,380)>
<!--- Set the drawing color to magenta. --->
<cfset ImageSetDrawingColor(myImage,"magenta")>
<!--- Turn on antialiasing to improve image quality. --->
<cfset ImageSetAntialiasing(myImage,"on")>
<!--- Draw a curved line that starts at (380,28) and ends at (32,56) with its first control point at (120,380)
and its second control point at (5,15). --->
<cfset ImageDrawCubicCurve(myImage,120,380,5,15,380,28,32,56)>
<!--- Display the image in a browser. --->
<cfimage source="#myImage#" action="writeToBrowser">
```

ImageDrawLine

説明

2 組の x 座標と y 座標で定義される 1 本の線を ColdFusion イメージ上に描画します。

戻り値

なし

カテゴリ

Image 関数

関数のシンタックス

```
ImageDrawLine(name, x1, y1, x2, y2)
```

関連項目

ImageDrawLines、ImageSetAntialiasing、ImageSetDrawingColor、ImageSetDrawingStroke

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。
x1	必須です。線の開始点の x 座標です。
y1	必須です。線の開始点の y 座標です。
x2	必須です。線の終了点の x 座標です。
y2	必須です。線の終了点の y 座標です。

使用方法
(x1,y1) と (x2,y2) を指定して、開始点と終了点を定義します。開始点と終了点を同じ座標にすることはできません。

この関数は、ImageSetDrawingStroke 関数と ImageSetDrawingColor 関数で定義された属性の影響を受けます。レンダリングされるイメージの質を高めるには、ImageSetAntialiasing 関数を使用します。

例
```
<!--- This example shows how to draw a square bisected by
    a diagonal line. --->
<!--- Use the ImageNew function to create a 100x100-pixel image. --->
<cfset myImage=ImageNew("",100,100)>
<!--- Turn on antialiasing to improve image quality. --->
<cfset ImageSetAntialiasing(myImage,"on")>
<!--- Draw a diagonal line that bisects the square. --->
<cfset ImageDrawLine(myImage,0,0,100,100)>
<!--- Display the image in a browser. --->
<cfimage source="#myImage#" action="writeToBrowser">
```

ImageDrawLines

説明
x 座標と y 座標の配列によって定義される、一続きの連結線を描画します。

戻り値
なし

カテゴリ
Image 関数

関数のシンタックス
```
ImageDrawLines(name, xcoords, ycoords [, isPolygon, filled])
```

関連項目
ImageDrawBeveledRect、ImageDrawCubicCurve、ImageDrawLine、ImageDrawRect、ImageDrawRoundRect、ImageSetAntialiasing、ImageSetDrawingColor、ImageSetDrawingStroke、IsImageFile

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。
xcoords	必須です。x 座標の CFML 配列です。

パラメータ	説明
ycoords	必須です。y 座標の CFML 配列です。
isPolygon	オプション。線を使用して多角形を形成するかどうかを指定します。 • yes : 多角形を形成するように線が連結されます。 • no : 多角形を形成しません (デフォルト)。
filled	オプション。多角形を塗りつぶすかどうかを指定します。 • yes : 指定した描画色で多角形が塗りつぶされます。 • no : 多角形の輪郭だけを描画します (デフォルト)。

使用方法

1 組の (x,y) 座標を指定することによって、1 つの点が定義されます。

多角形を描画するには、isPolygon を yes に設定します。開始点と終了点を同じ座標にすることはできません。isPolygon を yes に設定すると、開始点と終了点をつなぐ線が描画され、多角形が形成されます。isPolygon を no に設定した場合、多角形を完成させる線は描画されません。

isPolygon パラメータと filled パラメータを yes に設定すると、現在の描画色で多角形が塗りつぶされます。

色と線の属性を指定するには、ImageSetDrawingColor 関数と ImageSetDrawingStroke 関数を使用します。レンダリングされるイメージの質を高めるには、ImageSetAntialiasing 関数を使用します。

例

```
<!--- This example shows how to draw a zigzag line. --->
<!--- Use the ImageNew function to create a 250x250-pixel image. --->
<cfset myImage=ImageNew("",250,250)>
<!--- Set the drawing color to cyan. --->
<cfset ImageSetDrawingColor(myImage,"cyan")>
<!--- Turn on antialiasing to improve image quality. --->
<cfset ImageSetAntialiasing(myImage,"on")>
<!--- Create arrays for the x and y coordinates. --->
<cfset x = ArrayNew(1)>
<cfset y = ArrayNew(1)>
<!--- Set the values for the x and y coordinates for three connected line segments: the first segment begins
at (100,50) and ends at (50,100). The second segment begins at (50, 100) and ends at (200,100). The third
segment begins at (200,100) and ends at (100,200). --->
    <cfset x[1] = "100">
    <cfset x[2] = "50">
    <cfset x[3] = "200">
    <cfset x[4] = "100">
    <cfset y[1] = "50">
    <cfset y[2] = "100">
    <cfset y[3] = "100">
    <cfset y[4] = "200">
<!--- Draw the lines on the image. --->
<cfset ImageDrawLines(myImage,x,y)>
<!--- Display the image in a browser. --->
<cfimage source=#myImage# action="writeToBrowser">
```

ImageDrawOval

説明

楕円を描画します。

戻り値
なし

カテゴリ
Image 関数

関数のシンタックス
```
ImageDrawOval(name, x, y, width, height [, filled])
```

関連項目
ImageDrawArc、ImageDrawCubicCurve、ImageDrawQuadraticCurve、ImageDrawRoundRect、ImageSetAntialiasing、ImageSetDrawingColor、ImageSetDrawingStroke、IsImageFile

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。
x	必須です。描画する楕円の左上隅の x 座標です。
y	必須です。描画する楕円の左上隅の y 座標です。
width	必須です。描画する楕円の幅です。
height	必須です。描画する楕円の高さです。
filled	オプション。楕円を塗りつぶすかどうかを指定します。 • yes : 指定した描画色で楕円が塗りつぶされます。 • no : 楕円の輪郭だけを描画します (デフォルト)。

使用方法
x、y、width、height の各引数で指定された矩形内に収まる円または楕円が描画されます。

filled パラメータが yes に設定されている場合は、楕円内の領域が現在の描画色で塗りつぶされます。

楕円の色と線の属性を指定するには、ImageSetDrawingColor 関数と ImageSetDrawingStroke 関数を使用します。レンダリングされるイメージの質を高めるには、ImageSetAntialiasing 関数を使用します。

例
例 1

```
<!--- This example shows how to draw a green filled oval. --->
<!--- Use the ImageNew function to create a 200x110-pixel image. --->
<cfset myImage=ImageNew("",200,110)>
<!--- Set the drawing color to green. --->
<cfset ImageSetDrawingColor(myImage,"green")>
<!--- Turn on antialiasing to improve image quality. --->
<cfset ImageSetAntialiasing(myImage,"on")>
<!--- Draw a filled green oval on the image. --->
<cfset ImageDrawOval(myImage,5,5,190,100,"yes")>
<!--- Display the image in a browser. --->
<cfimage source="#myImage#" action="writeToBrowser">
```

例 2

```
<!--- This example shows how to draw a red circle with
    a line through it. --->
<!--- Use the ImageNew function to create a 201x201-pixel image. --->
<cfset myImage=ImageNew("",201,201)>
<!--- Set the drawing color to red. --->
<cfset ImageSetDrawingColor(myImage,"red")>
<!--- Turn on antialiasing to improve image quality. --->
<cfset ImageSetAntialiasing(myImage,"on")>
<!--- Set the line width to 10 pixels. --->
<cfset attr=StructNew()>
<cfset attr.width = 10>
<cfset ImageSetDrawingStroke(myImage,attr)>
<!--- Draw a diagonal line starting at (40,40) and ending at (165,165) on myImage. --->
<cfset ImageDrawLine(myImage,40,40,165,165)>
<!--- Draw a circle starting at (5,5) and is 190 pixels high and 190 pixels wide. --->
<cfset ImageDrawOval(myImage,5,5,190,190)>
<!--- Display the image in a browser. --->
<cfimage source="#myImage#" action="writeToBrowser">
```

ImageDrawPoint

説明
指定した (x,y) 座標に点を描画します。

戻り値
なし

カテゴリ
Image 関数

関数のシンタックス
```
ImageDrawPoint(name, x, y)
```

関連項目
ImageDrawLine、ImageDrawLines、ImageSetAntialiasing、ImageSetDrawingColor、ImageSetDrawingStroke

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。
x	必須です。点の x 座標です。
y	必須です。点の y 座標です。

使用方法
描画する点の外観を制御するには、ImageSetDrawingStroke 関数と ImageSetDrawingColor 関数を使用します。たとえば、ImageSetDrawingStroke 関数の width 属性を 10 ピクセルに設定すると、(x,y) を中心とする 20 ピクセルの正方形が描画されます。レンダリングされるイメージの質を高めるには、ImageSetAntialiasing 関数を使用します。

例
```
<!--- This example shows how to draw a large square in the middle of an image. --->
<!--- Use the ImageNew function to create a 200x200-pixel image. --->
<cfset myImage=ImageNew("",200,200)>
<!---Set the drawing color to orange. --->
<cfset ImageSetDrawingColor(myImage,"orange")>
<!--- Turn on antialiasing to improve image quality. --->
<cfset ImageSetAntialiasing(myImage,"on")>
<!--- Set the drawing area to 10 pixels. --->
<cfset attr = StructNew()>
<cfset attr.width = 10>
<cfset ImageSetDrawingStroke(myImage,attr)>
<!--- Draw the point at (100,100). --->
<cfset ImageDrawPoint(myImage,100,100)>
<!--- Display the image in a browser. --->
<cfimage source="#myImage#" action="writeToBrowser">
```

ImageDrawQuadraticCurve

説明
曲線を描画します。この曲線は 1 つの点で制御されます。

戻り値
なし

カテゴリ
Image 関数

関数のシンタックス
```
ImageDrawQuadraticCurve(name, ctrlx1, ctrly1, ctrlx2, ctrly2, x1, y1, x2, y2)
```

関連項目
ImageDrawArc、ImageDrawOval、ImageDrawRoundRect、ImageSetAntialiasing、ImageSetDrawingColor、ImageSetDrawingStroke

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。
ctrlx1	必須です。二次曲線の最初のコントロールポイントの x 座標です。
ctrly1	必須です。二次曲線の最初のコントロールポイントの y 座標です。

パラメータ	説明
ctrlx2	必須です。二次曲線の 2 番目のコントロールポイントの x 座標です。
ctrly2	必須です。二次曲線の 2 番目のコントロールポイントの y 座標です。
x1	必須です。二次曲線の開始点の x 座標です。
y1	必須です。二次曲線の開始点の y 座標です。
x2	必須です。二次曲線の終了点の x 座標です。
y2	必須です。二次曲線の終了点の y 座標です。

使用方法

二次曲線とは 1 つのコントロールポイントによって制御される曲線です。曲線は、図形の最後の点から、指定された x 座標および y 座標に向かって描画されます。座標は整数または実数で指定できます。

二次曲線の色と線の属性を指定するには、ImageSetDrawingColor 関数と ImageSetDrawingStroke 関数を使用します。レンダリングされるイメージの質を高めるには、ImageSetAntialiasing 関数を使用します。

例

```
<!--- This example shows how to draw a curved line. --->
<!--- Use the ImageNew function to create a 400x400-pixel image. --->
<cfset myImage=ImageNew("",400,400)>
<!--- Turn on antialiasing to improve image quality. --->
<cfset ImageSetAntialiasing(myImage,"on")>
<!--- Set the drawing color to green. --->
<cfset ImageSetDrawingColor(myImage,"green")>
<!--- Draw a curved line on the image. --->
<cfset ImageDrawQuadraticCurve(myImage,120,320,5,15,380,280)>
<!--- Display the image in a browser. --->
<cfimage source="#myImage#" action="writeToBrowser">
```

ImageDrawRect

説明

矩形を描画します。

戻り値

なし

カテゴリ

Image 関数

関数のシンタックス

```
ImageDrawRect(name, x, y, width, height [, filled])
```

関連項目

ImageDrawBeveledRect、ImageDrawCubicCurve、ImageDrawLine、ImageDrawLines、ImageDrawOval、ImageDrawQuadraticCurve、ImageDrawRoundRect、ImageSetAntialiasing、ImageSetDrawingColor、ImageSetDrawingStroke、ImageDrawText、IsImageFile

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。
x	必須です。矩形の x 座標です。
y	必須です。矩形の y 座標です。
width	必須です。矩形の幅です。
height	必須です。矩形の高さです。
filled	オプション。矩形を塗りつぶすかどうかを指定します。 • yes: 指定した描画色で矩形が塗りつぶされます。 • no: 矩形の輪郭だけを描画します (デフォルト)。

使用方法
矩形の左端は x で指定した座標に、右端は x に width の値を加えた座標に描画されます。矩形の上端は y で指定した座標に、下端は y に height の値を加えた座標に描画されます。

filled パラメータが yes に設定されている場合は、矩形が現在の描画色で塗りつぶされます。

矩形の色と線の形式を設定するには、ImageSetDrawingColor 関数と ImageSetDrawingStroke 関数を使用します。レンダリングされるイメージの質を高めるには、ImageSetAntialiasing 関数を使用します。

例
```
<!--- This example shows how to draw a rectangle. --->
<!--- Use the ImageNew function to create a ColdFusion image that is 150 pixels wide and 200 pixels high. --->
<cfset myImage=ImageNew("",150,200)>
<!--- Set the drawing color for the image to yellow. --->
<cfset ImageSetDrawingColor(myImage,"yellow")>
<!--- Turn on antialiasing to improve image quality. --->
<cfset ImageSetAntialiasing(myImage,"on")>
<!--- Draw a filled yellow rectangle on the image. --->
<cfset ImageDrawRect(myImage,25,45,100,20,"yes")>
<!--- Display the image in a browser. --->
<cfimage source="#myImage#" action="writeToBrowser">
```

ImageDrawRoundRect

説明
角を丸めた矩形を描画します。

戻り値
なし

カテゴリ
Image 関数

関数のシンタックス

```
ImageDrawRoundRect(name, x, y, width, height, arcWidth, arcHeight [, filled])
```

関連項目

ImageDrawBeveledRect、ImageDrawCubicCurve、ImageDrawLine、ImageDrawLines、ImageDrawOval、ImageDrawQuadraticCurve、ImageDrawRect、ImageSetAntialiasing、ImageSetDrawingColor、ImageSetDrawingStroke、IsImageFile

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。
x	必須です。矩形の x 座標です。
y	必須です。矩形の y 座標です。
width	必須です。矩形の幅です。
height	必須です。矩形の高さです。
arcWidth	必須です。4 つの角に使用する孤の水平方向の直径です。
arcHeight	必須です。4 つの角に使用する孤の垂直方向の直径です。
filled	オプション。矩形を塗りつぶすかどうかを指定します。 • yes: 指定した描画色で矩形が塗りつぶされます。 • no: 矩形の輪郭だけを描画します (デフォルト)。

使用方法

矩形の左端は x で指定した座標に、右端は x に width の値を加えた座標に描画されます。矩形の上端は y で指定した座標に、下端は y に height の値を加えた座標に描画されます。

filled パラメータが yes に設定されている場合は、矩形が現在の描画色で塗りつぶされます。

矩形の色と線の属性を指定するには、ImageSetDrawingColor 関数と ImageSetDrawingStroke 関数を使用します。レンダリングされるイメージの質を高めるには、ImageSetAntialiasing 関数を使用します。

例

例 1

```
<!--- This example shows how to draw a square with rounded corners. --->
<!--- Create a 200x200-pixel image. --->
<cfset myImage=ImageNew("",200,200)>
<!--- Set the drawing color for the image to blue. --->
<cfset ImageSetDrawingColor(myImage,"blue")>
<!--- Turn on antialiasing to improve image quality. --->
<cfset ImageSetAntialiasing(myImage,"on")>
<!--- Draw a blue filled square with round corners of arcWidth=10 and arcHeight=2. --->
<cfset ImageDrawRoundRect(myImage,5,5,190,190,"yes",10,2)>
<!--- Display the image in a browser. --->
<cfimage source="#myImage#" action="writeToBrowser">
```

例 2

```
<!--- Create an image. --->
<cfset myImage = imageNew("",100,100,"argb")>
<!--- Create a text attribute collection. --->
<cfset textStruct = structNew()>
<cfset textStruct.size=16>
<cfset textStruct.style="bold">
<cfset textStruct.font="Trebuchet MS">

<cfoutput>
<cfloop from="1" to="20" index="i">
    <!--- Turn on antialiasing to improve the quality of the rendered image. --->
    <cfset ImageSetAntialiasing(myImage,"on")>
    <!--- Set the background color. --->
    <cfset ImageSetBackgroundColor(myImage,"cyan") />
    <cfset ImageClearRect(myImage,0,0,myImage.getwidth(),myImage.getheight())>
    <!--- Set the drawing color. --->
    <cfset ImageSetDrawingColor(myImage,"black") />
    <!--- Draw a rectangle with rounded corners. --->
    <cfset ImageDrawRoundRect(myImage,10,10,myImage.width-20, myImage.height-20,i,i,"yes")>
    <!--- Set the text arc value. --->
    <cfset ImageSetDrawingColor(myImage,"##cccccc")>
    <cfset ImageDrawText(myImage, "#i#",30,30,textStruct)>
    <!--- Write the image to a file. --->
    <cfset imageWrite(myImage,"#expandPath("#i#.png")#")>
    <!--- Display the image. --->
    <img src="#i#.png">
</cfloop>
</cfoutput>
```

ImageDrawText

説明

ColdFusion イメージ上にテキスト文字列を描画します。最初の文字のベースラインは、イメージ内の (x,y) の位置に配置されます。

戻り値

描画されたテキストの幅と高さが含まれる構造体。

カテゴリ

Image 関数

関数のシンタックス

```
ImageDrawText(name, str, x, y [, attributeCollection])
```

関連項目

ImageDrawArc、ImageDrawBeveledRect、ImageDrawCubicCurve、ImageDrawLine、ImageDrawLines、ImageDrawOval、ImageDrawQuadraticCurve、ImageDrawRect、ImageDrawRoundRect、ImageSetAntialiasing、ImageSetDrawingColor、ImageTranslateDrawingAxis、IsImageFile

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。
str	必須です。描画するテキスト文字列です。
x	必須です。文字列の開始点の x 座標です。
y	必須です。文字列の開始点の y 座標です。
attributeCollection	オプション。テキストの属性を指定する構造体です。「使用方法」を参照してください。

使用方法

オプションのキーと値のペアはすべて、attributeCollection 構造体で指定します。テキストの色を指定するには、ImageSetDrawingColor 関数を使用します。

attributeCollection

要素	説明
font	テキスト文字列の描画に使用するフォントの名前です。font 属性をしない場合、テキストはデフォルトのシステムフォントで描画されます。
size	テキスト文字列のフォントサイズです。デフォルト値は 10 ポイントです。
style	フォントに適用するスタイルです。 • bold • italic • boldItalic • plain (デフォルト)
strikethrough	テキストイメージに打ち消し線を適用するかどうかを指定します。 • yes : 縦書きのテキストの場合は、個々の文字に打ち消し線が適用されます。 • no (デフォルト)
underline	テキストイメージに下線を適用するかどうかを指定します。 • yes : 縦書きのテキストの場合は、個々の文字に下線が適用されます。 • no (デフォルト)

例

例 1

```
<!--- This example shows how to create a text string image. --->
<!--- Use the ImageNew function to create a 200x100-pixel image. --->
<cfset myImage=ImageNew("",200,100)>
<!--- Set the drawing color to green. --->
<cfset ImageSetDrawingColor(myImage,"green")>
<!--- Specify the text string and the start point for the text. --->
<cfset ImageDrawText(myImage,"It's not easy being green.",10,50)>
<!--- Display the image in a browser. --->
<cfimage source="#myImage#" action="writeToBrowser">
```

例 2

```
<!--- This example shows how to draw three text strings with different text attributes. --->
<!--- Use the ImageNew function to create a 400x400-pixel image. --->
<cfset myImage=ImageNew("",400,400)>
<!--- Set the text attributes. --->
<cfset attr = StructNew()>
<cfset attr.underline = "yes">
<cfset attr.size = 25>
<cfset attr.style = "bold">
<cfset ImageSetDrawingColor(myImage,"yellow")>
<!--- Draw the text string "ColdFusion Rocks!" starting at (100,150). --->
<cfset ImageDrawText(myImage,"ColdFusion Rocks!",100,150,attr)>
<!--- Set new text attributes. --->
<cfset attr=StructNew()>
<cfset attr.size = 18>
<cfset attr.strikethrough = "yes">
<cfset attr.style = "bolditalic">
<cfset ImageSetDrawingColor(myImage,"red")>
<!--- Draw the text string "Powered by ColdFusion" starting at (100,200).
    --->
<cfset ImageDrawText(myImage,"Powered by ColdFusion",110,200,attr)>
<!--- Set new text attributes. --->
<cfset attr = StructNew()>
<cfset attr.font="Arial">
<cfset attr.style="italic">
<cfset attr.size=15>
<cfset ImageSetDrawingColor(myImage,"white")>
<!--- Draw the text string "Coming in 2007" starting at (150,250). --->
<cfset ImageDrawText(myImage,"We've arrived",150,250,attr)>
<!--- Display the text image in a browser. --->
<cfimage source="#myImage#" action="writeToBrowser">
```

ImageFlip

説明
軸を中心にイメージを反転します。

戻り値
なし

カテゴリ
Image 関数

関数のシンタックス
```
ImageFlip(name [, transpose])
```

関連項目
ImageBlur、ImageClearRect、ImageCrop、ImageNegative、ImageNew、ImageOverlay、ImagePaste、ImageResize、ImageRotate、ImageScaleToFit、ImageSetAntialiasing、ImageSharpen、ImageShear、ImageTranslate、IsImageFile

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。
transpose	オプション。イメージを反転します。 • vertical : イメージの中心を通る架空の水平線を中心にしてイメージを反転します (デフォルト)。 • horizontal : イメージの中心を通る架空の垂直線を中心にしてイメージを反転します。 • diagonal : イメージの左上隅から右下隅を通る対角線を中心にしてイメージを反転します。 • antidiagonal : イメージの右上隅から左下隅を通る対角線を中心にしてイメージを反転します。 • ("90|180|270") : イメージを時計回りに 90 度、180 度、または 270 度回転させます。

使用方法

ImageFlip 関数の transpose パラメータを指定しなかった場合は、水平軸を中心にしてイメージが反転され、ソースイメージと上下が逆さまになります。レンダリングされるイメージの質を高めるには、ImageSetAntialiasing 関数を使用します。

例

例 1

```
<!--- This example shows how to rotate an image by 270 degrees. --->
<!--- Create a ColdFusion image from an existing JPEG file. --->
<cfimage source="../cfdocs/images/artgallery/paul03.jpg" name="myImage">
<!--- Turn on antialiasing to improve image quality. --->
<cfset ImageSetAntialiasing(myImage,"on")>
<!--- Rotate the image by 270 degrees. --->
<cfset ImageFlip(myImage,"270")>
<!--- Display the modified image in a browser. --->
<cfimage source="#myImage#" action="writeToBrowser">
```

例 2

```
<!--- This example shows how to flip an image on a vertical axis. --->
<!--- Create a ColdFusion image from an existing JPEG file. --->
<cfimage source="../cfdocs/images/artgallery/paul03.jpg" name="myImage">
<!--- Turn on antialiasing to improve image quality. --->
<cfset ImageSetAntialiasing(myImage,"on")>
<!--- Flip the image so that it is upside down. --->
<cfset ImageFlip(myImage,"vertical")>
<!--- Display the modified image in a browser. --->
<cfimage source="#myImage#" action="writeToBrowser">
```

例 3

```
<!--- This example shows how to flip an image on a horizontal axis. --->
<!--- Create a ColdFusion image from an existing JPEG file. --->
<cfimage source="../cfdocs/images/artgallery/paul03.jpg" name="myImage">
<!--- Turn on antialiasing to improve image quality. --->
<cfset ImageSetAntialiasing(myImage,"on")>
<!--- Flip the image so that it is a mirror image of the source. --->
<cfset ImageFlip(myImage,"horizontal")>
<!--- Display the modified image in a browser. --->
<cfimage source="#myImage#" action="writeToBrowser">
```

例 4

```
<!--- This example shows how to flip an image on a diagonal axis. --->
<!--- Create a ColdFusion image from an existing JPEG file. --->
<cfimage source="../cfdocs/images/artgallery/paul03.jpg" name="myImage">
<!--- Turn on antialiasing to improve image quality. --->
<cfset ImageSetAntialiasing(myImage,"on")>
<!--- Flip the image on a diagonal axis. --->
<cfset ImageFlip(myImage,"diagonal")>
<!--- Display the modified image in a browser. --->
<cfimage source="#myImage#" action="writeToBrowser">
```

例 5

```
<!--- This example shows how to flip an image on an antidiagonal axis. --->
<!--- Create a ColdFusion image from an existing JPEG file. --->
<cfimage source="../cfdocs/images/artgallery/paul03.jpg" name="myImage">
<!--- Turn on antialiasing to improve image quality. --->
<cfset ImageSetAntialiasing(myImage,"on")>
<!--- Flip the image on an antidiagonal axis. --->
<cfset ImageFlip(myImage,"antidiagonal")>
<!--- Display the modified image in a browser. --->
<cfimage source="#myImage#" action="writeToBrowser">
```

ImageGetBlob

説明
ソースイメージのバイト数を取得します。バイト数はソースイメージと同じイメージ形式で計算されます。

戻り値
BLOB のソースイメージのバイト数

カテゴリ
Image 関数

関数のシンタックス
```
ImageGetBlob(source)
```

関連項目
cfimage、ImageGetBufferedImage、ImageGetEXIFTag、ImageGetHeight、ImageGetIPTCTag、ImageGetWidth、ImageInfo、IsImage

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
source	必須です。オペレーションの対象になる ColdFusion イメージです。

使用方法
この関数は、データベースの BLOB 列に ColdFusion イメージを挿入する場合に使用します。

注意：ソースイメージを指定しないと、パラメータ検証エラーが発生します。

例

例 1

```
<!--- This example shows how to add a ColdFusion image to a database. --->
<!--- Create ColdFusion image from a an existing JPEG file. --->
<cfimage source="aiden01.jpg" name="myImage">
<!--- Use the cfquery tag to insert the ColdFusion image as a BLOB in the database. --->
<cfquery name="InsertBlobImage" datasource="myBlobData" >
INSERT into EMPLOYEES (FirstName,LastName,Photo)
VALUES ('Aiden','Quinn',<cfqueryparam value="#ImageGetBlob(myImage)#" cfsqltype='cf_sql_blob'>)
</cfquery>
```

例 2

次の例では、データベースから取得した BLOB データを基に、ImageNew 関数を使用して JPEG 形式のサムネールイメージを作成します。

```
<!--- Use the cfquery tag to retrieve all employee photos and employee IDs from a database. --->
<cfquery name= "GetBLOBs" datasource= "myBlobData">
SELECT EMLPOYEEID, PHOTO FROM Employees
</cfquery>
<!--- Use the ImageNew function to create a ColdFusion image from the BLOB data that was retrieved from the
database. --->
<cfset myImage = ImageNew(#GetBLOBs.PHOTO#)>
<!--- Create thumbnail versions of the images by resizing them to a 100x100-pixel image, while maintaining
the aspect ratio of the
    source image. --->
<cfset ImageScaleToFit(myImage,100,"")>
<!--- Convert the images to JPEG format and save them to files in the thumbnails subdirectory, using the
employee ID as the filename. --->
<cfimage source="#myImage#" action-"write" destination="images/thumbnails/#GetBLOBs.EMPLOYEID#.jpg">
```

ImageGetBufferedImage

説明

現在の ColdFusion イメージの基になっている java.awt.BufferedImage オブジェクトを返します。

戻り値

java.awt.BufferedImage オブジェクト

カテゴリ

Image 関数

関数のシンタックス

```
ImageGetBufferedImage(name)
```

関連項目

cfimage、ImageGetBlob、ImageGetEXIFTag、ImageGetHeight、ImageGetIPTCTag、ImageGetWidth、ImageInfo、IsImage、IsImageFile

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。

使用方法
この関数は、ページに埋め込まれている他の Java AWT (Abstract Windowing Toolkit) オブジェクトとともに使用できるイメージオブジェクトを返します。

例
```
<!--- This example shows how to create a ColdFusion image, modify it, and retrieve the width of the buffered image. --->
<!--- Create a ColdFusion image from an existing JPEG file. --->
<cfimage source="../cfdocs/images/artgallery/paul05.jpg" name="myImage">
<!--- Blur the image by an order of 10. --->
<cfset ImageBlur(myImage,10)>
<!--- Get the blurred image from the buffer and set it to variable x. --->
<cfset x = ImageGetBufferedImage(myImage)>
<!--- Return the width of the buffered image. --->
<cfoutput>#x.getWidth()#
</cfoutput>
```

ImageGetEXIFMetadata

説明
イメージ内の EXIF (Exchangeable Image File Format) ヘッダーを CFML 構造体として取得します。

戻り値
EXIF ヘッダー値を含む構造体

カテゴリ
Image 関数

関数のシンタックス
```
ImageGetEXIFMetadata(name)
```

関連項目
cfimage、ImageGetEXIFTag、ImageGetBlob、ImageGetBufferedImage、ImageGetHeight、ImageGetIPTCTag、ImageGetWidth、ImageInfo、IsImage、IsImageFile

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。

使用方法

EXIF は、イメージファイル内に交換情報を格納するための規格で、おもに JPEG ファイルなどで使用されます。また、多くのデジタルカメラでも EXIF 形式が使用されています。

EXIF メタデータには、作成日、イメージの作成に使用されたソフトウェア、アパーチャ、カメラの製造元と型名、イメージの解像度など、イメージの作成に関する情報が含まれています。

ImageGetEXIFMetadata 関数の結果は、パフォーマンスを最適化するために ColdFusion イメージ内にキャッシュされます。

ImageGetEXIFMetadata 関数は JPEG イメージに対してのみ使用できます。他の形式のイメージ (Base64 や BLOB など) のメタデータを取得しようとすると、エラーが発生します。

例

```
<!--- This example shows how to retrieve the EXIF header information from a
JPEG file. --->
<!--- Create a ColdFusion image from an existing JPEG file. --->
<cfimage source="images\paul05.jpg" name="myImage">
<!--- Retrieve the metadata associated with the image. --->
<cfset data =ImageGetEXIFMetadata(myImage)>
<!--- Display the ColdFusion image parameters. --->
<cfdump var="#myImage#">
<!--- Display the EXIF header information associated with the image
(creation date, software, and so on). --->
<cfdump var="#data#">
```

ImageGetEXIFTag

説明

指定した EXIF タグをイメージから取得します。

戻り値

指定した EXIF タグの値

カテゴリ

Image 関数

関数のシンタックス

```
ImageGetEXIFTag(name, tagName)
```

関連項目

cfimage、ImageGetBlob、ImageGetBufferedImage、ImageGetHeight、ImageGetIPTCTag、ImageGetWidth、ImageInfo、IsImage、IsImageFile

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。
tagName	必須です。値を返す EXIF タグの名前です。

使用方法

ImageGetEXIFTag 関数は JPEG イメージに対してのみ使用できます。他の形式のイメージ (Base64 や BLOB など) のメタデータを取得しようとすると、エラーが発生します。

例

```
<!--- This example shows how to retrieve one element from the EXIF information associated with an image. --->
<!--- Create a ColdFusion image from an existing JPEG file. --->
<cfimage source="../cfdocs/images/artgallery/paul05.jpg" name="myImage">
<!--- Retrieve the name of the software application used to create the original image. --->
<cfset data = ImageGetEXIFTag(myImage,"software")>
<!--- Display the name of the software. --->
<cfdump var="#data#">
```

ImageGetHeight

説明

ColdFusion イメージの高さを取得します (単位 : ピクセル)。

戻り値

指定した ColdFusion イメージの高さ (単位 : ピクセル)

カテゴリ

Image 関数

関数のシンタックス

```
ImageGetHeight(name)
```

関連項目

cfimage、ImageGetBlob、ImageGetBufferedImage、ImageGetEXIFTag、ImageGetIPTCTag、ImageGetWidth、ImageInfo、IsImage、IsImageFile

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。

使用方法

この関数は、ColdFusion イメージの高さを取得する場合に使用します。

例

```
<!--- This example shows how to retrieve the height of an image. --->
<!--- Create a ColdFusion image from a JPEG file. --->
<cfimage source="../cfdocs/images/artgallery/jeff05.jpg" name="myImage">
<!--- Retrieve the height of the image. --->
<cfset height=#ImageGetHeight(myImage)#>
<!--- Display the height of the image. --->
<cfdump var="#height#">
```

ImageGetIPTCMetadata

説明

ColdFusion イメージ内の IPTC (International Press Telecommunications Council) ヘッダーを構造体として取得します。IPTC メタデータには、そのメタデータに関連付けられているイメージについての説明テキストが含まれています。IPTC メタデータには、キャプション、キーワード、作成者、著作権、オブジェクト名、作成日、署名、表題、出典などの情報が含まれています。

戻り値

IPTC ヘッダー値を含む構造体

カテゴリ

Image 関数

関数のシンタックス

```
ImageGetIPTCMetadata(name)
```

関連項目

cfimage、ImageGetBlob、ImageGetBufferedImage、ImageGetEXIFMetadata、ImageGetEXIFTag、ImageGetHeight、ImageGetWidth、ImageInfo、IsImage、IsImageFile

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。

使用方法

IPTC メタデータには、そのメタデータに関連付けられているイメージについての説明テキストが含まれています。IPTC メタデータには、キャプション、キーワード、作成者、著作権、オブジェクト名、作成日、署名、表題、出典などの情報が含まれています。

ImageGetIPTCMetadata 関数の結果は、パフォーマンスを最適化するために ColdFusion イメージ内にキャッシュされます。

ImageGetIPTCMetada 関数は JPEG イメージに対してのみ使用できます。他の形式のイメージ (Base64 や BLOB など) のメタデータを取得しようとすると、エラーが発生します。

例

```
<!--- This example shows how to retrieve the IPTC header information for a
JPEG file. --->
<!--- Create a ColdFusion image from a JPEG file. --->
<cfimage source="images\aiden01.jpg" name="myImage">
<!--- Retrieve the IPTC header information saved with the image, such as
copyright, caption, and headline. --->
<cfset data = ImageGetIPTCMetadata(myImage)>
<!--- Display the parameter values for the ColdFusion image. --->
<cfdump var="#myImage#">
<!--- Display the IPTC header information. --->
<cfdump var=#data#>
```

ImageGetIPTCTag

説明

ColdFusion イメージの IPTC タグの値を取得します。

戻り値

IPTC タグの値

カテゴリ

Image 関数

関数のシンタックス

```
ImageGetIPTCTag(name, tagName)
```

関連項目

cfimage、ImageGetBlob、ImageGetBufferedImage、ImageGetEXIFMetadata、ImageGetEXIFTag、ImageGetHeight、ImageGetIPTCMetadata、ImageGetWidth、ImageInfo、IsImage、IsImageFile

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。
tagName	必須です。値を返す IPTC タグの名前です。

使用方法

ImageGetIPTCTag 関数は JPEG イメージに対してのみ使用できます。他の形式のイメージ (Base64 や BLOB など) のメタデータを取得しようとすると、エラーが発生します。

例

```
<!--- This example shows how to retrieve the caption for a JPEG file. --->
<!--- Create a ColdFusion image from a JPEG file. --->
<cfimage source="../cfdocs/images/artgallery/paul05.jpg" name="myImage" action="read">
<!--- Retrieve the camera make used to take the original picture. --->
<cfset cameraMake=ImageGetIPTCTag(myImage,"make")>
<cfdump var="#cameraMake#>"
```

ImageGetWidth

説明
指定した ColdFusion イメージの幅を取得します。

戻り値
ColdFusion イメージの幅を示す整数 (単位 : ピクセル)

カテゴリ
Image 関数

関数のシンタックス
```
ImageGetWidth(name)
```

関連項目
cfimage、ImageGetBlob、ImageGetBufferedImage、ImageGetEXIFTag、ImageGetHeight、ImageGetIPTCTag、ImageInfo、IsImage、IsImageFile

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。

例
```
<!--- This example shows how to retrieve the width of an image. --->
<!--- Create a ColdFusion image from an existing JPEG file.--->
<cfimage source="../cfdocs/images/artgallery/jeff05.jpg" name="myImage">
<!--- Get the width of the image. --->
<cfset width=#ImageGetWidth(myImage)#>
<!--- Display the width of the image in pixels. --->
<cfdump var=#width#>
```

ImageGrayscale

説明
ColdFusion イメージをグレースケールに変換します。

戻り値
なし

カテゴリ
Image 関数

関数のシンタックス
```
ImageGrayscale(name)
```

関連項目
ImageBlur、ImageNegative、ImageSetAntialiasing、ImageSharpen、IsImageFile

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。

使用方法
レンダリングされるイメージの質を高めるには、ImageSetAntialiasing 関数を使用します。

例
```
<!--- This example shows how to change a color image to grayscale. --->
<!--- Create a ColdFusion image from an existing color image. --->
<cfimage source="../cfdocs/images/artgallery/jeff04.jpg" name="myImage">
<!--- Turn on antialiasing to improve image quality. --->
<cfset ImageSetAntialiasing(myImage,"on")>
<!--- Change the image to grayscale. --->
<cfset ImageGrayscale(myImage)>
<!--- Save the grayscale image to a JPEG file. --->
<cfimage source="#myImage#" action="write" destination="test_myImage.jpg" overwrite="yes">
<!--- Display the source image and the grayscale image. --->
<img src="../cfdocs/images/artgallery/jeff04.jpg"/>
<img src="test_myImage.jpg"/>
```

ImageInfo

説明
高さ、幅、カラーモデル、サイズ、ファイル名などの情報を含む構造体を返します。

戻り値
イメージパラメータの情報を含む構造体

カテゴリ
Image 関数

関数のシンタックス
```
ImageInfo(name)
```

関連項目
cfimage、ImageGetHeight、ImageGetWidth、IsImage、IsImageFile

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。

使用方法
この関数は、イメージに互換性があるかどうかを調べる場合に使用します。たとえば、ImageOverlay 関数を使用して 2 つのイメージを重ねる場合は、両方のイメージで同じカラーモデルが使用されている必要があります。

例

```
<!--- This example shows how to retrieve information associated with the image. --->
<!--- Create a ColdFusion image from a JPEG file.--->
<cfimage source="../cfdocs/images/artgallery/jeff05.jpg" name="myImage">
<!--- Retrieve the information associated with the image. --->
<cfset info=ImageInfo(myImage)>
<cfdump var="#info#"></cfdump>
<p>height = <cfoutput>#info.height#</cfoutput>
<p>width = <cfoutput>#info.width#</cfoutput>
<p>source = <cfoutput>#info.source#"</cfoutput>
<p>pixel size = <cfoutput>#info.colormodel.pixel_size#</cfoutput>
<p>transparency = <cfoutput>#info.colormodel.transparency#</cfoutput>
```

ImageMakeColorTransparent

説明
イメージを作成し、透明色を設定します。

戻り値
イメージオブジェクト

シンタックス
imageMakeColorTransparent(***img***, ***color***)

履歴
ColdFusion 10: この関数が追加されました。

プロパティ

パラメータ	説明
img	必須です。オペレーションの対象になる ColdFusion イメージです。
color	必須です。透明色です。 • RGB カラーの 16 進数値。たとえば、白の場合は "##FFFFFF" または "FFFFFF" を指定します。 • 色の文字列値 ("black"、"red"、"green" など)。 • 透明色のデフォルト値は "black" です。（R,G,B）値を示す 3 つの数値のリスト。各値の範囲は 0 ～ 255 です。

例

```
<cfset myImage=ImageNew("",200,110)>
<!--- Set the drawing color to green. --->
<cfset ImageSetDrawingColor(myImage,"green")>
<!--- Turn on antialiasing to improve image quality. --->
<cfset ImageSetAntialiasing(myImage,"on")>
<!--- Draw a filled green oval on the image. --->
<cfset ImageDrawOval(myImage,5,5,190,100,"yes")>
<!--- Display the image in a browser. --->
<cfoutput>PNG image<br></cfoutput>
<cfset ImageWrite(myImage,"#expandpath('.')#/png.png")>
<cfset myImage = ImageRead("#expandpath('.')#/png.png")>
<cfimage source="#myImage#" action="writeToBrowser" >
<cfset x =ImageMakeColorTransparent(myImage,"green")>
<cfimage source="#x#" action="writeToBrowser" >
```

ImageMakeTranslucent

説明

指定したパーセントの透明度で、新しい半透明のイメージを作成します。

戻り値

イメージオブジェクト

シンタックス

imageMakeTranslucent (***img***, ***percent***)

プロパティ

パラメータ	説明
img	必須です。必須です。オペレーションの対象になる ColdFusion イメージです。
percentage	必須です。透明度のパーセントです。 • .0 = 不透明 • 100 = 透明 10 進数の値がサポートされます。

例

次の例では、3 つのイメージを使用し、2 つ目のイメージは 1 つ目よりも透明度を高く、3 つ目のイメージは 2 つ目よりも透明度を高くします。

```
<cfset myImage=ImageNew("",200,110)>
<!--- Set the drawing color to green. --->
<cfset ImageSetDrawingColor(myImage,"green")>
<!--- Turn on antialiasing to improve image quality. --->
<cfset ImageSetAntialiasing(myImage,"on")>
<!--- Draw a filled green oval on the image. --->
<cfset ImageDrawOval(myImage,5,5,190,100,"yes")>
<!--- Display the image in a browser. --->
<cfoutput>PNG image<br></cfoutput>
<cfset ImageWrite(myImage,"#expandpath('.')#/png.png")>
<cfset myImage = ImageRead("#expandpath('.')#/png.png")>
<cfimage source="#myImage#" action="writeToBrowser" >
<cfset x =ImageMakeTranslucent(myImage,50)>
<cfimage source="#x#" action="writeToBrowser" >
<cfset x =ImageMakeTranslucent(myImage,75)>
<cfimage source="#x#" action="writeToBrowser" >
<cfset x =ImageMakeTranslucent(myImage,100)>
<cfimage source="#x#" action="writeToBrowser" >
</body></html?
```

ImageNegative

説明
ColdFusion イメージのピクセル値を反転します。

戻り値
なし

カテゴリ
Image 関数

関数のシンタックス
```
ImageNegative(name)
```

関連項目
ImageBlur、ImageGrayscale、ImageSetAntialiasing、ImageSharpen、IsImageFile

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。

使用方法
生成されるイメージの寸法はソースイメージと同じですが、バイト数は必ずしも一致しません。レンダリングされるイメージの質を高めるには、ImageSetAntialiasing 関数を使用します。

例

```
<!--- This example shows how to create a negative version of an image. --->
<!--- Create a ColdFusion image from and existing JPEG file. --->
<cfimage source="../cfdocs/images/artgallery/jeff05.jpg" name="myImage">
<!--- Turn on antialiasing to improve image quality. --->
<cfset ImageSetAntialiasing(myImage,"on")>
<!--- Create a negative version of the image. --->
<cfset ImageNegative(myImage)>
<!--- Save the modified image to a file. --->
<cfimage source="#myImage#" action="write" destination="test_myImage.jpg" overwrite="yes">
<!--- Display the source image and the negative image. --->
<img src="../cfdocs/images/artgallery/jeff05.jpg"/>
<img src="test_myImage.jpg"/>
```

ImageNew

説明

ColdFusion イメージを作成します。

戻り値

ColdFusion イメージ

カテゴリ

Image 関数

関数のシンタックス

```
ImageNew([source, width, height, imageType, canvasColor])
```

関連項目

cfimage、ImageCopy、ImageRead、ImageReadBase64、ImageSetDrawingColor、IsImageFile

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
source	オプション。新しい ColdFusion イメージまたは Java バッファー内のイメージに読み込むソースイメージのディスク上またはメモリ内のパス名、URL、ColdFusion 変数です。
width	オプション。新しい ColdFusion イメージの幅です。height が指定されていて、source が指定されていない場合にのみ有効です。

パラメータ	説明
height	オプション。新しい ColdFusion イメージの高さです。width が指定されていて、source が指定されていない場合にのみ有効です。
imageType	オプション。作成する ColdFusion イメージの種類です。 • rgb • argb • grayscale width と height が指定されている場合にのみ有効です。
canvasColor	オプション。イメージキャンバスの色です。 • RGB カラーの 16 進数値。たとえば、白の場合は ##FFFFFF または FFFFFF と指定します。 • 色の文字列値 ("black"、"red"、"green" など)。描画色のデフォルト値は "black" です。 • (R,G,B) 値を示す 3 つの数値のリスト。各値の範囲は 0 ～ 255 です。

使用方法

ImageNew 関数には、次のパラメータを渡すことができます。

- 絶対パス名または相対パス名 : ディスク上の指定したパスにあるイメージファイルが読み込まれ、ColdFusion イメージとして返されます。
- URL : 指定した URL からイメージが読み込まれ、ColdFusion イメージとして返されます。
- 幅および高さ (imageType はオプション) : 指定した属性を持つ空の ColdFusion イメージが返されます。
- ColdFusion イメージ変数 : メモリ内のイメージ変数 (#myImage# など)。
- イメージデータを含むデータベース上の BLOB。
- Base64 イメージデータを含むバイト配列。
- Java バッファー内のイメージ。

  有効な ColdFusion イメージを作成できない属性を渡した場合は、エラーが発生します。

  ImageNew 関数および cfimage の read アクションでは、SQL Server の Image Column データ型がサポートされています。

  Base64 イメージを読み込む場合は、ImageReadBase64 関数を使用します。

  色の値を文字列で指定する場合は、サポートされているカラー名を使用します。有効な HTML カラー名については、332 ページの「cfimage」を参照してください。16 進数値を入力するには、"##xxxxxx" または "xxxxxx" という形式を使用します (x は 0 ～ 9 または A ～ F です)。シャープ記号 (#) は 2 つ使用するか、または使用しないでください。

  注意 : イメージの種類として ARGB を指定するとイメージは白になります。一方、RGB またはグレースケールを指定するとイメージは黒になります。空のイメージを常に同じ設定で作成するには、canvasColor パラメータを使用します。

例

```
Example 1
<!--- Use the ImageNew function to create a 200x200-pixel image in ARGB format. --->
<cfset myImage = ImageNew("",200,200,"argb")>
<cfimage action="writeTobrowser" source="#myImage#">
```

例 2

```
<!--- This example shows how to create a ColdFusion image from a BLOB in a database. --->
<cfquery
name="GetBLOBs" datasource="myblobdata">
SELECT LastName,Photo
FROM Employees
</cfquery>
<cfset i = 0>
<table border=1>
    <cfoutput query="GetBLOBS">
        <tr>
            <td>
            #LastName#
            </td>
            <td>
            <cfset i = i+1>
            <cfset myImage=ImageNew("#GetBLOBS.Photo#")>
            <cfset ImageWrite(myImage,"photo#i#.png")>
            </td>
        </tr>
    </cfoutput>
</table>
```

例 3

```
<!--- This example shows how to create a ColdFusion image from a URL. --->
<cfset myImage = ImageNew("http://www.google.com/images/logo_sm.gif")>
<cfset ImageWrite(myImage,"google_via_imagenew.png")>
<img src="google_via_imagenew.png">
```

例 4

```
<!--- This example shows how to use the cffile tag to convert an image file to binary format and pass it
as a variable to the ImageNew function. --->
<!---Use the cffile tag to read an image file, convert it to binary format, and write the result to a
variable. --->
<cffile action = "readBinary" file = apple.jpg"
    variable = "aBinaryObj">
<!--- Use the ImageNew function to create a ColdFusion image from the variable. --->
<cfset myImage = ImageNew(aBinaryObj)>
```

例 5

```
<!--- This example shows how to use the cffile tag to write a ColdFusion image to a file. --->
<!--- Use the ImageNew function to create a ColdFusion image from a JPEG file. --->
<cfset myImage = ImageNew("../cfdocs/images/artgallery/aiden01.jpg")>
<!--- Turn on antialiasing to improve image quality. --->
<cfset ImageSetAntialiasing(myImage,"on")>
<!--- Resize the image. --->
<cfset ImageResize(myImage,"50%","")>
<!--- Pass the image object to the cffile tag and write the result to a file on the local drive. --->
<cffile file="#myImage#" action="write" output="c:\test_myImage.jpg">
<cfimage action="writeToBrowser" source="#myImage#">
```

例 6

```
<!--- This example uses cfscript to pass a Java buffered image to the ImageNew function. --->
<cfscript>
    bufferedImage = createObject("java", "java.awt.image.BufferedImage");
    bufferedImage.init(JavaCast("int", 100), JavaCast("int", 100), BufferedImage.TYPE_4BYTE_ABGR);
    myImage = imageNew(bufferedImage);
</cfscript>
```

ImageOverlay

説明
2 つの ColdFusion ソースイメージを読み込み、2 番目のソースイメージを最初のソースイメージの上に重ねます。

戻り値
なし

カテゴリ
Image 関数

関数のシンタックス
```
ImageOverlay(source1, source2 [, rule, alpha])
```

関連項目
ImageCopy、ImagePaste、ImageSetAntialiasing、IsImageFile

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
source1	必須です。ColdFusion イメージの下層レイヤーになる ColdFusion イメージです。
source2	必須です。ColdFusion イメージの上層レイヤーになる (source1 イメージの上に重ねられる) ColdFusion イメージです。
rule	オプション。サポートされている値は、SRC、DST_IN、DST_OUT、DST_OVER、SRC_IN、SRC_OVER または SRC_OUT です。 詳しくは、Java documentation を参照してください。
alpha	オプション。透明度のパーセント値です。

使用方法
生成されるイメージは最初のソースイメージと同じ境界矩形を持ち、イメージの種類は 2 つのソースと同じになります。2 つのソースイメージが交わらない場合、生成されるイメージは最初のソースイメージと同じになります。

2 つのソースイメージのカラーモデルは同じである必要があります。たとえば、RGB イメージに別の RGB イメージを重ねることは可能ですが、グレースケールイメージに RGB イメージを重ねることはできません。イメージのカラーモデルを確認するには、ImageInfo 関数を使用します。

レンダリングされるイメージの質を高めるには、ImageSetAntialiasing 関数を使用します。

例

```
<!--- This example shows how to overlay a smaller image on a
larger image. --->
<!--- Create a ColdFusion image from an existing JPEG file and enlarge it by 150%. This image is displayed
in the background. --->
<cfimage source="../cfdocs/images/artgallery/maxwell01.jpg" name="myImage" action="resize" width="150%"
height="150%">
<!--- Turn on antialiasing to improve image quality. --->
<cfset ImageSetAntialiasing(myImage,"on")>
<!--- Create a ColdFusion image from an existing JPEG file. This image is overlaid on the background image.
--->
<cfimage source="../cfdocs/images/artgallery/viata05.jpg" name="topImage">
<!--- Overlay the top image on the background image. --->
<cfset ImageOverlay(myImage,topImage)>
<!--- Display the combined image in a browser. --->
<cfimage source="#myImage#" action="writeToBrowser">
```

ImagePaste

説明

パラメータとして 2 つのイメージと (x,y) 座標を取り、2 番目のイメージを最初のイメージの上に描画します。(x,y) では、左上隅の座標を指定します。

戻り値

ColdFusion イメージ

カテゴリ

Image 関数

関数のシンタックス

```
ImagePaste(image1, image2, x, y)
```

関連項目

ImageCopy、ImageOverlay、ImageSetAntialiasing、IsImageFile

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
image1	必須です。下になる ColdFusion イメージです。
Image2	必須です。image1 の上に貼り付けられる ColdFusion イメージです。
x	必須です。image2 が貼り付けられる位置の左上隅の x 座標です。
y	必須です。image2 が貼り付けられる位置の左上隅の y 座標です。

使用方法

レンダリングされるイメージの質を高めるには、ImageSetAntialiasing 関数を使用します。

例

```
<!--- This example shows how to copy a small rectangular area of one image and paste it over a larger image.
--->
<!--- Create a ColdFusion image from an existing JPEG file and name it "myImage1". --->
<cfimage source="../cfdocs/images/artgallery/jeff05.jpg" name="myImage1">
<!--- Create a ColdFusion image from an existing JPEG file and name it "myImage2". --->
<cfimage source="../cfdocs/images/artgallery/maxwell01.jpg" name="myImage2">
<!--- Copy a rectangular region of myImage1. --->
<cfset resImage = ImageCopy(myImage1,1,1,50,50)>
<!--- Paste the rectangular area over myImage2. --->
<cfset ImagePaste(myImage2,resImage,100,100)>
<!--- Write the second ColdFusion image to result.jpg. --->
<cfimage source="#myImage2#" action="write" destination="test_myImage.jpg" overwrite="yes">
<!--- Display the two source images and the new image. --->
<img src="../cfdocs/images/artgallery/jeff05.jpg"/>
<img src="../cfdocs/images/artgallery/maxwell01.jpg"/>
<img src="test_myImage.jpg"/>
```

ImageRead

説明

ディスク上またはメモリ内のソースパス名または URL を読み込み、ColdFusion イメージを作成します。

戻り値

ColdFusion イメージ

カテゴリ

Image 関数

関数のシンタックス

```
ImageRead(path)
```

履歴

ColdFusion 8: この関数が追加されました。

関連項目

cfimage、ImageNew、ImageReadBase64、ImageWrite、IsImageFile

パラメータ

パラメータ	説明
path	必須です。ソースイメージのディスク上またはメモリ内のパス名または URL です。

使用方法

ImageRead 関数は、cfimage の read アクションと同じオペレーションを実行します。ただし、cfscript タグ内で cfimage タグを使用して ColdFusion イメージ変数を読み込んだり、イメージ変数を作成したりすることはできません。ColdFusion イメージを読み込むには、cfscript タグ内で ImageRead 関数を使用します。

次の例では、イメージファイル aiden01.jpg を myImage という変数に読み込み、そのイメージをブラウザに表示します。

```
<cfset myImage=ImageRead("../cfdocs/images/artgallery/aiden01.jpg")>
<cfimage action="writeToBrowser" source="#myImage#">
```

有効なイメージ形式については、332 ページの「cfimage」にあるサポートされているイメージファイル形式のリストを参照してください。ColdFusion アプリケーションがデプロイされているサーバーで読み込み可能な形式のリストを取得するには、GetReadableImageFormats 関数を使用します。

例

```
<!--- This example shows how to create a script that reads an image from a URL. --->
<cfscript>
    myImage=ImageRead("http://www.google.com/images/logo.gif");
    ImageWrite(myImage,"google-logo.gif");
</cfscript>
<p>This image has been downloaded by ColdFusion:</p>
<img src="google-logo.gif">
<p>This is the original image:</p>
<img src="http://www.google.com/images/logo.gif"/>
```

ImageReadBase64

説明

Base64 文字列から ColdFusion イメージを作成します。

戻り値

ColdFusion イメージ

カテゴリ

Image 関数

関数のシンタックス

```
ImageReadBase64(string)
```

関連項目

ImageNew、ImageRead、ImageWrite、ImageWriteBase64、BinaryDecode、BinaryEncode、IsImageFile

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
string	必須です。ColdFusion 変数または Base64 文字列です。

使用方法

Base64 は、バイナリデータを印刷可能文字列として表現する方法です。ImageReadBase64 関数は Base64 文字列を入力パラメータとして受け取り、その文字列からイメージを作成します。

HTML の <img src> タグに Base64 イメージを渡すためのヘッダは、Base64 文字列に含まれていても、含まれていなくてもかまいません。

この関数を使用すると、Base64 文字列を ColdFusion イメージに変換できます。一部のデータベースでは、BLOB データとしてではなく Base64 文字列としてイメージが格納されます。データベースに対してクエリーを実行し、ImageReadBase64 関数を使用することで、文字列を ColdFusion イメージに変換できます。これにより、BinaryEncode 関数と BinaryDecode 関数を使用してイメージを変換する手順を省略できます。

RSS (Really Simple Syndication) フィードの場合、イメージは Base64 文字列として XML ファイルに埋め込まれて転送されます。ColdFusion でこれらのイメージを読み込むには、ImageReadBase64 関数を使用します。

例

例 1

```
<!--- この例では、ヘッダを含む Base64 文字列を読み込みます。--->

<cfset myImage = ImageReadBase64(".............")>
<cfimage source="#myImage#" destination="test_my64.jpeg" action="write">
```

例 2

```
<!--- この例では、ヘッダを含まない Base64 文字列を読み込みます。--->

<cfset myImage = ImageReadBase64("/9j/4AAQSkZJRgABAQA.............")>
<cfimage source="#myImage#" destination="test_my64.jpeg" action="write">
```

ImageResize

説明

ColdFusion イメージのサイズを変更します。

戻り値

なし

カテゴリ

Image 関数

関数のシンタックス

```
ImageResize(name, width, height [, interpolation, blurFactor])
```

関連項目

cfimage、ImageSetAntialiasing、ImageScaleToFit、IsImageFile

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。
width	必須です。ColdFusion イメージの新しい幅です。 この値を空にした場合、イメージの幅は高さを基準として比例的に計算されます。

パラメータ	説明
height	必須です。ColdFusion イメージの新しい高さです。 この値を空にした場合、イメージの高さは幅を基準として比例的に計算されます。
interpolation	オプション。リサンプリングに使用する補間方式です。名前 (hamming など)、イメージ品質 (mediumQuality など)、またはパフォーマンス (highestPerformance など) を基準に特定の補間アルゴリズムを指定できます。有効な値は、次のとおりです。 • highestQuality (デフォルト) • highQuality • mediumQuality • highestPerformance • highPerformance • mediumPerformance • nearest • bilinear • bicubic • bessel • blackman • hamming • hanning • hermite • lanczos • mitchell • quadratic 詳細については、「使用方法」の「補間アルゴリズム」を参照してください。
blurFactor	オプション。リサンプリングに使用するぼかし係数です。ぼかし係数を大きくすると、ぼかしの度合いが高くなります (イメージのサイズ変更に要する時間も長くなります)。この値は、1 ～ 10 の間である必要があります。

使用方法

この関数を使用すると、イメージを拡大したり、サムネールイメージを作成できます。

高さまたは幅をピクセル単位で指定するには、「100」のように整数を入力します。高さまたは幅をパーセントで指定するには、「50%」のように数値とパーセント記号を続けて入力します。

イメージのサイズを 1 方向にのみ (たとえば、高さのみを) 変更するには、height の値を指定して width の値を空 ("") にします。この場合、イメージの幅は高さを基準として比例的に計算されます。

レンダリングされるイメージの質を高めるには、ImageSetAntialiasing 関数を使用します。

補間アルゴリズム

補間アルゴリズムを指定すると、イメージのリサンプリング方法を細かく調整できます。このアルゴリズムを指定することで、イメージの品質を優先するか、パフォーマンスを優先するかが決定されます。一般に、イメージの品質を高くすると、パフォーマンスは低下します。品質とパフォーマンスは、ソースイメージの種類とファイルサイズによって異なります。次の表に、平均的なテスト結果に基づく各アルゴリズムの説明と、一般的なアルゴリズム名との対応関係を示します。

値	対応する名前	説明
highestQuality (デフォルト)	lanczos	イメージ品質 - 最高 / パフォーマンス - 低い
highQuality、mediumPerformance	mitchell、quadratic	イメージ品質 - 高い / パフォーマンス - やや高い
mediumQuality、highPerformance	hamming、hanning、hermite	イメージ品質 - 中程度 / パフォーマンス - 中程度
	blackman、bessel	イメージ品質 - やや低い / パフォーマンス - 高い
highestPerformance	nearest、bicubic、bilinear	イメージ品質 - 低い / パフォーマンス - 最高

例

```
<!--- This example shows how to resize an image to 50% of original size and resize it proportionately to
the new width. Notice that the height is blank.--->
<cfset myImage=ImageNew("http://www.google.com/images/logo_sm.gif")>
<cfset ImageResize(myImage,"50%","","blackman",2)>
<!--- Save the modified image to a file called "test_myImage.jpeg" and display the image in a browser. --->
<cfimage source="#myImage#" action="write" destination="test_myImage.jpeg" overwrite="yes">
<!--- Display the source image and the thumbnail image. --->
<img src="http://www.google.com/images/logo_sm.gif"/>
<img src="test_myImage.jpeg"/>
```

ImageRotate

説明

指定した座標を中心として ColdFusion イメージを指定した角度だけ回転させます。

戻り値

なし

カテゴリ

Image 関数

関数のシンタックス

```
ImageRotate(name, angle [, x, y, interpolation])
```

関連項目

cfimage、ImageFlip、ImageSetAntialiasing、IsImageFile

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。
angle	必須です。回転する角度です (単位 : 度数)。
x	オプション。回転の中心点の x 座標です。デフォルト値は 2 です。
y	オプション。回転の中心点の y 座標です。デフォルト値は 2 です。
interpolation	オプション。補間のタイプです。 • nearest: ニアレストネイバー法の補間処理を適用します。他の補間方式よりもイメージの品質は低くなりますが、処理は最も速くなります (デフォルト)。 • bilinear: バイリニア法の補間処理を適用します。イメージの品質はデフォルトよりもやや高くなりますが、処理は遅くなります。 • bicubic: バイキュービック法の補間処理を適用します。一般に、イメージの品質は最も高くなりますが、処理は最も遅くなります。

使用方法

座標を指定する場合は、x と y を両方とも指定します。指定しない場合は、両方とも指定しないようにします。x 座標と y 座標を指定しない場合は、イメージの中心点 (デフォルトの位置) が回転の中心点になります。レンダリングされるイメージの質を高めるには、ImageSetAntialiasing 関数を使用します。

例

例 1

```
<!--- This example shows how to rotate an image by 10 degrees. --->
<!--- Create a ColdFusion image from an existing JPEG file. --->
<cfset myImage=ImageNew("../cfdocs/images/artgallery/jeff05.jpg")>
<cfset ImageSetAntialiasing(myImage,"on")>
<!--- Rotate the image by 10 degrees. --->
<cfset ImageRotate(myImage,10)>
<cfimage source="#myImage#" action="writeToBrowser">
```

例 2

```
<!--- This example shows how to rotate an image by 10 degrees and change the interpolation to bicubic for
higher resolution. The image is rotated at the (10,90) coordinates. --->
<cfimage source="../cfdocs/images/artgallery/jeff05.jpg" name="myImage">
<cfset ImageRotate(myImage,10,10,90,"bicubic")>
<cfimage source="#myImage#" destination="testMyImage.jpeg" action="write" overwrite="yes">
<img src="../cfdocs/images/artgallery/jeff05.jpg">
<img src="testMyImage.jpeg">
```

ImageRotateDrawingAxis

説明

指定した座標を中心として、以後に ColdFusion イメージ上に描画されるすべての図形を、指定した角度だけ回転させます。

戻り値

ColdFusion イメージ

カテゴリ
Image 関数

関数のシンタックス
```
ImageRotateDrawingAxis(name, angle [, x, y])
```

関連項目
ImageRotate、ImageSetAntialiasing、ImageSetBackgroundColor、ImageSetDrawingColor、ImageSetDrawingStroke、ImageSetDrawingTransparency、ImageShearDrawingAxis、IsImageFile

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。
angle	必須です。回転する角度です (単位 : 度数)。
x	オプション。回転の中心点の x 座標です。デフォルト値は 0 です。
y	オプション。回転の中心点の y 座標です。デフォルト値は 0 です。

使用方法
デフォルトの原点は 0,0 です。元の描画軸に戻すには、同じ (x,y) パラメータを指定し、回転させた角度だけ逆に回転させます。レンダリングされるイメージの質を高めるには、ImageSetAntialiasing 関数を使用します。

例
```
<!--- This example shows how to create an image with three shapes drawn on the same axis. --->
<!--- Use ImageNew to create a 300x300-pixel image. --->
<cfset myImage=ImageNew("",300,300)>
<!--- Turn on antialiasing to improve image quality. --->
<cfset ImageSetAntialiasing(myImage,"on")>
<!--- Set the drawing axis to 30 degrees and the point of rotation at (10,10). --->
<cfset ImageRotateDrawingAxis(myImage,30,10,10)>
<!--- Set the drawing color to blue. --->
<cfset ImageSetDrawingColor(myImage,"blue")>
<!--- Draw three shapes with the same color and drawing axis. --->
<cfset ImageDrawRect(myImage,150,10,10,150,"yes")>
<cfset ImageDrawOval(myImage,200,40,45,65,"yes")>
<cfset ImageDrawRect(myImage,275,10,10,150,"yes")>
<cfimage source="#myImage#" action="writeToBrowser">
```

ImageScaleToFit

説明
縦横比を維持したままイメージのサイズを変更します。

戻り値
なし

カテゴリ

Image 関数

関数のシンタックス

```
ImageScaleToFit(name, fitWidth, fitHeight [, interpolation , blurFactor])
```

関連項目

cfimage、ImageResize、ImageSetAntialiasing、IsImageFile

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。
fitWidth	必須です。境界ボックスの幅をピクセル単位で指定します。整数を指定できます。fitHeight が指定されている場合は、空の文字列 ("") を指定することもできます。詳細については、「使用方法」を参照してください。

パラメータ	説明
fitHeight	必須です。境界ボックスの高さをピクセル単位で指定します。整数を指定できます。fitWidth が指定されている場合は、空の文字列 ("") を指定することもできます。詳細については、「使用方法」を参照してください。
interpolation	オプション。リサンプリングに使用する補間方式です。名前 (hamming など)、イメージ品質 (mediumQuality など)、またはパフォーマンス (highestPerformance など) を基準に特定の補間アルゴリズムを指定できます。有効な値は、次のとおりです。 • highestQuality (デフォルト) • highQuality • mediumQuality • highestPerformance • highPerformance • mediumPerformance • nearest • bilinear • bicubic • bessel • blackman • hamming • hanning • hermite • lanczos • mitchell • quadratic 詳細については、「使用方法」の「補間アルゴリズム」を参照してください。
blurFactor	オプション。リサンプリングに使用するぼかし係数です。ぼかし係数を大きくすると、ぼかしの度合いが高くなります (イメージのサイズ変更に要する時間も長くなります)。有効な値は 1 ～ 10 です。

使用方法

このオペレーションは、縦横比を維持したままイメージのサイズを変更したりサムネールイメージを作成する場合に使用します。fitWidth パラメータと FitHeight パラメータを指定します。fitWidth または FitHeight のいずれか一方を指定した場合は、次のように他方のパラメータを空の文字列にできます。

```
<cfset ImageScaleToFit(myImage,100,"")>
```

この例では、ImageScaleToFit 関数を使用して 100×100 ピクセルの正方形に合うようにイメージのサイズが変更されます。その結果、イメージの幅は 100 ピクセルになり、高さは 100 ピクセル以下になります。たとえば、ソースイメージが 400×200 ピクセルの場合、結果のイメージは 100×50 ピクセルになります。

同様に、ImageScaleToFit 関数の fitHeight パラメータで整数を指定し、fitWidth パラメータで空の文字列を指定すると、イメージの高さが fitHeight パラメータと等しくなり、イメージの幅は比例的に決定されます。

```
<cfset ImageScaleToFit(myImage,"",100)>
```

この例では、400×200 ピクセルのソースイメージが 200×100 ピクセルのサイズに変更され、200×400 ピクセルのイメージは 50×100 ピクセルのサイズに変更されます。

ImageScaleToFit 関数の fitWidth パラメータと fitHeight パラメータの両方を設定すると、結果のイメージの幅は fitWidth 以下となり、高さも fitHeight 以下となるように比例的にサイズが決定されます。

```
<cfset ImageScaleToFit(myImage,100,200)>
```

この例では、400×200 ピクセルのソースイメージが 100×50 ピクセルのサイズに変更され、200×400 ピクセルのソースイメージは 100×200 ピクセルのサイズに変更されます。

レンダリングされるイメージの質を高めるには、ImageSetAntialiasing 関数を使用します。

例

```
<!--- This example shows how to resize an image to fit a 100x100-pixel square while maintaining the aspect
ratio. --->
<!--- Create a ColdFusion image from an existing JPEG file. --->
<cfimage source="../cfdocs/images/artgallery/jeff05.jpg" name="myImage">
<!--- Turn on antialiasing to improve image quality. --->
<cfset ImageSetAntialiasing(myImage,"on")>
<cfset ImageScaleToFit(myImage,100,"","lanczos")>
<!--- Display the modified image in a browser. --->
<cfimage source="#myImage#" action="writeToBrowser">
```

ImageSetAntialiasing

説明

グラフィックをレンダリングするときのアンチエイリアス処理を有効または無効にします。

戻り値

なし

カテゴリ

Image 関数

関数のシンタックス

```
ImageSetAntialiasing(name [, antialias])
```

関連項目

ImageRotateDrawingAxis、ImageSetBackgroundColor、ImageSetDrawingColor、ImageSetDrawingStroke、ImageSetDrawingTransparency、IsImageFile

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。
antialias	オプション。アンチエイリアス処理のスイッチです。 • on (デフォルト) • off

使用方法

ImageSetAntialiasing 関数を使用すると、イメージ内に図形やテキストを描画するときのアンチエイリアス処理を有効または無効にすることができます。アンチエイリアスとは、ジャギーを滑らかにするための技術です。ImageDrawRoundRect や ImageRotate などのイメージ関数を使用するときにアンチエイリアスを有効にすると、レンダリングされるイメージの質を高めることができます。アンチエイリアスを有効にするとパフォーマンスは低下します。

例

例 1

```
<!--- This example shows how to turn off antialiasing for a ColdFusion image. --->
<!--- Create a ColdFusion image from an existing JPEG file. --->
<cfset myImage=ImageNew("../cfdocs/images/artgallery/elecia02.jpg")>
<!--- Turn off antialiasing.--->
<cfset ImageSetAntialiasing(myImage,"off")>
<!--- Display the modified image in a browser. --->
<cfimage source="#myImage#" action="writeToBrowser">
```

例 2

```
<!--- This example shows how to turn on antialiasing to improve the output of the ImageDrawRoundRect
function. --->
<!--- Create a 200x200-pixel image. --->
<cfset myImage=ImageNew("",200,200)>
<!--- Set the drawing color for the image to blue. --->
<cfset ImageSetDrawingColor(myImage,"blue")>
<!--- Turn on antialiasing. --->
<cfset ImageSetAntialiasing(myImage)>
<!--- Draw a blue filled square with round corners of arcWidth=10 and arcHeight=2. --->
<cfset ImageDrawRoundRect(myImage,5,5,190,190,"yes",10,2)>
<!--- Rotate the image 30 degrees. --->
<cfset ImageRotate(myImage,30)>
<!--- Display the image in a browser. --->
<cfimage source="#myImage#" action="writeToBrowser">
```

ImageSetBackgroundColor

説明

ColdFusion イメージの背景色を設定します。背景色は特定の領域を消去する場合に使用します。背景色の設定は、それ以後の ImageClearRect の呼び出しに対してのみ影響します。

戻り値

なし

カテゴリ

Image 関数

関数のシンタックス

```
ImageSetBackgroundColor(name, color)
```

関連項目

ImageClearRect、ImageSetAntialiasing、IsImageFile

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。
color	必須です。次の背景色を指定できます。 • RGB カラーの 16 進数値。たとえば、白の場合は ##FFFFFF または FFFFFF と指定します。 • 色の文字列値 ("black"、"red"、"green" など)。描画色のデフォルト値は "black" です。 • (R,G,B) 値を示す 3 つの数値のリスト。各値の範囲は 0 ～ 255 です。

使用方法

色の値を文字列で指定する場合は、サポートされているカラー名を使用します。有効な HTML カラー名については、332 ページの「cfimage」を参照してください。16 進数値を入力するには、"##xxxxxx" または "xxxxxx" という形式を使用します (x は 0 ～ 9 または A ～ F です)。シャープ記号 (#) は 2 つ使用するか、または使用しないでください。

イメージの矩形領域を消去して、その領域を指定した色に設定するには、この関数と ImageClearRect 関数を使用します。

例

```
<!--- This example shows how to set the background color, and then draw a rectangle on an image filled with
that color. --->
<!--- Create a ColdFusion image from an existing JPEG file. --->
<cfimage name="myImage" source="../cfdocs/images/artgallery/maxwell01.jpg">
<!--- Turn on antialiasing to improve image quality. --->
<cfset ImageSetAntialiasing(myImage)>
<!--- Set the background color to magenta. --->
<cfset ImageSetBackgroundColor(myImage,"magenta")>
<!--- Clear the rectangle specified on myImage with the background color specified for the image. --->
<cfset ImageClearRect(myImage,36,45,100,100)>
<!--- Display the modified image in a browser. --->
<cfimage source="#myImage#" action="writeToBrowser">
```

ImageSetDrawingColor

説明

ColdFusion イメージの現在の描画色を設定します。以後のグラフィックオペレーションでは、この色が使用されます。

戻り値

なし

カテゴリ

Image 関数

関数のシンタックス

```
ImageSetDrawingColor(name, color)
```

関連項目

ImageSetAntialiasing、ImageSetBackgroundColor、ImageSetDrawingStroke、ImageSetDrawingTransparency、IsImageFile

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。
color	必須です。次の方法で描画色を指定します。 • RGB カラーの 16 進数値。たとえば、白の場合は "##FFFFFF" または "FFFFFF" を指定します。 • 色の文字列値 ("black"、"red"、"green" など)。描画色のデフォルト値は "black" です。 • (R,G,B) 値を示す 3 つの数値のリスト。各値の範囲は 0 ～ 255 です。

使用方法

ImageSetDrawingColor を呼び出すと、その後に ColdFusion イメージ上に描画されるオブジェクトの色を制御できます。たとえば、この関数を使用して描画色を赤に設定すると、その後に描画される円、四角形、線などのオブジェクトをすべてその色で描画することができます。

色の値を文字列で指定する場合は、サポートされているカラー名を使用します。有効な HTML カラー名については、332 ページの「cfimage」を参照してください。16 進数値で指定する場合は、"##xxxxxx" または "xxxxxx" という形式を使用します。ここで、x は 0 ～ 9 または A ～ F です。シャープ記号 (#) は 2 つ使用するか、または使用しないでください。

例

```
<!--- This example shows how to set the drawing color and draw several objects in that color. --->
<!--- Use the ImageNew function to create a ColdFusion image. --->
<cfset myImage=ImageNew("",200,300)>
<!--- Turn on antialiasing to improve image quality. --->
<cfset ImageSetAntialiasing(myImage)>
<!--- Set the drawing color to pink. --->
<cfset ImageSetDrawingColor(myImage,"pink")>
<!--- Draw a pink line. --->
<cfset ImageDrawLine(myImage,1,1,200,300)>
<!--- Draw a pink oval. --->
<cfset ImageDrawOval(myImage,40,50,80,40)>
<!--- Draw another pink oval. --->
<cfset ImageDrawOval(myImage,15,180,80,40)>
<!--- Draw a pink rectangle. --->
<cfset ImageDrawRect(myImage,100,100,45,65)>
<!--- Display the image in a browser. --->
<cfimage source="#myImage#" action="writeToBrowser">
```

ImageSetDrawingStroke

説明

この関数を呼び出した後に ColdFusion イメージ内で描画される点および線の描画ストロークを設定します。

戻り値
なし

カテゴリ
Image 関数

関数のシンタックス
```
ImageSetDrawingStroke(name [, attributeCollection])
```

関連項目
ImageDrawText、ImageSetAntialiasing、ImageSetBackgroundColor、ImageSetDrawingColor、ImageSetDrawingTransparency、IsImageFile

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。
attributeCollection	オプション。線の属性を指定する構造体です。「使用方法」を参照してください。

使用方法
ImageSetDrawingStroke を呼び出すと、その後に ColdFusion イメージ上に描画されるオブジェクトの線の属性を制御できます。たとえば、この関数を使用して描画ストロークをダッシュパターンに設定すると、その後に描画される矩形、楕円、線などのオブジェクトをすべてダッシュパターンで描画することができます。

空の属性構造体を渡した場合、または属性構造体を渡さなかった場合は、デフォルトの描画ストロークにリセットされます。

attributeCollection

要素	説明
width	ペンの太さです。この値は、ペンの軌道に対して垂直に計測されます。
endcaps	閉じられていないサブパスやダッシュセグメントの終端に適用される装飾です。閉じセグメントがない限り、開始点と終了点が同一であるサブパスは閉じられていないと見なされます。 • butt • round • square
lineJoins	線の結合点の種類です。 • bevel • miter • join

要素	説明
miterLimit	装飾の種類が留め継ぎ (miter) に設定された結合点をトリミングするときの限度です (lineJoins = "miter" の場合にのみ使用します)。ストローク幅と留め継ぎ幅の比が miterLimit の値よりも大きい場合は、結合点がトリミングされます。留め継ぎ幅とは、留め継ぎの対角線の長さです。つまり、交点の内角と外角の間の距離です。2 本の線によって形成される角度が小さくなるにつれて、留め継ぎ幅は長くなり、交点の角度は鋭くなります。デフォルト値は 10.0 です。この場合は、11 度より小さい角がすべてトリミングされます。留め継ぎがトリミングされると、結合点の装飾がベベル (bevel) に変換されます。
dashArray	ダッシュパターンを示す数値の配列です。たとえば、dashArray を (8,4) に設定した場合は、8 ピクセルの実線と 4 ピクセルの空白を交互に描画するダッシュパターンになります。
dash_phases	ダッシュパターンに適用されるオフセットです。たとえば、dash_phase を 2 に設定して、dashArray を (8,4) に設定した場合は、6 ピクセルの実線、4 ピクセルの空白、8 ピクセルの実線、4 ピクセルの空白という具合にダッシュパターンが描画されます。

例

例 1

```
<!--- This example shows how to create an attribute collection for the ImageSetDrawingStroke function and
draws a line with those attributes.
    --->
<!--- Use the ImageNew function to create a ColdFusion image. --->
<cfset myImage=ImageNew("",200,200)>
<!--- Create an attribute collection to pass to the ImageSetDrawingStroke function. Create a stroke that is
10-pixels wide, has round endcaps, and has a dash pattern of (8,4). --->
<cfset attr = StructNew()>
<cfset attr.width = 2>
<cfset attr.endcaps = "round">
<cfset dashPattern = ArrayNew(1)>
<cfset dashPattern[1] = 8>
<cfset dashPattern[2] = 4>
<cfset attr.dashArray = dashPattern>
<!--- Apply the attribute collection to the ImageSetDrawingStroke function for the image. --->
<cfset ImageSetDrawingStroke(myImage,attr)>
<!--- Draw a line on the ColdFusion image with the drawing stroke attributes. --->
<cfset ImageDrawLine(myImage,20,20,40,150)>
<!--- Display the image in a browser. --->
<cfimage source="#myImage#" action="writeToBrowser">
```

例 2

```
<!--- Use the ImageNew function to create a ColdFusion image. --->
<cfset myImage=ImageNew("",500,500)>
<!-- Set the drawing color of the image to cyan. --->
<cfset ImageSetDrawingColor(myImage,"cyan")>
<!--- Draw a line from (30,40) to (200,190). --->
<cfset ImageDrawLine(myImage,30,30,200,200)>
<!--- Create the attribute collection for the drawing stroke. --->
<cfset attr = StructNew()>
<cfset attr.width = 1>
<cfset attr.endcaps = "round">
<cfset dashPattern = ArrayNew(1)>
<cfset dashPattern[1] = 3>
<cfset dashPattern[2] = 4>
<cfset dashPattern[3] = 8>
<cfset attr.dashArray = dashPattern>
<!--- Pass the attribute collection as an argument to the set drawing stroke
function. --->
<cfset ImageSetDrawingStroke(myImage,attr)>
<!-- Set the drawing color of the image to yellow. --->
<cfset ImageSetDrawingColor(myImage,"yellow")>
<!--- Draw a rectangle with the drawing stroke specified. --->
<cfset ImageDrawRect(myImage,200,50,210,200)>
<!-- Reset the drawing stroke. -->
<cfset ImageSetDrawingStroke(myImage)>
<!--- Draw a green quadratic curve. --->
<cfset ImageSetDrawingColor(myImage,"green")>
<cfset ImageDrawQuadraticCurve(myImage,120,320,5,15,380,280)>
<!--- Display the image in a browser. --->
<cfimage source="#myImage#" action="writeToBrowser">
```

ImageSetDrawingTransparency

説明
描画関数の透明度を指定します。

戻り値
なし

カテゴリ
Image 関数

関数のシンタックス
```
ImageSetDrawingTransparency(name, percent)
```

関連項目
ImageSetAntialiasing、ImageSetBackgroundColor、ImageSetDrawingColor、ImageSetDrawingStroke、IsImageFile

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。
percent	必須です。透明度のパーセントです。 • 0 = 不透明 • 100 = 透明 10 進値で指定します。

使用方法

デフォルトで描画されるイメージは不透明です。この関数は、透かしや半透明のイメージを作成する場合に使用します。レンダリングされるイメージの質を高めるには、ImageSetAntialiasing 関数を使用します。

例

例 1

```
<!--- This example shows how to draw semitransparent text over an image.
    --->
<!--- Create a ColdFusion image from an existing JPEG file. --->
<cfimage source="../cfdocs/images/artgallery/austin01.jpg" name="myImage">
<!--- Turn on antialiasing to improve image quality. --->
<cfset ImageSetAntialiasing(myImage)>
<!--- Set the drawing transparency to 40%. --->
<cfset ImageSetDrawingTransparency(myImage,40)>
<!--- Set the text drawing attributes. --->
<cfset attr = StructNew()>
<cfset attr.size = 40>
<cfset attr.style = "bold">
<!--- Specify the text string and the location of the text on the image.
    --->
<cfset ImageDrawText(myImage,"SOLD!",40,100,attr)>
<!--- Display the image in a browser. --->
<cfimage source="#myImage#" action="writeToBrowser">
```

例 2

```
<!--- This example shows how to create a watermark from the a JPEG file.
--->
<!--- Create a ColdFusion image from a JPEG file. --->
<cfimage source="../cfdocs/getting_started/photos/somewhere.jpg" name="myImage" action="read">
<!--- Set the drawing transparency to 75%. --->
<cfset ImageSetDrawingTransparency(myImage,75)>
<!--- Create a ColdFusion image from a picture in the cfartgallery. --->
<cfimage source="../cfdocs/images/artgallery/raquel05.jpg" name="myImage2" action="read">
<!--- Set the drawing transparency to 30%. --->
<cfset ImagesetDrawingTransparency(myImage,30)>
<!--- Paste the ColdFusion log over the picture at coordinates (0,0).--->
<cfset ImagePaste(myImage,myImage2,0,0)>
<!--- Display the two source images and the result. --->
<cfimage source="#myImage#" destination="watermark.jpg" action="write" overwrite="yes">
<img src="../cfdocs/getting_started/photos/somewhere.jpg">
<Img src="../cfdocs/images/artgallery/raquel05.jpg">
<img src="watermark.jpg">
```

例 3

```
<!--- This code creates a ColdFusion image to be used as a watermark. --->
<cfimage action="read" name="logo" source="../cfdocs/getting_started/photos/somewhere.jpg">
<cfset imageGrayscale(logo)>
<cfset imageRotate(logo,45)>
<!--- This code creates the ColdFusion image to be used as the base image.
--->
<cfimage action="read" source="../cfdocs/images/artgallery/raquel05.jpg" name="baseImage">
<!--- This code sets the drawing transparency for the base image to 80%.
--->
<cfset ImageSetDrawingTransparency(baseImage,80)>
<!--- This code pastes the watermark image onto the base image at the coordinates (0,0). --->
<cfset ImagePaste(baseImage,logo,0,0)>
<!--- This code writes the result to a file. --->
<cfimage action="write" source="#baseImage#" destination="abc_watermark.jpg" overwrite="yes">
<!--- This code displays the image used as a watermark and the result. --->
<img src="../cfdocs/getting_started/photos/somewhere.jpg"/>
<img src="abc_watermark.jpg"/>
```

ImageSharpen

説明
アンシャープマスクフィルタを使用して、ColdFusion イメージにシャープ加工を適用します。

戻り値
なし

カテゴリ
Image 関数

関数のシンタックス
```
ImageSharpen(name [, gain])
```

関連項目
ImageBlur、ImageSetAntialiasing、IsImageFile

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。
gain	オプション。有効な値の範囲は -1 ～ 2 です。 gain の値は整数または実数で指定できます。デフォルト値は 1.0 です。 この値は、イメージをぼかすか、シャープにするかを決定します。 • 0 未満の場合、イメージはシャープになります。 • 0 の場合、効果はありません。 • 0 より大きい場合、イメージはぼかされます。

使用方法
この関数は、写真の輪郭をシャープにする場合に使用します。レンダリングされるイメージの質を高めるには、ImageSetAntialiasing 関数を使用します。

例
```
<!--- Create a ColdFusion image from an existing JPEG file. --->
<cfimage source="../cfdocs/images/artgallery/paul01.jpg" name="myImage">
<!--- Turn on antialiasing to improve image quality. --->
<cfset ImageSetAntialiasing(myImage,"on")>
<!--- Sharpen myImage by 2. --->
<cfset ImageSharpen(myImage,2)>
<!--- Write the sharpened image to a file. --->
<cfimage source="#myImage#" action="write" destination="test_myImage.jpg" overwrite="yes">
<!--- Display the original and the sharpened images. --->
<img src="../cfdocs/images/artgallery/paul01.jpg"/>
<img src="test_myImage.jpg"/>
```

ImageShear

説明
水平または垂直方向にイメージを歪めます。補間オブジェクトを使用して、ターゲットのピクセル (x',y') ごとにサブピクセルの位置 (x',y') のソース値が構築され、ターゲットに書き込まれます。

戻り値
なし

カテゴリ
Image 関数

関数のシンタックス
```
ImageShear(name, shear [, direction, interpolation])
```

関連項目
ImageSetAntialiasing、ImageShearDrawingAxis、IsImageFile

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。

パラメータ	説明
shear	必須です。歪みの値です。座標は整数または実数で指定できます。
direction	オプション。歪みの方向です。 • horizontal (デフォルト) • vertical
interpolation	オプション。補間のタイプです。 • nearest: ニアレストネイバー法の補間処理を適用します。他の補間方式よりもイメージの品質は低くなりますが、処理は最も速くなります (デフォルト)。 • bilinear: バイリニア法の補間処理を適用します。イメージの品質はデフォルトよりもやや高くなりますが、処理は遅くなります。 • bicubic: バイキュービック法の補間処理を適用します。一般に、イメージの品質は最も高くなりますが、処理は最も遅くなります。

使用方法
この関数は、イメージを歪める場合に使用します。

direction パラメータが horizontal に設定されている場合は、x' = (x - y*shear)、y' = y となります。

direction パラメータが vertical に設定されている場合は、x' = x、y' = (y - x*shear) となります。

レンダリングされるイメージの質を高めるには、ImageSetAntialiasing 関数を使用します。

例
```
<!--- This example shows how to shear an image. --->
<!--- Create a ColdFusion image from an existing JPEG file. --->
<cfimage source="../cfdocs/images/artgallery/paul03.jpg" name="myImage">
<!--- Turn on antialiasing to improve image quality. --->
<cfset ImageSetAntialiasing(myImage,"on")>
<!--- Shear the image by a factor of 1 on a horizontal axis. --->
<cfset ImageShear(myImage,1,"horizontal")>
<!--- Display the image in a browser. --->
<cfimage source="#myImage#" action="writeToBrowser">
```

ImageShearDrawingAxis

説明
描画キャンバスを歪めます。

戻り値
なし

カテゴリ
Image 関数

関数のシンタックス
```
ImageShearDrawingAxis(name, shx, shy)
```

関連項目
ImageRotateDrawingAxis、ImageSetAntialiasing、ImageShear、IsImageFile

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。
shx	必須です。y 座標の関数として、座標を正の x 軸方向にシフトするための乗数です。
shy	必須です。x 座標の関数として、座標を正の y 軸方向にシフトするための乗数です。

使用方法
補間オブジェクトを使用して、ターゲットのピクセル (x',y') ごとにサブピクセルの位置 (x',y') のソース値が構築され、ターゲットに書き込まれます。

direction パラメータが horizontal に設定されている場合は、x' = (x - y*shear)、y' = y となります。

direction パラメータが vertical に設定されている場合は、x' = x、y' = (y - x*shear) となります。

レンダリングされるイメージの質を高めるには、ImageSetAntialiasing 関数を使用します。

例
```
<!--- Create a ColdFusion image from an existing JPEG file. --->
<cfimage source="../cfdocs/images/artgallery/paul03.jpg" name="myImage">
<!--- Turn on antialiasing to improve image quality. --->
<cfset ImageSetAntialiasing(myImage,"on")>
<!--- Set the shear drawing axis. --->
<cfset ImageShearDrawingAxis(myImage,0.5,0.5)>
<!--- Draw a rectangle on the image with the shear settings. --->
<cfset ImageDrawRect(myImage,50,50,50,50)>
<!--- Display the image in a browser. --->
<cfimage source="#myImage#" action="writeToBrowser">
```

ImageTranslate

説明
平面上の新しい位置にイメージをコピーします。

戻り値
なし

カテゴリ
Image 関数

関数のシンタックス
```
ImageTranslate(name, xTrans, yTrans [, interpolation])
```

関連項目
ImageSetAntialiasing、ImageShear、ImageTranslateDrawingAxis、IsImageFile

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。
xTrans	必須です。x 方向の移動距離です。
yTrans	必須です。y 方向の移動距離です。
interpolation	オプション。リサンプリングに使用する補間のタイプです。 • nearest: ニアレストネイバー法の補間処理を適用します。他の補間方式よりもイメージの品質は低くなりますが、処理は最も速くなります (デフォルト)。 • bilinear: バイリニア法の補間処理を適用します。イメージの品質はデフォルトよりもやや高くなりますが、処理は遅くなります。 • bicubic: バイキュービック法の補間処理を適用します。一般に、イメージの品質は最も高くなりますが、処理は最も遅くなります。

使用方法

補間オブジェクトを使用して、ターゲットのピクセル (x, y) ごとにサブピクセルの位置 (x - xTrans, y - yTrans) のソース値が構築され、ターゲットに書き込まれます。xTrans と yTrans の両方が整数である場合は、ソースイメージをラップして座標面におけるイメージの位置を変更します。

レンダリングされるイメージの質を高めるには、ImageSetAntialiasing 関数を使用します。

例

```
<!--- Create a ColdFusion image from an existing JPEG file. --->
<cfimage source="../cfdocs/images/artgallery/aiden01.jpg" name="myImage">
<!--- Turn on antialiasing to improve image quality. --->
<cfset ImageSetAntialiasing(myImage,"on")>
<!--- Offset the image's position to (20,10). --->
<cfset ImageTranslate(myImage,20,10)>
<!--- Display the offset image in a browser. --->
<cfimage source="#myImage#" action="writeToBrowser">
```

ImageTranslateDrawingAxis

説明

イメージコンテキストの原点を、現在の座標系の点 (x,y) に移動します。新しい原点が元の座標系の点 (x,y) に対応するようにイメージコンテキストを変更します。

戻り値

なし

カテゴリ

Image 関数

関数のシンタックス

```
ImageTranslateDrawingAxis(name, x, y)
```

関連項目
ImageSetAntialiasing、ImageSetDrawingColor、ImageSetDrawingStroke、ImageSetDrawingTransparency、ImageShearDrawingAxis、ImageTranslate、IsImageFile

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。
x	x 座標です。
y	y 座標です。

使用方法
この関数を呼び出した後のレンダリングオペレーションで使用される座標はすべて、新しい原点が基準となります。レンダリングされるイメージの質を高めるには、ImageSetAntialiasing 関数を使用します。

例
```
<!--- Create a 200x200-pixel image. --->
<cfset myImage=ImageNew("",200,200)>
<!--- Turn on antialiasing to improve image quality. --->
<cfset ImageSetAntialiasing(myImage)>
<!--- Translate the origin to (100,20). --->
<cfset ImageTranslateDrawingAxis(myImage,100,20)>
<!--- Draw a rectangle at the offset location. --->
<cfset ImageDrawRect(myImage,50,60,40,50)>
<!--- Display the image in a browser. --->
<cfimage source="#myImage#" action="writeToBrowser">
```

ImageWrite

説明
指定したディスク上またはメモリ内のターゲットに ColdFusion イメージを書き込みます。

戻り値
なし

カテゴリ
Image 関数

関数のシンタックス
```
ImageWrite(name [, destination, quality, optional])
```

関連項目
cfimage、GetWriteableImageFormats、ImageNew、ImageRead、IsImageFile

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。
destination	オプション。ディスク上またはメモリ内の、イメージを書き込むファイルの絶対パス名または相対パス名です。 ImageNew 関数など、書き込み先のファイル名を指定しないオペレーションを使用してイメージを作成する場合は、destination パラメータを指定します。 ファイル形式はファイル名の拡張子に基づいて決定されます。このパラメータのデフォルト値は、ソースイメージのファイル名です。
overwrite	オプション。true に設定すると、書き込み先のファイルが上書きされます。
quality	オプション。イメージをエンコードするときの JPEG 品質を定義します。このパラメータは、destination パラメータで指定したファイルの拡張子が JPG または JPEG である場合にのみ適用されます。有効な値は、0 ～ 1 の間の小数です (値が小さいほど品質は低くなります)。デフォルト値は 0.75 です。

使用方法

メモリ内のファイルを指定するには、次のシンタックスを使用します。この場合、ディスクには書き込まれません。メモリ内のファイルを使用すると、一時的データの処理が速くなります。

```
ram:///filepath
```

ファイルパスには、ram:///petStore/images/poodle.jpg のようにディレクトリを含めることができます。ファイルを指定する前に、パスに含まれるディレクトリを作成しておく必要があります。メモリ内のファイルの使用方法について詳しくは、『ColdFusion アプリケーションの開発』の Working with in-memory files を参照してください。

ファイル形式はファイル拡張子に基づいて決定されますが、この関数を使用してイメージの形式を変換できます。

有効な書き込み形式については、332 ページの「cfimage」にあるサポートされているイメージファイル形式のリストを参照してください。ColdFusion アプリケーションがデプロイされているサーバー上で書き込み可能な形式のリストを取得するには、GetWriteableImageFormats 関数を使用します。

注意：イメージのファイル形式を変換する作業には時間がかかります。また、イメージの品質が低下する場合もあります。たとえば、PNG イメージでは 24 ビットの色がサポートされていますが、GIF イメージでは 256 色しかサポートされていません。透明のイメージ (アルファ値を持つイメージ) を変換すると、品質が低下する場合があります。

例

```
<!--- This example shows how to convert a GIF image to a PNG image. --->
<!--- Use the ImageNew function to create a ColdFusion image. --->
<cfset myImage = ImageNew("http://www.google.com/images/logo_sm.gif")>
<!--- Convert the image to a PNG format. --->
<cfset ImageWrite(myImage,"google.png")>
<!--- Display the PNG image. --->
<img src="google.png">
```

ImageWriteBase64

説明

指定したディスク上またはメモリ内のターゲットに Base64 イメージを書き込みます。

戻り値

Base64 文字列

COLDFUSION 10 CFML リファレンス
ColdFusion 関数
1040

カテゴリ
Image 関数

関数のシンタックス
```
ImageWriteBase64(name, destination, format [, inHTMLFormat, optional])
```

関連項目
cfimage、ImageReadBase64、IsImageFile

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。
destination	必須です。ディスク上またはメモリ内の、イメージを書き込むファイルの絶対パス名または相対パス名です。
format	必須です。形式です。
inHTMLFormat	オプション。HTML タグ内の Base64 イメージで使用するヘッダを Base64 出力に含めるかどうかを指定します ("data:image/base64,...")。• yes• no (デフォルト)
overwrite	オプション。true に設定すると、書き込み先のファイルが上書きされます。

使用方法
ImageWriteBase64 関数を使用すると、イメージデータを印刷可能文字列にエンコードできます。この関数は、電子メールでイメージを送信する場合や、データベースのテキストフィールドにイメージを保存する場合などに便利です。

メモリ内のファイルを指定するには、次のシンタックスを使用します。この場合、ディスクには書き込まれません。メモリ内のファイルを使用すると、一時的データの処理が速くなります。

```
ram:///filepath
```

ファイルパスには、ram:///petStore/images/poodle.jpg のようにディレクトリを含めることができます。ファイルを指定する前に、パスに含まれるディレクトリを作成しておく必要があります。メモリ内のファイルの使用方法について詳しくは、『ColdFusion アプリケーションの開発』の Working with in-memory files を参照してください。

ファイル形式を指定しないと、Base64 に変換する前にイメージをエンコードするための形式を ColdFusion が認識できないため、エラーが発生します。

ColdFusion が Base64 文字列を正しく読み込んでいるかどうかを確認するには、次の方法があります。

- cfdump タグを使用します。例 :
- ImageInfo 関数を使用します。例 :
- ImageWrite 関数を使用して、イメージを JPEG ファイルとして保存します。JPEG ファイルをブラウザまたは画像アプリケーションで開きます。

最終更新日 2012/7/20

例

```
<!--- This example shows how to convert a JPEG image to Base64 format and save it to a file. --->
<!--- Create a new ColdFusion image. --->
<cfset myImage=ImageNew("../cfdocs/images/artgallery/jeff01.jpg")>
<!--- Convert the image to Base64 format and write it to a file.--->
<cfset ImageWriteBase64(myImage,"jeffBase64.txt","jpg","yes")>
```

ImageXORDrawingMode

説明

現在の色と指定した色を入れ替えるようにイメージのペイントモードを設定します。

戻り値

なし

カテゴリ

Image 関数

関数のシンタックス

```
ImageXORDrawingMode(name, c1)
```

関連項目

ImageSetDrawingColor、IsImageFile

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。オペレーションの対象になる ColdFusion イメージです。
c1	必須です。XOR の交換色です。 • RGB カラーの 16 進数値。たとえば、白の場合は "##FFFFFF" または "FFFFFF" を指定します。 • 色の文字列値 ("black"、"red"、"green" など)。

使用方法

この関数は、ピクセルの現在の色と指定した XOR (排他的論理和) の色を入れ替えます。

描画オペレーションが実行されると、現在の色のピクセルは指定した色に変換され、指定した色のピクセルは現在の色に変換されます。

現在の色でも指定した色でもないピクセルの変換結果は予測不能ですが、この結果は元に戻せます。同じ図形を 2 回描画すると、すべてのピクセルが元の値に戻ります。

色の値を文字列で指定する場合は、サポートされているカラー名を使用します。有効な HTML カラー名については、332 ページの「cfimage」を参照してください。16 進数値を入力するには、"##xxxxxx" または "xxxxxx" という形式を使用します (x は 0 ～ 9 または A ～ F です)。シャープ記号 (#) は 2 つ使用するか、または使用しないでください。

例

```
<!--- This example shows how to draw rectangles with alternating colors where they overlap. --->
<!--- Use the ImageNew function to create a 300x200-pixel image in RGB format. --->
<cfset myImage = ImageNew("",300,200,"rgb")>
<!--- Turn on antialiasing to improve image quality. --->
<cfset ImageSetAntialiasing(myImage)>
<!--- Set the drawing color to white. --->
<cfset ImageSetDrawingColor(myImage,"white")>
<!--- Draw a white filled rectangle that is the size of the original image. --->
<cfset ImageDrawRect(myImage,0,0,300,200,"yes")>
<!--- Set the XOR drawing mode to white. --->
<cfset ImageXORDrawingMode(myImage,"white")>
<!--- Set the drawing color to red. --->
<cfset ImageSetDrawingColor(myImage,"red")>
<!--- Draw a filled red rectangle. --->
<cfset ImageDrawRect(myImage,50,50,150,100,"yes")>
<!--- Translate the drawing axis to (25,25). --->
<cfset ImageTranslateDrawingAxis(myImage,25,25)>
<!--- Set the drawing color to blue. --->
<cfset ImageSetDrawingColor(myImage,"blue")>
<!--- Draw a blue filled rectangle at the offset location. --->
<cfset ImageDrawRect(myImage,50,50,150,100,"yes")>
<!--- Save the ColdFusion image as a PNG file. --->
<cfset ImageWrite(myImage,"xortest.png")>
<!--- Display the PNG file.--->
<img src="xortest.png">
```

関数 in ～ k

IncrementValue

説明

整数値に 1 を加算します。

戻り値

1 を加算した数値 (**number**) の整数部分

カテゴリ

算術関数

関数のシンタックス

```
IncrementValue(number)
```

関連項目

DecrementValue

パラメータ

パラメータ	説明
number	インクリメントする数値です。

例

```
<h3>IncrementValue Example</h3>
<p>Returns the integer part of a number incremented by one.
<p>IncrementValue(0): <cfoutput>#IncrementValue(0)#</cfoutput>
<p>IncrementValue("1"): <cfoutput>#IncrementValue("1")#</cfoutput>
<p>IncrementValue(123.35): <cfoutput>#IncrementValue(123.35)#</cfoutput>
```

InputBaseN

説明

radix で指定された基数を使用して、文字列 **string** を整数に変換します。

戻り値

2 ～ 36 の範囲の数値の文字列

カテゴリ

算術関数

関数のシンタックス

```
InputBaseN(string, radix)
```

関連項目

FormatBaseN

パラメータ

パラメータ	説明
string	文字列、または文字列を含んでいる変数です。radix で指定された基数で数値を表す文字列です。
radix	string で表される数値の基数です。範囲は 2 ～ 36 です。

例

```
<h3>InputBaseN Example</h3>

<p>FormatBaseN converts a number to a string in the base specified by Radix.
<p>
<cfoutput>
<br>FormatBaseN(10,2): #FormatBaseN(10,2)#
<br>FormatBaseN(1024,16): #FormatBaseN(1024,16)#
<br>FormatBaseN(125,10): #FormatBaseN(125,10)#
<br>FormatBaseN(10.75,2): #FormatBaseN(10.75,2)#
</cfoutput>
<h3>InputBaseN Example</h3>
<p>InputBaseN returns the number obtained by converting a string,
    using the base specified by Radix,.
<cfoutput>
<br>InputBaseN("1010",2): #InputBaseN("1010",2)#
<br>InputBaseN("3ff",16): #InputBaseN("3ff",16)#
<br>InputBaseN("125",10): #InputBaseN("125",10)#
<br>InputBaseN(1010,2): #InputBaseN(1010,2)#
</cfoutput>
```

Insert

説明
文字列内で、指定された位置の後に部分文字列を挿入します。position=0 の場合は、文字列の先頭に部分文字列が挿入されます。

戻り値
文字列です。

カテゴリ
文字列関数

関数のシンタックス
```
Insert(substring, string, position)
```

関連項目
RemoveChars、Len

パラメータ

パラメータ	説明
substring	文字列、または文字列を含んでいる変数です。挿入する文字列を指定します。
string	文字列、または文字列を含んでいる変数です。substring を挿入する文字列を指定します。
position	整数または変数です。文字列のこの位置の後ろに substring が挿入されます。

例
```
<h3>Insert Example</h3>

<cfif IsDefined("FORM.myString")>
    <!--- if the position is longer than the length of the string, err --->
    <cfif FORM.insertPosition GT Len(MyString)>
        <cfoutput>
        <p>This string only has #Len(MyString)# characters; therefore,
    you cannot insert the substring #FORM.mySubString# at position
    #FORM.insertPosition#.
        </cfoutput>
    </cfif>
<cfelse>
    <cfoutput>
    <p>You inserted the substring #FORM.MySubstring# into the string
    #FORM.MyString#, resulting in the following string:
    <br>#Insert(FORM.MySubString, FORM.myString,
    FORM.insertposition)#
    </cfoutput>
</cfif>
```

Int

説明
指定された数値 (**number**) よりも小さく、その値に最も近い整数を返します。たとえば、Int(3.3) および Int(3.7) の場合は 3 が返されます。Int(-3.3) および Int(-3.7) の場合は -4 が返されます。

戻り値
文字列で表した整数

カテゴリ
算術関数

関数のシンタックス
```
Int(number)
```

関連項目
Ceiling、Fix、Round

パラメータ

パラメータ	説明
number	整数に切り下げる実数です。

例
```
<h3>Int Example</h3>
<p>Int returns the closest integer smaller than a number.

<p>Int(11.7) : <cfoutput>#Int(11.7)#</cfoutput>
<p>Int(-11.7) : <cfoutput>#Int(-11.7)#</cfoutput>
<p>Int(0) : <cfoutput>#Int(0)#</cfoutput>
```

Invoke

説明
次のいずれかを行います。

- ColdFusion ページまたは ColdFusion コンポーネント内からコンポーネントメソッドを呼び出します。
- Web サービスを呼び出します。

戻り値
呼び出したメソッドで返される結果が返されます。

シンタックス
invoke(cfcinstance, methodname [, arguments])

プロパティ

パラメータ	説明
cfcinstance	必須です。文字列またはコンポーネントオブジェクトです。コンポーネントへの参照、またはインスタンス化するコンポーネントを指定します。
methodname	必須です。メソッドの名前です。Web サービスの場合は、オペレーションの名前です。
arguments	オプション。メソッドに渡す引数です。

例

```
<cfscript>
obj = createObject("component","TestComponent");
invoke(obj,"function1",{a1="ColdFusion"});
</cfscript>
```

IsArray

説明

値 (value) が配列かどうかを調べます。

戻り値

値 (**value**) が配列またはクエリー列オブジェクトの場合は true

カテゴリ

配列関数、決定関数

関数のシンタックス

```
IsArray(value [, number ])
```

関連項目

配列関数、『ColdFusion アプリケーションの開発』の Modifying a ColdFusion XML object

履歴

ColdFusion MX:

- 動作が変更されました。**value** パラメータにクエリー結果列への参照が含まれている場合は true が返されるようになりました。たとえば、isArray(MyQuery['Column1']) では true が返されます (以前のリリースでは false が返されました)。
- 動作が変更されました。この関数を XML オブジェクトに対して使用できるようになりました。

パラメータ

パラメータ	説明
value	変数名または配列名です。
number	次元です。関数により配列がこの次元の配列であるかどうかがテストされます。

使用方法

この関数を使用して、値 (value) が配列またはクエリー列かどうかを調べます。この関数は、ベクターオブジェクトなどの Java 配列オブジェクトを 1 次元の配列として評価します。

例

```
<h3>IsArray Example</h3>

<!--- Make an array --->
<cfset MyNewArray = ArrayNew(1)>
<!--- set some elements --->
<cfset MyNewArray[1] = "element one">
<cfset MyNewArray[2] = "element two">
<cfset MyNewArray[3] = "element three">
<!--- is it an array? --->
<cfoutput>
    <p>Is this an array? #IsArray(MyNewArray)#
    <p>It has #ArrayLen(MyNewArray)# elements.
    <p>Contents: #ArrayToList(MyNewArray)#
</cfoutput>
```

IsAuthenticated

説明

この関数は廃止されました。新しいセキュリティツールを使用してください。詳細については、711 ページの「変換関数」および『ColdFusion アプリケーションの開発』の Securing Applications を参照してください。

履歴

ColdFusion MX: この関数は廃止されました。この関数は ColdFusion MX および ColdFusion の今後のリリースでは動作しません。

IsAuthorized

説明

この関数は廃止されました。新しいセキュリティツールを使用してください。詳細については、711 ページの「変換関数」および『ColdFusion アプリケーションの開発』の Securing Applications を参照してください。

履歴

ColdFusion MX: この関数は廃止されました。ColdFusion MX およびこれ以降のリリースでは機能しません。

IsBinary

説明

値がバイナリデータとして保管されているかどうかを調べます。

戻り値

値がバイナリの場合は true、それ以外は false。

カテゴリ

決定関数

関数のシンタックス

```
IsBinary(value)
```

関連項目

ToBinary、ToBase64、IsNumeric、YesNoFormat

パラメータ

パラメータ	説明
value	数値または文字列です。

例

```
<!--- Create a string of all ASCII characters (32-255)
    and concatenate them together. --->
<cfset charData ="">
<cfloop index="data" from="32" to="255">
    <cfset ch=chr(data)>
    <cfset charData=charData & ch>
</cfloop>

<b>The following string is the concatenation of all characters (32 to 255) from the ASCII table.</b><br><br>
<cfoutput>#htmleditformat(charData)#</cfoutput>
<br><br>
<!--- Create a Base 64 representation of this string. --->
<cfset data64=toBase64(charData)>
<!--- Convert string to binary. --->
<cfset binaryData=toBinary(data64)>
<!--- Check to see if it really is a binary value. --->
<cfif IsBinary(binaryData)>
The binaryData variable is binary!<br>
</cfif>
<!--- Convert binary data back to Base 64. --->
<cfset another64=toBase64(binaryData)>
<cfif Not IsBinary(another64)>
The another64 variable is NOT binary!<br>
</cfif>
<!--- Compare another64 with data64 to ensure that they are equal. --->
<cfif another64 eq data64>
    Base 64 representation of binary data is identical to the Base 64
    representation of string data.
<cfelse>
    <h3>Conversion error.</h3>
</cfif>
```

IsBoolean

説明

値をブール値に変換できるかどうかを調べます。

戻り値

パラメータをブール値に変換できる場合は true、変換できない場合は false。

カテゴリ

決定関数

関数のシンタックス

```
IsBoolean(value)
```

関連項目
IsNumeric、IsValid、YesNoFormat

パラメータ

パラメータ	説明
value	数値または文字列です。

例
```
<h3>IsBoolean Example</h3>

<cfif IsDefined("FORM.theTestValue")>
    <cfif IsBoolean(FORM.theTestValue)>
        <h3>The expression <cfoutput>#DE(FORM.theTestValue)#</cfoutput> is
    Boolean</h3>
    <cfelse>
        <h3>The expression <cfoutput>#DE(FORM.theTestValue)#</cfoutput> is not Boolean</h3>
    </cfif>
</cfif>

<form action = "isBoolean.cfm">
<p>Enter an expression, and discover whether it can be evaluated to a
    Boolean value.

<input type = "Text" name = "TheTestValue" value = "1">
<input type = "Submit" value = "Is it Boolean?" name = "">
</form>
```

IsClosure

説明
クロージャの名前かどうかを調べます。

戻り値
名前をクロージャとして呼び出せる場合は True、呼び出せない場合は False。

カテゴリ
決定関数

シンタックス
isClosure(closureName)

関連項目
他の決定関数。

履歴
ColdFusion 10: この関数が追加されました。

パラメータ

パラメータ	説明
closureName	クロージャの名前です。引用符で囲まないでください。 定義されている変数名または関数名ではない場合は、エラーになります。

使用方法
クロージャの名前かどうかを判定するには、この関数を使用します。

例
```
<cfscript>
    isClosure(closureName)
    {
     // do something
    }
    else
    {
     // do something
    }
</cfscript>
```

IsCustomFunction

説明
カスタム関数の名前かどうかを調べます。

戻り値
name をカスタム関数として呼び出せる場合は true、カスタム関数として呼び出せない場合は false。

カテゴリ
決定関数

関数のシンタックス
```
IsCustomFunction(name)
```

パラメータ

パラメータ	説明
name	カスタム関数の名前です。引用符で囲まないでください。定義された変数や関数名でない場合はエラーが発生します。

使用方法
IsCustomFunction 関数は、カスタム関数として呼び出せるすべての関数に対して true を返します。この中には、CFScript の function 宣言および cffunction タグで定義した関数や、ColdFusion コンポーネントメソッドの関数が含まれます。CFC メソッドの場合は、まずコンポーネントのインスタンスを作成します。

注意：未定義の変数による例外を防ぐには、次の例のように、IsCustomFunction の前に常に IsDefined テストを挿入してください。

例

```
<h3>IsCustomFunction Example</h3>
<cfscript>
function realUDF() {
    return 1;
}
</cfscript>
<cfset X = 1>

<!--- Example that fails existence test --->
<cfif IsDefined("Foo") AND IsCustomFunction(Foo)>
    Foo is a UDF.<br>
</cfif>

<!--- Example that passes existence test but fails IsCustomFunction --->
<cfif IsDefined("X") AND IsCustomFunction(X)>
    X is a UDF.<br>
</cfif>

<!--- Example that passes both tests--->
<cfif IsDefined("realUDF") AND IsCustomFunction(realUDF)>
    realUDF is a function.<br>
</cfif>

<!--- Example using a CFC, defined in TestCFC.cfc--->
<cfobject component="TestCFC" name="myTestCFCobject">
<CFIF IsDefined("myTestCFCobject.testFunc") AND
        IsCustomFunction(myTestCFCobject.testFunc)>
    myTestCFCobject.testFunc is a function.
</CFIF>
```

IsDate

説明

文字列または Java オブジェクトが日付時刻値に変換できるかどうかを調べます。

戻り値

文字列 (**string**) を日付時刻値に変換できる場合は true、変換できない場合は false。ColdFusion では、ブール値の戻り値は、それに対応する文字列 "Yes" または "No" に変換されます。

カテゴリ

日付および時刻関数、決定関数

関数のシンタックス

```
IsDate(string)
```

関連項目

CreateDateTime、IsNumericDate、IsValid、LSDateFormat、LSIsDate、ParseDateTime

パラメータ

パラメータ	説明
string	文字列、または文字列を含んでいる変数です。

使用方法
この関数によって調べられるのは、米国の日付形式のみです。他の日付形式のサポートについては、LSDateFormat を参照してください。

日付時刻オブジェクトは西暦 100 ～ 9999 年の範囲内となります。

例
```
<h3>IsDate Example</h3>
<cfif IsDefined("FORM.theTestValue")>
    <cfif IsDate(FORM.theTestValue)>
    <h3>The string <cfoutput>#DE(FORM.theTestValue)#</cfoutput>
     is a valid date</h3>
    <cfelse>
    <h3>The string <cfoutput>#DE(FORM.theTestValue)#</cfoutput>
     is not a valid date</h3>
    </cfif>
</cfif>
<form action = "isDate.cfm" method="post">
<p>Enter a string, find whether it can be evaluated to a date value.
<p><input type = "Text" name = "TheTestValue" value = "<cfoutput>#Now()#
    </cfoutput>">
<input type = "Submit" value = "Is it a Date?" name = "">
</form>
```

IsDDX

説明
DDX ファイルが存在し、そのファイルが有効であるかどうかを調べます。または、文字列に有効な DDX 命令が含まれているかどうかを調べます。

戻り値
値が有効な DDX ファイルまたは文字列を表している場合は true。それ以外の場合は false。

カテゴリ
決定関数

関数のシンタックス
```
IsDDX("path or string")
```

関連項目
IsPDFObject、IsPDFFile、cfpdf

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
path or string	DDX ファイルのパス名または DDX 命令の文字列です。パス名を指定する場合は、絶対パスまたは DDX ファイルを呼び出す CFM ページからの相対パスを引用符で囲む必要があります。

使用方法
DDX ファイルのパス名が無効な場合、DDX ファイルのパス名が指定されていない場合、DDX ファイルが ColdFusion でサポートされているスキーマに準拠していない場合、または DDX 命令が無効な場合は、false が返されます。

例
```
<cfif IsDDX("TOCformat.ddx")>

    <cfset inputStruct=StructNew()>
    <cfset inputStruct.Doc0="title.pdf">
    <cfset inputStruct.Doc1="Chap1.pdf">
    <cfset inputStruct.Doc2="Chap2.pdf">
    <cfset inputStruct.Doc3="Chap3.pdf">
    <cfset inputStruct.Doc4="Chap4.pdf">

    <cfset outputStruct=StructNew()>
    <cfset outputStruct.Out1="myBook.pdf">

    <cfpdf action="processddx" ddxfile="TOCformat.ddx" inputfiles="#inputStruct#"
outputfiles="#outputStruct#" name="ddxVar">

    <cfoutput>#ddxVar.Out1#</cfoutput>

<cfelse>
    <p>This is not a valid DDX file.</p>
</cfif>
```

IsDebugMode

説明
デバッグ出力が有効化されているかどうかを調べます。

戻り値
ColdFusion Administrator でデバッグモードが設定されている場合は true、デバッグモードが無効になっている場合は false。

カテゴリ
決定関数

関数のシンタックス
```
IsDebugMode()
```

関連項目
cfsetting

説明
ColdFusion Administrator でデバッグ出力が有効化されており、かつその設定が cfsetting タグの showDebugOutput 属性で上書きされていない (つまりこの属性が No に設定されていない) 場合には、IsDebugMode 関数は Yes を返します。それ以外の場合は No を返します。

例

```
<h3>IsDebugMode Example</h3>
<cfif IsDebugMode()>
    <h3>Debugging is set in the ColdFusion Administrator</h3>
<cfelse>
    <h3>Debugging is disabled</h3>
</cfif>
```

IsDefined

説明

文字列値を評価して、その中で指定されている変数が存在するかどうかを調べます。

この関数は、非推奨になった ParameterExists 関数の代わりとなるものです。

戻り値

変数が見つかった場合は true、それ以外の場合は false。

カテゴリ

決定関数

関数のシンタックス

```
IsDefined("variable_name")
```

関連項目

Evaluate

履歴

ColdFusion MX: 動作が変更されました。この関数は、次の変数および構造体のみを処理できます。

- 単純変数
- ドット表記の名前付き変数
- ドット表記の名前付き構造体 (例 : mystruct.key)

パラメータ

パラメータ	説明
variable_name	引用符で囲まれた文字列です。テストする変数の名前です。

使用方法

myArray[3] のような配列エントリをこの関数に渡すとエラーになります。特定のエントリが配列内に存在するかどうかをチェックするには、ArrayIsDefined 関数を使用します。

この関数または StructKeyExists 関数を使用すると、構造体内に特定のキーが存在するかどうかをテストできます。たとえば、ColdFusion が構造体として公開しているスコープを使用するときには、この関数の代わりに StructKeyExists 関数を使用できる場合もあります。次の 2 つの行は同等です。

```
if(isDefined("form.myVariable"))
if(structKeyExists(form,"myVariable"))
```

例

```
<cfif IsDefined("form.myString")>
    <p>The variable form.myString has been defined, so show its contents.
    This construction allows us to place a form and its resulting action code
    on the same page and use IsDefined to control the flow of execution.</p>
    <p>The value of "form.myString" is <b><i>
    <cfoutput>#form.myString#</cfoutput></i></b>
<cfelse>
    <p>During the first time through this template, the variable "form.myString"
    has not yet been defined, so we do not try to evaluate it.
</cfif>

<form action="#CGI.Script_Name" method="POST">
<input type="Text" name="MyString" value="My sample value">
<input type="Submit" name="">
</form>
```

IsImage

説明

変数が ColdFusion イメージを返すかどうかを調べます。

戻り値

値が ColdFusion イメージの場合は true、それ以外の場合は false。

カテゴリ

Image 関数

関数のシンタックス

```
IsImage(name)
```

関連項目

cfimage、ImageGetBlob、ImageInfo、ImageNew

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	必須です。調査する ColdFusion 変数です。

使用方法

変数が ColdFusion イメージを返すかどうかを調べる場合に、この関数を使用します。

例

```
<cfif IsImageFile("images/#form.art#")>
<cfset myImage=ImageNew("images/#form.art#")>
...
<cfset IsImage("#myImage#")>
<cfimage action="writeToBrowser" source="#myImage#">
</cfif>
```

IsImageFile

説明
イメージファイルが有効かどうかを調べます。

戻り値
値が有効なイメージファイルの場合は true、それ以外の場合は false。

カテゴリ
Image 関数

関数のシンタックス
```
IsImageFile("path")
```

関連項目
cfimage、ImageGetBlob、ImageInfo、ImageNew、IsImage

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
path	必須です。調査するディスク上またはメモリ内のファイル、あるいは URL のパス名です。絶対パスまたは CFM ページからの相対パスを引用符で囲んで指定します。

使用方法
イメージファイルが有効かどうかを調べる場合に、この関数を使用します。イメージファイルの形式が ColdFusion サーバーでサポートされていない場合、イメージファイルのパス名が指定されていない場合、またはパス名が無効な場合は、false が返されます。

ColdFusion でサポートされている標準のイメージファイル形式については、332 ページの「cfimage」にあるサポートされているイメージファイル形式のリストを参照してください。ColdFusion サーバーでサポートされているイメージファイル形式を確認するには、948 ページの「GetReadableImageFormats」および 963 ページの「GetWriteableImageFormats」を使用します。

例
```
<!--- Use the IsImageFile function to determine whether an image retrieved from the artwork table in the
cfartgallery database is a valid image file. --->
<cfif IsImageFile("images/#artwork.largeImage#")>
    <cfset myImage=ImageNew("images/#artwork.largeImage#")>
    <cfset ImageResize(myImage,50,"")>
    <cfimage action="writeToBrowser" source="#myImage#">
<cfelse>
    <p>I'm sorry, there is no image associated with the title you selected. Please click the Back button
and try again.</p>
    </p>
</cfif>
```

IsInstanceOf

説明
オブジェクトが ColdFusion インターフェイス、ColdFusion コンポーネント、または Java クラスのインスタンスであるかどうかを調べます。

戻り値
次のいずれかの条件に一致する場合は true を返します。

- 最初のパラメータで指定したオブジェクトが、2 番目のパラメータで指定したインターフェイスまたはコンポーネントのインスタンスである場合。
- 最初のパラメータで指定した Java オブジェクトが cfobject タグまたは CreateObject メソッドによって作成されたオブジェクトであり、2 番目のパラメータで指定した Java クラスのインスタンスである場合。
- 最初のパラメータで指定したオブジェクトが、2 番目のパラメータで指定したコンポーネントを拡張するコンポーネントのインスタンスである場合。
- 最初のパラメータで指定したオブジェクトが、2 番目のパラメータで指定したインターフェイスを実装するコンポーネントを拡張するコンポーネントのインスタンスである場合。
- 最初のパラメータで指定した Java オブジェクトが cfobject タグまたは CreateObject メソッドによって作成されたオブジェクトであり、2 番目のパラメータで指定したクラスを拡張する Java クラスのインスタンスである場合。

  それ以外の場合は false を返します。

  注意：object パラメータで指定した CFC で関数が定義されていない場合、isInstanceOf 関数は false を返します。

カテゴリ
決定関数、拡張関数

関数のシンタックス
```
IsInstanceOf(object, typeName)
```

関連項目
cfcomponent、cfinterface、cfobject

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
object	調査する CFC インスタンスまたは Java オブジェクトです。
typeName	インターフェイス名、コンポーネント名、または Java クラス名です。このオブジェクトが、ここで指定したもののインスタンスであるかどうかが調査されます。

使用方法
Java オブジェクトの場合は、最初のパラメータで指定したオブジェクトが cfobject タグまたは CreateObject メソッドによって作成されたものである場合にのみ比較が有効になります。

例

```
<!--- to use this example, create an I1.cfc interface, as follows: --->

<cfinterface>
    <cffunction name = "method1">
    </cffunction>
</cfinterface>

<!--- Create a C1.cfc component that implements the I1.cfc interface, as
    follows: --->
<cfcomponent implements=I1>
    <cffunction name = "method1">
        <cfoutput>C1.method1 called</cfoutput>
    </cffunction>
</cfcomponent>

<!--- Create a test.cfm file as follows and display the page. --->
<!--- Create an instance of the C1 CFC, which implements the I1 interface.
    --->
<cfset c1obj = CreateObject("Component", "C1")>
<!--- Create a Java object --->
<cfset javaObj = createobject("java", "java.lang.System")>

<cfoutput>
    IsInstanceOf(c1obj,"C1") = #IsInstanceOf(c1obj,"C1")#
        (Expected = YES)<br><br>
    IsInstanceOf(c1obj,"I1") = #IsInstanceOf(c1obj,"I1")#
        (Expected = YES)<br><br>
    IsInstanceOf(c1obj,"C2") = #IsInstanceOf(c1obj,"C2")#
        (Expected = NO)<br><br>
    IsInstanceOf(javaObj,"java.lang.System") =
            #IsInstanceOf(javaObj,"java.lang.System")# (Expected = YES)<br><br>
    IsInstanceOf(javaObj,"java.lang.String") =
            #IsInstanceOf(javaObj,"java.lang.String")# (Expected = NO)<br><br>
</cfoutput>
```

IsIPv6

説明

ホストが IPv6 をサポートしているかどうかを判別します。

戻り値

ホストが IPv6 をサポートする場合は true。

カテゴリ

決定関数

シンタックス

```
IsIPv6()
IsIPv6(hostname)
```

関連項目

GetLocalHostIP、IsLocalHost

履歴
ColdFusion 9: この関数が追加されました。

使用方法
この関数を使用してリモートホストが IPv6 をサポートするかどうかを確認する場合は、IP アドレスではなくホスト名を渡します。

この関数は、要求を行うサーバーが IPv6 対応の場合にのみ適用されます。

例
次の例は、ローカルホストが IPv6 をサポートするかどうかをチェックします。

```
<cfset hostname="localhost">
<cfset test=IsIPv6()>
<cfif test>
<cfoutput>localhost supports IPv6</cfoutput>
</cfif>
```

次の例は、リモートホストが IPv6 をサポートするかどうかをチェックします。

```
<cfset hostname="remote hostname">
<cfset test=IsIPv6(hostname)>
<cfif test>
<cfoutput>#hostname# supports IPv6</cfoutput>
</cfif>
```

IsJSON

説明
文字列が有効な JSON (JavaScript Object Notation) データ交換形式であるかどうかを評価します。

戻り値
パラメータが有効な JSON 値である場合は true。

パラメータが有効な JSON データ表現でない場合は false。

カテゴリ
変換関数

シンタックス
`IsJSON(var)`

関連項目
DeserializeJSON、SerializeJSON、cfajaxproxy、『ColdFusion アプリケーションの開発』の Using Ajax Data and Development Features、http://www.json.org

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
var	文字列、または文字列を表す変数です。

例

次の例では、SerializeJSON 関数の例で生成されたデータフィードに有効な JSON データが含まれているかどうかをチェックします。

このフィードは JavaScript 関数呼び出しの形式を持ち、フィードのデータを含む JSON 文字列がパラメータとして指定されています。この例では、次の処理を行います。

1. cfhttp タグを使用してフィードを取得します (cfhttp.fileContent 変数)。
2. テキストから関数呼び出しラッパーを取り除きます。
3. IsJSON 関数を使用して、前の処理の結果が有効な JSON 形式文字列であるかどうかを確認します。
4. テキストが有効な JSON データであり、その後にフィードのテキスト文字列が続いているかどうかを示すメッセージを表示します。

   この例を実行するときは、このファイルと SerializeJSON 関数の例を ColdFusion の Web ルート下の適切な場所に置き、URL をシリアル化の例の URL に置き換えてから、このページを実行してください。

```
<!--- Get the JSON Feed --->
<cfhttp url="http://localhost:8500/My_Stuff/Ajax/Books/CreateJSON_NEW.cfm">

<!--- JSON data is often distributed as a JavaScript function.
     The following REReplace functions strip the function wrapper. --->
<cfset theData=REReplace(cfhttp.FileContent,
        "^\s*[[:word:]]*\s*\(\s*","")>
<cfset theData=REReplace(theData, "\s*\)\s*$", "")>

<!--- Test to make sure you have JSON data. --->
<cfif isJSON(theData)>
    <h3>The URL you requested provides valid JSON</h3>
<cfelse>
    <h3>The URL you requested does not provide valid JSON</h3>
</cfif>
<!--- Display the data. --->
<cfoutput>#theData#</cfoutput>
```

   JSON データを変換するさらに複雑な例については、DeserializeJSON を参照してください。

IsK2ServerABroker

説明

この関数は非推奨となっています。

戻り値

K2Broker が ColdFusion に設定されている場合は true、設定されていない場合は false。

カテゴリ

決定関数、全文検索関数、クエリー関数

関数のシンタックス

```
IsK2ServerABroker()
```

関連項目
GetK2ServerDocCountLimit、IsK2ServerDocCountExceeded、IsK2ServerOnline

履歴
ColdFusion MX 6.1: この関数が非推奨になりました。今後のリリースでは、動作せずにエラーとなる可能性があります。

ColdFusion MX: この関数が追加されました。

例
```
<cfoutput>IsK2ServerABroker =
    $*#IsK2ServerABroker()#*$</cfoutput>
```

IsK2ServerDocCountExceeded

説明
この関数は非推奨となっています。

戻り値
ドキュメントの制限数を超えている場合は true、超えていない場合は false。

カテゴリ
決定関数、全文検索関数、クエリー関数

関数のシンタックス
```
IsK2ServerDocCountExceeded()
```

関連項目
GetK2ServerDocCountLimit、IsK2ServerABroker

履歴
ColdFusion MX 6.1: この関数が非推奨になりました。今後のリリースでは、動作せずにエラーとなる可能性があります。

ColdFusion 5: この関数が追加されました。

例
```
<cfoutput>IsK2ServerDocCountExceeded =
    $*#IsK2ServerDocCountExceeded()#*$</cfoutput>
```

IsK2ServerOnline

説明
K2 サーバーは ColdFusion の実行時に常に起動しているため、この関数は非推奨となっています。

戻り値
K2 サーバーが検索を実行できる状態であれば true、実行できない場合は false

カテゴリ
決定関数、全文検索関数、クエリー関数

関数のシンタックス
```
IsK2ServerOnline()
```

関連項目
IsK2ServerABroker

履歴
ColdFusion MX 6.1: この関数が非推奨になりました。今後のリリースでは、動作せずにエラーとなる可能性があります。

ColdFusion MX: この関数が追加されました。

例
```
<cfoutput>IsK2ServerOnline =
    $*#IsK2ServerOnline()#*$</cfoutput>
```

IsLeapYear

説明
指定された年が閏年かどうかを調べます。

戻り値
指定された年 (**year**) が閏年である場合は true、閏年でない場合は false。

カテゴリ
日付および時刻関数、決定関数

関数のシンタックス
```
IsLeapYear(year)
```

関連項目
DaysInYear

パラメータ

パラメータ	説明
year	年を表す数値です。

例

```
<h3>IsLeapYear Example</h3>

<cfif IsDefined("FORM.theTestValue")>
    <cfif IsLeapYear(FORM.theTestValue)>
        <h3>The year value <cfoutput>#DE(FORM.theTestValue)#</cfoutput> is a Leap Year</h3>
    <cfelse>
        <h3>The year value <cfoutput>#DE(FORM.theTestValue)#</cfoutput> is not a Leap Year</h3>
    </cfif>
</cfif>

<form action = "isLeapYear.cfm" method="POST">
<p>Enter a year value, and find out whether it is a Leap Year.
<p><input type = "Text" name = "TheTestValue" value = "
    <cfoutput>#Year(Now())#</cfoutput>">
<input type = "Submit" value = "Is it a Leap Year?" name = "">
</form>
```

IsLocalHost

説明

指定された IP アドレスがローカルホストであるかどうかを確認します。IPv4 と IPv6 の両方のアドレスをサポートしています。

戻り値

IP アドレスがローカルホストの場合は true、そうでない場合は false。

カテゴリ

決定関数

関数のシンタックス

```
IsLocalHost(ipaddress)
```

関連項目

GetLocalHostIP

履歴

ColdFusion MX 7.01: この関数が追加されました。

パラメータ

パラメータ	説明
ipaddress	有効な IP アドレスです。

例

```
<h3>IsLocalHost Example</h3>

<cfif IsDefined("FORM.theTestIPAddress")>
    <cfif IsLocalHost(FORM.theTestIPAddress)>
        <h3>The IP address <cfoutput>#FORM.theTestIPAddress)#</cfoutput> is the localhost</h3>
    <cfelse>
        <h3>The IP address <cfoutput>#DE(FORM.theTestIPAddress)#</cfoutput> is not the localhost.</h3>
    </cfif>
</cfif>

<form action = "isIPAddressLocalHost.cfm">
<p>Enter an IP address to find out if it is the localhost.
<p><input type = "Text" name = "TheTestIPAddress" value = "127.0.0.1"
<input type = "Submit" value = "Is this the localhost?" name = "">
</form>
```

IsNull

説明

指定したオブジェクトまたは式が null と評価されるかどうかをチェックします。

戻り値

指定したオブジェクトが null、または指定した式が null と評価される場合は true、それ以外の場合は false。

カテゴリ

決定関数

関数のシンタックス

```
IsNull(obj)
```

パラメータ

パラメータ	説明
obj	null チェックを実行するオブジェクトです。

IsNumeric

説明

文字列を数値に変換できるかどうかを調べます。米国の数字形式の数値をサポートします。他の数字形式については、LSIsNumeric を使用してください。

戻り値

指定された文字列 (**string**) を数値に変換できる場合は true、変換できない場合は false。

カテゴリ

決定関数

関数のシンタックス

```
IsNumeric(string)
```

関連項目
IsBinary、IsValid

パラメータ

パラメータ	説明
string	文字列、または文字列を含んでいる変数です。

例
```
<h3>IsNumeric Example</h3>
<cfif IsDefined("FORM.theTestValue")>
    <cfif IsNumeric(FORM.theTestValue)>
    <h3>The string <cfoutput>#DE(FORM.theTestValue)#</cfoutput> can be
    converted to a number</h3>
    <cfelse>
    <h3>The string <cfoutput>#DE(FORM.theTestValue)#</cfoutput> cannot be
    converted to a number</h3>
    </cfif>
</cfif>

<form action = "isNumeric.cfm">
<p>Enter a string, and find out whether it can be evaluated to a numeric value.

<p><input type = "Text" name = "TheTestValue" value = "123">
<input type = "Submit" value = "Is it a Number?" name = "">
</form>
```

IsNumericDate

説明
実数値が、日付を表す値 (日付時刻オブジェクト) として有効かどうかを調べます。

戻り値
パラメータが有効な日付時刻オブジェクトを表す場合は true、その他の場合は false。

カテゴリ
日付および時刻関数、決定関数

関数のシンタックス
```
IsNumericDate(number)
```

関連項目
IsDate、ParseDateTime

パラメータ

パラメータ	説明
number	実数値です。

使用方法

ColdFusion は、IsNumericDate 関数に入力パラメータを渡す前に、そのパラメータを評価して実数値への変換を試みるという処理をデフォルトで行います。このため、12/12/03 や {ts '2003-01-14 10:04:13'} といった値をパラメータに指定した場合、この関数は true を返します。これは、ColdFusion が、有効な日付形式の文字列を日付時刻オブジェクト (実数) に変換するためです。

例

```
<h3>IsNumericDate Example</h3>
<cfif IsDefined("form.theTestValue")>
<!--- test if the value represents a number or a pre-formatted Date value --->

    <cfif IsNumeric(form.theTestValue) or IsDate(form.theTestValue)>
<!--- if this value is a numericDate value, then pass --->
        <cfif IsNumericDate(form.theTestValue)>
            <h3>The string <cfoutput>#DE(form.theTestValue)#</cfoutput> can be converted to a valid numeric
date</h3>
        <cfelse>
            <h3>The string <cfoutput>#DE(form.theTestValue)#</cfoutput> can not be converted to a valid
numeric date</h3>
        </cfif>
    <cfelse>
        <h3>The string <cfoutput>#DE(form.theTestValue)#</cfoutput> is not a valid numeric date</h3>
    </cfif>

</cfif>

<form action="#cgi.script_name#" method="POST">
<p>Enter a value, and discover if it can be evaluated to a date value.
<p>
<input type="Text" name="TheTestValue" value="<CFOUTPUT>#Now()#</CFOUTPUT>">
<input type="Submit" value="Is it a Date?" name="">
</form>
```

IsObject

説明

値 (value) がオブジェクトかどうかを調べます。

戻り値

値が ColdFusion オブジェクトを表す場合は true、値が他の型のデータ (整数、文字列、日付、構造体など) である場合は false。

カテゴリ

決定関数

関数のシンタックス

```
IsObject(value)
```

関連項目

IsDate、IsImage、IsNumeric、IsNumericDate、IsQuery、IsSimpleValue、IsStruct、IsWDDX、IsXmlDoc、IsXmlElem、IsXmlRoot

履歴
ColdFusion MX: この関数が追加されました。

パラメータ

パラメータ	説明
value	値です。通常は変数名です。

使用方法
この関数は、クエリーおよび XML オブジェクトに対しては false を返します。

例
```
<!--- to use this example, create a color.cfc component as follows: --->
<!---
<cfcomponent>
<cffunction name="myFunction" access="public" returntype="string">
<!--- Create a structure object --->
        <cfset myColor = "Blue">
        <cfreturn myColor>
</cffunction>
</cfcomponent>
--->

<!--- Create an instance of the color.cfc component --->
<cfobject name="getColor" component="color">

<cfif IsObject(getColor)>
    <!--- Invoke the myFunction method --->
    <cfinvoke
        component="#getColor#"
        method="myFunction"
        returnVariable="myColor">
    </cfinvoke>

    <cfif IsDefined("myColor")>
        <!--- Output the returned variable --->
        The value of myColor = <cfoutput>#myColor#</cfoutput><p>
    </cfif>
</cfif>
```

IsPDFFile

説明
PDF ファイルが有効であるかどうかを調べます。

戻り値
有効な PDF ファイルが返される場合は true。それ以外の場合は false。

カテゴリ
決定関数

関数のシンタックス
```
IsPDFFile("path")
```

関連項目
IsDate、IsImage、IsImageFile、IsNumeric、IsNumericDate、IsObject、IsPDFObject、IsQuery、IsSimpleValue、IsStruct、IsWDDX、IsXmlDoc、IsXmlElem、IsXmlRoot、cfpdf、cfpdfform、cfprint

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
path	ディスク上またはメモリ内の PDF ファイルのパス名です。絶対パスまたは CFM ページからの相対パスを引用符で囲んで指定します。

使用方法
PDF ファイルのパス名が無効な場合、パス名が指定されていない場合、PDF ファイルが無効な場合、または PDF ファイルが破損している場合は、false が返されます。

例
```
<!--- The following code shows the action page for a form where a user chooses a PDF document to print. --->

<cfif IsPDFFile("#form.printMe#")>
        <cfprint type="PDF" source="#myPDF#">
<cfelse>
    <p>This is not a valid PDF file or the PDF document you have chosen is not available.</p>
</cfif>
```

IsPDFObject

説明
値 (value) が PDF オブジェクトかどうかを調べます。

戻り値
値が PDF オブジェクトを表す場合は true。値が他の型のデータ (整数、文字列、日付、構造体など) である場合は false。

カテゴリ
決定関数

関数のシンタックス
```
IsPDFObject(value)
```

関連項目
IsDate、IsImage、IsNumeric、IsNumericDate、IsObject、IsPDFFile、IsQuery、IsSimpleValue、IsStruct、IsWDDX、IsXmlDoc、IsXmlElem、IsXmlRoot、cfpdf、cfpdfform

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
value	値です。通常は、変数名として格納されている PDF オブジェクトを指定します。

使用方法
この関数は、クエリーおよび XML オブジェクトに対しては false を返します。

例
```
<cfpdf source="c:\forms\'uoteform.pdf" action="read" name="myPDFform"/>
<cfif IsPDFObject(myPDFform)>
    <cfpdf source=#myPDFform# action="write" destination = "c:\forms\newPDFForm.pdf">
<cfelse>
    <p>This is not a PDF.</p>
</cfif>
```

IsProtected

説明
この関数は廃止されました。新しいセキュリティツールを使用してください。詳細については、711 ページの「変換関数」および『ColdFusion アプリケーションの開発』の Securing Applications を参照してください。

履歴
ColdFusion MX: この関数は廃止されました。ColdFusion MX およびこれ以降のリリースでは機能しません。

IsQuery

説明
値 (**value**) がクエリーであるかどうかを調べます。

戻り値
値 (**value**) がクエリーである場合は true

カテゴリ
決定関数、クエリー関数

関数のシンタックス
```
IsQuery(value)
```

関連項目
QueryAddRow

パラメータ

パラメータ	説明
value	クエリー変数です。

例

```
<!--- Shows an example of IsQuery and IsSimpleValue --->
<h3>IsQuery Example</h3>
<!--- define a variable called "GetEmployees" --->
<CFPARAM name = "GetEmployees" DEFAULT = "#Now()#">

<p>Before the query is run, the value of GetEmployees is
    <cfoutput>#GetEmployees#</cfoutput>

<cfif IsSimpleValue(GetEmployees)>
    <p>GetEmployees is currently a simple value
</cfif>
<!--- make a query on the snippets datasource --->
<cfquery name = "GetEmployees" datasource = "cfdocexamples">
SELECT *
FROM employees
</cfquery>

<p>After the query is run, GetEmployees contains a number of rows
    that look like this (display limited to three rows):
<cfoutput QUERY = "GetEmployees" MaxRows = "3">
<pre>#Emp_ID# #FirstName# #LastName#</pre>
</cfoutput>
<cfif IsQuery(GetEmployees)>
    GetEmployees is no longer a simple value, but the name of a query
</cfif>
```

IsSimpleValue

説明

値の型を調べます。

戻り値

値が文字列、数値、ブール値、または日付時刻値である場合は true、そうでない場合は false

カテゴリ

決定関数

関数のシンタックス

```
IsSimpleValue(value)
```

関連項目

IsValid

パラメータ

パラメータ	説明
value	変数または式です。

例

```
<!--- Shows an example of IsQuery and IsSimpleValue --->
<h3>IsSimpleValue Example</h3>
<!--- define a variable called "GetEmployees" --->
<cfparam name = "GetEmployees" default = "#Now()#">

<p>Before the query is run, the value of GetEmployees is
    <cfoutput>#GetEmployees#</cfoutput>

<cfif IsSimpleValue(GetEmployees)>
    <p>GetEmployees is currently a simple value
</cfif>
<!--- make a query on the snippets datasource --->
<cfquery name = "GetEmployees" datasource = "cfdocexamples">
    SELECT *
    FROM employees
</cfquery>
<p>After the query is run, GetEmployees contains a number of rows
    that look like this (display limited to three rows):
<cfoutput QUERY = "GetEmployees" MaxRows = "3">
<pre>#Emp_ID# #FirstName# #LastName#</pre>
</cfoutput>

<cfif IsQuery(GetEmployees)>
    GetEmployees is no longer a simple value, but the name of a query
</cfif>
```

IsSOAPRequest

説明

CFC が Web サービスとして呼び出されるかどうかを調べます。

戻り値

CFC が Web サービスとして呼び出される場合は true、そうでない場合は false。

カテゴリ

XML 関数

履歴

ColdFusion MX 7: この関数が追加されました。

関数のシンタックス

```
IsSOAPRequest()
```

関連項目

AddSOAPRequestHeader、AddSOAPResponseHeader、GetSOAPRequest、GetSOAPRequestHeader、GetSOAPResponse、GetSOAPResponseHeader、『ColdFusion アプリケーションの開発』の Basic web service concepts

使用方法

CFC 内でこの関数を呼び出して、CFC が Web サービスとして起動しているかどうかを調べます。

例

この例では、IsSOAPRequest 関数のオペレーションを示す CFC Web サービスを作成するとともに、他の ColdFusion SOAP 関数のオペレーションを示す Web サービスも提供します。

次のコードを、Web ルート下の "soapheaders" というフォルダに "headerservice.cfc" として保存します。この Web サービスを起動する例を実行することにより、そのオペレーション、特に IsSOAPRequest 関数のオペレーションをテストします。たとえば、AddSOAPRequestHeader の例を参照してください。

```
 <h3>IsSOAPRequest Example</h3>
<cfcomponent displayName="tester" hint="Test for SOAP headers">

<cffunction name="echo_me"
            access="remote"
            output="false"
            returntype="string"
            displayname="Echo Test" hint="Header test">

<cfargument name="in_here" required="true" type="string">

<cfset isSOAP = isSOAPRequest()>
<cfif isSOAP>

    <!--- Get the first header as a string and as XML --->
    <cfset username = getSOAPRequestHeader("http://mynamespace/", "username")>
    <cfset return = "The service saw username: " & username>
    <cfset xmlusername = getSOAPRequestHeader("http://mynamespace/", "username", "TRUE")>
    <cfset return = return & "<br> as XML: " & xmlusername>

    <!--- Get the second header as a string and as XML --->
    <cfset password = getSOAPRequestHeader("http://mynamespace/", "password")>
    <cfset return = return & "The service saw password: " & password>
    <cfset xmlpassword = getSOAPRequestHeader("http://mynamespace/", "password", "TRUE")>
    <cfset return = return & "<br> as XML: " & xmlpassword>

    <!--- Add a header as a string --->
<cfset addSOAPResponseHeader("http://www.tomj.org/myns", "returnheader", "AUTHORIZED VALUE", false)>

    <!--- Add a second header using a CFML XML value --->
    <cfset doc = XmlNew()>
    <cfset x = XmlElemNew(doc, "http://www.tomj.org/myns", "returnheader2")>
    <cfset x.XmlText = "hey man, here I am in XML">
    <cfsetx.XmlAttributes["xsi:type"] = "xsd:string">
    <cfset tmp = addSOAPResponseHeader("ignoredNameSpace", "ignoredName", x)>

<cfelse>
    <!--- Add a header as a string - Must generate error!
<cfset addSOAPResponseHeader("http://www.tomj.org/myns", "returnheader", "AUTHORIZED VALUE", false)>
    --->
<cfset return = "Not invoked as a web service">
</cfif>

<cfreturn return>

</cffunction>

</cfcomponent>
```

IsSpreadsheetFile

説明
入力がスプレッドシートファイルかどうかを判別する値を返します。

戻り値
ブール値です。入力がスプレッドシートファイルの場合は True、スプレッドシートオブジェクトの場合は False が返されます。

カテゴリ
Microsoft Office の統合

関数のシンタックス
```
IsSpreadsheetFile(spreadsheetfile)
```

関連項目

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
spreadsheefile	スプレッドシートファイルを指定します。

例
```
<cfset filepath = "C:/Documents/SingleSheet.xls">
<cfoutput># isSpreadSheetObject(filepath)# NOT an Object<br></cfoutput>
<cfoutput># isSpreadSheetFile(filepath)# a File<br></cfoutput>
```

IsSpreadsheetObject

説明
入力がスプレッドシートオブジェクトかどうかを判別する値を返します。

戻り値
ブール値です。入力がスプレッドシートオブジェクトの場合は True、スプレッドシートファイルの場合は False が返されます。

カテゴリ
Microsoft Office の統合

関数のシンタックス
```
IsSpreadsheetObject(spreadsheetobject)
```

関連項目
IsSpreadsheetFile

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
spreadsheeobject	スプレッドシートオブジェクトを指定します。

例
```
<cfspreadsheet action="read" src="C:/documents/SingleSheet.xls" name="SpreadsheetObj" >
<cfoutput># isSpreadSheetObject(SpreadsheetObj)# an Object<br></cfoutput>
<cfoutput># isSpreadSheetFile(SpreadsheetObj)# NOT a File<br></cfoutput>
```

IsStruct

説明
変数が構造体かどうかを調べます。

戻り値
変数 (**variable**) が ColdFusion 構造体または java.lang.Map インターフェイスを実装する Java オブジェクトの場合は true、変数 (**variable**) 内のオブジェクトがユーザー定義関数 (UDF) である場合は false。

カテゴリ
決定関数、構造体関数

関数のシンタックス
```
IsStruct(variable)
```

関連項目
構造体関数、『ColdFusion アプリケーションの開発』の Modifying a ColdFusion XML object

履歴
ColdFusion MX: 動作が変更されました。この関数を XML オブジェクトに対して使用できるようになりました。

パラメータ

パラメータ	説明
variable	変数名です。

例

```
<!--- This view-only example shows the use of IsStruct. --->
<p>This file is similar to addemployee.cfm, which is called by StructNew,
    StructClear, and StructDelete. It is an example of a custom tag used to
    add employees. Employee information is passed through the employee
    structure (the EMPINFO attribute). In UNIX, you must also add the Emp_ID.
<!---
<cfswitch expression = "#ThisTag.ExecutionMode#">
<cfcase value = "start">
    <cfif IsStruct(attributes.EMPINFO)>
    <cfoutput>Error. Invalid data.</cfoutput>
    <cfexit method = "ExitTag">
    <cfelse>
    <cfquery name = "AddEmployee" datasource = "cfdocexamples">
    INSERT INTO Employees
    (FirstName, LastName, Email, Phone, Department)
    VALUES
    <cfoutput>
    (
    '#StructFind(attributes.EMPINFO, "firstname")#' ,
    '#StructFind(attributes.EMPINFO, "lastname")#' ,
    '#StructFind(attributes.EMPINFO, "email")#' ,
    '#StructFind(attributes.EMPINFO, "phone")#' ,
    '#StructFind(attributes.EMPINFO, "department")#'
    )
    </cfoutput>
    </cfquery>
</cfif>
<cfoutput><hr>Employee Add Complete</cfoutput>
</cfcase>
</cfswitch> --->
```

IsUserInAnyRole

説明

認証対象のユーザーがロールリスト内のいずれかのロールに属しているかどうかを調べます。

戻り値

認証対象のユーザーがリスト内のいずれかのロールに属している場合は true、属していない場合は false

カテゴリ

セキュリティ関数、決定関数

関数のシンタックス

```
IsUserInAnyRole(role_list)
```

関連項目

cflogin、cfloginuser、cflogout、GetAuthUser、GetUserRoles、IsUserInRole、IsUserLoggedIn、『ColdFusion アプリケーションの開発』の Securing Applications

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
role_list	テストの対象となるロールをカンマで区切ったリストです。

例

```
<cfif IsUserInAnyRole(allRoles) >
    <cfoutput>Authenticated user is in these roles: #GetUserRoles()#</cfoutput>
<cfelseif >
    <cfoutput>Authenticated user is in no roles</cfoutput>
</cfif>
```

IsUserInRole

説明

認証対象のユーザーが指定されたロールに属しているかどうかを調べます。

戻り値

認証対象のユーザーが指定したロールに属している場合は true、属していない場合は false

カテゴリ

セキュリティ関数、決定関数

関数のシンタックス

```
IsUserInRole("role_name")
```

関連項目

cflogin、cfloginuser、GetAuthUser、GetUserRoles、IsUserInAnyRole、IsUserLoggedIn、『ColdFusion アプリケーションの開発』の Securing Applications

履歴

ColdFusion MX: この関数が追加されました。

パラメータ

パラメータ	説明
role_name	セキュリティロールの名前です。

使用方法

ロール名では大文字と小文字が区別されません。

ユーザーが複数のロールに属しているかどうかを調べるには、それらのロールをカンマ区切りのリストで指定します。たとえば "Admin,HR" と指定します。複数のロールを指定したリストには、区切りとしてスペースを含めることはできません。たとえば、"Admin, HR" とは指定しないでください。

例

```
<cfif IsUserInRole("Admin") >
    <cfoutput>Authenticated user is an administrator</cfoutput>
<cfelse IsUserInRole("User") >
    <cfoutput>Authenticated user is a user</cfoutput>
</cfif>
```

IsUserLoggedIn

説明

ユーザーがログインしているかどうかを調べます。

戻り値

ユーザーがログインしている場合は true、ログインしていない場合は false。

カテゴリ

セキュリティ関数、決定関数

関数のシンタックス

```
IsUserLoggedIn()
```

関連項目

cflogin、cfloginuser、GetAuthUser、GetUserRoles、IsUserInAnyRole、IsUserInRole、『ColdFusion アプリケーションの開発』の Securing Applications

履歴

ColdFusion 8: この関数が追加されました。

例

```
<cfif IsUserLoggedIn() >
    <cfinclude template="welcome.cfm">
<cfelse>
    <cfinclude template="loginform.cfm">
    <cfabort>
</cfif>
```

IsValid

説明

値が検証やデータ型のルールに準拠しているかどうかをテストします。

戻り値

値がルールに準拠している場合は true、準拠していない場合は false

カテゴリ

決定関数

関数のシンタックス

```
IsValid(type, value)
isValid("range", value, min, max)
isValid("regex" or "regular_expression", value, pattern)
```

関連項目

cfparam、cfform、IsBoolean、IsDate、IsNumeric、IsSimpleValue、『ColdFusion アプリケーションの開発』の Validating data with the IsValid function and the cfparam tag

履歴

ColdFusion 8: type 属性の値として component が追加されました。

ColdFusion MX 7: この関数が追加されました。

パラメータ

パラメータ	説明
type	有効なデータの形式です。次のいずれかです。検証アルゴリズムの詳細については、『ColdFusion アプリケーションの開発』の Validating form data using hidden fields を参照してください。 • any: 任意の単純値です。クエリーオブジェクトなどの複雑な値の場合は false を返します。IsSimpleValue 関数に相当します。 • array: ColdFusion 配列です。IsArray 関数に相当します。 • binary: バイナリ値です。IsBinary 関数に相当します。 • boolean: ブール値 (yes、no、true、false、または数値) です。IsBoolean 関数に相当します。 • component: ColdFusion コンポーネント (CFC) です。 • creditcard: mod10 アルゴリズムに準拠する 13 ～ 16 桁の数値です。 • date or time: 日付と時刻を含む、任意の日付時刻値です。IsDate 関数に相当します。 • email: 有効な電子メールアドレスです。 • eurodate: 米国の日付形式と時刻を含む、任意の日付時刻値です。 • float or numeric: 数値です。IsNumeric 関数に相当します。 • guid: Universally Unique Identifier (ユニバーサル固有識別子) です。形式は "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX" で、'X' は 16 進数です。 • integer: 整数です。 • query: クエリーオブジェクトです。IsQuery 関数に相当します。 • range: 数値の範囲です。min 属性と max 属性で指定します。 • regex または regular_expression: 入力内容を pattern 属性と照合します。 • ssn または social_security_number: 米国の社会保障番号です。 • string: シングルバイトの文字と数字で構成された文字列値です。 • struct: 構造体です。IsStruct 関数に相当します。 • telephone: 米国の標準の電話番号です。 • URL: http、https、ftp、file、mailto、または news の URL です。 • UUID: ColdFusion Universally Unique Identifier (ユニバーサル固有識別子) です。形式は 'XXXXXXXX-XXXX-XXXX-XXXXXXXXXXXXXXXX' で、'X' は 16 進数です。CreateUUID を参照してください。 • USdate: mm/dd/yy 形式の米国の日付です。日および月は 1 ～ 2 桁、年は 1 ～ 4 桁です。 • variableName: ColdFusion 変数のネーミング規則に従った形式の文字列です。 • zipcode: 米国の 5 桁または 9 桁形式の郵便番号です。 • maxlength: 最大文字数を指定します。
value	テスト対象の値です。

パラメータ	説明
min	最小有効値です。range の検証の場合にのみ使用します。
max	最大有効値です。range の検証の場合にのみ使用します。
pattern	パラメータ表記の基準となる JavaScript 正規表現です。regex または regular_expression の検証の場合にのみ使用します。

使用方法

IsValid 関数を使用することにより、検証を必ずサーバー上で実行するようにします。cfparam タグを使用して、同様に検証を実行することができます。

例

次の例では、ユーザーが数値の ID、有効な電子メールアドレス、および有効な電話番号を送信したかどうかを確認します。送信された値のいずれかが検証テストの基準を満たさない場合は、エラーメッセージが表示されます。

```
<cfif isDefined("form.saveSubmit")>
    <cfif isValid("integer", form.UserID) and isValid("email", form.emailAddr)
            and isValid("telephone", form.phoneNo)>
        <cfoutput>
            <!--- Application code to update the database goes here --->
            <h3>The email address and phone number for user #Form.UserID#
                have been added</h3>
        </cfoutput>
    <cfelse>
        <H3>You must supply a valid User ID, phone number, and email address.</H2>
    </cfif>
    <cfelse>
</cfif>

<cfform action="#CGI.SCRIPT_NAME#">
    User ID:<cfinput type="Text" name="UserID"><br>
    Phone: <cfinput type="Text" name="phoneNo"><br>
    email: <cfinput type="Text" name="emailAddr"><br>
    <cfinput type="submit" name="saveSubmit" value="Save Data"><br>
</cfform>
```

IsWDDX

説明

値が整形式の WDDX パケットであるかどうかを調べます。

戻り値

値が整形式の WDDX パケットである場合は true、整形式の WDDX パケットでない場合は false。

カテゴリ

決定関数、XML 関数

シンタックス

```
IsWDDX(value)
```

関連項目

『ColdFusion アプリケーションの開発』の Using WDDX

履歴
ColdFusion MX: 動作が変更されました。値 (value) パラメータが WDDX パケットでない場合、ColdFusion は false を返します (以前のリリースでは、ColdFusion はエラーを返していました)。

パラメータ

パラメータ	説明
value	WDDX パケットです。

使用方法
この関数は、検証のための XML パーサーを使用して WDDX パケットを処理します。このパーサーでは WDDX DTD (ドキュメントタイプ定義) が使用されます。

CFWDDX のシリアル化解除エラーを回避するため、この関数を使用して不明なソースからの WDDX パケットを検証できます。

例
```
<cfset packet="
    <wddxPacket version='1.0'>
    <header></header>
    <data>
        <struct>
        <var name='ARRAY'>
            <array length='3'>
                <string>one</string>
            <string>two</string>
            </array>
        </var>
        <var name='NUMBER'>
            <string>5</string>
        </var>
        <var name='STRING'>
            <string>hello</string>
        </var>
        </struct>
    </data>
    </wddxPacket>"
    >
<hr>
<xmp>
<cfoutput>#packet#
</xmp>
IsWDDX() returns #IsWDDX(packet)#<br>
</cfoutput>
```

IsXML

説明
文字列が整形式の XML テキストかどうかを調べます。

戻り値
関数パラメータが整形式の XML テキストを含む文字列である場合は true、そうでない場合は false。

カテゴリ
決定関数、XML 関数

関数のシンタックス
```
IsXML(value)
```

関連項目
IsXmlAttribute、IsXmlDoc、IsXmlElem、IsXmlNode、IsXmlRoot、XmlParse、XmlValidate、『ColdFusion アプリケーションの開発』の Using XML and WDDX

履歴
ColdFusion MX 7: この関数が追加されました。

パラメータ

パラメータ	説明
value	XML ドキュメントテキストを含む文字列です。

使用方法
この関数は、テキストが整形式の XML であるかどうか、つまりすべての XML シンタックスおよび構築ルールに準拠しているかどうかを調べます。文字列は完全な XML ドキュメントでなくてもかまいません。この関数は、DTD (ドキュメントタイプ定義) または XML スキーマを使用した検証は行いません。

例
次の例では、2 つの文字列を作成し、それらが整形式の XML テキストであるかどうかをテストします。

```
<!--- A well formed XML string --->
<cfset xmlString1='<order id="4323251">
    <customer firstname="Philip" lastname="Cramer" accountNum="21"/>
    <items>
        <item id="43">
            <quantity>1</quantity>
            <unitprice>15.95</unitprice>
        </item>
    </items>
</order>'
>

<!--- An invalid XML string, missing the </item> close tag --->
<cfset xmlString2='<order id="4323251">
    <customer firstname="Philip" lastname="Cramer" accountNum="21"/>
    <items>
        <item id="43">
            <quantity>1</quantity>
            <unitprice>15.95</unitprice>
    </items>
</order>'
>

<!--- Test the strings to see if they are well formed XML --->
<cfoutput>
xmlString1 contains the following text:<br><br>
#HTMLCodeFormat(xmlstring1)#
Is it well formed XML text? #IsXML(xmlString1)#<br><br>
<hr>
xmlString2 contains the following text:<br><br>
#HTMLCodeFormat(xmlstring2)#
Is it well formed XML text? #IsXML(xmlString2)#
</cfoutput>
```

IsXmlAttribute

説明
関数パラメータが XML DOM (Document Object Model) 属性ノードかどうかを調べます。

戻り値
関数の引数が XML 属性ノードの場合は true、そうでない場合は false。

カテゴリ
決定関数、XML 関数

関数のシンタックス
```
IsXmlAttribute(value)
```

関連項目
IsXML、IsXmlDoc、IsXmlElem、IsXmlNode、IsXmlRoot、XmlGetNodeType、XmlValidate、『ColdFusion アプリケーションの開発』の Using XML and WDDX

履歴
ColdFusion MX 7: この関数が追加されました。

パラメータ

パラメータ	説明
value	XML 属性の名前です。

使用方法

この関数は、パラメータが XML DOM 属性ノードであるかどうか、つまり ATTRIBUTE の値が XMLType のノードであるかどうかを調べます。XmlSearch 関数が返す値が XML 属性であるかどうかを確認する場合に役立ちます。

DOM および ColdFusion は、XML 属性を要素のプロパティとして処理します。DOM ノードとして直接示すことはありません。したがって、ColdFusion XML ドキュメントオブジェクト内の XmlAttributes エントリは、DOM 属性ノードを示さず、次のようなテストでは常に false が返されます。

```
IsXmlAttribute(myxmlelement.XMlAttributes);
IsXmlAttribute(myxmlelement.XMlAttributes.myattribute);
```

XmlSearch 関数は、属性を XML DOM 属性ノードとして返します。たとえば次の行は、xmlobject ドキュメントオブジェクト内の quantity 属性を含む属性ノードの配列を返します。

```
quantities = XmlSearch(xmlobject, '//@quantity');
```

例

次の例では、XML ドキュメントを作成して、そのパーツを取得します。次に、それらのパーツが属性ノードであるかどうかをテストします。

```
<!--- Create an XML document object --->
<cfxml variable="xmlobject">
<order id="4323251">
    <customer firstname="Philip" lastname="Cramer" accountNum="21"/>
    <items>
        <item id="43">
            <quantity>1</quantity>
            <unitprice>15.95</unitprice>
        </item>
    </items>
</order>
</cfxml>

<!--- Get an array with all lastname quantity DOM attribute nodes
    (In this example there is only one entry) --->
<cfset lastnames = XmlSearch(xmlobject, '//@lastname')>

<!--- Test objects to see if they are attributes --->
<cfoutput>
<h3>Are the following XML Attribute nodes?</h3>
<!--- The order element id attribute.
         This a simple variable, not a DOM attribute node.--->
node.xmlobject.order.XmlAttributes.id:
        #IsXmlAttribute(xmlobject.order.XmlAttributes.id)#<br>
<!--- The items element --->
xmlobject.order.items: #IsXmlAttribute(xmlobject.order.items)#<br>
lastnames[1] returned by XmlSearch:
    #isXmlAttribute(lastnames[1])#<br>
</cfoutput>
```

IsXmlDoc

説明
関数パラメータが ColdFusion XML ドキュメントオブジェクトであるかどうかを調べます。

戻り値
関数の引数が XML ドキュメントオブジェクトの場合は true、そうでない場合は false。

カテゴリ
決定関数、XML 関数

関数のシンタックス
```
IsXmlDoc(value)
```

関連項目
IsXML、IsXmlAttribute、IsXmlElem、IsXmlNode、IsXmlRoot、XmlValidate、『ColdFusion アプリケーションの開発』の Using XML and WDDX

履歴
ColdFusion MX: この関数が追加されました。

パラメータ

パラメータ	説明
value	XML ドキュメントオブジェクトの名前です。

例
次の例では、XML ドキュメントオブジェクトと Java オブジェクトを作成し、それらが XML ドキュメントオブジェクトであるかどうかをテストします。

```
<!--- Create an XML document object --->
<cfxml variable="xmlobject">
<order id="4323251">
    <customer firstname="Philip" lastname="Cramer" accountNum="21"/>
    <items>
        <item id="43">
            <quantity>1</quantity>
            <unitprice>15.95</unitprice>
        </item>
    </items>
</order>
</cfxml>

<!--- Create a Java object --->
<cfobject type="JAVA" action="create" class="java.lang.Error" name="javaobject" >

<!--- Test the objects --->
<cfoutput>
Is xmlobject an XML document object? #IsXmlDoc(xmlobject)#<br>
Is javaobject an XML document object? #IsXmlDoc(javaobject)#<br>
</cfoutput>
```

IsXmlElem

説明
関数パラメータが XML ドキュメントオブジェクトの要素であるかどうかを調べます。

戻り値
関数の引数が XML ドキュメントオブジェクトの要素の場合は true、そうでない場合は false。

カテゴリ
決定関数、XML 関数

関数のシンタックス
```
IsXmlElem(value)
```

関連項目
IsXML、IsXmlAttribute、IsXmlDoc、IsXmlNode、IsXmlRoot、XmlGetNodeType、XmlValidate、『ColdFusion アプリケーションの開発』の Using XML and WDDX

履歴
ColdFusion MX: この関数が追加されました。

パラメータ

パラメータ	説明
value	XML ドキュメントオブジェクトの要素の名前です。

例
次の例では、XML ドキュメントオブジェクト、ドキュメントルート、および要素が要素であるかどうかをテストします。

```
<!--- Create an XML document object --->
<cfxml variable="xmlobject">
<order id="4323251">
    <customer firstname="Philip" lastname="Cramer" accountNum="21"/>
    <items>
        <item id="43">
            <quantity>1</quantity>
            <unitprice>15.95</unitprice>
        </item>
    </items>
</order>
</cfxml>

<!--- Test parts of the document object to see if they are elements --->
<cfoutput>
    <h3>Are the following XML document object elements?</h3>
    xmlobject: #IsXmlElem(xmlobject)#<br>
    xmlobject.XMLRoot: #IsXmlElem(xmlobject.XMLRoot)#<br>
    xmlobject.order.items: #IsXmlElem(xmlobject.order.items)#<br>
</cfoutput>
```

IsXmlNode

説明
関数パラメータが XML ドキュメントオブジェクトのノードかどうかを調べます。

戻り値
関数の引数が XML ドキュメントオブジェクトのノードである場合は true、そうでない場合は false。

カテゴリ
決定関数、XML 関数

関数のシンタックス
```
IsXmlNode(value)
```

関連項目
IsXML、IsXmlAttribute、IsXmlDoc、IsXmlElem、IsXmlRoot、XmlGetNodeType、XmlSearch、XmlValidate、『ColdFusion アプリケーションの開発』の Using XML and WDDX

履歴
ColdFusion MX 7: この関数が追加されました。

パラメータ

パラメータ	説明
value	XML ドキュメントオブジェクトのノードの名前です。

使用方法
この関数は、XML ドキュメントオブジェクトの次のコンポーネントについて true を返します。

- ドキュメントオブジェクト
- オブジェクト内の要素
- 要素の XmlNodes 配列内の XmlNode オブジェクト

  この関数は、XmlSearch 関数によって返された XML ノードオブジェクトに対しても true を返します。ただし、XmlText、XmlComment、XmlCdata、XmlAttributes 配列 (または個々の XML 属性) など、要素内のほとんどのエントリに対しては true を返しません。

例
次の例では、XML ドキュメントオブジェクト、要素、オブジェクト内の属性、および XmlSearch 関数が返す属性がノードであるかどうかをテストします。

```
<!--- Create an XML document object --->
<cfxml variable="xmlobject">
<?xml version="1.0" encoding="UTF-8"?>
<order id="4323251">
    <customer firstname="Philip" lastname="Cramer" accountNum="21"/>
    <items>
        <item id="43">
            <quantity>1</quantity>
            <unitprice>15.95</unitprice>
        </item>
    </items>
</order>
</cfxml>

<!--- use XmlSearch to get an attribute node. --->
<cfset lastnames = XmlSearch(xmlobject, '//@lastname')>

<!--- Test the objects to see if they are XML nodes--->
<cfoutput>
<h3>Are the following XML nodes?</h3>
xmlobject: #IsXmlNode(xmlobject)#<br>
<!--- The items element --->
xmlobject.order.items: #IsXmlNode(xmlobject.order.items)#<br>
<!--- The order element id attribute; a simple variable, not a DOM node.--->
xmlobject.order.XmlAttributes.id:
    #IsXmlNode(xmlobject.order.XmlAttributes.id)#<br>
lastnames[1] returned by XmlSearch:
#isXmlNode(lastnames[1])#
</cfoutput>
```

IsXmlRoot

説明
関数パラメータが XML ドキュメントオブジェクトのルート要素であるかどうかを調べます。

戻り値
関数の引数が XML ドキュメントオブジェクトのルートオブジェクトである場合は true、そうでない場合は false。

カテゴリ
決定関数、XML 関数

関数のシンタックス
```
IsXmlRoot(value)
```

関連項目
IsXML、IsXmlAttribute、IsXmlDoc、IsXmlElem、IsXmlNode、XmlGetNodeType、XmlValidate、『ColdFusion アプリケーションの開発』の Using XML and WDDX

履歴
ColdFusion MX: この関数が追加されました。

パラメータ

パラメータ	説明
value	XML ドキュメントオブジェクトの名前です。

例

次の例では、XML ドキュメント、そのルート要素、および子要素が XML ルート要素であるかどうかをテストします。

```
<!--- Create an XML document object --->
<cfxml variable="xmlobject">
<?xml version="1.0" encoding="UTF-8"?>
<order id="4323251">
    <customer firstname="Philip" lastname="Cramer" accountNum="21"/>
    <items>
        <item id="43">
            <quantity>1</quantity>
            <unitprice>15.95</unitprice>
        </item>
    </items>
</order>
</cfxml>

<!--- Test objects to see if they are XML root elements --->
<cfoutput>
<h3>Are the following the XML Root?</h3>
xmlobject: #IsXmlRoot(xmlobject)#<br>
xmlobject.order: #IsXmlRoot(xmlobject.order)#<br>
<!--- The order element id attribute --->
xmlobject.order.XmlAttributes.id:
    #IsXmlRoot(xmlobject.order.XmlAttributes.id)#<br>
</cfoutput>
```

JavaCast

説明

Java オブジェクトまたは .NET オブジェクトに引数として渡すために、ColdFusion 変数のデータ型を指定された Java 型に変換します。スカラー、文字列、および配列の引数のみに使用します。

戻り値

type で指定された型の変数

カテゴリ

文字列関数

関数のシンタックス

```
JavaCast(type, variable)
```

履歴

ColdFusion MX 8: bigdecimal、byte、char、および short データ型に対するサポートと、配列のキャストに対するサポートが追加されました。

ColdFusion MX 7: null に対するサポートが追加されました。

関連項目

CreateObject、cfobject、『ColdFusion アプリケーションの開発』の Converting between .NET and ColdFusion data types および Java and ColdFusion data type conversions

パラメータ

パラメータ	説明
type	変換後の変数のデータ型です。 • bigdecimal (java.math.BigDecimal に変換) • boolean • byte • char • int • long • float • double • short • string • null • **xxx**[] **xxx** の部分では、次のいずれかを指定します。 • 上記のいずれかの型 (null を除く) • Java クラス名
variable	スカラーまたは文字列のタイプを保持する ColdFusion 変数です。type が null の場合、"" でなければなりません。

使用方法

このメソッドを使用すると、型の変換処理が不明瞭な場合に Java または .NET メソッドの呼び出すときに使用する変数の Java データ型を指定できます。たとえば、メソッドがオーバーロードされていてパラメータの型のみが異なる場合や、.NET メソッドが System.Object クラスのパラメータを取るように宣言されている場合などに使用できます。

cfobject タグで Java オブジェクトを作成してから、そのメソッドの 1 つを呼び出すまでの間に使用します。メソッドがオーバーロードされた引数を複数取る場合は、各引数について JavaCast を呼び出します。JavaCast は、メソッドがオーバーロードされている場合にのみ使用します。これは、メソッドがさまざまな数の引数を取るためではなく、引数が複数のデータ型を取るためです。

JavaCast は、複雑なオブジェクト間のキャストや、上位クラスへのキャストには使用できません。

内部に保管された ColdFusion 型と Java スカラー型の間に 1 対 1 の対応関係がないために、実行できない変換もあります。

この関数の結果は、Java または .NET オブジェクトの呼び出しにのみ使用します。次の例では、Java メソッドを呼び出す際の使い方を示します。

```
<cfscript>
     x = CreateObject("java", "test.Hello");
     x.init();
     ret = x.sayHello(JavaCast("null", ""));
</cfscript>
```

注意：JavaCast("null","") の結果を ColdFusion 変数に割り当てないでください。予期しない結果が生じます。

JavaCast(**type**[], **variable**) という形式を使用すると、指定された型の 1 次元配列に ColdFusion 配列変数がキャストされます。多次元配列には変換できません。プリミティブ型またはクラス名をキャスト先の型として指定できます。たとえば、次の形式を使用すると、ColdFusion 配列を vom.x.yMyClass オブジェクトの配列にキャストできます。

```
javacast("vom.x.y.MyClass[]", myCFArr)
```

次のような場合は、JavaCast の最初のパラメータで配列を使用します。

- パラメータ数が同じでシグネチャが異なる 2 つの関数があり、それぞれのシグネチャにおいてパラメータが取る配列の型が異なる場合 (たとえば、foo(int[] x) 関数と foo(String[] strs) 関数がある場合)。
- メソッドパラメータのシグネチャでクラス配列が要求されている場合 (例 : foo(com.x.y.MyClass[]))。
- メソッドパラメータのシグネチャでオブジェクトが要求されていて、特定の型の配列を渡す必要がある場合。

  次の例では、JavaCast 関数を使用して配列をキャストする方法を示します。

  次の 2 つのメソッドが定義されている fooClass クラスがあるとします (それぞれ、第 1 引数の配列の型が異なります)。

```
public class fooClass {
    public fooClass () {
    }
    public String foo(long[] arg) {
return "Argument was a long array";
    }
    public String foo(int[] arg) {
return "Argument was an Integer array";
    }
}
```

  これらの関数を CFML で使用するには、次のコード例に示すように、JavaCast 関数を使用して ColdFusion 配列を適切な配列型に変換します。

```
<cfset arr = [1,2,4,20,10]>
<cfset fooObj = createObject("java", "fooClass")>

<cfset fooObj.foo(javacasr("int[]", arr))>
<cfset fooObj.foo(javacast("long[]", arr))>
```

例

fooClass クラスのメソッド fooMethod は、オーバーロードされた 1 つの引数を取ります。fooClass クラスは次のように定義されています。

```
public class fooClass {
    public fooClass () {
    }
    public String fooMethod(String arg) {
        return "Argument was a String";
    }
    public String fooMethod(int arg) {
        return "Argument was an Integer";
    }
}
```

ColdFusion では、次のコードを使用します。

```
<cfobject
action="create"
type = "java"
class = "fooClass"
name = obj>

<!--- ColdFusion can treat this as a string or a real number --->
<cfset x = 33>

Perform an explicit cast to an int and call fooMethod:<br>
<cfset myInt = JavaCast("int", x)>
<cfoutput>#obj.fooMethod(myInt)#</cfoutput>
<br><br>
Perform an explicit cast to a string and call fooMethod:<br>
<cfset myString = javaCast("String", x)>
<cfoutput>#obj.fooMethod(myString)#</cfoutput>
```

JSStringFormat

説明
引用符 (')、二重引用符 (")、および改行などの JavaScript 特殊文字をエスケープします。

戻り値
JavaScript で安全に使用できる文字列

カテゴリ
文字列関数

関数のシンタックス
```
JSStringFormat(string)
```

パラメータ

パラメータ	説明
string	文字列、または文字列を含んでいる変数です。

使用方法
この関数により JavaScript の特殊文字がエスケープされるため、JavaScript の中に任意の文字列を安全に入力できます。

例
```
<!--- This example shows the use of the JSStringFormat function. ---->
<h3>JSStringFormat</h3>
<cfset stringValue = "An example string value with a tab chr(8),
    a newline (chr10) and some ""quoted"" 'text'">

<p>This is the string we have created:<br>
<cfoutput>#stringValue#</cfoutput>
</p>
<cfset jsStringValue = JSStringFormat(#stringValue#)>
<!----- Generate an alert from the JavaScript string jsStringValue. ---->
<SCRIPT>
s = "<cfoutput>#jsStringValue#</cfoutput>";
alert(s);
</SCRIPT>
```

関数 I

LCase

説明
文字列のアルファベット文字を小文字に変換します。

戻り値
小文字に変換した文字列

カテゴリ
文字列関数

関数のシンタックス
```
LCase(string)
```

関連項目
UCase

パラメータ

パラメータ	説明
string	文字列、または文字列を含んでいる変数です。

例
```
<h3>LCase Example</h3>

<cfif IsDefined("FORM.sampleText")>
    <cfif FORM.sampleText is not "">
        <cfoutput>
        <p>Your text, <b>#FORM.sampleText#</b>, returned in lowercase is
        <b>#LCase(FORM.sampleText)#</b>.</p>
    <   /cfoutput>
    <cfelse>
        <   p><b><i>Please enter some text.</i></b></p>
    </cfif>
</cfif>

<p>Enter your text. Press "submit" to see it returned in lowercase: </p>

<form method="post" action = "<cfoutput>#cgi.script_name#</cfoutput>" name="lcaseForm">
<input type = "Text" name = "SampleText" value = "SAMPLE">
<input type = "Submit" name = "" value = "submit">
</form>
```

Left

説明
文字列の左端から、**count** パラメーターで指定した文字数分の文字列を返します。

戻り値
string パラメーターの先頭から **count** で指定した文字数分の文字列

カテゴリ
文字列関数

関数のシンタックス
```
Left(string, count)
```

関連項目
Right、Mid、Len

パラメータ

パラメータ	説明
string	文字列、または文字列を含んでいる変数です。
count	正の整数、または正の整数を含んでいる変数です。返す最大文字数を指定します。

例
```
<h3>Left Example</h3>

<cfif IsDefined("Form.myText")>
<!--- If len returns 0 (zero), then show error message. --->
    <cfif Len(Form.myText)>
        <cfif Len(Form.myText) LTE Form.RemoveChars>
            <cfoutput><p style="color: red; font-weight: bold">Your string #Form.myText# only has
#Len(Form.myText)# characters. You cannot output
    the #Form.removeChars# leftmost characters of this string because it is
    not long enough.</p></cfoutput>
        <cfelse>
            <cfoutput><p>Your original string: <strong>#Form.myText#</strong></p>
                <p>Your changed string, showing only the <strong>#Form.removeChars#
                    </strong> leftmost characters:
                    <strong>#Left(Form.myText, Form.removeChars)#</strong></p>
            </cfoutput>
        </cfif>
    <cfelse>
        <p style="color: red; font-weight: bold">Please enter a string of more than 0 (zero) characters.</p>
</cfif>
</cfif>

<form action="<cfoutput>#CGI.ScriptName#</cfoutput>" method="POST">
<p>Type in some text<br />
<input type="Text" name="myText"></p>
<p>How many characters from the left do you want to show?
<select name="RemoveChars">
<option value="1">1
<option value="3" selected>3
<option value="5">5
<option value="7">7
<option value="9">9</select>
<input type="Submit" name="Submit" value="Remove characters"></p>
</form>
```

Len

説明
文字列またはバイナリオブジェクトの長さを調べます。

戻り値
文字列またはバイナリオブジェクトの長さを表す数値

カテゴリ
文字列関数

関数のシンタックス
```
Len(string or binary object)
```

関連項目
ToBinary、Left、Right、Mid

履歴
ColdFusion MX: Unicode サポートが変更されました。Unicode 文字値 0 ～ 65535 の Java UCS-2 表現がサポートされるようになりました。ColdFusion 5 以前のリリースでは ASCII 値 1 ～ 255 がサポートされていました。長さを計算する場合、一部の文字列処理関数は、ASCII 0 (NUL) 文字を処理し、それに続く文字列内の文字を処理しないようになっていました。

パラメータ

パラメータ	説明
string	文字列、文字列名、またはバイナリオブジェクトです。

例
```
<h3>Len Example</h3>

<cfif IsDefined("Form.MyText")>
    <!--- If len returns 0 (zero), then show error message. --->
    <cfif Len(FORM.myText)>
        <cfoutput><p>Your string, <strong>"#FORM.myText#"</strong>,
        has <strong>#Len(FORM.myText)#</strong> characters.</cfoutput>
    <cfelse>
        <p style="color: red; font-weight: bold">Please enter a string of more
        than 0 characters.</p>
    </cfif>
</cfif>

<form action = "<cfoutput>#CGI.SCRIPT_NAME#</cfoutput>" method="POST">
<p>Type in some text to see the length of your string.</p>

<input type = "Text" name = "MyText"><br />
<input type = "Submit" name="Submit" value = "Count characters"><br>
</form>
```

ListAppend

説明
リストまたは要素をリストに付加します。

戻り値
value が付加されたリストのコピー。value = "" の場合は、コピーのリストがそのまま返されます。

カテゴリ
リスト関数

関数のシンタックス
```
ListAppend(list, value [, delimiters ])
```

関連項目
ListPrepend、ListInsertAt、ListGetAt、ListLast、ListSetAt、『ColdFusion アプリケーションの開発』の Lists

パラメータ

パラメータ	説明
list	リスト、またはリストを含んでいる変数です。
value	要素または要素のリストです。
delimiters	文字列、または文字列を含んでいる変数です。リスト要素を区切る文字です。デフォルト値はカンマ (,) です。 このパラメータに複数の文字が含まれている場合は、最初の文字だけが使用されます。

使用方法
ColdFusion では、**value** の前に区切り文字が挿入されます。

次の表は、ListAppend 処理の例を示しています。

ステートメント	出力	コメント
ListAppend('elem1,elem2', '')	elem1,elem2,	付加される要素は空です。リストの末尾に区切り文字が付加されます。リスト長は 2 です。
ListAppend('', 'elem1,elem2')	elem1,elem2	リスト長は 2 です。
ListAppend("one___two", "three", "___")	"one___two_three"	"three" の前に、区切り文字として delimiters の最初の文字が挿入されます。

例
```
<h3>ListAppend Example</h3>
<!--- First, query to get some values for our list elements--->
<cfquery name = "GetParkInfo" datasource = "cfdocexamples">
    SELECT PARKNAME,CITY,STATE
    FROM PARKS WHERE PARKNAME LIKE 'AL%'
</cfquery>
<cfset temp = ValueList(GetParkInfo.ParkName)>
<cfoutput>
<p>The original list: #temp#
</cfoutput>
<!--- now, append a park name to the list --->
<cfset temp2 = ListAppend(Temp, "ANOTHER PARK")>
```

ListChangeDelims

説明
リストの区切り文字を変更します。

戻り値
各区切り文字が **new_delimiter** に置き換えられたリストのコピー

カテゴリ
リスト関数

関数のシンタックス
```
ListChangeDelims(list, new_delimiter [, delimiters, includeEmptyValues ])
```

関連項目
ListFirst、ListQualify、『ColdFusion アプリケーションの開発』の Lists

パラメータ

パラメータ	説明
includeEmptyValues	オプション。空の値を含めるには、このパラメータを yes に設定します。
list	リスト、またはリストを含んでいる変数です。
new_delimiter	区切り文字、または区切り文字を含んでいる変数です。空の文字列も指定できます。この文字列全体が 1 つの区切り文字として処理されます。
delimiters	文字列、または文字列を含んでいる変数です。リスト要素を区切る文字です。デフォルト値はカンマ (,) です。 このパラメータに複数の文字が含まれている場合、それぞれの文字が出現するすべての箇所が区切り文字として処理されます。

例
```
<h3>ListChangeDelims Example</h3>
<p>ListChangeDelims lets you change the delimiters of a list.
<!--- First, query to get some values for our list elements--->
<CFQUERY NAME="GetParkInfo" DATASOURCE="cfdocexamples">
    SELECT PARKNAME,CITY,STATE
    FROM Parks
    WHERE PARKNAME LIKE 'BA%'
</CFQUERY>
<CFSET temp = ValueList(GetParkInfo.ParkName)>
<cfoutput>
<p>The original list: <p>#temp#
</cfoutput>
<!--- Change the delimiters in the list --->
<CFSET temp2 = ListChangeDelims(Temp, "|:P|", ",")>
<cfoutput>
<p>After executing the statement
    <strong>ListChangeDelims(Temp, "|:P|", ",")</strong>,
    the updated list: <p>#temp2#
</cfoutput>
```

ListContains

説明
指定された部分文字列を含んでいる最初のリスト要素のインデックスを調べます。

戻り値
substring を含んでいる最初のリスト要素のインデックス。検出されなかった場合は 0 が返されます。

カテゴリ
リスト関数

関数のシンタックス
```
ListContains(list, substring [, delimiters, includeEmptyValues ])
```

関連項目
ListContainsNoCase、ListFind、『ColdFusion アプリケーションの開発』の Lists

パラメータ

パラメータ	説明
includeEmptyValues	オプション。空の値を含めるには、このパラメータを yes に設定します。
list	リスト、またはリストを含んでいる変数です。
substring	文字列、または文字列を含んでいる変数です。この検索では大文字と小文字が区別されます。
delimiters	文字列、または文字列を含んでいる変数です。リスト要素を区切る文字です。デフォルト値はカンマ (,) です。 このパラメータに複数の文字が含まれている場合、それぞれの文字が出現するすべての箇所が区切り文字として処理されます。

使用方法
ColdFusion では空のリスト要素が無視されるため、"a,b,c,,,d" の場合は 4 つの要素があることになります。

例

```
<!--- This example shows differences between ListContains and ListFind --->
<!--- Create a list composed of the elements one, two, three. ----->
<cfset aList = "one">
<cfset aList = ListAppend(aList, "two")>
<cfset aList = ListAppend(aList, "three")>
<p>Here is a list: <cfoutput>#aList#</cfoutput>
<p><strong>ListContains</strong> checks for substring "wo" in the list elements:
<cfoutput>
    <p>   Substring "wo" is in
    <B>element #ListContains(aList, "wo")#</B> of list.
</cfoutput>
<p>ListFind cannot check for a substring within an element; therefore, in the
    code, it does not find substring "wo" (it returns 0):
<cfoutput>
    <p>   Substring "wo" is in <b>element #ListFind(aList, "wo")#
    </b> of the list.</cfoutput>
<p><p>If you specify a string that exactly equals an entire list element, such
    as "two", both ListContains and ListFind find it, in the second element:
    <p>   <strong>ListContains</strong>:
<cfoutput>
The string "two" is in <b>element #ListContains(aList, "two")#</b> of the list.
</cfoutput>
   <p><strong>ListFind</strong>:
<cfoutput>
The string "two" is in <b>element #ListFind(aList, "two")#</b> of the list.
</cfoutput>
```

ListContainsNoCase

説明

指定された部分文字列を含んでいる最初のリスト要素のインデックスを調べます。

戻り値

substring を含んでいる最初のリスト要素のインデックス。大文字と小文字は区別されません。検出されなかった場合は 0 が返されます。

カテゴリ

リスト関数

関数のシンタックス

```
ListContainsNoCase(list, substring [, delimiters, includeEmptyValues ])
```

関連項目

ListContains、ListFindNoCase、『ColdFusion アプリケーションの開発』の Lists

パラメータ

パラメータ	説明
includeEmptyValues	オプション。空の値を含めるには、このパラメータを yes に設定します。
list	リスト、またはリストを含んでいる変数です。
substring	文字列、または文字列を含んでいる変数です。この検索では大文字と小文字は区別されません。
delimiters	文字列、または文字列を含んでいる変数です。リスト要素を区切る文字です。デフォルト値はカンマ (,) です。 このパラメータに複数の文字が含まれている場合、それぞれの文字が出現するすべての箇所が区切り文字として処理されます。

使用方法
ColdFusion では空のリスト要素が無視されるため、"a,b,c,,,d" の場合は 4 つの要素があることになります。

例
```
<h3>ListContainsNoCase Example</h3>
<cfif IsDefined("form.letter")>
    <!--- First, query to get some values for our list --->
    <cfquery name="GetParkInfo" datasource="cfdocexamples">
        SELECT PARKNAME,CITY,STATE
        FROM Parks
        WHERE PARKNAME LIKE '#form.letter#%'
    </cfquery>
    <cfset tempList = #ValueList(GetParkInfo.City)#>
    <cfif ListContainsNoCase(tempList, form.yourCity) is not 0>
        There are parks in your city!
    <cfelse>
        <p>Sorry, there were no parks found for your city.
        Try searching under a different letter.
    </cfif>
</cfif>
```

ListDeleteAt

説明
リストから要素を削除します。

戻り値
指定された要素を削除したリストのコピー

カテゴリ
リスト関数

関数のシンタックス
```
ListDeleteAt(list, position [, delimiters ])
ListDeleteAt(list, position [, delimiters, includeEmptyValues ])
```

関連項目
ListGetAt、ListSetAt、ListLen、『ColdFusion アプリケーションの開発』の Lists

パラメータ

パラメータ	説明
list	リスト、またはリストを含んでいる変数です。
position	正の整数、または正の整数を含んでいる変数です。要素を削除する位置を指定します。リスト内の先頭を示す位置は 1 です。
delimiters	文字列、または文字列を含んでいる変数です。リスト要素を区切る文字です。デフォルト値はカンマ (,) です。 このパラメータに複数の文字が含まれている場合、それぞれの文字が出現するすべての箇所が区切り文字として処理されます。
includeEmptyValues	includeEmptyValues を false に設定すると、リストには空でない要素のみが含まれ、指定した位置の要素が削除されます。

使用方法

この関数や他の関数にデフォルトの区切り文字 (カンマ) を使用するには、次のコードを記述します。

```
<cfset temp2 = ListDeleteAt(temp, "3")>
```

別の区切り文字を指定するには、次のコードを記述します。

```
<cfset temp2 = ListDeleteAt(temp, "3", ";")>
```

ColdFusion では空のリスト要素が無視されるため、"a,b,c,,,d" の場合は 4 つの要素があることになります。

例

```
<!--- First, query to get some values for our list elements. --->
<CFQUERY NAME="GetParkInfo" DATASOURCE="cfdocexamples">
    SELECT PARKNAME,CITY,STATE
    FROM Parks
    WHERE PARKNAME LIKE 'CHI%'
</CFQUERY>
<CFSET temp = ValueList(GetParkInfo.ParkName)>
<CFSET deleted_element = ListGetAt(temp, "3", ",")>
<cfoutput><p>The original list: #temp#</p></cfoutput>
<!--- Delete the third element from the list. --->
<CFSET temp2 = ListDeleteAt(Temp, "3")>
<cfoutput>
<p>The changed list: #temp2#
<p><I>This list element:<br>#deleted_element#<br> is no longer present
    at position three of the list.</I> </cfoutput>
```

ListFilter

説明

リスト内の要素をフィルタリングする場合に使用します。

戻り値

新しいリスト

カテゴリ

クロージャ関数

シンタックス

listFilter(list,function(***listElement***){return true|false;});

関連項目
その他のクロージャ関数

履歴
ColdFusion 10: この関数が追加されました。

パラメータ

パラメータ	説明
list	リストオブジェクトの名前です。
function	リスト内の各要素に対して実行されるインライン関数です。結果のリストにリスト要素が含まれる場合は true が返されます。
listElement	アクセスするリスト要素です。

ListFind

説明
指定された値が出現する最初のリスト要素のインデックスを調べます。大文字と小文字は区別されます。

戻り値
value を含んでいる最初のリスト要素のインデックス。大文字と小文字は区別されます。検出されなかった場合は 0 が返されます。この検索では大文字と小文字が区別されます。

カテゴリ
リスト関数

関数のシンタックス
```
ListFind(list, value [, delimiters, includeEmptyValues ])
```

関連項目
ListContains、ListFindNoCase、『ColdFusion アプリケーションの開発』の Lists

パラメータ

パラメータ	説明
includeEmptyValues	オプション。空の値を含めるには、このパラメータを yes に設定します。
list	リスト、またはリストを含んでいる変数です。
value	文字列、数字、あるいは、文字列または数字を含んでいる変数です。検索する項目を指定します。この検索では大文字と小文字が区別されます。
delimiters	文字列、または文字列を含んでいる変数です。リスト要素を区切る文字です。デフォルト値はカンマ (,) です。 このパラメータに複数の文字が含まれている場合、それぞれの文字が出現するすべての箇所が区切り文字として処理されます。

使用方法
ColdFusion では空のリスト要素が無視されるため、"a,b,c,,,d" の場合は 4 つの要素があることになります。

例

```
<!--- Uses ListFind and ListFindNoCase to see if a substring exists
    in a list --->
<form action="./listfind.cfm" method="POST">
    <p>Try changing the case in Leary's last name:
    <br><input type="Text" size="25" name="myString" value="Leary">
    <p>Pick a search type:
        <select name="type">
            <option value="ListFind" selected>Case-Sensitive
            <option value="ListFindNoCase">Case-Insensitive
        </select>
    <input type="Submit" name="" value="Search Employee List">
</form>

<!--- wait to have a string for searching defined --->
<cfif IsDefined("form.myString") and IsDefined("form.type")>

<cfquery name="SearchEmpLastName" datasource="cfdocexamples">
    SELECT FirstName, RTrim(LastName) AS LName, Phone, Department
    FROM Employees
</cfquery>

<cfset myList = ValueList(SearchEmpLastName.LName)>
<!--- Is this case-sensitive or case-insensitive searching --->
<cfif form.type is "ListFind">
    <cfset temp = ListFind(myList, form.myString)>
        <cfif temp is 0>
            <h3>An employee with that exact last name was not found</h3>
        <cfelse>
            <cfoutput>
            <p>Employee #ListGetAt(ValueList(SearchEmpLastName.FirstName), temp)#
            #ListGetAt(ValueList(SearchEmpLastName.LName), temp)#, of the
            #ListGetAt(ValueList(SearchEmpLastName.Department), temp)# Department,
            can be reached at #ListGetAt(ValueList(SearchEmpLastName.Phone),
            temp)#.
            <p>This was the first employee found under this case-sensitive last name
            search.
            </cfoutput>
        </cfif>
        <cfelse>
            <cfset temp = ListFindNoCase(myList, form.myString)>
            <cfif temp is 0>
                <h3>An employee with that exact last name was not found</h3>
            <cfelse>
                <cfoutput>
                <p>Employee #ListGetAt(ValueList(SearchEmpLastName.FirstName), temp)#
                #ListGetAt(ValueList(SearchEmpLastName.LName), temp)#, of the
                #ListGetAt(ValueList(SearchEmpLastName.Department), temp)#
                Department, can be reached at
                #ListGetAt(ValueList(SearchEmpLastName.Phone), temp)#.
                <p>This was the first employee found under this case-insensitive last
                name search.
            </cfoutput>
        </cfif>
    </cfif>
</cfif>
```

ListFindNoCase

説明
指定された値が出現する最初のリスト要素のインデックスを調べます。

戻り値
value を含んでいる最初のリスト要素のインデックス。検出されなかった場合は 0 が返されます。この検索では大文字と小文字は区別されません。

カテゴリ
リスト関数

関数のシンタックス
```
ListFindNoCase(list, value [, delimiters, includeEmptyValues ])
```

関連項目
ListContains、ListFind、『ColdFusion アプリケーションの開発』の Lists

パラメータ

パラメータ	説明
includeEmptyValues	オプション。空の値を含めるには、このパラメータを yes に設定します。
list	リスト、またはリストを含んでいる変数です。
value	検索する数値または文字列です。この検索では大文字と小文字は区別されません。
delimiters	文字列、または文字列を含んでいる変数です。リスト要素を区切る文字です。デフォルト値はカンマ (,) です。 このパラメータに複数の文字が含まれている場合、それぞれの文字が出現するすべての箇所が区切り文字として処理されます。

使用方法
ColdFusion では空のリスト要素が無視されるため、"a,b,c,,,d" の場合は 4 つの要素があることになります。

例

```
<!--- Uses ListFind and ListFindNoCase to see if a substring exists
    in a list --->
<form action="./listfind.cfm" method="POST">
    <p>Try changing the case in Leary's last name:
    <br><input type="Text" size="25" name="myString" value="Leary">
    <p>Pick a search type:
        <select name="type">
            <option value="ListFind" selected>Case-Sensitive
            <option value="ListFindNoCase">Case-Insensitive
        </select>
    <input type="Submit" name="" value="Search Employee List">
</form>

<!--- wait to have a string for searching defined --->
<cfif IsDefined("form.myString") and IsDefined("form.type")>

<cfquery name="SearchEmpLastName" datasource="cfdocexamples">
    SELECT FirstName, RTrim(LastName) AS LName, Phone, Department
    FROM Employees
</cfquery>

<cfset myList = ValueList(SearchEmpLastName.LName)>
<!--- Is this case-sensitive or case-insensitive searching --->
<cfif form.type is "ListFind">
    <cfset temp = ListFind(myList, form.myString)>
        <cfif temp is 0>
            <h3>An employee with that exact last name was not found</h3>
        <cfelse>
            <cfoutput>
            <p>Employee #ListGetAt(ValueList(SearchEmpLastName.FirstName), temp)#
            #ListGetAt(ValueList(SearchEmpLastName.LName), temp)#, of the
            #ListGetAt(ValueList(SearchEmpLastName.Department), temp)# Department,
            can be reached at #ListGetAt(ValueList(SearchEmpLastName.Phone),
            temp)#.
            <p>This was the first employee found under this case-sensitive last name
            search.
            </cfoutput>
        </cfif>
        <cfelse>
            <cfset temp = ListFindNoCase(myList, form.myString)>
            <cfif temp is 0>
                <h3>An employee with that exact last name was not found</h3>
            <cfelse>
                <cfoutput>
                <p>Employee #ListGetAt(ValueList(SearchEmpLastName.FirstName), temp)#
                #ListGetAt(ValueList(SearchEmpLastName.LName), temp)#, of the
                #ListGetAt(ValueList(SearchEmpLastName.Department), temp)#
                Department, can be reached at
                #ListGetAt(ValueList(SearchEmpLastName.Phone), temp)#.
                <p>This was the first employee found under this case-insensitive last
                name search.
            </cfoutput>
        </cfif>
    </cfif>
</cfif>
```

ListFirst

説明
リストの最初の要素を取得します。

戻り値
リストの最初の要素。リストが空の場合は、空の文字列を返します。

カテゴリ
リスト関数

関数のシンタックス
```
ListFirst(list [, delimiters, includeEmptyValues ])
```

関連項目
ListGetAt、ListLast、ListQualify、『ColdFusion アプリケーションの開発』の Lists

パラメータ

パラメータ	説明
includeEmptyValues	オプション。空の値を含めるには、このパラメータを yes に設定します。
list	リスト、またはリストを含んでいる変数です。
delimiters	文字列、または文字列を含んでいる変数です。リスト要素を区切る文字です。デフォルト値はカンマ (,) です。 このパラメータに複数の文字が含まれている場合、それぞれの文字が出現するすべての箇所が区切り文字として処理されます。

使用方法
ColdFusion では空のリスト要素が無視されるため、"a,b,c,,,d" の場合は 4 つの要素があることになります。

例
```
<h3>ListFirst Example</h3>
<!--- Find a list of users who wrote messages --->
<cfquery name = "GetMessageUser" datasource = "cfdocexamples">
    SELECT Username, Subject, Posted
    FROMMessages
</cfquery>

<cfset temp = ValueList(GetMessageUser.Username)>
<p>Before editing the list, it is: 
<cfoutput>#ValueList(GetMessageUser.Username)#</cfoutput>.
<p>(Users who posted more than once are listed more than once.)
<!--- Show the first user in the list --->
<p>The first user in the list is:
<cfoutput>#ListFirst(temp)#</cfoutput>
<p>The rest of the list is: <cfoutput>#ListRest(temp)#</cfoutput>.
<p>(Users who posted more than once are listed more than once.)
<p>The last user in the list is: <cfoutput>#ListLast(temp)#</cfoutput>
```

ListGetAt

説明
指定された位置にあるリスト要素を取得します。

戻り値
position に位置するリスト要素の値

カテゴリ
リスト関数

関数のシンタックス
```
ListGetAt(list, position [, delimiters, includeEmptyValues ])
```

関連項目
ListFirst、ListLast、ListQualify、ListSetAt、『ColdFusion アプリケーションの開発』の Lists

パラメータ

パラメータ	説明
list	リスト、またはリストを含んでいる変数です。
position	正の整数、または正の整数を含んでいる変数です。要素を取得する位置を指定します。リスト内の先頭を示す位置は 1 です。
delimiters	文字列、または文字列を含んでいる変数です。リスト要素を区切る文字です。デフォルト値はカンマ (,) です。 このパラメータに複数の文字が含まれている場合、それぞれの文字が出現するすべての箇所が区切り文字として処理されます。
includeEmptyValues	オプション。空の値を含めるには、このパラメータを yes に設定します。

使用方法
文字列の中で区切り文字の後にスペースが付加されている場合、リスト関数を使用すると、返されるリスト要素の先頭にスペースが含まれることがあります。このような場合は、trim 関数を使用して、返された要素からスペースを削除します。たとえば、次のようなリストがこれに該当します。

```
<cfset myList = "one hundred, two hundred, three hundred">
```

このリストから値を取得するには、trim 関数を使用して、戻り値の先頭にあるスペースを削除します。

```
<cfset MyValue = #trim(listGetAt(myList, 2))#>
```

これにより、MyValue = " two hundred" ではなく "two hundred" となります。リスト要素の途中にあるスペースは保持されます。

ColdFusion では空のリスト要素が無視されるため、"a,b,c,,,d" の場合は 4 つの要素があることになります。

例

```
<h3>ListGetAt Example</h3>
<!--- Find a list of users who wrote messages --->
<cfquery name = "GetMessageUser" datasource = "cfdocexamples">
    SELECT Username, Subject, Posted
    FROMMessages
</cfquery>
<cfset temp = ValueList(GetMessageUser.Username)>
<!--- loop through the list and show it with ListGetAt --->
<h3>This list of usernames who have posted messages numbers
<cfoutput>#ListLen(temp)#</cfoutput> users.</h3>
<ul>
<cfloop From = "1" To = "#ListLen(temp)#" index = "Counter">
    <cfoutput><li>Username #Counter#: #ListGetAt(temp, Counter)# </cfoutput>
</cfloop>
</ul>
```

ListInsertAt

説明

リストに要素を挿入します。

戻り値

指定された位置に **value** が挿入されたリストのコピー

カテゴリ

リスト関数

関数のシンタックス

```
ListInsertAt(list, position, value [, delimiters, includeEmptyValues ])
```

関連項目

ListDeleteAt、ListAppend、ListPrepend、ListSetAt、『ColdFusion アプリケーションの開発』の Lists

パラメータ

パラメータ	説明
includeEmptyValues	オプション。空の値を含めるには、このパラメータを yes に設定します。
list	リスト、またはリストを含んでいる変数です。
position	正の整数、または正の整数を含んでいる変数です。要素を挿入する位置を指定します。リスト内の先頭を示す位置は 1 です。
value	要素または要素のリストです。
delimiters	文字列、または文字列を含んでいる変数です。リスト要素を区切る文字です。デフォルト値はカンマ (,) です。 このパラメータに複数の文字が含まれている場合、それぞれの文字が出現するすべての箇所が区切り文字として処理されます。

使用方法

要素を挿入すると、区切り文字が付加されます。delimiters に複数の区切り文字が含まれている場合は、最初の文字が区切り文字として使用されます。delimiters を省略した場合はカンマが使用されます。

ColdFusion では空のリスト要素が無視されるため、"a,b,c,,,d" の場合は 4 つの要素があることになります。

例

```
<!--- This example shows ListInsertAt --->
<cfquery name = "GetParkInfo" datasource = "cfdocexamples">
SELECT PARKNAME,CITY,STATE
FROM PARKS
WHERE PARKNAME LIKE 'DE%'
</cfquery>
<cfset temp = ValueList(GetParkInfo.ParkName)>
<cfset insert_at_this_element = ListGetAt(temp, "3", ",")>
<cfoutput>
<p>The original list: #temp#
</cfoutput>
<cfset temp2 = ListInsertAt(Temp, "3", "my Inserted Value")>
```

ListLast

説明

リストの最後の要素を取得します。

戻り値

リストの最後の要素

カテゴリ

リスト関数

関数のシンタックス

```
ListLast(list [, delimiters, includeEmptyValues ])
```

関連項目

ListGetAt、ListFirst、『ColdFusion アプリケーションの開発』の Lists

パラメータ

パラメータ	説明
includeEmptyValues	オプション。空の値を含めるには、このパラメータを yes に設定します。
list	リスト、またはリストを含んでいる変数です。
delimiters	文字列、または文字列を含んでいる変数です。リスト要素を区切る文字です。デフォルト値はカンマ (,) です。 このパラメータに複数の文字が含まれている場合、それぞれの文字が出現するすべての箇所が区切り文字として処理されます。複数の文字のひとまとまりを区切り文字として指定することはできません。

使用方法

文字列の中で区切り文字の後にスペースが付加されている場合、リスト関数を使用すると、返されるリスト要素の先頭にスペースが含まれることがあります。このような場合は、trim 関数を使用して、返された要素の先頭と末尾からスペースを削除します。たとえば、次のようなリストがこれに該当します。

```
<cfset myList = "one hundred, two hundred, three hundred">
```

このリストから値を取得するには、trim 関数を使用して、戻り値の先頭にあるスペースを削除します。

```
<cfset MyValue = #trim(ListLast(myList)#>
```

これにより、MyValue 変数の値は " three hundred" ではなく "three hundred" となります。リスト要素の途中にあるスペースは保持されます。

ColdFusion では空のリスト要素が無視されるため、"a,b,c,,,d" の場合は 4 つの要素があることになります。

例

```
<h3>ListFirst, ListLast, and ListRest Example</h3>
<!--- Find a list of users who wrote messages --->
<cfquery name = "GetMessageUser" datasource = "cfdocexamples">
    SELECT Username, Subject, Posted
    FROMMessages
</cfquery>
<cfset temp = ValueList(GetMessageUser.Username)>
<p>Before editing the list, it is: 
<cfoutput>#ValueList(GetMessageUser.Username)#</cfoutput>.
<p>(Users who posted more than once are listed more than once.)
<!--- Show the first user in the list --->
<p>The first user in the list is: <cfoutput>#ListFirst(temp)#</cfoutput>
<p>The rest of the list is: <cfoutput>#ListRest(temp)#</cfoutput>.
<p>(Users who posted more than once are listed more than once.)
<p>The last user in the list is: <cfoutput>#ListLast(temp)#</cfoutput>
```

ListLen

説明

リスト内の要素数を調べます。

リスト内の要素数を表す整数を返します。

カテゴリ

リスト関数

関数のシンタックス

```
ListLen(list [, delimiters ])
ListLen(list [, delimiters, includeEmptyValues ])
```

関連項目

ListAppend、ListDeleteAt、ListInsertAt、ListPrepend、『ColdFusion アプリケーションの開発』の Lists

パラメータ

パラメータ	説明
list	リスト、またはリストを含んでいる変数です。
delimiters	文字列、または文字列を含んでいる変数です。リスト要素を区切る文字です。デフォルト値はカンマ (,) です。 このパラメータに複数の文字が含まれている場合、それぞれの文字が出現するすべての箇所が区切り文字として処理されます。
includeEmptyValues	includeEmptyValues を true に設定した場合、長さの計算時に、リスト内のすべての空の値が考慮されます。false に設定した場合、空のリスト要素は無視されます。

使用方法

ColdFusion では空のリスト要素が無視されるため、"a,b,c,,,d" の場合は 4 つの要素があることになります。

次に、ListLen の処理の例を示します。

ステートメント	出力	コメント
ListLen('a,b, c,,,d')	4	3 番目の要素は " c" です。
ListLen('a,b, c,,,d',',')	4	4 番目の要素は "d'" です。
ListLen('elem_1___elem_2___elem_3')	1	
ListLen('elem*1***elem*2***elem*3')	1	
ListLen('elem_1___elem_2___elem_3','_')	6	

例

```
<h3>ListLen Example</h3>
<!--- Find a list of users who wrote messages --->
<cfquery name = "GetMessageUser" datasource = "cfdocexamples">
    SELECT Username, Subject, Posted
    FROMMessages
</cfquery>
<cfset temp = ValueList(GetMessageUser.Username)>
<!--- loop through the list and show it with ListGetAt --->
<h3>This is a list of usernames who have posted messages
<cfoutput>#ListLen(temp)#</cfoutput> users.</h3>
<ul>
<cfloop From = "1" TO = "#ListLen(temp)#" INDEX = "Counter">
    <cfoutput><li>Username #Counter#:
     #ListGetAt(temp, Counter)#</cfoutput>
</cfloop>
</ul>
```

ListPrepend

説明

リストの先頭に要素を挿入します。

戻り値

先頭位置に **value** が挿入されたリストのコピー

カテゴリ

リスト関数

関数のシンタックス

```
ListPrepend(list, value [, delimiters ])
```

関連項目

ListAppend、ListInsertAt、ListSetAt、『ColdFusion アプリケーションの開発』の Lists

パラメータ

パラメータ	説明
list	リスト、またはリストを含んでいる変数です。
value	要素または要素のリストです。
delimiters	文字列、または文字列を含んでいる変数です。リスト要素を区切る文字です。デフォルト値はカンマ (,) です。 このパラメータに複数の文字が含まれている場合は、最初の文字だけが使用され、残りは無視されます。

使用方法
リストの先頭に要素を挿入すると、区切り文字が付加されます。delimiters に複数の区切り文字が含まれている場合は、最初の文字が区切り文字として使用されます。delimiters を省略した場合はカンマが使用されます。

ColdFusion では空のリスト要素が無視されるため、"a,b,c,,,d" の場合は 4 つの要素があることになります。

delimiters パラメータが空の文字列 ("") の場合、value パラメータの内容が返されます。

例
```
<!--- This example shows ListPrepend --->
<cfquery name = "GetParkInfo" datasource = "cfdocexamples">
    SELECT PARKNAME,CITY,STATE
    FROM PARKS
    WHERE PARKNAME LIKE 'DE%'
</cfquery>
<cfset temp = ValueList(GetParkInfo.ParkName)>
<cfset first_element = ListFirst(temp)>
<cfoutput><p>The original list: #temp#</cfoutput>
<!--- now, insert an element at position 1--->
<cfset temp2 = ListPrepend(Temp, "my Inserted Value")>
```

ListQualify

説明
リスト要素の前後に文字列を挿入します。

戻り値
指定された要素の前後に **qualifier** が挿入されたリストのコピー

カテゴリ
リスト関数

関数のシンタックス
```
ListQualify(list, qualifier [, delimiters, elements, includeEmptyValues ])
```

関連項目
『ColdFusion アプリケーションの開発』の Using ColdFusion Variables の Lists

履歴
ColdFusion MX: 動作が変更されました。elements パラメータ値には、"all" または "char" を指定する必要があります。その他の値を指定すると例外が発生します (以前のリリースでは、無効な値は無視され "all" と見なされていましたが、この仕様は他の関数と整合性がありませんでした)。

パラメータ

パラメータ	説明
includeEmptyValues	オプション。空の値を含めるには、このパラメータを yes に設定します。
list	リスト、またはリストを含んでいる変数です。
qualifier	文字列、または文字列を含んでいる変数です。elements パラメータで指定したリスト要素の前後に挿入する文字または文字列を指定します。
delimiters	文字列、または文字列を含んでいる変数です。リスト要素を区切る文字です。デフォルト値はカンマ (,) です。 このパラメータに複数の文字が含まれている場合は、最初の文字だけが区切り文字として使用され、残りは無視されます。
elements	• all: すべての要素 • char: アルファベット文字で構成される要素

使用方法

返されるリストの中には、元のリスト内に含まれていた区切り文字がすべて保持されているとは限りません。

ColdFusion では空のリスト要素が無視されるため、"a,b,c,,,d" の場合は 4 つの要素があることになります。

例

```
<cfquery name = "GetEmployeeNames" datasource = "cfdocexamples">
SELECT FirstName, LastName
FROM Employees
</cfquery>

<h3>ListQualify Example</h3>
<p>This example uses ListQualify to put the full names of the
    employees in the query within quotation marks.</p>
<cfset myArray = ArrayNew(1)>

<!--- loop through query; append these names successively
    to the last element --->
<cfloop query = "GetEmployeeNames">
    <cfset temp = ArrayAppend(myArray, "#FirstName# #LastName#")>
</cfloop>

<!--- sort that array descending alphabetically --->
<cfset myAlphaArray = ArraySort(myArray, "textnocase")>

<!--- show the resulting array as a list --->
<cfset myList = ArrayToList(myArray, ",")>

<cfoutput>
    <p>The contents of the unqualified list are as follows:</p>
    #myList#
</cfoutput>
```

```
<!--- show the resulting alphabetized array as a qualified list with
    single quotation marks around each full name.--->
<cfset qualifiedList1 = ListQualify(myList,"'",",","CHAR")>

<!--- output the array as a list --->
<cfoutput>
    <p>The contents of the qualified list are as follows:</p>
    <p>#qualifiedList1#</p>
</cfoutput>

<!--- show the resulting alphabetized array as a qualified list with quotation
    marks around each full name. We use &quot; to denote quotation marks
    because the quotation mark character is a control character. --->
<cfset qualifiedList2 = ListQualify(myList,"&quot;",",","CHAR")>

<!--- output the array as a list --->
<cfoutput>
    <p>The contents of the second qualified list are:</p>
    <p>#qualifiedList2#</p>
</cfoutput>
```

ListRemoveDuplicates

説明
リスト内の重複する値を削除します（存在する場合）。

戻り値
重複する値が含まれないリスト

シンタックス
ListRemoveDuplicates(list[, delimiter] [, ignoreCase])

プロパティ

パラメータ	説明
list	必須です。オブジェクトのリストです。
delimiter	オプション。リスト要素を区切る文字です。デフォルト値はカンマ（,）です。
ignoreCase	オプション。true の場合、リスト内の文字列の大文字と小文字を無視します。デフォルトでは、値は false に設定されます。

例
```
<cfscript>
                myList = "one,two,three,four,five,one,five,three"
                newList = listremoveduplicates(myList);
                //default delimeter is ","
                //newList contains "one,two,three,four,five"
</cfscript>

<cfscript>
                myList = "one,two,three,four,five,ONE,TWO,THREE"
                newList = listremoveduplicates(myList, ",", true);
    //newList contains "one,two,three,four,five"
</cfscript>
```

ListRest

説明
最初の要素を削除した残りのリストを取得します。

戻り値
最初の要素を削除した **list** のコピー。**list** に要素が 1 つしかない場合は、空のリストが返されます。

カテゴリ
リスト関数

関数のシンタックス
```
ListRest(list [, delimiters, includeEmptyValues ])
```

関連項目
ListFirst、ListGetAt、ListLast、『ColdFusion アプリケーションの開発』の Lists

パラメータ

パラメータ	説明
includeEmptyValues	オプション。空の値を含めるには、このパラメータを yes に設定します。
list	リスト、またはリストを含んでいる変数です。
delimiters	文字列、または文字列を含んでいる変数です。リスト要素を区切る文字です。デフォルト値はカンマ (,) です。 このパラメータに複数の文字が含まれている場合、それぞれの文字が出現するすべての箇所が区切り文字として処理されます。

使用方法
リストの先頭に空のエントリが 1 つまたは複数ある場合は、それらも最初の要素とともに削除されます。

ColdFusion では空のリスト要素が無視されるため、"a,b,c,,,d" の場合は 4 つの要素があることになります。

例
```
<h3>ListFirst, ListLast, and ListRest Example</h3>
<!--- Find a list of users who wrote messages --->
<cfquery name = "GetMessageUser" datasource = "cfdocexamples">
    SELECT Username, Subject, Posted
    FROMMessages
</cfquery>

<cfset temp = ValueList(GetMessageUser.Username)>
<p>Before editing the list, it is: 
<cfoutput>#ValueList(GetMessageUser.Username)#</cfoutput>.
<p>(Users who posted more than once are listed more than once.)
<p>The first user in the list is:
<cfoutput>#ListFirst(temp)# </cfoutput>
<p>The rest of the list is: <cfoutput>#ListRest(temp)#</cfoutput>.
<p>(Users who posted more than once are listed more than once.)
<p>The last user in the list is: <cfoutput>#ListLast(temp)#</cfoutput>
```

ListSetAt

説明
リスト要素の内容を置換します。

戻り値
指定された位置の要素に新しい値を代入したリストのコピー

カテゴリ
リスト関数

関数のシンタックス
```
ListSetAt(list, position, value [, delimiters, includeEmptyValues ])
```

関連項目
ListDeleteAt、ListGetAt、ListInsertAt、『ColdFusion アプリケーションの開発』の Lists

履歴
ColdFusion MX: 区切り文字の修正動作が変更されました。ColdFusion MX では、リスト内の区切り文字は修正されません (以前のリリースでは、区切り文字が delimiters パラメータの最初の文字で置き換えられる場合がありました)。

パラメータ

パラメータ	説明
includeEmptyValues	オプション。空の値を含めるには、このパラメータを yes に設定します。
list	リスト、またはリストを含んでいる変数です。
position	正の整数、または正の整数を含んでいる変数です。値を設定する位置を指定します。リスト内の先頭を示す位置は 1 です。
value	要素または要素のリストです。
delimiters	文字列、または文字列を含んでいる変数です。リスト要素を区切る文字です。デフォルト値はカンマ (,) です。 このパラメータに複数の文字が含まれている場合、それぞれの文字が出現するすべての箇所が区切り文字として処理されます。

使用方法
リスト要素に値を設定すると、区切り文字が付加されます。delimiters に複数の区切り文字が含まれている場合は、最初の文字が区切り文字として使用されます。delimiters を省略した場合はカンマが使用されます。

ColdFusion では空のリスト要素が無視されるため、"a,b,c,,,d" の場合は 4 つの要素があることになります。

例

```
<h3>ListSetAt Example</h3>
<!--- Find a list of users who wrote messages --->
<cfquery name = "GetMessageUser" datasource = "cfdocexamples">
SELECT Username, Subject, Posted
FROMMessages
</cfquery>

<cfset temp = ValueList(GetMessageUser.Subject)>

<!--- loop through the list and show it with ListGetAt --->
<h3>This is a list of <cfoutput>#ListLen(temp)#</cfoutput>
subjects posted in messages.</h3>

<cfset ChangedElement = ListGetAt(temp, 2)>
<cfset TempToo = ListSetAt(temp, 2, "I changed this subject", ",")>
<ul>
<cfloop From = "1" To = "#ListLen(temptoo)#" INDEX = "Counter">
    <cfoutput><li>(#Counter#) SUBJECT: #ListGetAt(temptoo, Counter)#
    </cfoutput>
</cfloop>
</ul>
<p>Note that element 2, "<cfoutput>#changedElement#</cfoutput>",
    has been altered to "I changed this subject" using ListSetAt.
```

ListSort

説明

ソートのタイプとソート順に従って、リスト要素をソートします。

戻り値

ソートしたリストのコピー

カテゴリ

リスト関数

関数のシンタックス

```
ListSort(list, sort_type [, sort_order, delimiters, includeEmptyValues ])
```

関連項目

『ColdFusion アプリケーションの開発』の Using ColdFusion Variables の Lists

履歴

ColdFusion 10：すべての Java でサポートされているロケール固有の文字のサポート（ウムラウト文字のサポートなど）が追加されました。このサポートのフラグは、sorttype = "text" または sorttype = "textnocase" に対して追加されました。

ColdFusion MX: ソートした要素を返す順序が変更されました。textnocase の降順でソートした場合、以前のリリースと異なるソート順で要素を返すことがあります。sort_type="textnocase"、および sort_order="desc" である場合、大文字と小文字の違いだけがある要素については、ColdFusion MX と以前のリリースでは次のように処理が異なります。ColdFusion MX では、昇順を指定した場合の逆の結果になります。以前のリリースでは、大文字と小文字の区別だけが異なる要素の順序は変更されませんでした。これらはいずれも正しい処理ですが、新しい処理では、昇順と降順とで要素が逆の順序で出力されることが保証されます。

たとえば、d,a,a,b,A を textnocase および desc の指定でソートすると、次のようになります。

- ColdFusion MX では、d,b,A,a,a が返されます。
- 以前の ColdFusion のリリースでは、d,b,a,a,A が返されます。

(textnocase および **asc** の指定でソートすると、ColdFusion のすべてのリリースにおいて a,a,A,b,d が返されます。)

パラメータ

パラメータ	説明
includeEmptyValues	オプション。空の値を含めるには、このパラメータを yes に設定します。
list	リスト、またはリストを含んでいる変数です。
localeSensitive	ロケールを考慮したソートを実行するかどうかを指定します。デフォルト値は false です。
sort_type	• numeric: 数値をソートします。 • text: テキストを、大文字と小文字を区別してアルファベット順にソートします。大文字と小文字は分けられ、次のようにソートされます。 - sort_order = "asc" (昇順) の場合は、aabzABZ となります。 - sort_order = "desc" (降順) の場合は、ZBAzbaa となります。 • textnocase: テキストを、大文字と小文字を区別せずにアルファベット順にソートします。この場合、大文字と小文字に関係なく、次のようにアルファベット順にソートされます。 - 昇順では、aAaBbBzzZ のように、同じアルファベットが複数ある場合はそれらの元の順序が保持されます。 - 降順では、ZzzBbBaAa のように、同じアルファベットでの元の順番は逆になります。
sort_order	• asc - 昇順のソートです (デフォルト)。 - 文字の場合、sort_type の値に応じて aabzABZ または aAaBbBzzZ のようにソートされます。 - 数値の場合、小さい数から大きい数の順にソートされます。 • desc - 降順のソートです。 - 文字の場合、sort_type の値に応じて ZBAzbaa または ZzzBbBaAa のようにソートされます。 - 数値の場合、大きい数から小さい数の順にソートされます。
delimiters	文字列、または文字列を含んでいる変数です。リスト要素を区切る文字です。デフォルト値はカンマ (,) です。 このパラメータに複数の文字が含まれている場合は、最初の文字だけが区切り文字として使用され、残りは無視されます。

使用方法

ColdFusion では空のリスト要素が無視されるため、"a,b,c,,,d" の場合は 4 つの要素があることになります。

例

```
<h3>ListSort Example</h3>

<!--- Find a list of users who wrote messages --->
<cfquery name = "GetMessageUser" datasource = "cfdocexamples">
SELECT   Username, Subject, Posted
FROM     Messages
</cfquery>

<cfset myList = ValueList(GetMessageUser.UserName)>
<p>Here is the unsorted list. </p>
<cfoutput>#myList#
    </cfoutput>
<p>Here is the list sorted alphabetically:</p>
<cfset sortedList = ListSort(myList, "Text")>
<cfoutput>#sortedList#
    </cfoutput>

<p>Here is a numeric list that is to be sorted in descending order.</p>
<cfset sortedNums = ListSort("12,23,107,19,1,65","Numeric", "Desc")>
<cfoutput>#sortedNums# </cfoutput>

<p>Here is a list that must be sorted numerically, since it contains
    negative and positive numbers, and decimal numbers. </p>
<cfset sortedNums2 = ListSort("23.75;-34,471:100,-9745","Numeric", "ASC", ";,:")>
<cfoutput>#sortedNums2# </cfoutput>

<p>Here is a list to be sorted alphabetically without consideration of case.</p>
<cfset sortedMix =
    ListSort("hello;123,HELLO:jeans,-345,887;ColdFusion:coldfusion",
        "TextNoCase", "ASC", ";,:")>
<cfoutput>#sortedMix# </cfoutput>
```

ListToArray

説明

リストの要素を配列にコピーします。

戻り値

配列

カテゴリ

配列関数、変換関数、リスト関数

関数のシンタックス

```
ListToArray(list [, delimiters[, includeEmptyFields[, multiCharacterDelimiter]]])
```

関連項目

ArrayToList、『ColdFusion アプリケーションの開発』の Using Arrays and Structures

履歴

ColdFusion 9: multiCharacterDelimiter パラメータが追加されました。

パラメータ

パラメータ	説明
list	リスト、またはリストを含んでいる変数です。 リスト変数は、cfset ステートメントで定義します。
delimiters	文字列、または文字列を含んでいる変数です。この文字列に含まれるそれぞれの文字が区切り文字として扱われます。デフォルト値はカンマ (,) です。
includeEmptyFields	行内に区切り文字が 2 個ある場合に、空の配列エントリを作成するかどうかを指定するブール値です。 • false (デフォルト) リスト内の空の要素を無視します。たとえば、a,,c は、2 つの要素のみが含まれる配列に変換されます。 • true リスト内の空の要素を空の配列エントリに変換します。たとえば、a,,c は、3 つの要素が含まれる配列に変換され、2 番目の要素は空になります。
multiCharacterDelimiter	delimiters パラメータに複数文字で構成される区切り文字が指定されるかどうかを示すブール値です。デフォルトは false です。このパラメータが true の場合、delimiters パラメータには複数の文字で構成される単一の区切り文字を指定する必要があります。このパラメータを使用すると、ListToArray 関数で red:|orange:|yellow:|green:|blue:|indigo:|violet のようなリストを色の名前の配列に変換できます。

使用方法

デフォルトでは空のリスト要素が無視されるため、"a,b,c,,,d" の場合は 4 つの要素があることになります。

delimiters パラメータに含まれるそれぞれの文字が別個の区切り文字として扱われます。したがって、このパラメータがたとえば ",+" の場合、リストはカンマまたはプラス記号のいずれでも区切られます。

multiCharacterDelimiter パラメータを指定した場合、すべてのリスト要素が、指定したそのとおりの文字列で区切られている必要があります。たとえば次のコードでは、"red, orange"、"yellow, green"、および "blue, violet" という 3 つのエントリを含む配列が作成されます。

```
<cfset list = "red,orange,&yellow,green,&blue,violet">
<cfset arr = listToArray (list, ",&",false,true)>
```

例

```
<h3>ListToArray Example</h3>
<!--- Find a list of users who wrote messages --->
<cfquery name = "GetMessageUser" datasource = "cfdocexamples">
SELECT Username, Subject, Posted
FROMMessages
</cfquery>
<cfset myList = ValueList(GetMessageUser.UserName)>
<p>My list is a list with <cfoutput>#ListLen(myList)#</cfoutput>
    elements.
<cfset myArrayList = ListToArray(myList)>
<p>My array list is an array with <cfoutput>#ArrayLen(myArrayList)#
    </cfoutput> elements.
```

ListValueCount

説明

リスト内にある指定した値の個数を数えます。この検索では大文字と小文字が区別されます。

戻り値

リスト内の **value** のインスタンスの数

カテゴリ

リスト関数、文字列関数

関数のシンタックス

```
ListValueCount(list, value [, delimiters ])
```

関連項目

ListValueCountNoCase、『ColdFusion アプリケーションの開発』の Lists

パラメータ

パラメータ	説明
list	リスト、またはリストを含んでいる変数です。
value	文字列、数字、または文字列か数字を含む変数です。検索する項目を指定します。この検索では大文字と小文字が区別されます。
delimiters	文字列、または文字列を含んでいる変数です。リスト要素を区切る文字です。デフォルト値はカンマ (,) です。 このパラメータに複数の文字が含まれている場合、それぞれの文字が出現するすべての箇所が区切り文字として処理されます。

例

```
<cfquery name = "SearchByDepartment" datasource = "cfdocexamples">
SELECT Department
FROM Employees
</cfquery>
<h3>ListValueCount Example</h3>
<p>This example uses ListValueCount to count employees in a department.

<form action = "listvaluecount.cfm">
<p>Select a department:</p>
    <select name = "departmentName">
            <option value = "Accounting">
                Accounting
            </OPTION>
            <option value = "Administration">
                Administration
            </OPTION>
            <option value = "Engineering">
                Engineering
            </OPTION>
            <option value = "Sales">
                Sales
```

```
            </OPTION>
    </select>
<input type = "Submit" name = "Submit" value = "Search Employee List">
</form>

<!--- wait to have a string for searching defined --->
<cfif IsDefined("FORM.Submit") and IsDefined("FORM.departmentName")>
    <cfset myList = ValueList(SearchByDepartment.Department)>
    <cfset numberInDepartment = ListValueCount(myList, FORM.departmentName)>

    <cfif numberInDepartment is 0>
        <h3>There are no employees in <cfoutput>#FORM.departmentName#</cfoutput></h3>
    <cfelseIf numberInDepartment is 1>
        <cfoutput><p>There is only one person in #FORM.departmentName#.
        </cfoutput>
    <cfelse>
        <cfoutput><p>There are #numberInDepartment# people in #FORM.departmentName#.
        </cfoutput>
    </cfif>
</cfif>
```

ListValueCountNoCase

説明

リスト内にある指定した値の個数を数えます。この検索では大文字と小文字は区別されません。

戻り値

リスト内の **value** のインスタンスの数

カテゴリ

リスト関数

関数のシンタックス

```
ListValueCountNoCase(list, value [, delimiters ])
```

関連項目

ListValueCount、『ColdFusion アプリケーションの開発』の Lists

パラメータ

パラメータ	説明
list	リスト、またはリストを含んでいる変数です。
value	文字列、数字、または文字列か数字を含む変数です。検索する項目を指定します。この検索では大文字と小文字は区別されません。
delimiters	文字列、または文字列を含んでいる変数です。リスト要素を区切る文字です。デフォルト値はカンマ (,) です。 このパラメータに複数の文字が含まれている場合、それぞれの文字が出現するすべての箇所が区切り文字として処理されます。

例

```
<cfquery name = "SearchByDepartment" datasource = "cfdocexamples">
SELECT Department
FROM Employees
</cfquery>

<h3>ListValueCountNoCase Example</h3>
<p>This example uses ListValueCountNoCase to count employees in a department.

<form action = "listvaluecountnocase.cfm">
<p>Select a department:</p>
    <select name = "departmentName">
            <option value = "Accounting">
                Accounting
            </OPTION>
            <option value = "Administration">
                Administration
            </OPTION>
            <option value = "Engineering">
                Engineering
            </OPTION>
            <option value = "Sales">
                Sales
            </OPTION>
    </select>
    </select>
<input type = "Submit" name = "Submit" value = "Search Employee List">
</form>
<!--- wait to have a string for searching defined --->
<cfif IsDefined("FORM.Submit") and IsDefined("FORM.departmentName")>
    <cfset myList = ValueList(SearchByDepartment.Department)>
    <cfset numberInDepartment = ListValueCountNoCase(myList,
        FORM.departmentName)>

    <cfif numberInDepartment is 0>
        <h3>There are no employees in <cfoutput>#FORM.departmentName#</cfoutput></h3>
    <cfelseIf numberInDepartment is 1>
        <cfoutput><p>There is only one person in #FORM.departmentName#.
        </cfoutput>
    <cfelse>
        <cfoutput><p>There are #numberInDepartment# people in #FORM.departmentName#.
        </cfoutput>
    </cfif>
</cfif>
```

LJustify

説明

指定した長さの文字列内で、文字を左揃えにします。

戻り値

左揃えにした文字列のコピー

カテゴリ

表示および書式制御関数、文字列関数

関数のシンタックス

```
LJustify(string, length)
```

関連項目

CJustify、RJustify

パラメータ

パラメータ	説明
string	文字列、または文字列を含んでいる変数です。
length	文字列を左揃えにするためのフィールドの長さです。

例

```
<!--- This example shows how to use LJustify --->
<cfparam name = "jstring" default = "">

<cfif IsDefined("FORM.justifyString")>
    <cfset jstring = LJustify(FORM.justifyString, 35)>
</cfif>
<html>
<head>
    <title>LJustify Example</title>
</head>
<body>

<h3>LJustify Function</h3>
<p>Enter a string, and it will be left justified within the sample field

<form action = "ljustify.cfm">
<p><input type = "Text" value = "<cfoutput>#jString#</cfoutput>"
    size = 35 name = "justifyString">

<p><input type = "Submit" name = ""> <input type = "RESET">
</form>
```

Location

説明

cflocation タグと同様の関数で、<cfscript> モードで使用されます。

パラメータ

<cflocation> タグと同じです。

カテゴリ

データ出力関数

関数のシンタックス

```
location("url", addtoken, statusCode);
```

定位置表記の場合は、シンタックスで示した順序どおりに値を指定する必要があります。指定しないパラメータがある場合は、代わりに空の文字列を使用します。これはブール値には適用されません。ブール値の場合は、省略する必要がある場合でも適切な値を指定する必要があります。

関連項目
cfscript、cflocation

使用方法
この関数は、name=value のペアまたは定位置引数として呼び出すことができます。

例
```
<cfscript
    location(url="http://localhost:8500/administrator")
</cfscript>
```

Log

説明
数値の自然対数を計算します。自然対数は、定数 e (2.71828182845904) を底とします。

戻り値
数値の自然対数

カテゴリ
算術関数

関数のシンタックス
```
Log(number)
```

関連項目
Exp、Log10

パラメータ

パラメータ	説明
number	自然対数を計算する正の実数です。

例

```
<h3>Log Example</h3>

<cfif IsDefined("FORM.number")>
    <cfoutput>
        <p>Your number, #FORM.number#
        <br>#FORM.number# raised to the E power: #exp(FORM.number)#
        <cfif FORM.number LTE 0><br>Enter a positive real number to get its
    natural logarithm
            <cfelse><br>The natural logarithm of #FORM.number#: #log(FORM.number)#
        </cfif>
        <cfif FORM.number LTE 0><br>Enter a positive real number to get its
    logarithm to base 10
            <cfelse><br>The logarithm of #FORM.number# to base 10: #log10(FORM.number)#
        </cfif>
    </cfoutput>
</cfif>
<cfform action = "log.cfm">
Enter a number to see its value raised to the E power, its natural logarithm,
    and the logarithm of number to base 10.
<cfinput type = "Text" name = "number" message = "You must enter a number"
    validate = "float" required = "No">
<input type = "Submit" name = "">
</cfform>
```

Log10

説明

10 を底とする **number** の対数を計算します。

戻り値

10 を底とする **number** の対数を表す数値

カテゴリ

算術関数

関数のシンタックス

```
Log10(number)
```

関連項目

Exp、Log

パラメータ

パラメータ	説明
number	対数を計算する正の実数です。

例

```
<h3>Log10 Example</h3>
<cfif IsDefined("FORM.number")>
    <cfoutput>
        <p>Your number, #FORM.number#
        <br>#FORM.number# raised to the E power: #exp(FORM.number)#
        <cfif FORM.number LTE 0><br>You must enter a positive real number to
    see the natural logarithm of that number
        <cfelse><br>The natural logarithm of #FORM.number#: #log(FORM.number)#
        </cfif>
        <cfif #FORM.number# LTE 0><br>You must enter a positive real number to
    see the logarithm of that number to base 10
        <cfelse><br>The logarithm of #FORM.number# to base 10: #log10(FORM.number)#
        </cfif>
    </cfoutput>
</cfif>
<cfform action = "log10.cfm">
Enter a number to find its value raised to the E power, its natural
    logarithm, and the logarithm of number to base 10.
<cfinput type = "Text" name = "number" message = "You must enter a number"
    validate = "float" required = "No">
<input type = "Submit" name = "">
</cfform>
```

LSCurrencyFormat

説明

ロケール固有の通貨形式を使用して、数値を形式設定します。ユーロを使用する国の場合、結果は JVM のバージョンによって異なります。

戻り値

形式設定した通貨の値。

カテゴリ

表示および書式制御関数、各国語対応関数

関数のシンタックス

```
LSCurrencyFormat(number [, type, locale])
```

関連項目

LSEuroCurrencyFormat、LSIsCurrency、LSParseCurrency、LSParseEuroCurrency、SetLocale、『ColdFusion アプリケーションの開発』の Handling data in ColdFusion

履歴

ColdFusion 8: locale パラメータが追加されました。

ColdFusion MX: 形式設定の動作が変更されました。この関数では、以前のリリースと異なる形式で結果が返される場合があります。負の数を渡すと、負の数が返されます。type = "local" の場合、現在のロケールの標準形式で値が返されます。type = "international" の場合、現在のロケールの国際標準形式で値が返されます。この関数では、すべてのプラットフォームで Java 標準のロケール形式設定ルールが使用されます。

パラメータ

パラメータ	説明
number	通貨値です。
type	• local: 現在のロケールで使用される通貨形式と通貨記号です。 - JDK 1.3 では、ユーロ圏の国におけるデフォルトは、各国の通貨です。 - JDK 1.4 では、ユーロ圏の国におけるデフォルトは、ユーロです。 • international: 現在のロケールの国際標準の通貨形式と通貨記号です。 • none: 現在のロケールで使用される通貨形式です。通貨記号はありません。
locale	関数を処理するときに、ページのロケールの代わりに使用するロケールです。

使用方法

この関数では、すべてのプラットフォームで Java 標準のロケール形式設定ルールが使用されます。

注意：Sun の 1.3.1 規格に準拠した JVM の場合、ユーロ通貨値の形式設定には LSEuroCurrencyFormat 関数を使用します。

通貨の出力

次の表に、通貨出力の例を示します。ユーロを使用するロケールについては、Local 列と International 列に 2 つのエントリを示します。1 つめは Sun の 1.4.1 以降の規格に準拠した JVM の場合の結果、2 つめは 1.3.1 規格に準拠した JVM の場合の結果です。

ロケール	Type = Local	Type = International	Type = None
Chinese (China)	¥100,000.00	CNY100,000.00	100,000.00
Chinese (Hong Kong)	HK$100,000.00	HKD100,000.00	100,000.00
Chinese (Taiwan)	NT$100,000.00	TWD100,000.00	100,000.00
Dutch (Belgian)	100.000,00 € 100.000,00 BF	BEF100.000,00 EUR100.000,00	100.000,00
Dutch (Standard)	€ 100.000,00 fl 100.000,00	NLG100.000,00 EUR100.000,00	100.000,00
English (Australian)	$100,000.00	AUD100,000.00	100,000.00
English (Canadian)	$100,000.00	CAD100,000.00	100,000.00
English (New Zealand)	$100,000.00	NZD100,000.00	100,000.00
English (UK)	£100,000.00	GBP100,000.00	100,000.00
English (US)	$100,000.00	USD100,000.00	100,000.00
French (Belgian)	100.000,00 € 100.000,00 FB	EUR100.000,00 BEF100.000,00	100.000,00
French (Canadian)	100 000,00 $	CAD100 000,00	100 000,00
French (Standard)	100 000,00 € 100 000,00 F	EUR100 000,00 FRF100 000,00	100 000,00
French (Swiss)	SFr.100'000.00	CHF100'000.00	100'000.00

ロケール	Type = Local	Type = International	Type = None
German (Austrian)	€ 100.000,00 öS 100.000,00	EUR100.000,00 ATS100.000,00	100.000,00
German (Standard)	100.000,00 € 100.000,00 DM	EUR100.000,00 DEM100.000,00	100.000,00
German (Swiss)	SFr.100'000.00	CHF100'000.00	100'000.00
Italian (Standard)	€ 100.000,00 L. 10.000.000	EUR10.000.000 ITL10.000.000	10.000.000
Italian (Swiss)	SFr.100'000.00	CHF100'000.00	100'000.00
Japanese	¥100,000	JPY100,000	JPY100,000
Korean	W100,000	KRW100,000	100,000
Norwegian (Bokmal)	kr 100 000,00	NOK100 000,00	100 000,00
Norwegian (Nynorsk)	kr 100 000,00	NOK100 000,00	100 000,00
Portuguese (Brazilian)	R$100.000,00	BRC100.000,00	100.000,00
Portuguese (Standard)	100.000,00 € R$100.000,00	EUR100.000,00 BRC100.000,00	100.000,00
Spanish (Mexican)	$100,000.00	MXN100,000.00	100,000.00
Spanish (Modern)	100.000,00 € 10.000.000 Pts	EUR10.000.000 ESP10.000.000	10.000.000
Spanish (Standard)	100.000,00 € 10.000.000 Pts	ESP10.000.000 EUR10.000.000	10.000.000
Swedish	100.000,00 kr	SEK100.000,00	100.000,00

注意：ColdFusion では、Spanish (Modern) は Spanish (Standard) 形式にマッピングされます。

日付、時刻、数値、および通貨の値のデフォルトの表示形式を設定するには、SetLocale 関数を使用してください。

例

```
<h3>LSCurrencyFormat Example</h3>
<p>LSCurrencyFormat returns a currency value using the locale
    convention. Default value is "local."
<!--- loop through list of locales; show currency values for 100,000 units --->
<cfloop LIST = "#Server.Coldfusion.SupportedLocales#"
index = "locale" delimiters = ",">
    <cfset oldlocale = SetLocale(locale)>
    <cfoutput><p><b><I>#locale#</I></b><br>
        Local: #LSCurrencyFormat(100000, "local")#<br>
        International: #LSCurrencyFormat(100000, "international")#<br>
        None: #LSCurrencyFormat(100000, "none")#<br>
        <hr noshade>
    </cfoutput>
</cfloop>
```

LSDateFormat

説明
ロケール固有の形式を使用して、日付時刻値の日付部分の形式を設定します。

戻り値
形式設定された日付時刻値。マスクが指定されていない場合、この値はクライアントコンピュータのロケールの設定に従って形式が設定されます。

カテゴリ
日付および時刻関数、表示および書式制御関数、各国語対応関数

関数のシンタックス
```
LSDateFormat(date [, mask, locale])
```

関連項目
LSParseDateTime、LSTimeFormat、DateFormat、SetLocale、『ColdFusion アプリケーションの開発』の Handling data in ColdFusion

履歴
ColdFusion 8: locale パラメータが追加されました。

ColdFusion MX:

- 形式設定の動作が変更されました。この関数では、以前のリリースと異なる形式で結果が返される場合があります。この関数では、すべてのプラットフォームで Java 標準のロケール形式設定ルールが使用されます。
- 次の **mask** パラメータオプションのサポートを追加 : short、medium、long、および full

パラメータ

パラメータ	説明
date	日付時刻オブジェクトです。値の範囲は西暦 100 ～ 9999 年です。
mask	日付の表示方法を指定する文字のセットです。次の形式を組み合わせて指定します。 • d: 月の日。数字で表した日です。1 桁の場合は先頭に 0 が付きません。 • dd: 月の日。数字で表した日です。1 桁の場合は先頭に 0 が付きます。 • ddd: 曜日。短縮形で表します。 • dddd: 曜日。完全なつづりで表します。 • m: 月。数字で表した月です。1 桁の場合は先頭に 0 が付きません。 • mm: 月。数字で表した月です。1 桁の場合は先頭に 0 が付きます。 • mmm: 月。短縮形で表します (適切な場合)。 • mmmm: 月。完全なつづりで表します。 • y: 年。下 2 桁の数字で表した年です。1 桁の場合は先頭に 0 が付きません。 • yy: 年。下 2 桁の数字で表した年です。1 桁の場合は先頭に 0 が付きます。 • yyyy: 年。4 桁の数字で表した年です。 • gg: 時代および紀元を表す文字列です。処理されません。この形式は将来の使用のために予約されています。 以下の指定は、Java のロケール固有の時刻エンコード標準に適合した形式です。厳密な形式はロケールによって異なります。 • short: dd、mm、および yy を / 記号で区切ったもの。 • medium: mmm、d、および yyyy を使用したテキスト形式。 • long: mmmm、d、および yyyy を使用したテキスト形式。 • full: dddd、mmmm、d、および yyyy を使用したテキスト形式。 デフォルト値は medium です。 形式の詳細については、LSParseDateTime を参照してください。
locale	関数を処理するときに、ページのロケールの代わりに使用するロケールです。

使用方法

この関数では、すべてのプラットフォームで Java 標準のロケール形式設定ルールが使用されます。

日付時刻値を文字列として渡すときは、その値を引用符で囲む必要があります。引用符で囲まない場合、その値は日付時刻オブジェクトの数値表現として解釈されます。

タイムゾーン間の時差を計算するには、GetTimeZoneInfo 関数を使用します。

例

```
<h3>LSDateFormat Example</h3>
<p>LSDateFormat formats the date part of a date/time value using the
    locale convention.
<!--- loop through a list of locales; show date values for Now()--->
<cfloop list = "#Server.Coldfusion.SupportedLocales#"
index = "locale" delimiters = ",">
    <cfset oldlocale = SetLocale(locale)>

    <cfoutput><p><B><I>#locale#</I></B><br>
        #LSDateFormat(Now(), "mmm-dd-yyyy")#<br>
        #LSDateFormat(Now(), "mmmm d, yyyy")#<br>
        #LSDateFormat(Now(), "mm/dd/yyyy")#<br>
        #LSDateFormat(Now(), "d-mmm-yyyy")#<br>
        #LSDateFormat(Now(), "ddd, mmmm dd, yyyy")#<br>
        #LSDateFormat(Now(), "d/m/yy")#<br>
        #LSDateFormat(Now())#<br>
        <hr noshade>
    </cfoutput>
</cfloop>
```

LSDateTimeFormat

説明

ロケール固有の日付と時刻の形式設定規則を使用して、日付時刻値を形式設定します。

戻り値

形式設定された日付時刻値。

カテゴリ

日付および時刻関数、表示および書式制御関数、各国語対応関数

シンタックス

```
LSDateTimeFormat (date , mask)
LSDateTimeFormat (date [, mask, locale])
LSDateTimeFormat (date [, mask, locale, timeZone])
```

関連項目

LSParseDateTime、LSTimeFormat、DateFormat、SetLocale、『ColdFusion アプリケーションの開発』の Handling data in ColdFusion

履歴

ColdFusion 10: この関数が追加されました。

パラメータ

パラメータ	説明
date	必須です。日付時刻オブジェクトです。値の範囲は西暦 100 ～ 9999 年です。
mask	オプション。形式設定に使用するマスクです。
timeZone	タイムゾーン情報です。次のいずれかの形式で指定できます。 • GMT や PST などの省略形 • Europe/Dublin などの完全な名前 デフォルトでは、これは、システムに準拠したタイムゾーンです。
locale	関数を処理するときに、ページのロケールの代わりに使用するロケールです。

例

```
<cfset todayDateTime = Now()>
<body>
<h3>DateTimeFormat Example</h3>
<p>Today's date and time are <cfoutput>#todayDateTime#</cfoutput>.
<p>Using DateTimeFormat, we can display that date and time in different ways:
<cfoutput>
<ul>
    <li>#DateTimeFormat(todayDateTime, "yyyy.MM.dd G 'at' HH:nn:ss z")#
    <li>#DateTimeFormat(todayDateTime, "EEE, MMM d, ''yy")#
    <li>#DateTimeFormat(todayDateTime, "h:nn a")#
    <li>#DateTimeFormat(todayDateTime, "hh 'o''clock' a, zzzz")#
    <li>#DateTimeFormat(todayDateTime, "K:nn a, z")#
    <li>#DateTimeFormat(todayDateTime, "yyyyy.MMMMM.dd GGG hh:nn aaa")#
    <li>#DateTimeFormat(todayDateTime, "EEE, d MMM yyyy HH:nn:ss Z")#
    <li>#DateTimeFormat(todayDateTime, "yyMMddHHnnssZ", "English (UK)", "GMT")#
</ul>
</cfoutput>
```

LSEuroCurrencyFormat

説明

ロケール固有の通貨形式を使用して、数値を形式設定します。

戻り値

形式設定した通貨の値。ユーロ通貨圏に属する国の場合は、通貨をユーロで形式設定するためのロケールの規則が使われます。

カテゴリ

表示および書式制御関数、各国語対応関数

関数のシンタックス

```
LSEuroCurrencyFormat(currency-number [, type, locale])
```

関連項目

LSParseEuroCurrency、LSCurrencyFormat、SetLocale、『ColdFusion アプリケーションの開発』の Locale-specific content

履歴

ColdFusion 8: locale パラメータが追加されました。

ColdFusion MX: 形式設定の動作が変更されました。この関数では、以前のリリースと異なる形式で結果が返される場合があります。この関数では、すべてのプラットフォームで Java 標準のロケール形式設定ルールが使用されます。ただし、ユーロ通貨圏に属する国の場合は、使用方法のセクションに解説した規則が使用されます。したがって、ユーロ圏に属さないロケールでは、ユーロではなくその国の通貨を用いて形式設定されます。

パラメータ

パラメータ	説明
currency-number	通貨値です。
locale	関数を処理するときに、ページのロケールの代わりに使用するロケールです。
type	• local: 現在のロケールで使用される通貨形式です。これはデフォルトです。 • international: 現在のロケールの国際標準通貨形式です。たとえば、EUR10.00 のような形式です。 • none: 現在のロケールで使用される通貨形式です。通貨記号はありません。

使用方法

この関数では、すべてのバージョンの JVM で、ユーロ通貨の形式設定ルールが次のように使用されます。

- 現在のロケールの国がユーロ圏に属する場合 (ユーロに移行済みの参加国である場合) は、type に local を指定して形式設定した出力にはユーロ通貨記号 (€) が含まれます。また、type に international を指定して形式設定した出力には、ユーロ通貨を表す略号 (EUR) が含まれます。値が負の場合、現在のロケールの形式設定ルールに従って、値の前に負の符号が付くか、値が括弧で囲まれます。
- 現在のロケールの国がユーロ圏ではない場合、現在のロケールの通貨記号または略号が表示されます。値が負の場合、現在のロケールの形式設定ルールに従って、値の前に負の符号が付くか、値が括弧で囲まれます。

  ColdFusion がサポートするロケールオプションの一覧や、日付、時刻、数値、および通貨のデフォルトの表示形式の設定に関する情報の詳細については、1226 ページの「SetLocale」SetLocale を参照してください。

通貨の出力

次の表に、通貨出力の例を示します。

ロケール	Type = Local	Type = International	Type = None
Chinese (China)	¥100,000.00	CNY100,000.00	100,000.00
Chinese (Hong Kong)	HK$100,000.00	HKD100,000.00	100,000.00
Chinese (Taiwan)	NT$100,000.00	TWD100,000.00	100,000.00
Dutch (Belgian)	100.000,00 €	EUR100.000,00	100.000,00
Dutch (Standard)	€ 100.000,00	EUR100.000,00	100.000,00
English (Australian)	$100,000.00	AUD100,000.00	100,000.00
English (Canadian)	$100,000.00	CAD100,000.00	100,000.00
English (New Zealand)	$100,000.00	NZD100,000.00	100,000.00
English (UK)	£100,000.00	GBP100,000.00	100,000.00
English (US)	$100,000.00	USD100,000.00	100,000.00

ロケール	Type = Local	Type = International	Type = None
French (Belgian)	100.000,00 €	EUR100.000,00	100.000,00
French (Canadian)	100 000,00 $	CAD100 000,00	100 000,00
French (Standard)	100 000,00 €	EUR100 000,00	100 000,00
French (Swiss)	SFr.100'000.00	CHF100'000.00	100'000.00
German (Austrian)	€ 100.000,00	EUR100.000,00	100.000,00
German (Standard)	100.000,00 €	EUR100.000,00	100.000,00
German (Swiss)	SFr.100'000.00	CHF100'000.00	100'000.00
Italian (Standard)	€ 100.000,00	EUR10.000.000	10.000.000
Italian (Swiss)	SFr.100'000.00	CHF100'000.00	100'000.00
Japanese	¥100,000	JPY100,000	JPY100,000
Korean	W100,000	KRW100,000	100,000
Norwegian (Bokmal)	kr 100 000,00	NOK100 000,00	100 000,00
Norwegian (Nynorsk)	kr 100 000,00	NOK100 000,00	100 000,00
Portuguese (Brazilian)	R$100.000,00	BRC100.000,00	100.000,00
Portuguese (Standard)	100.000,00 €	EUR100.000,00	100.000,00
Spanish (Mexican)	$100,000.00	MXN100,000.00	100,000.00
Spanish (Modern)	100.000,00 €	EUR10.000.000	10.000.000
Spanish (Standard)	100.000,00 €	ESP10.000.000	10.000.000
Swedish	100.000,00 kr	SEK100.000,00	100.000,00

注意：ColdFusion では、Spanish (Modern) と Spanish (Standard) に対して Spanish (Standard) 形式が使用されます。

次の例は、この関数で負の数を形式設定した場合の動作を示しています。負の数は、現在のロケールの形式設定ルールに従って、値の前に負の符号が付くか、値が括弧で囲まれる形式となります。

入力値	ロケールが French (Standard) の場合の出力	ロケールが English (US) の場合の出力
-1234.56	-1 234,56 €	($1,234.56)

例

```
 <h3>LSEuroCurrencyFormat Example</h3>
<p>LSEuroCurrencyFormat returns a currency value using the locale
    convention. Default value is "local."
<!--- Loop through list of locales, show currency values for 100,000 units --->
<cfloop list = "#Server.Coldfusion.SupportedLocales#"
index = "locale" delimiters = ",">
    <cfset oldlocale = SetLocale(locale)>
    <cfoutput><p><B><I>#locale#</I></B><br>
        Local: #LSEuroCurrencyFormat(100000, "local")#<br>
        International: #LSEuroCurrencyFormat(100000, "international")#<br>
        None: #LSEuroCurrencyFormat(100000, "none")#<br>
        <Hr noshade>
    </cfoutput>
</cfloop>
```

LSIsCurrency

説明
文字列が現在のロケールで有効な通貨表現かどうかを判別します。

戻り値
パラメータが、適切な通貨記号などを含め、有効な通貨値として形式設定されている場合は true。local、international、または none の通貨形式による値の場合、戻り値は true です。

カテゴリ
表示および書式制御関数、決定関数、各国語対応関数

関数のシンタックス
```
LSIsCurrency(string [, locale])
```

関連項目
GetLocale、SetLocale、LSCurrencyFormat

履歴
ColdFusion 8: locale パラメータが追加されました。

ColdFusion MX: 形式設定の動作が変更されました。この関数では、以前のリリースと異なる結果が返される場合があります。この関数では、すべてのプラットフォームで Java 標準のロケール形式設定ルールが使用されます。結果は JVM によって変わる可能性があります。たとえば、Sun JVM 1.4.1 では、現在のロケールの国がユーロ圏に属する場合は、local の通貨形式はユーロ形式である必要があります。

パラメータ

パラメータ	説明
string	通貨文字列、または通貨文字列を含んでいる変数です。
locale	関数を処理するときに、ページのロケールの代わりに使用するロケールです。

使用方法
以前のリリースと ColdFusion MX の間の入力形式および出力の表示の違いを示す ColdFusion コード例および出力例については、LSCurrencyFormat を参照してください。

注意：ユーロ圏に属する国のロケールで、かつ通貨がそのロケールのユーロ値として正しく形式設定されている場合、この関数は Sun 1.3.1 を含むすべての JVM で true を返します。したがって、1.3.1 に準拠した JVM では、LSIsCurrency 関数が true となる場合でも、LSParseCurrency で値が返されるとは限りません。ユーロ圏のロケールで、通貨値が従来の各国固有形式の場合、LSIsCurrency 関数は、新しい JVM (たとえば Sun 1.4.1 や 1.6) では false、古い JVM (たとえば Sun 1.3.1) では true を返します。

注意：日付、時刻、数値、および通貨の値のデフォルトの表示形式を設定するには、SetLocale 関数を使用してください。

例

```
<h3>LSIsCurrency Example</h3>

<cfif IsDefined("FORM.locale")>
<!--- if locale is defined, set locale to that entry --->
<cfset NewLocale = SetLocale(FORM.locale)>

<p>Is the value "<cfoutput>#FORM.myValue#</cfoutput>"
    a proper currency value for <cfoutput>#GetLocale()#</cfoutput>?
<p>Answer: <cfoutput>#LSIsCurrency(FORM.myValue)#</cfoutput>
</cfif>

<p><form action = "LSIsCurrency.cfm">
<p>Select a locale for which you would like to check a currency value:
<!--- check the current locale for server --->
<cfset serverLocale = GetLocale()>
```

LSIsDate

説明

文字列が現在のロケールで有効な日付時刻値表現かどうかを判別します。

戻り値

現在のロケールで、日付時刻値として文字列を形式設定できる場合は true、設定できない場合は false。

カテゴリ

日付および時刻関数、表示および書式制御関数、各国語対応関数

関数のシンタックス

```
LSIsDate(string [, locale])
```

関連項目

CreateDateTime、GetLocale、IsNumericDate、LSDateFormat、ParseDateTime、SetLocale、『ColdFusion アプリケーションの開発』の Handling data in ColdFusion

履歴

ColdFusion 8: locale パラメータが追加されました。

ColdFusion MX:

- 形式設定の動作が変更されました。この関数では、以前のリリースと異なる結果が返される場合があります。この関数では、すべてのプラットフォームで Java 標準のロケール形式設定ルールが使用されます。
- 動作が変更されました。この関数では、Dutch (Standard) と Portuguese (Standard) のロケールでのみダッシュ文字またはハイフン文字を使用できます。他のロケールでこれらの文字を使用して呼び出した場合 (たとえば、LsIsDate("3-1-2002"))、どのロケールであっても、false が返されます。以前のリリースでは、true が返されました。
- 動作が変更されました。English (UK) ロケールで SUN JRE 1.3.1 を使用すると、この関数では、1 桁の月または日が含まれている日付 (1/1/01 など) に対して false が返されます。この問題を回避するために、1 桁の月日にゼロを挿入してください (例 01/01/01)。

パラメータ

パラメータ	説明
string	文字列、または文字列を含んでいる変数です。
locale	関数を処理するときに、ページのロケールの代わりに使用するロケールです。

使用方法
日付時刻オブジェクトの範囲は、西暦 100 ～ 9999 年です。

日付、時刻、数値、および通貨の値のデフォルトの表示形式を設定するには、SetLocale 関数を使用してください。

例
```
<h3>LSIsDate Example</h3>
<cfif IsDefined("FORM.locale")>
    <!--- if locale is defined, set locale to that entry --->
    <cfset NewLocale = SetLocale(FORM.locale)>
    <p>Is the value "<cfoutput>#FORM.myValue#</cFOUTPUT>" a proper date
    value for <cfoutput>#GetLocale()#</cfoutput>?
    <p>Answer: <cfoutput>#LSIsDate(FORM.myValue)#</cfoutput>
</cfif>
<p><form action = "LSIsDate.cfm">
<p>Select a locale for which you would like to check a date value:
<!--- check the current locale for server --->
<cfset serverLocale = GetLocale()>
```

LSIsNumeric

説明
文字列が現在のロケールで有効な数値表現かどうかを判別します。

戻り値
文字列が現在のロケールで有効な数値表現の場合は true、その他の場合は false

カテゴリ
決定関数、各国語対応関数、文字列関数

関数のシンタックス
```
LSIsNumeric(string [, locale])
```

関連項目
GetLocale、SetLocale、『ColdFusion アプリケーションの開発』の Handling data in ColdFusion

履歴
ColdFusion 8: locale パラメータが追加されました。

ColdFusion MX: 形式設定の動作が変更されました。この関数では、以前のリリースと異なる結果が返される場合があります。この関数では、すべてのプラットフォームで Java 標準のロケール形式設定ルールが使用されます。

パラメータ

パラメータ	説明
string	文字列、または文字列を含んでいる変数です。
locale	関数を処理するときに、ページのロケールの代わりに使用するロケールです。

使用方法
日付、時刻、数値、および通貨の値のデフォルトの表示形式を設定するには、SetLocale 関数を使用してください。

例
```
<h3>LSIsNumeric Example</h3>

<cfif IsDefined("FORM.locale")>
<!--- if locale is defined, set locale to that entry --->
<cfset NewLocale = SetLocale(FORM.locale)>

<p>Is the value "<cfoutput>#FORM.myValue#</cFOUTPUT>"
a proper numeric value for <cfoutput>#GetLocale()#</cfoutput>?

<p>Answer: <cfoutput>#LSIsNumeric(FORM.myValue)#</cfoutput>
</cfif>

<p><form action = "LSIsNumeric.cfm">

<p>Select a locale for which to check a numeric value:
...
```

LSNumberFormat

説明
ロケール固有の形式を使用して、数値を形式設定します。

戻り値
形式設定された数値

- マスクが指定されていない場合は、整数として形式設定された数値が返されます。
- マスクが指定されていない場合は、小数部分が丸められます。たとえば、34.57 は 35 となります。
- 指定されたマスクで数値を正しくマスクできない場合は、そのままの数値が返されます。
- パラメータ値が "" (空の文字列) の場合、0 が返されます。

カテゴリ
表示および書式制御関数、各国語対応関数

関数のシンタックス
```
LSNumberFormat(number [, mask, locale])
```

関連項目
GetLocale、SetLocale、『ColdFusion アプリケーションの開発』の Handling data in ColdFusion

履歴

ColdFusion 8: locale パラメータが追加されました。

ColdFusion MX:

- 動作が変更されました。指定されたマスク形式で数値を正しくマスクできない場合は、そのままの数値が返されます。以前のリリースでは、数値が丸められるか、例外が発生していました。マスクが指定されていない場合、ColdFusion MX では ColdFusion 5 と同様に小数部分が丸められます。たとえば、1234.567 は 1235 となります。
- 形式設定の動作が変更されました。この関数では、以前のリリースと異なる形式で結果が返される場合があります。この関数では、すべてのプラットフォームで Java 標準のロケール形式設定ルールが使用されます。

パラメータ

パラメータ	説明
number	形式設定する数値です。
mask	LSNumberFormat マスク文字が適用されます。ただし、ドル記号 ($)、カンマ (,)、およびドット (.) は、対応するロケール固有の文字にマッピングされます。
locale	関数を処理するときに、ページのロケールの代わりに使用するロケールです。

次の表に、LSNumberFormat マスク文字のリストを示します。

文字	説明
_	(アンダースコア) 桁プレースホルダーです。
9	桁プレースホルダーです。アンダースコア (_) よりも明確に小数点の位置を示します。
	この位置に小数点 (またはロケールに応じた記号) を固定します。
0	固定した小数点の左または右に置いて、0 でパディングすることを示します。
( )	数値が 0 よりも小さい場合は、マスクを丸括弧で囲みます。
+	正の数の前にプラス記号 (+) を付け、負の数の前にマイナス記号 (-) を付けます。
-	正の数の前にスペースを付け、負の数の前にマイナス記号 (-) を付けます。
,	3 桁ごとにカンマ (またはロケールに応じた記号) で区切ります。
L、C	マスク列の中で数値を左揃えまたは中央揃えにします。L または C はマスクの先頭に指定する必要があります。デフォルトは右揃えです。
$	形式設定された数値の先頭にドル記号 (またはロケールに応じた記号) を置きます。ドル記号 ($) はマスクの先頭に指定する必要があります。
^	右側の形式と左側の形式を区切ります。

注意：マスクで符号が指定されていない場合は、正の数値と負の数値の桁位置が揃いません。正の数値の前にプラス記号またはスペースを付け、負の数の前にマイナス記号を付けるには、プラス記号またはマイナス記号のマスク文字をそれぞれ使用します。

使用方法

この関数では、すべてのプラットフォームで Java 標準のロケール形式設定ルールが使用されます。

形式マスク内の記号の位置によって、これらのコードが作用する位置は異なります。たとえば、ドル記号を形式マスクの完全な左端に配置すると、形式設定された数値のフィールド全体の左端にドル記号が表示されます。形式マスクの左端とドル記号の間に 1 つまたは複数のアンダースコアを置くと、形式設定された数値のフィールド内で、数字のすぐ左に隣接してドル記号が表示されます。

次の例は、記号の位置がどのように形式設定に影響するかを示しています。

数値	マスク	結果
4.37	$____.__	"$ 4.37"
4.37	_$___.__	" $4.37"

同様に、負の数値にマイナス記号 (-) を表示する位置も次のように指定できます。

数値	マスク	結果
-4.37	-____.__	"- 4.37"
-4.37	_-___.__	" -4.37"

記号の位置は、左端、左隣、右隣、および右端です。左右の位置は、コード文字が表示される小数点の位置によって決まります。小数点以下の桁数が固定されていない形式の場合は、キャレット (^) を使用して左右のフィールドを区切ることができます。

コードを離れた位置に置くか、隣接した位置に置くかは、アンダースコア (_) によって決まります。ほとんどのコード文字は、いずれのフィールドに置くかによって効果が異なります。次の例は、負の数値を表示するための括弧を置く位置を指定する方法を示しています。

数値	マスク	結果
3.21	C(__^__)	"(3.21)"
3.21	C__(^__)	" (3.21)"
3.21	C(__^)__	"(3.21) "
3.21	C__(^)__	" (3.21) "

日付、時刻、数値、および通貨の値のデフォルトの表示形式を設定するには、SetLocale 関数を使用してください。

文字列を double 型の数値に変換するとき、この関数では、四捨五入による誤差を防ぐため、変換後の数値に四捨五入係数 1.5543122344752E-014 を加算します。たとえば、四捨五入係数を追加しないで、文字列値 1.275 を小数点以下 2 桁の double 値に変換すると、値は 1.27499999999999999 となり、1.27 に四捨五入されます。四捨五入係数を加えることにより、正しく変換が行われ、結果は 1.28 となります。

ただし、1.994999999999999999999999999999 のように 10E-14 の桁まで使用されている double 値の場合、四捨五入係数が加算されることで、誤った四捨五入の結果が生じる可能性があります。

例

```
<h3>LSNumberFormat Example</h3>
<p>LSNumberFormat returns a number value using the locale convention.
<!--- loop through a list of locales and show number values --->
<cfloop LIST = "#Server.Coldfusion.SupportedLocales#"
index = "locale" delimiters = ",">
    <cfset oldlocale = SetLocale(locale)>
    <cfoutput><p><b><i>#locale#</i></b><br>
        #LSNumberFormat(-1234.5678, "__________")#<br>
        #LSNumberFormat(-1234.5678, "__________.___")#<br>
        #LSNumberFormat(1234.5678, "__________")#<br>
        #LSNumberFormat(1234.5678, "__________.___")#<br>
        #LSNumberFormat(1234.5678, "$_(__________.___)")#<br>
        #LSNumberFormat(-1234.5678, "$_(__________.___)")#<br>
        #LSNumberFormat(1234.5678, "+__________.___")#<br>
        #LSNumberFormat(1234.5678, "-__________.___")#<br>
    </cfoutput>
</cfloop>
```

LSParseCurrency

説明

ロケール固有の通貨文字列を、形式設定された数値に変換します。文字列を、サポートされている 3 種類の通貨形式 (none、local、international) のそれぞれに照らし合わせ、最初に合致したものを使って変換します。

戻り値

パラメータの値と一致する、形式設定した数値 (数値の文字列表現)

カテゴリ

各国語対応関数、文字列関数

関数のシンタックス

```
LSParseCurrency(string [, locale])
```

関連項目

LSParseEuroCurrency、LSCurrencyFormat、LSEuroCurrencyFormat、LSIsCurrency、『ColdFusion アプリケーションの開発』の Locale-specific content

履歴

ColdFusion 8: locale パラメータが追加されました。

ColdFusion MX: 形式設定の動作が変更されました。この関数では、以前のリリースと異なる形式で結果が返される場合があります。この関数では、すべてのプラットフォームで Java 標準のロケール形式設定ルールが使用されます。

パラメータ

パラメータ	説明
string	ロケール固有の文字列、またはそれを含んでいる変数です。
locale	関数を処理するときに、ページのロケールの代わりに使用するロケールです。

使用方法

この関数では、すべてのプラットフォームで JVM のロケール形式設定ルール (ColdFusion Administrator の [Java と JVM の設定] ページで設定) が使用されます。このルールは、Sun JVM 1.3.1 と JVM 1.4.1 の間に変更になりました。

- JVM 1.3.1 の場合は、ユーロ圏に属する国の通貨 (local および international 形式のもの) は、従来の各国固有形式で形式設定されている必要があります。たとえば、German (Standard) ロケールなら、100.000,00 DM または DEM100.000,00 となります。LSParseEuroCurrency 関数を使用して、JVM 1.3.1 に基づくこれらのロケールでユーロ通貨を解析します。
- 一方、JVM 1.4.1 の場合は、ユーロ圏の国の通貨はユーロとして形式設定されている必要があります。たとえば、100.000,00 € や EUR100.000,00 です。

  注意 : ロケールがユーロ通貨圏のもので、通貨がユーロで表される場合 (JVM 1.3.1 を使用する場合など)、LSIsCurrency 関数は常に true を返します。結果的に、古い JVM を使用した場合、LSIsCurrency では LSParseCurrency が値を返すとは限りません。

  日付、時刻、数値、および通貨の値のデフォルトの表示形式を設定するには、SetLocale 関数を使用してください。

  通貨の変換に使用するロケール固有の形式のリストについては、LSCurrencyFormat を参照してください。

例

```
<h3>LSParseCurrency Example</h3>
<p>LSParseCurrency coverts a locale-specific currency string to a number.
    Attempts conversion through each of the three default currency formats.
<!--- loop through a list of locales; show currency values for 123,456 units --->
<cfloop LIST = "#Server.Coldfusion.SupportedLocales#"
index = "locale" delimiters = ",">
    <cfset oldlocale = SetLocale(locale)>
    <cfoutput><p><B><I>#locale#</I></B><br>
        Local: #LSCurrencyFormat(123456.78, "local")#<br>
        Parsed local Currency:
    #LSParseCurrency(LSCurrencyFormat(123456,"local"))#<br>
        International: #LSCurrencyFormat(123456.78999, "international")#<br>
        Parsed International Currency:
    #LSParseCurrency(LSCurrencyFormat(123456.78999,"international"))#<br>
        None: #LSCurrencyFormat(123456.78999, "none")#<br>
        Parsed None formatted currency:
    #LSParseCurrency(LSCurrencyFormat(123456.78999,"none"))#<br>
        <hr noshade>
    </cfoutput>
</cfloop>
```

LSParseDateTime

説明

現在のロケールの有効な日付時刻表現である文字列を、日付時刻オブジェクトに変換します。

戻り値

日付時刻オブジェクト

カテゴリ

日付および時刻関数、表示および書式制御関数、各国語対応関数、文字列関数

関数のシンタックス

```
LSParseDateTime(date/time-string [, locale, format])
```

関連項目

LSDateFormat、ParseDateTime、SetLocale、GetLocale、『ColdFusion アプリケーションの開発』の Locales

履歴

ColdFusion 10：format パラメーターが追加されました。

ColdFusion 8: locale パラメータが追加されました。

ColdFusion MX:

- 形式設定の動作が変更されました。この関数では、以前のリリースで使用できた形式の文字列が使用できなくなっている場合があります。この関数では、すべてのプラットフォームで Java 標準のロケール形式設定ルールが使用されます。
- date/time-string パラメータ値の処理方法が変更されました。「使用方法」で説明するように、date/time-string パラメータ値のタイムゾーン情報の処理方法が、以前のリリースとは異なっています。

パラメータ

パラメータ	説明
date/time-string	現在のロケールで読み取り可能な形式で表された文字列、またはそのような文字列を含んでいる変数です。
format	オプション。文字列の形式を指定します。この文字列は、指定した日付文字列を日付時刻オブジェクトに解析するために使用されます。
locale	関数を処理するときに、ページのロケールの代わりに使用するロケールです。

使用方法

この関数は、現在のロケールの Java 標準のロケール形式設定ルールにのっとった日付、時刻、および日付と時刻の組み合わせのいずれも変換できます。

次の表は、English (US) ロケールでこの関数に渡すことのできる日付時刻値の例をいくつか挙げたものです。これらの形式の日付部分のみまたは時刻部分のみを渡すことも可能です。

形式	例
m/dd/yy h:mm:ss	1/30/02 7:02:33
m/dd/yy h:mm tt	1/30/02 7:02 AM
m/dd/yyyy h:mm	1/30/2002 7:02 AM
mmm dd, yyyy h:mm:ss tt	Jan 30, 2002 7:02:12 AM
mmmm dd, yyyy h:mm:ss tt zzz	January 30, 2002 7:02:23 AM PST
ddd, mmm dd, yyyy hh:mm:ss	Wed, Jan 30, 2002 07:02:12
dddd, mmmm dd, yyyy h:mm:ss tt zzz	Wednesday, January 30, 2002 7:02:12 AM PST

有効な日付の範囲は、西暦 100 ～ 9999 年です。2 桁で年を指定する場合、00 ～ 29 は 2000 ～ 2029 年と解釈されます。30 ～ 99 は 1930 ～ 1999 年と解釈されます。

この関数は、現在のタイムゾーンと、入力パラメータで指定されたタイムゾーンの時差を補正します。

- date/time-string パラメータで指定されたタイムゾーンがコンピュータのタイムゾーン設定と異なっている場合、ColdFusion は時刻値をコンピュータのタイムゾーンと同じ値に調整します。
- date/time-string パラメータにタイムゾーンが指定されなかった場合、ColdFusion は時刻値を調整しません。

  注意：この関数では、POP 形式の日付 (タイムゾーンオフセット値を含む日付) は使用できません。

次の例では、指定した日付文字列を解釈する形式を format パラメーターで指定します。

```
<cfoutput>#lsParseDateTime("01/08/2011","en","MM/dd/yyyy")#</cfoutput>
```

例

```
<h3>LSParseDateTime Example - returns a locale-specific date/time object</h3>
<!--- loop through a list of locales and show date values for Now()--->
<cfloop LIST = "#Server.Coldfusion.SupportedLocales#"
index = "locale" delimiters = ",">
    <cfset oldlocale = SetLocale(locale)>
    <cfoutput><p><B><I>#locale#</I></B><br>
        <p>Locale-specific formats:
        <br>#LSDateFormat(Now(), "mmm-dd-yyyy")# #LSTimeFormat(Now())#<br>
        #LSDateFormat(Now(), "mmmm d, yyyy")# #LSTimeFormat(Now())#<br>
        #LSDateFormat(Now(), "mm/dd/yyyy")# #LSTimeFormat(Now())#<br>
        #LSDateFormat(Now(), "d-mmm-yyyy")# #LSTimeFormat(Now())#<br>
        #LSDateFormat(Now(), "ddd, mmmm dd, yyyy")# #LSTimeFormat(Now())#<br>
        #LSDateFormat(Now(), "d/m/yy")# #LSTimeFormat(Now())#<br>
        #LSDateFormat(Now())# #LSTimeFormat(Now())#<br>
    <p>Standard Date/Time:
    #LSParseDateTime("#LSDateFormat(Now())# #LSTimeFormat(Now())#")#<br>
    </cfoutput>
</cfloop>
```

LSParseEuroCurrency

説明

ロケール固有の通貨文字列を、数値として形式設定します。デフォルトの各通貨形式 (none、local、international) で変換を試みます。ユーロ圏の国では、ユーロ通貨を正しく処理します。

戻り値

文字列の値と一致する、形式設定した数値

カテゴリ

各国語対応関数、文字列関数

関数のシンタックス

```
LSParseEuroCurrency(currency-string [, locale])
```

関連項目

LSParseCurrency、LSEuroCurrencyFormat、SetLocale、『ColdFusion アプリケーションの開発』の Locale-specific content

履歴

ColdFusion 8: locale パラメータが追加されました。

ColdFusion MX: 形式設定の動作が変更されました。この関数では、以前のリリースと異なる形式で結果が返される場合があります。この関数では、すべてのプラットフォームで Java 標準のロケール形式設定ルールが使用されます。ただし、ユーロ通貨圏に属する国の場合は、使用方法のセクションに解説した規則が使用されます。

パラメータ

パラメータ	説明
currency-string	ロケール固有の文字列、またはそれを含んでいる変数です。
locale	関数を処理するときに、ページのロケールの代わりに使用するロケールです。

使用方法

この関数は、現在のロケールの国がユーロ圏に属し、すべてのメンバーがユーロに変換されているかどうかを指定します。ユーロ圏に属する場合は、currency-string パラメータが Sun JVM 1.3.1 を含むすべての JVM でユーロに形式設定される必要があります。ユーロ圏に属さない場合、文字列は JVM のロケール設定ルールに準拠する必要があります。サポートされるすべてのロケールでの有効な通貨形式の例については、1133 ページの「LSEuroCurrencyFormat」を参照してください。

ColdFusion がサポートするロケールオプションの一覧や、日付、時刻、数値、および通貨のデフォルトの表示形式の設定に関する情報の詳細については、SetLocale を参照してください。

例

```
<h3>LSParseEuroCurrency Example</h3>
<p>Loop through all available locales. Create string representations of the value
    123,456 in the three supported currency formats,
    and parse the results back to numbers.<p>
<cfloop list="#Server.Coldfusion.SupportedLocales#" index="locale" delimiters=",">
    <cfset oldlocale = SetLocale(locale)>
    <cfoutput><p>Current Locale: <b><i>#locale#</i></b><br>
    <cfset localCurrency = LSEuroCurrencyFormat(123456, "local")>
        Value in local currency: #localCurrency#<br>
        Parsed using LSParseEuroCurrency:
        #LSParseEuroCurrency(localCurrency)#<br>
    <cfset IntlCurrency = LSEuroCurrencyFormat(123456, "international")>
        Value with International currency formatting: #IntlCurrency#<br>
        Parsed using LSParseEuroCurrency:
        #LSParseEuroCurrency(IntlCurrency)#<br>
    <cfset Currency = LSEuroCurrencyFormat(123456, "none")>
        Value with no currency formatting: #currency#<br>
        Parsed using LSParseEuroCurrency:
        #LSParseEuroCurrency(Currency)#<br>
        <hr noshade>
    </cfoutput>
</cfloop>
```

LSParseNumber

説明

現在のロケールの有効な数値表現である文字列を、形式設定された数値に変換します。

戻り値

文字列の値と一致する、形式設定した数値

カテゴリ

各国語対応関数、文字列関数

関数のシンタックス

```
LSParseNumber(string [, locale])
```

関連項目
LSParseDateTime、SetLocale、『ColdFusion アプリケーションの開発』の Locales

履歴
ColdFusion 8: locale パラメータが追加されました。

ColdFusion MX: 形式設定の動作が変更されました。この関数では、以前のリリースと異なる形式で結果が返される場合があります。この関数では、すべてのプラットフォームで Java 標準のロケール形式設定ルールが使用されます。

パラメータ

パラメータ	説明
string	文字列、または文字列を含んでいる変数です。
locale	関数を処理するときに、ページのロケールの代わりに使用するロケールです。

使用方法
この関数では、すべてのプラットフォームで Java 標準のロケール形式設定ルールが使用されます。

日付、時刻、数値、および通貨の値のデフォルトの表示形式を設定するには、SetLocale 関数を使用してください。

例
```
<h3>LSParseNumber Example</h3>
<p>LSParseNumber converts a locale-specific string to a number.
    Returns the number matching the value of string.
<!--- loop through a list of locales and show number values --->
<cfloop LIST = "#Server.Coldfusion.SupportedLocales#"
index = "locale" delimiters = ",">
    <cfset oldlocale = SetLocale(locale)>

    <cfoutput><p><B><I>#locale#</I></B><br>
        #LSNumberFormat(-1234.5678, "_________")#<br>
        #LSNumberFormat(-1234.5678, "_________.___")#<br>
        #LSNumberFormat(1234.5678, "_________")#<br>
        #LSNumberFormat(1234.5678, "_________.___")#<br>
        #LSNumberFormat(1234.5678, "$_(_________.___)")#<br>
        #LSNumberFormat(-1234.5678, "$_(_________.___)")#<br>
        #LSNumberFormat(1234.5678, "+_________.___")#<br>
        #LSNumberFormat(1234.5678, "-_________.___")#<br>
        The actual number:
            #LSParseNumber(LSNumberFormat(1234.5678, "_________"))#<br>
        <hr noshade>
    </cfoutput>
</cfloop>
```

LSTimeFormat

説明
ロケール固有の形式を使用して、日付時刻文字列の時刻部分の形式を設定します。

戻り値
時刻値を表現する文字列

カテゴリ

日付および時刻関数、表示および書式制御関数、各国語対応関数

関数のシンタックス

```
LSTimeFormat(time [, mask ])
```

関連項目

LSParseDateTime、LSDateFormat、TimeFormat、『ColdFusion アプリケーションの開発』の Locales

履歴

ColdFusion MX 6.1: ミリ秒を表すマスク文字 L および l が追加されました。

ColdFusion MX:

- 形式設定の動作が変更されました。この関数では、以前のリリースと異なる形式で結果が返される場合があります。この関数では、すべてのプラットフォームで Java 標準のロケール形式設定ルールが使用されます。
- 次の **mask** パラメータオプションのサポートを追加 : short、medium、long、および full

パラメータ

パラメータ	説明
string	• 日付時刻値です。 • 時刻値に変換できる文字列です。 日付時刻オブジェクトの範囲は、西暦 100 ～ 9999 年です。
mask	形式を指定するマスク文字からなる文字列です。 • h: 時。1 桁の場合は先頭に 0 が付きません (12 時間形式)。 • hh: 時。1 桁の場合は先頭に 0 が付きます (12 時間形式)。 • H: 時。1 桁の場合は先頭に 0 が付きません (24 時間形式)。 • HH: 時。1 桁の場合は先頭に 0 が付きます (24 時間形式)。 • m: 分。1 桁の場合は先頭に 0 が付きません。 • mm: 分。1 桁の場合は先頭に 0 が付きます。 • s: 秒。1 桁の場合は先頭に 0 が付きません。 • ss: 秒。1 桁の場合は先頭に 0 が付きます。 • l: ミリ秒。 • t: 午前および午後を表す 1 文字の時刻マーカー文字列 (A および P など)。 • tt: 午前および午後を表す複数文字の時刻マーカー文字列 (AM および PM など)。 以下の指定は、Java のロケール固有の時刻エンコード標準に適合した形式です。厳密な形式はロケールによって異なります。 • short: 時、分。AM や PM が使われる場合もあります。 • medium: 時、分。AM や PM が使われる場合もあります。 • long: medium の内容に加えてタイムゾーンが含まれます。 • full: ロケールによっては、ある種の時刻を示すシンボルが用いられます。 デフォルト値は short です。

使用方法

この関数では、すべてのプラットフォームで Java 標準のロケール形式設定ルールが使用されます。

日付時刻値を文字列として渡すときは、その値を引用符で囲む必要があります。引用符で囲まない場合、その値は日付時刻オブジェクトの数値表現として解釈されます。

タイムゾーン間の時差を計算するには、GetTimeZoneInfo 関数を使用します。

日付、時刻、数値、および通貨の値のデフォルトの表示形式を設定するには、SetLocale 関数を使用してください。

秒の値がこの関数に渡されず、マスク値が s の場合、デフォルトで出力される秒の形式はゼロが 1 つです。たとえば、lstimeformat(6:39, "h:m:s") は 6:39:0 を返します。マスク値が ss の場合は、6:39:00 を返します。

例

```
<h3>LSTimeFormat Example</h3>

<p>LSTimeFormat returns a time value using the locale convention.

<!--- loop through a list of locales and show time values --->
<cfloop LIST = "#Server.Coldfusion.SupportedLocales#"
index = "locale" delimiters = ",">
    <cfset oldlocale = SetLocale(locale)>

    <cfoutput><p><B><I>#locale#</I></B><br>
    #LSTimeFormat(Now())#<br>
    #LSTimeFormat(Now(), 'hh:mm:ss')#<br>
    #LSTimeFormat(Now(), 'hh:mm:sst')#<br>
    #LSTimeFormat(Now(), 'hh:mm:sstt')#<br>
    #LSTimeFormat(Now(), 'HH:mm:ss')#<br>
        <hr noshade>
    </cfoutput>

</cfloop>
```

LTrim

説明

文字列の先頭のスペースを削除します。

戻り値

先頭のスペースがない文字列のコピー

カテゴリ

表示および書式制御関数、文字列関数

関数のシンタックス

```
LTrim(string)
```

関連項目

RTrim、ToBase64

パラメータ

パラメータ	説明
string	文字列、または文字列を含んでいる変数です。

例

```
<h3>LTrim Example</h3>

<cfif IsDefined("FORM.myText")>
<cfoutput>
<pre>
Your string:"#FORM.myText#"
Your string:"#LTrim(FORM.myText)#"
(left trimmed)
</pre>
</cfoutput>
</cfif>

<form action = "ltrim.cfm">
<p>Type in some text, and it will be modified by LTrim to remove
    leading spaces from the left
<p><input type = "Text" name = "myText" value = " TEST">

<p><input type = "Submit" name = "">
</form>
```

関数 m ～ r

Max

説明

2 つの数値のいずれが大きいかを調べます。

戻り値

2 つの数値のうち大きい方の値

カテゴリ

算術関数

関数のシンタックス

```
Max(number1, number2)
```

関連項目

Min

パラメータ

パラメータ	説明
number1、number2	数値

例

```
<h3>Max Example</h3>
<cfif IsDefined("FORM.myNum1")>
    <cfif IsNumeric(FORM.myNum1) and IsNumeric(FORM.myNum2)>
    <p>The maximum of the two numbers is <cfoutput>#Max(FORM.myNum1,
     FORM.myNum2)#</cfoutput>
    <p>The minimum of the two numbers is <cfoutput>#Min(FORM.myNum1,
     FORM.myNum2)#</cfoutput>
    <cfelse>
    <p>Please enter two numbers
    </cfif>
</cfif>

<form action = "max.cfm">
<h3>Enter two numbers, see the maximum and minimum of them</h3>

Number 1 <input type = "Text" name = "MyNum1">
<br>Number 2 <input type = "Text" name = "MyNum2">

<br><input type = "Submit" name = "" value = "See results">
</form>
```

Mid

説明

文字列から部分文字列を取り出します。

戻り値

文字列。**start** で開始される、**count** の長さの、**string** 内の文字のセットです。

カテゴリ

文字列関数

関数のシンタックス

```
Mid(string, start, count)
```

関連項目

Left、Len、Right

パラメータ

パラメータ	説明
string	文字列、または文字列を含んでいる変数です。一重引用符または二重引用符で囲む必要があります。
start	正の整数、または正の整数を含んでいる変数です。取り出しを開始する位置を指定します。先頭は 0 ではなく 1 です。
count	正の整数、または正の整数を含んでいる変数です。返す文字の数を指定します (0 は無効ですが、エラーは発生しません)。

例

```
<h3>Mid Example</h3>

<cfif IsDefined("Form.myText")>
    <!--- If len returns 0 (zero), then show error message. --->
    <cfif Len(Form.myText)>
        <cfif Len(Form.myText) LTE Form.RemoveChars>
            <cfoutput><p style="color: red; font-weight: bold">Your string
            #Form.myText# only has #Len(Form.myText)# characters. You cannot output
            the #Form.removeChars# middle characters of this string because it is
            not long enough.</p></cfoutput>
        <cfelseif Form.startPos GTE Len(Form.myText)>
            <cfoutput><p style="color: red; font-weight: bold">Your string
            #Form.myText# only has #Len(Form.myText)# characters. You cannot start
            at position #Form.startPos#.</p></cfoutput>
        <cfelse>
            <cfoutput><p>Your original string: <strong>#Form.myText#</strong></p>
            <p>Your changed string, showing only the <strong>#Form.removeChars#
            </strong> middle characters: <strong>#Mid(Form.myText,
            Form.startPos, Form.removeChars)#</strong></p></cfoutput>
        </cfif>
    <cfelse>
        <p style="color: red; font-weight: bold">Please enter a string of more
        than 0 (zero) characters.</p>
    </cfif>
</cfif>

<form action="<cfoutput>#CGI.ScriptName#</cfoutput>" method="POST">
<p>Type in some text<br />
<input type="Text" name="myText"></p>
<p>Enter a starting position (from the beginning of the entered text)<br />
<input name="startPos" type="text" size="1"></p>
<p>How many characters do you want to show?
<select name="RemoveChars">
<option value="1">1
<option value="3" selected>3
<option value="5">5
<option value="7">7
<option value="9">9</select>
<input type="Submit" name="Submit" value="Remove characters"></p>
</form>
```

Min

説明

2 つの数値のいずれが小さいかを調べます。

戻り値

2 つの数値のうち小さい方の値

カテゴリ

算術関数

関数のシンタックス

```
Min(number1, number2)
```

関連項目
Max

パラメータ

パラメータ	説明
number1、number2	数値

例
```
<h3>Min Example</h3>
<cfif IsDefined("FORM.myNum1")>
    <cfif IsNumeric(FORM.myNum1) and IsNumeric(FORM.myNum2)>
    <p>The maximum of the two numbers is <cfoutput>#Max(FORM.myNum1,
     FORM.myNum2)#</cfoutput>
    <p>The minimum of the two numbers is <cfoutput>#Min(FORM.myNum1,
     FORM.myNum2)#</cfoutput>
    <cfelse>
    <p>Please enter two numbers
    </cfif>
</cfif>

<form action = "min.cfm">
<h3>Enter two numbers, and see the maximum and minimum of the two numbers</h3>

Number 1 <input type = "Text" name = "MyNum1">
<br>Number 2 <input type = "Text" name = "MyNum2">

<br><input type = "Submit" name = "" value = "See results">
</form>
```

Minute

説明
日付時刻オブジェクトから分の値を取り出します。

戻り値
0 ～ 59 の範囲の分の序数値

カテゴリ
日付および時刻関数

関数のシンタックス
```
Minute(date)
```

関連項目
DatePart、Hash、Second

パラメータ

パラメータ	説明
date	日付時刻オブジェクトです。値の範囲は西暦 100 ～ 9999 年です。

使用方法
日付時刻値を文字列として渡すときは、その値を引用符で囲みます。引用符で囲まない場合、その値は日付時刻オブジェクトの数値表現として解釈されます。

例
```
<h3>Minute Example</h3>

<cfoutput>
The time is currently #TimeFormat(Now())#.
We are in hour #Hour(Now())#, Minute #Minute(Now())#
and Second #Second(Now())# of the day.
</cfoutput>
```

Month

説明
日付時刻オブジェクトから月の値を取り出します。

戻り値
1 (1 月) ～ 12 (12 月) の範囲の月の序数値

カテゴリ
日付および時刻関数

関数のシンタックス
```
Month(date)
```

関連項目
DatePart、MonthAsString、Quarter

パラメータ

パラメータ	説明
date	日付時刻オブジェクトです。値の範囲は西暦 100 ～ 9999 年です。

使用方法
日付時刻値を文字列として渡すときは、その値を引用符で囲みます。引用符で囲まない場合、その値は日付時刻オブジェクトの数値表現として解釈されます。

注意： CreateDate 関数または Now 関数をこの関数の date パラメータとして渡すことができます。例 :
#Month(CreateDate(2001, 3, 3))#

例

```
<h3>Month Example</h3>
<cfif IsDefined("FORM.year")>
More information about your date:
<cfset yourDate = CreateDate(FORM.year, FORM.month, FORM.day)>
<cfoutput>
    <p>Your date, #DateFormat(yourDate)#.
    <br>It is #DayofWeekAsString(DayOfWeek(yourDate))#, day
     #DayOfWeek(yourDate)# in the week.
    <br>This is day #Day(yourDate)# in the month of
     #MonthAsString(Month(yourDate))#, which has
     #DaysInMonth(yourDate)# days.
    <br>We are in week #Week(yourDate)# of #Year(yourDate)#
     (day #DayofYear(yourDate)# of #DaysinYear(yourDate)#). <br>
    <cfif IsLeapYear(Year(yourDat"))>This is a leap year
        <cfelse>This is not a leap year
    </cfif>
</cfoutput>
</cfif>
```

MonthAsString

説明

month_number に対応する月の名前を確認します。

戻り値

特定の月の名前を表す現在のロケールの文字列

カテゴリ

日付および時刻関数、文字列関数

関数のシンタックス

```
MonthAsString(month_number [, locale])
```

関連項目

DatePart、Month、Quarter

履歴

ColdFusion 8: locale パラメータが追加されました。

パラメータ

パラメータ	説明
month_number	1 ～ 12 の整数
locale	関数を処理するときに、ページのロケールの代わりに使用するロケールです。

例

```
<h3>MonthAsString Example</h3>

<cfif IsDefined("FORM.year")>
<p>More information about your date:
<cfset yourDate = CreateDate(FORM.year, FORM.month, FORM.day)>

<cfoutput>
<p>Your date, #DateFormat(yourDate)#.
<br>It is #DayofWeekAsString(DayOfWeek(yourDate))#, day
    #DayOfWeek(yourDate)# in the week.
<br>This is day #Day(YourDate)# in the month of
    #MonthAsString(Month(yourDate))#, which has
#DaysInMonth(yourDate)# days.
<br>We are in week #Week(yourDate)# of #Year(yourDate)#
    (day #DayofYear(yourDate)# of #DaysinYear(yourDate)#). <br>
    <cfif IsLeapYear(Year(yourDate))>This is a leap year
        <cfelse>This is not a leap year
    </cfif>
</cfoutput>
</cfif>
```

Now

説明

ColdFusion サーバーが動作しているコンピュータの現在の日付と時刻を取得します。戻り値は、パラメータとして DaysInYear や FirstDayOfMonth などの日付関数に渡すことができます。

戻り値

ColdFusion サーバーが動作しているコンピュータの現在の日付と時刻を表す日付時刻オブジェクト

カテゴリ

日付および時刻関数

関数のシンタックス

```
Now()
```

関連項目

CreateDateTime、DatePart

例

```
<h3>Now Example</h3>
<p>Now returns the current date and time as a valid date/time object.

<p>The current date/time value is <cfoutput>#Now()#</cfoutput>
<p>You can also represent this as <cfoutput>#DateFormat(Now())#,
    #TimeFormat(Now())#</cfoutput>
```

NumberFormat

説明
独自に形式設定した数値を作成します。米国で使用される数値形式をサポートします。国際数値形式については、LSNumberFormat を参照してください。

戻り値
形式設定された数値

- マスクが指定されていない場合は、1000 単位のセパレータ付きの整数で値が返されます。
- パラメータ値が "" (空の文字列) の場合、0 が返されます。

カテゴリ
表示および書式制御関数

関数のシンタックス
```
NumberFormat(number [, mask ])
```

関連項目
DecimalFormat、DollarFormat、IsNumeric、LSNumberFormat

履歴
ColdFusion MX: 動作が変更されました。指定されたマスク形式で数値を正しくマスクできない場合は、そのままの数値が返されます (数値が丸められたり、エラーが発生したりすることはありません)。マスクが選択されていない場合、ColdFusion MX では ColdFusion 5 と同様に小数部分が丸められます。たとえば、34.567 は 35 となります。

パラメータ

パラメータ	説明
number	数値。
mask	文字列、または文字列を含んでいる変数です。数値の表示方法を指定する文字のセットです。

次の表でマスク文字を解説します。

マスク文字	説明
アンダースコア (_)	オプション。桁プレースホルダーです。
9	オプション。桁プレースホルダーです。アンダースコア (_) よりも明確に小数点の位置を示します。
.	この位置に小数点を固定します。
0	固定した小数点の左または右に置いて、0 でパディングすることを示します。
( )	数値が 0 よりも小さい場合は、マスクを丸括弧で囲みます。
+	正の数の前にプラス記号 (+) を付け、負の数の前にマイナス記号 (-) を付けます。
-	正の数の前にスペースを付け、負の数の前にマイナス記号 (-) を付けます。
,	3 桁ごとにカンマで区切ります。

マスク文字	説明
L、C	マスク列の中で数値を左揃えまたは中央揃えにします。L または C はマスクの先頭に指定する必要があります。デフォルトは右揃えです。
$	形式設定された数値の先頭にドル記号を置きます。ドル記号 ($) はマスクの先頭に指定する必要があります。
^	右側の形式と左側の形式を区切ります。

注意：マスクで符号が指定されていない場合は、正の数値と負の数値の桁位置が揃いません。正の数値の前にプラス記号またはスペースを付け、負の数の前にマイナス記号を付けるには、プラス記号またはマイナス記号のマスク文字をそれぞれ使用します。

使用方法

この関数では、すべてのプラットフォームで Java 標準のロケール形式設定ルールが使用されます。

形式マスク内の記号の位置によって、これらのコードが作用する位置は異なります。たとえば、ドル記号を形式マスクの完全な左端に配置すると、形式設定された数値のフィールド全体の左端にドル記号が表示されます。形式マスクの左端とドル記号の間に 1 つまたは複数のアンダースコアを置くと、形式設定された数値のフィールド内で、数字のすぐ左に隣接してドル記号が表示されます。

次の例は、記号の位置がどのように形式設定に影響するかを示しています。

数値	マスク	結果
4.37	$____.__	"$ 4.37"
4.37	_$___.__	" $4.37"

同様に、負の数値にマイナス記号 (-) を表示する位置も次のように指定できます。

数値	マスク	結果
-4.37	-____.__	"- 4.37"
-4.37	_-___.__	" -4.37"

記号の位置は、左端、左隣、右隣、および右端です。左右の位置は、コード文字が表示される小数点の位置によって決まります。小数点以下の桁数が固定されていない形式の場合は、キャレット (^) を使用して左右のフィールドを区切ることができます。

コードを離れた位置に置くか、隣接した位置に置くかは、アンダースコア (_) によって決まります。ほとんどのコード文字は、いずれのフィールドに置くかによって効果が異なります。次の例は、負の数値を表示するための括弧を置く位置を指定する方法を示しています。

数値	マスク	結果
3.21	C(__^__)	"(3.21)"
3.21	C__(^__)	" (3.21)"
3.21	C(__^)__	"(3.21) "
3.21	C__(^)__	" (3.21) "

文字列を double 型の数値に変換するとき、この関数では、四捨五入による誤差を防ぐため、変換後の数値に四捨五入係数 1.5543122344752E-014 を加算します。たとえば、四捨五入係数を追加しないで、文字列値 1.275 を小数点以下 2 桁の double 値に変換すると、値は 1.27499999999999999 となり、1.27 に四捨五入されます。四捨五入係数を加えることにより、正しく変換が行われ、結果は 1.28 となります。

ただし、1.9949999999999999999999999999 のように 10E-14 の桁まで使用されている double 値の場合、四捨五入係数が加算されることで、誤った四捨五入の結果が生じる可能性があります。

日付、時刻、数値、および通貨の値のデフォルトの表示形式を設定するには、SetLocale 関数を使用してください。

例

```
<h3>NumberFormat Example</h3>

<cfloop FROM = 1000 TO = 1020 INDEX = "counter">
<cfset CounterRoot2 = counter * sqr(2)>

<!--- Show result in default format, adding comma for thousands place;
    and in custom format, displaying to two decimal places --->
<cfoutput>
<pre>#counter# * Square Root of 2: #NumberFormat(CounterRoot2,
'_____.__')#</pre>
<pre>#counter# * Square Root of 2: #NumberFormat(CounterRoot2)#</pre>
</cfoutput>
</cfloop>
```

ObjectEquals

説明

クライアントとサーバーにある CFC インスタンスの間の競合を特定します。

戻り値

カテゴリ

715 ページの「その他の関数」

関数のシンタックス

```
objectEquals( Param1 ,Param2)
```

関連項目

『ColdFusion アプリケーションの開発』の Conflict management

パラメータ

パラメータ	説明
param 1	クライアント上の CFC の新しいインスタンスです。
param 2	CFC の元のインスタンスです。

例

```
<cfset isNotConflict = ObjectEquals(originalobject, serverobject)>
<cfif isNotConflict>
<cfif operation eq "UPDATE">
<cfset obj = ORMGetSession().merge(clientobject)>
<cfset EntitySave(obj)>
<cfelseif operation eq "DELETE">
<cfset obj = ORMGetSession().merge(originalobject)>
<cfset EntityDelete(obj)>
</cfif>
<cfelse><!----Conflict--->
<cflog text = "is a conflict">
<cfset conflict = CreateObject("component","CFIDE.AIR.conflict")>
<cfset conflict.serverobject = serverobject>
<cfset conflict.clientobject = clientobject>
<cfset conflict.originalobject = originalobject>
<cfset conflict.operation = operation>
<cfset conflicts[conflictcount++] = conflict>
<cfcontinue>
```

ObjectLoad

説明

シリアル化された ColdFusion 配列、CFC、DateTime オブジェクト、Java オブジェクト、クエリー、または構造体を、対応するオブジェクトとしてメモリにロードします。

戻り値

シリアル化解除された ColdFusion オブジェクト (CFC やクエリーオブジェクトなど)

カテゴリ

その他の関数

関数のシンタックス

```
ObjectLoad(binaryObject)
ObjectLoad(filepath)
```

関連項目

ObjectSave

パラメータ

パラメータ	説明
binaryObject	ObjectSave 関数から返されるバイナリオブジェクトです。
filepath	クエリーや CFC などのシリアル化された複合オブジェクトを含むファイルへのパスを指定する文字列、または複合オブジェクトのシリアル化可能なバイナリ表現である変数。 このパラメータには、ファイルまたは SaveCFObject 関数によって返されるオブジェクトの名前を指定する必要があります。

使用方法

この関数は、有用な期間が比較的長く、取得するのに多くの時間またはリソースを必要とするような、動的データを処理するときに役に立ちます。そしてそのデータをファイルに保存し、複数のアプリケーションインスタンスで使用することができます。

たとえば、実行に長い時間がかかり、それほど頻繁には更新されないデータを取得するクエリーを格納した CFC を作成できます。最初に ObjectSave 関数を使用して CFC をファイルとして保存すれば、その後のアプリケーション起動時に CFC ファイルをシリアル化解除することで、アプリケーションのパフォーマンスが向上します。

例

```
<h3>Loading and saving an object.</h3>

<!--- Create the component object. --->
<cfobject component="tellTime" name="tellTimeObj">
<!--- Save the component object to a file. --->
<cfset ObjectSave(tellTimeObj, "data.out")/>
<!--- Load the component object again. --->
<cfset ObjLoaded = ObjectLoad("data.out") >
<!--- Invoke the methods from loaded objects. --->
<cfinvoke component="#ObjLoaded#" method="getLocalTime" returnvariable="localTime">
<cfinvoke component="#ObjLoaded#" method="getUTCTime" returnvariable="UTCTime">
<!--- Display the results. --->
<h3>Time Display Page</h3>
<cfoutput>
Server's Local Time: #localTime#<br>
Calculated UTC Time: #UTCTime#
</cfoutput>
```

ObjectSave

説明

ColdFusion 配列、CFC、DateTime オブジェクト、Java オブジェクト、クエリー、または構造体をシリアル化可能なバイナリオブジェクトに変換します。オプションで、オブジェクトをファイルに保存できます。

戻り値

シリアル化可能な、オブジェクトのバイナリ表現

カテゴリ

その他の関数

関数のシンタックス

```
ObjectSave(object[, filePath])
```

関連項目

ObjectLoad

パラメータ

パラメータ	説明
object	シリアル化する、クエリーや CFC などの複合オブジェクトです。
filePath	シリアル化されたデータを保存するファイルのパスです。

使用方法

この関数は、有用な期間が比較的長く、取得するのに多くの時間またはリソースを必要とするような、動的データを処理するときに役に立ちます。そしてそのデータをファイルに保存し、複数のアプリケーションインスタンスで使用することができます。

たとえば、実行に長い時間がかかり、それほど頻繁には更新されないデータを取得するクエリーを格納した CFC を作成できます。最初に ObjectSave 関数を使用して CFC をファイルとして保存し、その後のアプリケーション起動時に CFC ファイルをシリアル化解除すると、アプリケーションのパフォーマンスが向上します。

例

```
<h3>Saving and loading an object</h3>

<!--- Create the component object. --->
<cfobject component="tellTime" name="tellTimeObj">
<!--- Save the component object to a file. --->
<cfset ObjectSave(tellTimeObj, "data.out")/>

<!--- Load the component object again. --->
<cfset ObjLoaded = ObjectLoad("data.out") >

<!--- Invoke the methods from loaded objects. --->
<cfinvoke component="#ObjLoaded#" method="getLocalTime" returnvariable="localTime">
<cfinvoke component="#ObjLoaded#" method="getUTCTime" returnvariable="UTCTime">
<!--- Display the results. --->
<h3>Time Display Page</h3>
<cfoutput>
Server's Local Time: #localTime#<br>
Calculated UTC Time: #UTCTime#
</cfoutput>
```

onWSAuthenticate

説明

ユーザーを認証します。

シンタックス

OnWSAuthenticate(username, password, connectionInfo)

パラメータ

パラメータ	説明
username	認証するユーザーの名前です。
password	ユーザーのパスワードです。
connectionInfo	次のキーが含まれる構造体です。 • Authenticated：YES|NO • ConnectionTime：接続タイムスタンプ。 • clientID：クライアントの一意の ID。 また、カスタムキーもサポートされます。 例えば、ユーザーのロール、ステータスまたは年齢を指定できます。 connectionInfo は、所定の WebSocket クライアントのすべてのチャネルで共有されます。また、変更は、そのクライアントのすべてのサブスクリプションにわたって保持されます。

例

次の例では、onWSAuthenticate 関数を使用して、ユーザーを検証し、ユーザーロールの関連付けを行います。

注意：この例が機能するには、ユーザー定義関数を実装する必要があります。

```
component
{
    this.name="websocketsampleapp23";
    this.wschannels=[{name="stocks",cfclistener="stocksListener"}];

    function  onWSAuthenticate(username, password, connectionInfo)
       {
          //write appropriate logic to fetch user password in funtion checkPassword

          If(checkPassword(username) eq password)
          {
            connectionInfo.authenticated="YES";
        //Role is the custom information that you provide
     connectionInfo.role= "admin";

            return true;
          }
          else{
            connectionInfo.authenticated="NO";
           return false;
          }
          writedump("#connectionInfo#","console");

         }

}
```

ORMClearSession

説明
指定したデータソースに関連付けられている Hibernate セッションをクリアします。

この関数は、第 1 レベルのキャッシュをクリアし、データベースにまだ保存されていないオブジェクトを削除します。

データソースを指定しなかった場合は、デフォルトのデータソースに関連付けられている Hibernate セッションがクリアされます。

カテゴリ
ORM 関数

関数のシンタックス
Ormclearsession([datasource])

関連項目
ORMCloseSession、ORMGetSession、ORMFlush、ORMGetSessionFactory、『ColdFusion アプリケーションの開発』の ORMClearSession()

ORMCloseSession

説明
リクエストのデータソースに関連付けられている Hibernate セッションを閉じます。データソースを指定しなかった場合は、デフォルトのデータソースに関連付けられている Hibernate セッションが閉じられます。

カテゴリ
ORM 関数

関数のシンタックス
ormclosesession([datasource])

関連項目
ORMGetSession、ORMClearSession、ORMFlush、ORMGetSessionFactory、『ColdFusion アプリケーションの開発』の ORMCloseSession()

ORMCloseAllSessions

説明
リクエストのすべての Hibernate セッションを閉じます。

関数のシンタックス
ormcloseallsessions()

関連項目
ORMGetSession、ORMClearSession、ORMFlush、ORMGetSessionFactory、『ColdFusion アプリケーションの開発』の ORMCloseSession()

ORMEvictCollection

説明
このメソッドは、特定のエンティティ名に対応するコレクションデータまたは関連付けデータをセカンダリキャッシュからすべて削除する場合に使用します。プライマリキーを指定した場合は、そのプライマリキーを持つエンティティのコレクションデータまたは関連付けデータが削除されます。

カテゴリ
ORM 関数

関連項目
ORMEvictEntity、ORMEvictQueries、『ColdFusion アプリケーションの開発』の Evict content from secondary cache

関数のシンタックス
```
ormevictcollection("<entity_name>", "<collection_name>", [primarykey])
```

パラメータ

パラメータ	説明
entity name	永続 CFC のエンティティ名です。
collection_name	コンポーネント内のコレクションの名前
primarykey	エンティティのコレクションデータまたは関連付けデータのプライマリキー

例
コンポーネント CArtists に属するコレクション arts の関連付けデータまたはコレクションデータをすべて削除するには :

```
<cfset ORMEvictCollection("CArtists", "arts")>
```

ORMEvictEntity

説明
このメソッドは、特定のエンティティ名に対応する項目をセカンダリキャッシュから削除する場合に使用します。プライマリキーを指定した場合は、そのプライマリキーを持つエンティティのデータが削除されます。単純なプライマリキーの場合は値を使用し、複合的なプライマリキーの場合は構造体を使用する必要があります。

カテゴリ
ORM 関数

関連項目
ORMEvictCollection、ORMEvictQueries、『ColdFusion アプリケーションの開発』の Evict content from secondary cache

関数のシンタックス

```
ORMEvictEntity("<entity_name>", [primarykey])
```

パラメータ

パラメータ	説明
component_name	永続 CFC のエンティティ名です。
primarykey	コンポーネントのプライマリキー値

例
CArtist エンティティのキャッシュデータをすべて削除するには :

```
<cfset ORMEvictEntity("CArtists")>
```

プライマリキーが 1 である CArtists エンティティのキャッシュデータを削除するには :

```
<cfset ORMEvictEntity("CArtists", 1)>
```

ORMEvictQueries

説明
このメソッドは、指定したデータソースのデフォルトのクエリーキャッシュからすべてのクエリーのデータを削除する場合に使用します。キャッシュ名を指定した場合は、そのキャッシュ名を持つキャッシュ領域に属するすべてのクエリーのデータが削除されます。

データソースを指定しなかった場合は、デフォルトのデータソースのデフォルトのクエリーキャッシュが削除されます。

カテゴリ
ORM 関数

関連項目
ORMEvictEntity、ORMEvictCollection、『ColdFusion アプリケーションの開発』の Evict content from secondary cache

シンタックス
ORMEvictQueries([cachename]) ORMEvictQueries([cachename], datasource)

パラメータ

パラメータ	説明
cachename	削除するキャッシュ領域の名前です。
datasource	削除するキャッシュに関連付けられているデータソースの名前です。キャッシュを指定しなかった場合は、デフォルトのクエリーキャッシュが削除されます。

例
デフォルトのクエリーキャッシュからすべてのクエリーのデータを削除します。

```
<cfset ORMEvictQueries()>
```

availableArtsCache という名前のキャッシュ領域からすべてのクエリーのデータを削除します。

```
<cfset ORMEvictQueries("availableArtsCache")>
```

ORMExecuteQuery

説明
Hibernate Query Language (HQL) クエリーを実行します。

デフォルトでは、この関数は、ORM のデフォルトのデータソースに対して実行されます。別のデータソースに対してこの関数を使用する場合は、queryoptions の中でデータソースのキーと値のペアを指定します。

シンタックス
ORMExecuteQuery(hql, [params] [,unique]) ORMExecuteQuery(hql, [,unique] [, queryoptions]) ORMExecuteQuery(hql, params [,unique] [,queryOptions])

パラメータ

パラメータ	説明
hql	実行する HQL クエリーです。
params	エンティティのオブジェクトパラメータです。
unique	オブジェクトパラメータが一意であるかどうかを指定します。
queryoptions	クエリーのオプションのキーと値のペアです。

例

```
<cfset artistArr = ORMExecuteQuery("from Artists where artistid=1", true, {datasource="cfartgallery"})>
<cfset countArray = ORMExecuteQuery("select count(*) from Authors", [], false, {datasource="cfbookclub"})>
```

ORMFlush

説明
リクエストのデータソースに関連付けられている Hibernate セッションを消去します。ORMFlush を呼び出すと、そのリクエストで保留中の CRUD 操作がすべて消去されます。現在の ORM セッションでオブジェクトに加えられた変更はすべて、データベースに保存されます。

データソースを指定しなかった場合は、デフォルトのデータソースに関連付けられている Hibernate セッションが消去されます。

カテゴリ
ORM 関数

関数のシンタックス
ormflush([datasource])

関連項目
ORMCloseSession、ORMClearSession、ORMGetSession、ORMGetSessionFactory、『ColdFusion アプリケーションの開発』の ORMFlush()

ORMFlushall

説明
リクエストの現在の Hibernate セッションをすべて消去します。

関数のシンタックス
ormflushall()

関連項目
ORMCloseSession、ORMClearSession、ORMGetSession、ORMGetSessionFactory、『ColdFusion アプリケーションの開発』の ORMFlush()

ORMGetSession

説明
リクエストのデータソースに関連付けられている Hibernate セッションを返します。このデータソースが ORM に設定されていない場合は、例外が発生します。データソースが指定されていない場合は、デフォルトのデータソースの Hibernate セッションが返されます。

このセッションオブジェクトを使用して、通常は ColdFusion で公開されていない API を呼び出します。

セッション API については、次の資料を参照してください。

http://docs.jboss.org/hibernate/core/3.3/api/org/hibernate/Session.html

カテゴリ
ORM 関数

関数のシンタックス
ormgetsession([datasource])

関連項目
ORMCloseSession、ORMClearSession、ORMFlush、ORMGetSessionFactory、『ColdFusion アプリケーションの開発』の ORMGetSession()

ORMGetSessionFactory

説明
データソースに関連付けられている Hibernate セッションファクトリオブジェクトを返します。このデータソースが ORM に設定されていない場合は、エラーが発生します。データソースを指定しなかった場合は、デフォルトのデータソースに関連付けられている Hibernate セッションファクトリオブジェクトが返されます。

Session API の詳細については、次の URL を参照してください。

http://docs.jboss.org/hibernate/core/3.3/api/org/hibernate/SessionFactory.html

カテゴリ
ORM 関数

関数のシンタックス
Ormgetsessionfactory([datasource])

関連項目
ORMCloseSession、ORMClearSession、ORMFlush、『ColdFusion アプリケーションの開発』の ORMGetSession

ORMIndex

説明
オフラインのインデックス作成を実行します。

シンタックス
ORMIndex();

ORMIndex("***entity_name***");

ORMIndex("***entityName_list***");

ORMIndex (***entityObject***);

パラメータ

パラメータ	説明
entityName	インデックスを作成するエンティティの名前です。
entityName_list	インデックスを作成するエンティティ名のカンマ区切りリストです。
entityObject	インデックスを作成する特定のエンティティインスタンスの変数名です。

使用方法
パラメーターを指定せずにこの関数を使用した場合は、所定のアプリケーションのすべての永続エンティティのインデックスが作成されます。

例
```
EmpObjs = EntityLoad("Employee",{lastname="Bond"});
    for(EmpObj in EmpObjs)
    {
    ormindex(EmpObj);
    }
```

ORMIndexPurge

説明
現在の Application スコープのすべてのエンティティまたは指定したエンティティのインデックスデータをすべて消去します。

シンタックス
ORMIndexPurge();

ORMIndexPurge("entityName");

ORMIndexPurge("***entityName_list***");

パラメータ

パラメータ	説明
entityName	ロードされるエンティティの名前です。
entityName_list	消去するエンティティ名のカンマ区切りリストです。

使用方法
entityName を指定せずにこの関数を使用した場合は、アプリケーションのすべての永続エンティティが消去されます。

例
```
ORMIndexPurge();

ORMIndexPurge("Employee");
```

ORMReload

説明
アプリケーションに対して ORM を再初期化します。

CFC の永続メタデータを変更した場合は、ORM のリロードが必要になる場合があります。

戻り値
ORM セッションファクトリのインスタンスを返します。

カテゴリ
715 ページの「ORM 関数」

関数のシンタックス

```
ORMReload()
```

使用方法

この関数は、開発中にのみ使用することをお勧めします。

関連項目

ColdFusion ORM

例

```
component
{
    this.name = Hash( GetCurrentTemplatePath() );
    /* define the application wide datasource */
    this.datasource = "cfartgallery";
    /* enable hibernate support for this application */
    this.ormenabled = true;
    /* create a struct of ORM settings */
    this.ormsettings = {};
    /* turn on event handling */
    this.ormsettings.eventhandling = true;
    /**
    * @output true
    */
    public boolean function onRequestStart( targetPage )
    {
        /* this is to ensure that ORM is up-to-date for demo */
        ORMReload();
        return true;
    }
}
```

ORMSearch

説明

特定のプロパティまたはエンティティに含まれる指定したテキストを検索します。

戻り値

次のものが含まれる構造体

- 次の形式の構造体の配列（entity および score のキーが付いたもの）

  ```
  data -[{entity: entity1, score: entity1_score}, {entity: entity2, score: entity2_score}, ..... ]
  ```

- maxTotalRecord（有効な結果件数）

シンタックス

ORMSearch("query_text", "entityName")

ORMSearch("query_text", "entityName", fields)

ORMSearch("query_text", "entityName", fields, optionMap)

パラメータ

パラメータ	説明
query_text	検索するテキスト、または完全な Lucene クエリです。 ORMSearch("query_text", "entityName") の場合は、Lucene クエリのみサポートされます。 Lucene クエリについて詳しくは、http://lucene.apache.org/core/old_versioned_docs/versions/3_0_0/queryparsersyntax.html を参照してください。
entityName	検索するエンティティの名前です。
fields	検索を実行するフィールドです。これには、文字列の配列を使用できます。 Lucene クエリを実行する場合は、このフィールドを指定する必要はありません。言い換えれば、この値を指定しない場合は、Lucene クエリが実行されます。 フィールド名の大文字と小文字は区別されます。
optionMap	Lucene クエリの実行時に渡すことができる追加オプションです。 オプションは次のとおりです。 • sort：指定した indexfieldname に基づいてソートを実行します。 • offSet：オブジェクトを取得する位置を指定します。 • maxResults：取得されるオブジェクトの最大数を指定します。

使用方法

日付の検索を実行する場合は、次の例に示すように yyyymmdd の形式を使用します。

```
objs = ORMSearch("datecheck:[#dateformat(dateadd("d",5,now()),"yyyymmdd")# TO
#dateformat(dateadd("d",35,now()),"yyyymmdd")#]","C2",[],{maxresults=2});
```

時刻の検索を実行する場合は、UTC の形式を使用します。

例 1：Lucene クエリに基づいた ORM 検索

```
ORMSearch("FirstName:ch*","Employee");
ORMSearch("ch*","Employee",["FirstName"]);
objs = ORMSearch('FirstName:ch*',"Employee",[],{sort="salary",maxresults=5,offset=2});
```

例 2：複数のエンティティの ORM 検索

```
ORMSearch("john*","DeveloperEntity,UserEntity",["firstname"]);
```

この例では、DeveloperEntity と UserEntity で firstname が検索され、複数のエンティティの複合配列が返されます。

例 3：Lucene クエリに基づいたすべてのサブエンティティの ORM 検索

この例は、スーパーエンティティを継承するすべてのサブエンティティに対して ORM 検索を実行する方法を示しています。例えば、USEmployeeEntity と UKEmpoyeeEntity は、EmployeeEntity を拡張していると仮定します。次のコードを使用して、この両方のサブエンティティを検索できます。

```
ORMSearch("john*","EmployeeEntity",["FirstName"]);
```

例 4：関係間の ORM 検索

この例では、製品とカテゴリは多対 1 の関係にあります。次のコードを使用して、特定のカテゴリのすべての製品を検索できます。

```
ORMSearch("CategoryID.CategoryName:In*","cproducts",[]);
```

関連オブジェクトの検索は、多対 1 の関係および 1 対 1 の関係にある場合にのみ機能することに注意してください。

ORMSearchOffline

説明
インデックスが作成されているプロパティの検索を実行しますが、格納されているフィールドのみが返されます。

この関数を機能させるには、検索を実行するプロパティで indexStore=true を指定します。

戻り値
次のものが含まれる構造体

- 次の形式の構造体の配列（entity および score のキーが付いたもの）

```
data -[{entity: entity1, score: entity1_score}, {entity: entity2, score: entity2_score}, ..... ]
```

- maxTotalRecord（有効な結果件数）
- fields_to_be_selected（キー）

シンタックス
ORMSearchOffline(query_text, entityName, fields_to_be_selected);

ORMSearchOffline(query_text, entityName, fields_to_be_selected, fields);

ORMSearchOffline(query_text, entityName, fields_to_be_selected, fields, optionMap);

パラメータ

パラメータ	説明
query_text	検索するテキスト、または完全な Lucene クエリです。 Lucene クエリについて詳しくは、http://lucene.apache.org/core/old_versioned_docs/versions/ を参照してください。
entityName	検索するエンティティの名前です。
fieldss_to_be_selected	結果の構造体でキーとして返されるフィールドです。
fields	検索を実行するフィールドです。
optionMap	Lucene クエリの実行時に渡すことができる追加オプションです。指定可能なオプションは次のとおりです。 • sort：指定した indexfieldname に基づいてソートを実行します。 • offSet：オブジェクトを取得する位置を指定します。 • maxResults：取得されるオブジェクトの最大数を指定します。

例 1
```
ORMSearchOffline('FirstName:"ch*"',"Employee",["id","firstname"]);
```

例 2
次の例では、FirstName プロパティのオフライン検索が実行され、結果の構造体に姓と名がキーとして返されます。

```
ORMSearchOffline("ch*","Employee",["FirstName","LastName"],["FirstName"],{sort="salary",maxresults=5,offset=2});
```

例 3
この例では、クエリの resultObj は構造体の配列です。個々の構造体には、（3 番目のパラメーターで渡された）すべての選択フィールドが含まれます。

```
<cfset resultObj =ORMSearchOffline('Java Rocks', 'Book', [bookId, summary, Author.name, title],[title,
short_summary])>
```

ParagraphFormat

説明
文字列内の文字を次のように置き換えます。

- 1 つの改行文字 (CR/LF シーケンス) をスペースに置換します。
- 2 つの改行文字を HTML の段落タグ (<p>) に置換します。

戻り値
置換後の文字列のコピー

カテゴリ
表示および書式制御関数、文字列関数

関数のシンタックス
```
ParagraphFormat(string)
```

関連項目
StripCR

パラメータ

パラメータ	説明
string	文字列、または文字列を含んでいる変数です。

使用方法
この関数は、textarea フィールドに入力されたデータを表示するときに役立ちます。

例
```
<h3>ParagraphFormat Example</h3>
<p>Enter text into this textarea, and see it returned as HTML.
<cfif IsDefined("FORM.myTextArea")>
    <p>Your text area, formatted
    <p><cfoutput>#ParagraphFormat(FORM.myTextArea)#</cfoutput>
</cfif>
<!--- use #Chr(10)##Chr(13)# to simulate a line feed/carriage
return combination; i.e, a return --->
<form action = "paragraphformat.cfm">
<textArea name = "MyTextArea" cols = "35" ROWS = 8>
This is sample text and you see how it scrolls
    <cfoutput>#Chr(10)##Chr(13)#</cfoutput>
    From one line
    <cfoutput>#Chr(10)##Chr(13)##Chr(10)##Chr(13)#</cfoutput>
    to the next
</textArea>
<input type = "Submit" name = "Show me the HTML version">
</form>
```

ParameterExists

説明
この関数は非推奨となっています。代わりに IsDefined 関数を使用してください。

パラメータが存在するかどうかを調べます。引数の評価は行われません。

履歴
ColdFusion MX: この関数が非推奨になりました。今後のリリースではこのタグは機能せずエラーを引き起こす可能性があります。

ParseDateTime

説明
English (U.S.) ロケールの表記規則に従って、日付時刻文字列を、日付時刻オブジェクトに変換します。他のロケールで日付時刻文字列を形式設定するには、LSParseDateTime 関数を使用します。

戻り値
日付時刻オブジェクトです。

カテゴリ
日付および時刻関数、表示および書式制御関数

関数のシンタックス
```
ParseDateTime(date/time-string [, pop-conversion ])
```

関連項目
IsDate、IsNumericDate、SetLocale

パラメータ

パラメータ	説明
date/time-string	米国ロケールの表記規則に従って形式設定された日付時刻値を含む文字列です。有効な日付時刻の範囲は、西暦 100 ～ 9999 年です。2 桁の年については、00 ～ 29 の場合は 2000 ～ 2029 年、30 ～ 99 の場合は 1930 ～ 1999 と解釈されます。
pop-conversion	• pop : 日付時刻文字列が POP 形式である (つまり送信元の現地時刻と UTC からのタイムゾーンオフセットが含まれている) ことを指定します。文字列に含まれるオフセットが適用され、UTC 時間で値が返されます。 • standard : これがデフォルトです。POP 変換は行われません。

使用方法
この関数は CreateDateTime と似ていますが、引数には日付時刻値を列挙するのではなく、1 つの文字列を渡します。これらの関数は、複合式のコードを読みやすくすることを主な目的として提供されています。

タイムゾーン間の時差を計算するには、GetTimeZoneInfo 関数を使用します。

日付、時刻、数値、および通貨の値のデフォルトの表示形式を設定するには、SetLocale 関数を使用してください。

例

```
<h3>ParseDateTime Example</h3>
<cfif IsDefined("form.theTestValue")>
    <cfif IsDate(form.theTestValue)>
    <h3>The expression <cfoutput>#DE(form.theTestValue)#</cfoutput> is a valid date</h3>
    <p>The parsed date/time is:
    <cfoutput>#ParseDateTime(form.theTestValue)#</cfoutput>
    <cfelse>
    <h3>The expression <cfoutput>#DE(form.theTestValue)#</cfoutput> is not a valid date</h3>
    </cfif>
</cfif>

<form action="#CGI.ScriptName#" method="POST">
<p>Enter an expression, and discover if it can be evaluated to a date value.
<input type="Text" name="TheTestValue" value="<CFOUTPUT>#DateFormat(Now())#
    #TimeFormat(Now())#</CFOUTPUT>">
<input type="Submit" value="Parse the Date" name="">
</form>
```

Pi

説明

数学定数 p を 15 桁の精度で取得します。

戻り値

数値 3.14159265358979

カテゴリ

算術関数

関数のシンタックス

```
Pi()
```

関連項目

ASin、Cos、Sin、Tan

例

```
<h3>Pi Example</h3>
<!--- By default, ColdFusion displays only 11 significant digits.
    Use NumberFormat to display all 15. --->
The Pi function Returns the number
<cfoutput>
#NumberFormat(Pi(), "_.________________")#,
</cfoutput> the mathematical constant pi, accurate to 15 digits.
```

PrecisionEvaluate

説明

BigDecimal 精度演算を使用して文字列式 (複数指定可) を左から右へダイナミックに評価し、任意精度の計算式の値を計算します。

戻り値
評価結果のオブジェクト

カテゴリ
算術関数、ダイナミック評価関数

関数のシンタックス
```
PrecisionEvaluate(string_expression1 [, string_expression2 , ... ])
```

関連項目
Evaluate、『ColdFusion アプリケーションの開発』の Using Expressions and Number Signs

パラメータ

パラメータ	説明
string_expression1、string_expression2...	評価対象となる式です。

使用方法
PrecisionEvaluate 関数を使用すると、任意の長さ (BigDecimal 精度) の小数値を計算できます。BigDecimal 精度演算では、任意の長さの小数が受け入れられ、任意の長さの小数が生成されます。指数表記は使用されません。

PrecisionEvaluate 関数で取得した任意精度の結果は、加算、減算、乗算、および除算のみに使用できます。次の演算を使用する場合は、通常の整数演算または浮動小数点演算が実行されるので、BigDecimal 値は返されません。

- 累乗 (^)
- 剰余 (MOD または %)
- 整数の除算 (/)

この関数が Evaluate 関数と異なる点は、数値の計算に BigDecimal 精度演算を使用する点のみです。それ以外の動作は完全に同じです。左部分の評価結果が右部分に対して意味を持つことがあります。この関数は一番右にある式の評価結果を返します。文字列式に引用符または二重引用符が含まれている場合は、その引用符をエスケープする必要があります。

1/3 のように評価結果が無限小数になる場合、小数部は 20 桁までに制限されます。

注意：処理効率を高めるため、数式を評価する場合は引用符 (") で囲まないようにします。得られる結果は同じですが、PrecisionEvaluate("a*b") とするよりも PrecisionEvaluate(a*b) としたほうが処理効率が高くなります。

例

```
<h3>PrecisionEvaluate Example</h3>
<cfif IsDefined("FORM.myExpression")>
    <cftry>
        <!--- Evaluate the expression and display the result. --->
        <cfset theExpression = PrecisionEvaluate(Form.myExpression)>
        <cfoutput>
            The value of the expression #FORM.MyExpression#
            is #theExpression#.<br>
        </cfoutput>

        <cfcatch type="any">
            <cfoutput>Could not evaluate the expression #Form.myExpression#.
            </cfoutput>
        </cfcatch>
    </cftry>
</cfif>

<cfform preservedata="yes">
    <h3>Enter a ColdFusion expression for evaluation.</h3>
    <p>Try using some really big decimal numbers.</p>
    <cfinput type="text" name="myExpression" size="60"><br>
    <br>
    <cfinput type="submit" name="submit">
</cfform>
```

PreserveSingleQuotes

説明

変数に含まれている一重引用符 (') が ColdFusion によって自動的にエスケープされるのを防ぎます。引数の評価は行われません。

戻り値

(なし)

カテゴリ

その他の関数

関数のシンタックス

```
PreserveSingleQuotes(variable)
```

履歴

ColdFusion MX: 動作が変更されました。cfquery タグ本文またはブロック内での単純変数、配列変数、および構造体変数の参照は ColdFusion によって自動的にエスケープされます。以前のリリースでは、配列変数の参照は自動的にエスケープされませんでした。

パラメータ

パラメータ	説明
variable	一重引用符をエスケープせず維持する必要がある文字列を含んでいる変数です。

使用方法

SQL ステートメントでこの関数を使用すると、変数の参照の評価を実行時まで遅らせることができます。これにより、データ文字に含まれる一重引用符またはアポストロフィ (たとえば "Joe's Diner") が区切り文字として評価されるのを防ぎ、エラーを防止できます。

例 A : 次のようなコードがあるとします。

```
<cfset mystring = "'Newton's Law', 'Fermat's Theorem'">
PreserveSingleQuotes(#mystring#) is
<cfoutput>
    #PreserveSingleQuotes(mystring)#
</cfoutput>
```

出力は次のようになります。

```
PreserveSingleQuotes(#mystring#) is 'Newton's Law', 'Fermat's Theorem'
```

例 B : 次のようなコードがあるとします。

```
<cfset list0 = " '1','2', '3' ">
<cfquery sql = "select * from foo where bar in (#list0#)">
```

リスト内の一重引用符が ColdFusion によって次のようにエスケープされます。

```
""1"", ""2"", ""3""
```

このため、cfquery タグでエラーが発生します。

この例が正しく動作するようにするには、次のようにコーディングします。

```
<cfquery sql = "select * from foo where bar in (#preserveSingleQuotes(list0)#)"> **tharwood 11/16
```

この関数を使用することで、コードが次のように評価されます。

```
'1', '2', '3'
```

例

```
<h3>PreserveSingleQuotes Example</h3><p>This is a useful function for
    creating lists of information to return from a query. In this example,
    we pick the list of Centers in Suisun, San Francisco, and San Diego,
    using the SQL grammar IN to modify a WHERE clause, rather than looping
    through the result set after the query is run.
<cfset List = "'Suisun', 'San Francisco', 'San Diego'">
<cfquery name = "GetCenters" datasource = "cfdocexamples">
    SELECT Name, Address1, Address2, City, Phone
    FROM Centers
    WHERE City IN (#PreserveSingleQuotes(List)#)
</cfquery>
<p>We found <cfoutput>#GetCenters.RecordCount#</cfoutput> records.
<cfoutput query = "GetCenters">
<p>#Name#<br>
#Address1#<br>
<cfif Address2 is not "">#Address2#
    </cfif>
#City#<br>
#Phone#<br>
</cfoutput>
```

Quarter

説明

指定した日付がどの四半期に属するかを計算します。

戻り値
1 ～ 4 の整数

カテゴリ
日付および時刻関数

関数のシンタックス
```
Quarter(date)
```

関連項目
DatePart、Month

パラメータ

パラメータ	説明
date	日付時刻オブジェクトです。値の範囲は西暦 100 ～ 9999 年です。

使用方法
日付時刻値を文字列として渡すときは、その値を引用符で囲みます。引用符で囲まない場合、その値は日付時刻オブジェクトの数値表現として解釈されます。

例
```
<h3>Quarter Example</h3>

Today, <cfoutput>#DateFormat(Now())#</cfoutput>,
is in Quarter <cfoutput>#Quarter(Now())#</cfoutput>.
```

QueryAddColumn

説明
クエリーに新しい列を追加し、各行のその列に 1 次元配列の内容を挿入します。すべての列の行数が同じになるように、必要に応じてクエリー列のパディングが行われます。

戻り値
追加された列の列番号

カテゴリ
クエリー関数

関数のシンタックス
```
QueryAddColumn(query, column-name [, datatype], array-name)
```

関連項目
QueryNew、QueryAddRow、QuerySetCell、『ColdFusion アプリケーションの開発』の Managing data types for columns

履歴
ColdFusion MX 7: datatype パラメータが追加されました。

ColdFusion MX: 動作が変更されました。無効な名前の列を追加しようとするとエラーが発生します (以前のリリースでは、こうした列を追加することはできましたが、追加後にその列を参照することはできませんでした)。

パラメータ

パラメータ	説明
query	クエリーオブジェクトの名前です。
column-name	新しい列の名前です。
datatype	(オプション) 列のデータ型です。列に追加したデータがこの型ではない場合、またはデータをこの型に変換できない場合、ColdFusion ではエラーが生成されます。次のデータ型が有効です。 • Integer: 32 ビットの整数です。 • BigInt: 64 ビットの整数です。 • Double: 64 ビットの小数です。 • Decimal: java.math.BigDecimal で指定される、可変長の小数です。 • VarChar: 文字列です。 • Binary: バイト配列です。 • Bit: ブール値 (1 = true、0 = false) です。 • Time: 時刻です。 • Date: 日付です。時刻情報を含めることができます。
array-name	配列の名前です。新しい列にはこの配列の要素が挿入されます。

使用方法

cfquery タグで取得したクエリーや QueryNew 関数で作成したクエリーなどのクエリーオブジェクトに列を追加できます。キャッシュされたクエリーに対して QueryAddColumn 関数を使用することはできません。この関数は、Oracle のストアドプロシージャが生成できる出力パラメータの配列からクエリーオブジェクトを生成するときに有用です。

datatype パラメータはオプションですが、常に使用することをお勧めします。このパラメータを使用しない場合、ColdFusion ではクエリーオブクエリー内のクエリーオブジェクトを使用するときに、列のデータ型を判別する必要があります。データ型を判別するには追加処理が必要であり、ColdFusion がデータ型を正しく推測しない場合、エラーが発生する可能性があります。

例

次の例では、クエリーオブジェクトを作成し、QueryAddColumn 関数を使用してクエリーオブジェクトに 3 つの列を追加して、結果を表示します。データを提供する配列のうち 2 つは残り 1 つの配列より短いため、クエリー内の対応する列には QueryAddColumn によってパディングが追加されます。

```
<!--- Make a query. --->
<cfset myQuery = QueryNew("")>

<!--- Create an array. --->
<cfset FastFoodArray = ArrayNew(1)>
<cfset FastFoodArray[1] = "French Fries">
<cfset FastFoodArray[2] = "Hot Dogs">
<cfset FastFoodArray[3] = "Fried Clams">
<cfset FastFoodArray[4] = "Thick Shakes">
<!--- Use the array to add a column to the query. --->
<cfset nColumnNumber = QueryAddColumn(myQuery, "FastFood", "VarChar",
            FastFoodArray)>

<!--- Create a second array. --->
<cfset FineCuisineArray = ArrayNew(1)>
<cfset FineCuisineArray[1] = "Lobster">
<cfset FineCuisineArray[2] = "Flambe">
<!--- Use the array to add a second column to the query. --->
<cfset nColumnNumber2 = QueryAddColumn(myQuery, "FineCuisine", "VarChar",
             FineCuisineArray)>

<!--- Create a third array. --->
<cfset HealthFoodArray = ArrayNew(1)>
<cfset HealthFoodArray[1] = "Bean Curd">
<cfset HealthFoodArray[2] = "Yogurt">
<cfset HealthFoodArray[3] = "Tofu">
<!--- Use the array to add a third column to the query. --->
<cfset nColumnNumber3 = QueryAddColumn(myQuery, "HealthFood", "VarChar",
             HealthFoodArray)>

<!--- Display the results. --->
<table cellspacing = "2" cellpadding = "2" border = "0">
<tr>
<th align = "left">Fast Food</th>
<th align = "left">Fine Cuisine</th>
<th align = "left">Health Food</th>
</tr>
<cfoutput query = "myQuery">
<tr>
<td>#FastFood#</td>
<td>#FineCuisine#</td>
<td>#HealthFood#</td>
</tr>
</cfoutput>
</table>
```

QueryAddRow

説明
指定した数の空の行をクエリーに追加します。

戻り値
クエリーの行数

カテゴリ
クエリー関数

関数のシンタックス

```
QueryAddRow(query [, number])
```

関連項目

QueryAddColumn、QuerySetCell、QueryNew、『ColdFusion アプリケーションの開発』の Creating a recordset with the QueryNew() function

パラメータ

パラメータ	説明
query	実行するクエリーの名前です。
number	クエリーに追加する行の数です。デフォルト値は 1 です。

使用方法

ColdFusion 10 での機能強化により、構造体、構造体の配列、または 1 次元または多次元の配列を指定して、クエリに行を追加できるようになりました。次に例を示します。

```
queryAddRow(myQuery1,
     [
                {id=2,name="Two"},
                {id=3,name="Three"},
                {id=4,name="Four"}
]);
queryAddRow(myQuery2,{id=4,name="Four"});

queryAddRow(myQuery1,
[
[1,"One"],
[2,"Two"],
{3,"Three"]
      ]);
```

例

```
<h3>QueryAddRow Example</h3>

<!--- start by making a query --->
<cfquery name = "GetCourses" datasource = "cfdocexamples">
    SELECT Course_ID, Number, Descript
    FROM Courses
</cfquery>

<p>The Query "GetCourses" has <cfoutput>#GetCourses.RecordCount#</cfoutput> rows.

<cfset CountVar = 0>
<cfloop CONDITION = "CountVar LT 15">
    <cfset temp = QueryAddRow(GetCourses)>
    <cfset CountVar = CountVar + 1>
    <cfset Temp = QuerySetCell(GetCourses, "Number", 100*CountVar)>
    <cfset Temp = QuerySetCell(GetCourses, "Descript",
    "Description of variable #Countvar#")>
</cfloop>

<P>After the QueryAddRow action, the query has <CFOUTPUT>#GetCourses.RecordCount#</CFOUTPUT>
records.
<CFOUTPUT query="GetCourses">
<PRE>#Course_ID# #Number# #Descript#</pre>
</cfoutput>
```

QueryConvertForGrid

説明
クエリーデータを変換して、そのクエリーのサブセット (1 ページ分のデータ) を含む構造体を返します。バインド式に対するレスポンスとして Ajax 形式の cfgrid コントロールにデータを返す CFC 関数で使用します。

戻り値
1 ページ分のクエリーデータを含む構造体

カテゴリ
クエリー関数

関数のシンタックス
```
QueryConvertForGrid(query, page, pageSize)
```

関連項目
cfgrid、『ColdFusion アプリケーションの開発』の Dynamically filling form data

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
query	クエリーの名前です。このクエリーのデータが返されます。
page	クエリーデータを返すページの番号です。ページ番号は 1 から始まります。
pageSize	1 ページあたりに含まれるクエリーデータの行数です。

使用方法
クエリーが一度に 1 ページ分のグリッドデータのみ返す場合、cfgrid のバインド CFC の戻り値は、この関数を使用しなくても作成できます。詳細については、『ColdFusion アプリケーションの開発』の Using Ajax User Interface Components and Features を参照してください。

例
次の例は、Ajax 形式の cfgrid タグの bind 属性が CFC 関数をどのように呼び出すかを示します。グリッドに返すクエリーデータを準備するために QueryConvertForGrid を使用しています。cfgrid タグを含む CFML ページには次のコードがあります。

```
<cfform>
    <cfgrid format="html" name="grid01" pagesize=5 sort=true
            bind="cfc:places.getData({cfgridpage},{cfgridpagesize},
                {cfgridsortcolumn},{cfgridsortdirection})" selectMode="row">
        <cfgridcolumn name="Emp_ID" display=true header="Employee ID"/>
        <cfgridcolumn name="FirstName" display=true header="Name"/>
        <cfgridcolumn name="Email" display=true header="Email"/>
    </cfgrid>
</cfform>
```

places.cfc ページ内の getData 関数には次のコードがあります。

```
<cffunction name="getData" access="remote" output="false">
    <cfargument name="page">
    <cfargument name="pageSize">
    <cfargument name="gridsortcolumn">
    <cfargument name="gridstartdirection">
    <cfset query = "SELECT Emp_ID, FirstName, EMail
         FROM Employees" >
    <cfif gridsortcolumn neq "" or gridstartdirection neq "">
     <cfset query=query & " order by #gridsortcolumn#
            #gridstartdirection#">
    </cfif>
    <cfquery name="team" datasource="cfdocexamples">
         <cfoutput>#query#</cfoutput>
    </cfquery>
    <cfreturn QueryConvertForGrid(team, page, pageSize)>
</cffunction>
```

QueryNew

説明
空のクエリー (クエリーオブジェクト) を作成します。

戻り値
指定した名前の一連の列を持つ空のクエリー、または列を持たない空のクエリー

カテゴリ
クエリー関数

関数のシンタックス
```
QueryNew(columnlist [, columntypelist])
```

履歴
ColdFusion MX 7: columntypelist パラメータが追加されました。

関連項目
QueryAddColumn、QueryAddRow、QuerySetCell、『ColdFusion アプリケーションの開発』の Managing data types for columns

パラメータ

パラメータ	説明
columnlist	カンマで区切った列名のリスト、または空の文字列を指定します。
columntypelist	(オプション) 列のデータ型を指定する、カンマで区切ったリストです。列に追加したデータがこの型ではない場合、またはデータをこの型に変換できない場合、ColdFusion ではエラーが生成されます。次のデータ型が有効です。 • Integer: 32 ビットの整数です。 • BigInt: 64 ビットの整数です。 • Double: 64 ビットの小数です。 • Decimal: java.math.BigDecimal で指定される、可変長の小数です。 • VarChar: 文字列です。 • Binary: バイト配列です。 • Bit: ブール値 (1 = true、0 = false) です。 • Time: 時刻です。 • Date: 日付です。時刻情報を含めることができます。

使用方法

columnlist パラメータで空の文字列を指定した場合は、QueryAddColumn 関数を使用してクエリーに列を追加します。

columntypelist パラメータはオプションですが、常に使用することをお勧めします。このパラメータを使用しない場合、ColdFusion ではクエリーオブクエリー内のクエリーオブジェクトを使用するときに、データ型を判別する必要があります。データ型を判別するには追加処理が必要であり、ColdFusion がデータ型を正しく推測しない場合、エラーが発生する可能性があります。

ColdFusion 10 での機能強化により、クエリデータを初期化できるようになりました。次の例に示すように、構造体、構造体の配列、または 1 次元または多次元の配列を指定して、クエリを初期化できます。

```
myQuery1  = queryNew("id,name","Integer,Varchar", {id=1,name="One"});
myQuery2  = queryNew("id,name","Integer,Varchar",
        [
                {id=1,name="One"},
                {id=2,name="Two"},
                {id=3,name="Three"}
      ]);
myQuery2  = queryNew("id,name","Integer,Varchar",  [  [1,"One"],  [2,"Two"],  {3,"Three"]        ]);
```

例

次の例では、QueryNew 関数を使用して、3 つの列がある空のクエリーを作成します。クエリーの 2 つの行を挿入し、クエリーオブジェクトの内容およびそのメタデータを表示します。

```
<!--- Create a new three-column query, specifying the column data types --->
<cfset myQuery = QueryNew("Name, Time, Advanced", "VarChar, Time, Bit")>

<!--- Make two rows in the query --->
<cfset newRow = QueryAddRow(MyQuery, 2)>

<!--- Set the values of the cells in the query --->
<cfset temp = QuerySetCell(myQuery, "Name", "The Wonderful World of CMFL", 1)>
<cfset temp = QuerySetCell(myQuery, "Time", "9:15 AM", 1)>
<cfset temp = QuerySetCell(myQuery, "Advanced", False, 1)>
<cfset temp = QuerySetCell(myQuery, "Name", "CFCs for Enterprise
        Applications", 2)>
<cfset temp = QuerySetCell(myQuery, "Time", "12:15 PM", 2)>
<cfset temp = QuerySetCell(myQuery, "Advanced", True, 2)>

<h4>The query object contents</h4>
<cfoutput query = "myQuery">
    #Name# #Time# #Advanced#<br>
</cfoutput><br>
<br>
<h4>Using individual query data values</h4>
<cfoutput>
    #MyQuery.name[2]# is at #MyQuery.Time[2]#<br>
</cfoutput><br>
<br>
<h4>The query metadata</h4>
<cfset querymetadata=getMetaData(myQuery)>
<cfdump var="#querymetadata#">
```

QuerySetCell

説明

セルに値を設定します。行番号を指定しない場合は、最後の行のセルに値が設定されます。

以前のバージョンの ColdFusion では可能でしたが、ColdFusion MX 7 以降のリリースでは、数値型の列に文字列リテラル ("All" など) を追加できなくなりました。

戻り値

正しく設定できた場合は true、設定できなかった場合は false

カテゴリ

クエリー関数

関数のシンタックス

```
QuerySetCell(query, column_name, value [, row_number ])
```

関連項目

QueryAddColumn、QueryAddRow、QueryNew、『ColdFusion アプリケーションの開発』の Creating a recordset with the QueryNew() function

履歴

ColdFusion MX 7: 型の検証を行うように関数の動作が変更されました。

パラメータ

パラメータ	説明
query	実行するクエリーの名前です。
column_name	クエリー内の列の名前です。
value	セルに設定する値です。
row_number	行番号です。デフォルト値は、最後の行です。

例

```
<!--- This example shows the use of QueryAddRow and QuerySetCell --->

<!--- start by making a query --->
<cfquery name = "GetCourses" datasource = "cfdocexamples">
    SELECT Course_ID, Descript
    FROM Courses
</cfquery>
<p>The Query "GetCourses" has <cfoutput>#GetCourses.RecordCount#</cfoutput> rows.

<cfset CountVar = 0>
<cfloop CONDITION = "CountVar LT 15">
    <cfset temp = QueryAddRow(GetCourses)>
    <cfset CountVar = CountVar + 1>
    <cfset Temp = QuerySetCell(GetCourses, "Number", 100*CountVar)>
    <cfset CountVar = CountVar + 1>
    <cfset Temp = QuerySetCell(GetCourses, "Descript",
    "Description of variable #Countvar#")>
</cfloop>

<P>After the QueryAddRow action, the query has
    <CFOUTPUT>#GetCourses.RecordCount#</CFOUTPUT>
    records.
    <CFOUTPUT query="GetCourses">
    <PRE>#Course_ID# #Course_Number# #Descript#</pre> </cfoutput>
```

QuotedValueList

説明

実行したクエリーから返された各レコードの値を取得します。引数の評価は行われません。

戻り値

実行したクエリーから返された各レコードの値を、区切り文字で区切ったリストそれぞれの値は一重引用符で囲まれます。

カテゴリ

クエリー関数、リスト関数

関数のシンタックス

```
QuotedValueList(query.column [, delimiter ])
```

関連項目

ValueList

パラメータ

パラメータ	説明
query.column	実行するクエリーと列の名前です。クエリー名と列名の間はピリオド (.) で区切ります。
delimiter	文字列、または文字列を含んでいる変数です。列データの区切り文字を指定します。

例

```
<!--- use the contents of one query to create another dynamically --->
<cfset List = "'BIOL', 'CHEM'">
<!--- first, get the department IDs in our list --->
<cfquery name = "GetDepartments" datasource = "cfdocexamples">
    SELECT Dept_ID FROM Departments
    WHERE Dept_ID IN (#PreserveSingleQuotes(List)#)
</cfquery>

<!--- now, select the courses for that department based on the
quotedValueList produced from our previous query --->
<cfquery name = "GetCourseList" datasource = "cfdocexamples">
    SELECT *
    FROM CourseList
    WHERE Dept_ID IN ('#GetDepartments.Dept_ID#')
</cfquery>

<!--- now, output the results --->

List the course numbers that are in BIOL and CHEM (uses semicolon (;) as the delimiter):<br>
<cfoutput>
#QuotedValueList(GetCourseList.CorNumber,";")#<br>
</cfoutput>
```

Rand

説明

擬似乱数を生成します。

戻り値

擬似乱数 (0 ～ 1 の範囲の小数)

カテゴリ

算術関数、セキュリティ関数

関数のシンタックス

```
Rand([algorithm])
```

履歴

ColdFusion MX 7: **algorithm** パラメータが追加されました。

関連項目

Randomize、RandRange

パラメータ

パラメータ	説明
algorithm	(オプション) 乱数を生成するために使用するアルゴリズムです。ColdFusion では、次のアルゴリズムを使用する暗号ライブラリがインストールされます。 • CFMX_COMPAT: (デフォルト) ColdFusion で使用されるアルゴリズムです。 • SHA1PRNG: Sun Java SHA1PRNG アルゴリズムを使用して数値を生成します。このアルゴリズムでは、デフォルトのアルゴリズムよりもランダム性を高めることができます。 • IBMSecureRandom: IBM WebSphere 用のアルゴリズムです。IBM JVM は SHA1PRNG アルゴリズムをサポートしません。

使用方法

乱数ジェネレータのシードを設定するには、この関数を呼び出す前に Randomize 関数を呼び出します。乱数ジェネレータのシードを設定すると、Rand 関数は常に擬似乱数の同じシーケンスを生成します。この動作は、一貫したパターンを再現する必要がある場合に役立ちます。

ColdFusion MX では、JCE (Java Cryptography Extension) を使用し、Sun JCE デフォルトセキュリティプロバイダを含む Sun Java 1.4.2 ランタイムがインストールされます。このプロバイダには、前の「パラメータ」に示したアルゴリズム (デフォルトのアルゴリズムを除く) が含まれています。JCE フレームワークには、他のプロバイダを実装するための機能も含まれています。ただし、当社ではサードパーティのセキュリティプロバイダに対するテクニカルサポートは提供していません。

例

次の例では、SHA1PRNG アルゴリズムを使用して、単一の乱数を生成します。

```
<h3>Rand Example</h3>
<cfoutput>
    <p>Rand("SHA1PRNG") returned: #Rand("SHA1PRNG")#</p>
    <p><A HREF = "#CGI.SCRIPT_NAME#">Try again</A>
</cfoutput>
```

Randomize

説明

乱数ジェネレータで使用するシードを整数値で設定し、一貫した数値パターンが繰り返されるようにします。

戻り値

擬似乱数 (0 ～ 1 の範囲の小数)

カテゴリ

算術関数、セキュリティ関数

関数のシンタックス

```
Randomize(number[, algorithm])
```

履歴

ColdFusion MX 7: **algorithm** パラメータが追加されました。

関連項目

Rand、RandRange

パラメータ

パラメータ	説明
number	整数です。数値が -2,147,483,648 ～ 2,147,483,647 の範囲外である場合、ColdFusion ではエラーが生成されます。
algorithm	(オプション) シード値を生成するために使用するアルゴリズムです。ColdFusion では、次のアルゴリズムを使用する暗号ライブラリがインストールされます。 • CFMX_COMPAT: (デフォルト) ColdFusion で使用されるアルゴリズムです。 • SHA1PRNG: Sun Java SHA1PRNG アルゴリズムを使用して数値を生成します。このアルゴリズムでは、デフォルトのアルゴリズムよりもランダム性を高めることができます。 • IBMSecureRandom: IBM WebSphere 用のアルゴリズムです。IBM JVM は SHA1PRNG アルゴリズムをサポートしません。

使用方法

乱数ジェネレータのシードを設定するには、Rand を呼び出す前に、この関数を呼び出します。乱数ジェネレータのシードを設定すると、Rand 関数は常に擬似乱数の同じシーケンスを生成します。この動作は、一貫したパターンを再現する必要がある場合に役立ちます。

標準版では、デフォルトのアルゴリズム以外のすべてのアルゴリズムに対して JCE (Java Cryptography Extension) が使用され、Sun JCE のデフォルトセキュリティプロバイダを含む Sun Java ランタイムがインストールされます。このプロバイダには、前の「パラメータ」に示したアルゴリズムが含まれています。JCE フレームワークには、他のプロバイダの実装を使用するための機能も含まれています。ただし、Adobe ではサードパーティのセキュリティプロバイダに対するテクニカルサポートは提供していません。

エンタープライズ版の ColdFusion では、RSA BSafe Crypto-J ライブラリもインストールされます。このプロバイダでは、FIPS186PRNG、MD5PRNG、DummyPRNG、OBFPRNG の各アルゴリズムが追加されます。DummyPRNG は常に 0 を返します。

例

次の例では、Randomize 関数を呼び出して乱数ジェネレータのシードを設定し、10 個の乱数を生成します。シードの効果を確認するには、同じ値のフォームを複数回送信します。

```
<h3>Randomize Example</h3>

    <!--- Do the following only if the form has been submitted. --->
<cfif IsDefined("Form.myRandomInt")>

    <!--- Make sure submitted value is a number and display its value. --->
    <cfif IsNumeric(FORM.myRandomInt)>
        <cfoutput>
            <b>Seed value is #FORM.myRandomInt#</b><br>
        </cfoutput><br>

        <!--- Call Randomize to seed the random number generator. --->
        <cfset r = Randomize(FORM.myRandomInt, "SHA1PRNG")>

        <cfoutput>
            <b>Random number returned by Randomize(#Form.myRandomInt#,
                    "SHA1PRNG"):</b><br>
            #r#<br>
            <br>
            <b>10 random numbers generated using the SHA1PRNG algorithm:</b><br>
            <cfloop index = "i" from = "1" to = "10" step = "1">
                #Rand("SHA1PRNG")#<br>
            </cfloop><br>
        </cfoutput>

    <cfelse>
        <p>Please enter a number.
    </cfif>
</cfif>

<!--- Form to specify the seed value. --->
<form action="#CGI.SCRIPT_NAME#" method="post">
<p>Enter a number to seed the randomizer:
<input type = "Text" name = "MyRandomInt" value="12345">
<p><input type = "Submit" name = "">
</form>
```

RandRange

説明
指定された 2 つの数値間の範囲で擬似乱数 (整数) を生成します。

戻り値
擬似乱数 (整数)

カテゴリ
算術関数、セキュリティ関数

関数のシンタックス
```
RandRange(number1, number2[, algorithm])
```

履歴
ColdFusion MX 7: **algorithm** パラメータが追加されました。

関連項目

Rand、Randomize

パラメータ

パラメータ	説明
number1、number2	整数です。数値が -2,147,483,648 ～ 2,147,483,647 の範囲外である場合、ColdFusion ではエラーが生成されます。
algorithm	(オプション) 乱数を生成するために使用するアルゴリズムです。ColdFusion では、次のアルゴリズムを使用する暗号ライブラリがインストールされます。 • CFMX_COMPAT: (デフォルト) ColdFusion で使用されるアルゴリズムです。 • SHA1PRNG: Sun Java SHA1PRNG アルゴリズムを使用して数値を生成します。このアルゴリズムでは、デフォルトのアルゴリズムよりもランダム性を高めることができます。 • IBMSecureRandom: IBM WebSphere 用のアルゴリズムです。IBM JVM は SHA1PRNG アルゴリズムをサポートしません。

使用方法

number1 パラメータおよび **number2** パラメータが正または負の非常に大きな値である場合、得られる結果のランダム性が低下することがあります。この問題を回避するには、-1,000,000,000 ～ 1,000,000,000 の範囲外の数値を指定しないようにします。

ColdFusion では、JCE (Java Cryptography Extension) を使用し、Sun JCE デフォルトセキュリティプロバイダを含む Sun Java 1.4.2 ランタイムがインストールされます。このプロバイダには、前の「パラメータ」に示したアルゴリズム (デフォルトのアルゴリズムを除く) が含まれています。JCE フレームワークには、他のプロバイダを実装するための機能も含まれています。ただし、当社ではサードパーティのセキュリティプロバイダに対するテクニカルサポートは提供していません。

例

次の例では、乱数の値の範囲を指定し、必要に応じて乱数のシード値を指定するフォームを使用します。cfform のコントロールおよび属性を使用して、デフォルトの範囲が指定され、範囲フィールドに値が設定され、そのフィールド値が指定の範囲内にあることが検証されます。フォームを送信すると、シードフィールドに空の文字列がないかどうかが確認されます。フィールドに値がある場合、コードはその値を使用して乱数ジェネレータのシードを設定します。次に、乱数が生成され、表示されます。

```
<h3>RandRange Example</h3>

<!--- Do the following only if the form has been submitted. --->
<cfif IsDefined("Form.mySeed")>

    <!--- Do the following only if the seed field has a non-empty string. --->
    <cfif Form.mySeed NEQ "">
        <cfoutput>
            <b>Seed value is #FORM.mySeed#</b><br>
        </cfoutput>
        <br>

        <!--- Call Randomize to seed the random number generator. --->
        <cfset r = Randomize(FORM.mySeed, "SHA1PRNG")>
    <cfelse>
        <b>No Seed value submitted</b><br>
    </cfif>

    <!--- Generate and display the random number. --->
    <cfoutput><p><b>
        RandRange returned: #RandRange(FORM.myInt, FORM.myInt2, "SHA1PRNG")#
```

```
    </cfoutput></b></p>
</cfif>

<!--- This form uses cfform input validation to check the input range. --->
<cfform action = "#CGI.SCRIPT_NAME#">
<p>Enter the random number Range: From
<cfinput type = "Text" name = "MyInt" value = "1"
    RANGE = "-1000000000,1000000000"
    message = "Please enter a value between -1,000,000,000 and 1,000,000,000"
    validate = "integer" required = "Yes">
To
<cfinput type = "Text" name = "MyInt2" value = "9999"
    RANGE = "-1000000000,1000000000"
    message = "Please enter a value between --1,000,000,000and 1,000,000,000"
    validate = "integer" required = "Yes"></p>
<p>Enter a number to seed the randomizer:
<cfinput type = "Text" name = "mySeed" RANGE = "-1000000000,1000000000"
    message = "Please enter a value between -1,000,000,000 and 1,000,000,000"
    validate = "integer" required = "No"></p>
<p><input type = "Submit" name = "">
</cfform>
```

ReEscape

説明
文字列を取得し、正規表現の制御文字に一致する文字をエスケープします。

戻り値
末尾にエスケープ文字が追加された文字列

シンタックス
reEscape(***string***)

プロパティ

パラメータ	説明
string	エスケープ対象の正規表現文字に一致する文字が含まれる文字列です。

例

```
<cfoutput>
#reescape("*.{}[]exam?ple")#
</cfoutput>
```

REFind

説明
正規表現 (RE) を使用して、パターンに一致する文字列を検索します。この検索では大文字と小文字が区別されます。

正規表現や、そのエスケープシーケンス、アンカー、および修飾子の詳細については、『ColdFusion アプリケーションの開発』の Using Regular Expressions in Functions を参照してください。

戻り値
returnsubexpressions パラメータの値により異なります。

- returnsubexpressions= "False" の場合 :
  - 文字列中で一致した文字列の先頭位置が返されます。
  - 指定した正規表現に一致する部分が文字列中にない場合は 0 が返されます。
- returnsubexpressions = "True" の場合 : len および pos という 2 つの配列を持つ構造体が返されます。配列の要素は次のとおりです。
  - 指定した正規表現に一致する部分が文字列中にあった場合、len 配列と pos 配列の先頭の要素には、正規表現全体に一致した最初の部分文字列の長さと位置がそれぞれ格納されます。
  - 正規表現の中に括弧でグループ化されている部分がある場合、後続の各配列要素には、各グループに最初に一致した部分文字列の長さと位置がそれぞれ格納されます。
  - 指定した正規表現に一致する部分が見つからなかった場合、len 配列と pos 配列の先頭の要素には 0 が格納されます。

カテゴリ
文字列関数

関数のシンタックス
```
REFind(reg_expression, string [, start, returnsubexpressions ] )
```

関連項目
Find、FindNoCase、REFindNoCase、REReplace、REReplaceNoCase

パラメータ

パラメータ	説明
reg_expression	検索に使用する正規表現です。大文字と小文字は区別されます。
string	検索対象の文字列、またはそれを含んでいる変数です。
start	オプション。正の整数、または正の整数を含んでいる変数です。文字列中で検索を開始する位置を指定します。デフォルト値は 1 です。
returnsubexpressions	オプション。ブール値です。reg_expression に一致した部分文字列を、配列 len および pos として返すかどうかを指定します。 • true: 正規表現に一致する部分が見つかった場合、その長さと位置が、両配列の先頭要素にそれぞれ格納されます。正規表現の中に括弧でグループ化されている部分がある場合、後続の各配列要素には、各グループに最初に一致した部分文字列の長さと位置がそれぞれ格納されます。正規表現に一致する部分が見つからなかった場合、両配列の要素数は 1 となり、要素の値として 0 が格納されます。 • false: 文字列内で正規表現に一致した部分の先頭位置が返されます (デフォルト)。

使用方法
この関数は、文字列内で最初に出現する正規表現を検索します。正規表現、またはその中の部分文字列が 2 回目以降に出現する箇所を検索するには、この関数を複数回、そのつど開始位置を変えて呼び出します。次回の開始位置を決定するには、returnsubexpressions パラメータを使用し、len 配列と pos 配列のそれぞれの先頭要素に返された値の和を求めます。

例

```
 <h3>REFind Example</h3>
<p>This example shows the use of the REFind function with and without the
    <i>returnsubexpressions</i> parameter set to True.
    If you do not use the <i>returnsubexpressions</i> parameter,
    REFind returns the position of the first occurrence of a regular
    expression in a string starting from the specified position.
    Returns 0 if no occurrences are found.</p>

<p>REFind("a+c+", "abcaaccdd"):
<cfoutput>#REFind("a+c+", "abcaaccdd")#</cfoutput></p>
<p>REFind("a+c*", "abcaaccdd"):
<cfoutput>#REFind("a+c*", "abcaaccdd")#</cfoutput></p>
<p>REFind("[[:upper:]]", "abcaacCDD"):
<cfoutput>#REFind("[[:upper:]]", "abcaacCDD")#</cfoutput></p>
<p>REFind("[\?&]rep = ", "report.cfm?rep = 1234&u = 5"):
    <cfoutput>#REFind("[\?&]rep = ", "report.cfm?rep = 1234&u = 5")#
    </cfoutput>
</p>
<!--- Set startPos to one; returnMatchedSubexpressions = TRUE --->
<hr size = "2" color = "#0000A0">
<p>If you use the <i>returnssubexpression</i> parameter, REFind returns the
    position and length of the first occurrence of a regular expression
    in a string starting from the specified position. The position and
    length variables are stored in a structure. To access position and length
    information, use the keys <i>pos</i> and <i>len</i>, respectively.</p>
<cfset teststring = "The cat in the hat hat came back!">
<p>The string in which the function is to search is:
<cfoutput><b>#teststring#</b></cfoutput>.</p>
<p>The first call to REFind to search this string is:
        <b>REFind("[A-Za-z]+",testString,1,"TRUE")</b></p>
<p>This function returns a structure that contains two arrays: pos and len.</p>
<p>To create this structure you can use a CFSET statement, for example: </p>
&lt;CFSET st = REFind("[[:alpha:]]",testString,1,"TRUE")&gt;
<cfset st = REFind("[[:alpha:]]",testString,1,"TRUE")>
<p>
    <cfoutput>
    The number of elements in each array: #ArrayLen(st.pos)#.
    </cfoutput></p>
<p><b>The number of elements in the pos and len arrays is always one
    if you do not use parentheses in the regular expression.</b></p>
<p>The value of st.pos[1] is: <cfoutput>#st.pos[1]#.</cfoutput></p>
<p>The value of st.len[1] is: <cfoutput>#st.len[1]#.</cfoutput></p>
<p>
<cfoutput>
    Substring is <b>[#Mid(testString,st.pos[1],st.len[1])#]</B>
    </cfoutput></p>
<hr size = "2" color = "#0000A0">
<p>However, if you use parentheses in the regular expression, the first
    element contains the position and length of the first instance
```

```
    of the whole expression. The position and length of the first instance
    of each parenthesized subexpression within is included in additional
    array elements.</p>
<p>For example:
&lt;CFSET st1 = REFind("([[:alpha:]])[ ]+(\1)",testString,1,"TRUE")&gt;</p>
<cfset st1 = REFind("([[:alpha:]]+)[ ]+(\1)",testString,1,"TRUE")>
<p>The number of elements in each array is <cfoutput>#ArrayLen(st1.pos)#
    </cfoutput>.</p>
<p>First whole expression match; position is
    <cfoutput>#st1.pos[1]#;
        length is #st1.len[1]#; whole expression match is
        <B>[#Mid(testString,st1.pos[1],st1.len[1])#]</B>
    </cfoutput></p>
<p>Subsequent elements of the arrays provide the position and length of
    the first instance of each parenthesized subexpression therein.</p>
    <cfloop index = "i" from = "2" to = "#ArrayLen(st1.pos)#">
    <p><cfoutput>Position is #st1.pos[i]#; Length is #st1.len[i]#;
    Substring is <B>[#Mid(testString,st1.pos[i],st1.len[i])#]
    </B></cfoutput></p>
</cfloop><br>
```

REFindNoCase

説明

正規表現 (RE) を使用して、パターンに一致する文字列を指定位置以降から検索します。この検索では大文字と小文字は区別されません。

正規表現や、そのエスケープシーケンス、アンカー、および修飾子の詳細については、『ColdFusion アプリケーションの開発』の Using Regular Expressions in Functions を参照してください。

戻り値

returnsubexpressions パラメータの値により異なります。

- returnsubexpressions= "False" の場合 :
  - 文字列中で一致した文字列の先頭位置が返されます。
  - 指定した正規表現に一致する部分が文字列中にない場合は 0 が返されます。
- returnsubexpressions = "True" の場合 : len および pos という 2 つの配列を持つ構造体が返されます。配列の要素は次のとおりです。
  - 指定した正規表現に一致する部分が文字列中にあった場合、len 配列と pos 配列の先頭の要素には、正規表現全体に一致した最初の部分文字列の長さと位置がそれぞれ格納されます。
  - 正規表現の中に括弧でグループ化されている部分がある場合、後続の各配列要素には、各グループに最初に一致した部分文字列の長さと位置がそれぞれ格納されます。
  - 指定した正規表現に一致する部分が見つからなかった場合、len 配列と pos 配列の先頭の要素には 0 が格納されます。

カテゴリ

文字列関数

関数のシンタックス

```
REFindNoCase(reg_expression, string [, start, returnsubexpressions])
```

関連項目

Find、FindNoCase、REFind、REReplace、REReplaceNoCase

パラメータ

パラメータ	説明
reg_expression	検索に使用する正規表現です。大文字と小文字は区別されません。 詳細については、『ColdFusion アプリケーションの開発』の Using Regular Expressions in Functions を参照してください。
string	文字列、または文字列を含んでいる変数です。検索対象の文字列を指定します。
start	オプション。正の整数、または正の整数を含んでいる変数です。検索を開始する位置を指定します。デフォルト値は 1 です。
returnsubexpressions	オプション。ブール値です。reg_expression に一致した部分文字列を、配列 len および pos として返すかどうかを指定します。 • true: 正規表現に一致する部分が見つかった場合、その長さと位置が、両配列の先頭要素にそれぞれ格納されます。正規表現の中に括弧でグループ化されている部分がある場合、後続の各配列要素には、各グループに最初に一致した部分文字列の長さと位置がそれぞれ格納されます。正規表現に一致する部分が見つからなかった場合、両配列の要素数は 1 となり、要素の値として 0 が格納されます。 • false: 文字列内で正規表現に一致した部分の先頭位置が返されます (デフォルト)。

使用方法

この関数は、文字列内で最初に出現する正規表現を検索します。正規表現、またはその中の部分文字列が 2 回目以降に出現する箇所を検索するには、この関数を複数回、そのつど開始位置を変えて呼び出します。次回の開始位置を決定するには、returnsubexpressions パラメータを使用し、len 配列と pos 配列のそれぞれの先頭要素に返された値の和を求めます。

例

```
<h3>REFindNoCase Example</h3>
<p>This example demonstrates the use of the REFindNoCase function with and
    without the <i>returnsubexpressions</i> parameter set to True.</p>
<p>If you do not use the <i>returnsubexpressions</i> parameter, REFindNoCase
    returns the position of the first occurrence of a regular expression
    in a string starting from the specified position. Returns 0 if no
    occurrences are found. </p>
<p>REFindNoCase("a+c+", "abcaaccdd"):
<cfoutput>#REFindNoCase("a+c+", "abcaaccdd")#</cfoutput></p>
<p>REFindNoCase("a+c*", "abcaaccdd"):
<cfoutput>#REFindNoCase("a+c*", "abcaaccdd")#</cfoutput></p>
<p>REFindNoCase("[[:alpha:]]+", "abcaacCDD"):
<cfoutput>#REFindNoCase("[[:alpha:]]+", "abcaacCDD")#</cfoutput></p>
<p>REFindNoCase("[\?&]rep = ", "report.cfm?rep = 1234&u = 5"):
<cfoutput>#REFindNoCase("[\?&]rep = ", "report.cfm?rep = 1234&u = 5")#
    </cfoutput></p>
<!--- Set startPos to one; returnMatchedSubexpressions = True --->
<hr size = "2" color = "#0000A0">
<p>If you do use the <i>returnssubexpression</i> parameter, REFindNoCase returns
    the position and length of the first occurrence of a regular expression
    in a string starting from the specified position. The position and length
    variables are stored in a structure. To access position and length
    information, use the keys <i>pos</i> and <i>len</i>, respectively.</p>

<cfset teststring = "The cat in the hat hat came back!">
<p>The string in which the function is to search is:
    <cfoutput><b>#teststring#</b></cfoutput>.</p>
<p>The first call to REFindNoCase to search this string is:
```

```
<b>REFindNoCase("[[:alpha:]]+",testString,1,"True")</b></p>
<p>This function returns a structure that contains two arrays: pos and len.</p>
<p>To create this structure you can use a CFSET statement,
    for example:</p>
&lt;CFSET st = REFindNoCase("[[:alpha:]]+",testString,1,"True")&gt;
<cfset st = REFindNoCase("[[:alpha:]]+",testString,1,"True")>
<p>
    <cfoutput>
    The number of elements in each array: #ArrayLen(st.pos)#.
    </cfoutput></p>
<p><b>The number of elements in the pos and len arrays will always be one,
    if you do not use parentheses to denote subexpressions in the regular
    expression.</b></p>
<p>The value of st.pos[1] is: <cfoutput>#st.pos[1]#.</cfoutput></p>
<p>The value of st.len[1] is: <cfoutput>#st.len[1]#.</cfoutput></p>
<p>
    <cfoutput>
    Substring is <b>[#Mid(testString,st.pos[1],st.len[1])#]</B>
    </cfoutput></p>
<hr size = "2" color = "#0000A0">
<p>However, if you use parentheses to denote subexpressions in the regular
    expression, the first element contains the position and length of
    the first instance of the whole expression. The position and length
    of the first instance of each subexpression within will be included
    in additional array elements.</p>
<p>For example:
&lt;CFSET st1 = REFindNoCase("([[:alpha:]]+)[ ]+(\1)",testString,1,"True")&gt;</p>
<cfset st1 = REFindNoCase("([[:alpha:]]+)[ ]+(\1)",testString,1,"True")>

<p>The number of elements in each array is
<cfoutput>
    #ArrayLen(st1.pos)#
</cfoutput>.</p>

<p>First whole expression match; position is
<cfoutput>
    #st1.pos[1]#; length is #st1.len[1]#;
    whole expression match is <B>[#Mid(testString,st1.pos[1],st1.len[1])#]</B>
</cfoutput></p>

<p>Subsequent elements of the arrays provide the position and length of the
    first instance of each parenthesized subexpression therein.</p>
    <cfloop index = "i" from = "2" to = "#ArrayLen(st1.pos)#">
    <p><cfoutput>Position is #st1.pos[i]#; Length is #st1.len[i]#;
    Substring is <B>[#Mid(testString,st1.pos[i],st1.len[i])#]</B>
    </cfoutput></p>
</cfloop><br>
```

REMatch

説明

正規表現 (RE) を使用して、パターンに一致する文字列を指定位置以降から検索します。この検索では大文字と小文字が区別されます。

正規表現や、そのエスケープシーケンス、アンカー、および修飾子の詳細については、『ColdFusion アプリケーションの開発』の Using Regular Expressions in Functions を参照してください。

戻り値

式に一致する文字列の配列

カテゴリ
文字列関数

関数のシンタックス
```
REMatch(reg_expression, string)
```

関連項目
Find、FindNoCase、REFind、REReplace、REReplaceNoCase、REMatchNoCase

パラメータ

パラメータ	説明
reg_expression	検索に使用する正規表現です。大文字と小文字は区別されます。 詳細については、『ColdFusion アプリケーションの開発』の Using Regular Expressions in Functions を参照してください。
string	文字列、または文字列を含んでいる変数です。検索対象の文字列を指定します。

使用方法
この関数は、文字列内で正規表現に一致する箇所をすべて検索します。

例
```
<!--- Find all the URLs in a web page retrieved via cfhttp:. --->
<!--- The search is case sensitive. --->
result = REMatch("https?://([-\w\.]+)+(:\d+)?(/([\w/_\.]*(\?\S+)?)?)?", cfhttp.filecontent);
```

REMatchNoCase

説明
正規表現 (RE) を使用して、パターンに一致する文字列を指定位置以降から検索します。この検索では大文字と小文字は区別されません。

正規表現や、そのエスケープシーケンス、アンカー、および修飾子の詳細については、『ColdFusion アプリケーションの開発』の Using Regular Expressions in Functions を参照してください。

戻り値
式に一致する文字列の配列

カテゴリ
文字列関数

関数のシンタックス
```
REMatchNoCase(reg_expression, string)
```

関連項目
Find、FindNoCase、REFind、REReplace、REReplaceNoCase、REMatch

パラメータ

パラメータ	説明
reg_expression	検索に使用する正規表現です。大文字と小文字は区別されません。 詳細については、『ColdFusion アプリケーションの開発』の Using Regular Expressions in Functions を参照してください。
string	文字列、または文字列を含んでいる変数です。検索対象の文字列を指定します。

例

```
<!--- Find all the URLs in a web page retrieved via cfhttp:. --->
result = REMatch("https?://([-\w\.]+)+(:\d+)?(/([\w/_\.]*(\?\S+)?)?)?", cfhttp.filecontent);
```

ReleaseComObject

説明

COM オブジェクトをリリースし、使用していたリソースを解放します。

戻り値

なし

カテゴリ

拡張関数

関数のシンタックス

```
ReleaseComObject(objectName)
```

関連項目

CreateObject、cfobject

履歴

ColdFusion MX 6.1: この関数が追加されました。

パラメータ

パラメータ	説明
objectName	CreateObject 関数または cfobject タグを使って作成した COM オブジェクトの変数名です。

使用方法

この関数は、指定した COM オブジェクトと、それが作成したすべての COM オブジェクトを、強制的に終了およびリリースします。使用が済んだオブジェクトに対してこの関数を使えば、リソースをすばやく解放できます。プログラムを終了する quit のようなメソッドがある COM オブジェクトの場合は、そのメソッドを呼び出してから ReleaseComObject 関数を呼び出します。

この関数を使うことで処理効率が改善される可能性がありますが、アプリケーションを動作させるために必須のものではありません。この関数を使用しなくても、何らかの時点で Java のガベージコレクション機構が作動してリソースが解放されます。使用中のオブジェクトに対してこの関数を呼び出すと、必要なオブジェクトがリリースされて使用不可能となり、アプリケーションに例外が送信されます。

例

```
<h3>ReleaseComObject Example</h3>
    <cfscript>
    obj = CreateObject("Com", "excel.application.9");
    //code that uses the object goes here???I'd like to fill this in with something???
    obj.quit();
    ReleaseComObject(obj);
    </cfscript>
```

RemoveCachedQuery

説明

指定した詳細を持つクエリをクエリキャッシュから削除します。

戻り値

なし

シンタックス

removeCachedQuery(SQL*, datasource, [params], [region*])

履歴

ColdFusion 10: この関数が追加されました。

プロパティ

パラメータ	説明
SQL	クエリの SQL です。
datasource	クエリを実行したときのデータソースです。
params	（オプション）SQL に渡されるパラメーター値の配列です。
region	（オプション）キャッシュオブジェクトを配置可能なキャッシュ領域を指定します。

例

```
<cfset sql = "SELECT * from art where artid = ?">
<cfquery name="q" datasource="cfartgallery" cachedwithin="#CreateTimeSpan(0, 6, 0, 0)#">
  SELECT * from art where artid = <cfqueryPARAM value = "1" CFSQLType = 'CF_SQL_INTEGER'>
</cfquery>
<cfset a = arrayNew(1)>
<cfset a[1] = 1>
<cfset removeCachedQuery(sql,"cfartgallery", a)>
```

RemoveChars

説明

文字列から文字を削除します。

戻り値

指定した開始位置から **count** 個の文字を削除した文字列のコピー。文字が見つからなかった場合は 0 が返されます。

カテゴリ
文字列関数

関数のシンタックス
```
RemoveChars(string, start, count)
```

関連項目
Insert、Len

パラメータ

パラメータ	説明
string	文字列、または文字列を含んでいる変数です。検索対象の文字列を指定します。
start	正の整数、または正の整数を含んでいる変数です。検索を開始する位置を指定します。
count	削除する文字の数を指定します。

例
```
<h3>RemoveChars Example</h3>
Returns a string with <I>count</I> characters removed from the
    start position. Returns 0 if no characters are found.

<cfif IsDefined("FORM.myString")>
    <cfif (FORM.numChars + FORM.start) GT Len(FORM.myString)>
        <p>Your string is only <cfoutput>#Len(FORM.myString)#
         </cfoutput> characters long.
        Please enter a longer string, select fewer characters to remove or
        begin earlier in the string.
    <cfelse>
        <cfoutput>
        <p>Your original string: #FORM.myString#
        <p>Your modified string:#RemoveChars(FORM.myString,
    FORM.start, FORM.numChars)#
        </cfoutput>
    </cfif>
</cfif>
```

RepeatString

説明
指定の文字列を指定回数だけ繰り返して文字列を作成します。

戻り値
文字列です。

カテゴリ
文字列関数

関数のシンタックス
```
RepeatString(string, count)
```

関連項目
CJustify、LJustify、RJustify

パラメータ

パラメータ	説明
string	文字列、または文字列を含んでいる変数です。
count	繰り返す回数です。

例
```
<h3>RepeatString Example</h3>
<p>RepeatString returns a string created from <I>string</I>, repeated
    a specified number of times.
<ul>
    <li>RepeatString("-", 10): <cfoutput>#RepeatString("-", 10)#</cfoutput>
    <li>RepeatString("&lt;BR&gt;", 3): <cfoutput>#RepeatString("<br>", 3)#
    </cfoutput>
    <li>RepeatString("", 5): <cfoutput>#RepeatString("", 5)#</cfoutput>
    <li>RepeatString("abc", 0): <cfoutput>#RepeatString("abc", 0)#</cfoutput>
    <li>RepeatString("Lorem Ipsum", 2):
     <cfoutput>#RepeatString("Lorem Ipsum", 2)#</cfoutput>
</ul>
```

Replace

説明
文字列内の指定した範囲にある **substring1** を **substring2** に置き換えます。この検索では大文字と小文字が区別されます。

戻り値
置換後の文字列

カテゴリ
文字列関数

関数のシンタックス
```
Replace(string, substring1, substring2 [, scope ])
```

関連項目
Find、REFind、ReplaceNoCase、ReplaceList、REReplace

パラメータ

パラメータ	説明
string	文字列、または文字列を含んでいる変数です。検索対象の文字列を指定します。
substring1	文字列、または文字列を含んでいる変数です。この文字列が出現する箇所を検索します。
substring2	substring1 を置き換える文字列です。
scope	• one: 最初に一致した箇所を置き換えます (デフォルト)。 • all: 一致したすべての箇所を置き換えます。

使用方法
一致する箇所を削除するには、**substring2** に空の文字列 ("") を指定します。

文字列内のカンマ文字をエスケープする必要はありません。たとえば、次の例は文中からカンマを削除するコードです。

```
replace("The quick brown fox jumped over the lazy cow, dog, and cat.",",","","All")
```

例
```
<h3>Replace Example</h3>

<p>The Replace function returns <I>string</I> with <I>substring1</I>
    replaced by <I>substring2</I> in the specified scope. This
    is a case-sensitive search.

<cfif IsDefined("FORM.MyString")>
<p>Your original string, <cfoutput>#FORM.MyString#</cfoutput>
<p>You wanted to replace the substring <cfoutput>#FORM.MySubstring1#
    </cfoutput>
    with the substring <cfoutput>#FORM.MySubstring2#</cfoutput>.
<p>The result: <cfoutput>#Replace(FORM.myString,
    FORM.MySubstring1, FORM.mySubString2)#</cfoutput>
</cfif>
```

ReplaceList

説明
文字列内で複数種類の要素を検索し、対応する別の要素に置き換えます。検索対象の要素と、それらに対応する置き換え要素は、それぞれカンマ区切りリストで指定します。この検索では大文字と小文字が区別されます。

戻り値
置換後の文字列のコピー

カテゴリ
リスト関数、文字列関数

関数のシンタックス
```
ReplaceList(string, list1, list2)
ReplaceList(string, list1, list2, delimiter)
ReplaceList(string, list1, list2, delimiter_list1, delimiter_list2)
```

関連項目
Find、REFind、Replace、REReplace

パラメータ

パラメータ	説明
string	置き換えを実行する対象の部分文字列、またはそのような部分文字列を含んでいる変数です。
list1	検索する部分文字列を指定するカンマ区切りリストです。
list2	部分文字列を置き換える文字列を指定するカンマ区切りリストです。

パラメータ	説明
delimiter	検索と置き換えの両方に使用する共通の区切り文字です。
delimiter_list1	検索用の区切り文字です。
delimiter_list2	置き換え用の区切り文字です。

使用方法

検索する部分文字列のリストは、先頭の要素から順番に処理されます。**list1** に含まれる要素のいずれかの内容が **list2** の要素内にも出現する場合、再帰的な置換処理が発生する可能性があります。そのような例を次に示します。

例 1

```
 <p>The ReplaceList function returns <I>string</I> with
<I>substringlist1</I> (e.g. "a,b") replaced by <I>substringlist2</I>
    (e.g. "c,d") in the specified scope.
<cfif IsDefined("FORM.MyString")>
<p>Your original string, <cfoutput>#FORM.MyString#</cfoutput>
<p>You wanted to replace the substring <cfoutput>#FORM.MySubstring1#
    </cfoutput>
with the substring <cfoutput>#FORM.MySubstring2#</cfoutput>.
<p>The result: <cfoutput>#Replacelist(FORM.myString,
FORM.MySubstring1, FORM.mySubString2)#</cfoutput>
</cfif>
<form action = "replacelist.cfm" method="post">
<p>String 1
<br><input type = "Text" value = "My Test String" name = "MyString">
<p>Substring 1 (find this list of substrings)
<br><input type = "Text" value = "Test, String" name = "MySubstring1">
<p>Substring 2 (replace with this list of substrings)
<br><input type = "Text" value = "Replaced, Sentence" name = "MySubstring2">
<p><input type = "Submit" value = "Replace and display" name = "">
</form>

<h3>Replacelist Example Two</h3>
<cfset stringtoreplace = "The quick brown fox jumped over the lazy dog.">
<cfoutput>
    #ReplaceList(stringtoreplace,"dog,brown,fox,black", "cow,black,ferret,white")#
</cfoutput>
```

例 2

次の例では、区切り文字を両方のリストに適用します。

```
<h3>Replacelist Example One</h3>
<cfset stringtoreplace = "The quick brown fox jumped over the lazy dog.">
<cfoutput>
#ReplaceList(stringtoreplace,"dog:brown:fox:black", "cow:black:ferret:white", ":")#
</cfoutput>
```

例 3

次の例では、区切り文字は、それぞれのリストに固有のものです。

```
<h3>Replacelist Example Two</h3>
<cfset stringtoreplace = "The quick brown fox jumped over the lazy dog.">
<cfoutput>
#ReplaceList(stringtoreplace,"dog:brown:fox:black", "cow-black-ferret-white", ":" ,
"-")#
</cfoutput>
```

ReplaceNoCase

説明
指定した範囲にある **substring1** を **substring2** に置き換えます。この検索では大文字と小文字は区別されません。

戻り値
置換後の文字列のコピー

カテゴリ
文字列関数

関数のシンタックス
```
ReplaceNoCase(string, substring1, substring2 [, scope ])
```

関連項目
Find、REFind、Replace、ReplaceList、REReplace

パラメータ

パラメータ	説明
string	置き換えを実行する対象の部分文字列、またはそのような部分文字列を含んでいる変数です。
substring1	検索する文字列、またはその文字列を含んでいる変数です。これに一致する箇所が見つかると置き換えが実行されます。
substring2	substring1 に一致する箇所を置換する文字列、またはその文字列を含んでいる変数です。
scope	• one: 最初に一致した箇所を置き換えます (デフォルト)。 • all: 一致したすべての箇所を置き換えます。

例
```
<h3>ReplaceNoCase Example</h3>
<p>The ReplaceNoCase function returns <I>string</I> with <I>substring1</I>
    replaced by <I>substring2</I> in the specified scope.
    The search/replace is case-insensitive.

<cfif IsDefined("FORM.MyString")>
<p>Your original string, <cfoutput>#FORM.MyString#</cfoutput>
<p>You wanted to replace the substring <cfoutput>#FORM.MySubstring1#
    </cfoutput>
with the substring <cfoutput>#FORM.MySubstring2#</cfoutput>.
<p>The result: <cfoutput>#ReplaceNoCase(FORM.myString,
FORM.MySubstring1, FORM.mySubString2)#</cfoutput>
</cfif>
```

REReplace

説明
正規表現 (RE) を使用して、パターンに一致する文字列を検索し、別の文字列に置き換えます。この検索では大文字と小文字が区別されます。

戻り値

scope パラメータが one に設定されている場合は、正規表現に最初に一致した箇所を **substring** に置き換えた文字列が返されます。

scope パラメータが all に設定されている場合は、正規表現に一致したすべての箇所を **substring** に置き換えた文字列が返されます。

正規表現に一致する箇所が見つからなかった場合は、元の文字列のコピーがそのまま返されます。

カテゴリ

文字列関数

関数のシンタックス

```
REReplace(string, reg_expression, substring [, scope ])
```

関連項目

REFind、Replace、ReplaceList、REReplaceNoCase

履歴

ColdFusion MX: 大文字と小文字の変換を制御する機能を新たにサポートしました。置換する部分文字列の中で次の特殊コードを使用できます。

- ¥u - 直後の 1 文字を大文字にします。
- ¥l - 直後の 1 文字を小文字にします。
- ¥U - ¥E までの文字をすべて大文字にします。
- ¥L - ¥E までの文字をすべて小文字にします。
- ¥E - ¥U または ¥L の有効範囲を終了します。

新機能の詳細については、REFind を参照してください。

パラメータ

パラメータ	説明
string	文字列、または文字列を含んでいる変数です。この文字列内を検索します。
reg_expression	置換対象箇所を検索する正規表現です。この検索では大文字と小文字が区別されます。
substring	文字列、または文字列を含んでいる変数です。reg_expression に一致した箇所をこの文字列で置き換えます。
scope	• one: 最初に一致した箇所を置き換えます (デフォルト)。 • all: 一致したすべての箇所を置き換えます。

使用方法

正規表現の使用法の詳細については、『ColdFusion アプリケーションの開発』の Using Regular Expressions in Functions を参照してください。

例

```
<p>The REReplace function returns <i>string</i> with a regular expression replaced
    with <i>substring</i> in the specified scope. Case-sensitive search.
<p>REReplace("CABARET","C|B","G","ALL"):
<cfoutput>#REReplace("CABARET","C|B","G","ALL")#</cfoutput>
<p>REReplace("CABARET","[A-Z]","G","ALL"):
<cfoutput>#REReplace("CABARET","[A-Z]","G","ALL")#</cfoutput>
<p>REReplace("I love jellies","jell(y|ies)","cookies"):
<cfoutput>#REReplace("I love jellies","jell(y|ies)","cookies")#
    </cfoutput>
<p>REReplace("I love jelly","jell(y|ies)","cookies"):
<cfoutput>#REReplace("I love jelly","jell(y|ies)","cookies")#</cfoutput>
```

REReplaceNoCase

説明

正規表現を使用して、パターンに一致する文字列を検索し、別の文字列に置き換えます。この検索では大文字と小文字は区別されません。

戻り値

- scope = "one" の場合、正規表現に最初に一致した箇所を **substring** に置き換えた文字列
- scope = "all" の場合、正規表現に一致したすべての箇所を **substring** に置き換えた文字列
- 正規表現に一致する箇所が見つからなかった場合は、元の文字列のコピーがそのまま返されます。

カテゴリ

文字列関数

関数のシンタックス

```
REReplaceNoCase(string, reg_expression, substring [, scope ])
```

関連項目

REFind、REFindNoCase、Replace、ReplaceList

履歴

ColdFusion MX: 動作が変更されました。この関数では、大文字と小文字の変換を制御するための特殊文字 ¥u、¥U、¥l、¥L、¥E が、正規表現の置換文字列に挿入されます。ColdFusion 5 アプリケーションでこれらの文字列を使用している場合は、その前に円記号を挿入します ("¥u" を "¥¥u" に変更するなど)。

パラメータ

パラメータ	説明
string	文字列、または文字列を含んでいる変数です。
reg_expression	置換対象箇所を検索する正規表現です。詳細については、『ColdFusion アプリケーションの開発』の Using Regular Expressions in Functions を参照してください。
substring	文字列、または文字列を含んでいる変数です。reg_expression に一致した箇所をこの文字列で置き換えます。
scope	• one : 正規表現に最初に一致した箇所を置き換えます (デフォルト)。 • all : 正規表現に一致したすべての箇所を置き換えます。

使用方法
正規表現の使用法の詳細については、『ColdFusion アプリケーションの開発』の Using Regular Expressions in Functions を参照してください。

例
```
<p>The REReplaceNoCase function returns <i>string</i> with a regular
    expression replaced with <i>substring</i> in the specified scope.
    This is a case-insensitive search.
<p>REReplaceNoCase("cabaret","C|B","G","ALL"):
<cfoutput>#REReplaceNoCase("cabaret","C|B","G","ALL")#</cfoutput>
<p>REReplaceNoCase("cabaret","[A-Z]","G","ALL"):
<cfoutput>#REReplaceNoCase("cabaret","[A-Z]","G","ALL")#</cfoutput>
<p>REReplaceNoCase("I LOVE JELLIES","jell(y|ies)","cookies"):
<cfoutput>#REReplaceNoCase("I LOVE JELLIES","jell(y|ies)","cookies")#
</cfoutput>
<p>REReplaceNoCase("I LOVE JELLY","jell(y|ies)","cookies"):
<cfoutput>#REReplaceNoCase("I LOVE JELLY","jell(y|ies)","cookies")#
    </cfoutput>
```

RestDeleteApplication

説明
ディレクトリパスの登録を削除します（既に登録されている場合）。

戻り値
なし

シンタックス
RestDeleteApplication("dirPath")

プロパティ

パラメータ	説明
dirPath	必須です。登録を削除するディレクトリのパスです。パスが有効でない場合、エラーが発生します。

例
```
<cfset RestDeleteApplication("C:/ColdFusion10/cfusion/wwwroot/restexample/")>
```

RestSetResponse

説明
カスタムレスポンスを設定します。

シンタックス
restSetResponse(response)

パラメータ

パラメータ	説明
response	レスポンスの詳細が含まれる構造体です。

例

```
<cffunction name="create" httpMethod="POST" produces="application/xml">
    <cfargument name="id" type="numeric" argtype="FormParam">
        <cfargument name="name" type="String" argtype="FormParam">
    <!--- create the customer. --->
        <cfset response=structNew()>
        <cfset response.status=201>
        <cfset response.content="<customer id="&id&"><name>"&name&"</name></customer>">
        <cfset response.headers=structNew()>
        <cfset response.headers.location="http://localhost:8500/rest/CustomerService/customers/123">
        restSetResponse( response );
<cffunction>
```

RestInitApplication

説明

指定したサービスマッピングでディレクトリパスを登録します。サービスマッピングを指定しない場合は、アプリケーション名が使用されます。REST アプリケーションが既に登録済みの場合は更新されます。

戻り値

なし

シンタックス

RestInitApplication("dirPath" [, "serviceMapping"])

プロパティ

パラメータ	説明
dirPath	必須です。登録するディレクトリのパスです。
serviceMapping	オプション。REST サービスの呼び出し時に、アプリケーション名に使用される代替文字列です。

例

```
<cfset RestInitApplication("C:/ColdFusion10/cfusion/wwwroot/restexample/", "restexample")>
```

Reverse

説明

項目 (文字列に含まれる文字や、数値に含まれる数字など) の順序を逆にします。

戻り値

文字の順序が逆になった **string** のコピー

カテゴリ

文字列関数

関数のシンタックス

```
Reverse(string)
```

関連項目

Left、Mid、Right

パラメータ

パラメータ	説明
string	文字列、または文字列を含んでいる変数です。

使用方法

この関数を数値に対して使用したコードの例を次に示します。

```
<cfoutput>reverse(6*2) equals #reverse(6*2)#<cfoutput>
```

このコードの出力は次のようになります。

```
reverse(6*2) equals 21
```

例

```
<h3>Reverse Example</h3>

<p>Reverse returns your string with the positions of the characters reversed.
<cfif IsDefined("FORM.myString")>
    <cfif FORM.myString is not "">
        <p>Reverse returned:
        <cfoutput>#Reverse(FORM.myString)#</cfoutput>
    <cfelse>
        <p>Please enter a string to be reversed.
    </cfif>
</cfif>

<form action = "reverse.cfm">
<p>Enter a string to be reversed:
<input type = "Text" name = "MyString">
<p><input type = "Submit" name = "">
</form>
```

Right

説明

文字列の右端から、指定した数の文字を返します。

指定した文字列の最後 (または右端) から、指定した数の文字を返します。

戻り値

- 文字列の長さが **count** 以上の場合は、文字列の右端から **count** 個の文字が返されます。
- 文字列の長さが **count** 未満の場合は、文字列全体が返されます。
- **count** が 1 より大きく、文字列が空の場合は、空の文字列が返されます。

カテゴリ

文字列関数

関数のシンタックス

```
Right(string, count)
```

関連項目

Left、Mid、Reverse

パラメータ

パラメータ	説明
string	文字列、または文字列を含んでいる変数です。
count	返す最大文字数を指定する正の整数です。

例

```
<!--- Simple Right Example--->
<cfoutput>
#Right("See the quick red fox jump over the fence", 9)#
<br>
#Right("ColdFusion", 6)#
</cfoutput>

<!--- Right Example using form input --->
<h3>Right Example</h3>
<cfif IsDefined("Form.MyText")>
<!--- If len returns 0 (zero), then show error message. --->
    <cfif Len(FORM.myText)>
        <cfif Len(FORM.myText) LTE FORM.RemoveChars>
            <cfoutput><p style="color: red; font-weight: bold">Your string
            #FORM.myText# only has #Len(FORM.myText)# characters. You cannot output
            the #FORM.removeChars# rightmost characters of this string because it
            is not long enough.</p></cfoutput>
        <cfelse>
            <cfoutput><p>Your original string: <strong>#FORM.myText#</strong>
            <p>Your changed string, showing only the <strong>#FORM.removeChars#
            </strong> rightmost characters:
            <strong>#right(Form.myText, FORM.removeChars)#</strong></p>
            </cfoutput>
        </cfif>
    <cfelse>
        <p style="color: red; font-weight: bold">Please enter a string of more
        than 0 (zero) characters.</p>
    </cfif>
</cfif>

<form action="<cfoutput>#CGI.ScriptName#</cfoutput>" method="POST">
<p>Type in some text<br />
<input type="Text" name="myText"></p>
<p>How many characters from the right do you want to show?
<select name="RemoveChars">
<option value="1">1
<option value="3" selected>3
<option value="5">5
<option value="7">7
<option value="9">9</select>
<input type="Submit" name="Submit" value="Remove characters"></p>
</form>
```

RJustify

説明
文字列内の文字を右揃えにします。

戻り値
指定した長さのフィールド内で右揃えにした文字列のコピー

カテゴリ
表示および書式制御関数、文字列関数

関数のシンタックス
```
RJustify(string, length)
```

関連項目
CJustify、LJustify

パラメータ

パラメータ	説明
string	引用符で囲まれた文字列、またはそのような文字列を含んでいる変数です。
length	正の整数、または正の整数を含んでいる変数です。文字列を右揃えにするためのフィールドの長さです。

例
```
<!--- This example shows how to use RJustify --->
<cfparam name = "jstring" default = "">

<cfif IsDefined("FORM.justifyString")>
    <cfset jstring = rjustify(FORM.justifyString, 35)>
</cfif>
<html>
<head>
<title>RJustify Example</title>
</head>
<body>
<h3>RJustify Function</h3>
<p>Enter a string. It will be right justified within the sample field

<form action = "rjustify.cfm">
<p><input type = "Text" value = "<cfoutput>#jString#</cfoutput>"
    size = 35 name = "justifyString">

<p><input type = "Submit" name = ""> <input type = "reset">
</form>
```

Round

説明
数値を直近の整数に丸めます。

戻り値
整数。

カテゴリ
算術関数

関数のシンタックス
```
Round(number)
```

関連項目
Ceiling、Fix、Int

パラメータ

パラメータ	説明
number	丸める対象の数値です。

使用方法
この関数を使用して、数値を丸めます。この関数は、末尾が .5 の数値を直近の整数に繰り上げます。つまり、3.5 を 4 に、–3.5 を -3 に繰り上げます。

例
```
<h3>Round Example</h3>
<p>This function rounds a number to the closest integer.
<ul>
    <li>Round(7.49) : <cfoutput>#Round(7.49)#</cfoutput>
    <li>Round(7.5) : <cfoutput>#Round(7.5)#</cfoutput>
    <li>Round(-10.775) : <cfoutput>#Round(-10.775)#</cfoutput>
    <li>Round(-35.5) : <cfoutput>#Round(-35.5)#</cfoutput>
    <li>Round(35.5) : <cfoutput>#Round(35.5)#</cfoutput>
    <li>Round(1.2345*100)/100 : <cfoutput>#Round(1.2345*100)/100#</cfoutput>
</ul>
```

RTrim

説明
文字列の末尾のスペースを削除します。

戻り値
string から末尾のスペースを削除した結果のコピー

カテゴリ
文字列関数

関数のシンタックス
```
RTrim(string)
```

関連項目
LTrim、Trim

パラメータ

パラメータ	説明
string	文字列、または文字列を含んでいる変数です。

例

```
<h3>RTrim Example</h3>

<cfif IsDefined("FORM.myText")>
<cfoutput>
<pre>
Your string:"#FORM.myText#"
Your string:"#RTrim(FORM.myText)#"
(right trimmed)
</pre>
</cfoutput>
</cfif>

<form action = "Rtrim.cfm" method="post">
<p>Enter some text. It will be modified by Rtrim to remove spaces from the right.
<p><input type = "Text" name = "myText" value = "TEST ">

<p><input type = "Submit" name = "">
</form>
```

関数 S

Second

説明

日付時刻オブジェクトから秒の序数値を取り出します。

戻り値

0 ～ 59 の整数

カテゴリ

日付および時刻関数

関数のシンタックス

```
Second(date)
```

関連項目

DatePart、Hash、Minute

パラメータ

パラメータ	説明
date	日付時刻オブジェクトです。

使用方法
日付時刻オブジェクトを文字列として渡すときは、そのオブジェクトを引用符で囲みます。引用符で囲まない場合、その値は日付時刻オブジェクトの数値表現として解釈されます。

例
```
<!--- This example shows the use of Hour, Minute, and Second --->
<h3>Second Example</h3>
<cfoutput>
The time is currently #TimeFormat(Now())#.
We are in hour #Hour(Now())#, Minute #Minute(Now())#
and Second #Second(Now())# of the day.
</cfoutput>
```

SendGatewayMessage

説明
ColdFusion イベントゲートウェイを通じて発信メッセージを送信します。

戻り値
文字列。返される値は、ゲートウェイタイプによって異なります。

カテゴリ
拡張関数

関数のシンタックス
```
SendGatewayMessage(gatewayID, data)
```

関連項目
GetGatewayHelper、IM ゲートウェイメッセージ送信コマンド、SMS ゲートウェイ CFEvent の構造体とコマンド、CFML イベントゲートウェイ SendGatewayMessage の data パラメータ、および『ColdFusion アプリケーションの開発』の Sending a message using the SendGatewayMessage function

履歴
ColdFusion MX 7: この関数が追加されました。

パラメータ

パラメータ	説明
gatewayID	メッセージを送信するゲートウェイの識別子です。ColdFusion Administrator の [イベントゲートウェイ] セクションの [ゲートウェイ] ページで設定された、いずれかの ColdFusion イベントゲートウェイインスタンスのゲートウェイ ID でなければなりません。
data	ColdFusion 構造体です。構造体の内容はイベントゲートウェイタイプによって異なりますが、通常は、送信するメッセージを示す MESSAGE フィールドおよび送信先アドレスを示すフィールドが含まれています。

使用方法
SendGatewayMessage 関数は、指定されたゲートウェイの outgoingMessage メソッドを呼び出します。この関数が返す値は、ゲートウェイタイプによって異なります。次の表に、標準の ColdFusion ゲートウェイタイプの戻り値を示します。

ゲートウェイタイプ	戻り値
非同期 CFML	CFC に送信されるメッセージがキューに入れられると、true が返されます。それ以外の場合は、false が返されます。
Lotus SameTime	メッセージまたはコマンドが正常に処理されると、OK が返されます。 エラーが発生した場合は、エラーの理由を示す文字列が返されます。
SMS	ゲートウェイが非同期モードの場合、直ちに空の文字列が返されます。 ゲートウェイが同期モードの場合、この関数はゲートウェイからのレスポンスを待機します。メッセージが SMSC (Short Message Service Center) に正常に送信された場合は、SMSC からメッセージ ID が返されます。エラーが発生した場合は、エラーの理由を示す文字列が返されます。
XMPP	メッセージまたはコマンドが正常に処理されると、OK が返されます。 エラーが発生した場合は、エラーの理由を示す文字列が返されます。

例

次の例では、CFML ゲートウェイのインスタンスを使用して、メッセージをファイルに非同期で記録します。この例を使用するには、ColdFusion Administrator で "Asynch Logger" という名前を使用して CFML ゲートウェイのインスタンスを設定します。このゲートウェイインスタンスでは、メッセージを受け取って記録する CFC を使用する必要があります。CFC コードの例については、『ColdFusion アプリケーションの開発』の Using the CFML event gateway for asynchronous CFCs を参照してください。

```
Sending an event to the CFML event gateway that is registered in the
    ColdFusion Administrator as Asynch Logger.<br>
<cfscript>
    status = false;
    props = structNew();
    props.message = "Replace me with a variable with data to log";
    status = SendGatewayMessage("Asynch Logger", props);
    if (status IS True) WriteOutput('Event Message "#props.message#" has been sent.');
</cfscript>
```

SerializeJSON

説明

ColdFusion データを JSON (JavaScript Object Notation) 表現に変換します。

戻り値

パラメータ値の JSON 表現を含む文字列

カテゴリ

変換関数

シンタックス

```
SerializeJSON(var[, serializeQueryByColumns])
```

関連項目

DeserializeJSON、IsJSON、cfajaxproxy、『ColdFusion アプリケーションの開発』の Using data interchange formats、http://www.json.org

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
var	ColdFusion データ値、または ColdFusion データ値を表す変数です。
serializeQueryByColumns	ColdFusion クエリーをシリアル化する方法を指定するブール値です。 • false (デフォルト) : 2 つのエントリを含むオブジェクトを作成します。これら 2 つのエントリは、列名の配列と、行配列の配列です。この形式は、HTML 形式の cfgrid タグに必須です。 • true: WDDX クエリー形式に対応するオブジェクトを作成します。 詳細については、「使用方法」を参照してください。

使用方法

この関数は、Ajax アプリケーションで使用する JSON 形式のデータを生成する場合に便利です。

SerializeJSON 関数は、ColdFusion の日付時刻オブジェクトを変換して、JavaScript の Date オブジェクトで簡単に解析できる文字列を返します。この文字列の形式は次のようになります。

```
MonthName, DayNumber Year Hours:Minutes:Seconds
```

たとえば、SerializeJSON 関数では、2007 年 10 月 3 日 PM 3:01 を表す ColdFusion の日付時刻オブジェクトが、"October, 03 2007 15:01:00" という JSON 文字列に変換されます。

serializeQueryByColumns パラメータが false (デフォルト) に設定された SerializeJSON 関数では、ColdFusion クエリーは、次の要素を含む行指向の JSON オブジェクトに変換されます。

要素	説明
COLUMNS	列名の配列です。
DATA	次のエントリを含む 2 次元配列です。 • 外側の配列の各エントリはクエリーデータの行に相当します。 • 内側の配列の各エントリは行の列フィールドの値に相当し、COLUMNS 配列エントリと同じ順序になります。

たとえば、serializeQueryByColumns パラメータの値が false に設定された SerializeJSON 関数では、2 つの列 (City と State) と 2 つのデータ行を含む ColdFusion クエリーが次の形式に変換されます。

```
{"COLUMNS":["CITY","STATE"],"DATA":[["Newton","MA"],["San Jose","CA"]]}
```

serializeQueryByColumns パラメータの値が true に設定された SerializeJSON 関数では、ColdFusion クエリーは、WDDX クエリー表現に相当する列指向の JSON オブジェクトに変換されます。JSON オブジェクトには、次の 3 つの要素があります。

要素	説明
ROWCOUNT	クエリーの行数です。
COLUMNS	列名の配列です。
DATA	次のキーと値を含むオブジェクトです。 • キーはクエリー列名です。 • 値は列データが含まれている配列です。

serializeQueryByColumns パラメータの値が true に設定された SerializeJSON 関数では、2 つの列 (City と State) と 2 つのデータ行を含む ColdFusion クエリーが次の形式に変換されます。

```
{"ROWCOUNT":2, "COLUMNS":["CITY","STATE"],"DATA":{"City":["Newton","San Jose"],"State":["MA","CA"]}}
```

注意：SerializeJSON 関数でバイナリデータを JSON 形式に変換しようとするとエラーが発生します。

SerializeJSON 関数は、他のすべての ColdFusion データ型を、その型に対応する JSON データ型に変換します。構造体は JSON オブジェクトに、配列は JSON 配列に、数値は JSON 数値に、文字列は JSON 文字列に変換されます。

注意：ColdFusion の内部では、構造体のキー名はすべて大文字で表現されます。そのためキー名は、すべて大文字の JSON 表現にシリアル化されます。ColdFusion 構造体の JSON 表現を処理する JavaScript では、CITY や STATE のように、すべて大文字の構造体キー名を使用する必要があります。また、ColdFusion クエリーを JSON 形式で表現する 2 つの配列のキーにも、COLUMNS および DATA というすべて大文字の名前を使用します。

例

次の例では、2 つの都市の簡単な天気データを提供する JSON 形式のデータフィードを作成します。このデータフィードは、パラメータとして JSON オブジェクトを取る 1 つの関数呼び出しから構成される JavaScript アプリケーションの形式になっています。このコード例では、次の処理が実行されます。

1. 2 行の天気データを含むクエリーオブジェクトを作成します。各行には、都市名、現在の気温、および予報データの構造体の配列があり、それぞれに 1 日の最高気温、最低気温、天気予報データが含まれています。通常、これらのデータはデータソースから提供されます。この例ではコードを簡素化するために、すべての都市と日に対して同じ予報データを使用します。
2. クエリーを JSON 形式文字列に変換し、その文字列を JavaScript 関数呼び出しでラップします。
3. 結果を出力に書き込みます。

このページをブラウザで表示すると、使用された JavaScript 関数と JSON パラメータが表示されます。このページの結果をアプリケーションで使用するには、このファイルと DeserializeJSON 関数の例を ColdFusion の Web ルート配下の適切な場所に置き、DeserializeJSON の例の URL をこのページの URL に置き換えてから、DeserializeJSON の例を実行します。

```
<!--- Generate a clean feed by suppressing white space and debugging
         information. --->
<cfprocessingdirective suppresswhitespace="yes">
<cfsetting showdebugoutput="no">
<!--- Generate the JSON feed as a JavaScript function. --->
<cfcontent type="application/x-javascript">

<cfscript>
    // Construct a weather query with information on cities.
    // To simplify the code, we use the same weather for all cities and days.
    // Normally this information would come from a data source.
    weatherQuery = QueryNew("City, Temp, Forecasts");
    QueryAddRow(weatherQuery, 2);
    theWeather=StructNew();
    theWeather.High=73;
    theWeather.Low=53;
    theWeather.Weather="Partly Cloudy";
    weatherArray=ArrayNew(1);
    for (i=1; i<=5; i++) weatherArray[i]=theWeather;
    querySetCell(weatherQuery, "City", "Newton", 1);
    querySetCell(weatherQuery, "Temp", "65", 1);
    querySetCell(weatherQuery, "ForeCasts", weatherArray, 1);
    querySetCell(weatherQuery, "City", "San Jose", 2);
    querySetCell(weatherQuery, "Temp", 75, 2);
    querySetCell(weatherQuery, "ForeCasts", weatherArray, 2);

    // Convert the query to JSON.
    // The SerializeJSON function serializes a ColdFusion query into a JSON
    // structure.
    theJSON = SerializeJSON(weatherQuery);

    // Wrap the JSON object in a JavaScript function call.
    // This makes it easy to use it directly in JavaScript.
    writeOutput("onLoad( "&theJSON&" )");
</cfscript>
</cfprocessingdirective>
```

SessionInvalidate

説明
現在のセッションを無効化またはクリーンアップします。

注意：sessionInvalidate() メソッドでは、基盤となる J2EE セッションは無効化されません。

戻り値
なし

カテゴリ
表示および書式制御関数

シンタックス
sessionInvalidate()

関連項目
SessionRotate

履歴
ColdFusion 10: この関数が追加されました。

パラメータ
なし

使用方法
この関数は、既存のセッションを無効にするために使用します。

例
Application.cfc

```
<cfcomponent>
<cfset this.sessionManagement = true />
<cfset this.name = "session_app" />
</cfcomponent>
```

sessionInvalidate.cfm

```
<cfif isDefined("url.invalidate") >
    <cfset sessionInvalidate()/>
</cfif>
<cfif isDefined("url.name") >
    <cfset session.name = url.name />
</cfif>
<cfdump var="#session#" label="SESSION">
<cfoutput>
<a href="sessionInvalidate.cfm?name=BOB">Set session.name = BOB </a> <br/>
<a href="sessionInvalidate.cfm?invalidate=TRUE">Invalidate the session</a>
</cfoutput>
```

SessionRotate

説明
セッションの開始時に新しいセッションに取り替えます。例えば、ログインに成功した後に、新しいセッションを生成したい場合などです。認証が成功する前と後でセッションが異なるので、セッションへの攻撃を防ぐことができます。

方法は次のとおりです。

- セッションを作成します。
- 古いセッションから新しいセッションにデータをコピーします。
- 古いセッションを無効にします。
- 古いセッションの Cookie を無効化または上書きします。
- 古いセッションの Cookie を無効にした場合は、新しいセッションの Cookie を作成します。
- クライアントのストレージデータを新しいセッションキーにコピーして更新します。

戻り値
なし

カテゴリ
表示および書式制御関数

シンタックス
SessionRotate()

関連項目
SessionInvalidate

履歴
ColdFusion 10: この関数が追加されました。

パラメータ
なし

使用方法
この関数は、セッションを交代するために使用します。

例
Application.cfc

```
<cfcomponent>
<cfset this.sessionManagement = true />
<cfset this.name = "session_app" />
</cfcomponent>
```

sessionRotate.cfm

```
<cfif isDefined("url.rotate") >
    <cfset sessionRotate()/>
</cfif>
<cfif isDefined("url.name") >
    <cfset session.name = url.name />
</cfif>
<cfdump var="#session#" label="SESSION">
<cfoutput>
<a href="sessionRotate.cfm?name=BOB">Set session.name = BOB </a> <br/>
<a href="sessionRotate.cfm?rotate=TRUE">Rotate the session</a>
</cfoutput>
```

sessionGetMetaData

説明
現在のセッションに関連するメタデータを返します。

戻り値
現在のセッションに関連するメタデータ構造体

構造体の要素	説明
starttime	現在のセッションの開始時刻です。

シンタックス
sessionGetMetaData()

履歴
ColdFusion 10: この関数が追加されました。

例
```
<cfset metaData = sessionGetMetaData() >
```

SetEncoding

説明
Form スコープ変数値と URL スコープ変数値の文字エンコード (文字セット) を設定します。フォーム入力用の文字エンコードや URL の文字エンコードが UTF-8 エンコードでない場合に使用します。

戻り値
なし

カテゴリ
各国語対応関数、システム関数

関数のシンタックス
```
SetEncoding(scope_name,charset)
```

関連項目
GetEncoding、cfcontent、cfprocessingdirective、URLDecode、URLEncodedFormat、『ColdFusion アプリケーションの開発』の Locales

履歴
ColdFusion MX: この関数が追加されました。

パラメータ

パラメータ	説明
scope_name	• url • form
charset	文字エンコードです。スコープ変数のテキストがこの方式でエンコードされます。一般的に使用される値を次に示します。 • utf-8 • iso-8859-1 • windows-1252 • us-ascii • shift_jis • iso-2022-jp • euc-jp • euc-kr • big5 • euc-cn • utf-16

使用方法

この関数は、フォーム入力の文字エンコードや URL の文字エンコードが UTF-8 エンコードでない場合に使用します。たとえば、中国語 (Traditional) の文字には多くの場合 Big5 エンコードが使用されます。この関数は URL 変数や Form 変数をリセットするので、(通常は Application.cfm ページまたは "Application.cfc" ファイルで) これらの変数を使用する前にこの関数を呼び出す必要があります。この関数を最初に呼び出すことにより、これらの変数の文字が誤って解釈されることも回避できます。

文字エンコードの詳細については、次の Web ページを参照してください。

- www.w3.org/International/O-charset.html には、文字エンコードと Web に関する全般的な情報や、いくつかの有用なリンクがあります。
- www.iana.org/assignments/character-sets には、インターネットで使用され、Internet Assigned Numbers Authority によって管理されている文字セット名の完全なリストが掲載されています。
- java.sun.com/j2se/1.4.1/docs/guide/intl/encoding.doc.html には、Java 1.4.1 が解釈できる文字エンコード (つまりデフォルトの ColdFusion 設定で解釈できる文字エンコード) のリストがあります。Sun Java 2 Platform, Standard Edition, v 1.4.1 に準拠していない JVM を使用している場合には、サポートされているロケールが異なることがあります。このリストで使われているのは Java の内部名です。IANA 文字エンコード名ではありません。SetEncoding の charset パラメータや、ColdFusion のその他の属性およびパラメータでは、通常は IANA 文字エンコード名を使用します。Java では、標準的な IANA 名から内部名への変換が必要に応じて自動的に行われます。

例

```
<!--- This example sends and interprets the contents of two fields as
        big5 encoded text. Note that the form fields are received as URL variables
        because the form uses the GET method. --->
<cfcontent type="text/html; charset=big5">
<form action='#cgi.script_name#' method='get'>
<input name='xxx' type='text'>
<input name='yyy' type='text'>
<input type="Submit" value="Submit">
</form>

<cfif IsDefined("URL.xxx")>
<cfscript>
    SetEncoding("url", "big5");
    WriteOutput("URL.XXX is " & URL.xxx & "<br>");
    WriteOutput("URL.YYY is " & URL.yyy & "<br>");
theEncoding = GetEncoding("URL");
     WriteOutput("The URL variables were decoded using '" &
    theEncoding & "' encoding.");

WriteOutput("The encoding is " & theEncoding);
</cfscript>
</cfif>
```

SetLocale

説明

国 / 言語ロケールを設定します。このロケールは、ColdFusion の処理と、クライアントに返されるページに適用されます。言語および地域の表記規則に従って、日付、時刻、数値、および通貨値のデフォルト表示形式がロケール値によって決定されます。

戻り値

ロケールを設定する前に設定されていたロケール値を表す文字列

カテゴリ

各国語対応関数、システム関数

関数のシンタックス

```
SetLocale(new_locale)
```

関連項目

GetHttpTimeString、GetLocale、GetLocaleDisplayName、『ColdFusion アプリケーションの開発』の Locales

履歴

ColdFusion MX 7: ColdFusion Java ランタイムがサポートするすべてのロケールに対応できるようになりました。

ColdFusion MX:

- 形式設定の動作が変更されました。この関数では、以前のリリースと異なる値が返される場合があります。この関数では、すべてのプラットフォームで Java 標準のロケール決定ルールおよび形式設定ルールが使用されます。
- Spanish (Mexican) ロケールオプションが非推奨になりました。今後のリリースでは、動作せずにエラーとなる可能性があります。

- Spanish (Modern) オプションが変更されました。このロケールは Spanish (Standard) に設定されるようになりました。

パラメータ

パラメータ	説明
new_locale	ロケール名です。例 : "English (US)"

使用方法

Server.Coldfusion.SupportedLocales 変数にリストされている任意のロケール名を指定できます。この変数は、JVM がサポートするすべてのロケール名、および ColdFusion MX 7 より前の ColdFusion で必要であったロケール名のカンマ区切りリストです。

次のロケール名は、ColdFusion MX 6.1 までの ColdFusion リリースで使用され、継続してサポートされます。これらの値のいずれかを SetLocale 関数で使用する場合は、対応する Java ロケール名ではなくユーザーが設定した名前を GetLocale 関数が返します。

Chinese (China)	French (Belgian)	Korean
Chinese (Hong Kong)	French (Canadian)	Norwegian (Bokmal)
Chinese (Taiwan)	French (Standard)	Norwegian (Nynorsk)
Dutch (Belgian)	French (Swiss)	Portuguese (Brazilian)
Dutch (Standard)	German (Austrian)	Portuguese (Standard)
English (Australian)	German (Standard)	Spanish (Modern)
English (Canadian)	German (Swiss)	Spanish (Standard)
English (New Zealand)	Italian (Standard)	Swedish
English (UK)	Italian (Swiss)	
English (US)	Japanese	

ColdFusion では、ロケールの値を次のように決定します。

- デフォルトでは、JVM ロケールが使用されます。デフォルトの JVM ロケールはオペレーティングシステムのロケールです。ColdFusion では、ColdFusion Administrator の [Java と JVM の設定] ページの [JVM 引数] フィールドで JVM ロケール値を明示的に設定できます。例 :

  ```
  -Duser.language=de -Duser.region=DE.
  ```

- SetLocale 関数で設定されたロケールは現在のリクエストの終了まで、または、現在のリクエストの途中で再度の SetLocale 呼び出しによる再設定が行われるまで保持されます。
- 1 回のリクエスト中で複数回 SetLocale 関数が使用された場合、ロケールの影響を受ける ColdFusion タグおよび関数 (LS で始まる関数など) がデータを形式設定する方法は、処理時点のロケール設定により決定されます。レスポンスの Content-Language HTTP ヘッダの値は、リクエスタ (通常はクライアントブラウザ) にレスポンスを送信する前に最後に処理された SetLocale 関数によって決まります。ページをリクエストしたブラウザは、Content-Language ヘッダで指定されている言語の規則に従ってレスポンスを表示します。
- cfflush タグの後の SetLocale 関数は無視されます。

この関数は変更前のロケール設定を戻り値として返します。これを使えば、元のロケール値を保存しておくことができます。元のロケールに戻すには、保存しておいた変数を使って SetLocale 関数を再度呼び出します。たとえば次のコードでは、元のロケールを Session 変数に保存しています。

```
<cfset Session.oldlocale = SetLocale(newLocale)>
```

server.ColdFusion.SupportedLocales 変数は起動時に初期化され、ColdFusion およびオペレーティングシステムがサポートするロケールのカンマ区切りリストが格納されます。このリストに含まれないロケールを指定して SetLocale を呼び出すと、エラーが発生します。

注意：ColdFusion では、Spanish (Modern) と Spanish (Standard) に対して Spanish (Standard) 形式が使用されます。

例

```
<h3>SetLocale Example</h3>
<p>SetLocale sets the locale to the specified new locale for the current session.
<p>A locale encapsulates the set of attributes that govern the display and
    formatting of date, time, number, and currency values.
<p>The locale for this system is <cfoutput>#GetLocale()#</cfoutput>
<p><cfoutput><I>the old locale was #SetLocale("English (UK)")#</I>
<p>The locale is now #GetLocale()#</cfoutput>
```

SetProfileString

説明

初期化ファイルのプロファイルエントリの値を設定します。

戻り値

正常に実行された場合は空の文字列、そうでない場合はエラーメッセージ

カテゴリ

システム関数

関数のシンタックス

```
SetProfileString(iniPath, section, entry, value)
```

関連項目

GetProfileSections、GetProfileString

パラメータ

パラメータ	説明
iniPath	初期化ファイルの絶対パスです。
section	初期化ファイルのセクションです。ここで指定したセクションの中にエントリが設定されます。
entry	設定するエントリの名前です。
value	エントリに設定する値です。

例

```
 <h3>SetProfileString Example</h3>
This example uses SetProfileString to set the time-out value in an
    initialization file. Enter the full path of your initialization
    file, specify the time-out value, and submit the form.
<!--- This section checks whether the form was submitted. If so, this
    section sets the initialization path and time-out value to the
    path and time-out value specified in the form --->
<cfif Isdefined("Form.Submit")>

    <cfset IniPath = FORM.iniPath>
    <cfset Section = "boot loader">
    <cfset MyTimeout = FORM.MyTimeout>
    <cfset timeout = GetProfileString(IniPath, Section, "timeout")>

    <cfif timeout Is Not MyTimeout>
     <cfif MyTimeout Greater Than 0>
         <hr size = "2" color = "#0000A0">
        <p>Setting the time-out value to <cfoutput>#MyTimeout#</cfoutput>
        </p>
            <cfset code = SetProfileString(IniPath,
                Section, "timeout", MyTimeout)>
        <p>Value returned from SetProfileString:
                <cfoutput>#code#</cfoutput></p>
        <cfelse>
            <hr size = "2" color = "red">
            <p>The time-out value should be greater than zero in order to
                provide time for user response.</p>
            <hr size = "2" color = "red">
        </cfif>
    <cfelse>
        <p>The time-out value in your initialization file is already
            <cfoutput>#MyTimeout#</cfoutput>.</p>
    </cfif>
    <cfset timeout = GetProfileString(IniPath, Section, "timeout")>
    <cfset default = GetProfileString(IniPath, Section, "default")>

    <h4>Boot Loader</h4>
    <p>The time-out is set to: <cfoutput>#timeout#</cfoutput>.</p>
    <p>Default directory is: <cfoutput>#default#</cfoutput>.</p>

</cfif>

<form action = "setprofilestring.cfm">
<hr size = "2" color = "#0000A0">
<table cellspacing = "2" cellpadding = "2" border = "0">
<tr>
<td>Full Path of Init File</td>
<td><input type = "Text" name = "IniPath"
                value = "C:\myboot.ini"></td>
</tr>
<tr>
<td>Time-out</td>
<td><input type = "Text" name = "MyTimeout" value = "30"></td>
</tr>
<tr>
<td><input type = "Submit" name = "Submit" value = "Submit"></td>
<td></td>
</tr>
</table>
</form>
```

SetVariable

説明
name パラメータで指定した変数を、value パラメータの値に設定します。

戻り値
変数の新しい値

カテゴリ
ダイナミック評価関数

関数のシンタックス
```
SetVariable(name, value)
```

関連項目
DE、Evaluate、IIf

パラメータ

パラメータ	説明
name	変数名
value	文字列、文字列名、または数値です。

使用方法
ダイナミックに命名した変数に値を設定するには、この関数の代わりに、直接代入ステートメントを使用できます。そのためには、ダイナミックに命名した変数を引用符とシャープ記号 (#) で囲みます。例 :

```
<cfset DynamicVar2 = "ABD">
<cfset "#DynamicVar2#" = "Test Value2">
```

また、次の 2 つの行は同等です。

```
<cfset "myVar#i#" = myVal>
SetVariable("myVar" & i, myVal)
```

詳細については、『ColdFusion アプリケーションの開発』の Using Expressions and Number Signs を参照してください。

例
```
<h3>SetVariable Example</h3>

<cfif IsDefined("FORM.myVariable")>
<!--- strip out url, client., cgi., session., caller. --->
<!--- This example only lets you set form variables --->
<cfset myName = ReplaceList(FORM.myVariable,
    "url,client,cgi,session,caller", "FORM,FORM,FORM,FORM,FORM")>

<cfset temp = SetVariable(myName, FORM.myValue)>
<cfset varName = myName>
<cfset varNameValue = Evaluate(myName)>
<cfoutput>
    <p>Your variable, #varName#
    <p>The value of #varName# is #varNameValue#
</cfoutput>
</cfif>
```

Sgn

説明
数値の符号を調べます。

戻り値
- **number** が正の数の場合は 1
- **number** が 0 の場合は 0
- **number** が負の数の場合は -1

カテゴリ
算術関数

関数のシンタックス
```
Sgn(number)
```

関連項目
Abs

パラメータ

パラメータ	説明
number	数値です。

例
```
<h3>Sgn Example</h3>
<p>Sgn determines the sign of a number. Returns 1 if number is positive;
    0 if number is 0; -1 if number is negative.
<p>Sgn(14): <cfoutput>#Sgn(14)#</cfoutput>
<p>Sgn(21-21): <cfoutput>#Sgn(21-21)#</cfoutput>
<p>Sgn(-0.007): <cfoutput>#Sgn(-0.007)#</cfoutput>
```

Sin

説明
ラジアン単位で指定した角度のサインを計算します。

戻り値
角度のサインを表す数値

カテゴリ
算術関数

関数のシンタックス
```
Sin(number)
```

関連項目
ASin、Cos、ACos、Tan、Atn、Pi

パラメータ

パラメータ	説明
number	サインを計算する角度です (単位 : ラジアン)。

使用方法

結果の範囲は -1 ～ 1 です。

度数をラジアンに変換するには、度数に p/180 を乗算します。ラジアンを度数に変換するには、ラジアンに 180/p を乗算します。

注意 : 浮動小数点演算の誤差により、本来は戻り値が 0 となるような角度を指定した場合、実際には非常に小さい値 (たとえば 6.12323399574E-017) が返されます。したがって、結果が 0 かどうかを調べる場合は、戻り値が 0.0000000000001 より小さいかどうかで判定してください。

例

```
 <h3>Sin Example</h3>
<!--- Calculate sine if form has been submitted --->
<cfif IsDefined("FORM.sinNum")>
<!--- Make sure input is a number --->
    <cfif IsNumeric(#FORM.sinNum#)>
<!--- Convert degrees to radians, call the Sin function. --->
<cfset sinValue=#Sin((Form.sinNum * PI()) / 180)#>
<!--- 0.0000000000001 is the function's precision limit.
    If absolute value of returned sine value is
    less, set result to 0 --->
<cfif Abs(sinValue) LT 0.0000000000001>
    <cfset sinValue=0>
</cfif>
<cfoutput>
        Sin(#FORM.sinNum#) = #sinValue#<br><br>
</cfoutput>
    <cfelse>
<!--- If input is not a number, show an error message --->
        <h4>You must enter a numeric angle in degrees.</h4>
    </cfif>
</cfif>
<form action = "#CGI.script_name#" method="post">
Enter an angle in degrees to get its sine:
<br><input type = "Text" name = "sinNum" size = "15">
<br><br>
<input type = "Submit" name = "">  
<input type = "RESET"
</form>
```

Sleep

説明

現在のスレッドの処理を一定時間停止します。

戻り値

なし

カテゴリ

システム関数

シンタックス

```
Sleep(duration)
```

関連項目

cfthread、『ColdFusion アプリケーションの開発』の Using ColdFusion Threads

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
duration	スレッドの処理を停止する時間です (単位 : ミリ秒)。

説明

Sleep 関数は、別のスレッドの処理が完了するまで現在のスレッドを停止する必要がある場合などに便利です。停止する必要があるスレッドで Sleep 関数を使用すると、そのスレッドの処理が一定時間停止され、他のスレッドを開始する準備ができているかどうかが確認されます。開始の準備が整っていない場合は、再度そのスレッドをスリープ状態にすることができます。このような動作は、両方のスレッドに適用する初期化プロセスが他のスレッドで完了するまで現在のスレッドを停止させる必要がある場合などに便利です。

Sleep 関数の機能は、action 属性の値が sleep に設定された cfthread タグの機能と同じです。

例

次の例では 2 つのスレッドを使用します。2 番目のスレッド (threadB) で sleep 関数を使用し、最初のスレッド (threadA) が終了したことを確認してから threadB の処理を開始します。

```
<!--- ThreadA loops to simulate an initialization activity that might take time. --->
<cfthread name="threadA" action="run">
        <cfset thread.j=1>
        <cfloop index="i" from="1" to="99999">
        <cfset thread.j=thread.j+1>
    </cfloop>
</cfthread>

<!--- ThreadB loops while threadA is not finished, sleeping for
        1/2 second each time. --->
<cfthread name="threadB" action="run">
    <cfscript>
        thread.sleepTimes=0;
        thread.initialized=false;
        while ((threadA.Status != "COMPLETED") && (threadA.Status
                != "TERMINATED")) {
            sleep(500);
            thread.sleeptimes++;
        }
        // Only do the post-initilization code if the threadA completed.
        If (threadA.Status == "COMPLETED") {
            thread.initialized=true;
            // Post-initialization code would go here.
        }
    </cfscript>
</cfthread>

<!Join the threads. --->
<cfthread action="join" name="threadA,threadB" timeout="10000"/>

<!--- Display the thread information. --->
<!--- Different actions might be taken based on the thread status information. --->
<cfoutput>
    threadA index value: #threadA.j#<br />
    threadA status: #threadA.Status#<br>
    threadB status: #threadB.Status#<br>
    threadB sleepTimes: #threadB.sleepTimes#<br>
    threadB initialized: #threadB.initialized#<br>
</cfoutput>
```

SpanExcluding

説明

文字列の先頭から文字を取得し、指定した文字セットに含まれる文字が出現すると停止します。この検索では大文字と小文字が区別されます。

たとえば、SpanExcluding("MyString", "inS") と指定すると、"MyString" から "String" が除外されて、"My" が返されます。これは、文字列 "MyString" の "My" の後に、文字 'S' (2 番目の文字列 "inS" に含まれる文字) が出現するためです。

戻り値

string の先頭から、**set** に含まれるいずれかの文字が出現する位置までの間にある文字列

カテゴリ

文字列関数

関数のシンタックス

```
SpanExcluding(string, set)
```

関連項目

GetToken、SpanIncluding、『ColdFusion アプリケーションの開発』の Caching parts of ColdFusion pages

パラメータ

パラメータ	説明
string	文字列、または文字列を含んでいる変数です。
set	文字列、または文字列を含んでいる変数です。1 つまたは複数の文字が含まれている必要があります。

例

```
<h3>SpanExcluding Example</h3>

<cfif IsDefined("FORM.myString")>
<p>Your string was <cfoutput>#FORM.myString#</cfoutput>
<p>Your set of characters was <cfoutput>#FORM.mySet#</cfoutput>
<p>Your string up until one of the characters in the set is:
<cfoutput>#SpanExcluding(FORM.myString, FORM.mySet)#</cfoutput>
</cfif>

<p>Returns all characters from string from beginning to a character
    from the set of characters. The search is case-sensitive.

<form method = post action = "spanexcluding.cfm">
<p>Enter a string:
<br><input type = "Text" name = "myString" value = "Hey, you!">
<p>And a set of characters:
<br><input type = "Text" name = "mySet" value = "Ey">
<br><input type = "Submit" name = "">
</form>
```

SpanIncluding

説明

文字列の先頭から文字を取得し、指定した文字セットに含まれない文字が出現すると停止します。この検索では大文字と小文字が区別されます。

たとえば、SpanIncluding("mystring", "mystery") と指定すると、"mystr" が返されます。これは、文字列 "mystring" の "mystr" の後に、文字 'i' (2 番目の文字列 "mystery" に含まれない文字) が出現するためです。

戻り値

string の先頭から、**set** に含まれない文字が出現する位置までの間にある文字列

カテゴリ

文字列関数

関数のシンタックス

```
SpanIncluding(string, set)
```

関連項目

GetToken、SpanExcluding、『ColdFusion アプリケーションの開発』の Caching parts of ColdFusion pages

パラメータ

パラメータ	説明
string	検索文字列、または検索文字列を含んでいる変数です。
set	文字セットを指定する文字列、またはそれを含んでいる変数です。少なくとも 1 つの文字が含まれている必要があります。

例

```
<h3>SpanIncluding Example</h3>
<cfif IsDefined("FORM.myString")>
<p>Your string was <cfoutput>#FORM.myString#</cfoutput>
<p>Your set of characters was <cfoutput>#FORM.mySet#</cfoutput>
<p>Your string, until the characters in the set have been found, is:
<cfoutput>#SpanIncluding(FORM.myString, FORM.mySet)#</cfoutput>
</cfif>

<p>Returns characters of a string, from beginning to a character
    that is not in set. The search is case-sensitive.

<form action = "spanincluding.cfm" method="post">
<p>Enter a string:
<br><input type = "Text" name = "myString" value = "Hey, you!">
<p>And a set of characters:
<br><input type = "Text" name = "mySet" value = "ey,H">
<br><input type = "Submit" name = "">
</form>
```

SpreadsheetAddColumn

説明

Excel スプレッドシートオブジェクトに列または列データを追加します。

戻り値

なし

カテゴリ

Microsoft Office の統合

関数のシンタックス

```
SpreadsheetAddColumn(SpreadsheetObj, data[, startRow, startColumn, insert]);
```

関連項目

SpreadsheetAddImage、SpreadsheetAddRow、SpreadsheetAddRows、SpreadsheetDeleteColumn、SpreadsheetDeleteColumns、SpreadsheetFormatColumn、SpreadsheetFormatColumns、SpreadsheetShiftColumns

履歴

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
spreadsheetObj	列の追加先となる Excel スプレッドシートオブジェクトです。
data	セルエントリのカンマ区切りのリスト。列に追加される行ごとに 1 つあります。
startRow	このパラメータはオプションです。 列データの追加を開始する行の番号です。insert="true" の場合は、開始行の上にある列のすべての行に空のセルが含まれます。 このパラメータを省略すると、列は、最後の現行行に続く最初の行から挿入されます。列を指定することはできません。
startColumn	このパラメータはオプションです。 列データの追加先となる列の番号です。
insert	このパラメータはオプションです。 列を挿入するかどうかを指定するブール値です。false の場合は、指定された列エントリのデータが置き換えられます。

使用方法

spreadsheetaddcolumn 関数では、2 つの引数または 5 つの引数のいずれかを指定できます。

次に示すように、2 つのパラメータを使用して、spreadsheetaddcolumn 関数を指定できます。

```
<cfset spreadsheetAddColumn(SpreadsheetObj,"newcol1,newcol2,newcol3")>
```

次に示すように、5 つのパラメータを使用して、spreadsheetaddcolumn 関数を指定できます。

```
<cfset spreadsheetAddColumn(SpreadsheetObj,"newcol1,newcol2,newcol3",2,3,false)>
```

例

次の例では、クエリーから Excel スプレッドシートオブジェクトを作成し、行 3 から開始されるデータを含む新しい列 2 を挿入します。既存の列 2 およびそれ以降は 1 つずつインクリメントされます。

```
<!--- Get the spreadsheet data as a query. --->
<cfquery
       name="courses" datasource="cfdocexamples"
       cachedwithin="#CreateTimeSpan(0, 6, 0, 0)#">
       SELECT CORNUMBER,DEPT_ID,COURSE_ID,CORNAME
       FROM COURSELIST
</cfquery>

<cfscript>
    ///We need an absolute path, so get the current directory path.
    theFile=GetDirectoryFromPath(GetCurrentTemplatePath()) & courses.xls";
    //Create a new Spreadsheet object and add the query data.
    theSheet = SpreadsheetNew("CourseData");
    SpreadsheetAddRows(theSheet,courses);
    //Insert a new second column to the sheet, with data starting in row 3.
    SpreadsheetAddColumn(theSheet,
    "Basic,Intermediate,Advanced,Basic,Intermediate,Advanced,Basic,Intermediate,Advanced"
    ,3,2,true);
</cfscript>

<!--- Write the spreadsheet to a file, replacing any existing file. --->
<cfspreadsheet action="write" filename="#theFile#" name="theSheet" sheet=1 sheetname="courses"
overwrite=true>
```

SpreadsheetAddImage

説明
Excel スプレッドシートオブジェクトにイメージを追加します。

戻り値
なし

カテゴリ
Microsoft Office の統合

関数のシンタックス
```
SpreadsheetAddImage(SpreadsheetObj, imageFilePath, anchor)
SpreadsheetAddImage(SpreadsheetObj, imageData, imageType, anchor)
```

関連項目
SpreadsheetAddColumn、SpreadsheetAddRow、SpreadsheetAddRows

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
spreadsheetObj	列の追加先となる Excel スプレッドシートオブジェクトです。
anchor	イメージの場所です。次のいずれかの形式のカンマ区切りリストになります。 • *startRow,startColumn,endRow,endColumn* • *startXPosition,startYPosition,endXPosition,endYPositions,startRow,startColumn,endRow,endColumn* 1 番目の形式では、行番号と列番号のみが指定され、2 番目の形式では、セルの左上隅と相対的なピクセルの X 座標と Y 座標を使用してセル内の位置も示されます。1 番目の形式を使用する場合は、左上と右下のセル内のイメージの角の位置が 0,0 と 0,255 になります。
imageData	ColdFusion イメージオブジェクトです。
imageFilePath	イメージファイルの絶対パスです。
imageType	イメージの形式は、次のいずれかです。 • jpg または jpeg • png • dib

使用方法

例
次の例では、PNG 形式のチャートを作成し、新しいスプレッドシートの行 5 から 12 および列 5 から 10 に配置し、このシートをディスクに保存します。

```
<cfchart format="png"
    scalefrom="-100" scaleTo="100"
    gridlines="5"
    name="test">
    <cfchartseries type="line">
        <cfchartdata item="Point1" value="-50">
        <cfchartdata item="Point2" value="-25">
        <cfchartdata item="Point3" value="1">
        <cfchartdata item="Point4" value="25">
        <cfchartdata item="Point5" value="50">
        <cfchartdata item="Point6" value="75">
        <cfchartdata item="Point7" value="99">
    </cfchartseries>
</cfchart>

<cfscript>
       theDir=GetDirectoryFromPath(GetCurrentTemplatePath());
    SpreadsheetObj=SpreadsheetNew();
    SpreadsheetAddImage(SpreadsheetObj,test,"png","5,5,12,10");
</cfscript>

<cfspreadsheet action="write" name=SpreadsheetObj filename="#theDir#imagesheet.xls"
        sheetname="chart" overwrite=true>
```

SpreadsheetAddFreezePane

説明
ワークシートの特定の行または列をロックまたはフリーズします。

戻り値
なし

カテゴリ
Microsoft Office の統合

関数のシンタックス
```
SpreadsheetAddFreezePane(spreadsheetobj, freezcol, freezrow[, col, row])
```

関連項目
SpreadsheetAddColumn、SpreadsheetAddImage、SpreadsheetAddRow、SpreadsheetDeleteRow、SpreadsheetDeleteRows、SpreadsheetFormatRow、SpreadsheetFormatRows、SpreadsheetShiftRows

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
spreadsheetobj	フリーズペインの追加先となる Excel スプレッドシートオブジェクトです。
freezcol	フリーズペインの列境界を指定します。列境界内に含まれている列がフリーズされます。ワークシートの残りの部分はスクロールできます。
freezrow	フリーズペインの行境界を指定します。行境界内に含まれている行がフリーズされます。ワークシートの残りの部分はスクロールできます。
col	このパラメータはオプションです。 指定する freezcol の次に表示する列です。このパラメータは、データを非表示にする場合に役立ちます。たとえば、ワークシートで列 5 が列 2 のすぐ次に表示されるよう指定し、列 3 と 4 を非表示にできます。
row	このパラメータはオプションです。指定する freezrow の次に表示する行です。このパラメータは、データを非表示にする場合に役立ちます。たとえば、ワークシートで行 10 が行 7 のすぐ次に表示されるよう指定し、行 8 と 9 を非表示にできます。

使用方法

ペインをフリーズすることによって、ワークシートのある領域を表示したまま別の領域をスクロールできます。ペインをフリーズする場合は、ワークシートの特定の行および列をロックまたはフリーズします。ワークシートでフリーズされた行および列は、実線で示されます。

注意：ペインを 2 つのワークシート領域に分割することはできません。

例

次の例では、ワークシートの列 3 および行 2 でスプレッドシートをフリーズします。

```
SpreadSheetAddFreezePane(SpreadsheetObj,3,2);
```

次の例では、列 3 および行 2 でスプレッドシートをフリーズし、列 4 および行 3 から 10 を非表示にします。

```
SpreadSheetAddFreezePane(SpreadsheetObj,3,2,5,10);
```

SpreadsheetAddInfo

説明

新しいスプレッドシートのドキュメントプロパティを設定するか、または既存のスプレッドシートのプロパティを変更します。

戻り値

この関数は値を返しません。

カテゴリ

Microsoft Office の統合

関数のシンタックス

```
SpreadsheetAddInfo(spreadsheetobj, property_struct)
```

関連項目

SpreadsheetAddColumn、SpreadsheetAddImage、SpreadsheetAddRow、SpreadsheetDeleteRow、SpreadsheetDeleteRows、SpreadsheetFormatRow、SpreadsheetFormatRows、SpreadsheetShiftRows

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
spreadsheetobj	値の取得元となる Excel スプレッドシートオブジェクトです。
property_struct	次のスプレッドシートのプロパティは変更または設定できます。 • AUTHOR • CATEGORY • LASTAUTHOR • COMMENTS • KEYWORDS • MANAGER • COMPANY • SUBJECT • TITLE

使用方法
この関数は、Microsoft Office Excel 2007 (.xlsx) および Microsoft Office 2003 (.xls) でサポートされています。

例
```
<cfspreadsheet action="read" src="#filename#" name="a"  >
<cfset info = StructNew()>
<cfset info.title="Title">
<cfset info.category="Category test">
<cfset info.author="ABC">
<cfset info.comments="Comments for this file">
<cfset spreadsheetaddInfo(a,info)>
<cfspreadsheet action="write" filename="#dirname#SingleSheet.xls" name=a overwrite="yes">
```

SpreadsheetAddRow

説明
Excel スプレッドシートオブジェクトに行を追加します。

戻り値
なし

カテゴリ
Microsoft Office の統合

関数のシンタックス
```
SpreadsheetAddrow(spreadsheetObj, data [,row, column, insert])
```

関連項目

SpreadsheetAddColumn、SpreadsheetAddImage、SpreadsheetAddRows、SpreadsheetDeleteRow、SpreadsheetDeleteRows、SpreadsheetFormatRow、SpreadsheetFormatRows、SpreadsheetShiftRows

履歴

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
spreadsheetObj	列の追加先となる Excel スプレッドシートオブジェクトです。
data	セルエントリのカンマ区切りリスト。列ごとに 1 つあります。
row	挿入する行の番号です。この値以上の番号を持つ既存の行の行番号は、1 ずつインクリメントされます。このパラメータの値を指定する場合は、column の値も指定する必要があります。 このパラメータを省略すると、行は最後の現行行の後に挿入されます。列を指定することはできません。
column	列データの追加先となる列の番号です。行内の開始列より左側にあるすべての列のセルは空になります。このパラメータの値を指定する場合は、row の値も指定する必要があります。
insert	このパラメータはオプションです。デフォルト値は true です。 行を挿入するかどうかを指定するブール値です。false の場合は、指定された行エントリのデータが置き換えられます。

使用方法

例

次の例では、1 つのデータ行を行 10 として追加します。データは列 2 で開始され、10 より大きい既存の行番号は 1 ずつインクリメントされます。

```
<!--- Get the spreadsheet data as a query. --->
<cfquery
       name="courses" datasource="cfdocexamples"
       cachedwithin="#CreateTimeSpan(0, 6, 0, 0)#">
       SELECT CORNUMBER,DEPT_ID,COURSE_ID,CORNAME
       FROM COURSELIST
</cfquery>

<cfscript>
    ///We need an absolute path, so get the current directory path.
    theFile=GetDirectoryFromPath(GetCurrentTemplatePath()) & courses.xls";
    //Create a new Spreadsheet object and add the query data.
    theSheet = SpreadsheetNew("CourseData");

    SpreadsheetAddRows(theSheet,courses);
    //Insert a new eighth row to the sheet, with data starting in column 1.
    SpreadsheetAddRow(theSheet,"150,ENGL,95,Poetry 1",8,1);
</cfscript>

<!--- Write the spreadsheet to a file, replacing any existing file. --->
<cfspreadsheet action="write" filename="#theFile#" name="theSheet"
    sheet=1 sheetname="courses" overwrite=true>
```

SpreadsheetAddRows

説明
Excel スプレッドシートオブジェクトにクエリーから複数の行を追加します。

戻り値
なし

カテゴリ
Microsoft Office の統合

関数のシンタックス
```
SpreadsheetAddrows(spreadsheetObj, data[, row, column, insert])
```

関連項目
SpreadsheetAddColumn、SpreadsheetAddImage、SpreadsheetAddRow、SpreadsheetDeleteRow、SpreadsheetDeleteRows、SpreadsheetFormatRow、SpreadsheetFormatRows、SpreadsheetShiftRows

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
spreadsheetObj	列の追加先となる Excel スプレッドシートオブジェクトです。
data	行データまたは配列を含むクエリーオブジェクトです。
row	行を挿入する行番号です。この値以上の番号を持つ既存の行の行番号は、追加される行の数だけインクリメントされます。このパラメータの値を指定する場合は、column の値も指定する必要があります。 このパラメータを省略すると、行は最後の現行行の後に挿入されます。
column	列データの追加先となる列の番号です。行内の開始列より左側にあるすべての列のセルは空になります。このパラメータの値を指定する場合は、row の値も指定する必要があります。
insert	このパラメータはオプションです。デフォルト値は true です。 行を挿入するかどうかを指定するブール値です。false の場合は、指定された行エントリのデータが置き換えられます。

例
次の例では、新しい Excel スプレッドシートオブジェクトを作成し、AddRows 関数を使用してクエリーからデータを追加することによって、スプレッドシートを作成します。

```
<!--- Get the spreadsheet data as a query. --->
<cfquery
       name="courses" datasource="cfdocexamples"
       cachedwithin="#CreateTimeSpan(0, 6, 0, 0)#">
       SELECT CORNUMBER,DEPT_ID,COURSE_ID,CORNAME
       FROM COURSELIST
</cfquery>

<cfscript>
    ///We need an absolute path, so get the current directory path.
    theFile=GetDirectoryFromPath(GetCurrentTemplatePath()) & courses.xls";
    //Create a new Excel spreadsheet object and add the query data.
    theSheet = SpreadsheetNew("CourseData");

    SpreadsheetAddRows(theSheet,courses);SpreadsheetAddRows(theSheet,["1,a", "2,B,b"]);
</cfscript>

<!--- Write the spreadsheet to a file, replacing any existing file. --->
<cfspreadsheet action="write" filename="#theFile#" name="theSheet"
    sheet=1 sheetname="courses" overwrite=true>
```

SpreadsheetAddSplitPane

説明
ペインを 4 つのワークシート領域に分割します。

戻り値
なし

カテゴリ
Microsoft Office の統合

関数のシンタックス
```
SpreadsheetAddSplitPane(spreadsheetobj, x-position, y-position, splitcol, splitrow [,position])
```

関連項目
SpreadsheetAddColumn、SpreadsheetAddImage、SpreadsheetAddRow、SpreadsheetDeleteRow、SpreadsheetDeleteRows、SpreadsheetFormatRow、SpreadsheetFormatRows、SpreadsheetShiftRows

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
spreadsheetobj	分割ペインの追加先となる Excel スプレッドシートオブジェクトです。
x-position	最初の象限の X 軸の位置を指定します。X 位置の値は、ワークシートのピクセル値の 20 分の 1 になります。
y-position	最初の象限の Y 軸の位置を指定します。Y 位置の値は、ワークシートのピクセル値の 20 分の 1 になります。

パラメータ	説明
splitcol	スプレッドシートの象限 2 に表示される列を指定します。
splitrow	スプレッドシートの象限 3 に表示される行を指定します。
position	この属性はオプションです。 ペインを分割する分割バーの位置を指定します。次のいずれかを指定します。 • LOWER_LEFT • LOWER_RIGHT • UPPER_RIGHT • UPPER_LEFT

使用方法
スプレッドシートでペインを 4 つのワークシート領域に分割できます。分割はピクセルレベルで適用されます。ワークシート領域は、必要に応じて分割バーをドラッグして調整できます。

注意：ペインを 2 つのワークシート領域に分割することはできません。

例
次の例では、スプレッドシートを 4 つの象限に分割します。X 位置および Y 位置の値はそれぞれ 2000 と 2000 です。スプレッドシートの列 5 は象限 2 に表示され、行 7 は象限 3 に表示されます。分割バーは、スプレッドシートの左側の一番下に表示されます。

```
SpreadSheetAddSplitPane(spreadsheetobj, 2000, 2000, 5, 7, LOWER_LEFT );
```

SpreadsheetCreateSheet

説明
追加のスプレッドシートを作成します。

戻り値
なし

カテゴリ
Microsoft Office の統合

関数のシンタックス
```
SpreadsheetCreateSheet(spreadsheetObj, [sheetname])
```

関連項目
SpreadsheetAddColumn、SpreadsheetDeleteColumns、SpreadsheetDeleteRow、SpreadsheetDeleteRows、SpreadsheetFormatColumn、SpreadsheetFormatColumns、SpreadsheetShiftColumns、SpreadsheetSetActiveSheet

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
spreadsheetObj	追加シートの作成元となる Excel スプレッドシートオブジェクトです。
sheetname	新しいシートの名前です。このパラメータはオプションです。

例

次の例では、CourseData と EvaluationSheet という 2 つのシートを作成します。

```
<!--- Get the spreadsheet data as a query. --->
<cfquery
       name="courses" datasource="cfdocexamples"
       cachedwithin="#CreateTimeSpan(0, 6, 0, 0)#">
       SELECT CORNUMBER,DEPT_ID,COURSE_ID,CORNAME
       FROM COURSELIST
</cfquery>

<cfscript>
    ///We need an absolute path, so get the current directory path.
    theFile=GetDirectoryFromPath(GetCurrentTemplatePath()) & courses.xls";
    //Create a new Excel spreadsheet object and add the query data.
    theSheet = SpreadsheetNew("CourseData");

    SpreadsheetAddRows(theSheet,courses);
    //Add a new sheet.
    SpreadsheetCreateSheet (theSheet, "EvaluationSheet");
</cfscript>

<!--- Write the spreadsheet to a file, replacing any existing file. --->
<cfspreadsheet action="write" filename="#theFile#" name="theSheet"
    sheet=1 sheetname="courses" overwrite=true>
```

SpreadsheetDeleteColumn

説明

Excel スプレッドシートオブジェクトの列からデータを削除します。列は削除されません。

戻り値

なし

カテゴリ

Microsoft Office の統合

関数のシンタックス

```
SpreadsheetDeleteColumn(spreadsheetObj, column)
```

関連項目

SpreadsheetAddColumn、SpreadsheetDeleteColumns、SpreadsheetDeleteRow、SpreadsheetDeleteRows、SpreadsheetFormatColumn、SpreadsheetFormatColumns、SpreadsheetShiftColumns

履歴

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
spreadsheetObj	列を削除する Excel スプレッドシートオブジェクトです。
column	削除対象の列です。

例

次の例では、スプレッドシートの列 2 からデータを削除します。

```
<!--- Get the spreadsheet data as a query. --->
<cfquery
       name="courses" datasource="cfdocexamples"
       cachedwithin="#CreateTimeSpan(0, 6, 0, 0)#">
       SELECT CORNUMBER,DEPT_ID,COURSE_ID,CORNAME
       FROM COURSELIST
</cfquery>

<cfscript>
    ///We need an absolute path, so get the current directory path.
    theFile=GetDirectoryFromPath(GetCurrentTemplatePath()) & courses.xls";
    //Create a new Excel spreadsheet object and add the query data.
    theSheet = SpreadsheetNew("CourseData");

    SpreadsheetAddRows(theSheet,courses);
    //Delete the second column of the sheet.
    SpreadsheetDeleteColumn(theSheet,2);
</cfscript>

<!--- Write the spreadsheet to a file, replacing any existing file. --->
<cfspreadsheet action="write" filename="#theFile#" name="theSheet"
    sheet=1 sheetname="courses" overwrite=true>
```

SpreadsheetDeleteColumns

説明

Excel スプレッドシートオブジェクトの複数の列からデータを削除します。この関数は列を削除しません。

戻り値

なし

カテゴリ

Microsoft Office の統合

関数のシンタックス

```
SpreadsheetDeleteColumns(spreadsheetObj, range)
```

関連項目

SpreadsheetAddColumn、SpreadsheetDeleteColumn、SpreadsheetDeleteRow、SpreadsheetDeleteRows、SpreadsheetFormatColumn、SpreadsheetFormatColumns、SpreadsheetShiftColumns

履歴

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
spreadsheetObj	列を削除する Excel スプレッドシートオブジェクトです。
range	削除対象の列を含む文字列です。次の形式の任意の組み合わせを使用して指定します。 • *startColumn-endColumn*: 単一の範囲の列を削除します。 • *column,column,column...*: 複数の列を個別に削除します。 両方の形式を同時に使用することもできます。例 : 1, 2, 3-5, 7-12。

例

次の例では、スプレッドシートの列 2 から 4 および列 6 からデータを削除します。

```
<!--- Get the spreadsheet data as a query. --->
<cfquery
       name="courses" datasource="cfdocexamples"
       cachedwithin="#CreateTimeSpan(0, 6, 0, 0)#">
       SELECT CORNUMBER,DEPT_ID,CORLEVEL,COURSE_ID,CORNAME,CORDESC,LASTUPDATE
       FROM COURSELIST
</cfquery>

<cfscript>
    ///We need an absolute path, so get the current directory path.
    theFile=GetDirectoryFromPath(GetCurrentTemplatePath()) & courses.xls";
    //Create a new Excel spreadsheet object and add the query data.
    theSheet = SpreadsheetNew("CourseData");

    SpreadsheetAddRows(theSheet,courses);
    //Delete columns 2 though 4 and 6.
    SpreadsheetDeleteColumns(theSheet,"2-4,6");
</cfscript>

<!--- Write the spreadsheet to a file, replacing any existing file. --->
<cfspreadsheet action="write" filename="#theFile#" name="theSheet"
    sheet=1 sheetname="courses" overwrite=true>
```

SpreadsheetDeleteRow

説明

Excel スプレッドシートオブジェクトの行からデータを削除します。行は削除されません。

戻り値

なし

カテゴリ

Microsoft Office の統合

関数のシンタックス

```
SpreadsheetDeleteRow(spreadsheetObj, row)
```

関連項目

SpreadsheetDeleteColumn、SpreadsheetDeleteColumns、SpreadsheetDeleteRows、SpreadsheetFormatRows、SpreadsheetShiftRows

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
spreadsheetObj	行の削除元となる Excel スプレッドシートオブジェクトです。
row	削除する行です。

例
次の例では、スプレッドシートから行 10 を削除します。

```
<!--- Get the spreadsheet data as a query. --->
<cfquery
       name="courses" datasource="cfdocexamples"
       cachedwithin="#CreateTimeSpan(0, 6, 0, 0)#">
       SELECT CORNUMBER,DEPT_ID,CORLEVEL,COURSE_ID,CORNAME,CORDESC,LASTUPDATE
       FROM COURSELIST
</cfquery>

<cfscript>
    ///We need an absolute path, so get the current directory path.
    theFile=GetDirectoryFromPath(GetCurrentTemplatePath()) & courses.xls";
    //Create a new Excel spreadsheet object and add the query data.
    theSheet = SpreadsheetNew("CourseData");
    SpreadsheetAddRows(theSheet,courses);

    //Delete row 10.
    SpreadsheetDeleteRow(theSheet,"10");
</cfscript>

<!--- Write the spreadsheet to a file, replacing any existing file. --->
<cfspreadsheet action="write" filename="#theFile#" name="theSheet"
    sheet=1 sheetname="courses" overwrite=true>
```

SpreadsheetDeleteRows

説明
Excel スプレッドシートオブジェクトの複数の行からすべてのデータを削除します。行は削除されません。

戻り値
なし

カテゴリ
Microsoft Office の統合

関数のシンタックス
```
SpreadsheetDeleteRows(spreadsheetObj, range)
```

関連項目
SpreadsheetDeleteColumn、SpreadsheetDeleteColumns、SpreadsheetDeleteRow、SpreadsheetFormatRow、SpreadsheetFormatRows、SpreadsheetShiftRows

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
spreadsheetObj	行を削除する Excel スプレッドシートオブジェクトです。
range	削除対象の行です。次のいずれかの形式の組み合わせを使用します。 • *startRow-endRow*: 単一の範囲の行を削除します。 • *row,row,row...*: 複数の行を個別に削除します。 両方の形式を同時に使用することもできます。例 : 1, 2, 3-5, 7-12。

例
次の例では、スプレッドシートから行 1 および行 5 から 10 を削除します。

```
<!--- Get the spreadsheet data as a query. --->
<cfquery
       name="courses" datasource="cfdocexamples"
       cachedwithin="#CreateTimeSpan(0, 6, 0, 0)#">
       SELECT CORNUMBER,DEPT_ID,CORLEVEL,COURSE_ID,CORNAME,CORDESC,LASTUPDATE
       FROM COURSELIST
</cfquery>

<cfscript>
    ///We need an absolute path, so get the current directory path.
    theFile=GetDirectoryFromPath(GetCurrentTemplatePath()) & courses.xls";
    //Create a new Excel spreadsheet object and add the query data.
    theSheet = SpreadsheetNew("CourseData");

    SpreadsheetAddRows(theSheet,courses);
    //Delete rows 1 and 5 though 10.
    SpreadsheetDeleteRows(theSheet,"1,5-10");
</cfscript>

<!--- Write the spreadsheet to a file, replacing any existing file. --->
<cfspreadsheet action="write" filename="#theFile#" name="theSheet"
    sheet=1 sheetname="courses" overwrite=true>
```

SpreadsheetFormatCell

説明
Excel スプレッドシートオブジェクトの単一セルの内容を形式設定します。

戻り値
なし

カテゴリ
Microsoft Office の統合

関数のシンタックス

```
SpreadsheetFormatCell(spreadsheetObj, format, row, column)
```

関連項目

SpreadsheetFormatColumn、SpreadsheetFormatColumns、SpreadsheetFormatRow、
SpreadsheetFormatRowsSpreadsheetShiftRows

履歴

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
spreadsheetObj	形式を設定する Excel スプレッドシートオブジェクトです。
format	形式情報を含む構造体です。詳細については、「使用方法」を参照してください。
row	セルの行番号です。
column	セルの列番号です。

使用方法

format 構造体では、次の値の一部またはすべてを指定できます。

名前	有効な値
alignment	left（デフォルト）、right、center、justify、general、fill および center_selection
bold	ブール値です。デフォルト値は false です。
bottomborder	境界線の形式です。次のいずれかを指定します。 none (デフォルト)、thin、medium、dashed、hair、thick、double、dotted、medium_dashed、dash_dot、medium_dash_dot、dash_dot_dot、medium_dash_dot_dot、slanted_dash_dot
bottombordercolor	カラーのリストについては、color フィールドの説明を参照してください。
color	org.apache.poi.hssf.util.HSSFColor クラスの任意の値です。 black、brown、olive_green、dark_green、dark_teal、dark_blue、indigo、grey_80_percent、orange、dark_yellow、green、teal、blue、blue_grey、grey_50_percent、red、light_orange、lime、sea_green、aqua、light_blue、violet、grey_40_percent、pink、gold、yellow、bright_green、turquoise、dark_red、sky_blue、plum、grey_25_percent、rose、light_yellow、light_green、light_turquoise、light_turquoise、pale_blue、lavender、white、cornflower_blue、lemon_chiffon、maroon、orchid、coral、royal_blue、light_cornflower_blue

名前	有効な値
dataformat	Excel データ形式です。MS Excel でサポートされているほとんどの形式がサポートされています。ビルトイン形式は次のとおりです。 General 0 0.00 #,##0 #,##0.00 ($#,##0_($#,##0) ($#,##0_[Red]($#,##0) ($#,##0.00($#,##0.00) ($#,##0.00_[Red]($#,##0.00) 0% 0.00% 0.00E+00 # ?/? # ??/?? m/d/yy d-mmm-yy d-mmm mmm-yy h:mm AM/PM h:mm:ss AM/PM h:mm h:mm:ss m/d/yy h:mm (#,##0_(#,##0) (#,##0_[Red](#,##0) (#,##0.00_(#,##0.00) (#,##0.00_[Red](#,##0.00) _(*#,##0__(*(#,##0_(* \-__(@_) _($*#,##0__($*(#,##0_($* \-\__(@_) _(*#,##0.00__(*(#,##0.00_(*\-\??__(@_) _($*#,##0.00__($*(#,##0.00_($*\-\??__(@_) mm:ss [h]:mm:ss mm:ss.0 ##0.0E+0 @
fgcolor	カラーのリストについては、color フィールドの説明を参照してください。
fillpattern	次のいずれかです。 big_spots (デフォルト)、squares、nofill、solid_foreground、fine_dots、alt_bars、sparse_dots、thick_horz_bands、thick_vert_bands、thick_backward_diag、thick_forward_diag、diamonds、less_dots、least_dots
font	有効なシステムフォント名です。
fontsize	整数ポイント値です。
hidden	ブール値です。デフォルト値は false です。
indent	デフォルト文字スペースの正の整数値です。
italic	値を指定する必要はありません。
leftborder	境界線の形式です。有効な値については、「bottomborder」を参照してください。
leftbordercolor	カラーのリストについては、color フィールドの説明を参照してください。
locked	ブール値です。デフォルト値は false です。
rightborder	境界線の形式です。有効な値については、「bottomborder」を参照してください。
rightbordercolor	カラーのリストについては、color フィールドの説明を参照してください。
rotation	角度の整数値です。値の範囲は、-90 から 90 です。
strikeout	値を指定する必要はありません。
textwrap	ブール値です。デフォルト値は false です。
topborder	境界線の形式です。有効な値については、「bottomborder」を参照してください。

名前	有効な値
topbordercolor	カラーのリストについては、color フィールドの説明を参照してください。
verticalalignment	vertical_top、vertical_bottom、vertical_center、vertical_justify のいずれかです。 次に例を示します。 <cfscript>SpreadsheetFormatCellRange(theSheet,{verticalalignment="VERTICAL_TOP"}, 3,4,30,10);</cfscript>
underline	ブール値です。デフォルト値は false です。

ColdFusion 9.0.1 で行われた機能強化

SpreadSheetformatcell を使用するときに、あらかじめセルの書式設定を行えます。次に例を示します。

```
<cfscript>
sheet= SpreadSheetNew();
Spreadsheetformatcell(sheet,{dataformat="@"},1,1);
spreadsheetSetCellValue(sheet,'000006534',1,1);
</cfscript>
```

これで、あらかじめセルの書式設定が行われ、指定したデータが設定されます。

例

次の例では、シートを作成し、行 3 列 4 のセルのシンプルな形式を設定し、ファイルに結果を書き込みます。

```
<!--- Get the spreadsheet data as a query. --->
<cfquery
       name="courses" datasource="cfdocexamples"
       cachedwithin="#CreateTimeSpan(0, 6, 0, 0)#">
       SELECT CORNUMBER,DEPT_ID,CORLEVEL,COURSE_ID,CORNAME,CORDESC,LASTUPDATE
       FROM COURSELIST
</cfquery>

<cfscript>
    ///We need an absolute path, so get the current directory path.
       theFile=GetDirectoryFromPath(GetCurrentTemplatePath()) & "courses.xls";
    //Create a new Excel spreadsheet object and add the query data.
    theSheet = SpreadsheetNew("CourseData");
    SpreadsheetAddRows(theSheet,courses);
    // Define a format for the cell.
    format1-SructNew();
    format1.font="serif";
    format1.fontsize="12";
    format1.color="dark_green";
    format1.bold="true";
    format1.alignment="center";
    SpreadsheetFormatCell(theSheet,format1,3,4);
</cfscript>

<!--- Write the spreadsheet to a file, replacing any existing file. --->
<cfspreadsheet action="write" filename="#theFile#" name="theSheet"
    sheet=1 sheetname="courses" overwrite=true>
```

次の例では、dataformat の使用方法を示します。

```
<cfset a = spreadsheetnew()>
<cfset format = structnew()>
<cfset format.dataformat = "0.00">
<cfset spreadsheetaddrow(a,"1,2,3,4",2,1)>
<cfset spreadsheetformatrow(a,format,2)>
<cfset format.dataformat = "0.00%">
<cfset spreadsheetaddrow(a,"1,2,3,4",4,1)>
<cfset spreadsheetformatrow(a,format,4)>
<cfset format.dataformat = "0.00E+00">
<cfset spreadsheetaddrow(a,".00000000000001",5,1)>
<cfset spreadsheetformatrow(a,format,5)>
<cfset format.dataformat = "## ??/??">
<cfset spreadsheetaddrow(a,"3.33",7,1)>
<cfset spreadsheetformatrow(a,format,7)>
<cfset format.dataformat = "m/d/yy">
<cfset spreadsheetaddrow(a,"01/06/09",8,1)>
<cfset spreadsheetformatrow(a,format,8)>
<cfset format.dataformat = "##,##0.00">
<cfset spreadsheetaddrow(a,"2100000",13,1)>
<cfset spreadsheetformatrow(a,format,13)>
<cfset format.dataformat = " (##,##0_);(##,##0) ">
<cfset spreadsheetaddrow(a,"-300",14,1)>
<cfset spreadsheetformatrow(a,format,14)>
<cfspreadsheet action="write" filename="#expandpath('.')#/test.xls" name="a" overwrite="true">
```

SpreadsheetFormatColumn

説明
Excel スプレッドシートオブジェクトの単一列の内容を形式設定します。

戻り値
なし

カテゴリ
Microsoft Office の統合

関数のシンタックス
```
SpreadsheetFormatColumn(spreadsheetObj, format, column)
```

関連項目
SpreadsheetFormatCell、SpreadsheetFormatColumns、SpreadsheetFormatRow、SpreadsheetFormatRows

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
spreadsheetObj	形式を設定する Excel スプレッドシートオブジェクトです。
format	形式情報を含む構造体です。詳細については、SpreadsheetFormatCell を参照してください。
column	列番号です。

例

次の例では、シートを作成し、列 5 の形式を設定し、ファイルに結果を書き込みます。

```
<!--- Get the spreadsheet data as a query. --->
<cfquery
       name="courses" datasource="cfdocexamples"
       cachedwithin="#CreateTimeSpan(0, 6, 0, 0)#">
       SELECT CORNUMBER,DEPT_ID,CORLEVEL,COURSE_ID,CORNAME,CORDESC,LASTUPDATE
       FROM COURSELIST
</cfquery>

<cfscript>
    ///We need an absolute path, so get the current directory path.
       theFile=GetDirectoryFromPath(GetCurrentTemplatePath()) & "courses.xls";
    //Create a new Excel spreadsheet object and add the query data.
    theSheet = SpreadsheetNew("CourseData");
    SpreadsheetAddRows(theSheet,courses);
    // Define a format for the column.
    format1-SructNew()
    format1.font="Courier";
    format1.fontsize="10";
    format1.color="dark_blue;";
    format1.italic="true";
    format1.bold="true";
    format1.alignment="left";
    format1.textwrap="true";
    format1.fgcolor="pale_blue";
    format1.bottomborder="dotted";
    format1.bottombordercolor="blue_grey";
    format1.leftborder="thick";
    format1.leftbordercolor="blue_grey";
    format1.rightborder="thick";
    format1.rightbordercolor="blue_grey";
    SpreadsheetFormatColumn(theSheet,format1,5);
</cfscript>

<!--- Write the spreadsheet to a file, replacing any existing file. --->
<cfspreadsheet action="write" filename="#theFile#" name="theSheet"
    sheet=1 sheetname="courses" overwrite=true>
```

SpreadsheetFormatCellRange

説明

指定範囲内のセルの書式設定を行います。

戻り値

なし

カテゴリ

Microsoft Office の統合

関数のシンタックス

SpreadsheetFormatCellRange (spreadsheetObj, format, startRow, startColumn, endRow, endColumn)

関連項目

SpreadsheetFormatCell、SpreadsheetFormatColumns、SpreadsheetFormatRow、SpreadsheetFormatRows

履歴
ColdFusion 9.0.1: この関数が追加されました。この関数を使用するときには、セルの事前書式設定がサポートされます。

パラメータ

パラメータ	説明
spreadsheetObj	セルの書式設定を行う Excel スプレッドシートオブジェクトです。
format	書式設定の情報が含まれる構造体です。
startRow	書式設定を行う最初の行の行番号です。
startColumn	書式設定を行う最初の列の列番号です。
endRow	書式設定を行う最後の行の行番号です。
endColumn	書式設定を行う最後の列の列番号です。

例
```
<!--- Get the spreadsheet data as a query. --->
<cfquery
    name="courses" datasource="cfdocexamples"
    cachedwithin="#CreateTimeSpan(0, 6, 0, 0)#">
    SELECT CORNUMBER,DEPT_ID,CORLEVEL,COURSE_ID,CORNAME,CORDESC,LASTUPDATE
    FROM COURSELIST
</cfquery>
<cfscript>
    ///We need an absolute path, so get the current directory path.
    theFile=GetDirectoryFromPath(GetCurrentTemplatePath()) & "courses.xls";
    //Create a new Excel spreadsheet object and add the query data.
    theSheet = SpreadsheetNew("CourseData");
    SpreadsheetAddRows(theSheet,courses);
    // Define a format for the column.
    format1=SructNew()
    format1.font="Courier";
    format1.fontsize="10";
    format1.color="dark_blue;";
    format1.italic="true";
    format1.bold="true";
    format1.alignment="left";
    SpreadsheetFormatCellRange(theSheet,format1, 3,4,30,10);
</cfscript>
<!--- Write the spreadsheet to a file, replacing any existing file. --->
<cfspreadsheet action="write" filename="#theFile#" name="theSheet"
sheet=1 sheetname="courses" overwrite=true>
```

SpreadsheetFormatColumns

説明
Excel スプレッドシートオブジェクトの複数列の内容を形式設定します。

戻り値
なし

カテゴリ
Microsoft Office の統合

関数のシンタックス

```
SpreadsheetFormatColumns(spreadsheetObj, format, columns)
```

関連項目

SpreadsheetFormatCell、SpreadsheetFormatColumn、SpreadsheetFormatRow、SpreadsheetFormatRows

履歴

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
spreadsheetObj	形式を設定する Excel スプレッドシートオブジェクトです。
format	形式情報を含む構造体です。詳細については、SpreadsheetFormatCell を参照してください。
columns	形式設定する列です。次のいずれかの形式になります。 • *startColumn-endColumn*: 単一の範囲の列を形式設定します。 • *column,column,column...*: 複数の列を個別に形式設定します。

例

次の例では、シートを作成し、列 1 から 5 の形式を設定し、ファイルに結果を書き込みます。

```
<!--- Get the spreadsheet data as a query. --->
<cfquery
       name="courses" datasource="cfdocexamples"
       cachedwithin="#CreateTimeSpan(0, 6, 0, 0)#">
       SELECT CORNUMBER,DEPT_ID,CORLEVEL,COURSE_ID,CORNAME,CORDESC,LASTUPDATE
       FROM COURSELIST
</cfquery>

<cfscript>
    ///We need an absolute path, so get the current directory path.
       theFile=GetDirectoryFromPath(GetCurrentTemplatePath()) & "courses.xls";
    //Create a new Excel spreadsheet object and add the query data.
    theSheet = SpreadsheetNew("CourseData");
    SpreadsheetAddRows(theSheet,courses);
    // Define a format for the column.
    format1-SructNew()
    format1.font="Courier";
    format1.fontsize="10";
    format1.color="dark_blue;";
    format1.italic="true";
    format1.bold="true";
    format1.alignment="left";
    format1.textwrap="true";
    format1.fgcolor="pale_blue";
    format1.bottomborder="dotted";
    format1.bottombordercolor="blue_grey";
    format1.leftborder="thick";
    format1.leftbordercolor="blue_grey";
    format1.rightborder="thick";
    format1.rightbordercolor="blue_grey";
    SpreadsheetFormatColumns(theSheet,format1,"1-5");
</cfscript>

<!--- Write the spreadsheet to a file, replacing any existing file. --->
<cfspreadsheet action="write" filename="#theFile#" name="theSheet"
    sheet=1 sheetname="courses" overwrite=true>
```

SpreadsheetFormatRow

説明

Excel スプレッドシートオブジェクトの単一行の内容を形式設定します。

戻り値

なし

カテゴリ

Microsoft Office の統合

関数のシンタックス

```
SpreadsheetFormatRow(spreadsheetObj, format, row)
```

関連項目

その他の Spreadsheet* 関数

SpreadsheetFormatCell、SpreadsheetFormatColumn、SpreadsheetFormatColumns、SpreadsheetFormatRows

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
spreadsheetObj	形式を設定する Excel スプレッドシートオブジェクトです。
format	形式情報を含む構造体です。詳細については、SpreadsheetFormatColumn を参照してください。
row	行番号です。

例
次の例では、シートを作成し、行 1、3、および 5 の形式を設定し、ファイルに結果を書き込みます。

```
<!--- Get the spreadsheet data as a query. --->
<cfquery
       name="courses" datasource="cfdocexamples"
       cachedwithin="#CreateTimeSpan(0, 6, 0, 0)#">
       SELECT CORNUMBER,DEPT_ID,CORLEVEL,COURSE_ID,CORNAME,CORDESC,LASTUPDATE
       FROM COURSELIST
</cfquery>

<cfscript>
    ///We need an absolute path, so get the current directory path.
       theFile=GetDirectoryFromPath(GetCurrentTemplatePath()) & "courses.xls";
    //Create a new Excel spreadsheet object and add the query data.
    theSheet = SpreadsheetNew("CourseData");
    SpreadsheetAddRows(theSheet,courses);
    // Define a format for the column.
    format1-SructNew()
    format1.font="Courier";
    format1.fontsize="10";
    format1.color="dark_blue;";
    format1.italic="true";
    format1.bold="true";
    format1.alignment="left";
    format1.textwrap="true";
    format1.fgcolor="pale_blue";
    format1.bottomborder="thick";
    format1.bottombordercolor="blue_grey";
    format1.topbordercolor="blue_grey";
    format1.topborder="thick";
    format1.leftborder="dotted";
    format1.leftbordercolor="blue_grey";
    format1.rightborder="dotted";
    format1.rightbordercolor="blue_grey";
    SpreadsheetFormatRow(theSheet,format1,"5");
</cfscript>

<!--- Write the spreadsheet to a file, replacing any existing file. --->
<cfspreadsheet action="write" filename="#theFile#" name="theSheet"
    sheet=1 sheetname="courses" overwrite=true>
```

SpreadsheetFormatRows

説明
Excel スプレッドシートオブジェクトの複数行の内容を形式設定します。

戻り値
なし

カテゴリ
Microsoft Office の統合

関数のシンタックス
```
SpreadsheetFormatRows(spreadsheetObj, format, rows)
```

関連項目
SpreadsheetFormatCell、SpreadsheetFormatColumn、SpreadsheetFormatColumns、SpreadsheetFormatRow

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
spreadsheetObj	形式を設定する Excel スプレッドシートオブジェクトです。
format	形式情報を含む構造体です。詳細については、SpreadsheetFormatColumn を参照してください。
row	形式設定する行です。次のいずれかの形式になります。 • *startRow-endRow*: 単一の範囲の行を削除します。 • *row,row,row...*: 複数の行を個別に削除します。 これらの形式を同時に使用することもできます。例 : 1-5, 6, 7, 9-12。

例
次の例では、シートを作成し、行 1、3、および 5 の形式を設定し、ファイルに結果を書き込みます。

```
<!--- Get the spreadsheet data as a query. --->
<cfquery
       name="courses" datasource="cfdocexamples"
       cachedwithin="#CreateTimeSpan(0, 6, 0, 0)#">
       SELECT CORNUMBER,DEPT_ID,CORLEVEL,COURSE_ID,CORNAME,CORDESC,LASTUPDATE
       FROM COURSELIST
</cfquery>

<cfscript>
    ///We need an absolute path, so get the current directory path.
       theFile=GetDirectoryFromPath(GetCurrentTemplatePath()) & "courses.xls";
    //Create a new Excel spreadsheet object and add the query data.
    theSheet = SpreadsheetNew("CourseData");
    SpreadsheetAddRows(theSheet,courses);
    // Define a format for the column.
    format1-SructNew()
    format1.font="Courier";
    format1.fontsize="10";
    format1.color="dark_blue;";
    format1.italic="true";
    format1.bold="true";
    format1.alignment="left";
    format1.textwrap="true";
    format1.fgcolor="pale_blue";
    format1.bottomborder="thick";
    format1.bottombordercolor="blue_grey";
    format1.topbordercolor="blue_grey";
    format1.topborder="thick";
    format1.leftborder="dotted";
    format1.leftbordercolor="blue_grey";
    format1.rightborder="dotted";
    format1.rightbordercolor="blue_grey";
    SpreadsheetFormatRows(theSheet,format1,"1,3,5");
</cfscript>

<!--- Write the spreadsheet to a file, replacing any existing file. --->
<cfspreadsheet action="write" filename="#theFile#" name="theSheet"
    sheet=1 sheetname="courses" overwrite=true>
```

SpreadsheetGetCellComment

説明

Excel スプレッドシートオブジェクトセルに関するコメントを、形式情報を持つ構造体として取得するか、またはそのオブジェクトに関するすべてのコメントを取得します。

戻り値

パラメータに行および列が含まれている場合は、特定のセルに関するコメント情報を含む構造体が返されます。この関数に spreadsheetObj パラメータのみが指定されている場合は、各コメントの構造体を含む配列が返されます。各構造体には次の情報が含まれています。

フィールド	内容
Author	コメントの作成者の名前を含む文字列です。

フィールド	内容
Column	セルの列番号です。
Comment	コメント文字列を含む文字列です。
Row	セルの行番号です。

カテゴリ
Microsoft Office の統合

関数のシンタックス
```
SpreadsheetGetCellComment(spreadsheetObj[, row, column])
```

関連項目
SpreadsheetFormatCell、SpreadsheetGetCellFormula、SpreadsheetGetCellValue、SpreadsheetMergeCells、SpreadsheetSetCellComment、SpreadsheetSetCellFormula、SpreadsheetSetCellValue

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
spreadsheetObj	コメントの取得元となる Excel スプレッドシートオブジェクトです。
row	コメントの取得元となるセルの行番号です。
column	コメントの取得元となるセルの列番号です。

例
次の例では、行 3 列 5 のセルのコメントを設定し、取得します。

```
<cfscript>
    ///We need an absolute path, so get the current directory path.
       theFile=GetDirectoryFromPath(GetCurrentTemplatePath()) & "comment.xls";
    //Create an Excel spreadsheet object.
    theSheet = SpreadsheetNew();
    // Define a cell comment.

    comment1.anchor="1,1,15,20";
    comment1.author="Adobe Systems";
    comment1.bold="true";
    comment1.color="lavender";
    comment1.comment="This is the cell in row three, column 5 (E).";
    comment1.fillcolor="yellow";
    comment1.font="Courier";
    comment1.horizontalalignment="left";
    comment1.linestyle="dashsys";
    comment1.size="10";
    comment1.verticalalignment="top";
    //Set the comment.
    SpreadsheetSetCellComment(theSheet,comment1,3,5);
    //Get the comment from the Excel spreadsheet object.
    theComment=SpreadsheetGetCellComment(theSheet,3,5);
</cfscript>
<cfoutput>
Row,Column: #theComment.row#,#theComment.column#<br />
Author: #theComment.author#<br />
Comment: #theComment.comment#
</cfoutput>
<!--- Write the spreadsheet to a file, replacing any existing file. --->
<cfspreadsheet action="write" filename="#theFile#" name="theSheet"
    sheet=1 sheetname="courses" overwrite=true>
```

SpreadsheetGetCellFormula

説明

Excel スプレッドシートオブジェクトセルの式またはそのオブジェクトのすべての式を取得します。

戻り値

パラメータに行および列が含まれている場合は、式を含む文字列が返されます。この関数に spreadsheetObj パラメータのみが指定されている場合は、各式の構造体を含む配列が返されます。

カテゴリ

Microsoft Office の統合

関数のシンタックス

```
SpreadsheetGetCellFormula(spreadsheetObj[, row, column])
```

関連項目

SpreadsheetGetCellComment、SpreadsheetFormatCell、SpreadsheetGetCellValue、SpreadsheetMergeCells、SpreadsheetSetCellComment、SpreadsheetSetCellFormula、SpreadsheetSetCellValue

履歴

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
spreadsheetObj	式の取得元となる Excel スプレッドシートオブジェクトです。
row	式の取得元となるセルの行番号です。
column	式の取得元となるセルの列番号です。

使用方法

この関数に spreadsheetObj パラメータのみを指定すると、次の内容の構造体の配列が返されます。この配列には、式を含むセルそれぞれに対して 1 つのエントリがあります。

フィールド	有効な値
formula	セルの式です。
row	セルの行番号です。
column	セルの列番号です。

例

次の例では、セルの式を設定し、セルの式および値を取得します。

```
<cfscript>
    //Create a new Excel spreadsheet object.
    theSheet=SpreadsheetNew();
    //Set the values of column 3 rows 1-10 to the row number.
    for (i=1; i<= 10; i=i+1)
        SpreadsheetSetCellValue(theSheet,i,i,3);
    //Set the formula for the cell in row 11 column 3 to be the sum of
    //Columns 1-10.
    SpreadsheetSetCellFormula(theSheet,"SUM(C1:C10)",11,3);
    //Get the formula from the Excel spreadsheet object.
    theValue=SpreadsheetGetCellFormula(theSheet,11,3);
    //Get the value of row 11 column 5 from the Excel spreadsheet object.
    theValue=SpreadsheetGetCellValue(theSheet,11,3);
</cfscript>

<cfoutput>
Row 11, Column 3 value: #SpreadsheetGetCellValue(theSheet,11,3)#<br />
Row 11, Column 3 formula: #SpreadsheetGetCellFormula(theSheet,11,3)#<br />
</cfoutput>
```

SpreadsheetGetCellValue

説明

Excel スプレッドシートオブジェクトセルの値を取得します。

戻り値

セルの値を含む文字列です。

カテゴリ

Microsoft Office の統合

関数のシンタックス

```
SpreadsheetGetCellValue(spreadsheetObj, row, column)
```

関連項目

SpreadsheetGetCellComment、SpreadsheetFormatCell、SpreadsheetGetCellFormula、SpreadsheetMergeCells、SpreadsheetSetCellComment、SpreadsheetSetCellFormula、SpreadsheetSetCellValue

履歴

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
spreadsheetObj	値の取得元となる Excel スプレッドシートオブジェクトです。
row	式の取得元となるセルの行番号です。
column	式の取得元となるセルの列番号です。

例

次の例では、Excel スプレッドシートオブジェクトを作成して、行 3 列 5 のセルの値を 365 に設定し、その値を取得して表示します。

```
<cfscript>
    //Create a new Excel spreadsheet object.
    theSheet=SpreadsheetNew();
    //Set the value of the cell at row 3 column 5.
    SpreadsheetSetCellValue(theSheet,365,3,5);
    //Get the value from the Excel spreadsheet object.
    theValue=SpreadsheetGetCellValue(theSheet,3,5);
    WriteOutput("The value of column 5 row 3 is: " & theValue);
</cfscript>
```

SpreadsheetInfo

説明

Excel スプレッドシートオブジェクトのプロパティを取得します。

戻り値

次のいずれかのスプレッドシートプロパティが返されます。

- AUTHOR
- CATEGORY
- COMMENTS
- CREATIONDATE
- LASTEDITED
- LASTAUTHOR
- LASTSAVED
- KEYWORDS

- MANAGER
- COMPANY
- SUBJECT
- TITLE
- SHEETS
- SHEETNAMES
- SPREADSHEETTYPES

カテゴリ
Microsoft Office の統合

関数のシンタックス
`SpreadsheetInfo(spreadsheetobj)`

関連項目
SpreadsheetAddColumn、SpreadsheetAddImage、SpreadsheetAddRow、SpreadsheetDeleteRow、SpreadsheetDeleteRows、SpreadsheetFormatRow、SpreadsheetFormatRows、SpreadsheetShiftRows

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
spreadsheetobj	値の取得元となる Excel スプレッドシートオブジェクトです。

使用方法
この関数は、Microsoft Office Excel 2007 および Microsoft Office 2003 でサポートされています。

例
```
<cfspreadsheet action="read" src="#dirname#SingleSheet.xls" name="SpreadsheetObj"  >
    <cfset info = SpreadSheetInfo(SpreadsheetObj)>
    <cfoutput> AUTHOR : #info.author#<br> </cfoutput>
    <cfoutput> Creation Date : #info.creationdate#<br> </cfoutput>
    <cfoutput> LAST AUTHOR : #info.lastauthor#<br> </cfoutput>
    <cfoutput> SHEETS : #info.sheets#<br> </cfoutput>
    <cfoutput> SPREADSHEETTYPE : #info.SPREADSHEETTYPE#<br> </cfoutput>
    <cfoutput>SUBJECT : #info.SUBJECT#<br></cfoutput>
    <cfoutput>TITLE : #info.TITLE#<br></cfoutput>
```

SpreadsheetMergeCells

説明
Excel スプレッドシートオブジェクトの 2 つ以上のセルから成る長方形ブロックをマージします。

戻り値
なし

カテゴリ
Microsoft Office の統合

関数のシンタックス
```
SpreadsheetMergeCells(spreadsheetObj, startRow, endRow, startColumn, endColumn)
```

関連項目
SpreadsheetGetCellComment、SpreadsheetFormatCell、SpreadsheetGetCellFormula、SpreadsheetGetCellValue、SpreadsheetSetCellComment、SpreadsheetSetCellFormula、SpreadsheetSetCellValue

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
spreadsheetObj	マージするセルを含む Excel スプレッドシートオブジェクトです。
startRow	マージする最初の行の行番号です。
endRow	マージする最後の行の行番号です。
startColumn	マージする最初のセルの列番号です。
endColumn	マージする最後のセルの列番号です。

使用方法
この関数を使用して 2 つのセルをマージする場合、デフォルトではマージされたセルには、スプレッドシートの左側にあるセルの値が表示されます。たとえば、セル (20,3) とセル (20,4) をマージすると、セル (20, 3) の値が表示されます。セル (20, 3) が空白の場合、マージ後のセルも空白になります。

例
次の例では、Excel スプレッドシートオブジェクトの行 1 から 3 のセル 4 から 6 をマージします。マージされたセルにテキストを配置し、マージ結果を確認できるよう、ファイルにシートを保存します。

```
<cfscript>
    ///We need an absolute path, so get the current directory path.
      theFile=GetDirectoryFromPath(GetCurrentTemplatePath()) & "mergecells.xls";
    //Create a new Excel spreadsheet object and add the query data.
    theSheet = SpreadsheetNew("CourseData");
    //Merges cells 4-6 in the first three rows of the Excel spreadsheet object.
    SpreadsheetMergeCells(theSheet,1,3,4,6);
    //Set the value of the merged cell.
    SpreadsheetSetCellValue(theSheet,"Columns 4-6 of rows 1-3 are merged",1,4);
</cfscript>

<!--- Write the spreadsheet to a file, replacing any existing file. --->
<cfspreadsheet action="write" filename="#theFile#" name="theSheet" overwrite=true>
```

SpreadsheetNew

説明
Excel ドキュメントの単一シートを表す ColdFusionExcel スプレッドシートオブジェクトを作成します。

戻り値
ColdFusionExcel スプレッドシートオブジェクト。

カテゴリ
Microsoft Office の統合

関数のシンタックス
```
SpreadsheetNew([sheetName, xmlformat])
```

関連項目
その他すべての Spreadsheet* 関数。「Microsoft Office の統合」のリストを参照してください。

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
sheetName	Excel スプレッドシートオブジェクトに割り当てるシート名を含む文字列です。
xmlformat	ブール値です。 True または Yes: Microsoft Office Excel 2007 でサポートされている .xlsx ファイルを作成します。 False または No: .xls ファイルを作成します。

使用方法
この関数は、Microsoft Office Excel 2007 をサポートしています。

デフォルトのワークシート名で設定なしの .xls スプレッドシートオブジェクトを作成するには、次のようなコードを使用します。

```
<cfset SpreadsheetObj = spreadsheetNew()>
```

ワークシート名 "mySheet" を指定して、設定なしの .xls スプレッドシートオブジェクトを作成するには、次のようなコードを使用します。

```
<cfset SpreadsheetObj = spreadsheetNew("mySheet")>
```

Microsoft Office Excel 2007 (.xlsx) でサポートされているスプレッドシートオブジェクトを作成するには、次のようなコードを使用します。

```
<cfset SpreadsheetObj = spreadsheetNew("true")>
```

```
<cfset SpreadsheetObj = spreadsheetNew("mysheet","yes")>
```

注意："true" または "yes" を指定して、.xlsx ファイルを作成できます。

例
次の例では、シート名が Expenses の Excel スプレッドシートオブジェクトを作成し、セルの値を設定して、結果をファイルに保存します。

```
<cfscript>
    ///We need an absolute path, so get the current directory path.
      theFile=GetDirectoryFromPath(GetCurrentTemplatePath()) & "newSpreadsheet.xls";
    //Create a new Excel spreadsheet object.
    theSheet = SpreadsheetNew("Expenses");
    //Set the value a cell.
    SpreadsheetSetCellValue(theSheet,"365",1,4);
</cfscript>

<!--- Write the spreadsheet to a file, replacing any existing file. --->
<cfspreadsheet action="write" filename="#theFile#" name="theSheet" overwrite=true>
```

SpreadsheetRead

説明
スプレッドシートファイルからシートを読み取り、そのシートを ColdFusion スプレッドシートオブジェクトに保存します。

戻り値
スプレッドシートオブジェクトを返します。

カテゴリ
Microsoft Office の統合

関数のシンタックス
```
SpreadSheetRead(fileName [, sheetName|sheet])
```

関連項目
SpreadsheetWrite、SpreadsheetAddColumn、SpreadsheetAddImage、SpreadsheetAddRow、SpreadsheetDeleteRow、SpreadsheetDeleteRows、SpreadsheetFormatRow、SpreadsheetFormatRows、SpreadsheetShiftRows

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
fileName	スプレッドシートファイルのパスを指定する文字列です。
sheetName	シートの名前を指定するオプションのパラメータです。sheet または sheetName を指定できます。
sheet	シートの番号を指定するオプションのパラメータです。sheet または sheetName を指定できます。

使用方法
この関数は、複数のシートを含む Excel ファイルを読み取る場合に使用します。

例
```
<cfscript>
    a = SpreadSheetRead("C:\Files\Report.xls","Annual Report")
</cfscript>
```

SpreadsheetReadBinary

説明
スプレッドシートの内容を読み取り、保管して、バイト配列とし返します。

戻り値
cfcontent タグを使用して保管されているスプレッドシート情報のバイト配列が返されます。

カテゴリ
Microsoft Office の統合

関数のシンタックス
```
SpreadsheetReadBinary(spreadsheetobj)
```

関連項目
SpreadsheetAddColumn、SpreadsheetAddImage、SpreadsheetAddRow、SpreadsheetDeleteRow、SpreadsheetDeleteRows、SpreadsheetFormatRow、SpreadsheetFormatRows、SpreadsheetShiftRows

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
spreadSheetObject	読み取り対象の Excel スプレッドシートオブジェクトです。

使用方法

例
```
<cfheader name="Content-Disposition" value="inline; filename=test.xls">
<cfset a = spreadhsheetnew()>
<cfset spreadsheetAddRow(a,"a,b,c")>
<!---You can do all the processing--->
<cfset bin = spreadsheetReadBinary(a)>
<cfcontent type="application/vnd-ms.excel" variable="#bin#" reset="true">
```

SpreadsheetRemoveSheet

説明
スプレッドシートを削除します。

戻り値
なし

カテゴリ
Microsoft Office の統合

関数のシンタックス
SpreadsheetRemoveSheet (spreadsheetObj, sheetname)

関連項目
SpreadsheetSetActiveSheetNumber、SpreadsheetAddColumn、SpreadsheetAddImage、SpreadsheetAddRow、SpreadsheetDeleteRow、SpreadsheetDeleteRows、SpreadsheetFormatRow、SpreadsheetFormatRows、SpreadsheetShiftRows

履歴
ColdFusion 9.0.1: この関数が追加されました。

パラメータ

パラメータ	説明
spreadsheetObj	シートから削除する Excel スプレッドシートオブジェクトです。
sheetname	削除するシートの名前です。

例
```
<cfset spreadsheetVar= spreadSheetNew("New")>
<cfset spreadsheetCreateSheet(spreadsheetVar,"A")>
<cfset spreadsheetCreateSheet(spreadsheetVar,"B")>
<cfspreadsheet action="write" filename="#dirname#mySpreadSheet.xls"  name="spreadsheetVar"
overwrite="true" >
<cfspreadsheet action="read" src="#dirname#mySpreadSheet.xls" name="spreadSheetVar" >
<cfset spreadsheetRemoveSheet(spreadsheetVar,"B")>
<cfspreadsheet action="write" filename="#dirname#mySpreadSheet.xls"  name="spreadsheetVar"
overwrite="true" >
```

SpreadsheetSetActiveSheet

説明
指定したシートをアクティブなシートとして設定します。

戻り値
なし

カテゴリ
Microsoft Office の統合

関数のシンタックス
```
SpreadsheetSetActiveSheet(spreadsheetobj, sheetname)
```

関連項目
SpreadsheetSetActiveSheetNumber、SpreadsheetAddColumn、SpreadsheetAddImage、SpreadsheetAddRow、SpreadsheetDeleteRow、SpreadsheetDeleteRows、SpreadsheetFormatRow、SpreadsheetFormatRows、SpreadsheetShiftRows

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
spreadsheetobj	分割ペインの追加先となる Excel スプレッドシートオブジェクトです。
sheetname	アクティブとして設定する必要があるスプレッドシートです。

使用方法
アクティブとして設定できるシートは一度に 1 つのみです。したがって、シート操作は特定のシートに制限されます。他のシートを操作する場合、そのシートをアクティブとして設定する必要があります。

例
次の例では、シートを切り替えて操作を実行する方法を示します。

```
<!--- Get the spreadsheet data as a query. --->
<cfquery
       name="courses" datasource="cfdocexamples"
       cachedwithin="#CreateTimeSpan(0, 6, 0, 0)#">
       SELECT CORNUMBER,DEPT_ID,COURSE_ID,CORNAME
       FROM COURSELIST
</cfquery>

<cfscript>
    ///We need an absolute path, so get the current directory path.
    theFile=GetDirectoryFromPath(GetCurrentTemplatePath()) & courses.xls";
    //Create a new Excel spreadsheet object and add the query data.
    theSheet = SpreadsheetNew("CourseData");

    SpreadsheetAddRows(theSheet,courses);
    //Create a new sheet.
    SpreadsheetCreateSheet (theSheet, "EvaluationSheet");
    //Set the sheet as active.
    SpreadsheetSetActiveSheet (theSheet, "EvaluationSheet");
    //Add a new row to the sheet.
    SpreadsheetAddRows(theSheet,courses);
</cfscript>

<!--- Write the spreadsheet to a file, replacing any existing file. --->
<cfspreadsheet action="write" filename="#theFile#" name="theSheet"
    sheet=1 sheetname="courses" overwrite=true>
```

SpreadsheetSetActiveSheetNumber

説明
指定した sheetnumber をアクティブなシートとして設定します。

戻り値
なし

カテゴリ
Microsoft Office の統合

関数のシンタックス

```
SpreadsheetSetActiveSheetNumber(spreadsheetobj, sheetnumber)
```

関連項目
SpreadsheetSetActiveSheet、SpreadsheetAddColumn、SpreadsheetAddImage、SpreadsheetAddRow、SpreadsheetDeleteRow、SpreadsheetDeleteRows、SpreadsheetFormatRow、SpreadsheetFormatRows、SpreadsheetShiftRows

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
spreadsheetobj	分割ペインの追加先となる Excel スプレッドシートオブジェクトです。
sheetnumber	アクティブとして設定する必要があるスプレッドシート番号です。

使用方法
アクティブとして設定できるシートは一度に 1 つのみです。したがって、シート操作は特定のシートに制限されます。他のシートを操作する場合、そのシートをアクティブとして設定する必要があります。

例
次の例では、シートを切り替えて操作を実行する方法を示します。

```
<!--- Get the spreadsheet data as a query. --->
<cfquery
       name="courses" datasource="cfdocexamples"
       cachedwithin="#CreateTimeSpan(0, 6, 0, 0)#">
       SELECT CORNUMBER,DEPT_ID,COURSE_ID,CORNAME
       FROM COURSELIST
</cfquery>
<cfset theFile=GetDirectoryFromPath(GetCurrentTemplatePath()) & courses.xls">
<cfspreadsheet action="read" name=theSheet src="#theFile#" sheet="1">

<cfscript>
    SpreadsheetAddRows(theSheet,courses);
    //Set the sheetnumber 2 as active.
    SpreadsheetSetActiveSheetNumber (theSheet, 2);
    //Add a new row to the sheet 2.
    SpreadsheetAddRows(theSheet,courses);
</cfscript>

<!--- Write the spreadsheet to a file, replacing any existing file. --->
<cfspreadsheet action="write" filename="#theFile#" name="theSheet"
    sheet=1 sheetname="courses" overwrite=true>
```

SpreadsheetSetCellComment

説明
Excel スプレッドシートオブジェクトセルに関するコメントを指定します。

戻り値
なし

カテゴリ
Microsoft Office の統合

関数のシンタックス

```
SpreadsheetSetCellComment(spreadsheetObj, comment, row, column)
```

関連項目

SpreadsheetGetCellComment、SpreadsheetFormatCell、SpreadsheetGetCellFormula、SpreadsheetGetCellValue、SpreadsheetMergeCells、SpreadsheetSetCellFormula、SpreadsheetSetCellValue

履歴

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
spreadsheetObj	コメントの追加先となる Excel スプレッドシートオブジェクトです。
comment	テキスト、形式、セル内での配置などのコメントを含む構造体です。構造体の内容については、「使用方法」を参照してください。
row	コメントの追加先となるセルの行番号です。
column	コメントの追加先となるセルの列番号です。

使用方法

このコメント構造体には次のフィールドを格納できます。Excel では、デフォルトのフィールド値が判別されます。

フィールド	有効な値
anchor	コメントの位置とサイズを行と列で指定する整数値のカンマ区切りリストです。指定順序は、上の列、左の行、下の列、右の行です。たとえば、"4,8,6,11" は、左上隅が行 4 列 8 で右下隅が行 6 列 11 のコメントを指定します。
author	作成者の名前です。
bold	テキストをボールドにするかどうかを指定するブール値です。
color	テキストカラーです。次の Apache org.apache.poi.hssf.util.HSSFColor クラスの任意の値です。 black、brown、olive_green、dark_green、dark_teal、dark_blue、indigo、grey_80_percent、orange、dark_yellow、green、teal、blue、blue_grey、grey_50_percent、red、light_orange、lime、sea_green、aqua、light_blue、violet、grey_40_percent、pink、gold、yellow、bright_green、turquoise、dark_red、sky_blue、plum、grey_25_percent、rose、light_yellow、light_green、light_turquoise、light_turquoise、pale_blue、lavender、white、cornflower_blue、lemon_chiffon、maroon、orchid、coral、royal_blue、light_cornflower_blue
comment	コメント文字列を含む文字列です。
fillcolor	J2SE v1.4 java.awt.Color クラスのカラー値 : white、lightGray、light_gray、gray、darkGray、dark_gray、black、red、pink、orange、yellow、green、magenta、cyan、blue。(ColdFusion では大文字と小文字が区別されないため、Java クラスで定義されている場合はこれらの値を指定する必要はありません。)
font	有効なシステムフォント名です。
horizontalalignment	テキストの水平方向の整列 : left、center、right、justify、distributed。
italic	テキストをイタリックにするかどうかを指定するブール値です。
linestyle	コメントボックスの上および右の境界線のスタイル : solid、dashsys、dotsys、dashdotsys、dashdotdotsys、dotgel、dashgel、longdashgel、dashdotgel、longdashdotgel、longdashdotdotgel。
linestylecolor	Java カラー値です。
size	テキストのサイズです (単位 : ポイント)。
strikeout	テキストを独立させるかどうかを指定するブール値です。

フィールド	有効な値
underline	テキストにアンダーラインを引くかどうかを指定するブール値です。
verticalalignment	テキストの縦揃え : top、center、bottom、justify、distributed。
visible	テキストを表示するかどうかを指定するブール値です。

例

次の例では、行 3 列 5 のセルのコメントを設定し、取得します。

```
<cfscript>
    ///We need an absolute path, so get the current directory path.
       theFile=GetDirectoryFromPath(GetCurrentTemplatePath()) & "comment.xls";
    //Create an Excel spreadsheet object.
    theSheet = SpreadsheetNew();
    // Define a cell comment.

    comment1-structNew()
    comment1.anchor="0,0,5,8";
    comment1.author="Adobe Systems";
    comment1.bold="true";
    comment1.color="dark_green";
    comment1.comment="This is the cell in row three, column 5 (E).";
    comment1.fillcolor="light_gray";
    comment1.font="Courier";
    comment1.horizontalalignment="left";
    comment1.linestyle="dashsys";
    comment1.size="10";
    comment1.verticalalignment="top";

    //Set the comment.
    SpreadsheetSetCellComment(theSheet,comment1,3,5);
    //Get the comment from the Excel spreadsheet object.
    theComment=SpreadsheetGetCellComment(theSheet,3,5);
</cfscript>

<cfoutput>
Row,Column: #theComment.row#,#theComment.column#<br />
Author: #theComment.author#<br />
Comment: #theComment.comment#
</cfoutput>

<!--- Write the spreadsheet to a file, replacing any existing file. --->
<cfspreadsheet action="write" filename="#theFile#" name="theSheet"
    sheet=1 sheetname="courses" overwrite=true>
```

SpreadsheetSetCellFormula

説明

Excel スプレッドシートオブジェクトセルの式を指定します。

戻り値

なし

カテゴリ

Microsoft Office の統合

関数のシンタックス

```
SpreadsheetSetCellFormula(spreadsheetObj, formula, row, column)
```

関連項目

SpreadsheetGetCellComment、SpreadsheetFormatCell、SpreadsheetGetCellFormula、SpreadsheetGetCellValue、SpreadsheetMergeCells、SpreadsheetSetCellComment、SpreadsheetSetCellValue

履歴

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
spreadsheetObj	コメントの追加先となる Excel スプレッドシートオブジェクトです。
formula	式を含む文字列です。
row	式の追加先となるセルの行番号です。
column	式の追加先となるセルの列番号です。

使用方法

この関数は、特定の入力値を含む既存の値を置き換えます。

例

次の例では、行 11 列 3 のセルの式が列 3 の行 1 から 10 のセルの合計となるよう設定します。

次の例では、セルの式を設定し、セルの式および値を取得します。

```
<cfscript>
    //Create a new Excel spreadsheet object.
    theSheet=SpreadsheetNew();
    //Set the values of column 3 rows 1-10 to the row number.
    for (i=1; i<= 10; i=i+1)
        SpreadsheetSetCellValue(theSheet,i,i,3);
    //Set the fomula for the cell in row 11 column 3 to be the sum of
    //Columns 1-10.
    SpreadsheetSetCellFormula(theSheet,"SUM(C1:C10)",11,3);
    //Get the formula from the Excel spreadsheet object.
    theValue=SpreadsheetGetCellFormula(theSheet,11,3);
    //Get the value of row 11 column 5 from the Excel spreadsheet object.
    theValue=SpreadsheetGetCellValue(theSheet,11,3);
</cfscript>

<cfoutput>
Row 11, Column 3 value: #SpreadsheetGetCellValue(theSheet,11,3)#<br />
Row 11, Column 3 formula: #SpreadsheetGetCellFormula(theSheet,11,3)#<br />
</cfoutput>
```

SpreadsheetSetCellValue

説明

Excel スプレッドシートオブジェクトセルの値を指定します。

カテゴリ
Microsoft Office の統合

関数のシンタックス
```
SpreadsheetSetCellValue(spreadsheetObj, value, row, column)
```

関連項目
SpreadsheetGetCellComment、SpreadsheetFormatCell、SpreadsheetGetCellFormula、SpreadsheetGetCellValue、SpreadsheetMergeCells、SpreadsheetSetCellComment、SpreadsheetSetCellFormula

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
spreadsheetObj	コメントの追加先となる Excel スプレッドシートオブジェクトです。
value	セルの値を含む文字列です。
row	値を設定するセルの行番号です。
column	値を設定するセルの列番号です。

例
次の例では、Excel スプレッドシートオブジェクトを作成して、行 3 列 5 のセルの値を 365 に設定し、その値を取得します。

```
<cfscript>
    //Create a new Excel spreadsheet object.
    theSheet=SpreadsheetNew();
    //Set the value of the cell at row 3 column 5.
    SpreadsheetSetCellValue(theSheet,365,3,5);
    //Get the value from the Excel spreadsheet object.
    theValue=SpreadsheetGetCellValue(theSheet,3,5);
    WriteOutput("The value of column 5 row 3 is: " & theValue);
</cfscript>
```

SpreadsheetSetColumnWidth

説明
ワークシートの列の幅を設定します。

戻り値
なし

カテゴリ
Microsoft Office の統合

関数のシンタックス
```
SpreadSheetSetColumnWidth(spreadhsheetobj, column number, width)
```

関連項目

SpreadsheetGetCellComment、SpreadsheetFormatCell、SpreadsheetGetCellFormula、SpreadsheetGetCellValue、SpreadsheetMergeCells、SpreadsheetSetCellFormula、SpreadsheetSetCellValue

履歴

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
spreadsheetobj	列の高さを設定する Excel スプレッドシートオブジェクトです。
column number	幅を設定するセルを指定します。
width	幅をポイントで指定します。

例

次の例では、スプレッドシートを作成して列を追加し、この新しく追加した列の幅を設定します。

```
<cfscript>
a=SpreadSheetNew();
SpreadSheetAddRow(a,"1,2,3,4,5,6,7,8");
SpreadSheetAddRow(a,"1,2,3,4,5,6,7,8",2,1);
</cfscript>
<cfset SpreadSheetSetColumnWidth(a,2,10)>
<cfset SpreadSheetSetColumnWidth(a,3,25)>
<cfspreadsheet action="write" filename="#expandpath('.')#/b.xls" name="a" overwrite="true">
```

SpreadsheetSetFooter

説明

指定されたワークシートにフッタを追加します。

カテゴリ

Microsoft Office の統合

関数のシンタックス

```
SpreadsheetSetFooter(spreadhsheetobj, left footer, center footer, right footer)
```

関連項目

SpreadsheetGetCellComment、SpreadsheetFormatCell、SpreadsheetGetCellFormula、SpreadsheetGetCellValue、SpreadsheetMergeCells、SpreadsheetSetCellFormula、SpreadsheetSetCellValue

履歴

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
spreadsheetobj	フッタの追加先となる Excel スプレッドシートオブジェクトです。
left footer	ワークシートの左側にフッタを追加します。
center footer	ワークシートの中央にフッタを追加します。
right footer	ワークシートの右側にフッタを追加します。

使用方法
追加したフッタは、印刷されたワークシートドキュメントに表示されます。

例
次の例では、ワークシートの左側にフッタを追加します。

```
<cfspreadsheet action="read" src="#dirname#SingleSheet.xls" sheet="2" name="SpreadsheetObj" >
<cfset spreadsheetSetHeader(SpreadsheetObj,"left header","center header","right header")>
<cfset spreadsheetSetFooter(SpreadsheetObj,"left footer","center footer","right footer")>
<cfspreadsheet action="write" filename="#dirname#MySheet.xls"  name="SpreadsheetObj" overwrite="true" >
```

SpreadsheetSetHeader

説明
指定されたワークシートにヘッダを追加します。

戻り値
なし

カテゴリ
Microsoft Office の統合

関数のシンタックス

```
SpreadsheetSetHeader(spreadhsheetobj, left header, center header, right header)
```

関連項目
SpreadsheetGetCellComment、SpreadsheetFormatCell、SpreadsheetGetCellFormula、SpreadsheetGetCellValue、SpreadsheetMergeCells、SpreadsheetSetCellFormula、SpreadsheetSetCellValue

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
spreadsheetobj	ヘッダの追加先となる Excel スプレッドシートオブジェクトです。

パラメータ	説明
left header	ワークシートの左側にヘッダを追加します。
center header	ワークシートの中央にヘッダを追加します。
right header	ワークシートの右側にヘッダを追加します。

使用方法
追加したヘッダは、印刷されたワークシートドキュメントに表示されます。

例
次の例では、ワークシートの中央にヘッダを追加します。

```
<cfspreadsheet action="read" src="#dirname#SingleSheet.xls" sheet="2" name="SpreadsheetObj" ><cfset
spreadsheetSetHeader(SpreadsheetObj,"left header","center header","right header")><cfset
spreadsheetSetFooter(SpreadsheetObj,"left footer","center footer","right footer")><cfspreadsheet
action="write" filename="#dirname#MySheet.xls"  name="SpreadsheetObj" overwrite="true" >
```

SpreadsheetSetRowHeight

説明
ワークシートの行の高さを設定します。

戻り値
なし

カテゴリ
Microsoft Office の統合

関数のシンタックス
```
SpreadSheetSetRowHeight(spreadhsheetobj, row number, height)
```

関連項目
SpreadsheetGetCellComment、SpreadsheetFormatCell、SpreadsheetGetCellFormula、SpreadsheetGetCellValue、SpreadsheetMergeCells、SpreadsheetSetCellFormula、SpreadsheetSetCellValue

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
spreadsheetobj	列の高さを設定する Excel スプレッドシートオブジェクトです。
row number	高さを設定する行を指定します。
height	高さを指定します (単位 : ポイント)。

使用方法

例
次の例では、スプレッドシートを作成して行を追加し、この新しく追加した行の高さを設定します。

```
<cfscript>
a=SpreadSheetNew();
SpreadSheetAddRow(a,"1,2,3,4,5,6,7,8");
SpreadSheetAddRow(a,"1,2,3,4,5,6,7,8",2,1);
</cfscript>
<cfset SpreadSheetSetRowHeight(a,2,10)>
<cfset SpreadSheetSetRowHeight(a,3,25)>
<cfspreadsheet action="write" filename="#expandpath('.')#/b.xls" name="a" overwrite="true">
```

SpreadsheetShiftColumns

説明
Excel スプレッドシートオブジェクトの複数の列を左または右に移動します。

戻り値
なし

カテゴリ
Microsoft Office の統合

関数のシンタックス
```
SpreadsheetShiftColumns(spreadsheetObj, start[, cols]
SpreadsheetShiftColumns(spreadsheetObj, start, end [, cols])
```

関連項目
SpreadsheetAddColumn、SpreadsheetDeleteColumn、SpreadsheetDeleteColumns、SpreadsheetFormatColumn、SpreadsheetFormatColumns、SpreadsheetShiftRows

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
spreadsheetObj	移動を行う Excel スプレッドシートオブジェクトです。
start	移動する最初の列または唯一の列の列番号です。
end	移動する最後の列の列番号です。このパラメータを省略すると、1 列移動されます。
columns	列を移動する正 (右) の列数または負 (左) の列数です。このパラメータを省略すると、列は右に 1 ユニット分移動されます。

使用方法

例
次の例では、列 6 と 7 を 2 列分左に移動します。

```
<cfset SpreadsheetShiftColumns(SpreadsheetObj,10,11,2)><cfscript>
    ///We need an absolute path, so get the current directory path.
      theFile=GetDirectoryFromPath(GetCurrentTemplatePath()) & "shiftcolumns.xls";
    //Create a new Excel spreadsheet object.
    theSheet = SpreadsheetNew("Expenses");
    //Set some cell values, indicating their initial location.
    SpreadsheetSetCellValue(theSheet,"Cell D10",10,4);
    SpreadsheetSetCellValue(theSheet,"Cell E12",12,5);
    SpreadsheetSetCellValue(theSheet,"Cell F12",12,6);
    SpreadsheetSetCellValue(theSheet,"Cell G13",13,7);
    //Shift columns 6 and 7 left 2 columns.
    SpreadsheetShiftColumns(theSheet,6,7,-2);
</cfscript>

<!--- Write the spreadsheet to a file, replacing any existing file. --->
<cfspreadsheet action="write" filename="#theFile#" name="theSheet" overwrite=true>
```

SpreadsheetShiftRows

説明

Excel スプレッドシートオブジェクトの複数の行を上または下に移動します。空白セルを含む移動する行の内容は、移動先の列のデータで上書きされます。

戻り値

なし

カテゴリ

Microsoft Office の統合

関数のシンタックス

```
SpreadsheetShiftRows(spreadsheetObj, start[, rows]
SpreadsheetShiftRows(spreadsheetObj, start, end, rows)
```

関連項目

SpreadsheetAddRow、SpreadsheetAddRows、SpreadsheetDeleteRow、SpreadsheetDeleteRows、SpreadsheetFormatRow、SpreadsheetFormatRows、SpreadsheetShiftColumns

履歴

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
spreadsheetObj	移動を行う Excel スプレッドシートオブジェクトです。
start	移動する最初の行の行番号です。
end	移動する最後の行の行番号です。このパラメータを省略すると、1 行移動されます。
rows	行を移動する正 (下) の行数または負 (上) の行数です。このパラメータを省略すると、行は下に 1 ユニット分移動されます。

使用方法

例

次の例では、行 10 と 11 を 2 行分下に移動します。移動された行は元の行 12 と 13 を完全に上書きすることに注意してください。

```
<cfscript>
    ///We need an absolute path, so get the current directory path.
      theFile=GetDirectoryFromPath(GetCurrentTemplatePath()) & "shiftrows.xls";
    //Create a new Excel spreadsheet object.
    theSheet = SpreadsheetNew("Expenses");
    //Set some cell values, indicating their initial location.
    SpreadsheetSetCellValue(theSheet,"Cell D10",10,4);
    SpreadsheetSetCellValue(theSheet,"Cell E11",11,5);
    SpreadsheetSetCellValue(theSheet,"Cell A12",12,1);
    SpreadsheetSetCellValue(theSheet,"Cell B13",13,2);
    //Shift rows 10 and 11 down 2 rows.
    SpreadsheetShiftRows(theSheet,10,11,2);
</cfscript>

<!--- Write the spreadsheet to a file, replacing any existing file. --->
<cfspreadsheet action="write" filename="#theFile#" name="theSheet" overwrite=true>
```

SpreadsheetWrite

説明

ColdFusion スプレッドシートオブジェクトから新しい XLS ファイルに 1 つのシートを書き込みます。

カテゴリ

Microsoft Office の統合

関数のシンタックス

```
SpreadSheetWrite(SpreadsheetObj, fileName)
SpreadSheetWrite(SpreadsheetObj, fileName [,overwrite])
SpreadSheetWrite(SpreadsheetObj, fileName [, password])
SpreadSheetWrite(SpreadsheetObj, fileName [, password,overwrite])
```

関連項目

SpreadsheetRead、SpreadsheetAddRow、SpreadsheetAddRows、SpreadsheetDeleteRow、SpreadsheetDeleteRows、SpreadsheetFormatRow、SpreadsheetFormatRows、SpreadsheetShiftColumns

履歴

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
spreadSheetObj	書き込み対象の Excel スプレッドシートオブジェクトです。

パラメータ	説明
fileName	書き込まれるファイルのパス名です。
overwrite	既存のファイルを上書きするかどうかを指定するブール値です。上書きする場合は yes を指定します。
password	アクティブシートを保護するためのパスワードです。パスワードは Excel 97 ～ 2003 のファイル形式のみに適用されます。XML ファイル形式 (Excel 2007) の場合は無視されます。

使用方法
この関数は次のような目的に使用できます。

- 複数のシートを 1 つのファイルに書き込む。
- 既存のファイルを更新し、ファイル内のすべてのシートを読み取り、1 つまたは複数のシートを変更し、ファイル全体を再度書き込む。

例
```
<cfscript>
    spreadsheet = SpreadSheetRead("C:\Files\Report.xls","Annual Report");
    SpreadSheetWrite(spreadsheet,"C:\Files\Report.xls","P@ssword","yes");
</cfscript>
```

例 2

```
<cfscript>
        spObj = spreadsheetread("#dirname#SingleSheet.xls","Sheet2");
        spreadsheetCreateSheet(spObj,"A");
        spreadsheetaddrow(spObj,"x,x,x,x,x",3,1);
        spreadsheetsetActiveSheet(spObj,"A");
        spreadsheetaddrow(spObj,"z,z,z,z,z",3,1);
        spreadsheetsetActiveSheetNumber(spObj,1);
        spreadsheetaddrow(spObj,"a,b,c,d,e",3,1);
        SpreadsheetWrite(spObj,"#dirname#SingleSheet1.xls","yes");
</cfscript>
```

Sqr

説明
数値の平方根を計算します。

戻り値
number の平方根を表す数値

カテゴリ
算術関数

関数のシンタックス
```
Sqr(number)
```

関連項目
Abs

パラメータ

パラメータ	説明
number	正の整数、または正の整数を含んでいる変数です。この数値の平方根を求めます。

使用方法
number には 0 以上の数値を指定する必要があります。

例
```
<h3>Sqr Example</h3>

<p>Returns the square root of a positive number.

<p>Sqr(2): <cfoutput>#Sqr(2)#</cfoutput>
<p>Sqr(Abs(-144)): <cfoutput>#Sqr(Abs(-144))#</cfoutput>
<p>Sqr(25^2): <cfoutput>#Sqr(25^2)#</cfoutput>
```

StripCR

説明
文字列から改行文字を削除します。

戻り値
string から改行文字を削除した結果のコピー

カテゴリ
表示および書式制御関数、文字列関数

関数のシンタックス
```
StripCR(string)
```

関連項目
ParagraphFormat

パラメータ

パラメータ	説明
string	文字列、または文字列を含んでいる変数です。

使用方法
textarea フィールドに入力されたデータを、<pre> タグと </pre> タグで囲んで形式設定済み HTML として表示する場合に有用です。

例

```
<h3>StripCR Example</h3>

<p>Function StripCR is useful for preformatted HTML display of data
    (PRE) entered in textarea fields.
<cfif isdefined("Form.myTextArea")>

<pre>
<cfoutput>#StripCR(Form.myTextArea)#</cfoutput>
</pre>
</cfif>
<!--- use #Chr(10)##Chr(13)# to simulate line feed/carriage return combination --->
<form action = "stripcr.cfm">
<textarea name = "MyTextArea" cols = "35" rows = 8>
This is sample text and you see how it scrolls
    <cfoutput>#Chr(10)##Chr(13)#</cfoutput>
From one line
    <cfoutput>#Chr(10)##Chr(13)##Chr(10)##Chr(13)#</cfoutput>
    to the next
</textarea>
<input type = "Submit" name = "Show me the HTML version">
</form>
```

StructAppend

説明

ある構造体を別の構造体に追加します。

戻り値

正常に完了した場合は true、そうでない場合は false。

カテゴリ

構造体関数

関数のシンタックス

```
StructAppend(struct1, struct2, overwriteFlag)
```

関連項目

構造体関数、『ColdFusion アプリケーションの開発』の Modifying a ColdFusion XML object

履歴

ColdFusion MX: 動作が変更されました。この関数を XML オブジェクトに対して使用できるようになりました。

パラメータ

パラメータ	説明
struct1	struct2 の追加先の構造体です。
struct2	struct1 に追加するデータを含んだ構造体です。
overwriteFlag	• true または Yes: struct2 の値で、struct1 の対応する値を上書きします (デフォルト)。 • false または No: struct2 の値で、struct1 の対応する値を上書きしません。

使用方法

この関数は、struct2 のフィールドと値を struct1 に追加します。struct2 は変更されません。struct1 に既に struct2 のフィールドが含まれている場合、struct2 の値で上書きするかどうかは overwriteFlag により決定されます。

構造体のキーはソートされていません。

例

```
<html>
<body>
<!---- Create a Name structure --->
<cfset nameCLK=StructNew()>
<cfset nameCLK.first="Chris">
<cfset nameCLK.middle="Lloyd">
<cfset nameCLK.last="Gilson">
<!--- Create an address struct --->
<cfset addrCLK=StructNew()>
<cfset addrCLK.street="17 Gigantic Rd">
<cfset addrCLK.city="Watertown">
<cfset addrCLK.state="MA">
<cfset addrCLK.zip="02472">
<!---- Create a Person structure --->
<cfset personCLK=StructNew()>
<cfset personCLK.name=#nameCLK#>
<cfset personCLK.addr=#addrCLK#>
<!--- Display the contents of the person struct before the Append --->
<p>
The person struct <b>before</b> the Append call:<br>
<cfloop collection=#personCLK# item="myItem">
<cfoutput>
#myItem#<br>
</cfoutput>
</cfloop>
<!--- Merge the address struct into the top-level person struct --->
<cfset bSuccess = StructAppend( personCLK, addrCLK )>

<!--- Display the contents of the person struct, after the Append --->
<p>
The person struct <b>after</b> the Append call:<br>
<cfloop collection=#personCLK# item="myItem">
    <cfoutput>
        #myItem#<br>
    </cfoutput>
</cfloop>
```

StructClear

説明

構造体からすべてのデータを削除します。

戻り値

実行できた場合は true、実行できなかった場合は false

カテゴリ

構造体関数

関数のシンタックス

```
StructClear(structure)
```

関連項目

構造体関数、『ColdFusion アプリケーションの開発』の Modifying a ColdFusion XML object

履歴

ColdFusion MX: 動作が変更されました。この関数を XML オブジェクトに対して使用できるようになりました。

パラメータ

パラメータ	説明
structure	データを削除する構造体です。

使用方法

セッション変数に対してこの関数を呼び出さないでください。詳細については、go.adobe.com/kb/ts_tn_17479_en-us のテクニカルノート「**ColdFusion 4.5 and the StructClear(Session) function**」を参照してください。この記事は、ColdFusion 4.5、5.x、および ColdFusion MX を対象としています。

例

```
<!--- Shows StructClear function. Calls cf_addemployee custom tag which
    uses the addemployee.cfm file. --->
<body>
<h1>Add New Employees</h1>
<!--- Establish params for first time through --->
<cfparam name = "Form.firstname" default = "">
<cfparam name = "Form.lastname" default = "">
<cfparam name = "Form.email" default = "">
<cfparam name = "Form.phone" default = "">
<cfparam name = "Form.department" default = "">
<cfif form.firstname eq "">
    <p>Please fill out the form.
<cfelse>
    <cfoutput>
<cfscript>
    employee = StructNew();
    StructInsert(employee, "firstname", Form.firstname);
    StructInsert(employee, "lastname", Form.lastname);
    StructInsert(employee, "email", Form.email);
    StructInsert(employee, "phone", Form.phone);
    StructInsert(employee, "department", Form.department);
</cfscript>
    </cfoutput>
<!--- Call the custom tag that adds employees --->
    <cf_addemployee empinfo = "#employee#">
    <cfscript>StructClear(employee);</cfscript>
    </cfif>
```

StructCopy

説明

構造体をコピーします。トップレベルのキー、値、および配列は値がコピーされます。ネストされた構造体は参照がコピーされます。

戻り値
同じキーと値を持つ構造体のコピー。**structure** が存在しない場合は例外が発生します。

カテゴリ
構造体関数

関数のシンタックス
```
StructCopy(structure)
```

関連項目
構造体関数、『ColdFusion アプリケーションの開発』の Structure functions

パラメータ

パラメータ	説明
structure	コピーする構造体です。

使用方法
文字フィールド、数値フィールド、および 2 次元配列をトップレベルに持つ構造体を、この関数でコピーすると、次のコードと同様の処理が行われます。

```
<cfoutput>
    <cfset assignedCopy = StructNew()>
<cfset assignedCopy.string = #struct.string#>
    <cfset assignedCopy.number = #struct.number#>
    <cfset assignedCopy.array = ArrayNew(2)>
    <cfset assignedCopy.array[1][1] = #struct.array[1][1]#>
    <cfset assignedCopy.array[1][2] = #sruct.array[1][2]#>
</cfoutput>
```

ネストされた構造体を StructCopy でコピーすると、次のコードと同様の処理が行われます。

```
<cfoutput>
<cfset assignedCopy.nestedStruct = struct.nestedStruct>
</cfoutput>
```

構造体全体を値でコピーするには、862 ページの「Duplicate」を使用します。

次の表に、各種の変数が代入される方法を示します。

変数タイプ	代入方法
structure.any_simple_value (単純値) ブール値 バイナリ Base64	値
structure.array (配列)	値
structure.nested_structure (ネストされた構造体)	参照
structure.object (オブジェクト)	参照
structure.query (クエリー)	参照

例

```
<!--- This code shows assignment by-value and by-reference. --->
// This script creates a structure that StructCopy copies by value. <br>
<cfscript>
    // Create elements.
    s = StructNew();
    s.array = ArrayNew(2);

    // Assign simple values to original top-level structure fields.
    s.number = 99;
    s.string = "hello tommy";

    // Assign values to original top-level array.
    s.array[1][1] = "one one";
    s.array[1][2] = "one two";
</cfscript>

<!--- Output original structure --->
<hr>
<b>Original Values</b><br>
<cfoutput>
    // Simple values <br>
    s.number = #s.number#<br>
    s.string = #s.string#<br>
    // Array value <br>
    s.array[1][1] = #s.array[1][1]#<br>
    s.array[1][2] = #s.array[1][2]#<br>
</cfoutput>

// Copy this structure to a new structure. <br>
<cfset copied = StructCopy(s)>

<cfscript>
// Change the values of the original structure. <br>
    s.number = 100;
    s.string = "hello tommy (modified)";
    s.array[1][1] = "one one (modified)";
    s.array[1][2] = "one two (modified)";
</cfscript>
<hr>
<b>Modified Original Values</b><br>
<cfoutput>
    // Simple values <br>
    s.number = #s.number#<br>
    s.string = #s.string#<br>
    // Array value <br>
    s.array[1][1] = #s.array[1][1]#<br>
    s.array[1][2] = #s.array[1][2]#<br>
</cfoutput>
<hr>
<b>Copied structure values should be the same as the original.</b><br>
<cfoutput>
    // Simple values <br>
    copied.number = #copied.number#<br>
    copied.string = #copied.string#<br>
    // Array value <br>
    copied.array[1][1] = #copied.array[1][1]#<br>
    copied.array[1][2] = #copied.array[1][2]#<br>
</cfoutput>

// This script creates a structure that StructCopy copies by reference.
<cfscript>
```

```
    // Create elements.
    s = StructNew();
    s.nested = StructNew();
    s.nested.array = ArrayNew(2);
    // Assign simple values to nested structure fields.
    s.nested.number = 99;
    s.nested.string = "hello tommy";
    // Assign values to nested array.
    s.nested.array[1][1] = "one one";
    s.nested.array[1][2] = "one two";
</cfscript>

<!--- Output original structure --->
<hr>
<b>Original Values</b><br>
<cfoutput>
    // Simple values <br>
    s.nested.number = #s.nested.number#<br>
    s.nested.string = #s.nested.string#<br>

    // Array values <br>
    s.nested.array[1][1] = #s.nested.array[1][1]#<br>
    s.nested.array[1][2] = #s.nested.array[1][2]#<br>
</cfoutput>

// Use StructCopy to copy this structure to a new structure. <br>
<cfset copied = StructCopy(s)>
// Use Duplicate to clone this structure to a new structure. <br>
<cfset duplicated = Duplicate(s)>

<cfscript>
// Change the values of the original structure.
    s.nested.number = 100;
    s.nested.string = "hello tommy (modified)";
    s.nested.array[1][1] = "one one (modified)";
    s.nested.array[1][2] = "one two (modified)";
</cfscript>
<hr>
<b>Modified Original Values</b><br>
<cfoutput>
    // Simple values <br>
    s.nested.number = #s.nested.number#<br>
    s.nested.string = #s.nested.string#<br>

    // Array value <br>
    s.nested.array[1][1] = #s.nested.array[1][1]#<br>
    s.nested.array[1][2] = #s.nested.array[1][2]#<br>
</cfoutput>

<hr>
```

```
<b>Copied structure values should reflect changes to original.</b><br>
<cfoutput>
    // Simple values <br>
    copied.nested.number = #copied.nested.number#<br>
    copied.nested.string = #copied.nested.string#<br>
    // Array values <br>
    copied.nested.array[1][1] = #copied.nested.array[1][1]#<br>
    copied.nested.array[1][2] = #copied.nested.array[1][2]#<br>
</cfoutput>

<hr>
<b>Duplicated structure values should remain unchanged.</b><br>
<cfoutput>
    // Simple values <br>
    duplicated.nested.number = #duplicated.nested.number#<br>
    duplicated.nested.string = #duplicated.nested.string#<br>
    // Array value <br>
    duplicated.nested.array[1][1] = #duplicated.nested.array[1][1]#<br>
    duplicated.nested.array[1][2] = #duplicated.nested.array[1][2]#<br>
</cfoutput>
```

StructCount

説明
構造体内のキーの数を数えます。

戻り値
数値。**structure** が存在しない場合は例外が発生します。

カテゴリ
構造体関数

関数のシンタックス
`StructCount(structure)`

関連項目
構造体関数、『ColdFusion アプリケーションの開発』の Modifying a ColdFusion XML object

履歴
ColdFusion MX: 動作が変更されました。この関数を XML オブジェクトに対して使用できるようになりました。

パラメータ

パラメータ	説明
structure	キーの数を調べる構造体です。

例

```
<!--- This view-only example shows use of StructCount. --->
<p>This file is similar to addemployee.cfm, which is called by
    StructNew, StructClear, and StructDelete. To test, copy
    StructCount function to appropriate place in addemployee.cfm.
<!---
<cfswitch expression = "#ThisTag.ExecutionMode#">
<cfcase value = "start">
    <cfif StructIsEmpty(attributes.EMPINFO)>
<cfoutput>Error. No employee data was passed.</cfoutput>
    <cfexit method = "ExitTag">
    <cfelse>
<cfquery name = "AddEmployee" datasource = "cfdocexamples">
    INSERT INTO Employees
    (FirstName, LastName, Email, Phone, Department)
VALUES
    <cfoutput>
(
                '#StructFind(attributes.EMPINFO, "firstname")#' ,
                '#StructFind(attributes.EMPINFO, "lastname")#' ,
                '#StructFind(attributes.EMPINFO, "email")#' ,
                '#StructFind(attributes.EMPINFO, "phone")#' ,
                '#StructFind(attributes.EMPINFO, "department")#'
    )
    </cfoutput>
    </cfquery>
    </cfif>
    <cfoutput><hr>Employee Add Complete
    <p>#StructCount(attributes.EMPINFO)# columns added.</cfoutput>
</cfcase>
</cfswitch> --->
```

StructDelete

説明

構造体から要素を削除します。

戻り値

ブール値。indicatenotexisting パラメータの値によって異なります。

カテゴリ

構造体関数

関数のシンタックス

```
StructDelete(structure, key [, indicatenotexisting ])
```

関連項目

構造体関数、『ColdFusion アプリケーションの開発』の Modifying a ColdFusion XML object

履歴

ColdFusion MX: 動作が変更されました。この関数を XML オブジェクトに対して使用できるようになりました。

パラメータ

パラメータ	説明
structure	構造体、または構造体を含んでいる変数です。この中の要素が削除されます。
key	削除する要素です。
indicatenotexisting	• true: key が存在する場合は yes、存在しない場合は no を返すことを指定します。 • false: key が存在するかどうかにかかわらず yes を返すことを指定します (デフォルト)。

例

```
<h3>StructDelete Function</h3>
<!--- Delete the surrounding comments to make this page work
<p>This example uses the StructInsert and StructDelete functions.
<!--- Establish params for first time through --->
<cfparam name = "firstname" default = "Mary">
<cfparam name = "lastname" default = "Sante">
<cfparam name = "email" default = "msante@allaire.com">
<cfparam name = "phone" default = "777-777-7777">
<cfparam name = "department" default = "Documentation">

    <cfif IsDefined("FORM.Delete")>
    <cfoutput>
    Field to be deleted: #form.field#
</cfoutput>
<p>
<CFScript>
    employee = StructNew();
    StructInsert(employee, "firstname", firstname);
    StructInsert(employee, "lastname", lastname);
    StructInsert(employee, "email", email);
    StructInsert(employee, "phone", phone);
    StructInsert(employee, "department", department);
</CFScript>
    Before deletion, employee structure looks like this:
<cfdump var="#employee#">
<br>
        <cfset rc = StructDelete(employee, "#form.field#", "True")>
        <cfoutput>
    Did I delete the field "#form.field#"? The code indicates: #rc#<br>
        The structure now looks like this:<br>
<cfdump var="#employee#">
<br>
    </cfoutput>
</cfif>
<br><br>
<form method="post" action = "#CGI.Script_Name#">
    <p>Select the field to be deleted: 
    <select name = "field">
    <option value = "firstname">first name
    <option value = "lastname">last name
    <option value = "email">email
    <option value = "phone">phone
    <option value = "department">department
    </select>
    <input type = "submit" name = "Delete" value = "Delete">
</form>
Delete this comment to make this page work --->
```

StructEach

説明
キーと値のペアにアクセスして、構造体内の要素をループする場合に使用します。

戻り値
なし

カテゴリ
クロージャ関数

シンタックス
structEach(array,function(key, value) {});

関連項目
その他のクロージャ関数

履歴
ColdFusion 10: この関数が追加されました。

パラメータ

パラメータ	説明
struct	構造体オブジェクトの名前です。
function	構造体内のキーと値の各ペアに対して実行されるインライン関数です。
key	構造体内のキーです。
value	構造体内の値です。

StructFilter

説明
構造体内のキーと値のペアをフィルタリングする場合に使用します。

戻り値
新しい構造体

カテゴリ
クロージャ関数

シンタックス
structFilter(struct,function(***key, value***){return true|false;});

関連項目
その他のクロージャ関数

履歴
ColdFusion 10: この関数が追加されました。

パラメータ

パラメータ	説明
struct	構造体オブジェクトの名前です。
function	配列内の各要素に対して実行されるインライン関数です。結果の構造体に、構造体内のキーと値のペアが含まれる場合は true が返されます。
key	構造体内のキーです。
value	構造体内の値です。

StructFind

説明
構造体内のキーに関連付けられている値を調べます。

戻り値
構造体内のキーに関連付けられている値。**structure** が存在しない場合は例外が発生します。

カテゴリ
構造体関数

関数のシンタックス
```
StructFind(structure, key)
```

関連項目
構造体関数、『ColdFusion アプリケーションの開発』の Structure functions

パラメータ

パラメータ	説明
structure	調べる値が含まれている構造体です。
key	キーです。これに対応する値が返されます。

使用方法
構造体のキーはソートされていません。

例

```
<!--- This view-only example shows the use of StructFind. --->
<p>This file is identical to addemployee.cfm, which is called by StructNew,
    StructClear, and StructDelete. It adds employees. Employee information
    is passed through the employee structure (EMPINFO attribute). In UNIX,
    you must also add the Emp_ID.
<!--- <cfswitch expression = "#ThisTag.ExecutionMode#">
<cfcase value = "start">
    <cfif StructIsEmpty(attributes.EMPINFO)>
<cfoutput>Error. No employee data was passed.</cfoutput>
    <cfexit method = "ExitTag">
    <cfelse>
    <cfquery name = "AddEmployee" datasource = "cfdocexamples">
    INSERT INTO Employees (FirstName, LastName, Email, Phone, Department)
VALUES
    <cfoutput>
(
                '#StructFind(attributes.EMPINFO, "firstname")#' ,
                '#StructFind(attributes.EMPINFO, "lastname")#' ,
                '#StructFind(attributes.EMPINFO, "email")#' ,
                '#StructFind(attributes.EMPINFO, "phone")#' ,
                '#StructFind(attributes.EMPINFO, "department")#' )
    </cfoutput>
</cfquery>
    </cfif>
    <cfoutput><hr>Employee Add Complete</cfoutput>
</cfcase>
</cfswitch> --->
```

StructFindKey

説明

ネストされた配列、構造体、および他の要素を再帰的に検索して、**value** パラメータで指定された検索キーに一致する値を持つ構造体を探します。

戻り値

value に一致する値を持つ構造体を含んでいる配列

カテゴリ

構造体関数

関数のシンタックス

```
StructFindKey(top, value, scope)
```

関連項目

構造体関数、『ColdFusion アプリケーションの開発』の Structure functions

パラメータ

パラメータ	説明
top	検索の開始位置である ColdFusion オブジェクトです (構造体または配列)。このパラメータには、オブジェクト名ではなくオブジェクトが必要です。
value	文字列、または文字列を含む変数です。これで指定した値を検索します。
scope	• one: 一致するキーを 1 つ返します (デフォルト)。 • all: 一致するキーをすべて返します。 キーが見つからない場合は、空の配列が返されます。

使用方法

検索結果は構造体の配列で返されます。各検索結果がそれぞれ 1 つの構造体に入っています。各構造体は次のようなフィールドを持ちます。

- Value: 見つかったキーに格納されている値
- Path: 見つかったキーに到達するパスの文字列
- Owner: 見つかったキーを含んでいる親オブジェクト

構造体のキーはソートされていません。

例

```
<cfset aResults = StructFindKey( #request#, "bass" )>
```

StructFindValue

説明

ネストされた配列、構造体、および他の要素を再帰的に検索して、value パラメータで指定された検索キーに一致する値を持つ構造体を探します。

戻り値

検索キー **value** に一致する値を持つ構造体を含んでいる配列。何も見つからなかった場合はサイズ 0 の配列が返されます。

カテゴリ

構造体関数

関数のシンタックス

```
StructFindValue( top, value [, scope])
```

関連項目

構造体関数、『ColdFusion アプリケーションの開発』の Structure functions

パラメータ

パラメータ	説明
top	検索の開始位置である ColdFusion 構造体です。このパラメータには、オブジェクト名ではなくオブジェクトが必要です。
value	文字列、または文字列を含む変数です。これで指定した値を検索します。 検索できるのは基本オブジェクトのみです。配列や構造体には対応していません。
scope	• one: 一致するキーを 1 つ返します (デフォルト)。 • all: 一致するキーをすべて返します。

使用方法
戻り値の配列に含まれている各構造体は次のようなフィールドを持ちます。

- Key: 見つかった値が格納されているキーの名前
- Path: 見つかったキーに到達するパスの文字列
- Owner: 見つかったキーを含んでいる親オブジェクト

構造体のキーはソートされていません。

例
```
<cfset aResults = StructFindValue( #request#, "235" )>
```

StructGet

説明
指定したパスから構造体を取得します。

戻り値
pathDesired パラメータに指定された変数へのエイリアス。StructGet は、**pathDesired** が有効なパスとなるよう、必要に応じて構造体または配列を作成します。

カテゴリ
構造体関数

関数のシンタックス
```
StructGet(pathDesired)
```

関連項目
構造体関数、『ColdFusion アプリケーションの開発』の Modifying a ColdFusion XML object

履歴
ColdFusion MX:

- 動作が変更されました。この関数を XML オブジェクトに対して使用できるようになりました。
- 動作が変更されました。**pathDesired** に構造体や配列がない場合には、この関数は構造体や配列を作成して、**pathDesired** が有効なパスとなるようにします。

パラメータ

パラメータ	説明
pathDesired	構造体または配列を含んでいる変数のパス名です。この中から構造体を取得します。

使用方法

この関数を不用意に使用すると、誤って無効な構造体を作成してしまう可能性があります。たとえば、既存の配列を拡張するために配列表記を使用すると、配列で現在保持されているタイプにかかわらず、指定された新しい要素が作成されます。

例

```
<!--- GetStruct() test --->
<cfset test = StructGet( "dog.myscope.test" )>
<cfset test.foo = 1>
<cfif NOT IsDefined("dog")>
    Dog is not defined<br>
</cfif>
<cfif NOT IsDefined("dog.myscope")>
    Dog.Myscope is not defined<br>
</cfif>
<cfif NOT Isdefined("dog.myscope.test")>
    Dog.Myscope.Test is not defined<br>
</cfif>
<cfif NOT Isdefined("dog.myscope.test.foo")>
    Dog.Myscope.Test.Foo is not defined<br>
</cfif>
<cfoutput>
    #dog.myscope.test.foo#<br>
</cfoutput>
<cfset test = StructGet( "request.myscope[1].test" )>
<cfset test.foo = 2>
<cfoutput>
    #request.myscope[1].test.foo#<br>
</cfoutput>
<cfset test = StructGet( "request.myscope[1].test[2]" )>
<cfset test.foo = 3>
<cfoutput>
    #request.myscope[1].test[2].foo#<br>
</cfoutput>
```

StructInsert

説明

キーと値のペアを構造体に挿入します。

戻り値

正常に完了した場合は true。structure が存在しない場合、または key が既に存在して allowoverwrite = "False" の場合は、例外が発生します。

カテゴリ

構造体関数

関数のシンタックス

```
StructInsert(structure, key, value [, allowoverwrite ])
```

関連項目
構造体関数、『ColdFusion アプリケーションの開発』の Modifying a ColdFusion XML object

履歴
ColdFusion MX: 動作が変更されました。この関数を XML オブジェクトに対して使用できるようになりました。

パラメータ

パラメータ	説明
structure	新しいキーと値のペアを挿入する構造体です。
key	挿入する値を持つキーです。
value	挿入する値です。
allowoverwrite	オプション。既存のキーへの上書きを認めるかどうかを指定します。デフォルト値は false です。

使用方法
構造体のキーはソートされていません。

例
```
 <h1>Add New Employees</h1>
<!--- Establish params for first time through --->
<cfparam name = "FORM.firstname" default = "">
<cfparam name = "FORM.lastname" default = "">
<cfparam name = "FORM.email" default = "">
<cfparam name = "FORM.phone" default = "">
<cfparam name = "FORM.department" default = "">

<cfif FORM.firstname EQ "">
    <p>Please fill out the form.
<cfelse>
    <cfoutput>
<CFScript>
    employee = StructNew();
    StructInsert(employee, "firstname", FORM.firstname);
    StructInsert(employee, "lastname", FORM.lastname);
    StructInsert(employee, "email", FORM.email);
    StructInsert(employee, "phone", FORM.phone);
    StructInsert(employee, "department", FORM.department);
    </CFScript>

    <p>First name is #StructFind(employee, "firstname")#</p>
    <p>Last name is #StructFind(employee, "lastname")#</p>
    <p>EMail is #StructFind(employee, "email")#</p>
    <p>Phone is #StructFind(employee, "phone")#</p>
    <p>Department is #StructFind(employee, "department")#</p>
    </cfoutput>
```

```
    <!--- Call the custom tag that adds employees --->
    <CF_ADDEMPLOYEE EMPINFO = "#employee#">
</cfif>

<Hr>
<form action = "structinsert.cfm">
    <p>First Name: 
    <input name = "firstname" type = "text" hspace = "30" maxlength = "30">
    <p>Last Name: 
    <input name = "lastname" type = "text" hspace = "30" maxlength = "30">
    <p>EMail: 
    <input name = "email" type = "text" hspace = "30" maxlength = "30">
    <p>Phone: 
    <input name = "phone" type = "text" hspace = "20" maxlength = "20">
    <p>Department: 
    <input name = "department" type = "text" hspace = "30" maxlength = "30">
    <p>
    <input type = "submit" value = "OK">
</form>
```

StructIsEmpty

説明
構造体に何もデータが含まれていないかどうかを調べます。

戻り値
structure が空の場合は true。**structure** が存在しない場合は例外が発生します。

カテゴリ
決定関数、構造体関数

関数のシンタックス
```
StructIsEmpty(structure)
```

関連項目
構造体関数、『ColdFusion アプリケーションの開発』の Modifying a ColdFusion XML object

履歴
ColdFusion MX: 動作が変更されました。この関数を XML オブジェクトに対して使用できるようになりました。

パラメータ

パラメータ	説明
structure	データの有無を調べる構造体です。

例

```
<!--- This example illustrates use of StructIsEmpty. --->
<p>This file is identical to addemployee.cfm, which is called by StructNew,
    StructClear, and StructDelete. It adds employees. Employee information
    is passed through employee structure (EMPINFO attribute). In UNIX, you
    must also add the Emp_ID.
<cfswitch expression = "#ThisTag.ExecutionMode#">
<cfcase value = "start">
    <cfif StructIsEmpty(attributes.EMPINFO)>
<cfoutput>Error. No employee data was passed.</cfoutput>
    <cfexit method = "ExitTag">
    <cfelse>
<!--- Add the employee; In UNIX, you must also add the Emp_ID --->
    <cfquery name = "AddEmployee" datasource = "cfdocexamples">
    INSERT INTO Employees
    (FirstName, LastName, Email, Phone, Department)
VALUES
    <cfoutput>
(
                '#StructFind(attributes.EMPINFO, "firstname")#' ,
                '#StructFind(attributes.EMPINFO, "lastname")#' ,
                '#StructFind(attributes.EMPINFO, "email")#' ,
                '#StructFind(attributes.EMPINFO, "phone")#' ,
                '#StructFind(attributes.EMPINFO, "department")#'
    )
    </cfoutput>
</cfquery>
    </cfif>
    <cfoutput><hr>Employee Add Complete</cfoutput>
</cfcase>
</cfswitch>
```

StructKeyArray

説明

構造体内のキーを取り出します。

戻り値

キーの配列。**structure** が存在しない場合は例外が発生します。

カテゴリ

構造体関数

関数のシンタックス

```
StructKeyArray(structure)
```

関連項目

構造体関数、『ColdFusion アプリケーションの開発』の Modifying a ColdFusion XML object

パラメータ

パラメータ	説明
structure	構造体です。この中にあるキーのリストを取り出します。

使用方法

構造体のキーはソートされていません。

例

```
<!--- Shows StructKeyArray function to copy keys from a structure to an array.
    Uses StructNew to create structure and fills its fields with the
    information the user enters in the form fields. --->
<h3>StructKeyArray Example</h3>
<h3>Extracting the Keys from the Employee Structure</h3>
<!-- Create structure. Check whether Submit was pressed. If so, define fields
    in employee structure with user entries on form. ----->
<cfset employee = StructNew()>
<cfif Isdefined("Form.Submit")>
    <cfif Form.Submit is "OK">
        <cfset employee.firstname = FORM.firstname>
        <cfset employee.lastname = FORM.lastname>
        <cfset employee.email = FORM.email>
        <cfset employee.phone = FORM.phone>
        <cfset employee.company = FORM.company>
    <cfelseIf Form.Submit is "Clear">
        <cfset rc = StructClear(employee)>
    </cfif>
</cfif>
<p> This example uses the StructNew function to create a structure called
    "employee" that supplies employee info. Its fields are filled by
    the form. After you enter employee information in structure, the
    example uses StructKeyArray function to copy all of the keys from
    the structure into an array. </p>
<hr size = "2" color = "#0000A0">
<form action = "structkeyarray.cfm">
<table cellspacing = "2" cellpadding = "2" border = "0">
    <tr>
    <td>First Name:</td>
    <td><input name = "firstname" type = "text"
        value = "" hspace = "30" maxlength = "30"></td>
    </tr>
    <tr>
    <td>Last Name:</td>
    <td><input name = "lastname" type = "text"
        value = "" hspace = "30" maxlength = "30"></td>
    </tr>
    <tr>
    <td>EMail</td>
    <td><input name = "email" type = "text"
        value = "" hspace = "30" maxlength = "30"></td>
    </tr>
    <tr>
    <td>Phone:</td>
    <td><input name = "phone" type = "text"
        value = "" hspace = "20" maxlength = "20"></td>
    </tr>
    <tr>
    <td>Company:</td>
```

```
    <td><input name = "company" type = "text"
        value = "" hspace = "30" maxlength = "30"></td>
    </tr>
    <tr>
    <td><input type = "submit" name = "submit"
        value = "OK"></td>
    <td><b>After you submit the FORM, scroll down to see the array.</b>
    </td>
    </tr>
</table>
</form>
<cfif NOT StructISEmpty(employee)>
    <hr size = "2" color = "#0000A0">
    <cfset keysToStruct = StructKeyArray(employee)>
    <cfloop index = "i" from = "1" to = "#ArrayLen(keysToStruct)#">
        <p><cfoutput>Key#i# is #keysToStruct[i]#</cfoutput></p>
        <p><cfoutput>Value#i# is #employee[keysToStruct[i]]#</cfoutput>
        </p>
    </cfloop>
</cfif>
```

StructKeyExists

説明
構造体に特定のキーが存在するかどうかを調べます。

戻り値
structure に **key** が存在する場合は true

カテゴリ
決定関数、構造体関数

関数のシンタックス
```
StructKeyExists(structure, "key")
```

関連項目
構造体関数、『ColdFusion アプリケーションの開発』の Structure functions

パラメータ

パラメータ	説明
structure	調べる対象の構造体の名前です。
key	存在を調べるキーです。

使用方法
場合によっては、構造体である URL スコープや Form スコープを操作する際、IsDefined 関数の代わりにこの関数を使用できます。次の 2 つのコードは同等の処理を行います。

```
cfif IsDefined("Form.JediMaster")>
<cfif StructKeyExists(Form,"JediMaster")>
```

構造体のキーはソートされていません。

例

```
<!--- This example shows the use of StructKeyExists. --->
<p>This file is similar to addemployee.cfm, which is called by StructNew,
    StructClear, and StructDelete. To test, copy the &LT;CFELSEif&GT;
    statement to the appropriate place in addemployee.cfm. It is a custom tag
    to add employees. Employee information is passed through the employee
    structure (the EMPINFO attribute). In UNIX, you must also add the Emp_ID.

<cfswitch expression = "#ThisTag.ExecutionMode#">
<cfcase value = "start">
    <cfif StructIsEmpty(attributes.EMPINFO)>
<cfoutput>Error. No employee data was passed.</cfoutput>
    <cfexit method = "ExitTag">
    <cfelseIf NOT StructKeyExists(attributes.EMPINFO, "department")>
<cfscript>StructUpdate(attributes.EMPINFO, "department",
"Unassigned");
        </cfscript>
<cfexit method = "ExitTag">
         <cfelse>
```

StructKeyList

説明

構造体からキーを取り出します。

戻り値

キーのリスト。**structure** が存在しない場合は例外が発生します。

カテゴリ

構造体関数

関数のシンタックス

```
StructKeyList(structure [, delimiter])
```

関連項目

構造体関数、『ColdFusion アプリケーションの開発』の Modifying a ColdFusion XML object

パラメータ

パラメータ	説明
structure	構造体です。この中にあるキーのリストを取り出します。
delimiter	オプション。リスト内のキーの区切り文字を指定します。デフォルト値はカンマ (,) です。

使用方法

構造体のキーはソートされていません。

例

```
<!--- This example shows how to use StructKeyList to list the keys
    in a structure. It uses StructNew function to create structure
    and fills it with information user enters in form fields. --->
<!--- This section creates structure and checks whether Submit has been pressed.
    If so, code defines fields in the employee structure with what the
    user entered in the form. --->
<cfset employee = StructNew()>
<cfif Isdefined("Form.Submit")>
    <cfif Form.Submit is "OK">
        <cfset employee.firstname = FORM.firstname>
        <cfset employee.lastname = FORM.lastname>
        <cfset employee.email = FORM.email>
        <cfset employee.phone = FORM.phone>
        <cfset employee.company = FORM.company>
    <cfelseIf Form.Submit is "Clear">
        <cfset rc = StructClear(employee)>
    </cfif>
</cfif>
<html>
<head>
    <title>StructKeyList Function</title>
</head>
<body>
<h3>StructKeyList Function</h3>
<h3>Listing the Keys in the Employees Structure</h3>
<p>This example uses StructNew function to create structure "employee" that
    supplies employee information. The fields are filled with the
    contents of the following form.</p>
<p>After you enter employee information into structure, example uses
    <b>StructKeyList</b> function to list keys in structure.</p>
<p>This code does not show how to insert information into a database.
    See cfquery for more information about database insertion.
<hr size = "2" color = "#0000A0">
<form action = "structkeylist.cfm">
<table cellspacing = "2" cellpadding = "2" border = "0">
    <tr>
    <td>First Name:</td>
    <td><input name = "firstname" type = "text"
        value = "" hspace = "30" maxlength = "30"></td>
    </tr>
    <tr>
    <td>Last Name:</td>
    <td><input name = "lastname" type = "text"
        value = "" hspace = "30" maxlength = "30"></td>
    </tr>
    <tr>
    <td>EMail</td>
    <td><input name = "email" type = "text"
        value = "" hspace = "30" maxlength = "30"></td>
    </tr>
    <tr>
    <td>Phone:</td>
    <td><input name = "phone" type = "text"
        value = "" hspace = "20" maxlength = "20"></td>
    </tr>
    <tr>
```

```
    <td>Company:</td>
    <td><input name = "company" type = "text"
        value = "" hspace = "30" maxlength = "30"></td>
    </tr>
    <tr>
    <td><input type = "submit" name = "submit" value = "OK"></td>
    <td><b>After you submit form, scroll down to see the list.</b></td>
    </tr>
</table>
</form>
<cfif NOT StructISEmpty(employee)>
    <hr size = "2" color = "#0000A0">
    <cfset keysToStruct = StructKeyList(employee,"<li>")>
    <p>Here are the keys to the structure:</p>
    <ul>
    <li><cfoutput>#keysToStruct#</cfoutput>
    </ul>
    <p>If fields are correct, we can process new employee information.
    If they are not correct, consider rewriting application.</p>
</cfif>
```

StructNew

説明
構造体を作成します。

戻り値
構造体

カテゴリ
構造体関数

関数のシンタックス
```
StructNew()
```

関連項目
構造体関数、『ColdFusion アプリケーションの開発』の Structure functions

パラメータ
なし

例

```
<!--- Shows StructNew. Calls CF_ADDEMPLOYEE, which uses the |
    addemployee.cfm file to add employee record to database. --->
<h1>Add New Employees</h1>
<cfparam name = "FORM.firstname" default = "">
<cfparam name = "FORM.lastname" default = "">
<cfparam name = "FORM.email" default = "">
<cfparam name = "FORM.phone" default = "">
<cfparam name = "FORM.department" default = "">
<cfif FORM.firstname EQ "">
    <p>Please fill out the form.
<cfelse>
    <cfoutput>
<cfscript>
    employee = StructNew();
    StructInsert(employee, "firstname", FORM.firstname);
    StructInsert(employee, "lastname", FORM.lastname);
    StructInsert(employee, "email", FORM.email);
    StructInsert(employee, "phone", FORM.phone);
    StructInsert(employee, "department", FORM.department);
    </cfscript>
    <p>First name is #StructFind(employee, "firstname")#
    <p>Last name is #StructFind(employee, "lastname")#
    <p>EMail is #StructFind(employee, "email")#
    <p>Phone is #StructFind(employee, "phone")#
    <p>Department is #StructFind(employee, "department")#
    </cfoutput>
<!--- Call the custom tag that adds employees --->
    <CF_ADDEMPLOYEE EMPINFO = "#employee#">
</cfif>
```

StructSort

説明

構造体のトップレベルのキーを取り出してソートした配列を返します。ソートはアルファベット順か数値順で、構造体の任意の要素の値に基づいて行うことができます。

戻り値

トップレベルのキー名 (文字列) の配列。指定したサブ要素の値に基づいてソートされています。

カテゴリ

構造体関数

関数のシンタックス

```
StructSort(base, sortType, sortOrder, pathToSubElement)
```

関連項目

構造体関数、『ColdFusion アプリケーションの開発』の Structure functions

パラメータ

パラメータ	説明
base	1 つのフィールド (連想配列) を持つ構造体です。
localeSensitive	ロケールを考慮したソートを実行するかどうかを指定します。デフォルト値は false です。
sortType	• 数値 • text: 大文字と小文字が区別されます。すべての小文字が優先され、大文字はその後ろに並べられます。(デフォルト)。 • textnocase
sortOrder	• asc: 昇順 (a ～ z) のソートです (デフォルト)。 • desc: 降順 (z ～ a) のソートです。
pathToSubElement	文字列、または文字列を含んでいる変数です。 ソートのキーとなる要素の値にアクセスするためのパスです。トップレベルの各キーからのパスで指定します。デフォルト値は、なしです (トップレベルのエントリ自身の値に基づいてソートが行われます)。

使用方法

pathToSubElement 文字列では、配列表記はサポートされていません。また、構造体のサブ構造体だけがサポートされます。

この関数では、構造体そのものに対するソートや変更は行われません。

ColdFusion 10 では、すべての Java でサポートされているロケール固有の文字のサポート（ウムラウト文字のサポートなど）が追加されました。このサポートのフラグは、sorttype = "text" または sorttype = "textnocase" に対して追加されました。

例

```
<cfscript>
    salaries = StructNew() ;
    employees = StructNew() ;
    departments = StructNew() ;
    for ( i=1; i lt 6; i=i+1 )
    {
        salary = 120000 - i*10000 ;
        salaries["employee#i#"] = salary ;

        employee = StructNew() ;
        employee["salary"] = salary ;
        // employee.salary = salary ;
        employees["employee#i#"] = employee ;

        departments["department#i#"] = StructNew() ;
        departments["department#i#"].boss = employee ;
    }
</cfscript>

<cfoutput>
<p>list of employees based on the salary (text search): <br>
1) #ArrayToList( StructSort( salaries ) )#<br>
2) #ArrayToList( StructSort( salaries, "text", "ASC" ) )#<br>
3) #ArrayToList( StructSort( salaries, "textnocase", "ASC" ) )#<br>
4) #ArrayToList( StructSort( salaries, "text", "DESC" ) )#<br>
<p>list of employees based on the salary (numeric search): <br>
5) #ArrayToList( StructSort( salaries, "numeric", "ASC" ) )#<br>
6) #ArrayToList( StructSort( salaries, "numeric", "DESC" ) )#<br>
<p>list of employees based on the salary (subfield search): <br>
```

```
7) #ArrayToList( StructSort( employees, "numeric", "DESC", "salary" ) )#<br>
8) #ArrayToList( StructSort( employees, "text", "ASC","salary" ) )#<br>
<p>list of departments based on the salary (sub-sub-field search): <br>
9) #ArrayToList( StructSort( departments, "text", "ASC", "boss.salary" ) )#<br>
</cfoutput>

<!--- add an invalid element and test that it throws an error --->
<p><p>
<cfset employees[ "employee4" ] = StructNew()>
<cftry>
    <cfset temp = StructSort( employees, "text", "ASC", "salary" )>
    <cfoutput>We have a problem - this was supposed to throw an exception!<br>
    </cfoutput>
<cfcatch type="any">
    <cfoutput>
        ERROR: <b>This error was expected!</b><br>
        #cfcatch.message# - #cfcatch.detail#<br>
    </cfoutput>
</cfcatch>
</cftry>
```

StructUpdate

説明
指定したキーの値を更新します。

戻り値
正常に実行された場合は true。structure が存在しない場合はエラーが発生します。

カテゴリ
構造体関数

関数のシンタックス
```
StructUpdate(structure, key, value)
```

関連項目
構造体関数、『ColdFusion アプリケーションの開発』の Modifying a ColdFusion XML object

履歴
ColdFusion MX: 動作が変更されました。この関数を XML オブジェクトに対して使用できるようになりました。

パラメータ

パラメータ	説明
structure	更新する構造体です。
key	値を更新するキーです。
value	新しい値です。

例

```
<!--- This example shows the use of StructUpdate. --->
<p>This file is similar to addemployee.cfm, which is called by StructNew,
    StructClear, and StructDelete. To test this file, copy the
    &LT;CFELSEIF&GT; statement to the appropriate place in
    addemployee.cfm. It is an example of a custom tag used to add
    employees. Employee information is passed through the employee
    structure (the EMPINFO attribute). In UNIX, you must also add the Emp_ID.

<cfswitch expression = "#ThisTag.ExecutionMode#">
<cfcase value = "start">
    <cfif StructIsEmpty(attributes.EMPINFO)>
<cfoutput>Error. No employee data was passed.</cfoutput>
        <cfexit method = "ExitTag">
    <cfelseIf StructFind(attributes.EMPINFO, "department") EQ "">
<cfscript>
    StructUpdate(attributes.EMPINFO, "department", "Unassigned");
</cfscript>
<cfexit method = "ExitTag">
            <cfelse>
```

関数 t ～ z

Tan

説明

ラジアン単位で指定した角度のタンジェントを計算します。

戻り値

角度のタンジェントを表す数値

カテゴリ

算術関数

関数のシンタックス

```
Tan(number)
```

関連項目

Atn、Cos、ACos、Sin、ASin、Pi

パラメータ

パラメータ	説明
number	タンジェントを計算する角度です (単位 : ラジアン)。

使用方法

度数をラジアンに変換するには、度数に p/180 を乗算します。ラジアンを度数に変換するには、ラジアンに 180/p を乗算します。

注意：浮動小数点演算の誤差により、本来は戻り値が 0 となるような角度を指定した場合、実際には小さい値 (たとえば 6.12323399574E-017) が返される可能性があります。また、本来は戻り値が無限大または非数値 (not a number) となるような角度を指定した場合、実際には大きい値 (たとえば 1.63312393532E+016) が返される可能性があります。したがって、結果が 0 かどうかを調べる場合は、戻り値が 0.0000000000001 より小さいかどうかで判定してください。同様に、結果が無限大かどうかを調べる場合は、戻り値が 1E15 より大きいかどうかで判定してください。

例

```
<h3>Tan Example</h3>
<!--- Calculate tangent if form has been submitted --->
<cfif IsDefined("FORM.tanNum")>
<!--- Make sure input is a number --->
    <cfif IsNumeric(#FORM.tanNum#)>
<!--- Convert degrees to radians, call the Tan function. --->
<cfset tanValue=#Tan((Form.tanNum * PI()) / 180)#>
<!--- 0.0000000000001 is the function's precision limit.
    If absolute value of returned value is
    less, set result to 0 --->
<cfif Abs(tanValue) LT 0.0000000000001>
    <cfset tanValue=0>
</cfif>
<cfoutput>
        Tan(#FORM.tanNum#) = #tanValue#<br><br>
</cfoutput>
    <cfelse>
<!--- If input is not a number, show an error message --->
        <h4>You must enter a numeric angle in degrees.</h4>
    </cfif>
</cfif>
<form action = "#CGI.script_name#" method="post">
Enter an angle in degrees to get its tangent:
<br><input type = "Text" name = "tanNum" size = "15">
<br><br>
<input type = "Submit" name = "">  
<input type = "RESET"
</form>
```

ThreadJoin

説明

指定したスレッドの処理が完了するまで、現在のスレッドの処理を待機します。

カテゴリ

例外処理関数、データ出力関数

関数のシンタックス

```
ThreadJoin([threadName], [timeout])
```

関連項目

cfscript、cfthrow、cftry、cfcatch

履歴

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
threadName	現在のスレッドに結合するスレッドの名前を指定します。複数のスレッドを指定するには、カンマ区切りリストを使用します。
timeout	スレッドの処理を中断する時間です (単位 : ミリ秒)。

使用方法

threadName パラメータで指定したスレッドの処理が完了するか、timeout パラメータで指定した時間が経過するまで待機してから、現在のスレッドの処理を再開します。

- ThreadJoin(): 現在のスレッドは、すべての ColdFusion スレッドの処理が完了するまで待機します。
- ThreadJoin(***threadName***): 指定したスレッドの実行が完了するまで、現在のスレッドの実行を待機します。
- ThreadJoin(***threadName***, ***timeout***): threadName で指定された 1 つまたは複数のスレッドの実行がタイムアウトするまで、現在のスレッドを実行せずに待機します。タイムアウトを指定しないと、結合対象のスレッドが完了するまで、現在のスレッドの処理も完了できません。

例

```
<cfscript>
thread name="t1"
    {
        sleep(5000);
    }
thread name="t2"
    {
        threadjoin("t1",1000);
    }
threadjoin("t2");
</cfscript>
<cfoutput>Status of the thread T1 = #t1.Status#<br></cfoutput>
<cfoutput>Status of the thread T2 = #t2.Status#<br></cfoutput>
```

ThreadTerminate

説明

threadName で指定されたスレッドを終了します。cfthread action="terminate" と同じように動作します。

カテゴリ

例外処理関数、データ出力関数

関数のシンタックス

```
ThreadTerminate(threadName)
```

関連項目

cfscript、cfthrow、cftry、cfcatch

履歴

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
threadName	停止するスレッドの名前を指定します。

使用方法

threadName で指定されたスレッドの処理を停止するには、この関数を使用します。スレッドを終了すると、終了に関する情報を提供する ERROR メタデータ構造体がスレッドスコープに追加されます。

例

```
<cfscript>
        thread name="t1"
    {
        sleep(3000);
    }
    sleep(1000);
    threadTerminate("t1");
    sleep(1000);
</cfscript>
<cfoutput>Status of the thread = #t1.Status#<br></cfoutput>
```

Throw

説明

cfthrow タグと同様の関数で、<cfscript> モードで使用されます。

パラメータ

<cfthrow> タグと同じです。

カテゴリ

例外処理関数、データ出力関数

関数のシンタックス

name=value ペアの場合 :

```
throw (message = "message", type = "exception type", detail, errorCode = "error code", extendedInfo =
"additional info", object = "java exception object")
```

定位置表記の場合は、シンタックスで示した順序どおりに値を指定する必要があります。指定しない属性がある場合は、代わりに空の文字列を使用します。

関連項目

cfscript、cfthrow、cftry、cfcatch

使用方法

この関数を呼び出すには、name=value のペアまたは定位置引数として引数を渡します。定位置表記の場合は、関数のシンタックスで示した順序どおりに引数を指定します。

例

```
<cfscript>
function TotalInterest(principal, annualRate, months)
{   var years = 0;
    var interestRate = 0;
    var totalInterest = 0;
    principal = REReplace(trim(principal), "[\$]", "", "ALL");
    annualRate = Replace(trim(annualRate), "%", "", "ALL");

    if ((principal <= 0) OR (annualRate <= 0) OR (months <= 0)) {
        Throw(type="InvalidData",message="All values must be greater than 0.");

    //Use of Throw function in cfscript
    }
    interestRate = annualRate / 100;
    years = months / 12;
    totalInterest = principal * (((1 + interestRate) ^ years) - 1);
    return DollarFormat(totalInterest);
}

try {
    WriteOutput(TotalInterest("$2500.00", "5.5%", "12"));
    } catch(InvalidData ex){
    //Displayig exception details on screen
    WriteOutput("<p>An InvalidData exception was thrown.</p>");
    WriteOutput("<p>#ex.message#</p>");
}
</cfscript>
```

TimeFormat

説明

米国英語の時刻形式設定規則に基づいて、時刻値を形式設定します。

戻り値

カスタム形式に形式設定された時刻値。マスクが指定されていない場合は、hh:mm tt 形式で時刻値が返されます。国際時刻形式については、LSTimeFormat を参照してください。

カテゴリ

日付および時刻関数、表示および書式制御関数

関数のシンタックス

```
TimeFormat(time [, mask ])
```

関連項目

CreateTime、Now、ParseDateTime、LSTimeFormat、DateFormat

履歴

ColdFusion 10：分のマスクの「m または M」は非推奨になりました。「n または N」を使用することをお勧めします。

ColdFusion MX 6.1: ミリ秒を表すマスク文字 L および l が追加されました。

ColdFusion MX:

- 余分な文字の処理方法が変更されました。この関数では、mask 値の中の余分な文字の処理方法が、以前のリリースとは変更になっています。具体的には次のとおりです。
  - ColdFusion 5 以前では、形式設定された時刻と、余分な文字のアポストロフィ区切りリストが返されます。たとえば、TimeFormat(Now(), "hh:mm:ss dog") は、8:17:23 d'o'g を返します。
  - ColdFusion MX では、形式設定された時刻と、余分な文字がそのまま返されます。たとえば上の例では、8:17:23 dog が返されます。

  余分な文字が一重引用符で囲まれている場合 (例 : hh:mm:ss 'dog') には、ColdFusion 5 と ColdFusion MX のいずれでも、形式設定された時刻と余分な文字がそのまま返されます。たとえばこの例では、8:17:23 dog が返されます。
- 次の **mask** パラメータオプションのサポートを追加 : short、medium、long、および full

パラメータ

パラメータ	説明
time	変換する日付時刻値または文字列です。
mask	形式を指定するマスク文字からなる文字列です。 • h: 時。1 桁の場合は先頭に 0 が付きません (12 時間形式)。 • hh: 時。1 桁の場合は先頭に 0 が付きます (12 時間形式)。 • H: 時。1 桁の場合は先頭に 0 が付きません (24 時間形式)。 • HH: 時。1 桁の場合は先頭に 0 が付きます (24 時間形式)。 • m: 分。1 桁の場合は先頭に 0 が付きません。 • mm: 分。1 桁の場合は先頭に 0 が付きます。 • s: 秒。1 桁の場合は先頭に 0 が付きません。 • ss: 秒。1 桁の場合は先頭に 0 が付きます。 • l または L: ミリ秒。先頭に 0 が付きません。 • t: 午前および午後を表す 1 文字の時刻マーカー文字列 (A および P など)。 • tt: 午前および午後を表す複数文字の時刻マーカー文字列 (AM および PM など)。 • short: h:mm tt と同等。 • medium: h:mm:ss tt と同等。 • long: medium の内容の後に 3 文字のタイムゾーン。例 : 2:34:55 PM EST • full: long と同じ。

使用方法

日付時刻値を文字列として渡すときは、その値を引用符で囲みます。引用符で囲まない場合、その値は日付時刻オブジェクトの数値表現として解釈されます。

日付および時刻の形式を設定する関数を使用しないと、データベースクエリーの結果の日付時刻値は、その順序や形式がばらばらになる可能性があります。日付や時刻を適切な形式で表示し、アプリケーションのユーザーに混乱を与えないようにするには、DateFormat 関数と TimeFormat 関数を使用して、クエリーからの日付時刻値を形式設定することをお勧めします。詳細と例については、go.adobe.com/kb/ts_tn_18070_en-us のテクニカルノート「**ColdFusion (5 and 4.5.x) with Oracle: Formatting Date and Time Query Results**」を参照してください。

例

```
<cfset todayDate = #Now()#>
<body>
<h3>TimeFormat Example</h3>
<p>Today's date is <cfoutput>#todayDate#</cfoutput>.
<p>Using Timeformat, we can display the value in different ways:
<cfoutput>
<ul>
    <li>#TimeFormat(todayDate)#
    <li>#TimeFormat(todayDate, "hh:mm:ss")#
    <li>#TimeFormat(todayDate, "hh:mm:sst")#
    <li>#TimeFormat(todayDate, "hh:mm:sstt")#
    <li>#TimeFormat(todayDate, "HH:mm:ss")#
</ul>
</cfoutput>
<p>To generate a standard ISO 8601 W3C Date and Time string like
1997-07-16T19:20, concatenate a DateFormat function, the character T, and a
TimeFormat function.
For example: dateformat(now(), "yyyy-mm-dd")#T#TimeFormat(now(), "HH:mm:ss")
produces:</p>
<cfoutput>#dateformat(now(), "yyyy-mm-dd")#T#TimeFormat(now(), "HH:mm:ss")#</cfoutput>
</body>
```

ToBase64

説明

文字列またはバイナリオブジェクトを Base64 表現に変換します。Base64 表現は印刷可能な文字を使用する形式で、バイナリデータをフォームや電子メールで送信したり、データベースやファイルに保存することができます。

戻り値

文字列またはバイナリオブジェクトの Base64 表現。

カテゴリ

変換関数、文字列関数

関数のシンタックス

```
ToBase64(string or binary_object[, encoding])
```

関連項目

- BinaryEncode (バイナリデータから base64 への変換について)
- cffile (バイナリデータのロードと読み込みについて)
- cfwddx (バイナリデータのシリアル化とシリアル化解除について)
- IsBinary および ToBinary (バイナリデータかどうかの確認と、Base 64 オブジェクトのバイナリ形式への変換について)

履歴
ColdFusion MX: **encoding** パラメータが追加されました。

パラメータ

パラメータ	説明
string または binary_object	文字列、文字列名、またはバイナリオブジェクトです。
encoding	文字列の場合は、バイト配列内での文字の表現方法を指定します。一般的に使用される値を次に示します。 • utf-8 • iso-8859-1 • windows-1252 • us-ascii • shift_jis • iso-2022-jp • euc-jp • euc-kr • big5 • euc-cn • utf-16 文字エンコードの詳細については、www.w3.org/International/O-charset.html を参照してください。 デフォルト値は、この関数を呼び出すページのエンコードです。cfcontent を参照してください。バイナリオブジェクトの場合は、このパラメータは無視されます。

使用方法
新しく開発するアプリケーションでバイナリデータを Base64 エンコードデータに変換する場合は、BinaryEncode 関数を使用することをお勧めします。

```
<h3>ToBase64 Example</h3>
<!--- Initialize data. ---->
<cfset charData = "">
<!--- Create string of ASCII characters (32-255); concatenate them --->
<cfloop index = "data" from = "32" to = "255">
    <cfset ch = chr(data)>
    <cfset charData = charData & ch>
</cfloop>
<p>
The following string is the concatenation of all characters (32 to 255)
    from the ASCII table.<br>
<cfoutput>#charData#</cfoutput>
</p>
<!--- Create a Base64 representation of this string. ---->
<cfset data64 = toBase64(charData)>

<!----- Convert string to binary. ------->
<cfset binaryData = toBinary(data64)>
<!--- Convert binary back to Base64. ---->
<cfset another64 = toBase64(binaryData)>
<!---- Compare another64 with data64 to ensure that they are equal. ---->
<cfif another64 eq data64>
    <h3>Base64 representations are identical.</h3>
<cfelse>
    <h3>Conversion error.</h3>
</cfif>
```

ToBinary

説明

Base64 でエンコードされたデータまたは PDF ドキュメントをバイナリ形式に変換します。

戻り値

データをバイナリ形式に変換したもの

カテゴリ

変換関数、文字列関数

関数のシンタックス

```
ToBinary(Data)
```

関連項目

- BinaryDecode (Base64 形式などでエンコードされたデータからバイナリデータへの変換について)
- cffile (バイナリデータのロードと読み込みについて)
- cfwddx (バイナリデータのシリアル化とシリアル化解除について)
- IsBinary および ToBase64 (バイナリデータかどうかの確認と、Base 64 への変換について)
- バイナリオブジェクトの長さを調べる方法については、Len を参照してください。
- 『ColdFusion アプリケーションの開発』の Binary data type and binary encoding

パラメータ

パラメータ	説明
Data	Base64 エンコード形式のデータを表す変数または PDF ドキュメントです。

使用方法

ToBinary 関数は、PDF ドキュメント変数をパラメータとして使用 (cfpdf タグの name 属性で指定) できます。この場合、ToBinary 関数はドキュメントのバイト配列 (byte[]) 表示を返します。この関数の結果を使用して、PDF をデータベースに BLOB として保管したり、cfcontent タグ内で PDF をブラウザに書き出したりできます。このバイナリ表現を cfpdf タグの読み込み操作に使用して、変数を作成できます。

次の例では、保護されていない PDF ファイルを読み込み、保護を適用して、ブラウザに表示します。

```
<cfpdf action="read" source="Copy of coldfusion11.pdf" name="p">
<cfpdf action="protect" source="p" newUserpassword="user" permissions="none"
        newOwnerpassword="owner">
<cfcontent type="application/pdf" variable="#tobinary(p)#">
```

新しく開発するアプリケーションで Base64 エンコードデータをバイナリデータに変換する場合は、BinaryDecode 関数を使用することをお勧めします。

バイナリ値をこの関数に渡すと、関数は入力値を返します。

例

```
<h3>ToBinary Example</h3>
<!---- Initialize data. ---->
<cfset charData = "">
<!---- Create a string of ASCII characters (32-255); concatenate them. ---->
<cfloop index = "data" from = "32" to = "255">
    <cfset ch = chr(data)>
    <cfset charData = charData & ch>
</cfloop>
<p>The following string is the concatenation of all characters (32 to 255)
    from the ASCII table.<br>
<cfoutput>#charData#</cfoutput></p>
<!----- Create a Base64 representation of this string. ----->
<cfset data64 = toBase64(charData)>

<!--- Convert string to binary. ---->
<cfset binaryData = toBinary(data64)>
<!--- Convert binary back to Base64. --->
<cfset another64 = toBase64(binaryData)>
<!---- Compare another64 with data64 to ensure that they are equal. ---->
<cfif another64 eq data64>
    <h3>Base64 representation of binary data is identical to the Base64
    representation of string data.</h3>
<cfelse>
    <h3>Conversion error.</h3>
</cfif>
```

ToScript

説明

ColdFusion 変数の値を JavaScript または ActionScript 変数に割り当てる、JavaScript または ActionScript 式を作成します。この関数を使用して、ColdFusion 文字列、数値、配列、構造体、およびクエリーを、同等の変数および値を定義する JavaScript または ActionScript シンタックスに変換することができます。

戻り値
指定された ColdFusion 変数値に対応する JavaScript または ActionScript 変数定義を含む文字列

カテゴリ
変換関数、拡張関数

関数のシンタックス
```
ToScript(cfvar, javascriptvar, outputformat, ASFormat)
```

関連項目
cfwddx、WDDX JavaScript オブジェクト

履歴
ColdFusion MX 7: この関数が追加されました。

パラメータ

パラメータ	説明
cfvar	ColdFusion 変数です。次のいずれかを含めることができます。 • 文字列 • 数値 • 配列 • 構造体 • クエリー
javascriptvar	ToScript 関数によって作成される JavaScript 変数の名前を指定する文字列です。
outputformat	オプション。構造体およびクエリーの場合に、WDDX (JavaScript) 方式と ActionScript 方式のどちらを使用して出力を作成するかを指定するブール値です。 • true: WDDX 方式の出力を作成します (デフォルト)。 • false: ActionScript 方式の出力を作成します。
ASFormat	オプション。スクリプトで ActionScript の省略形を使用するかどうかを指定するブール値です。 • true: ActionScript の省略形を使用して新しいオブジェクトまたは配列を作成します。New Array() に対しては [] を、New Object に対しては {} を使用します。ActionScript の省略形を使用すると、ActionScript 検証を実行しなくても、ActionScript を cfform 属性に渡すことができます。 • false: スクリプトの生成時に ActionScript の省略形を使用しないで新しいオブジェクトまたは配列を作成します。代わりに、スクリプトで New Object() や New Array() を生成します (デフォルト)。

使用方法
JavaScript または ActionScript で ColdFusion 変数を使用するには、ToScript 関数が cfoutput 領域内にあり、シャープ記号 (#) で囲まれている必要があります。たとえば、次のコードは、ToScript 関数を使用して ColdFusion 変数を JavaScript 変数に変換します。

```
<cfset thisString="hello world">
<script type="text/javascript" language="JavaScript">
    <cfoutput>
        var #toScript(thisString, "jsVar")#;
    </cfoutput>
</script>
```

ColdFusion は、このコードを実行するときに次のコードをクライアントに送信します。

```
<script type="text/javascript" language="JavaScript">
    var jsVar = "hello world";
</script>
```

HTML script タグで JavaScript コードを囲む必要があります。cfoutput タグは、script ブロックの内部で使用する必要はありません。このタグでブロックを囲むこともできます。

WDDX 方式の出力により、WDDXRecordset オブジェクトを作成する JavaScript コードが生成されます。各レコードセットエントリのキーは列名であり、レコードリストエントリの値は対応するクエリー列エントリの配列です。次のようになります。

```
WDDXQuery = new WddxRecordset();
col0 = new Array();
col0[0] = "John";
col0[1] = "John";
WDDXQuery["firstname"] = col0;
col0 = null;
col1 = new Array();
col1[0] = "Lund";
col1[1] = "Allen";
WDDXQuery["lastname"] = col1;
col1 = null;
```

WDDX 方式の出力を使用するには、最初に "<ColdFusion の Web ルート >/CFIDE/scripts/wddx.js" スクリプトをロードします。このスクリプトでは、次に示すように JavaScript WDDX オブジェクトが定義されています。

```
<script type="text/javascript" src="/CFIDE/scripts/wddx.js"> </script>
```

JavaScript での WDDX の詳細については、1638 ページの「WDDX JavaScript オブジェクト」を参照してください。

ActionScript 方式の出力により、オブジェクトの配列を作成するコードが生成されます。配列は行番号でインデックス付けされ、オブジェクトは列名と列値のペアで構成されます。次のようになります。

```
    ActionScriptQuery = new Array();
ActionScriptQuery[0] = new Object();
ActionScriptQuery[0]['firstname'] = "John";
ActionScriptQuery[0]['lastname'] = "Lund";
ActionScriptQuery[1] = new Object();
ActionScriptQuery[1]['firstname'] = "John";
ActionScriptQuery[1]['lastname'] = "Allen";
```

ActionScript 方式の配列では、ユーザーが "wddx.js" ファイルを含める必要はなく、Flash 形式フォーム上の ActionScript (たとえば onChange 属性) で使用可能な変数が作成されます。

outputformat パラメータが false の場合に ASFormat を true に設定すると、ToScript は NewArray() の代わりに ActionScript の省略形 [] を使用し、NewObject() の代わりに省略形 {} を使用します。これらの省略形を使用すると、ActionScript 検証を実行しなくても、ActionScript を cfform 属性に渡すことができます。ASFormat が false の場合、toScript はスクリプトに NewArray() と NewObject() を生成します。

例

次の例では、ColdFusion 文字列、配列、クエリーオブジェクトを JavaScript 変数に変換した結果を示します。JavaScript コードの文字列および配列も使用します。

```
 <h2>ToScript</h2>

<h3>Converting a string variable</h3>
<cfset thisString = "This is a string">
<cfoutput>
    <b>The thisString variable in ColdFusion</b><br>
    #thisString#<br>
    <br>
    <strong>The output of ToScript(thisString, "jsVar")</strong><br>
    #ToScript(thisString, "jsVar")#<br>
    <br>
    <strong>In a JavaScript script, convert thisString Variable to JavaScript
     and output the resulting variable:</strong><br>
    <script type="text/javascript" language="JavaScript">
        var #ToScript(thisString, "jsVar")#;
        document.write("jsVar in JavaScript is: " + jsVar);
    </script>
</cfoutput>

<h3>Converting an array</h3>
<!--- Create and populate a one-dimensional array --->
<cfset myArray=ArrayNew(1)>
<cfloop index="i" from="1" to="4">
            <cfset myArray[i]="This is array element" & i>
</cfloop>

<cfoutput>
<b>The ColdFusion myArray Array</b><br>
<!--- Write the contents of the myArray variable in ColdFusion --->
    <cfloop index="i" from="1" to="#arrayLen(myArray)#">
        myArry[#i#]: #myArray[i]#<br>
    </cfloop>
    <br>
    <strong>The output of ToScript(myArray, "jsArray")</strong><br>
    #toScript(myArray, "jsArray")#<br>
    <br>
    <strong>In JavaScript, convert myArray to a JavaScript variable and write it's contents</strong><br>
    <script type="text/javascript" language="JavaScript">
        var #ToScript(myArray, "jsArray")#;
        for (i in jsArray)
        {
            document.write("myArray[" + i + "]: " + jsArray[i] + "<br>");
        }
    </script>
<br>
<h3>Converting a query</h3>
This section converts the following query object to both WDDX format
and ActionScript type JavaScript objects.<br>

<!--- Query a database --->
<cfquery name="thisQuery" datasource="cfdocexamples">
    SELECT FirstName,LastName
    FROM employee
    WHERE FirstName = 'John'
</cfquery>
<br>
```

```
The Query in ColdFusion
<cftable query="thisQuery" headerlines="1" colheaders>
    <cfcol align="left" width="9" header="<b>FirstName</b>" text="#FirstName#">
    <cfcol align="left" width="9" header="<b>LastName</b>" text="#LastName#">
</cftable>

<strong>JavaScript generated by ToScript(thisQuery, "WDDXQuery"):</strong><br>
    #toScript(thisQuery, "WDDXQuery")#;<br>
    <br>
<strong>JavaScript generated by ToScript(thisQuery, "ActionScriptQuery",
            False):</strong><br>
    #toScript(thisQuery, "ActionScriptQuery", False)#<br>
    <br>
<!--- Convert to both WDDX format and ActionScript format --->
<script type="text/javascript" language="JavaScript">
    #ToScript(thisQuery, "WDDXQuery")#;
    #ToScript(thisQuery, "ActionScriptQuery", False)#;
</script>
<!--- For brevity, this example does not use JavaScript query variables --->
</cfoutput>
```

ToString

説明
値を文字列に変換します。

戻り値
文字列です。

カテゴリ
変換関数、文字列関数

関数のシンタックス
```
ToString(value[, encoding])
```

関連項目
ToBase64、ToBinary、CharsetEncode、『ColdFusion アプリケーションの開発』の Using XML and WDDX

履歴
ColdFusion MX:

- Unicode サポートが変更されました。Unicode 文字値 0 ～ 65535 の Java UCS-2 表現がサポートされるようになりました (ColdFusion 5 以前のリリースでは ASCII 値 1 ～ 255 がサポートされていました)。
- **encoding** パラメータが追加されました。
- XML ドキュメントオブジェクトを文字列に変換できるようになりました。

パラメータ

パラメータ	説明
value	文字列に変換する値です。整数、バイナリオブジェクト、XML ドキュメントオブジェクトなどの単純値です。
encoding	文字列の文字エンコード (文字セット) を指定します。バイナリデータ用のオプションであり、単純値や XML ドキュメントオブジェクトに使用するとエラーが生成されます。 一般的に使用される値を次に示します。 • utf-8 • iso-8859-1 • windows-1252 • us-ascii • shift_jis • iso-2022-jp • euc-jp • euc-kr • big5 • euc-cn • utf-16 文字エンコードの詳細については、www.w3.org/International/O-charset.html を参照してください。 デフォルト値は、この関数を呼び出すページのエンコードです。cfcontent を参照してください。

使用方法

この関数では、単純値と、値 0 のバイトを含まないバイナリ値を変換できます。この関数で変換できない値の場合は例外が発生します。この関数では、XML ドキュメントオブジェクトを XML 文字列表現に変換することもできます。

バイナリデータを文字列に変換する場合は、CharsetEncode 関数を使用することをお勧めします。

例

```
<h3>ToString Example</h3>
<!---- Initialize data. ------>
<cfset charData = "">
<!----- Create string of ASCII characters (32-255) and concatenate them. ---->
<cfloop index = "data" from = "32" to = "255">
    <cfset ch = chr(data)>
    <cfset charData = charData & ch>
</cfloop>
<p>The following string is the concatenation of characters (32 to 255)
    from the ASCII table.<br>
<cfoutput>#charData#</cfoutput></p>

<!------ Create a Base64 representation of this string. ---->
<cfset data64 = toBase64(#charData#)>
<p>
The following string is the Base64 representation of the string.<br>
<cfoutput>#data64#</cfoutput></p>
<!---- Create a binary representation of Base64 data. --->
<cfset dataBinary = toBinary(data64)>

<!---- Create the string representation of the binary data. ----->
<cfset dataString = ToString(dataBinary)>
<p>The following is the string representation of the binary data.<br>
<cfoutput>#dataString#</cfoutput></p>
```

Trace

説明

<cftrace> タグと同様の関数で、<cfscript> モードで使用されます。

パラメータ

<cftrace> タグと同じです。

カテゴリ

デバッグ関数

関数のシンタックス

```
trace (var, text, type, category, inline, abort)
```

定位置表記の場合は、シンタックスで示した順序どおりに値を指定する必要があります。指定しないパラメータがある場合は、代わりに空の文字列を使用します。これはブール値には適用されません。ブール値の場合は、省略する必要がある場合でも適切な値を指定する必要があります。

関連項目

cfscript、cftrace

使用方法

この関数は、name=value のペアまたは定位置引数として呼び出すことができます。

例

```
<cfscript>
    function TotalInterest(principal, annualRate, months) {
    var years = 0;
    var interestRate = 0;
    var totalInterest = 0;
    principal = REReplace(trim(principal), "[\$]", "", "ALL");
    annualRate = Replace(trim(annualRate), "%", "", "ALL");

    if ((principal <= 0) OR (annualRate <= 0) OR (months <= 0)) {
        Throw(type="InvalidData",message="All values must be greater than 0.");
    }
    interestRate = annualRate / 100;
    years = months / 12;
    totalInterest = principal * (((1 + interestRate) ^ years) - 1);
    return DollarFormat(totalInterest);
}
try {
    Trace(type="Information", inline="true", text="Calculating interest."); //Use of
    trace function in cfsript
    WriteOutput(TotalInterest("$2500.00", "5.5%", "12"));
    Trace(type="Information", inline="true", text="Interest calculation done.");
    }

catch(InvalidData ex) {
    //Displayig exception details on screen
    WriteOutput("<p>An InvalidData exception was thrown.</p>");
    WriteOutput("<p>#ex.message#</p>");
    //Writting the exception to log file under logs folder of web server.
    WriteLog(type="Error", file="myapp.log", text="[#ex.type#] #ex.message#");
}
```

TransactionCommit

説明

現在アクティブなトランザクションをコミットします。

戻り値

なし

カテゴリ

718 ページの「トランザクション関数」

関数のシンタックス

TransactionCommit()

関連項目

TransactionRollback、TransactionSetSavePoint

履歴

ColdFusion 9: この関数が追加されました。

使用方法

この関数は、アクティブなトランザクションの内部からのみ呼び出すことができます。

例

```
<cfscript>
q = new query();
q.setDatasource("cfartgallery");
WriteDump(q.execute(sql="select * from art where artid = 60").getResult());
transaction
{
        q.execute(sql="insert into art (artid,artistid,artname,description,price) values
(60,3,'tom','tom',2000)");
        transactionSetSavePoint('sp1');
        WriteDump(q.execute(sql="select * from art where artid = 60").getResult());
        transaction
        {
                q.execute(sql="update art set artistid=4 where artid = 60");
                transactionSetSavePoint('sp2');
                WriteDump(q.execute(sql="select * from art where artid = 60").getResult());
                transaction
                {
                    try
                    {
                        q.execute(sql="update art set artistid='badvalue' where artid = 60");
                    }
                    catch(any e)
                    {
                        WriteLog("rolling back the transaction");
                        transactionRollback("sp1");
                    }
                }
        }
}
WriteDump(q.execute(sql="select * from art where artid = 60").getResult());
transaction
{
    WriteLog("deleting the record");
    q.execute(sql="delete from art where artid = 60");
    WriteDump(q.execute(sql="select * from art where artid = 60").getResult());
}
</cfscript>
```

TransactionRollback

説明

現在アクティブなトランザクションを、指定したセーブポイントまでロールバックします。

戻り値

なし

カテゴリ

718 ページの「トランザクション関数」

関数のシンタックス

TransactionRollback([savepoint])

関連項目

TransactionCommit、TransactionSetSavePoint

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
savepoint	セーブポイントを指定するオプションの文字列識別子です。セーブポイントを使用すると、トランザクションを部分的にロールバックできます。

使用方法
この関数は、アクティブなトランザクションの内部からのみ呼び出すことができます。

セーブポイントを指定しないと、現在アクティブなトランザクションはトップレベルの (元の) トランザクションまでロールバックされます。

例
TransactionCommit の例を参照してください。

TransactionSetSavePoint

説明
現在のトランザクションの新しいセーブポイントを作成して保存します。この関数を繰り返し呼び出すことで、複数のセーブポイントを追加できます。

戻り値
なし

カテゴリ
718 ページの「トランザクション関数」

関数のシンタックス
TransactionSetSavepoint(savepoint)

関連項目
TransactionCommit、TransactionRollback

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
savepoint	セーブポイントを指定するオプションの文字列識別子です。セーブポイントを使用すると、トランザクションを部分的にロールバックできます。

使用方法
この関数は、アクティブなトランザクションの内部からのみ呼び出すことができます。

例
TransactionCommit の例を参照してください。

Trim

説明
文字列の先頭と末尾のスペースと制御文字を削除します。

戻り値
string パラメータから先頭と末尾のスペースと制御文字を削除した結果のコピー

カテゴリ
文字列関数

関数のシンタックス
```
Trim(string)
```

関連項目
LTrim、RTrim

パラメータ

パラメータ	説明
string	文字列、または文字列を含む変数です。

例
```
<h3>Trim Example</h3>
<cfif IsDefined("FORM.myText")>
    <cfoutput>
        <pre>
            Your string:"#FORM.myText#"
            Your string:"#Trim(FORM.myText)#"
            (trimmed on both sides)
        </pre>
    </cfoutput>
</cfif>
<form method = "post" action = "trim.cfm">
<p>Type in some text, and it will be modified by trim to remove leading
    spaces from the left and right
<p><input type = "Text" name = "myText" value = " TEST ">
<p><input type = "Submit" name = "">
</form>
```

UCase

説明
文字列のアルファベット文字を大文字に変換します。

戻り値
大文字に変換された文字列のコピー

カテゴリ
文字列関数

関数のシンタックス
```
UCase(string)
```

関連項目
LCase

パラメータ

パラメータ	説明
string	文字列、または文字列を含んでいる変数です。

例
```
<h3>UCase Example</h3>

<cfif IsDefined("FORM.sampleText")>
    <cfif FORM.sampleText is not "">
        <p>Your text, <cfoutput>#FORM.sampleText#</cfoutput>,
    returned in uppercase is <cfoutput>#UCase(FORM.sampleText)#</cfoutput>.
    <cfelse>
        <p>Please enter some text.
    </cfif>
</cfif>

<form action = "ucase.cfm">
<p>Enter your sample text, and press "submit" to see the text returned in
    uppercase:
<p><input type = "Text" name = "SampleText" value = "sample">

<input type = "Submit" name = "" value = "submit">
</form>
```

URLDecode

説明
URL エンコード文字列をデコードします。

戻り値
デコードされた文字列

カテゴリ
変換関数、その他の関数、文字列関数

関数のシンタックス
```
URLDecode(urlEncodedString[, charset])
```

関連項目
URLEncodedFormat、『ColdFusion アプリケーションの開発』の Tags and functions for globalizing applications

履歴

ColdFusion MX 6.1: デフォルトの文字セットが変更されました。URL スコープの文字エンコードがデフォルトの文字セットとなりました。

ColdFusion MX:

- Unicode サポートが変更されました。Unicode 文字値 0 ～ 65535 の Java UCS-2 表現がサポートされるようになりました (以前のリリースでは ASCII 値がサポートされていました)。
- charset パラメータが追加されました。

パラメータ

パラメータ	説明
urlEncodedString	URL エンコードされた文字列、またはそのような文字列を含んでいる変数です。
charset	URL のエンコードに使用する文字エンコードです。オプション 一般的に使用される値を次に示します。 • utf-8 • iso-8859-1 • windows-1252 • us-ascii • shift_jis • iso-2022-jp • euc-jp • euc-kr • big5 • euc-cn • utf-16 文字エンコードの詳細については、www.w3.org/International/O-charset.html を参照してください。 デフォルトは、URL スコープの文字エンコードです。

使用方法

URL エンコードでは、一部の文字が、パーセント記号と 16 進表記を使って表現されます。たとえば、コード 129 の文字は %81 にエンコードされます。また、スペースはプラス記号にエンコードされます。

HTTP のクエリー文字列は常に URL エンコードされています。

例

この例では、ASCII 文字コードを含んでいる文字列の作成、エンコード、およびデコードを行います。

```
<cfscript>
    // Build string
    s = "";
    for (c = 1; c lte 256; c = c + 1)
    {
        s = s & chr(c);
    }
    // Encode string and display result
    enc = URLEncodedFormat(s);
    WriteOutput("Encoded string is: '#enc#'.<br>");
    // Decode and compare result with original
    dec = URLDecode(enc);
    if (dec neq s)
    {
        WriteOutput("Decoded is not the same as encoded.");
    }
    else
    {
        WriteOutput("All's quiet on the Western front.");
    }
</cfscript>
```

URLEncodedFormat

説明
文字列を URL エンコードします。たとえば、スペースは %20 に置き換えられ、英数字以外の文字はその文字に相当する 16 進数のエスケープシーケンスに置き換えられます。これにより、任意の文字列を URL 文字列に含めて送信できます (ColdFusion は、ページに渡される URL パラメータを自動的にデコードします)。

戻り値
URL エンコードされた文字列のコピー

カテゴリ
変換関数、その他の関数、文字列関数

関数のシンタックス
```
URLEncodedFormat(string [, charset ])
```

関連項目
URLDecode、『ColdFusion アプリケーションの開発』の Tags and functions for globalizing applications

履歴
ColdFusion MX 6.1: デフォルトのエンコードが、レスポンスの文字エンコードに変更されました。

ColdFusion MX: charset パラメータが追加されました。

パラメータ

パラメータ	説明
string	文字列、または文字列を含んでいる変数です。
charset	文字列のエンコードに使用する文字エンコードです。オプション 一般的に使用される値を次に示します。 • utf-8 • iso-8859-1 • windows-1252 • us-ascii • shift_jis • iso-2022-jp • euc-jp • euc-kr • big5 • euc-cn • utf-16 文字エンコードの詳細については、www.w3.org/International/O-charset.html を参照してください。 デフォルトは、レスポンスの文字エンコードです。cfcontent を参照してください。

使用方法

URL エンコードでは、一部の文字が、パーセント記号と 16 進表記を使って表現されます。たとえば、コード 129 の文字は %81 にエンコードされます。スペースは %20 に置換されます。

HTTP のクエリー文字列は常に URL エンコードされています。

例

```
<h3>URLEncodedFormat Example</h3>
<cfif IsDefined("url.myExample")>
    <p>The url variable url.myExample was passed from the previous link ...
    its value is:
    <br><b>"<cfoutput>#url.myExample#</cfoutput>"</b>
</cfif>
<p>This function returns a URL encoded string.
<cfset s = "My url-encoded string has special characters & other stuff">
<p> <A HREF = "urlencodedformat.cfm?myExample=<cfoutput>#URLEncodedFormat(s)#
</cfoutput>">Click me</A>
```

URLSessionFormat

説明

クライアントコンピュータが Cookie を受け入れるかどうかに応じて、この関数は次の動作をします。

- クライアントが Cookie を受け入れない場合は、必要なすべてのクライアント ID 情報を URL に自動的に追加します。
- クライアントが Cookie を受け入れる場合は、情報を追加しません。

この関数は、どの識別子が必要であるかを自動的に判別し、必要な情報だけを送信します。これにより、各 URL 内の情報を手動でエンコードする場合よりも、安全かつ信頼性のある方法でクライアントを識別できます。必要なときに必要な情報だけが送信され、またコーディングが容易です。

戻り値
URL。ブラウザで Cookie が無効化されている場合は、クライアントとセッションのデータが追加されます。

カテゴリ
その他の関数、『ColdFusion アプリケーションの開発』の Maintaining client identity

関数のシンタックス
```
URLSessionFormat(request_URL)
```

パラメータ

パラメータ	説明
request_URL	ColdFusion ページの URL です。

使用方法
次の例では、cfform タグが他のページにリクエストを送信し、必要であればそのクライアントの識別情報を送信します。Cookie を受け入れるようになっていることが検出された場合は、この関数は次の値を返します。

```
myactionpage.cfm
```

Cookie を受け入れないようになっていることが検出された場合、および Cookie を受け入れるかどうかが明確に検出できなかった場合は、この関数は次の値を返します。

```
myactionpage.cfm?jsessionid=xxxx;cfid=xxxx&cftoken=xxxxxxxx
```

例
```
<cfform
    method="Post"
    action="#URLSessionFormat("MyActionPage.cfm")#">
</cfform>
```

Val

説明
文字列の先頭にある数字を数値に変換します。

戻り値
数値。変換に失敗した場合は 0 が返されます。

カテゴリ
変換関数、文字列関数

関数のシンタックス
```
Val(string)
```

関連項目
IsNumeric

パラメータ

パラメータ	説明
string	文字列、または文字列を含んでいる変数です。

使用方法

この関数は、次のように処理を行います。

- TestValue = "234A56?7'" の場合、Val(TestValue) は 234 を返します。
- TestValue = "234'5678'9?'" の場合、Val(TestValue) は 234 を返します。
- TestValue = "BG234" の場合、Val(TestValue) は値 0 を返します (エラーではありません)。
- TestValue = "0" の場合、Val(TestValue) は値 0 を返します (エラーではありません)。

例

```
<h3>Val Example</h3>
<cfif IsDefined("FORM.theTestValue")>
    <cfif Val(FORM.theTestValue) is not 0>
        <h3>The string <cfoutput>#DE(FORM.theTestValue)#</cfoutput>
    can be converted to a number:
        <cfoutput>#Val(FORM.theTestValue)#</cfoutput></h3>
    <cfelse>
        <h3>The beginning of the string <cfoutput>#DE(FORM.theTestValue)#
         </cfoutput> cannot be converted to a number</h3>
    </cfif>
</cfif>
<form action = "val.cfm">
<p>Enter a string, and determine whether its beginning can be evaluated
    to a numeric value.
<p>
<input type = "Text"
    name = "TheTestValue"
    value = "123Boy">
<input type = "Submit"
    value = "Is the beginning numeric?"
    name = "">
</form>
```

ValueList

説明

実行したクエリーの各値の間に区切り文字を挿入します。引数の評価は行われません。

戻り値

実行したクエリーから返された各レコードの値を、区切り文字で区切ったリスト

カテゴリ

リスト関数、クエリー関数

関数のシンタックス

```
ValueList(query.column [, delimiter ])
```

関連項目
QuotedValueList

パラメータ

パラメータ	説明
query.column	実行するクエリーと列の名前です。クエリー名と列名の間はピリオド (.) で区切ります。
delimiter	列データ項目を区切るための区切り文字です。デフォルト値はカンマ (,) です。

例
```
<h3>ValueList Example</h3>

<!--- use the contents of a query to create another dynamically --->
<cfquery name = "GetDepartments" datasource = "cfdocexamples">
    SELECT Dept_ID FROM Departments
    WHERE Dept_ID IN ('BIOL')
</cfquery>

<cfquery name = "GetCourseList" datasource = "cfdocexamples">
    SELECT *
    FROM CourseList
    WHERE Dept_ID IN ('#GetDepartments.Dept_ID#')
</cfquery>

Value list of all BIOL Course ID's using (--) as the delimiter:<br>
<cfoutput>
#ValueList(GetCourseList.Course_ID,"--")#<br>
</cfoutput>

Value list of all BIOL Course Numbers using (;) as the delimiter:<br>
<cfoutput>
#ValueList(GetCourseList.CorNumber,";")#<br>
</cfoutput>
```

VerifyClient

説明
ページのリモート呼び出しまたはページでの関数呼び出しに対して、暗号化されたセキュリティトークンの使用を義務付けます。

戻り値
なし

カテゴリ
セキュリティ関数

関数のシンタックス
```
VerifyClient()
```

関連項目
cffunction、『ColdFusion アプリケーションの開発』の Improving security

履歴
ColdFusion 8: この関数が追加されました。

パラメータ
パラメータは取りません。

使用方法
この関数を使用すると、権限のない第三者がサーバーに対してアクション (パスワードの変更など) を実行しようとするセキュリティアタックの防止に役立ちます。通常は、機密性の高いアクション (パスワードの更新など) をサーバーに対して実行する Ajax リクエストでこの機能を使用します。

この関数を呼び出す場合は、アプリケーションでのクライアント管理またはセッション管理を有効にする必要があります。有効でない場合、エラーは発生しませんが、ColdFusion によるクライアントの検証は行われません。この関数は、バインド式など、クライアントサイドの ColdFusion の AJAX 機能に応答するページでのみ使用します。これらの機能には、必要な場合にセキュリティトークンを正しく送信するコードが含まれています。

Week

説明
日付時刻オブジェクトから、その年の第何週かを調べます。

戻り値
その年の第何週かを示す 1 ～ 53 の範囲の整数

カテゴリ
日付および時刻関数

関数のシンタックス
```
Week(date)
```

関連項目
DatePart

パラメータ

パラメータ	説明
date	日付時刻オブジェクトです。値の範囲は西暦 100 ～ 9999 年です。

使用方法
日付を文字列として渡すときは、その値を引用符で囲む必要があります。引用符で囲まない場合、その値は日付の数値表現として解釈されます。

例

```
<h3>Week Example</h3>
<cfif IsDefined("FORM.year")>
More information about your date:
<cfset yourDate = CreateDate(FORM.year, FORM.month, FORM.day)>
<cfoutput>
    <p>Your date, #DateFormat(yourDate)#.
    <br>It is #DayOfWeekAsString(DayOfWeek(yourDate))#, day
         #DayOfWeek(yourDate)# in the week.
    <br>This is day #Day(YourDate)# in the month of
         #MonthAsString(Month(yourDate))#, which has #DaysInMonth(yourDate)# days.
    <br>We are in week #Week(yourDate)# of #Year(yourDate)# (day
         #DayOfYear(yourDate)# of #DaysInYear(yourDate)#). <br>
    <cfif IsLeapYear(Year(yourDate))>This is a leap year
        <cfelse>This is not a leap year
    </cfif>
</cfoutput>
</cfif>
```

Wrap

説明

テキストを折り返して、各行の文字数が指定した最大文字数に収まるようにします。

注意：wrap 関数は、HTML テキストに
 タグを追加することによって改行を行うわけではありません。
 タグを追加せずに、表示テキストを折り返します。

戻り値

折り返したテキストが格納された文字列

カテゴリ

文字列関数

関数のシンタックス

```
Wrap(string, limit[, strip])
```

関連項目

cfmail

履歴

ColdFusion MX 6.1: この関数が追加されました。

パラメータ

パラメータ	説明
string	文字列、または文字列を含んでいる変数です。このテキストを折り返します。
limit	正の整数です。1 行の最大文字数を指定します。
strip	ブール値です。折り返し処理の前に、入力文字列内の既存の復帰改行文字をスペースとともに削除するかどうかを指定します。デフォルト値は false です。

使用方法

行の文字数が指定した最大数に達する位置からさかのぼって最初に出てくる空白文字 (スペース、タブ、改行など) の箇所に改行が挿入されます。最大文字数に達する位置の前に空白文字がない場合は、最大文字数の位置に改行が挿入されます。この関数の処理には、各オペレーティングシステム固有の改行文字が使われます。これは UNIX の場合は LF、Windows の場合は CR/LF です。

strip パラメータを指定した場合は、既存の改行文字はすべて削除されるので、段落の整形はすべて損なわれます。

この関数は、メールメッセージに含めるテキストなどで、各行の長さに上限を設けるときに使用します。cfmail および cfmailpart タグの wraptext 属性ではこの関数が使われます。

例

```
<h3>Wrap Example</h3>
<cfset inputText="This is an example of a text message that we want to wrap. It is rather long and needs to
be broken into shorter lines.">
<cfoutput>#Wrap(inputText, 59)#</cfoutput>
```

WriteDump

説明

<cfdump> タグと同様の関数で、<cfscript> で使用できます。

パラメータ

<cfdump> タグと同じです。

カテゴリ

その他の関数、データ出力関数

関数のシンタックス

```
WriteDump (var, output, format, abort, label, metainfo, top, show, hide, keys, expand, showUDFs);
```

関連項目

cfdump、cfscript

履歴

ColdFusion 9: この関数が追加されました。

使用方法

この関数を呼び出すには、name=value のペアまたは定位置引数として引数を指定します。

定位置表記の場合は、シンタックスで示した順序どおりに値を指定する必要があります。指定しないパラメータがある場合は、代わりに空の文字列を使用します。これはブール値には適用されません。ブール値の場合は、省略する必要がある場合でも適切な値を指定する必要があります。

例

```
<cfscript>
filename = "log.txt";
    try {
    result = FileOpen(expandpath(filename));
    WriteDump(result);
}
catch(Expression exception) {
    WriteOutput("<p>An Expression exception was thrown.</p>");
    WriteOutput("<p>#exception.message#</p>");
    WriteLog(type="Error", file="myapp.log", text="[exception.type]
    #exception.message#");
    }
</cfscript>
```

WriteLog

説明

cflog タグと同様の関数で、<cfscript> で使用できます。

パラメータ

<cflog> タグと同じです。

カテゴリ

データ出力関数

関数のシンタックス

```
WriteLog (text, type, application, file, log)
```

定位置表記の場合は、シンタックスで示した順序どおりに値を指定する必要があります。指定しないパラメータがある場合は、代わりに空の文字列を使用します。これはブール値には適用されません。ブール値の場合は、省略する必要がある場合でも適切な値を指定する必要があります。

関連項目

cfscript、cflog

使用方法

この関数は、name=value のペアまたは定位置引数として呼び出すことができます。

例

```
<cfscript>
    //Example: 1
    function TotalInterest(principal, annualRate, months) {
    var years = 0;
    var interestRate = 0;
    var totalInterest = 0;
    principal = REReplace(trim(principal), "[\$]", "", "ALL");
    annualRate = Replace(trim(annualRate), "%", "", "ALL");

    if ((principal <= 0) OR (annualRate <= 0) OR (months <= 0)) {
        Throw(type="InvalidData",message="All values must be greater than 0.");
    }

    interestRate = annualRate / 100;
    years = months / 12;
    totalInterest = principal * (((1 + interestRate) ^ years) - 1);
    return DollarFormat(totalInterest);
}

try {
    Trace(type="Information", inline="true", text="Calculating interest.");
    WriteOutput(TotalInterest("$2500.00", "5.5%", "12"));
    Trace(type="Information", inline="true", text="Interest calculation done.");
}

catch(InvalidData ex) {
    //Writting the exception to log file under logs folder of web server.
    WriteLog(type="Error", file="myapp.log", text="[#ex.type#] #ex.message#");
}
</cfscript>
```

WriteOutput

説明

ページ出力ストリームにテキストを追加します。

この関数は、cfsetting タグで設定された条件とは無関係に、ページ出力ストリームに対して書き出しを行います。

カテゴリ

その他の関数、システム関数、データ出力関数

関数のシンタックス

```
WriteOutput(string)
```

パラメータ

パラメータ	説明
string	文字列、または文字列を含んでいる変数です。

使用方法

cfquery タグと cfmail タグの中では、この関数の出力先は現在のページではなく、現在の SQL ステートメントまたはメール本文となります。cfquery タグと cfmail タグの中では、WriteOutput を使用しないでください。

この関数は、ページ内の任意の場所で呼び出すことができますが、cfscript ブロックで使用するのが最も有用です。

例

```
...
<cfscript>
employee = StructNew();
StructInsert(employee, "firstname", FORM.firstname);
StructInsert(employee, "lastname", FORM.lastname);
StructInsert(employee, "email", FORM.email);
StructInsert(employee, "phone", FORM.phone);
StructInsert(employee, "department", FORM.department);
WriteOutput("About to add " & FORM.firstname & " " & FORM.lastname);
</cfscript>
```

WsGetAllChannels

説明
フィルター条件（構造体です）に基づいて、特定のチャネルにメッセージを送信します。

戻り値
なし

関数のシンタックス
```
WSGetAllChannels()

WSGetAllChannels("channelName")
```

パラメータ

パラメータ	説明
channelName	指定した場合は、入力したチャネル名の下にリストされているすべてのサブチャネルが返されます。指定なしのままにした場合は、現在のアプリケーションの下に登録されているすべてのチャネルが返されます。

WsGetSubscribers

説明
clientID および subcriberInfo をキーとして構造体の配列を返します。

戻り値
なし

関数のシンタックス
```
WSgetSubscribers(channel)
```

パラメータ

パラメータ	説明
channel	サブスクライバのリストを取得するチャネルです。

WsPublish

説明
フィルター条件（構造体です）に基づいて、特定のチャネルにメッセージを送信します。

カテゴリ
その他の関数、システム関数、データ出力関数

関数のシンタックス
```
WSPublish(String channel, Object message)
WSPublish(channel,message [,filterCriteria])
```

パラメータ

パラメータ	説明
channel	サーバーが応答を公開する所定のチャネルです。
message	所定のチャネルにサブスクライブしているすべてのクライアントに対してサーバーから送信される応答です。
filterCriteria	所定のチャネルにおいて通知が必要な対象クライアントをフィルタリングする条件です。

例
```
<cfscript>
    if(isdefined("form.publish"))
        WsPublish(#form.channel#, #form.message#);
</cfscript>
<cfform method="post">
    <cfselect name="channel">
        <option>
            stocks
        </option>
        <option>
            news
        </option>
        <option>
            products
        </option>
    </cfselect>
    Message:
    <input id="message" name="message" type="text">
    <cfinput id="publish" name="publish" value="publish" type="submit">
</cfform>
```

WSSendMessage

説明
メソッドを呼び出した特定のクライアントにメッセージを送信します。これは、invoke WebSocket JavaScript メソッドによって呼び出された関数の一部として含めることができます。

戻り値
なし

シンタックス
WSSendMessage(message)

パラメータ

パラメータ	説明
message	必須。メッセージオブジェクトです。Any 型を指定できます。

XmlChildPos

説明
XML ドキュメントオブジェクト内の子要素の位置を取得します。

戻り値
XmlChildren 配列内で、指定された名前を持つ **N** 番目の子要素の位置

カテゴリ
XML 関数

関数のシンタックス
```
XmlChildPos(elem, childName, N)
```

関連項目
IsXmlElem、XmlElemNew、XmlSearch、XmlTransform、『ColdFusion アプリケーションの開発』の Using XML and WDDX

履歴
ColdFusion MX: この関数が追加されました。

パラメータ

パラメータ	説明
elem	XML 要素です。この中を検索します。
childName	検索する XML 子要素です。elem パラメータの直接の子でなければなりません。
N	検索する XML 子要素のインデックスです。

使用方法
ArrayInsertAt 関数と ArrayDeleteAt 関数の返されたインデックスを使用して、XML ドキュメントオブジェクトを変更することができます。指定された子が見つからない場合、この関数は -1 を返します。

例
次の例では、XML ドキュメント要素 xmlobject.employee.name[1] を、その 2 番目の Status 子要素について検索し、ArrayDeleteAt 関数内の位置を使用して要素を削除します。

```
<!--- Create an XML document object --->
<cfxml variable="xmlobject">
<employee>
    <!-- A list of employees -->
    <name EmpType="Regular">
        <first>Almanzo</first>
        <last>Wilder</last>
        <Status>Medical Absence</Status>
        <Status>Extended Leave</Status>
    </name>
    <name EmpType="Contract">
        <first>Laura</first>
        <last>Ingalls</last>
    </name>
</employee>
</cfxml>

<!--- Find the second Status child of the first employee.name element --->
<cfscript>
elempos=XMLChildPos(xmlobject.employee.name[1], "Status", 2);
ArrayDeleteAt(xmlobject.employee.name[1].XmlChildren, elempos);
</cfscript>

<!--- Dump the resulting document object to confirm the deletion --->
<cfdump var="#xmlobject#">
```

XmlElemNew

説明
XML ドキュメントオブジェクト要素を作成します。

戻り値
XML ドキュメントオブジェクト要素

カテゴリ
XML 関数

関数のシンタックス
```
XmlElemNew(xmlObj[, namespace], childName)
```

関連項目
cfxml、IsXmlElem、XmlChildPos、XmlFormat、XmlNew、XmlParse、『ColdFusion アプリケーションの開発』の Using XML and WDDX

履歴
ColdFusion MX 7: namespace パラメータが追加されました。

ColdFusion MX: この関数が追加されました。

パラメータ

パラメータ	説明
xmlObj	要素を作成する対象の XML ドキュメントオブジェクトの名前です。
namespace	(オプション) この要素が属する名前空間の URI です。
childName	作成する要素の名前です。

使用方法

この関数の戻り変数は、ドキュメントオブジェクト内の新規要素の位置を指定します。**xmlObj** パラメータで識別される、ドキュメントオブジェクト内の有効な位置を指定する必要があります。次のステートメントに、その使用方法を示します。

```
MyDoc.MyRoot.XmlChildren[2] = XmlElemNew(MyDoc,"childNode");
ArrayAppend(MyDoc.MyRoot.XmlChildren, XmlElemNew(MyDoc,"childNode"));
```

namespace で URI を指定しないで、childName パラメータで名前空間の接頭辞を使用した場合は、その接頭辞に対応する名前空間の URI が既に指定されているかどうかが確認され、指定されている場合はその名前空間が使用されます。

例

次の例では、ColdFusion ドキュメントオブジェクトを作成および表示します。

```
<cfscript>
    MyDoc = XmlNew();
    MyDoc.xmlRoot = XmlElemNew(MyDoc,"MyRoot");
    if (testVar IS TRUE)
        MyDoc.MyRoot.XmlText = "The value of testVar is True.";
    else
        MyDoc.MyRoot.XmlText = "The value of testVar is False.";
    for (i = 1; i LTE 4; i = i + 1)
        {
        MyDoc.MyRoot.XmlChildren[i] = XmlElemNew(MyDoc,"childNode");
        MyDoc.MyRoot.XmlChildren[i].XmlText = "This is Child node " & i &".";
        }
</cfscript>
<cfdump var=#MyDoc#>
```

XmlFormat

説明

文字列を XML 内でテキストとして使用できるように、文字列に含まれる XML の特殊文字をエスケープします。

戻り値

XML 形式のテキストとして安全に使用できる **string** パラメータのコピー

カテゴリ

文字列関数、XML 関数

関数のシンタックス

```
XmlFormat(string, escapeChars)
```

関連項目

cfxml、XmlNew、XmlParse、XmlValidate、『ColdFusion アプリケーションの開発』の Using XML and WDDX

履歴
ColdFusion MX: この関数が追加されました。

パラメータ

パラメータ	説明
string	文字列、または文字列を含んでいる変数です。
escapeChars	XML 標準で制限されている文字をエスケープする場合は、true に設定します。詳細については、http://www.w3.org/TR/2006/REC-xml11-20060816/#NT-RestrictedChar を参照してください。

使用方法
この関数は、次のような文字をエスケープ処理します。

テキスト文字	エスケープ表記
右不等号 (>)	&gt;
左不等号 (<)	&lt;
一重引用符 (')	&apos;
二重引用符 (")	&quot;
アンパサンド記号 (&)	&amp;
CR 文字 (LF 文字を除く)	テキストから削除
159 ～ 255 の範囲の High ASCII Characters (ハイアスキー文字)	Unicode エスケープシーケンスで置き換えられます。たとえば、É (アクセント記号付きの大文字 E) は É で置き換えられます。

例
次の例は、XmlFormat 関数が XML 文字をエスケープ処理する方法を示します。ブラウザの [表示]-[ソース] などのコマンドを使用して、結果を確認します。ColdFusion は、XmlFormat 関数の適用前は、2 番目のテキスト文字列内の "" を一重引用符を表すものと解釈します。

```
<?xml version = "1.0"?>
<cfoutput>
<someXML>
    <someElement someAttribute="#XmlFormat("'a quoted value'", "true")#">
        #XmlFormat("Body of element with <, >, "" and & goes here.", "true")#
    </someElement>
</someXML>
</cfoutput>
```

XmlGetNodeType

説明
XML ドキュメントオブジェクトノードのタイプを調べます。

戻り値
XML ノードタイプを識別する文字列。使用できる値は次のとおりです。

ATTRIBUTE_NODE	CDATA_SECTION_NODE
COMMENT_NODE	DOCUMENT_FRAGMENT_NODE
DOCUMENT_NODE	DOCUMENT_TYPE_NODE
ELEMENT_NODE	ENTITY_NODE
ENTITY_REFERENCE_NODE	NOTATION_NODE
PROCESSING_INSTRUCTION_NODE	TEXT_NODE

引数がドキュメントオブジェクトノードでない場合、この関数はエラーを生成します。

カテゴリ
XML 関数

関数のシンタックス
```
XmlGetNodeType(xmlNode)
```

関連項目
IsXmlAttribute、IsXmlDoc、IsXmlElem、IsXmlNode、IsXmlRoot、XmlChildPos、XmlValidate、『ColdFusion アプリケーションの開発』の Using XML and WDDX

履歴
ColdFusion MX 7: この関数が追加されました。

パラメータ

パラメータ	説明
xmlNode	XML DOM オブジェクトノードです。

使用方法
XmlGetNodeType 関数を使用して、XmlSearch 関数によって返されたノードのタイプ、または要素の XmlNodes 配列内のエントリのタイプを確認できます。

例
次の例では、XML ドキュメントオブジェクトのさまざまな部分のノードタイプを確認します。

```
<!--- Create an XML document object --->
<cfxml variable="xmlobject">
<?xml version="1.0" encoding="UTF-8"?>
<order id="4323251">
    <customer firstname="Philip" lastname="Cramer" accountNum="21"/>
    <items>
        <item id="43">
            <!-- This item is coded to show several node types -->
            <![CDATA["Our Best" hammer & chisel set!!!]]> Imported from France
            <quantity>1</quantity>
            <unitprice>15.95</unitprice>
        </item>
    </items>
</order>
</cfxml>

<!--- Display the node types --->
<cfoutput>
<h3>Node Types</h3>
xmlobject: #XMLGetNodeType(xmlobject)#<br>
xmlobject.order: #XMLGetNodeType(xmlobject.order)#<br>
<br>
Now check the types of all the nodes in the xmlobject.order.items.item
element's XmlNodes array.<br>
    Note the many apparently empty Text nodes generated by whitespace characters in the XML text
source.<br><br>
<cfset descnodes=xmlobject.order.items.item.XmlNodes>
<cfloop from="1" to="#ArrayLen(descnodes)#" index="i">
    #i# Node type is: #XMLGetNodeType(descnodes[i])#<br>
    #i# Node name is: #descnodes[i].XmlName#<br>
    <cfif (descnodes[#i#].XmlValue NEQ "")>
        #i# Node value is: #descnodes[i].XmlValue#<br>
    </cfif>
    <br>
</cfloop>
</cfoutput>
```

XmlNew

説明
XML ドキュメントオブジェクトを作成します。

戻り値
空の XML ドキュメントオブジェクト

カテゴリ
XML 関数

関数のシンタックス
```
XmlNew([caseSensitive])
```

関連項目
cfxml、IsXmlDoc、ToString、XmlFormat、XmlParse、XmlValidate、『ColdFusion アプリケーションの開発』の Using XML and WDDX

履歴
ColdFusion MX: この関数が追加されました。

パラメータ

パラメータ	説明
caseSensitive	XML ドキュメントオブジェクトのコンポーネント識別子の大文字と小文字をどのように処理するかを指定します。 • true: 大文字と小文字の区別は保持されます。 • false: 大文字と小文字は区別されません (デフォルト)。

使用方法
ColdFusion では、XML ドキュメントオブジェクトを構造体として表現します。

大文字と小文字だけが異なる複数の識別子がそれぞれ別のコンポーネントを参照するかどうかは、**caseSensitive** パラメータ値によって決まります。例 :

- true の場合、要素または属性名 "name" および "NAME" は、別々の要素または属性を参照します。
- false の場合、これらの名前は同じ要素または属性を参照します。

大文字と小文字が区別される XML オブジェクトの要素名や属性名は、ドット表記法では参照できません。連想配列 (括弧) 表記法の名前を使用するか、大文字と小文字が区別される名前を使用せずに参照する (xmlChildren[1] などのようにする) 必要があります。次のコードで、最初の行は大文字と小文字を区別する XML オブジェクトに対して機能します。2 番目と 3 番目の行ではエラーが発生します。

```
MyDoc.xmlRoot.XmlAttributes["Version"] = "12b";
MyDoc.xmlRoot.XmlAttributes.Version = "12b";
MyDoc.MyRoot.XmlAttributes["Version"] = "12b";
```

XML ドキュメントオブジェクトを文字列に変換するには、ToString 関数を使用してください。

例
次の例では、ColdFusion ドキュメントオブジェクトを作成および表示します。

```
<cfset testVar = True>
<cfscript>
    MyDoc = XmlNew();
    MyDoc.xmlRoot = XmlElemNew(MyDoc,"MyRoot");
    if (testVar IS TRUE)
        MyDoc.MyRoot.XmlText = "The value of testVar is True.";
    else
        MyDoc.MyRoot.XmlText = "The value of testVar is False.";
    for (i = 1; i LTE 4; i = i + 1){
        MyDoc.MyRoot.XmlChildren[i] = XmlElemNew(MyDoc,"childNode");
        MyDoc.MyRoot.XmlChildren[i].XmlText = "This is Child node " & i &".";
        }
</cfscript>
<cfdump var=#MyDoc#>
```

XmlParse

説明
XML テキストを XML ドキュメントオブジェクトに変換します。

戻り値
XML ドキュメントオブジェクト

カテゴリ
変換関数、XML 関数

関数のシンタックス
```
XmlParse(xmlText [, caseSensitive ], validator])
```

関連項目
cfxml、IsXML、ToString、XmlFormat、XmlNew、XmlSearch、XmlTransform、XmlValidate、『ColdFusion アプリケーションの開発』の Using XML and WDDX

履歴
ColdFusion MX 7:

- validator パラメータが追加されました。
- **xmlText** パラメータでファイル名と URL を使用できるようになりました。
- 相対 URL および相対パス名を使用できるようになりました。

ColdFusion MX: この関数が追加されました。

パラメータ

パラメータ	説明
xmlText	次のいずれかです。 • XML テキストを含む文字列 • XML ファイルの名前 • XML ファイルの URL。有効なプロトコル識別子は http、https、ftp、および file です。
caseSensitive	• Yes: ドキュメント要素と属性の大文字と小文字の区別を保持します。 • No: デフォルト
validator	次のいずれかです。 • DTD (ドキュメントタイプ定義) または XML スキーマファイルの名前 • DTD またはスキーマファイルの URL。有効なプロトコル識別子は http、https、ftp、および file です。 • DTD またはスキーマの文字列表記 • 空の文字列。この場合、XML ファイルには、ドキュメントの検証に使用する DTD またはスキーマ識別子が埋め込まれている必要があります。

使用方法
パラメータで相対 URL または相対パス名を指定した場合は、現在の ColdFusion ページを含んでいるディレクトリ (URL の場合は論理ディレクトリ) がパスのルートとして使用されます。

大文字と小文字だけが異なる複数の識別子がそれぞれ別のコンポーネントを参照するかどうかは、**caseSensitive** パラメータ値によって決まります。例 :

- true の場合、要素または属性名 "name" および "NAME" は、別々の要素または属性を参照します。

- false の場合、これらの名前は同じ要素または属性を参照します。

大文字と小文字が区別される XML オブジェクトの要素名や属性名は、ドット表記法では参照できません。連想配列 (括弧) 表記法の名前を使用するか、大文字と小文字が区別される名前を使用せずに参照する (xmlChildren[1] などのようにする) 必要があります。次のコードで、最初の行は大文字と小文字を区別する XML オブジェクトに対して機能します。2 番目と 3 番目の行ではエラーが発生します。

```
MyDoc.xmlRoot.XmlAttributes["Version"] = "12b";
MyDoc.xmlRoot.XmlAttributes.Version = "12b";
MyDoc.MyRoot.XmlAttributes["Version"] = "12b";
```

オプションの **validator** パラメータは、ドキュメントを検証するために使用する DTD またはスキーマを指定します。パーサーで検証エラーが発生すると、ColdFusion はエラーを生成してドキュメントの解析を停止します。XmlParse 関数でドキュメントを検証するには、**validator** パラメータを指定します。**validator** パラメータを指定しない場合は、XML ファイルで xsi:noNamespaceSchemaLocation タグを使用して DTD またはスキーマが指定されている必要があります。**validator** パラメータを指定する場合は、**caseSensitive** パラメータも指定します。

注意：XMLParse 関数の 3 番目のパラメータとして空の文字列を指定し、XML ドキュメント内で xsi:noNamespaceSchemaLocation などの検証を指定する場合は、完全な URL を指定する必要があります。ただし、XMLValidate を使用した場合は、検証を実行している CFML テンプレートからの相対位置を示すファイル名だけでもかまいません。

validator パラメータを指定しない場合は、**xmlText** パラメータで整形式の XML フラグメントを指定することができます。ドキュメント全体を指定する必要はありません。

注意：XML ドキュメントオブジェクトを変換して文字列に戻すには、ToString 関数を使用してください。

例

次の例には 3 つの部分があります。つまり、XML ファイル、DTD ファイル、および検証のために XML ファイルを解析して DTD ファイルを使用する CFML ページです。CFML ファイルは、返された XML ドキュメントオブジェクトを表示します。無効な XML の結果を表示するには、bmenuD.xml を修正します。

注意：次の例で使用する DTD は、XmlValidate の例で使用するスキーマと同じ XML 構造を表しています。

"custorder.xml" ファイルは次のとおりです。

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE order SYSTEM "C:\ColdFusion\wwwroot\examples\custorder.dtd">
<order id="4323251">
    <customer firstname="Philip" lastname="Cramer" accountNum="21"/>
    <items>
        <item id="43">
            <name>Deluxe Carpenter&apos;s Hammer</name>
            <quantity>1</quantity>
            <unitprice>15.95</unitprice>
        </item>
        <item id="54">
            <name>36&quot; Plastic Rake</name>
            <quantity>2</quantity>
            <unitprice>6.95</unitprice>
        </item>
        <item id="68">
            <name>Standard paint thinner</name>
            <quantity>3</quantity>
            <unitprice>8.95</unitprice>
        </item>
    </items>
</order>
```

"custorder.dtd" ファイルは次のとおりです。

```
<!ELEMENT order (customer, items)>
<!ATTLIST order
    id CDATA #REQUIRED>
<!ELEMENT customer EMPTY>
<!ATTLIST customer
    firstname CDATA #REQUIRED
    lastname CDATA #REQUIRED
    accountNum CDATA #REQUIRED>
<!ELEMENT items (item+)>
<!ELEMENT item (name, quantity, unitprice)>
<!ATTLIST item
    id CDATA #REQUIRED>
<!ELEMENT name (#PCDATA)>
<!ELEMENT quantity (#PCDATA)>
<!ELEMENT unitprice (#PCDATA)>
```

CFML ファイルは次のとおりです。XML ファイルのファイル名および DTD の URL を使用します。XML および URL のパスは絶対パスでなければなりません。

```
<cfset
myDoc=XMLParse("C:\ColdFusion\wwwroot\examples\custorder.xml",
false, "http://localhost:8500/examples/custorder.dtd")>
Dump of myDoc XML document object<br>
<cfdump var="#myDoc#">
```

XmlSearch

説明

XPath 言語式を使用して XML ドキュメントオブジェクトを検索します。

戻り値

XPath 検索の結果。詳細については、「使用方法」を参照してください。

カテゴリ

XML 関数

関数のシンタックス

```
XmlSearch(xmlDoc, xPathString, params)
```

関連項目

cfxml、IsXML、XmlChildPos、XmlParse、XmlTransform、『ColdFusion アプリケーションの開発』の Using XML and WDDX

履歴

ColdFusion 10：XPath 2.0 シンタックスのサポート、および XPath 式への変数の引き渡しのサポートが追加されました。

ColdFusion 8: XML オブジェクトノードの配列だけでなく、有効な XPath 結果の配列を返すことができるようになりました。

ColdFusion MX 7: 属性の検索が可能になりました。

ColdFusion MX: この関数が追加されました。

パラメータ

パラメータ	説明

params	オプション。キーと値のペア（変数名と値）のドキュメントオブジェクトが含まれる構造体です。次に例を示します。
	`<cfscript>`
	`params = structNew();`
	`params["test"] = "food";`
	`</cfscript>`
	`<!--- find all nodes with element name as passed by variable test --->`
	`<cfset result = xmlSearch(xmldoc,"/breakfast_menu/*[local-name() eq $test]", params)>`

xmlDoc	XML ドキュメントオブジェクト
xPathString	XPath 式です。

使用方法

XmlSearch 関数は、検索で返された値を可能な限り返します。たとえば、XPath 式がブール値を返す場合、CFML 変数には true または false の値が割り当てられます。

次の表に、XPath 式の結果のデータ型と、そのデータ型が CFML 戻り値でどのように表現されるかを示します。

XPath の戻り型	ColdFusion での表現
Boolean	Boolean
Null	"" (空の文字列)
Number	Number
String	String
NodeSet	XML ノードの配列
Result Tree Fragment	XML ノードの配列

結果が Unknown、または式に未解決の変数がある場合は、エラーが返されます。

XPath は W3C (World Wide Web Consortium) によって仕様が定められています。XPath 式シンタックスなどの XPath の詳細については、W3C の Web サイト www.w3.org/TR/xpath を参照してください。

例

次の例では、従業員の姓が格納されている **last** という名前の要素を XML ファイルから抽出し、姓を表示します。

"employeesimple.xml" ファイルには次の XML が格納されています。

```
<?xml version="1.0" encoding="UTF-8"?>
<employee>
<!-- A list of employees -->
    <name EmpType="Regular">
<first>Almanzo</first>
<last>Wilder</last>
    </name>
    <name EmpType="Contract">
<first>Laura</first>
<last>Ingalls</last>
    </name>
</employee>
```

この CFML ファイルには次の行が含まれています。

```
<cfscript>
    myxmldoc = XmlParse("C:\CFusionMX7\wwwroot\examples\employeesimple.xml");
    selectedElements = XmlSearch(myxmldoc, "/employee/name/last");
    for (i = 1; i LTE ArrayLen(selectedElements); i = i + 1)
        writeoutput(selectedElements[i].XmlText & "<br>");
</cfscript>
```

XmlTransform

説明

XSLT (Extensible Stylesheet Language Transformation) を XML に適用します。XML は、文字列形式ドキュメントの場合と XML ドキュメントオブジェクトの場合があります。

戻り値

XSLT を XML に適用した結果を含む文字列

カテゴリ

変換関数、XML 関数

関数のシンタックス

```
XmlTransform(xml, xsl[, parameters])
```

関連項目

cfxml、XmlFormat、XmlNew、XmlParse、XmlSearch、XmlValidate、『ColdFusion アプリケーションの開発』の Using XML and WDDX

履歴

ColdFusion 10：XSLT 2.0 シンタックスのサポートが追加されました。

ColdFusion MX 7: parameters パラメータが追加され、XSL のファイルを使用できるようになりました。

ColdFusion MX: この関数が追加されました。

パラメータ

パラメータ	説明
xml	文字列形式または XML ドキュメントまたは XML ドキュメントオブジェクトです。
xsl	適用する XSLT 変換であり、次のいずれかに相当します。 次のいずれかです。 • XSL テキストを含む文字列 • XSTLT ファイルの名前。現在の CFML ページを含むディレクトリで始まる相対パスです。 • XSLT ファイルの URL。有効なプロトコル識別子は http、https、ftp、および file です。現在の CFML ページを含むディレクトリで始まる相対パスです。
parameters	ドキュメントの変換に使用する XSL テンプレートパラメータの名前 / 値のペアを含む構造体です。xslString パラメータで定義された XSL 変換では、XML を処理するときにこれらのパラメータ値が使用されます。

使用方法

XSLT では、Extensible Stylesheet Language (XSL) スタイルシートを適用することにより、XML ドキュメントを別の形式や表現に変換します。XSLT シンタックスを含む XSL は、W3C (World Wide Web Consortium) によって仕様が定められています。XSL および XSLT の詳細については、W3C の Web サイト www.w3.org/Style/XSL/ を参照してください。

インクルードステートメントが相対パスとともに XSLT コードに含まれている場合、ColdFusion は XSLT ファイルの場所を基準として相対パスを解決します。XSL 文字列の場合は、現在の ColdFusion ページの場所を基準として相対パスを解決します。

例

次の例では、顧客注文を示す XML ドキュメントを、顧客名および注文品と数量の表を含む HTML ドキュメントに変換します。

顧客注文を示す "custorder.xml" ファイルには、次の行があります。

```
<?xml version="1.0" encoding="UTF-8"?>
<order id="4323251">
    <customer firstname="Philip" lastname="Cramer" accountNum="21"/>
    <items>
        <item id="43">
            <name>Deluxe Carpenter&apos;s Hammer</name>
            <quantity>1</quantity>
            <unitprice>15.95</unitprice>
        </item>
        <item id="54">
            <name>36&quot; Plastic Rake</name>
            <quantity>2</quantity>
            <unitprice>6.95</unitprice>
        </item>
        <item id="68">
            <name>Standard paint thinner</name>
            <quantity>3</quantity>
            <unitprice>8.95</unitprice>
        </item>
    </items>
</order>
```

XML を HTML に変換して、顧客名および発注された商品と数量を表示する XSLT ファイル "custorder.xsd" には、次の行があります。

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="html" doctype-public="-//W3C//DTD HTML 4.0 Transitional//EN" />
    <xsl:template match="/">
        <html>
            <body>
                <table border="2" bgcolor="yellow">
                    <tr>
                        <th>Name</th>
                        <th>Price</th>
                    </tr>
                    <xsl:for-each select="breakfast_menu/food">
                        <tr>
                            <td>
                                <xsl:value-of select="name"/>
                            </td>
                            <td>
                                <xsl:value-of select="price"/>
                            </td>
                        </tr>
                    </xsl:for-each>
                </table>
            </body>
        </html>
    </xsl:template>
</xsl:stylesheet>
```

この CFML ファイルには次の行が含まれています。

```
<cffile action="read" file="C:\CFusionMX7\wwwroot\examples\custorder.xsl" variable="xmltrans">
<cfset xmldoc = XmlParse("C:\CFusionMX7\wwwroot\examples\custorder.xml")>
<cfoutput>#XmlTransform(xmldoc, xmltrans)#</cfoutput>
```

XmlValidate

説明

DTD (ドキュメントタイプ定義) または XML スキーマを使用して、XML テキストドキュメントまたは XML ドキュメントオブジェクトを検証します。

戻り値

次の検証構造体

フィールド	説明
Errors	検証機能のエラーメッセージを含む配列です。これらのメッセージは、ドキュメントが DTD またはスキーマに準拠していないこと (ドキュメントが有効でないこと) を示します。
FatalErrors	検証機能の重大エラーメッセージを含む配列です。重大エラーは、ドキュメントに XML 形式設定エラーがあること (ドキュメントが整形式の XML でないこと) を示します。
Status	ブール値です。 • ドキュメントが有効である場合は true • 検証チェックに失敗した場合は false
Warning	検証機能の警告を含む配列です。整形式で有効なドキュメントでも警告メッセージが生成される場合があります。

カテゴリ
XML 関数

関数のシンタックス
```
XmlValidate(xmlDoc[, validator])
```

関連項目
cfxml、IsXmlDoc、IsXML、XmlFormat、XmlNew、XmlParse、XmlSearch、XmlTransform、『ColdFusion アプリケーションの開発』の Using XML and WDDX

履歴
ColdFusion MX 7: この関数が追加されました。

パラメータ

パラメータ	説明
xmlDoc	次のいずれかです。 • XML ドキュメントを含む文字列 • XML ファイルの名前 • XML ファイルの URL。有効なプロトコル識別子は http、https、ftp、および file です。 • XML ドキュメントオブジェクト。たとえば、XmlParse 関数によって生成された XML ドキュメントオブジェクトなどです。
validator	次のいずれかです。 • DTD またはスキーマを含む文字列 • DTD またはスキーマファイルの名前 • DTD またはスキーマファイルの URL。有効なプロトコル識別子は http、https、ftp、および file です。

使用方法
パラメータで相対 URL または相対ファイル名を指定した場合は、現在の ColdFusion ページを含んでいるディレクトリ (URL の場合は仮想ディレクトリ) がパスのルートとして使用されます。

validator パラメータは、ドキュメントを検証するために使用する DTD またはスキーマを指定します。このパラメータを省略した場合は、XML ドキュメントに次のいずれかを含める必要があります。

- DTD または DTD の場所を指定する !DOCTYPE タグ
- スキーマの場所を指定する xsi:schemaLocation タグまたは xsi:noNamespaceSchemaLocation タグ

validator パラメータを使用し、XML ドキュメントで DTD またはスキーマを指定する場合、XmlValidate 関数は **validator** パラメータを使用し、XML ドキュメントでの指定を無視します。

validator パラメータを使用せず、XML ドキュメントで DTD またはスキーマを指定しない場合、この関数は Errors フィールドにエラーメッセージを含む構造体を返します。

この関数は、XML ドキュメント全体に対して検証処理を行い、処理中に検出されたエラーをすべてレポートします。その結果、返される構造体は Warning、Error、および FatalError の各フィールドの組み合わせになる可能性があり、各フィールドには複数のエラーメッセージが含まれる場合があります。

例

次の例には 3 つの部分があります。つまり、XML ファイル、XSD スキーマファイル、および検証のために XML ファイルを解析してスキーマを使用する CFML ページです。CFML ファイルでは、返される構造体の Status フィールドの値が表示され、返される構造体が表示されます。無効な XML の結果を表示するには、custorder.xml ファイルを修正します。

注意：次の例で使用するスキーマは、XmlParse の例で使用する DTD と同じ XML 構造を表しています。

"custorder.xml" ファイルは次のとおりです。

```
<?xml version="1.0" encoding="UTF-8"?>
<order xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://localhost:8500/something.xsd" id="4323251" >
    <customer firstname="Philip" lastname="Cramer" accountNum="21"/>
    <items>
        <item id="43">
            <name>Deluxe Carpenter&apos;s Hammer</name>
            <quantity>1</quantity>
            <unitprice>15.95</unitprice>
        </item>
        <item id="54">
            <name>36&quot; Plastic Rake</name>
            <quantity>2</quantity>
            <unitprice>6.95</unitprice>
        </item>
        <item id="68">
            <name>Standard paint thinner</name>
            <quantity>3</quantity>
            <unitprice>8.95</unitprice>
        </item>
    </items>
</order>
```

"custorder.xsd" ファイルは次のとおりです。

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
    elementFormDefault="qualified">
    <xs:element name="customer">
        <xs:complexType>
            <xs:attribute name="firstname" type="xs:string" use="required"/>
            <xs:attribute name="lastname" type="xs:string" use="required"/>
            <xs:attribute name="accountNum" type="xs:string" use="required"/>
        </xs:complexType>
    </xs:element>
    <xs:element name="name" type="xs:string"/>
    <xs:element name="quantity" type="xs:string"/>
    <xs:element name="unitprice" type="xs:string"/>
    <xs:element name="item">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="name"/>
                <xs:element ref="quantity"/>
                <xs:element ref="unitprice"/>
            </xs:sequence>
            <xs:attribute name="id" type="xs:integer" use="required">
```

```
            </xs:attribute>
        </xs:complexType>
    </xs:element>
    <xs:element name="items">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="item" maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
    <xs:element name="order">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="customer"/>
                <xs:element ref="items"/>
            </xs:sequence>
            <xs:attribute name="id" type="xs:string" use="required"/>
        </xs:complexType>
    </xs:element>
</xs:schema>
```

CFML ファイルは次のとおりです。XML ファイルのファイル名およびスキーマの URL を使用します。XML パスおよび URL パスは絶対パスでなければなりません。

```
<cfset
myResults=XMLValidate("C:\CFusionMX7\wwwroot\examples\custorder.xml",
"http://localhost:8500/examples/custorder.xsd")>
<cfoutput>
Did custorder.xml validate against custorder.xsd? #myResults.status#<br><br>
</cfoutput>
Dump of myResults structure returned by XMLValidate<br>
<cfdump var="#myResults#">
```

Year

説明
日付時刻オブジェクトから年の値を取得します。

戻り値
date の年の値

カテゴリ
日付および時刻関数

関数のシンタックス
`Year(date)`

関連項目
DatePart、IsLeapYear

パラメータ

パラメータ	説明
date	日付時刻オブジェクトです。値の範囲は西暦 100 ～ 9999 年です。

使用方法
日付を文字列として渡すときは、その値を引用符で囲む必要があります。引用符で囲まない場合、その値は日付の数値表現として解釈されます。

例
```
<h3>Year Example</h3>
<cfif IsDefined("FORM.year")>
    More information about your date:
    <cfset yourDate = CreateDate(FORM.year,FORM.month,FORM.day)>
    <cfoutput>
    <p>Your date, #DateFormat(yourDate)#.
    <br>It is #DayOfWeekAsString(DayOfWeek(yourDate))#,
    day #DayOfWeek(yourDate)# in the week.
    <br>This is day #Day(yourDate)#
    in the month of #MonthAsString(Month(yourDate))#,
    which has #DaysInMonth(yourDate)# days.
    <br>We are in week #Week(yourDate)# of #Year(YourDate)#
    (day #DayOfYear(yourDate)# of #DaysInYear(yourDate)#). <br>
    <cfif IsLeapYear(Year(yourDate))>
        This is a leap year
        <cfelse>This is not a leap year
    </cfif>
    </cfoutput>
</cfif>
```

YesNoFormat

説明
数値またはブール値を評価します。

戻り値
ゼロ以外の数値の場合は Yes、ゼロ、ブール値 (false および no)、空の文字列 ("") の場合は No。

カテゴリ
決定関数、表示および書式制御関数

関数のシンタックス
```
YesNoFormat(value)
```

関連項目
IsBinary、IsNumeric

パラメータ

パラメータ	説明
value	数値またはブール値です。

例

```
 <h3>YesNoFormat Example</h3>
<p>The YesNoFormat function returns non-zero values as "Yes"; zero, false and no Boolean values, and empty
strings ("") as "No".

<cfoutput>
<ul>
    <li>YesNoFormat(1):#YesNoFormat(1)#
    <li>YesNoFormat(0):#YesNoFormat(0)#
    <li>YesNoFormat("1123"):#YesNoFormat("1123")#
    <li>YesNoFormat("No"):#YesNoFormat("No")#
    <li>YesNoFormat(True):#YesNoFormat(True)#
</ul>
</cfoutput>
```

第 5 章：AJAX JavaScript 関数

ColdFusion AJAX 機能を利用するページでは、次に示す JavaScript 関数を使用できます。

関数一覧

次の表では、AJAX 機能を利用する ColdFusion ページで使用可能な JavaScript 関数について簡単に説明します。

関数	説明
ColdFusion.Ajax.submitForm	結果が返されたときに、ページ全体を更新せずにフォームデータを送信します。
ColdFusion.Autosuggest.getAutosuggestObject	基盤となる YUI AutoComplete オブジェクトにアクセスできるので、オブジェクトをより詳細に管理できます。たとえば、オブジェクトにイベントを関連付けたりできます。
ColdFusion.FileUpload.cancelUpload	ファイルのアップロード中の任意の時点でファイルのアップロードをキャンセルします。
ColdFusion.FileUpload.clearAllFiles	アップロード対象として選択されたすべてのファイルをクリアします。
Coldfusion.fileUpload.setUrl	ファイルアップロードコントロールの URL をダイナミックに設定します。
ColdFusion.FileUpload.startUpload	選択されたファイルのアップロードを開始します。
ColdFusion.getElementValue	バインド可能な ColdFusion コントロールの属性の値を取得します。
ColdFusion.grid.clearSelectedRows	グリッドの選択した行をクリアします。
ColdFusion.Grid.getBottomToolbar	下部のツールバーを取得します。ツールバーは、アイコンやボタンなどのコントロールの追加に使用します。
ColdFusion.Grid.getGridObject	指定された HTML の cfgrid コントロールに対して、基盤となる Ext JS - JavaScript Library オブジェクトを取得します。
ColdFusion.grid.getSelectedRows	グリッドの選択した行のデータを取得します。
ColdFusion.Grid.getTopToolbar	上部のツールバーを取得します。ツールバーは、アイコンやボタンなどのコントロールの追加に使用します。
ColdFusion.Grid.hideBottomToolbar	下部のツールバーを非表示にします。ツールバーは、アイコンやボタンなどのコントロールの追加に使用します。
ColdFusion.Grid.hideTopToolbar	上部のツールバーを非表示にします。ツールバーは、アイコンやボタンなどのコントロールの追加に使用します。
ColdFusion.Grid.refresh	グリッドの表示を手動で更新します。
ColdFusion.Grid.refreshBottomToolbar	下部のツールバーを更新します。ツールバーは、アイコンやボタンなどのコントロールの追加に使用します。
ColdFusion.Grid.refreshTopToolbar	上部のツールバーを更新します。ツールバーは、アイコンやボタンなどのコントロールの追加に使用します。
ColdFusion.Grid.showBottomToolbar	下部のツールバーを表示します。ツールバーは、アイコンやボタンなどのコントロールの追加に使用します。
ColdFusion.Grid.showTopToolbar	上部のツールバーを表示します。ツールバーは、アイコンやボタンなどのコントロールの追加に使用します。
ColdFusion.Grid.sort	指定された HTML グリッドをソートします。
ColdFusion.JSON.decode	JSON エンコードの文字列を JavaScript 変数に変換します。

関数	説明
ColdFusion.JSON.encode	JavaScript 変数を JSON 文字列に変換します。
ColdFusion.Layout.collapseAccordion	アコーディオンレイアウト領域を折り畳みます。
ColdFusion.Layout.collapseArea	ボーダー付きレイアウト (type 属性で border が指定されている cflayout タグ) の領域を折り畳みます。
ColdFusion.Layout.createAccordionPanel	既存のアコーディオンレイアウト (type 属性で accordion が指定されている cflayout タグ) にパネルを作成します。
ColdFusion.Layout.createTab	既存のタブ付きレイアウト (type 属性で tab が指定されている cflayout タグ) にタブを作成します。
ColdFusion.Layout.disableSourceBind	ソースバインドを無効にします。
ColdFusion.Layout.disableTab	特定のタブを無効にして選択できないようにします。
ColdFusion.Layout.enableSourceBind	ソースバインドが無効になっている場合は有効にします。
ColdFusion.Layout.enableTab	特定のタブを有効にし、ユーザーがそのタブを選択して領域のコンテンツを表示できるようにします。
ColdFusion.Layout.expandAccordion	アコーディオンレイアウトの折り畳まれている領域を展開します。
ColdFusion.Layout.expandArea	ボーダー付きレイアウトの折り畳まれている領域を展開します。
ColdFusion.Layout.getAccordionLayout	指定されたアコーディオンタイプの cflayout コントロールに対して、基盤となる Ext JS - JavaScript Library オブジェクトを取得します。
ColdFusion.Layout.getBorderLayout	指定された border タイプの cflayout コントロールに対して、基盤となる Ext JS - JavaScript Library オブジェクトを取得します。
ColdFusion.Layout.getTabLayout	指定された tab タイプの cflayout コントロールに対して、基盤となる Ext JS - JavaScript Library オブジェクトを取得します。
ColdFusion.Layout.hideAccordion	アコーディオンを非表示にします。
ColdFusion.Layout.hideArea	ボーダー付きレイアウトの領域を非表示にします。
ColdFusion.Layout.hideTab	タブを非表示にします。
ColdFusion.Layout.selectAccordion	アコーディオンを選択して、レイアウト領域のコンテンツを表示します。
ColdFusion.Layout.selectTab	タブを選択して、レイアウト領域のコンテンツを表示します。
ColdFusion.Layout.showAccordion	inithide 属性または hideArea() 関数によって非表示にされたアコーディオンを表示します。
ColdFusion.Layout.showArea	inithide 属性または hideArea() 関数によって非表示にされたボーダー付きレイアウトの領域を表示します。
ColdFusion.Layout.showTab	inithide 属性または hideTab() 関数によって非表示にされたタブを表示します。
ColdFusion.Log.debug	デバッグレベルのメッセージをログウィンドウに表示します。
ColdFusion.Log.dump	複合変数に関する情報をログウィンドウに表示します。
ColdFusion.Log.error	エラーレベルのメッセージをログウィンドウに表示します。
ColdFusion.Log.info	情報レベルのメッセージをログウィンドウに表示します。
ColdFusion.Map.addEvent	マップ内のイベント処理を有効にします。
ColdFusion.Map.addMarker	マップにマーカーを追加します。
ColdFusion.Map.getLatitudeLongitude	特定の住所の緯度 / 経度座標を取得します。
ColdFusion.Map.getMapObject	Google マップコンポーネントを取得します。

関数	説明
ColdFusion.Map.hide	マップが表示されている場合は非表示にします。
ColdFusion.Map.refresh	マップをリロードします。
ColdFusion.Map.setCenter	指定した住所をマップの中心に設定します。
ColdFusion.Map.setZoomlevel	マップのズームレベルを新しい値に設定します。
ColdFusion.Map.show	マップが非表示になっている場合は表示します。
ColdFusion.Mediaplayer.resize	メディアプレーヤの現在のサイズを変更します。
ColdFusion.Mediaplayer.setMute	メディアプレーヤのサウンドをミュートまたはミュート解除します。
ColdFusion.Mediaplayer.setSource	FLV ファイルの URL を設定します。
ColdFusion.Mediaplayer.setVolume	メディアプレーヤのサウンドのボリュームを設定します。
ColdFusion.Mediaplayer.startPlay	FLV ファイルを再生します。
ColdFusion.Mediaplayer.stopPlay	FLV ファイルの再生を停止します。
ColdFusion.MessageBox.create	ColdFusion メッセージボックスを作成します。cfmessagebox タグと同等です。
ColdFusion.MessageBox.getMessageBoxObject	指定された HTML の cfmessagebox コントロールに対して、基盤となる Ext JS - JavaScript Library オブジェクトを取得します。
ColdFusion.MessageBox.isMessageBoxDefined	メッセージボックスが定義されているかどうかをチェックします。
ColdFusion.MessageBox.show	ColdFusion メッセージボックスを表示します。
ColdFusion.MessageBox.update	メッセージボックスのプロパティを更新します。
ColdFusion.MessageBox.updateMessage	メッセージのプロパティを更新します。
ColdFusion.MessageBox.updateTitle	メッセージボックスのタイトルを更新します。
ColdFusion.navigate	リンク URL の出力を、指定した cfdiv、cflayoutarea、cfpod、または cfwindow コンテナ内に表示します。
ColdFusion.ProgressBar.getProgressBarObject	プログレスバーオブジェクトを取得します。
ColdFusion.ProgressBar.hide	プログレスバーが表示されている場合は非表示にします。
ColdFusion.ProgressBar.reset	進行状況をリセットします。
ColdFusion.ProgressBar.show	プログレスバーが非表示になっている場合は表示します。
ColdFusion.ProgressBar.start	実行中の、基盤となるプログレスバーオブジェクトを停止します。
ColdFusion.ProgressBar.stop	プログレスバーオブジェクトを開始します。
ColdFusion.ProgressBar.update	duration、interval、および oncomplete の各属性を更新します。
ColdFusion.ProgressBar.updatestatus	プログレスバーのステータスとメッセージを手動で更新できます。
ColdFusion.setGlobalErrorHandler	ColdFusion の AJAX エラーに関する情報を表示するためのグローバル JavaScript エラーハンドラを置き換えます。
ColdFusion.Slider.disable	スライダコントロールを無効にします。
ColdFusion.Slider.enable	スライダコントロールを有効にします。
ColdFusion.Slider.getSliderObject	スライダコントロールを取得します。

関数	説明
ColdFusion.Slider.getValue	スライダコントロールの数値を取得します。
ColdFusion.Slider.hide	スライダコントロールを非表示にします。
ColdFusion.Slider.setValue	スライダコントロールの数値を設定します。
ColdFusion.Slider.show	スライダコントロールを表示します。
ColdFusion.Tree.getTreeObject	指定された HTML の cftree コントロールに対して、基盤となる Yahoo YUI Library オブジェクトを取得します。
ColdFusion.Tree.refresh	表示されている HTML のツリーを手動で更新します。
ColdFusion.Window.create	ColdFusion ポップアップウィンドウを作成します。cfwindow タグと同等です。
ColdFusion.Window.getWindowObject	指定された HTML の cfwindow コントロールの基盤となる Ext JS - JavaScript Library オブジェクトを取得します。
ColdFusion.Window.hide	ウィンドウを非表示にします。
ColdFusion.Window.onHide	特定のウィンドウを非表示にするたびに実行される JavaScript 関数を指定します。
ColdFusion.Window.onShow	特定のウィンドウを表示するたびに実行される JavaScript 関数を指定します。
ColdFusion.Window.show	非表示になっているウィンドウを表示します。

関連項目

1463 ページの「JavaScript 関数」

ColdFusion.Ajax.submitForm

説明

結果が返されたときに、ページを更新せずにフォームデータを送信します。

関数のシンタックス

```
ColdFusion.Ajax.submitForm(formId, URL[, callbackhandler, errorHandler, httpMethod, asynch])
```

関連項目

cfajaxproxy、ColdFusion.navigate、『ColdFusion アプリケーションの開発』の Using the ColdFusion.Ajax.submitForm function

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
formId	フォームの ID 属性または name 属性です。
URL	フォームの送信先 URL です。
callbackhandler	通常のレスポンスを処理する JavaScript 関数です。この関数は 1 つの引数 (レスポンス本文) を取る必要があります。このメソッドは、フォームの送信を非同期で行う場合にのみ使用します。

パラメータ	説明
errorHandler	HTTP エラーレスポンスを処理する JavaScript 関数です。この関数は 2 つの引数 (HTTP ステータスコードとエラーメッセージ) を取る必要があります。このメソッドは、フォームの送信を非同期で行う場合にのみ使用します。
httpMethod	送信に使用する HTTP メソッドです。次のいずれかを指定する必要があります。 • GET • POST (デフォルト)
asynch	フォームを非同期で送信するかどうかを指定するブール値です。デフォルト値は true です。

戻り値
asynch 引数が false の場合は、レスポンス本文が返されます。それ以外の場合、この関数は値を返しません。

使用方法
この関数を呼び出すページに ColdFusion AJAX ベースのコントロールがない場合は、そのページで cfajaximport タグを使用します。これにより、この関数に必要な JavaScript 定義がページ内に確実に含まれることになります。

注意：この関数は、ファイルフィールドの内容は送信しません。

例
『ColdFusion アプリケーションの開発』の Using the ColdFusion.Ajax.submitForm function を参照してください。

ColdFusion.Autosuggest.getAutosuggestObject

説明
基盤となる YUI AutoComplete オブジェクトにアクセスできるので、オブジェクトをより詳細に管理できます。たとえば、オブジェクトにイベントを関連付けたりできます。

戻り値
基盤となる AutoComplete オブジェクト

関数のシンタックス
ColdFusion.Autosuggest.getAutosuggestObject (Id)

パラメータ
• Id: 入力候補オブジェクトの名前です。

例

```
<html>
      <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
      <head>
            <cfajaximport tags="cfinput-autosuggest">
            <script>
                  var init = function()
                  {
                        autosuggestobj = ColdFusion.Autosuggest.getAutosuggestObject('state');
                        autosuggestobj.itemSelectEvent.subscribe(foo);
                  }
                  var foo = function(event,args)
                  {
                        var msg  = "";
                        msg = msg + "Event: " + event + "\n\n";
                        msg = msg + "Selected Item: " + args[2] + "\n\n";
                        msg = msg + "Index: " + args[1]._nItemIndex + "\n\n";
                        alert(msg);
                  }
                  var getStates = function(){
                        return
["California","Connecticut","Colorado","Illinois","Alabama","Iowa","Utah","Alaska"];
                  }
            </script>
      </head>
      <body>
            <h3>Attaching an event handler to the autosuggest object</h3>
            <cfform name="mycfform" method="post" >
                  State:<BR>
                  <cfinput
                        type="text"
                        name="state"
                        autosuggest="javascript:getStates({cfautosuggestvalue})"
                        autosuggestMinLength=1
                        autosuggestBindDelay=1>
                  <cfset ajaxOnLoad("init")>
            </cfform>
      </body>
</html>
```

ColdFusion.Chart.getChartHandle

説明

様々な JavaScript 操作の実行（例えば、チャート内の系列やノードの追加または削除など）に使用するチャートオブジェクトを取得するために使用します。

戻り値

JavaScript オブジェクト

関数のシンタックス

```
ColdFusion.Chart.getChartHandle()
```

履歴

ColdFusion 10: この関数が追加されました。

例
次の例は、チャート ID が interactivebar のチャートの最初の系列に新しい値を追加する方法を示しています。

```
ColdFusion.Chart.getChartHandle().exec('interactivebar', 'appendseriesvalues', '{"plotindex": 0, "values"
: [40]}' );
```

例
次の例は、新しい値を追加してチャートのクリックイベントを登録する方法を示しています。

```
ColdFusion.Chart.getChartHandle().click = function(dataObj){
alert("Chart Clicked - ID: " + data["id"]);
}
```

ColdFusion.FileUpload.cancelUpload

説明
ファイルのアップロード中の任意の時点で、アップロードのキャンセルを選択できます。アップロードをキャンセルした場合、以降のファイルのアップロードがすべて停止されます。

関数のシンタックス
```
ColdFusion.FileUpload.cancelUpload(name)
```

関連項目
ColdFusion.FileUpload.clearAllFiles、ColdFusion.FileUpload.startUpload

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
name	cffileupload タグの name 属性の値です。

戻り値
この関数は値を返しません。

例
この例には、Cancel Upload アクションのボタンが含まれます。このボタンをクリックすると、ファイルのアップロードがキャンセルされます。

```
<h3>This is an example of the FileUpload.cancelUpload function. Add a few files to upload and click Upload.
During upload, click the Cancel Upload HTML button to cancel the upload.</h3>
    <script language="JavaScript" type="text/javascript">
    function onCancel() {
    ColdFusion.FileUpload.cancelUpload('myupload');
    };
    </script>
    <cffileupload
    url="uploadAll.cfm"
    progressbar="true"
    name="myupload"
    addButtonLabel = "Add File"
    clearButtonlabel = "Clear it"
    width=600
    height=400
    title = "File Upload"
    maxuploadsize="30"
    extensionfilter="*.jpg, *.png, *.flv, *.txt"
    BGCOLOR="##FFFFFF"
    MAXFILESELECT=10
    UPLOADBUTTONLABEL="Upload now"/>
    <cfform name="form01">
    <cfinput type="button" name="cancelupld" value="Cancel Upload"
    onclick="onCancel()">
    </cfform>
```

ColdFusion.FileUpload.clearAllFiles

説明

アップロードする前、またはアップロードをキャンセルした後に、アップロード対象として選択されたすべてのファイルをクリアできます。クリアすると、選択されたファイルがアップロードのビューリストから削除されます。

関数のシンタックス

```
ColdFusion.FileUpload.clearAllFiles(name)
```

関連項目

ColdFusion.FileUpload.cancelUpload、ColdFusion.FileUpload.startUpload

履歴

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
name	cffileupload タグの name 属性の値です。

戻り値

この関数は値を返しません。

例

この例には、ClearAllFiles アクションのボタンが含まれます。このボタンをクリックすると、アップロード対象として選択されたファイルが削除されます。

```
<h3>This is an example of the FileUpload.clearAllFiles function. Add a few files to upload and click Upload.
During upload, click the Clear Upload HTML button to clear all the files selected for upload.</h3>
    <script language="JavaScript" type="text/javascript">
    function onClear() {
    ColdFusion.FileUpload.clearAllFiles('myupload');
    };
    </script>
    <cffileupload
    url="uploadAll.cfm"
    progressbar="true"
    name="myupload"
    addButtonLabel = "Add File"
    clearButtonlabel = "Clear it"
    width=600
    height=400
    title = "File Upload"
    maxuploadsize="30"
    extensionfilter="*.jpg, *.png, *.flv, *.txt"
    BGCOLOR="##FFFFFF"
    MAXFILESELECT=10
    UPLOADBUTTONLABEL="Upload now"/>
    <cfform name="form01">
    <cfinput type="button" name="clearupload" value="Cancel Upload"
    onclick="onClear()">
    </cfform>
```

Coldfusion.fileUpload.setUrl

説明
ファイルアップロードコントロールの URL をダイナミックに設定する場合に使用します。

戻り値
なし

関数のシンタックス
ColdFusion.fileUpload.setUrl(id, url)

パラメータ
- Id: アップロードコントロールの名前です。
- Url: URL には、絶対 URL、相対 URL、または完全修飾 URL を設定できます。

例

```
<script language="javascript">
    var uploadDone = function(result){
        alert("File uploaded");
    }

    var setUploadUrl = function(id)
    {
        var selectedFiles = ColdFusion.FileUpload.getSelectedFiles(id);
        var uploadUrl = "/manual/ajaxui/cffileupload/setUrl/includes/_uploadall.cfm";
        alert("Upload URL : " + uploadUrl);
        if(selectedFiles.length){
            ColdFusion.FileUpload.setURL(id,uploadUrl);
            ColdFusion.FileUpload.startUpload(id);
        }
    }
    var callbackhandler = function(obj)
    {
        var fileName = obj["FILENAME"];
        var status = obj["STATUS"];
        var message = obj["MESSAGE"];
        var msg = "In callbackhandler()" + "\n\n" +
                  "FILENAME: " + fileName + "\n\n" +
                  "STATUS: " + status + "\n\n" +
                  "MESSAGE: " + message
        alert(msg);
    }
    var errorhandler = function()
    {
        alert("In errorhandler()");
    }
    var uploadcompleted = function()
    {
        alert("All files have been uploaded successfully");
    }
</script>
<cfform name="frmUpload">
    <br>
    <cffileupload name="uploader" hideuploadbutton="true" onComplete="uploadDone" onError="errorhandler"
onUploadComplete="uploadcompleted">
    <br>
    <cfinput type="button" name="submit" value="Click to set URL and Upload Files"
onClick="setUploadUrl('uploader')">
</cfform>
```

ColdFusion.FileUpload.startUpload

説明

選択されたファイルのアップロードを開始します。

関数のシンタックス

```
ColdFusion.FileUpload.startUpload(name)
```

関連項目

ColdFusion.FileUpload.cancelUpload、ColdFusion.FileUpload.clearAllFiles

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
name	cffileupload タグの name 属性の値です。

戻り値
この関数は値を返しません。

例
この例には、Upload アクションのボタンが含まれます。このボタンをクリックすると、選択されたファイルのアップロードが開始されます。

```
<script language="JavaScript" type="text/javascript">
function onUpload() {
ColdFusion.FileUpload.startUpload('myupload');
};
</script>
<cffileupload
url="uploadAll.cfm"
progressbar="true"
name="myupload"
addButtonLabel = "Add File"
clearButtonlabel = "Clear it"
width=600
height=400
title = "File Upload"
maxuploadsize="30"
extensionfilter="*.jpg, *.png, *.flv, *.txt"
BGCOLOR="##FFFFFF"
MAXFILESELECT=10
UPLOADBUTTONLABEL="Upload now"/>
<cfform name="form01">
<cfinput type="button" name="startupload" value="Start Upload"
onclick="onUpload()">
</cfform>
```

ColdFusion.getElementValue

説明
バインド可能な ColdFusion コントロールの属性の値を取得します。

関数のシンタックス
```
ColdFusion.getElementValue(elementId [, formId, attributeName])
```

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
elementId	コントロールの ID 属性または name 属性です。
formId	そのコントロールを含むフォームの ID 属性です。要素 ID がページ上で一意な場合は、この属性を省略します。要素 ID が一意でない場合、この属性を省略すると、指定された ID を持ち、ページ上で最初に出現する要素が使用されます。
attributeName	取得するコントロール属性です。デフォルトでは value 属性になります。cfselect の場合は、コントロール内で選択された要素の値になります。 cfgrid コントロールの場合は、この属性を使用して、値を取得する列の名前を指定します。この場合は、現在選択されている行のエントリが返されます。 cftree コントロールの場合は、この属性を使用して、PATH または NODE を指定します。この場合は、現在選択されているツリー項目のパスまたはノードの値が返されます。

戻り値
指定された属性の値

使用方法
属性値へのバインドおよび属性値の取得が可能な HTML コントロールは次のとおりです。

- cfgrid
- checkbox、datefield、file、hidden、radio、または text タイプが指定されている cfinput コントロール
- cfselect
- cftextarea
- cftree

ColdFusion.grid.clearSelectedRows

説明
グリッドの選択した行をクリアする場合に使用します。

戻り値
なし

関数のシンタックス
ColdFusion.grid.clearSelectedRows(id)

パラメータ
- Id: cfgrid を使用して定義したグリッドの名前です。

使用方法
次の例を参照してください。

例
Employee.cfm

```
<html>
<head>
    <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
    <cfajaxproxy cfc="emp" jsclassname="emputils">
    <script language="javascript">
        var emp = new emputils();
        var deleteAllSelectedRows = function(grid)
        {
            emp.setHTTPMethod("POST");
            emp.deleteEmployees(getAllSelectedRows(grid,false));
            ColdFusion.Grid.refresh(grid);
        }
        var getAllSelectedRows = function(grid,showalert)
        {
            obj = ColdFusion.Grid.getSelectedRows(grid);
            jsonbj = ColdFusion.JSON.encode(obj);
            if(showalert)
                alert(jsonbj);
            return obj;
        }
        var clearAllSelectedRows = function(grid)
        {
            ColdFusion.Grid.clearSelectedRows(grid);
        }
    </script>
</head>
<body>
<cfform>
    <cfgrid
        format="html"
        name="empListing"
        selectmode="edit"

bind="cfc:emp.getEmployees({cfgridpage},{cfgridpagesize},{cfgridsortcolumn},{cfgridsortdirection})"
        onchange="cfc:emp.editEmployees({cfgridaction},{cfgridrow},{cfgridchanged})"
        autowidth="true"
        multirowselect=true
        delete="true"
        insert="true"
        title="Employee database"
        pagesize="25"
        >
        <cfgridcolumn name="EMP_ID" header="EMP_ID" select="false" display="false">
        <cfgridcolumn name="FIRSTNAME" header="First Name" select="true" />
        <cfgridcolumn name="LASTNAME" header="Last Name" select="true" />
        <cfgridcolumn name="DEPARTMENT" header="Department" select="true" />
        <cfgridcolumn name="EMAIL" header="Email" select="true" />
    </cfgrid>
    <br>
    <cfinput type="button" onClick="javascript:getAllSelectedRows('empListing',true)" name="getRows"
value="Get Selected Rows">
    <cfinput type="button" onClick="javascript:clearAllSelectedRows('empListing')" name="clearRows"
value="Clear Selected Rows">
    <cfinput type="button" onClick="javascript:deleteAllSelectedRows('empListing')" name="deleteRows"
value="Delete Selected Rows">
</cfform>
</body>
</html>
```

Employee.cfc

```
<cfcomponent>
    <cfscript>
        empQuery = new query(name="emps", datasource="cfdocexamples");
        remote any function
getEmployees(page,pageSize,gridsortcolumn="EMP_ID",gridsortdirection="ASC",empName)
        {
            var orderBy = "EMP_ID";
            var mysql = "SELECT Emp_ID, FirstName,  LastName, EMail, Department, Email FROM Employees";
            if(isdefined("arguments.empName") and trim(arguments.empName) neq ""){
                mysql = mysql & " WHERE " & "firstname = '#arguments.empName#'";
            }
            if(arguments.gridsortcolumn eq ""){
                mysql = mysql & " ORDER BY " & orderBy;
            }
            mysql = mysql & " " & gridsortdirection;
            return QueryConvertForGrid(empQuery.execute(sql=mysql).getResult(), page, pageSize);
        }
        remote void function editEmployees(gridaction,gridrow,gridchanged)
        {
            switch(gridaction)
            {
                case "I":
                {
                    var eFName = gridrow["FIRSTNAME"];
                    var eLNAme = gridrow["LASTNAME"];
                    var eDept = gridrow["DEPARTMENT"];
                    var eEmail = gridrow["EMAIL"];
                    var insertSql = "insert into Employees(FirstName,LastName,Department,Email) values ("
& "'" & eFName & "', '" & eLName & "', '" & eDept & "', '" & eEmail & "')";
                    empQuery.execute(sql=insertSql);
                    break;
                }
                case "U":
                {
                    var empId = gridrow["EMP_ID"];
                    var changedCol = structkeylist(gridchanged);
                    var updateSql =  "UPDATE Employees SET " & changedCol & "='" &  gridchanged[changedCol]
& "' WHERE emp_id=" & empId;
                    empQuery.execute(sql=updateSql);
                    break;
                }
                case "D":
                {
                    deleteEmployees(gridrow);
                }
```

```
            }
        }
        remote void function deleteEmployees(empdata)
        {
            var i = 1;
            var emp = {};
            if(isArray(empdata) and not ArrayIsEmpty(empdata)){
                for(emp in empdata){
                    if(isStruct(emp) and structkeyexists(emp,"emp_id")){
                        empid = emp["emp_id"];
                        writelog("deleting " & empid);
                        //var deleteSql =  "delete from Employees where emp_id=" & empid;
                        //empQuery.execute(sql=deleteSql);
                    }
                }
            }
        }

    </cfscript>
</cfcomponent>
```

この例では、multirowselect=true を設定して、複数のレコードを削除したりするバッチ操作をグリッドデータに対して実行しています。

この例はバッチ操作であるため、誤ってデータを削除しないように、deleteemployees 関数内の 2 行をコメントアウトしています。削除を確認する場合は、このコードのコメントアウトを外します。

フォームにある deleteAllSelectedRows ボタンは、外部からどのようにレコードを削除できるかを示します。つまり、グリッドに組み込まれた削除ボタンを使用せずに削除します。同じ方法を使用して、その他のバッチ操作 (複数ファイルの別フォルダへの移動など) やバッチ更新を実行することもできます。

注意：Employee.cfm の deleteAllSelectedRows メソッドに示すとおり、プロキシオブジェクトの httpMethod に POST を設定して、[Request URL Too Large] エラーが発生しないように注意してください。

ColdFusion.Grid.getBottomToolbar

説明

下部のツールバーを取得します。ツールバーは、アイコンやボタンなどのコントロールの追加に使用します。

関数のシンタックス

ColdFusion.Grid.getBottomToolbar(Id)

パラメータ

- Id: グリッドの名前です。

例

ColdFusion.Grid.refreshBottomToolbar の例を参照してください。

ColdFusion.Grid.getGridObject

説明

指定された HTML グリッドに対して、基盤となる Ext (Ext JS JavaScript library) オブジェクトを取得します。

関数のシンタックス

```
ColdFusion.Grid.getGridObject(name)
```

関連項目

cfgrid、ColdFusion.Grid.refresh、ColdFusion.Grid.sort、Ext JS - JavaScript ライブラリのマニュアル、『ColdFusion アプリケーションの開発』の Using HTML grids

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	そのオブジェクトを使用する cfgrid タグの name 属性の値です。

戻り値

グリッドが編集可能な場合は、Ext.grid.EditableGrid タイプのオブジェクトが返され、それ以外の場合は、Ext.grid.Grid タイプのオブジェクトが返されます。

使用方法

この関数は、ColdFusion HTML の cfgrid コントロールの基盤になる Ext Toolkit (Ext.grid) オブジェクトを取得するときに使用します。この未処理オブジェクトを使用することで、表示されるグリッドに変更を加えることができます。このオブジェクトの詳細および管理方法については、Ext のマニュアルを参照してください。

ColdFusion.Grid.hideBottomToolbar

説明

下部のツールバーを非表示にします。ツールバーは、アイコンやボタンなどのコントロールの追加に使用します。

関数のシンタックス

ColdFusion.Grid.hideBottomToolbar(Id)

パラメータ

- Id: グリッドの名前です。

例

ColdFusion.Grid.refreshBottomToolbar の例を参照してください。

ColdFusion.grid.getSelectedRows

説明

グリッドの選択した行のデータを取得する場合に使用します。

戻り値

行データが含まれたオブジェクトの配列

関数のシンタックス
ColdFusion.grid.getSelectedRows(id)

パラメータ
- Id: cfgrid を使用して定義したグリッドの名前です。

関連項目
FileUpload

使用方法
ColdFusion.grid.clearSelectedRows の例を参照してください。

例
ColdFusion.grid.clearSelectedRows の例を参照してください。

ColdFusion.Grid.getTopToolbar

説明
上部のツールバーを取得します。ツールバーは、アイコンやボタンなどのコントロールの追加に使用します。

関数のシンタックス
ColdFusion.getTopToolbar(Id)

パラメータ
- Id: グリッドの名前です。

例
ColdFusion.Grid.refreshBottomToolbar の例を参照してください。

ColdFusion.Grid.hideTopToolbar

説明
上部のツールバーを非表示にします。ツールバーは、アイコンやボタンなどのコントロールの追加に使用します。

関数のシンタックス
ColdFusion.Grid.hideTopToolbar(Id)

パラメータ
- Id: グリッドの名前です。

例
ColdFusion.Grid.refreshBottomToolbar の例を参照してください。

ColdFusion.Grid.refresh

説明
グリッドの表示を手動で更新します。

関数のシンタックス
```
ColdFusion.Grid.refresh(name [, preservePage])
```

関連項目
cfgrid、ColdFusion.Grid.getGridObject、ColdFusion.Grid.sort、Ext JS - JavaScript ライブラリのマニュアル、『ColdFusion アプリケーションの開発』の Using HTML grids

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	更新する cfgrid タグの name 属性の値です。
preservePage	現在のページを再表示するか (true)、最初のページを表示するか (false、デフォルト) を指定するブール値です。この属性は、グリッドデータの表示に複数のグリッドページが必要になる場合にのみ適用されます。

戻り値
この関数は値を返しません。

使用方法
この関数は、基盤となるデータを変更するが、通常、グリッドの更新を実行しないイベントが発生したときに、グリッドを更新するのに役立ちます。

例
次のコードは、ユーザーがグリッドから行を削除できる例です。ユーザーがグリッドの行を選択して削除ボタンをクリックすると、AJAX プロキシによって mycfc.deleteRow 関数が呼び出され、該当する行がデータベースから削除されます。この関数が正常に返されると、プロキシによって ColdFusion.Grid.refresh が呼び出され、グリッドの更新と行の削除が実行されます。

```
<cfajaxproxy bind="cfc:mycfc.deleteRow({deletebutton@click},{mygrid.id@none}"
    onSuccess="ColdFusion.Grid.refresh('mygrid', true)">
```

ColdFusion.Grid.refreshBottomToolbar

説明
下部のツールバーを更新します。ツールバーは、アイコンやボタンなどのコントロールの追加に使用します。この関数は、内部で JavaScript 関数 ColdFusion.Grid.showBottomToolbar を呼び出します。

関数のシンタックス
ColdFusion.Grid.refresheBottomToolbar(Id)

パラメータ

- Id: グリッドコントロールの名前です。

例

grid.cfc

```
<cfcomponent>
      <cfscript>
         remote any function getEmployees(page,pageSize,gridsortcolumn="EMP_ID",gridsortdirection="ASC"){
                  var startRow = (page-1)*pageSize;
                  var endRow = page*pageSize;

                  if(!isdefined("arguments.gridsortcolumn") or isdefined("arguments.gridsortcolumn") and
trim(arguments.gridsortcolumn) eq "")
                        gridsortcolumn = "EMP_ID";
                 if(!isdefined("arguments.gridsortdirection") or isdefined("arguments.gridsortdirection")
and arguments.gridsortdirection eq "")
                        gridsortdirection = "ASC";
                  var mysql = "SELECT Emp_ID, FirstName,  EMail, Department FROM Employees";
                  if(isdefined("arguments.gridsortcolumn") and arguments.gridsortcolumn neq "")
                        mysql = mysql & " ORDER BY " & gridsortcolumn;
                  if(isdefined("arguments.gridsortdirection") and arguments.gridsortdirection neq "")
                        mysql = mysql & " " & gridsortdirection ;
                  rs1 = new query(name="team", datasource="cfdocexamples", sql=mysql).execute();
                        return QueryConvertForGrid(rs1.getResult(), page, pageSize);
            }


            remote any function editEmployees(gridaction,gridrow,gridchanged){
                        writelog("edit employee info");
            }

      </cfscript>
</cfcomponent>
```

grid.cfm

```
<script>
      var refreshToolbar = function(id,type){
         type == "top" ? ColdFusion.Grid.refreshTopToolbar(id) : ColdFusion.Grid.refreshBottomToolbar(id);
      }

      var hideToolbar = function(id,type){
            type == "top" ? ColdFusion.Grid.hideTopToolbar(id) : ColdFusion.Grid.hideBottomToolbar(id);
      }

      var showToolbar = function(id,type){
            (type == "top") ? ColdFusion.Grid.showTopToolbar(id) : ColdFusion.Grid.showBottomToolbar(id);
      }

      var handleToolbar = function(id,type){
            if(type == "top"){
                  tbar = ColdFusion.Grid.getTopToolbar(id);
                  tbar.addButton({
                        text: "Add User Account",
                        tooltip: "Add a user account",
                        handler: addUserAccount
                  });
            }
            else{
                  bbar = ColdFusion.Grid.getBottomToolbar(id);
                  bbar.add(new Ext.Toolbar.Separator());
```

```
                bbar.addButton({
                      text: "Delete User Account",
                      tooltip: "Delete a user account",
                      handler: deleteUserAccount
                });
          }
    }

    var GetUserInfo = function(){
          alert("Retrieving user account");
    }

    var addUserAccount = function(){
          alert("Adding new user account")
    }
    var deleteUserAccount = function(){
          alert("Deleting user account")
    }
</script>
<cfform>
    <br>
    <cfinput type="button" onClick="showToolbar('empGrid','top')" name="btn1" value="Show Top Toolbar">
    <cfinput type="button" onClick="handleToolbar('empGrid','top')" name="btn2" value="Add button to Top
Toolbar">
    <cfinput type="button" onClick="refreshToolbar('empGrid','top')" name="btn3" value="Refresh Top
Toolbar">
    <cfinput type="button" onClick="hideToolbar('empGrid','top')" name="btn4" value="Hide Top Toolbar">
    <br><br>

    <cfgrid
                format="html"
                name="empGrid"
                width="800"
                pagesize=5
                sort=true
                title="Employee database"
                collapsible="true"
                insert="yes"
                delete="yes"

bind="cfc:grid.getEmployees({cfgridpage},{cfgridpagesize},{cfgridsortcolumn},{cfgridsortdirection})"
                onChange="cfc:grid.editEmployees({cfgridaction},{cfgridrow},{cfgridchanged})"
                selectMode="edit"
                >
                <cfgridcolumn name="Emp_ID" display=false header="ID" />
                <cfgridcolumn name="FirstName" display=true header="First Name"/>
                <cfgridcolumn name="Email" display=true header="Email"/>
                <cfgridcolumn name="Department" display=true header="Department" />
    </cfgrid>

    <br><br>
    <cfinput type="button" onClick="hideToolbar('empGrid','bottom')" name="btn5" value="Hide Bottom
Toolbar">
    <cfinput type="button" onClick="showToolbar('empGrid','bottom')" name="btn6" value="Show Bottom
Toolbar">
    <cfinput type="button" onClick="handleToolbar('empGrid','bottom')" name="btn7" value="Add button to
Bottom Toolbar">
    <cfinput type="button" onClick="refreshToolbar('empGrid','bottom')" name="btn8" value="Refresh Bottom
Toolbar">
</cfform>
```

ColdFusion.Grid.refreshTopToolbar

説明
上部のツールバーを更新します。ツールバーは、アイコンやボタンなどのコントロールの追加に使用します。この関数は、内部で JavaScript 関数 ColdFusion.Grid.showTopToolbar を呼び出します。

関数のシンタックス
ColdFusion.Grid.refreshTopToolbar(Id)

パラメータ
- Id: グリッドの名前です。

例
ColdFusion.Grid.refreshBottomToolbar の例を参照してください。

ColdFusion.Grid.showBottomToolbar

説明
下部のツールバーを表示します。ツールバーは、アイコンやボタンなどのコントロールの追加に使用します。

関数のシンタックス
ColdFusion.Grid.showBottomToolbar(Id)

パラメータ
- Id: グリッドの名前です。

例
ColdFusion.Grid.refreshBottomToolbar の例を参照してください。

ColdFusion.Grid.showTopToolbar

説明
上部のツールバーを表示します。ツールバーは、アイコンやボタンなどのコントロールの追加に使用します。

関数のシンタックス
ColdFusion.Grid.showTopToolbar(Id)

パラメータ
- Id: グリッドの名前です。

例
ColdFusion.Grid.refreshBottomToolbar の例を参照してください。

ColdFusion.Grid.sort

説明
指定された HTML グリッドをソートします。

関数のシンタックス
```
ColdFusion.Grid.sort(name [, columnName, direction])
```

関連項目
cfgrid、ColdFusion.Grid.getGridObject、ColdFusion.Grid.refresh、Ext JS - JavaScript ライブラリのマニュアル、『ColdFusion アプリケーションの開発』の Using HTML grids

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	ソートする cfgrid タグの name 属性の値です。
columnName	ソート順を決定する列の名前です。
direction	ソートの方向です。次のいずれかの値を指定する必要があります。 • ASC (デフォルト) • DESC

戻り値
この関数は値を返しません。

使用方法
この関数は、グリッドに表示されるデータをソートします。文字列データの場合は大文字と小文字が区別されずにソートされ、数値データの場合は数値としてソートされます。指定された列の内容を基準にして、表示されるグリッドの順序が決定されます。リモートデータソースを使用するグリッドの場合は、バインド属性 cfgridsortcolumn および cfgridsortdirection を通じて、データを提供する CFC 関数に列名とソート方向が通知されます。この CFC 関数がこれらの値を使用して適切にソートを実行できることを確認する必要があります。

ColdFusion.JSON.decode

説明
JSON エンコードの文字列を JavaScript 変数に変換します。

関数のシンタックス
```
ColdFusion.JSON.decode(string)
```

関連項目
ColdFusion.JSON.encode、DeserializeJSON、SerializeJSON、『ColdFusion アプリケーションの開発』の Using Ajax Data and Development Features、http://www.json.org

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
string	エンコードする文字列です。

戻り値
JSON エンコード文字列のデータを含む JavaScript 変数。

使用方法
この関数は、JavaScript と JSON の形式間で明示的に変換を行う必要がある場合に使用します。たとえば、CFC にないリモート関数を呼び出す必要がある場合などです。

JSON 文字列にセキュリティ接頭辞が含まれている場合は、文字列をデコードする前に接頭辞を取り除きます。セキュリティ接頭辞は、ColdFusion Administrator の [サーバーの設定]-[設定] ページで定義されているか、cfapplication タグまたは cffunction タグで指定されています。

例
次の例では、ColdFusion.JSON.decode 関数と ColdFusion.JSON.encode 関数を使用しています。ユーザーが "Call" リンクをクリックすると、callMe 関数によって String が JSON としてエンコードされ、echo CFC の plainEcho 関数が結果とともに呼び出されます。また、関数によって戻り値の形式が plain に設定されるため、CFC 関数は戻り値を JSON に自動変換せず、代わりにプレーンテキストを送信します。

echo.cfc コンポーネントには、次の 2 つの関数があります。

- plainEcho 関数は、引数を JSON から ColdFusion 変数に変換し、echo 関数を呼び出し、結果を JSON に変換して呼び出し元に返します。
- echo 関数は、構造体を作成し、構造体のエントリを入力パラメータに設定し、結果を返します。(プレーンの戻り型をリクエストする際に JSON をエンコードしない関数を呼び出した結果を参照する目的で、この関数をリモートから呼び出すことができます。結果を表示するには、メインページを実行する際に cfdebug HTTP パラメータを使用します。)

  メインページには次の行が含まれています。

```
<cfajaxproxy cfc="echo">
<cfajaximport>

<html>
    <head>
        <script>
        function callme()
        {
            var e = new echo();
            e.setReturnFormat('plain');
            var args = {a:"Hello again!"};
            var argsJSON = ColdFusion.JSON.encode(args);
            var json = e.plainEcho(argsJSON);
            var o = ColdFusion.JSON.decode(json);
            alert(o.A);
        }
        </script>
    </head>

    <body>
        <a href="javascript:callme()">Call</a>
    </body>
</html>
```

echo.cfc ファイルには次の行が含まれています。

```
<cfcomponent output="false">

    <cffunction name="echo" access="remote">
        <cfargument name="text">
        <cfset var ret = StructNew()>
        <cfset ret.a = text>
        <cfreturn ret>
    </cffunction>

    <cffunction name="plainEcho" access="remote">
        <cfargument name="text">
        <cfset t = deserializeJSON(text)>
        <cfset ret = echo(t.A)>
        <cfreturn serializeJSON(ret)>
    </cffunction>

</cfcomponent>
```

ColdFusion.JSON.encode

説明

JavaScript 式を JSON エンコードの文字列に変換します。

関数のシンタックス

```
ColdFusion.JSON.encode(expression)
```

関連項目

ColdFusion.JSON.decode、DeserializeJSON、SerializeJSON、『ColdFusion アプリケーションの開発』の Using Ajax Data and Development Features、**http://www.json.org**

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	エンコードするデータを含む式です。

戻り値
JSON エンコード形式のデータを含む文字列。

使用方法
この関数は、JavaScript と JSON の形式間で明示的に変換を行う必要がある場合に使用します。たとえば、CFC にないリモート関数を呼び出す必要がある場合などです。

例
ColdFusion.JSON.decode の例を参照してください。

ColdFusion.Layout.collapseAccordion

説明
アコーディオンレイアウト領域を折り畳みます。

関数のシンタックス
```
ColdFusion.Layout.collapseAccordion(layoutname, layoutareaname)
```

関連項目
cflayout、ColdFusion.Layout.createAccordionPanel、ColdFusion.Layout.expandAccordion、ColdFusion.Layout.getAccordionLayout、ColdFusion.Layout.hideAccordion、ColdFusion.Layout.selectAccordion、ColdFusion.Layout.showAccordion、『ColdFusion アプリケーションの開発』の Using layouts

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
layoutname	折り畳むパネルが含まれているアコーディオンレイアウトの name 属性です。
layoutareaname	折り畳む対象となる、アコーディオンレイアウト内のパネルの名前です。

戻り値
この関数は値を返しません。

使用方法
この関数は、アコーディオンが既に折り畳まれている場合には効果がありません。

例
次のコードでは、ユーザーがボタンをクリックすると、アコーディオンレイアウトが折り畳まれます。

```
<cfinput name="collapse2" width="100" value="Collapse Area 2" type="button"
        onClick="ColdFusion.Layout.collapseAccordion('thelayout', 'panel2')>
```

ColdFusion.Layout.collapseArea

説明
ボーダー付きレイアウトの領域を折り畳みます。

関数のシンタックス
```
ColdFusion.Layout.collapseArea(layout, layoutArea)
```

関連項目
cflayout、cflayoutarea、ColdFusion.Layout.expandArea、ColdFusion.Layout.getTabLayout、ColdFusion.Layout.showArea、『ColdFusion アプリケーションの開発』の Using layouts

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
layout	折り畳む領域が含まれているボーダー付きレイアウトの name 属性です。
layoutArea	折り畳む領域のレイアウト内の位置です。bottom、left、right、top のいずれかになります。

戻り値
この関数は値を返しません。

使用方法
この関数は、領域が既に折り畳まれている場合には効果がありません。

例
次のコードでは、ユーザーがボタンをクリックすると、ボーダー付きレイアウトの左側の領域が折り畳まれます。

```
<cfinput name="collapse2" width="100" value="Collapse Area 2" type="button"
        onClick="ColdFusion.Layout.collapseArea('thelayout', 'left');">
```

ColdFusion.Layout.createAccordionPanel

説明
ColdFusion アコーディオンレイアウトにパネルを作成します。

関数のシンタックス
```
ColdFusion.Layout.createAccordionPanel(layoutname, layoutareaname, title, URL [, config])
```

関連項目

cflayout、ColdFusion.Layout.collapseAccordion、ColdFusion.Layout.expandAccordion、ColdFusion.Layout.getAccordionLayout、ColdFusion.Layout.hideAccordion、ColdFusion.Layout.selectAccordion、ColdFusion.Layout.showAccordion、『ColdFusion アプリケーションの開発』の Using layouts

履歴

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
layoutname	パネルを追加するアコーディオンレイアウトの name 属性です。
layoutareaname	新しいアコーディオンパネルに割り当てる名前です。アコーディオン上で一意である必要があります。
title	パネルに表示するテキストです。HTML マークアップを使用してタイトルの外観を制御することもできます。
URL	パネル領域のコンテンツをどの URL から取得するかを指定します。この属性で URL パラメータを使用すると、ページにデータを渡すことができます。ColdFusion は標準のページパス解決ルールを使用してページを探します。
config	設定パラメータを含むオブジェクトです。詳細については、「使用方法」を参照してください。

戻り値

この関数は値を返しません。

使用方法

この関数は、アコーディオンレイアウト内のパネルをダイナミックに作成します。この関数には、type 属性が accordion に設定された cflayout タグの中に cflayoutarea タグを配置した場合と同じ効果があります。**configuration** パラメータはパネルの特性を定義します。このパラメータでは、次のエントリの一部またはすべてを指定できます。

エントリ	デフォルト	説明
align	cflayout タグの align 属性の値	パネル領域内の子コントロールをどのように配置するかを指定します。使用できる値は次のとおりです。 • center • justify • left • right
callbackHandler		レイアウトのアコーディオン本体がロードされたときに呼び出される関数です。この関数には引数を渡せません。
errorHandler		タブ本体のロード時にエラーが発生した場合に呼び出される関数です。この関数は次の 2 つの引数を取る必要があります。 • HTTP ステータスコード (エラーが HTTP のエラーでない場合は -1) • エラーメッセージ

エントリ	デフォルト	説明
overflow	auto	子コンテンツのサイズが大きいためにタブのレイアウト領域がウィンドウの境界を超えてしまう場合に、子コンテンツをどのように表示するかを指定します。使用できる値は次のとおりです。 • auto: 必要に応じてスクロールバーを表示します。 • hidden: 境界からはみ出したコンテンツにアクセスできないようにします。 • scroll: 必要がない場合でも常に水平および垂直方向のスクロールバーを表示します。 • visible: レイアウト領域の外側にコンテンツを表示します。 **メモ :** Internet Explorer では、レイアウト領域を visible に設定した場合、レイアウト領域を超えてコンテンツを表示するのではなく、コンテンツのサイズに合わせてレイアウト領域が拡張されます。
selected	false	このタブを最初から選択された状態にして、そのコンテンツをレイアウト内に表示するかどうかを指定するブール値です。
style		レイアウト領域の外観を制御する CSS スタイル指定です。
titleicon		タイトルとともに表示するアイコンの位置を指定します。

例

次の例では、パネルが 1 つ含まれるアコーディオンレイアウトを作成します。ボタンをクリックすると、2 番目のパネルが作成され、選択された状態で直ちに表示されます。

メインページは次のようになります。

```
<html>
<head>
</head>
<body>
<cfform name="panels">
    <cfinput type="button" name="CreateAccordionPanel"
        onClick="ColdFusion.Layout.createAccordionPanel('AccordionPanel','panel2',
            'Panel 2','_panelUrl.cfm',{inithide:false,selected:true})"
        value="Create Panel">
</cfform>

<cflayout type="panel" name="AccordionPanel">
    <cflayoutarea name="panel1" title="Panel 1" align="left">
        Default Panel
    </cflayoutarea>
</cflayout>
</body>
</html>
```

_tabURL.cfm ページは次のようになります。

```
<h3>Panel 2</h3>
This is an accordion panel
```

ColdFusion.Layout.createTab

説明

ColdFusion のタブ付きレイアウト内にタブとレイアウト領域を作成します。

関数のシンタックス

```
ColdFusion.Layout.createTab(layout, layoutArea, Title, URL [, configObject])
```

関連項目

cflayout、cflayoutarea、ColdFusion.Layout.disableTab、ColdFusion.Layout.enableTab、ColdFusion.Layout.showArea、ColdFusion.Layout.hideTab、ColdFusion.Layout.selectTab、ColdFusion.Layout.showTab、『ColdFusion アプリケーションの開発』の Using layouts

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
layout	タブを追加するタブ付きレイアウトの name 属性です。
layoutArea	新しいタブを追加するために作成されるレイアウト領域の名前です。ページ内で一意である必要があります。
title	タブに表示するテキストです。HTML マークアップを使用してタイトルの外観を制御することもできます。
URL	レイアウト領域のコンテンツをどの URL から取得するかを指定します。この属性で URL パラメータを使用すると、ページにデータを渡すことができます。ColdFusion は標準のページパス解決ルールを使用してページを探します。
configObject	ウィンドウ設定パラメータを含むオブジェクトです。詳細については、「使用方法」を参照してください。

戻り値

この関数は値を返しません。

使用方法

この関数は、タブ付きレイアウト内のタブをダイナミックに作成します。この関数には、type 属性が tab に設定された cflayout タグの中に cflayoutarea タグを配置した場合と同じ効果があります。**configuration** パラメータはタブの特性を定義します。このパラメータでは、次のエントリの一部またはすべてを指定できます。

エントリ	デフォルト	説明
align	cflayout タグの align 属性の値	レイアウト領域内の子コントロールをどのように配置するかを指定します。使用できる値は次のとおりです。 • center • justify • left • right
callbackhandler		レイアウトのタブ本体がロードされたときに呼び出される関数です。この関数には引数を渡せません。
closable	false	ユーザーがウィンドウを閉じることを許可するかどうかを指定するブール値です。true の場合は、タブ上に X 印のアイコンが表示されます。
disabled	false	タブを無効にするかどうか (ユーザーがタブを選択してそのコンテンツを表示できるようにするかどうか) を指定するブール値です。無効なタブはグレーで表示されます。 selected が true の場合は無視されます。
errorHandler		タブ本体のロード時にエラーが発生した場合に呼び出される関数です。この関数は次の 2 つの引数を取る必要があります。 • HTTP ステータスコード (エラーが HTTP のエラーでない場合は -1) • エラーメッセージ

エントリ	デフォルト	説明
inithide	false	初期状態でタブを非表示にするかどうかを指定するブール値です。初期状態で非表示になっているタブを表示するには、ColdFusion.Layout.showTab 関数を使用します。
overflow	auto	子コンテンツのサイズが大きいためにタブのレイアウト領域がウィンドウの境界を超えてしまう場合に、子コンテンツをどのように表示するかを指定します。使用できる値は次のとおりです。 • auto: 必要に応じてスクロールバーを表示します。 • hidden: 境界からはみ出したコンテンツにアクセスできないようにします。 • scroll: 必要がない場合でも常に水平および垂直方向のスクロールバーを表示します。 • visible: レイアウト領域の外側にコンテンツを表示します。 **メモ :** Internet Explorer では、レイアウト領域を visible に設定した場合、レイアウト領域を超えてコンテンツを表示するのではなく、コンテンツのサイズに合わせてレイアウト領域が拡張されます。
selected	false	このタブを最初から選択された状態にして、そのコンテンツをレイアウト内に表示するかどうかを指定するブール値です。
style		レイアウト領域の外観を制御する CSS スタイル指定です。

例

次の例では、1 つのタブが含まれたタブ付きレイアウトを作成します。ボタンをクリックすると、2 番目のタブが作成され、選択された状態で直ちに表示されます。

メインページは次のようになります。

```
<html>
<head>
</head>
<body>
<cfform name="layouts">
    <cfinput type="button" name="CreateTab"
        onClick="ColdFusion.Layout.createTab('tabLayout','tab2',
            'Tab 2','_tabUrl.cfm',{inithide:false,selected:true})"
        value="Create Tab">
</cfform>

<cflayout type="tab" name="tabLayout">
    <cflayoutarea name="tab1" title="Tab 1" align="left">
        Default Tab
    </cflayoutarea>
</cflayout>
</body>
</html>
```

_tabURL.cfm ページは次のようになります。

```
<h3>Tab 2</h3>
This is a simple tab
```

ColdFusion.Layout.disableSourceBind

説明

ソースバインドを無効にします。

関数のシンタックス
ColdFusion.Layout.disableSourceBind(Id)

パラメータ
- Id: レイアウト領域の名前です。

使用方法
たとえば、Coldfusion.navigate を使用して、タブまたはアコーディオンパネルに内容を挿入するとします。cflayoutarea で source 属性が定義されている場合は、(ColdFusion.navigate から取得するのではなく) ソースバインドの呼び出しで取得した内容のインスタンスを使用できます。

このようなインスタンスでは、ソースバインドを無効にして、Coldfusion.navigate で内容を取得したい場合が考えられます。

例
layout.cfm では、Tab1_Src.cfm、Tab2_Src.cfm、および Tab3_Src.cfm のテンプレートを使用します。layout.cfm を実行すると、次のことがわかります。

- [Navigate] をクリックすると、navigate.cfm ではなく tab2_src.cfm の内容が挿入されます。
- [Disable Source Bind] をクリックすると、navigate.cfm の内容が tab2_src に挿入されます。
- [Enable Source Bind] をクリックしてから、tab2_src をクリックすると、tab2_src の内容が再び挿入されます。

Tab1_Src.cfm

```
<br><cfdump var="#CGI#" keys="15" label="[CGI scope]"><br>
```

Tab2_Src.cfm

```
<br><cfdump var="#server#" label="[Server scope]"><br>
```

Tab3_Src.cfm

```
<br><cfdump var="#server.coldfusion#" label="[Showing key coldfusion in server scope]"><br>
```

Tab4_Src.cfm

```
<br><cfdump var="#server.os#" label="[Showing key OS in server scope]"><br>
```

layout.cfm

```
<script>
      var navigateToTab = function(layoutId,tabId){
            alert("Navigating to " + tabId);
            ColdFusion.Layout.selectTab(layoutId,tabId);
            ColdFusion.navigate('navigate.cfm',tabId);
      }
      var disableBind = function(tabId){
            alert("Disabling binding on source for " + tabId);
            ColdFusion.Layout.disableSourceBind(tabId);
      }
      var enableBind = function(tabId){
            alert("Enabling binding on source for " + tabId);
            ColdFusion.Layout.enableSourceBind(tabId);
      }

</script>
<cflayout type="tab" name="layout1">
      <cflayoutarea
            name = "tab1"
            overflow = "auto"
            refreshonactivate = "yes"
            title = "Tab 1"
```

```
            source = "tab1_src.cfm"/>
      <cflayoutarea
            name = "tab2"
            overflow = "auto"
            refreshonactivate = "false"
            title = "Tab 2"
            source = "tab2_src.cfm"
            bindonload=false
            />
      <cflayoutarea
            name = "tab3"
            overflow = "auto"
            refreshonactivate = "yes"
            title = "Tab 3"
            source = "tab3_src.cfm"
            />
</cflayout>
<cfform name="myform">
      <cfinput type="button" name="disable" value="Disable Source Bind"
onClick="javascript:disableBind('tab2')">
     <cfinput type="button" name="b" value="Navigate" onClick="javascript:navigateToTab('layout1','tab2')">
      <cfinput type="button" name="disable" value="Enable Source Bind"
onClick="javascript:enableBind('tab2')">
</cfform>
```

ColdFusion.Layout.enableSourceBind

説明
ソースバインドが無効になっている場合は有効にします。

関数のシンタックス
ColdFusion.Layout.enableSourceBind(Id)

パラメータ
- Id: レイアウト領域の名前です。

使用方法
ColdFusion.Layout.disableSourceBind の「使用方法」を参照してください。

例
ColdFusion.Layout.disableSourceBind の例を参照してください。

ColdFusion.FileUpload.getSelectedFiles

説明
アップロード対象として選択されたファイルの名前とサイズが含まれるオブジェクトの配列を返します。ファイルサイズは、バイト単位で返されます。

また、この関数は、ファイルアップロードステータスを YES|NO|Error で返します。

関数のシンタックス
ColdFusion.FileUpload.getSelectedFiles(Id)

パラメータ
- Id: cffileupload コントロールの名前です。

使用方法
実際のシナリオでは、通常、アップローダは別のコントロールと一緒に使用します。たとえば、名前、電子メール、アップローダの 3 つのフィールドがあるフォームなどがあります。仮に、ファイルをアップロードしたが、[送信] をクリックし忘れてしまった、またはファイルを選択してフォームを送信したが、[アップロード] をクリックし忘れてしまったということがあるとします。

この関数を使用すると、アップロード対象として選択したファイルがあることや、次の詳細情報をユーザーに通知することができます。

- FILENAME: アップロード対象として選択されたファイルの名前です。
- SIZE: ファイルのサイズです (バイト単位)。
- STATUS：YES|NO|Error。YES はアップロードが成功したことを示し、NO はアップロードがまだ実行されていないことを示し、Error はアップロード処理中に例外が発生したことを示します。

例
次の例は、ユーザーが [Submit] をクリックすると、アップロード対象として選択されたファイルに関する通知が行われるシナリオを示しています。

```
<html>
<head>
    <script language="javascript">
        var formatNumber = function(num){
            if(num < 1024) return num + "  bytes";
            if(num < (1024 * 1024)) return (num/1024).toFixed(2) + " KB";
            if(num < (1024 * 1024  *1024)) return (num/(1024 * 1024)).toFixed(2) + " MB";
            return (num/(1024 * 1024 * 1024)).toFixed(2) + " GB";
        }
        var getSelectedList = function(id){
            var files = ColdFusion.FileUpload.getSelectedFiles(id);
            var fileslist = "";
            if(files.length)
                fileslist = "You have selected The following files for upload: \n\n";
            for(var i=0;i < files.length; i++){
                fileslist = fileslist + files[i].FILENAME + "  (" + formatNumber(files[i].SIZE) + ")"
                if(i != files.length-1)
                    fileslist = fileslist + "\r\n";
            }
            if(files.length)
            {
```

```
                alert(fileslist);
            }
        }
    </script>
</head>
<body>
<br>
<cfform name="frmUpload" method="POST">
    First Name: <cfinput type="text" name="fname" value=""><br>
    Last Name: <cfinput type="text" name="lname" value=""><br><br>
    <cffileupload
        url="uploadAll.cfm"
        name="myuploader1"
        hideUploadButton=false
        onUploadComplete="foo"
        /><br><br>
<cfinput type="button" name="submitForm2" value="Submit" onClick="getSelectedList('myuploader1')">
</cfform>
</body>
</html>
```

ColdFusion.Layout.disableTab

説明
特定のタブを無効にして選択できないようにします。

関数のシンタックス
```
ColdFusion.Layout.disableTab(layout, layoutArea)
```

関連項目
cflayout、cflayoutarea、ColdFusion.Layout.createTab、ColdFusion.Layout.enableTab、ColdFusion.Layout.showArea、ColdFusion.Layout.hideTab、ColdFusion.Layout.selectTab、ColdFusion.Layout.showTab、『ColdFusion アプリケーションの開発』の Using layouts

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
layout	無効にする領域が含まれているタブ付きレイアウトの name 属性です。
layoutArea	無効にするタブ付きレイアウトの領域の name 属性です。

戻り値
この関数は値を返しません。

使用方法
この関数は、現在選択されているタブには効果がありません。無効なタブはグレーで表示されます。

例
次の例では、リンクをクリックしてタブを有効または無効にできます。

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
</head>

<body>
<!--- The tabheight attribute sets the height of all tab content areas. --->
<cflayout type="tab" name="mainTab" tabheight="300px" style="width:400px">
    <cflayoutarea title="First Tab" name="tab1">
        <h2>The First Tab</h2>
        Here are the contents of the first tab.
    </cflayoutarea>

    <cflayoutarea title="Second Tab" name="tab2">
        <h2>The Second Tab</h2>
        This is the content of the second tab.
    </cflayoutarea>
</cflayout>

<p>
Use these links to test disabling/enabling via JavaScript.
Note that you cannot disable the currently selected tab.<br />
<a href="" onClick="ColdFusion.Layout.enableTab('mainTab','tab1');
    return false;">Click here to enable tab 1.</a><br />
<a href="" onClick="ColdFusion.Layout.disableTab('mainTab','tab1');
    return false;">Click here to disable tab 1.</a><br />
</p>
</body>
</html>
```

ColdFusion.Layout.enableTab

説明

特定のタブを有効にして選択できるようにします。

関数のシンタックス

```
ColdFusion.Layout.enableTab(layout, layoutArea)
```

関連項目

cflayout、cflayoutarea、ColdFusion.Layout.createTab、ColdFusion.Layout.disableTab、ColdFusion.Layout.showArea、ColdFusion.Layout.hideTab、ColdFusion.Layout.selectTab、ColdFusion.Layout.showTab、『ColdFusion アプリケーションの開発』の Using layouts

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
layout	有効にする領域が含まれているタブ付きレイアウトの name 属性です。
layoutArea	有効にするタブ付きレイアウトの領域の name 属性です。

戻り値
この関数は値を返しません。

例
ColdFusion.Layout.disableTab を参照してください。

ColdFusion.Layout.expandAccordion

説明
アコーディオンレイアウトのパネルを展開します。

関数のシンタックス
```
ColdFusion.Layout.expandAccordion(layoutname, layoutareaname)
```

関連項目
cflayout、ColdFusion.Layout.collapseAccordion、ColdFusion.Layout.createAccordionPanel、ColdFusion.Layout.getAccordionLayout、ColdFusion.Layout.hideAccordion、ColdFusion.Layout.selectAccordion、ColdFusion.Layout.showAccordion、『ColdFusion アプリケーションの開発』の Using layouts

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
layoutname	展開するパネルが含まれているアコーディオンレイアウトの name 属性です。
layoutareaname	展開するパネルの名前です。bottom、left、right、top のいずれかになります。

戻り値
この関数は値を返しません。

使用方法
この関数は、パネルが既に展開されている場合には効果がありません。

例
次のコードでは、ユーザーがボタンをクリックすると、パネルの左側の領域が展開されます。

```
<cfinput name="expand2" width="100" value="Expand Area 2" type="button"
        onClick="ColdFusion.Layout.expandAccordion('thelayout', 'left');">
```

ColdFusion.Layout.expandArea

説明
ボーダー付きレイアウトの領域を展開します。

関数のシンタックス

```
ColdFusion.Layout.expandArea(layout, layoutArea)
```

関連項目

cflayout、cflayoutarea、ColdFusion.Layout.collapseArea、ColdFusion.Layout.getTabLayout、ColdFusion.Layout.showArea、『ColdFusion アプリケーションの開発』の Using layouts

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
layout	展開する領域が含まれているボーダー付きレイアウトの name 属性です。
layoutArea	展開する領域のレイアウト内の位置です。bottom、left、right、top のいずれかになります。

戻り値

この関数は値を返しません。

使用方法

この関数は、領域が既に展開されている場合には効果がありません。

例

次のコードでは、ユーザーがボタンをクリックすると、ボーダー付きレイアウトの左側の領域が展開されます。

```
<cfinput name="expand2" width="100" value="Expand Area 2" type="button"
        onClick="ColdFusion.Layout.expandArea('thelayout', 'left');">
```

ColdFusion.Layout.getAccordionLayout

説明

指定されたアコーディオンレイアウトに対して、基盤となる Ext (Ext JS JavaScript library) オブジェクトを取得します。

関数のシンタックス

```
ColdFusion.Layout.getAccordionLayout(name)
```

関連項目

cflayout、ColdFusion.Layout.collapseAccordion、ColdFusion.Layout.createAccordionPanel、ColdFusion.Layout.expandAccordion、ColdFusion.Layout.hideAccordion、ColdFusion.Layout.selectAccordion、ColdFusion.Layout.showAccordion、『ColdFusion アプリケーションの開発』の Using layouts

履歴

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
name	そのオブジェクトを使用する accordion タイプの cflayout タグの name 属性の値です。

戻り値
Ext.AccordionLayout タイプのオブジェクト

使用方法
この関数は、ColdFusion HTML の cflayout コントロールの基盤になる Ext Toolkit (Ext.AccordionLayout) オブジェクトを取得するときに使用します。この未処理オブジェクトを使用することで、表示されるレイアウトに変更を加えることができます。このオブジェクトの詳細および管理方法については、Ext のマニュアルを参照してください。

ColdFusion.Layout.getBorderLayout

説明
指定されたボーダー付きレイアウトに対して、基盤となる Ext (Ext JS JavaScript library) オブジェクトを取得します。

関数のシンタックス
```
ColdFusion.Layout.getBorderLayout(name)
```

関連項目
cflayout、cflayoutarea、ColdFusion.Layout.getTabLayout、Ext JS - JavaScript ライブラリのマニュアル、『ColdFusion アプリケーションの開発』の Using layouts

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	そのオブジェクトを使用する border タイプの cflayout タグの name 属性の値です。

戻り値
Ext.BorderLayout タイプのオブジェクト

使用方法
この関数は、ColdFusion HTML の cflayout コントロールの基盤になる Ext Toolkit (Ext.BorderLayout) オブジェクトを取得するときに使用します。この未処理オブジェクトを使用することで、表示されるレイアウトに変更を加えることができます。このオブジェクトの詳細および管理方法については、Ext のマニュアルを参照してください。

ColdFusion.Layout.getTabLayout

説明
指定されたタブ付きレイアウトに対して、基盤となる Ext (Ext JS JavaScript library) オブジェクトを取得します。

関数のシンタックス

```
ColdFusion.Layout.getTabLayout(name)
```

関連項目

cflayout、cflayoutarea、ColdFusion.Layout.getBorderLayout、Ext JS - JavaScript ライブラリのマニュアル、『ColdFusion アプリケーションの開発』の Using layouts

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	そのオブジェクトを使用する border タイプの cflayout タグの name 属性の値です。

戻り値

Ext.BorderLayout タイプのオブジェクト

使用方法

この関数は、ColdFusion HTML の cflayout コントロールの基盤になる Ext Toolkit (Ext.BorderLayout) オブジェクトを取得するときに使用します。この未処理オブジェクトを使用することで、表示されるレイアウトに変更を加えることができます。このオブジェクトの詳細および管理方法については、Ext のマニュアルを参照してください。

ColdFusion.Layout.hideAccordion

説明

指定されたパネルとそのアコーディオンレイアウトを非表示にします。

関数のシンタックス

```
ColdFusion.Layout.hideAccordion(layoutname, layoutareaname)
```

関連項目

cflayout、ColdFusion.Layout.collapseAccordion、ColdFusion.Layout.createAccordionPanel、ColdFusion.Layout.expandAccordion、ColdFusion.Layout.getAccordionLayout、ColdFusion.Layout.selectAccordion、ColdFusion.Layout.showAccordion、『ColdFusion アプリケーションの開発』の Using layouts

履歴

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
layoutname	非表示にするパネルが含まれているアコーディオンレイアウトの name 属性です。
layoutareaname	非表示にするパネルの name 属性です。

戻り値
この関数は値を返しません。

例
次の例では、パネルが 2 つ含まれるアコーディオンレイアウトを作成します。ボタンをクリックすると、2 番目のパネルを表示または非表示にできます。

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
</head>

<body>
<cflayout type="accordion" name="accordionLayout" accordionheight="300px"
    style="width:400px">
    <cflayoutarea title="First Panel" name="panel1">
        <h2>The First Panel</h2>
        Here are the contents of the first panel.
    </cflayoutarea>

    <cflayoutarea title="Second Panel" name="panel2">
        <h2>The Second Panel</h2>
        This is the content of the second panel.
    </cflayoutarea>
</cflayout>
<br />

<cfform name="layouts">
    <cfinput type="button" name="ShowAccordion" value="Show Accordion"
        onClick="ColdFusion.Layout.showAccordion('accordionLayout','panel2')">
    <cfinput type="button" name="ShowAccordion" value="Hide Panel"
        onClick="ColdFusion.Layout.hideAccordion('accordionLayout','panel2')">
</cfform>
</body>
</html>
```

ColdFusion.Layout.hideArea

説明
ボーダー付きレイアウトの領域を非表示にします。

関数のシンタックス
```
ColdFusion.Layout.hideArea(layout, layoutArea)
```

関連項目
cflayout、cflayoutarea、ColdFusion.Layout.collapseArea、ColdFusion.Layout.expandArea、ColdFusion.Layout.showArea、『ColdFusion アプリケーションの開発』の Using layouts

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
layout	非表示にする領域が含まれているボーダー付きレイアウトの name 属性です。
layoutArea	非表示にする領域のレイアウト内の位置です。bottom、left、right、top のいずれかになります。

戻り値
この関数は値を返しません。

使用方法
この関数は、領域が既に非表示にされている場合には効果がありません。

例
次のコードでは、ユーザーがボタンをクリックすると、ボーダー付きレイアウトの左側の領域が非表示になります。

```
<cfinput name="hide2" width="100" value="Hide Area 2" type="button"
        onClick="ColdFusion.Layout.hideArea('thelayout', 'left');">
```

ColdFusion.Layout.hideTab

説明
特定のタブと、そのレイアウトの領域を非表示にします。

関数のシンタックス
```
ColdFusion.Layout.hideTab(layout, layoutArea)
```

関連項目
cflayout、cflayoutarea、ColdFusion.Layout.createTab、ColdFusion.Layout.disableTab、ColdFusion.Layout.enableTab、ColdFusion.Layout.selectTab、ColdFusion.Layout.showTab、『ColdFusion アプリケーションの開発』の Using layouts

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
layout	非表示にする領域が含まれているタブ付きレイアウトの name 属性です。
layoutArea	非表示にするタブ付きレイアウトの領域の name 属性です。

戻り値
この関数は値を返しません。

例
次の例では、2 つのタブが含まれたレイアウトを作成します。ボタンをクリックすると、2 番目のタブを表示または非表示にできます。

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
</head>

<body>
<cflayout type="tab" name="tabLayout" tabheight="300px"
    style="width:400px">
    <cflayoutarea title="First Tab" name="tab1">
        <h2>The First Tab</h2>
        Here are the contents of the first tab.
    </cflayoutarea>

    <cflayoutarea title="Second Tab" name="tab2">
        <h2>The Second Tab</h2>
        This is the content of the second tab.
    </cflayoutarea>
</cflayout>
<br />

<cfform name="layouts">
    <cfinput type="button" name="ShowTab" value="Show Tab"
        onClick="ColdFusion.Layout.showTab('tabLayout','tab2')">
    <cfinput type="button" name="ShowTab" value="Hide Tab"
        onClick="ColdFusion.Layout.hideTab('tabLayout','tab2')">
</cfform>
</body>
</html>
```

ColdFusion.Layout.selectAccordion

説明
指定されたアコーディオンレイアウトを選択し、そのパネルを表示します。

関数のシンタックス
```
ColdFusion.Layout.selectAccordion(layoutname, layoutareaname)
```

関連項目
cflayout、ColdFusion.Layout.collapseAccordion、ColdFusion.Layout.createAccordionPanel、ColdFusion.Layout.expandAccordion、ColdFusion.Layout.getAccordionLayout、ColdFusion.Layout.hideAccordion、ColdFusion.Layout.showAccordion、『ColdFusion アプリケーションの開発』の Using layouts

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
layoutname	選択する領域が含まれているアコーディオンレイアウトの name 属性です。
layoutareaname	選択するパネルの name 属性です。

戻り値
この関数は値を返しません。

使用方法
この関数は、無効にされているパネルには効果がありません。

例
次のコードでは、アコーディオンレイアウト内の 2 つのパネルをそれぞれ選択できます。

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
</head>

<body>
<cflayout type="accordion" name="mainAccordion" accordionheight="300px" style="width:400px">
    <cflayoutarea title="First Panel" name="panel1">
        <h2>The First Panel</h2>
        Here are the contents of the first panel.
    </cflayoutarea>

    <cflayoutarea title="Second Panel" name="panel2">
        <h2>The Second Panel</h2>
        This is the content of the second panel.
    </cflayoutarea>
</cflayout>

<p>
Use these links to test selecting tabs via JavaScript:<br />
<a href="" onClick="ColdFusion.Layout.selectAccordion('mainAccordion','panel1');
    return false;">Click here to select panel 1.</a><br />
<a href="" onClick="ColdFusion.Layout.selectAccordion('mainAccordion','panel2');
    return false;">Click here to select panel 2.</a><br />
</p>

</body>
</html>
```

ColdFusion.Layout.selectTab

説明
特定のタブを選択して、そのレイアウトの領域を表示します。

関数のシンタックス
```
ColdFusion.Layout.selectTab(layout, layoutArea)
```

関連項目
cflayout、cflayoutarea、ColdFusion.Layout.createTab、ColdFusion.Layout.disableTab、ColdFusion.Layout.enableTab、ColdFusion.Layout.hideTab、ColdFusion.Layout.showTab、『ColdFusion アプリケーションの開発』の Using layouts

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
layout	選択する領域が含まれているタブ付きレイアウトの name 属性です。
layoutArea	選択するタブ付きレイアウトの領域の name 属性です。

戻り値
この関数は値を返しません。

使用方法
この関数は、無効にされているタブには効果がありません。

例
次のコードでは、レイアウト内の 2 つのタブをそれぞれ選択できます。

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
</head>

<body>
<cflayout type="tab" name="mainTab" tabheight="300px" style="width:400px">
    <cflayoutarea title="First Tab" name="tab1">
        <h2>The First Tab</h2>
        Here are the contents of the first tab.
    </cflayoutarea>

    <cflayoutarea title="Second Tab" name="tab2">
        <h2>The Second Tab</h2>
        This is the content of the second tab.
    </cflayoutarea>
</cflayout>

<p>
Use these links to test selecting tabs via JavaScript:<br />
<a href="" onClick="ColdFusion.Layout.selectTab('mainTab','tab1');
    return false;">Click here to select tab 1.</a><br />
<a href="" onClick="ColdFusion.Layout.selectTab('mainTab','tab2');
    return false;">Click here to select tab 2.</a><br />
</p>

</body>
</html>
```

ColdFusion.Layout.showAccordion

説明
coldfusion.layout.hideAccordion 関数を使用して非表示にされたアコーディオンレイアウト内のパネルを表示します。

関数のシンタックス
```
ColdFusion.Layout.showAccordion(layoutname, layoutareaname)
```

関連項目

cflayout、ColdFusion.Layout.collapseAccordion、ColdFusion.Layout.createAccordionPanel、ColdFusion.Layout.expandAccordion、ColdFusion.Layout.getAccordionLayout、ColdFusion.Layout.hideAccordion、ColdFusion.Layout.selectAccordion、『ColdFusion アプリケーションの開発』の Using layouts

履歴

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
layoutname	表示するパネルが含まれているアコーディオンレイアウトの name 属性です。
layoutareaname	表示するパネルが含まれているアコーディオンレイアウト領域の name 属性です。

戻り値

この関数は値を返しません。

使用方法

この関数では、アコーディオンレイアウト領域のパネルのみが表示されます。表示領域は表示されません。非表示になっているパネルの表示領域を表示するには、この関数の後に ColdFusion.Layout.selectAccordion を呼び出します。

この関数では、ユーザーがパネルの上にある x のアイコンをクリックして閉じたパネルは表示されません。

例

ColdFusion.Layout.hideAccordion を参照してください。

ColdFusion.Layout.showArea

説明

cflayoutarea タグの inithide 属性または ColdFusion.Layout.hideArea() JavaScript 関数によって非表示にされたボーダー付きレイアウトの領域を表示します。

関数のシンタックス

```
ColdFusion.Layout.showArea(layout, layoutArea)
```

関連項目

cflayout、cflayoutarea、ColdFusion.Layout.collapseArea、ColdFusion.Layout.expandArea、ColdFusion.Layout.getTabLayout、『ColdFusion アプリケーションの開発』の Using layouts

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
layout	表示する領域が含まれているボーダー付きレイアウトの name 属性です。
layoutArea	表示する領域のレイアウト内の位置です。bottom、left、right、top のいずれかになります。

戻り値
この関数は値を返しません。

使用方法
この関数では、ユーザーがタイトルバーの上にある X のアイコンをクリックして閉じた領域は表示されません。その領域を表示するために他の領域を移動する必要がある場合は、自動的に他の領域が移動されます。

この関数は、領域が既に表示されている場合には効果がありません。

例
次のコードでは、ユーザーがボタンをクリックすると、ボーダー付きレイアウトの左側の領域が表示されます。

```
<cfinput name="show2" width="100" value="Show Area 2" type="button"
        onClick="ColdFusion.Layout.showArea('thelayout', 'left');">
```

ColdFusion.Layout.showTab

説明
cflayoutarea タグの inithide 属性または hideTab() JavaScript 関数によって非表示にされたタブを表示します。

関数のシンタックス
```
ColdFusion.Layout.showTab(layout, layoutArea)
```

関連項目
cflayout、cflayoutarea、ColdFusion.Layout.createTab、ColdFusion.Layout.disableTab、ColdFusion.Layout.enableTab、ColdFusion.Layout.hideTab、ColdFusion.Layout.selectTab、『ColdFusion アプリケーションの開発』の Using layouts

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
layout	表示するタブが含まれているタブ付きレイアウトの name 属性です。
layoutArea	表示するタブのタブ付きレイアウトの領域の name 属性です。

戻り値
この関数は値を返しません。

使用方法
この関数では、レイアウト領域のタブのみが表示されます。表示領域は表示されません。非表示になっているタブの表示領域を表示するには、この関数の後に ColdFusion.Layout.selectTab 関数を呼び出します。

この関数では、ユーザーがタブの上にある x のアイコンをクリックして閉じたタブは表示されません。

例
ColdFusion.Layout.hideTab を参照してください。

ColdFusion.Log.debug

説明
デバッグレベルのメッセージをログウィンドウに表示します。

関数のシンタックス
```
ColdFusion.Log.debug(message [, category])
```

関連項目
ColdFusion.Log.dump、ColdFusion.Log.error、ColdFusion.Log.info、『ColdFusion アプリケーションの開発』の Logging information

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
message	ログウィンドウに表示するテキストメッセージです。ログメッセージには HTML マークアップや JavaScript 変数を含めることもできます。
category	出力をフィルタ処理するためにロギングウィンドウで使用するカテゴリ識別子です。この関数では任意のカテゴリを指定できます。デフォルト値は global です。

戻り値
この関数は値を返しません。

使用方法
この関数を呼び出すページに ColdFusion AJAX ベースのコントロールがない場合は、そのページで cfajaximport タグを使用します。これにより、この関数に必要な JavaScript 定義がページ内に確実に含まれることになります。

ログウィンドウは、ページリクエストで cfdebug 形式の URL パラメータまたは cfdebug="true" が指定され、かつ ColdFusion Administrator の [デバッグとロギング]-[デバッグ出力の設定] ページで [AJAX デバッグログウィンドウの有効化] オプションが選択されている場合にのみ表示されます。

例
```
ColdFusion.Log.debug("<b>Debug argument:</b><br>" + arg.A, "Pod A");
```

ColdFusion.Log.dump

説明
cfdump に似た表現で複雑な JavaScript オブジェクトを表示できるログウィンドウにデバッグレベルのメッセージを表示します。このログウィンドウには、固有のダンプレベルはありません。

関数のシンタックス
```
ColdFusion.Log.dump(object [, category])
```

関連項目
ColdFusion.Log.debug、ColdFusion.Log.error、ColdFusion.Log.info、『ColdFusion アプリケーションの開発』の Logging information

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
object	どの変数の内容を表示するかを指定します。複雑なオブジェクトをダンプする場合は、テキストメッセージなどの追加の内容は指定できません。追加情報を提供するには、ColdFusion.Log.debug 関数も使用する必要があります。
category	出力をフィルタ処理するためにロギングウィンドウで使用するカテゴリ識別子です。この関数では任意のカテゴリを指定できます。デフォルト値は global です。

戻り値
この関数は値を返しません。

使用方法
この関数を呼び出すページに ColdFusion AJAX ベースのコントロールがない場合は、そのページで cfajaximport タグを使用します。これにより、この関数に必要な JavaScript 定義がページ内に確実に含まれることになります。

ログウィンドウは、ページリクエストで cfdebug 形式の URL パラメータまたは cfdebug="true" が指定され、かつ ColdFusion Administrator の [デバッグとロギング]-[デバッグ出力の設定] ページで [AJAX デバッグログウィンドウの有効化] オプションが選択されている場合にのみ表示されます。

例
```
ColdFusion.Log.dump(objArg, "Pod A");
```

ColdFusion.Log.error

説明
エラーレベルのメッセージをログウィンドウに表示します。

関数のシンタックス
```
ColdFusion.Log.error(message [, category])
```

関連項目

ColdFusion.Log.debug、ColdFusion.Log.dump、ColdFusion.Log.info、『ColdFusion アプリケーションの開発』の Logging information

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
message	ログウィンドウに表示するテキストメッセージです。ログメッセージには HTML マークアップや JavaScript 変数を含めることもできます。
category	出力をフィルタ処理するためにロギングウィンドウで使用するカテゴリ識別子です。この関数では任意のカテゴリを指定できます。デフォルト値は global です。

戻り値

この関数は値を返しません。

使用方法

この関数を呼び出すページに ColdFusion AJAX ベースのコントロールがない場合は、そのページで cfajaximport タグを使用します。これにより、この関数に必要な JavaScript 定義がページ内に確実に含まれることになります。

ログウィンドウは、ページリクエストで cfdebug 形式の URL パラメータまたは cfdebug="true" が指定され、かつ ColdFusion Administrator の [デバッグとロギング]-[デバッグ出力の設定] ページで [AJAX デバッグログウィンドウの有効化] オプションが選択されている場合にのみ表示されます。

例

```
ColdFusion.Log.error("<b>Invalid value:</b><br>" + arg.A, "Pod A");
```

ColdFusion.Log.info

説明

情報レベルのメッセージをログウィンドウに表示します。

関数のシンタックス

```
ColdFusion.Log.info(message [, category])
```

関連項目

ColdFusion.Log.debug、ColdFusion.Log.dump、ColdFusion.Log.error、『ColdFusion アプリケーションの開発』の Logging information

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
message	ログウィンドウに表示するテキストメッセージです。ログメッセージには HTML マークアップや JavaScript 変数を含めることもできます。
category	出力をフィルタ処理するためにロギングウィンドウで使用するカテゴリ識別子です。この関数では任意のカテゴリを指定できます。デフォルト値は global です。

戻り値
この関数は値を返しません。

使用方法
この関数を呼び出すページに ColdFusion AJAX ベースのコントロールがない場合は、そのページで cfajaximport タグを使用します。これにより、この関数に必要な JavaScript 定義がページ内に確実に含まれることになります。

ログウィンドウは、ページリクエストで cfdebug 形式の URL パラメータまたは cfdebug="true" が指定され、かつ ColdFusion Administrator の [デバッグとロギング]-[デバッグ出力の設定] ページで [AJAX デバッグログウィンドウの有効化] オプションが選択されている場合にのみ表示されます。

例
```
ColdFusion.Log.info("<b>arg.A is:</b><br>" + arg.A, "Window Z");
```

ColdFusion.Map.addEvent

説明
カスタム JavaScript 関数を実行して、マップ内のイベント処理を有効にします。

関数のシンタックス
```
ColdFusion.Map.addEvent(name, event, listener, scopeObject)
```

関連項目
ColdFusion.Map.getLatitudeLongitude、ColdFusion.Map.getMapObject、ColdFusion.Map.setCenter、ColdFusion.Map.setZoomlevel

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
name	マップの名前です。
event	処理するイベントです (click、dblclick、singleRightClick、mapTypeChange など)。イベントの詳細については、『Google Maps API リファレンス』の「イベント」セクションを参照してください。
listener	イベントが発生したときに呼び出される関数です。
scopeObject	this スコープで設定されている JavaScript オブジェクトです。

使用方法
この関数は値を返しません。

例
```
<h3>This is an example of the Map.addmarker function. Click the HTML button labeled "Add marker" to set the
marker to the specified Address.</h3>
    <script>
    var markerObj={
    address: '201 S. Division St. Suite 500 Ann Arbor, MI 48104'
    };
    function addmarker(){
    ColdFusion.Map.addMarker('mapID', markerObj);
    }
    </script>
    <h3>MAP 1</h3>
    <cfform name="map01">
    <cfinput type="button" value="Add marker" name="markerbutton"
    onclick="javascript:addmarker();">
    </cfform>
    <cfmap name="mapID"
    centerlatitude=42.261
    centerlongitude=-87.717
    displayscale=true
    doubleclickzoom="true"
    overview=true
    scrollwheelzoom=true
    tips="My Map" zoomlevel="4">
    </cfmap>
```

ColdFusion.Map.addMarker

説明
マップにマーカーを追加します。

関数のシンタックス
```
ColdFusion.Map.addMarker(name, markerObj)
```

関連項目
ColdFusion.Map.getLatitudeLongitude、ColdFusion.Map.getMapObject、ColdFusion.Map.setCenter、ColdFusion.Map.setZoomlevel

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
name	cfmap タグの name 属性の値を指定します。
markerObj	指定したアドレスのマーカーオブジェクトを指定します。関連するプロパティは、latitude、longitude、address、title、markercolor、markericon、address、markerwindowcontent、および showmarkerwindow です。

戻り値
この関数は値を返しません。

例
```
<h3>This is an example of the Map.addmarker function. Click the HTML button labeled "Add marker" to set the
marker to the specified Address.</h3>
    <script>
    var markerObj={
    address: '201 S. Division St. Suite 500 Ann Arbor, MI 48104'
    };
    function addmarker(){
    ColdFusion.Map.addMarker('mapID', markerObj);
    }
    </script>
    <h3>MAP 1</h3>
    <cfform name="map01">
    <cfinput type="button" value="Add marker" name="markerbutton"
    onclick="javascript:addmarker();">
    </cfform>
    <cfmap name="mapID"
    centerlatitude=42.261
    centerlongitude=-87.717
    displayscale=true
    doubleclickzoom="true"
    overview=true
    scrollwheelzoom=true
    tips="My Map" zoomlevel="4">
    </cfmap>
```

ColdFusion.Map.getLatitudeLongitude

説明
特定の住所の緯度 / 経度座標を取得します。

関数のシンタックス
```
ColdFusion.Map.getLatitudeLongitude("address", "callBack")
```

関連項目
ColdFusion.Map.addMarker、ColdFusion.Map.getMapObject、ColdFusion.Map.setCenter、ColdFusion.Map.setZoomlevel

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
address	cfmap タグの address 属性の値を指定します。
callBack	緯度および経度の値の特定が成功した後に実行されるコールバック関数です。

戻り値
この関数は、指定された住所の緯度および経度の値を取得するコールバック関数を返します。

例

```
<h3>This is an example of the Map.getLatitudeLongitude function. Click the HTML button labeled "GetLatitude-
Longitude" to get the latitude and longitude of Ann Arbor,MI.</h3>
    <script>
    function getLongitudeLatitude()
    {
    ColdFusion.Map.getLatitudeLongitude('201 S. Division St. Suite 500 Ann Arbor, MI 48104',
callbackHandler);
    }
    function callbackHandler(result)
    {
    alert("The latitude-longitude of Ann Arbor,MI is: "+result);
    }
    </script>
    <h3>MAP 1</h3>
    <cfform name="map01">
    <cfinput type="button" value="GetLatitude-Longitude" name="buttn03"
    onclick="javascript:getLongitudeLatitude()">
    </cfform>
    <cfmap name="mapID" centerlatitude= 42
    centerlongitude=-87
    doubleclickzoom="true"
    overview=true
    scrollwheelzoom=true tips="My Map" zoomlevel="4">
    </cfmap>
```

ColdFusion.Map.getMapObject

説明

Google マップコンポーネントを取得します。サポートされる Google Map API を使用してマップを操作できます。

関数のシンタックス

```
ColdFusion.Map.getMapObject("name")
```

関連項目

ColdFusion.Map.addMarker、ColdFusion.Map.getLatitudeLongitude、ColdFusion.Map.setCenter、ColdFusion.Map.setZoomlevel

履歴

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
name	cfmap タグの name 属性の値を指定します。

戻り値

この関数は Google マップコンポーネントを返します。マップタイプには、map、satellite、または hybrid を指定できます。

例

```
<h3>This is an example of the Map.getmapObject function. Click the HTML button labeled "GetMap" to get the
map object and set the center to Palo Alto.</h3>
    <script>
    function getMapObject()
    {
    var mapObj = ColdFusion.Map.getMapObject('mapID');
    mapObj.setCenter(new GLatLng(37.4419, -122.1419), 13);
    }
    </script>
    <h3>MAP 1</h3>
    <cfform name="map01">
    <cfinput type="button" value="getMapObject and setCenter" name="htmlbutton"
    onclick="javascript:getMapObject()">
    </cfform>
    <cfmap name="mapID"
    centerAddress='201 S. Division St. Suite 500 Ann Arbor, MI 48104'
    displayscale=true
    doubleclickzoom="true"
    overview=true
    scrollwheelzoom=true
    tips="My Map"
    zoomlevel="4">
    </cfmap>
```

ColdFusion.Map.hide

説明

マップが表示されている場合は非表示にします。

関数のシンタックス

ColdFusion.Map.hide(Id)

パラメータ

- Id: マップの名前です。

例

ColdFusion.Map.show の例を参照してください。

ColdFusion.Map.refresh

説明

マップをリロードします。

関数のシンタックス

ColdFusion.Map.refresh (Id)

パラメータ

- Id: マップの名前です。

使用方法

マップが、非表示の折り畳み可能な spry パネルまたは div の中に埋め込まれている場合 (つまり、マップコンテナは表示されているが、実際のマップは非表示である場合)、この関数を使用して、マップを強制的に表示できます。

例

```
<script type="text/javascript" src="/CFIDE/scripts/ajax/spry/includes_minified/SpryCollapsiblePanel.js"
></script>
<link type="text/css" href="/CFIDE/scripts/ajax/spry/widgets/collapsiblepanel/SpryCollapsiblePanel.css"
rel="stylesheet">
<div id="cp" class="CollapsiblePanel" style="width:500px;">
      <div class="CollapsiblePanelTab" tabindex="0">SHOW MAP</div>
    <div class="CollapsiblePanelContent">
    <cfmap
            width="500"
            height="200"
            zoomlevel="12"
            name="mainMap"
            markercolor="333444"
            showscale="false"
            typecontrol="none"
            showcentermarker="true"
            centeraddress="The Key Learning centre, Oxford, UK"
            >
      </cfmap>
      </div>
</div>
<script type="text/javascript">
      var myTabClick = function()
      {
            !cpanel.isOpen() ? cpanel.open() : cpanel.close();
            cpanel.focus();
            ColdFusion.Map.refresh('mainMap');
      }
      var cpanel = new Spry.Widget.CollapsiblePanel("cp", {contentIsOpen:false});
      cpanel.onTabClick = myTabClick;
</script>
```

ColdFusion.Map.setCenter

説明

指定した住所をマップの中心に設定します。

関数のシンタックス

```
ColdFusion.Map.setCenter("name", centerConfigObject)
```

関連項目

ColdFusion.Map.addMarker、ColdFusion.Map.getLatitudeLongitude、ColdFusion.Map.getMapObject、ColdFusion.Map.setZoomlevel

履歴

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
name	cfmap タグの name 属性の値を指定します。
centerConfigObject	中央アドレスオブジェクトです。 このオブジェクトの値には、経度および緯度の値または住所プロパティのいずれかを指定できます。

戻り値
この関数は値を返しません。

例 1
```
<h3>This is an example of the Map.setcenter function using latitude-longitude and address values</h3>
    <script>
    var centerLongLat={
    latitude: 71.094224,
    longitude: 42.339641
    };
    var center={
    address: '345 Park Avenue, san jose, CA 95110-2704, USA'
    };
    function setcenter()
    {
    ColdFusion.Map.setCenter('mapID', centerLongLat);
    }
    function setcenterlatlong()
    {
    ColdFusion.Map.setCenter('mapID', center);
    }
    </script>
    <h3>MAP 1</h3>
    <cfform name="mapID">
    Click this button to set the center using Latitude and Longitude.
    <cfinput type="button" value="setCenter using lattitude-longitude"
    name="buttn01" onclick="javascript:setcenterlatlong();">
    <br>Click this button to set the center using Address.
    <cfinput type="button" value="setCenter using Address"
    name="buttn01" onclick="javascript:setcenter();">
    </cfform>
    <cfmap name="mapID" centerlatitude=71.094224
    centerlongitude=42.339641
    displayscale=true
    doubleclickzoom="true"
    overview=true
    scrollwheelzoom=true
    tips="My Map"
    zoomlevel="4">
    </cfmap>
```

ColdFusion.Map.setZoomlevel

説明
マップのズームレベルを新しい値に設定します。

関数のシンタックス

```
ColdFusion.Map.setZoomlevel("name", zoomLevelValue)
```

関連項目

ColdFusion.Map.addMarker、ColdFusion.Map.getLatitudeLongitude、ColdFusion.Map.getMapObject、ColdFusion.Map.setCenter

履歴

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
name	cfmap タグの name 属性の値を指定します。
zoomLevelValue	cfmap タグの zoomlevel 属性の整数値です。

戻り値

この関数は値を返しません。

例

```
<h3>This is an example of the Map.setzoomlevel function. Click the Set Zoom Level button to set the zoom
level to 6.</h3>
    <script>
    function setZoom(zoomlevel)
    {
    ColdFusion.Map.setZoomLevel('mapID', zoomlevel);
    }
    </script>
    <h3>MAP 1</h3>
    <cfform name="map01">
    <cfinput type="button" value="Set Zoom Level" name="buttn04"
    onclick="javascript:setZoom(6)">
    </cfform>
    <cfmap name="mapID"
    centerlatitude=42.094224
    centerlongitude=72.339641
    displayscale=true
    doubleclickzoom="true"
    overview=true
    scrollwheelzoom=true
    tips="My Map"
    zoomlevel="4" >
    </cfmap>
```

ColdFusion.Map.show

説明

マップが非表示になっている場合は表示します。

関数のシンタックス

ColdFusion.Map.show(Id)

パラメータ

- Id: マップの名前です。

例

```
<script>
    function showMap(mapId)
      {
                  ColdFusion.Map.show(mapId);
    }

    function hideMap(mapId)
      {
                  ColdFusion.Map.hide(mapId);
    }
</script>
<a href="##" id="a1" onclick="return showMap('mainMap')">Show Map</a> | <a href="##" id="a1" onclick="return
hideMap('mainMap')">Hide Map</a>
<cfmap
       zoomlevel = "12"
    name = "mainMap"
    showcentermarker= "true"
    centeraddress = "The Key Learning centre, Oxford, UK"
      title="Venue Address"
      hideborder=false
      collapsible=true
      initShow=false/>
```

ColdFusion.MediaPlayer.getPlayer

説明

Strobe メディアプレーヤーの JavaScript API. を呼び出すために使用するプレーヤーオブジェクトを返します。

関数のシンタックス

```
ColdFusion.MediaPlayer.getPlayer("Name")
```

関連項目

ColdFusion.Mediaplayer.resize、ColdFusion.Mediaplayer.setMute、ColdFusion.Mediaplayer.setSource、ColdFusion.Mediaplayer.setVolume、ColdFusion.Mediaplayer.startPlay、ColdFusion.Mediaplayer.getType、ColdFusion.Mediaplayer.stopPlay

履歴

ColdFusion 10: この関数が追加されました。

パラメータ

パラメータ	説明
Name	cfmediaplayer タグの name 属性の値を指定します。

例

Dynamic streaming の例を参照してください。

ColdFusion.Mediaplayer.getType

説明
現在の再生タイプ（Flash Player と HTML プレーヤーのどちらであるか）を取得するために使用します。

関数のシンタックス
```
ColdFusion.MediaPlayer.getType("name")
```

関連項目
ColdFusion.Mediaplayer.resize、ColdFusion.Mediaplayer.setMute、ColdFusion.Mediaplayer.setSource、
ColdFusion.Mediaplayer.setVolume、ColdFusion.Mediaplayer.startPlay、ColdFusion.Mediaplayer.stopPlay

履歴
ColdFusion 10: この関数が追加されました。

パラメータ

パラメータ	説明
name	cfmediaplayer タグの name 属性の値を指定します。

戻り値
この関数は、再生タイプを Flash または HTML 5 で返します。

例
```
<!DOCTYPE html>
<html>
<head>
<script type="text/javascript">
function getPlayback(player)
{
    alert(ColdFusion.MediaPlayer.getType("player"));
}
</script>
</head>
<body style="background:#">
<input name = "Source" value="Type" type="button" onClick="getPlayback('player')">
<div>
<cfmediaplayer
    fullScreenControl="yes"
    name="player"
    width=800
    height=500
    >
    <source src="/videos/cathy2_HD.mp4" type="video/mp4" />
    <source src="/videos/gizmo.ogv" type="video/ogg" />
</cfmediaplayer>
</div>
</body>
</html>
```

ColdFusion.Mediaplayer.logError

説明
再生中にエラーが発生した場合に、メディアプレーヤーにカスタムエラーメッセージを表示できます。

関数のシンタックス
```
ColdFusion.MediaPlayer.logError("Name", "Error")
```

関連項目
ColdFusion.Mediaplayer.resize、ColdFusion.Mediaplayer.setMute、ColdFusion.Mediaplayer.setSource、ColdFusion.Mediaplayer.setVolume、ColdFusion.Mediaplayer.startPlay、ColdFusion.Mediaplayer.getType、ColdFusion.Mediaplayer.stopPlay

履歴
ColdFusion 10: この関数が追加されました。

パラメータ

パラメータ	説明
Name	cfmediaplayer タグの name 属性の値を指定します。
Error	表示するカスタムエラーメッセージです。

例
1423 ページの「ColdFusion.Mediaplayer.getType」の例を参照してください。

ColdFusion.Mediaplayer.resize

説明
メディアプレーヤの現在のサイズを変更します。

関数のシンタックス
```
ColdFusion.Mediaplayer.resize("name", "height", "width")
```

関連項目
ColdFusion.Mediaplayer.setMute、ColdFusion.Mediaplayer.setSource、ColdFusion.Mediaplayer.setVolume、ColdFusion.Mediaplayer.startPlay、ColdFusion.Mediaplayer.stopPlay

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
name	cfmediaplayer タグの name 属性の値を指定します。
height	メディアプレーヤーに対して設定する高さ (単位 : ピクセル) です。高さはデフォルトで 360 ピクセルです。
width	メディアプレーヤーに対して設定する幅 (単位 : ピクセル) です。幅はデフォルトで 480 ピクセルです。

戻り値
この関数は値を返しません。

例
この例では、FLV ファイルは、ColdFusion サーバーが使用する Web ルートに保存されています。FLV ファイル (video.flv) の保存場所は、"<Web ルートディレクトリ >¥xyz¥" です。

```
 <h3>This is an example of the Mediaplayer.resize function. Clicking the Resize Mediaplayer button resizes
the media player component.</h3>
    <script language="JavaScript" type="text/javascript">
    function onResize() {
    ColdFusion.Mediaplayer.resize('Myvideo', 500, 800);
    };
    </script>
    <br>
    <form>
    <input type="button" name="resize" value="Resize Mediaplayer" onClick="onResize()">
    </form>
    <br/>
    <cfmediaplayer name="Myvideo" source="\xyz\video.flv"
    width=500 height=400 align="middle"
    quality="high" fullscreencontrol="true"/>
```

ColdFusion.Mediaplayer.setTitle

説明
メディアプレーヤーのタイトルを左上隅に割り当てます。

関数のシンタックス
```
ColdFusion.Mediaplayer.setTile("name", "title")
```

関連項目
ColdFusion.Mediaplayer.setMute、ColdFusion.Mediaplayer.setSource、ColdFusion.Mediaplayer.setVolume、ColdFusion.Mediaplayer.startPlay、ColdFusion.Mediaplayer.stopPlay

履歴
ColdFusion 10: この関数が追加されました。

パラメータ

パラメータ	説明
name	cfmediaplayer タグの name 属性の値を指定します。
title	メディアプレーヤーの左上隅に表示するタイトルを指定します。

戻り値
この関数は値を返しません。

例
```
 <h3>This is an example of the Mediaplayer.setTitle function. Clicking the Resize Mediaplayer button resizes
the media player component.</h3>
    <script language="JavaScript" type="text/javascript">
    function assignTitle() {
    ColdFusion.Mediaplayer.setTitle('Myvideo', 'Video');
    };
    </script>
    <br>
    <form>
    <input type="button" name="resize" value="Resize Mediaplayer" onClick="assignTitle()">
    </form>
    <br/>
    <cfmediaplayer name="Myvideo" source="\xyz\video.flv"
    width=500 height=400 align="middle"
    quality="high" fullscreencontrol="true"/>
```

ColdFusion.Mediaplayer.setMute

説明
メディアプレーヤのサウンドをミュートまたはミュート解除します。

関数のシンタックス
```
ColdFusion.Mediaplayer.setMute("name", mute)
```

関連項目
ColdFusion.Mediaplayer.resize、ColdFusion.Mediaplayer.setSource、ColdFusion.Mediaplayer.setVolume、ColdFusion.Mediaplayer.startPlay、ColdFusion.Mediaplayer.stopPlay

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
name	cfmediaplayer タグの name 属性の値を指定します。
mute	サウンドをミュートにするか (true)、またはミュートをオフにするか (false) を指定するブール値です。

戻り値
この関数は値を返しません。

例

この例では、FLV ファイルは、ColdFusion サーバーが使用する Web ルートに保存されています。FLV ファイル (video.flv) の保存場所は、"<Web ルートディレクトリ >¥xyz¥" です。

```
 <h3>This is an example of the Mediaplayer.setmute function. Clicking the Mute button mutes the media
player.</h3>
    <script language="JavaScript" type="text/javascript">
    function onMute() {
    ColdFusion.Mediaplayer.setMute('Myvideo', true);
    };
    </script>
    <br>
    <form>
    <input type="button" name="mute" value="Mute" onClick="onMute()">
    </form>
    <cfmediaplayer name="Myvideo" source="/xyz/video.flv"
    width=500 height=400 align="middle"
    quality="high" fullscreencontrol="true"/>
```

ColdFusion.Mediaplayer.setSource

説明

FLV ファイルの URL を設定します。URL では、ColdFusion サーバーまたはその他のサーバー上の場所を指定できます。

関数のシンタックス

```
ColdFusion.Mediaplayer.setSource("name", newURL)
```

関連項目

ColdFusion.Mediaplayer.setMute、ColdFusion.Mediaplayer.setMute、ColdFusion.Mediaplayer.setVolume、ColdFusion.Mediaplayer.startPlay、ColdFusion.Mediaplayer.stopPlay

履歴

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
name	cfmediaplayer タグの name 属性の値を指定します。
newURL	FLV ファイルの URL です。このパラメータでは、現在のページからの相対 URL を指定できます。FLV ファイルは、ColdFusion サーバーまたはその他のストリーミングサーバー上に格納できます。

戻り値

この関数は値を返しません。

例

この例では、FLV ファイルのソースは、video.flv から newvideo.flv に変更されています。これにより、メディアプレーヤーでは、newvideo.flv ファイルを再生するようになります。

```
 <h3>This is an example of the Mediaplayer.setsource function. Clicking the Set Source button changes the
video playing in the media player.</h3>
    <script language="JavaScript" type="text/javascript">
    function setSource()
    {
    ColdFusion.Mediaplayer.setSource('Myvideo',"/xyz/newvideo.flv");
    };
    </script>
    <br>
    <form>
    <input type="button" name="setSource" value="Set Source"
    onClick="setSource()">
    </form>
    <cfmediaplayer name="Myvideo"
    source="/xyz/video.flv"
    width=500 height=400 align="middle"
    quality="high" fullscreencontrol="true"/>
```

ColdFusion.Mediaplayer.setVolume

説明
メディアプレーヤのサウンドのボリュームを設定します。

関数のシンタックス
```
ColdFusion.Mediaplayer.setVolume("name", "volume")
```

関連項目
ColdFusion.Mediaplayer.resize、ColdFusion.Mediaplayer.setMute、ColdFusion.Mediaplayer.setSource、ColdFusion.Mediaplayer.startPlay、ColdFusion.Mediaplayer.stopPlay

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
name	cfmediaplayer タグの name 属性の値を指定します。
volume	ボリュームの値です。この値には、0.0 から 1.0 の範囲を指定できます。

戻り値
この関数は値を返しません。

例
この例では、FLV ファイルは、ColdFusion サーバーが使用する Web ルートに保存されています。FLV ファイル (video.flv) の保存場所は、"<Web ルートディレクトリ >¥xyz¥" です。

```
<h3>This is an example of the Mediaplayer.setvolume function. Clicking the Set Volume button increases the
volume of the media player.</h3>
    <script language="JavaScript" type="text/javascript">
    function setVol() {
        ColdFusion.Mediaplayer.setVolume('Myvideo', 1.0);
    };
    </script>
    <br>
    <form>
    <input type="button" name="setVolume" value="Set Volume" onClick="setVol()">
    </form>
    <cfmediaplayer name="Myvideo" source="\xyz\video.flv"
    width=500 height=400 align="middle"
    quality="high" fullscreencontrol="true"/>
```

ColdFusion.Mediaplayer.startPlay

説明
FLV ファイルを再生します。

関数のシンタックス
```
ColdFusion.Mediaplayer.startPlay("name")
```

関連項目
ColdFusion.Mediaplayer.resize、ColdFusion.Mediaplayer.setMute、ColdFusion.Mediaplayer.setSource、ColdFusion.Mediaplayer.setVolume、ColdFusion.Mediaplayer.stopPlay

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
name	cfmediaplayer タグの name 属性の値を指定します。

戻り値
この関数は値を返しません。

例
この例では、FLV ファイルは、ColdFusion サーバーが使用する Web ルートに保存されています。FLV ファイル (video.flv) の保存場所は、"<Web ルートディレクトリ >¥xyz¥" です。

```
<h3>This is an example of the Mediaplayer.startplay function. Clicking the Start Play button plays the
video.</h3>
    <script language="JavaScript" type="text/javascript">
    function onStart() {
    ColdFusion.Mediaplayer.startPlay('Myvideo');
    };
    </script>
    <br>
    <form>
    <input type="button" name="startplay" value="Start Play" onClick="onStart()">
    </form>
    <cfmediaplayer name="Myvideo" source="/xyz/video.flv"
    width=500 height=400 align="middle"
    quality="high" fullscreencontrol="true"/>
```

ColdFusion.Mediaplayer.stopPlay

説明
FLV ファイルの再生を停止します。

関数のシンタックス
```
ColdFusion.Mediaplayer.stopPlay("name")
```

関連項目
ColdFusion.Mediaplayer.resize、ColdFusion.Mediaplayer.setMute、ColdFusion.Mediaplayer.setSource、ColdFusion.Mediaplayer.setVolume、ColdFusion.Mediaplayer.startPlay

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
name	cfmediaplayer タグの name 属性の値を指定します。

戻り値
この関数は値を返しません。

例
この例では、FLV ファイルは、ColdFusion サーバーが使用する Web ルートに保存されています。FLV ファイル (video.flv) の保存場所は、"<Web ルートディレクトリ >¥xyz¥" です。

```
<h3>This is an example of the Mediaplayer.stopplay function. Clicking the Stop Play button stops playing
the video.</h3>
    <script language="JavaScript" type="text/javascript">
    function onStop()
    {
    ColdFusion.Media player.stopPlay('Myvideo' );
    };
    </script>
    <br>
    <form>
    <input type="button" name="stopPlay" value="Stop Play" onClick="onStop()">
    </form>
    <cfmediaplayer name="Myvideo" source="/xyz/video.flv"
    width=500 height=400 align="middle"
    quality="high" fullscreencontrol="true"/>
```

ColdFusion.MessageBox.create

説明

ColdFusion メッセージボックスを作成します。この関数は cfmessagebox タグに相当します。

関数のシンタックス

`ColdFusion.MessageBox.create(`**`name, type, title, message, callbackhandler [, configuration]`**`)`

関連項目

ColdFusion.MessageBox.isMessageBoxDefined、ColdFusion.MessageBox.getMessageBoxObject、ColdFusion.MessageBox.update、ColdFusion.MessageBox.updateMessage、ColdFusion.MessageBox.updateTitle、ColdFusion.MessageBox.show

履歴

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
name	メッセージボックスの名前です。 この属性は、メッセージボックスを表示および更新するために必要となります。メッセージボックス名はページ上で一意である必要があります。
type	コントロールタイプです。次のいずれかになります。 • alert - OK ボタンが 1 つ表示されるメッセージです。 • confirm - 2 つのボタン (YES と NO) または 3 つのボタン (YES、NO、CANCEL) が表示されるメッセージボックスです。 • prompt - 単一行または複数行のテキスト入力領域、および OK と CANCEL のボタンが表示されるメッセージです。
title	メッセージボックスのタイトルバーに表示するテキストです。
message	メッセージボックス内部に表示されるテキストです。
callbackhandler	ユーザーがいずれかのボタンをクリックしたときにコントロールによって呼び出される関数です。
configuration	メッセージボックスの設定パラメータが含まれるオブジェクトです。詳細については、「使用方法」を参照してください。

戻り値
この関数は値を返しません。

使用方法
この関数は cfmessagebox タグに相当します。

この関数を呼び出すページで cfmessagebox タグも使用していない場合は、そのページで cfajaximport タグを指定し、tags 属性に cfmessagebox を指定する必要があります。これにより、そのページにはメッセージボックスを作成するために必要な JavaScript が含まれます。たとえば、他の ColdFusion AJAX 機能を使用するために JavaScript をインポートする必要がない場合は、次のように記述します。

```
<cfajaximport tags="cfmessagebox">
```

configuration パラメータはメッセージボックスの特性を定義します。このパラメータでは、次のエントリの一部またはすべてを指定できます。

エントリ	デフォルト	説明
bodystyle		メッセージボックス本体の CSS スタイル指定です。一般に、この属性は色とフォントスタイルを設定する場合に使用します。
buttontype	yesno	コントロールのタイプ - confirm に適用されます。 メッセージボックスに表示するボタンは次のとおりです。 • yesno: [はい] と [いいえ] の 2 つのボタンを表示します。 • yesnocancel: [はい]、[いいえ]、および [キャンセル] のボタンを表示します。
icon		次の CSS クラスを指定します。 • error: エラーアイコンを示します。エラーメッセージを表示する場合にこのアイコンを使用できます。 • info: 情報アイコンを示します。情報を表示する場合にこのアイコンを使用できます。 • question: 質問アイコンを示します。ユーザーレスポンスを要求する確認メッセージボックスでこのアイコンを使用できます。 • warning: 警告アイコンを示します。警告メッセージを表示する場合にこのアイコンを使用できます。
labelcancel	Cancel	prompt タイプのメッセージボックスのキャンセルボタンに表示されるテキストです。
labelok	OK	アラートボタンと、prompt タイプのメッセージボックスの OK ボタンに表示されるテキストです。
labelno	No	confirm タイプのメッセージボックスの NO のボタンに表示されるテキストです。
labelyes	Yes	confirm タイプのメッセージボックスの YES のボタンに表示されるテキストです。
modal	yes	メッセージボックスがモーダルウィンドウかどうかを指定するブール値です。 • yes • no
multiline	false	prompt タイプのメッセージボックスでのみ有効です。prompt タイプのメッセージボックスの入力テキストボックスが単一行であるか、または複数行であるかを指定するブール値です。

エントリ	デフォルト	説明
width		メッセージボックスの幅です (単位 : ピクセル)。
x		メッセージボックスの左上隅の x (水平) 座標です。 y 属性を設定しない場合、この属性は無視されます。
y		メッセージボックスの左上隅の y (垂直) 座標です。 x 属性を設定しない場合、この属性は無視されます。

注意 : configuration オブジェクトのエントリ名はすべて小文字にする必要があります。

例

次の CFML アプリケーションでは、confirmation タイプのメッセージボックスが作成されます。

```
<cfajaximport tags="cfmessagebox">

<cfform name="test">
    <cfinput type="button" name="x" value="Create Message Box"
        onClick="ColdFusion.MessageBox.create('Messagebox1', 'confirm','Confirm',
        'Do you want to save the file?',
        onfinish, {width:200, modal:false})">
</cfform>
    <script language="JavaScript" type="text/javascript">
    function onfinish()
    {
    alert('Button clicked');
    };
    </script>
```

ColdFusion.MessageBox.show

説明

ColdFusion メッセージボックスを表示する場合に使用されます。

関数のシンタックス

ColdFusion.MessageBox.show(***name***)

関連項目

ColdFusion.MessageBox.isMessageBoxDefined、ColdFusion.MessageBox.getMessageBoxObject、
ColdFusion.MessageBox.update、ColdFusion.MessageBox.updateMessage、ColdFusion.MessageBox.updateTitle、
ColdFusion.MessageBox.create

履歴

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
name	表示するメッセージボックスの名前です。

戻り値
この関数は値を返しません。

使用方法
メッセージボックスを作成するには、cfmessagebox タグまたは JavaScript 関数 ColdFusion.MessageBox.create を使用します。ただし、メッセージボックスを表示するには、この関数を使用する必要があります。

例
```
<cfajaximport tags="cfmessagebox,cfform">
<cfform name="ajax">
    <br><input type="button" name="showMessageBox" value="Show Message Box"
onClick="ColdFusion.MessageBox.create('mb','Alert','ALERT','Sample
Alert!');ColdFusion.MessageBox.show('mb');">
</cfform>
```

ColdFusion.MessageBox.getMessageBoxObject

説明
指定された HTML の cfmessagebox コントロールに対して、基盤となる Ext JS - JavaScript Library オブジェクトを取得します。

関数のシンタックス
```
ColdFusion.MessageBox.getMessageBoxObject(name)
```

関連項目
ColdFusion.MessageBox.create、ColdFusion.MessageBox.isMessageBoxDefined、ColdFusion.MessageBox.update、ColdFusion.MessageBox.updateMessage、ColdFusion.MessageBox.updateTitle

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
name	メッセージボックスオブジェクトの名前です。

戻り値
JavaScript オブジェクト。

使用方法
この関数は、定義されたプロパティがすべて含まれる JavaScript オブジェクトを取得するときに使用します。

ColdFusion.MessageBox.isMessageBoxDefined

説明
メッセージボックスが定義されているかどうかをチェックします。

関数のシンタックス
```
ColdFusion.MessageBox.isMessageBoxDefined(name)
```

関連項目
ColdFusion.MessageBox.create、ColdFusion.MessageBox.getMessageBoxObject、ColdFusion.MessageBox.update、ColdFusion.MessageBox.updateMessage、ColdFusion.MessageBox.updateTitle

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
name	メッセージボックスオブジェクトの名前です。

戻り値
ブール値 (つまり、true または false)

使用方法
この関数は、特定の名前に対してメッセージボックスが定義されているかどうかをチェックする場合に使用します。

ColdFusion.MessageBox.update

説明
ColdFusion メッセージボックスのプロパティを更新します。この JavaScript 関数を使用すると、name および type を除くすべてのメッセージボックスプロパティを更新できます。

関数のシンタックス
```
ColdFusion.MessageBox.update(name, configuration)
```

関連項目
ColdFusion.MessageBox.create、ColdFusion.MessageBox.getMessageBoxObject、ColdFusion.MessageBox.isMessageBoxDefined、ColdFusion.MessageBox.updateMessage、ColdFusion.MessageBox.updateTitle

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
name	メッセージボックスの名前です。 この属性は、メッセージボックスを表示および更新するために必要となります。メッセージボックス名はページ上で一意である必要があります。
configuration	メッセージボックスの設定パラメータが含まれるオブジェクトです。詳細については、「使用方法」を参照してください。

戻り値
この関数は値を返しません。

使用方法

この関数は、一連のメッセージボックスプロパティを更新する場合に使用します。たとえば、**configuration** パラメータを使用して、メッセージボックスの幅、メッセージ、およびタイトルを更新できます。

configuration パラメータはメッセージボックスの特性を定義します。このパラメータでは、次のエントリの一部またはすべてを指定できます。

エントリ	説明
bodystyle	メッセージボックス本体の CSS スタイル指定です。一般に、この属性は色とフォントスタイルを設定する場合に使用します。
buttontype	コントロールのタイプ - confirm に適用されます。 メッセージボックスに表示するボタンは次のとおりです。 • yesno: [はい] と [いいえ] の 2 つのボタンを表示します。 • yesnocancel: [はい]、[いいえ]、および [キャンセル] のボタンを表示します。
callbackhandler	ユーザーがいずれかのボタンをクリックしたときにコントロールによって呼び出される関数です。詳細については、「使用方法」を参照してください。
icon	次の CSS クラスを指定します。 • error: エラーアイコンを示します。エラーメッセージを表示する場合にこのアイコンを使用できます。 • info: 情報アイコンを示します。情報を表示する場合にこのアイコンを使用できます。 • question: 質問アイコンを示します。ユーザーレスポンスを要求する確認メッセージボックスでこのアイコンを使用できます。 • warning: 警告アイコンを示します。警告メッセージを表示する場合にこのアイコンを使用できます。
labelcancel	prompt タイプのメッセージボックスのキャンセルボタンに表示されるテキストです。
labelok	アラートボタンと、prompt タイプのメッセージボックスの OK ボタンに表示されるテキストです。
labelno	confirm タイプのメッセージボックスの NO のボタンに表示されるテキストです。
labelyes	confirm タイプのメッセージボックスの YES のボタンに表示されるテキストです。
modal	メッセージボックスがモーダルウィンドウかどうかを指定するブール値です。 • yes • no
message	メッセージボックス内部に表示されるテキストです。
multiline	prompt タイプのメッセージボックスでのみ有効です。prompt タイプのメッセージボックスの入力テキストボックスが単一行であるか、または複数行であるかを指定するブール値です。
title	メッセージボックスのタイトルです。タイトルを指定しない場合、ColdFusion では、コントロールのタイプ値がデフォルトタイトルとして割り当てられます。
width	メッセージボックスの幅です (単位 : ピクセル)。
x	メッセージボックスの左上隅の x (水平) 座標です。 y 属性を設定しない場合、この属性は無視されます。
y	メッセージボックスの左上隅の y (垂直) 座標です。 x 属性を設定しない場合、この属性は無視されます。

注意：configuration オブジェクトのエントリ名はすべて小文字にする必要があります。

例
次の CFML アプリケーションでは、メッセージボックスが更新されます。

```
<cfajaximport tags="cfmessagebox">

<cfform name="test">
    <cfinput type="button" name="x" value="Update Message Box"
        onClick="ColdFusion.MessageBox.update('Messagebox1',
        {width:400, modal:false, x:200, y:300, labelyes:'yes'})">
</cfform>
```

ColdFusion.MessageBox.updateMessage

説明
ColdFusion メッセージボックスコンポーネントの message プロパティを更新します。

関数のシンタックス
```
ColdFusion.MessageBox.updateMessage(name, newmessage)
```

関連項目
ColdFusion.MessageBox.create、ColdFusion.MessageBox.getMessageBoxObject、ColdFusion.MessageBox.isMessageBoxDefined、ColdFusion.MessageBox.update、ColdFusion.MessageBox.updateTitle

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
name	メッセージボックスオブジェクトの名前です。
newmessage	既存のメッセージを上書きします。

戻り値
この関数は値を返しません。

使用方法
この関数は、メッセージボックスの message プロパティを更新または変更する場合に使用します。

ColdFusion.MessageBox.updateTitle

説明
ColdFusion メッセージボックスコンポーネントのプロパティを更新します。

関数のシンタックス
```
ColdFusion.MessageBox.updateTitle(name, newtitle)
```

関連項目
ColdFusion.MessageBox.create、ColdFusion.MessageBox.getMessageBoxObject、ColdFusion.MessageBox.isMessageBoxDefined、ColdFusion.MessageBox.update、ColdFusion.MessageBox.updateMessage

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
name	メッセージボックスオブジェクトの名前です。
newmessage	既存のタイトルを上書きします。

戻り値
この関数は値を返しません。

使用方法
この関数は、メッセージボックスの title プロパティを更新または変更する場合に使用します。

ColdFusion.navigate

説明
リンクターゲットの出力を AJAX の cfdiv、cflayoutarea、cfpod、または cfwindow コンテナ内に表示します。この関数が埋め込まれたリンクをブラウザ上でクリックした場合、現在のページは置き換えらません。代わりに、container 属性で指定されたコントロールにデータが挿入されます。

関数のシンタックス
```
ColdFusion.navigate(URL [, container, callbackhandler, errorHandler, httpMethod, formId])
```

関連項目
AjaxLink、cfajaximport、ColdFusion.Ajax.submitForm、『ColdFusion アプリケーションの開発』の Control container contents

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
URL	リンクの URL です。
container	リンク出力を表示するコントロールの name 属性の値です。このコントロールは、cfdiv、cflayoutarea、cfpod、cfwindow などのコンテナコントロールである必要があります。 この引数を省略すると、リンクが通常の URL として扱われ、ページ全体が更新されます。
callbackhandler	ターゲットが表示された後に呼び出す JavaScript 関数の名前です。

パラメータ	説明
errorHandler	この関数の実行時にエラーが発生したときに呼び出す JavaScript 関数の名前です。この関数は 2 つのパラメータ (HTTP エラーコードとエラーメッセージ) を取ることができます。
formId	URL に送信するフォームの ID または name 属性です。
httpMethod	URL に移動するときに使用する次の HTTP メソッドです。 • GET (デフォルト) • POST

戻り値
この関数は値を返しません。

使用方法
この関数を呼び出すページに ColdFusion AJAX ベースのコントロールがない場合は、そのページで cfajaximport タグを使用する必要があります。これにより、この関数に必要な JavaScript 定義がページ内に確実に含まれることになります。

callbackhandler パラメータは、コンテンツが表示された後にその表示を変更するのに役立ちます。たとえば、ColdFusion.navigate を呼び出す前に、ポッドのタイトルバーをイタリックにしてロード中であることを示したとします。callbackhandler 関数を使用すると、このタイトルバーは、移動の完了後に標準文字に切り替えたり、太字に変更することができます。同様に、callbackhandler を使用すると、ブックリーダーのページ番号を更新できます。

FormID 属性を使用すると、特定の URL に送信するフォームを指定できます。ColdFusion.Navigate 関数をこの属性とともに使用すると、フォーム外からフォームデータを非同期で送信し (ユーザーがメニューアイテムをクリックした場合など)、返された結果を特定のコンテナコントロールに出力できます。

例
ユーザーがウィンドウ 1 のリンクをクリックすると、ColdFusion.navigate 関数によって、ウィンドウ 2 のテキストが windowsrc.cfm のコンテンツに置き換わり、myCallback コールバックハンドラが呼び出されてコールバックの div 領域の innerHTML が変更されます。

アプリケーションのメインページは次のようになります。

```
<html>
<head>
<!--- The Callback handler puts text in the window.cfm callback div. --->
<script language="javascript">
    var mycallBack = function(){
        document.getElementById("callback").innerHTML = "<br><br><b>This is printed by the callback
handler.</b>";
    }

<!--- The error handler pops an alert with the error code and message. --->
    var myerrorhandler = function(errorCode,errorMessage){
        alert("[In Error Handler]" + "\n\n" + "Error Code: " + errorCode + "\n\n" + "Error Message: " +
errorMessage);
    }
</script>
</head>

<body>
<cfwindow name="w1" title="CF Window 1" initShow=true
    x=10 y=10 width="200">
    This is a cfwindow control.<br><br>
    <a href="javascript:ColdFusion.navigate('windowsource.cfm','w2',
        mycallBack,myerrorhandler);">Click</a> to navigate Window 2</a>
</cfwindow>

<cfwindow name="w2" title="CF Window 2" initShow=true
    x=250 y=10 width="200">
    This is a second cfwindow control.
</cfwindow>
</body>
</html>
```

windowsource.cfm ページは次のようになります。

```
This is markup from "windowsource.cfm"
<!-- The callback handler puts its output in the following div block. -->
<div id="callback"></div>
```

ColdFusion.ProgressBar.getProgressBarObject

説明

プログレスバーオブジェクトを取得します。

関数のシンタックス

```
ColdFusion.ProgressBar.getProgressBarObject(name)
```

関連項目

ColdFusion.ProgressBar.start、ColdFusion.ProgressBar.stop

履歴

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
name	プログレスバーオブジェクトの名前です。

戻り値
この関数は、基盤となる Ext JavaScript プログレスバーオブジェクトを返します。

使用方法
この関数は、プログレスバーオブジェクトを取得する場合に使用します。

ColdFusion.ProgressBar.hide

説明
プログレスバーが表示されている場合は非表示にします。

関数のシンタックス
```
ColdFusion.ProgressBar.hide(progressBarId)
```

関連項目
ColdFusion.ProgressBar.show、ColdFusion.ProgressBar.update

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
progressBarId	プログレスバーオブジェクトの名前です。有効な ColdFusion 識別子である必要があります。

戻り値
この関数は値を返しません。

使用方法
この関数により、プログレスバーが表示されている場合に非表示にできます。

ColdFusion.ProgressBar.show 関数を使用するか、cfprogressbar タグ内の autodisplay 属性を true に設定すると、プログレスバーが表示されます。

次の例は、この関数の使用方法について示しています。

```
<cfprogressbar name="pBar" autodisplay="true">
<cfinput type="button"
    onClick="ColdFusion.ProgressBar.hide('pBar')">
```

ColdFusion.ProgressBar.reset

説明
進行状況とメッセージをリセットします。

関数のシンタックス
```
ColdFusion.ProgressBar.reset(progressBarId)
```

関連項目
ColdFusion.ProgressBar.start、ColdFusion.ProgressBar.stop

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
progressBarId	プログレスバーオブジェクトの名前です。有効な ColdFusion 識別子である必要があります。

戻り値
この関数は値を返しません。

使用方法
次の例に示すように、プログレスバーの進行状況をリセットします。

```
<cfform>
<cfprogressbar name="pBar" width="500"/>
<cfinput type="button" name="pBtn" onClick="ColdFusion.ProgressBar.reset('pBar')">
</cfform>
```

ColdFusion.ProgressBar.show

説明
プログレスバーが非表示になっている場合は表示します。

関数のシンタックス
```
ColdFusion.ProgressBar.show(progressBarId)
```

関連項目
ColdFusion.ProgressBar.hide、ColdFusion.ProgressBar.update

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
progressBarId	プログレスバーオブジェクトの名前です。有効な ColdFusion 識別子である必要があります。

戻り値
この関数は値を返しません。

使用方法
この関数により、プログレスバーが非表示になっている場合に表示できます。

ColdFusion.ProgressBar.hide 関数を使用するか、cfprogressbar タグ内の autodisplay 属性を false に設定して、プログレスバーを非表示にします。

次の例は、この関数の使用方法について示しています。

```
<cfprogressbar name="pBar" autodisplay="false">
<cfinput type="button"
    onClick="ColdFusion.ProgressBar.show('pBar')">
```

ColdFusion.ProgressBar.start

説明
基盤となる (Ext JS JavaScript library) プログレスバーオブジェクトを開始します。

関数のシンタックス
```
ColdFusion.ProgressBar.start(name)
```

関連項目
ColdFusion.ProgressBar.getProgressBarObject、ColdFusion.ProgressBar.stop

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
name	プログレスバーオブジェクトの名前です。

戻り値
この関数は値を返しません。

使用方法
cfprogressbar タグによって HTML マークアップが作成されます。実行時にこの関数を使用して、プログレスバーオブジェクトを開始します。初期化時、ColdFusion では、バインド式を使用して定義されている、基盤となる CFC が指定された間隔で呼び出されます。CFC によって進行状況が返されます。進行状況は、プログレスバーの値を更新するために、基盤となる Ext プログレスバーオブジェクトに渡されます。

CFC によって返される進行状況オブジェクトでは、STATUS プロパティおよび MESSAGE プロパティが指定されている必要があります。STATUS プロパティの値は、0.0 から 1.0 の数値です。

ColdFusion.ProgressBar.stop

説明
実行中の、基盤となるプログレスバーオブジェクトを停止します。

関数のシンタックス
```
ColdFusion.ProgressBar.stop(name, calloncomplete)
```

関連項目
ColdFusion.ProgressBar.getProgressBarObject、ColdFusion.ProgressBar.start

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
name	プログレスバーオブジェクトの名前です。
calloncomplete	oncomplete 関数を呼び出すかどうかを指定するブール値です。 • true • false デフォルト値は true です。

戻り値
この関数は値を返しません。

使用方法
この関数は、プログレスバーオブジェクトを停止する場合に使用します。

ColdFusion.ProgressBar.update

説明
属性値を更新します。

関数のシンタックス
```
ColdFusion.ProgressBar.update(progressBarId,config)
```

関連項目
ColdFusion.ProgressBar.hide、ColdFusion.ProgressBar.show

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
progressBarId	プログレスバーオブジェクトの名前です。有効な ColdFusion 識別子である必要があります。
configObject	設定パラメータ interval、duration、および oncomplete を含むオブジェクトです。

戻り値
この関数は値を返しません。

使用方法
次の例に示すように、属性値 duration、interval、または oncomplete のいずれかを更新します。

```
<cfprogressbar name="pBar" autodisplay="false">
<cfinput type="button"
    onClick="ColdFusion.ProgressBar.update('pBar', {interval:6000, duration:10000,
oncomplete:newoncomplete})">
```

ColdFusion.ProgressBar.updatestatus

説明
プログレスバーのステータスとメッセージを手動で更新できます。

関数のシンタックス
`ColdFusion.ProgressBar.updatestatus(`**`progressBarId, status, message`**`)`

関連項目
ColdFusion.ProgressBar.update、ColdFusion.ProgressBar.reset

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
progressBarId	プログレスバーオブジェクトの名前です。有効な ColdFusion 識別子である必要があります。
message	プログレスバーに表示されるテキストです。
status	進行状況です。

戻り値
この関数は値を返しません。

使用方法
この関数により、プログレスバーのメッセージとステータスを手動で設定できます。たとえば、ファイルアップロードの場合、この関数を使用して進行状況を手動で制御できます。

次のコードは、この関数の使用方法を示しています。

```
<cfform>
<cfprogressbar name="pBar" width="500"/>
<cfinput type="button" name="pBtn" onClick=" ColdFusion.ProgressBar.updateStatus('pBar',0.5,'50%')">
</cfform>
```

ColdFusion.RichText.getEditorObject

説明
既存の textarea コントロールのリッチテキストエディターの基盤になる FCKEditor インスタンスを返します。

関数のシンタックス
```
object = ColdFusion.RichText.getEditorObject(textareaName)
```

関連項目
ColdFusion.ProgressBar.update、ColdFusion.ProgressBar.reset

履歴
ColdFusion 8 アップデート 1：この関数が追加されました。

パラメータ

パラメータ	説明
textareaName	リッチテキストエディターを指定する textarea タグの name 属性の値です。

戻り値
オブジェクト

使用方法
この関数は、内部の FCKEditor の操作へのアクセスを提供するオブジェクトを返します。

ColdFusion.RichText.onComplete

説明
リッチテキストエディターインスタンスが初期化を完了すると、ColdFusion によってこの関数が呼び出され、リッチテキストエディターの基盤になる FCKEditor インスタンスにこの関数が渡されます。

関数のシンタックス
```
ColdFusion.RichText.onComplete(editorInstance)
```

関連項目
ColdFusion.ProgressBar.update、ColdFusion.ProgressBar.reset

履歴
ColdFusion 8 アップデート 1：この関数が追加されました。

パラメータ

パラメータ	説明
editorInstance	リッチテキストエディターの基盤になる FCKEditor インスタンスです。

戻り値
この関数は値を返しません。

使用方法
リッチテキストエディターインスタンスが初期化を完了すると、ColdFusion によってこの関数が呼び出され、リッチテキストエディターの基盤になる FCKEditor インスタンスにこの関数が渡されます。この関数では、さらに初期化後に必要な操作を実行できます。

例

```
<html>
    <head>
        <script type="text/javascript">
        //User defines this function to get control once the rich text editor
        //is done with its initialization
        ColdFusion.RichText.OnComplete = function(editorInstance)
        {
            alert("on complete called");
            alert("editor instance = " + editorInstance));
        }
        </script>
    </head>
<body>
    <cfform>
        <cftextarea richtext = true name="richtext1">
    </cfform>
</body>
</html>
```

ColdFusion.setGlobalErrorHandler

説明
ColdFusion AJAX 機能の利用時にエラーが発生した場合にデフォルトの ColdFusion AJAX エラーハンドラの代わりに呼び出す関数を指定します。

関数のシンタックス

```
ColdFusion.setGlobalErrorHandler(functionName)
```

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
functionName	ColdFusion AJAX コードにエラー (バインドエラーなど) が存在する場合に実行する JavaScript 関数の名前です。この関数は 1 つの引数 (エラーメッセージ文字列) を取る必要があります。

戻り値
この関数は値を返しません。

使用方法
この関数を呼び出すページに ColdFusion AJAX ベースのコントロールがない場合は、そのページで cfajaximport タグを使用します。これにより、この関数に必要な JavaScript 定義がページ内に確実に含まれることになります。

このグローバルエラーハンドラは、ColdFusion AJAX 機能で発生したエラーに関する情報を表示します。デフォルトのグローバルエラーハンドラは、エラーメッセージを含む警告を表示します。この関数を使用すると、アプリケーションに関する追加情報を含むカスタムエラーウィンドウを表示するカスタムグローバルエラーハンドラなどを作成できます。

ColdFusion.Slider.disable

説明
スライダコントロールを無効にします。

関数のシンタックス
```
ColdFusion.Slider.disable(name)
```

関連項目
ColdFusion.Slider.enable、ColdFusion.Slider.getValue、ColdFusion.Slider.getSliderObject、ColdFusion.Slider.hide、ColdFusion.Slider.show、ColdFusion.Slider.setValue

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
name	cfslider タグの name 属性の値を指定します。

戻り値
この関数は値を返しません。

例
```
<h3>This is an example of the Slider.disable function. Click the Disable Slider button to disable the
slider.</h3>
    <script language="JavaScript" type="text/javascript">
    function disable(){
    ColdFusion.Slider.disable('sliderID');
    }
    </script>
    <br>
    <cfform name="form01">
    <br/>
    <cfslider name="sliderID" format="HTML"
    vertical="false" width="350"
    value="100" min="0" max="200" increment="10" tip="true"/>
    <cfinput type="button" name="htmlbutton"
    value="Disable Slider" onclick="disable()">
    </cfform>
```

ColdFusion.Slider.enable

説明
スライダコントロールを有効にします。

関数のシンタックス
```
ColdFusion.Slider.enable(name)
```

関連項目
ColdFusion.Slider.disable、ColdFusion.Slider.getValue、ColdFusion.Slider.getSliderObject、ColdFusion.Slider.hide、ColdFusion.Slider.show、ColdFusion.Slider.setValue

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
name	cfslider タグの name 属性の値を指定します。

戻り値
この関数は値を返しません。

例
```
<h3>This is an example of the Slider.enable function. Click the Enable Slider button to enable the
slider.</h3>
    <script language="JavaScript" type="text/javascript">
    function onload()
    {
    ColdFusion.Slider.disable('sliderID');
    }
    function enable(){
    ColdFusion.Slider.enable('sliderID');
    }
    </script>
    <br>
    <cfform name="form01">
    <br/>
    <cfslider name="sliderID" format="HTML"
    body onLoad="onload()"
    vertical="false" width="350"
    value="100" min="0" max="200"
    increment="10" tip="true"/>
    <cfinput type="button" name="htmlbutton"
    value="Enable Slider" onclick="enable()">
    </cfform>
```

ColdFusion.Slider.getValue

説明
スライダコントロールの数値を取得します。

関数のシンタックス

```
ColdFusion.Slider.getValue(name)
```

関連項目

ColdFusion.Slider.disable、ColdFusion.Slider.enable、ColdFusion.Slider.getSliderObject、ColdFusion.Slider.hide、ColdFusion.Slider.show、ColdFusion.Slider.setValue

履歴

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
name	cfslider タグの name 属性の値を指定します。

戻り値

この関数は数値を返します。

例

```
<h3>This is an example of the Slider.getvalue function. Click the Get Value button to get the value of the
slider.</h3>
    <script language="JavaScript" type="text/javascript">
    function getvalue(){
    alert("The slider is currently at: "+ColdFusion.Slider.getValue('sliderID'));
    }
    </script>
    <br>
    <cfform name="form01">
    <br/>
    <cfslider name="sliderID" format="HTML"
    vertical="false" width="350"
    value="100" min="0" max="200" increment="10" tip="true"/>
    <cfinput type="button" name="htmlbutton"
    value="Get Value" onclick="getvalue()">
    </cfform>
```

ColdFusion.Slider.getSliderObject

説明

基盤となる Ext JavaScript スライダコントロールを取得します。

関数のシンタックス

```
ColdFusion.Slider.getSliderObject(name)
```

関連項目

ColdFusion.Slider.disable、ColdFusion.Slider.enable、ColdFusion.Slider.getValue、ColdFusion.Slider.hide、ColdFusion.Slider.show、ColdFusion.Slider.setValue

履歴

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
name	cfslider タグの name 属性の値を指定します。

戻り値
この関数は、基盤となる Ext JavaScript スライダコントロールを返します。

ColdFusion.Slider.hide

説明
スライダコントロールを非表示にします。

関数のシンタックス
```
ColdFusion.Slider.hide(name)
```

関連項目
ColdFusion.Slider.disable、ColdFusion.Slider.enable、ColdFusion.Slider.getSliderObject、ColdFusion.Slider.getValue、ColdFusion.Slider.show、ColdFusion.Slider.setValue

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
name	cfslider タグの name 属性の値を指定します。

戻り値
この関数は値を返しません。

例
```
<h3>This is an example of the Slider.hide function. Click the Hide button to hide the slider.</h3>
    <script language="JavaScript" type="text/javascript">
    function hide(){
    ColdFusion.Slider.hide('sliderID');
    }
    </script>
    <br>
    <cfform name="form01">
    <br/>
    <cfslider name="sliderID" format="HTML"
    vertical="false" width="350"
    value="100" min="0" max="200"
    increment="10" tip="true"/>
    <cfinput type="button" name="htmlbutton"
    value="Hide" onclick="hide()">
    </cfform>
```

ColdFusion.Slider.show

説明
スライダコントロールを表示します。

関数のシンタックス
```
ColdFusion.Slider.show(name)
```

関連項目
ColdFusion.Slider.disable、ColdFusion.Slider.enable、ColdFusion.Slider.getSliderObject、ColdFusion.Slider.getValue、ColdFusion.Slider.hide、ColdFusion.Slider.setValue

履歴
ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
name	cfslider タグの name 属性の値を指定します。

戻り値
この関数は値を返しません。

例
```
<h3>This is an example of the Slider.show function. Click the Show Slider button to show the slider.</h3>
    <script language="JavaScript" type="text/javascript">
    function onload(){
    ColdFusion.Slider.hide('sliderID');
    }
    function show(){
    ColdFusion.Slider.show('sliderID');
    }
    </script>
    <br>
    <cfform name="form01">
    <br/>
    <cfslider name="sliderID" format="HTML"
    vertical="false" width="350"
    value="100" min="0" max="200"
    increment="10" tip="true"/>
    <cfinput type="button" name="htmlbutton"
    value="Show Slider" onclick="show()">
    </cfform>
```

ColdFusion.Slider.setValue

説明
スライダコントロールの数値を設定します。

関数のシンタックス

```
ColdFusion.Slider.setValue(name, newValue)
```

関連項目

ColdFusion.Slider.disable、ColdFusion.Slider.enable、ColdFusion.Slider.getSliderObject、ColdFusion.Slider.getValue、ColdFusion.Slider.hide、ColdFusion.Slider.show

履歴

ColdFusion 9: この関数が追加されました。

パラメータ

パラメータ	説明
name	cfmediaplayer タグの name 属性の値を指定します。
newValue	スライダコントロールの数値です。

戻り値

この関数は値を返しません。

例

```
<h3>This is an example of the Slider.setValue function. Click the Set Value button to set the pointer to a
new value,150.</h3>
    <script language="JavaScript" type="text/javascript">
    function setvalue(){
    ColdFusion.Slider.setValue('sliderID','150');
    }
    </script>
    <br>
    <cfform name="form01">
    <br/>
    <cfslider name="sliderID" format="HTML"
    vertical="false" width="350"
    value="100" min="0" max="200"
    increment="10" tip="true"/>
    <cfinput type="button" name="htmlbutton"
    value="Set Value" onclick="setvalue()">
    </cfform>
```

ColdFusion.Tree.getTreeObject

説明

指定された HTML ツリーに対して、基盤となるオブジェクトを取得します。

関数のシンタックス

```
ColdFusion.Tree.getTreeObject(name)
```

関連項目

cftree、cfajaximport、ColdFusion.Tree.refresh、『ColdFusion アプリケーションの開発』の Using HTML trees

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	そのオブジェクトを使用する cftree タグの name 属性の値です。

戻り値
YAHOO.widget.TreeView タイプのオブジェクト

使用方法
この関数は、HTML の cftree コントロールの基盤になる Yahoo User Interface Library の YAHOO.widget.TreeView オブジェクトを取得するときに使用します。この未処理オブジェクトを使用することで、表示されるツリーに変更を加えることができます。このオブジェクトの詳細および管理方法については、Yahoo Toolkit のマニュアルを参照してください。

ColdFusion.Tree.refresh

説明
HTML ツリーを更新して、すべてのアイテムを最新の値に更新します。

関数のシンタックス
```
ColdFusion.Tree.refresh(name)
```

関連項目
cftree、cfajaximport、ColdFusion.Tree.getTreeObject、『ColdFusion アプリケーションの開発』の Using HTML trees

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	そのオブジェクトを使用する cftree タグの name 属性の値です。

戻り値
YAHOO.widget.TreeView タイプのオブジェクト

使用方法
この関数を使用すると、手動でツリーを更新できます。バインド式を使用してツリーを挿入する場合は、refresh を呼び出すと、バインド式が再評価されてツリーのルートノードが再挿入されます。cftree バインド式を実行するイベントに依存しないサーバーから最新のデータを取得する必要がある場合は、この関数を使用します。たとえば、この関数を使用してファイルやフォルダのツリーを定期的に更新し、サーバーのステータスを表示します。

ColdFusion.Window.create

説明
ColdFusion ポップアップウィンドウを作成します。この関数は cfwindow タグに相当します。

関数のシンタックス
```
ColdFusion.Window.create(name, title, URL [, configuration])
```

関連項目
cfwindow、ColdFusion.Window.getWindowObject、ColdFusion.Window.hide、ColdFusion.Window.onHide、ColdFusion.Window.onShow、ColdFusion.Window.show、ColdFusion.Tree.getTreeObject、『ColdFusion アプリケーションの開発』の Using pop-up windows

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	ウィンドウの名前です。ウィンドウを操作する必要がある場合 (ウィンドウをダイナミックに表示したり隠したりする必要がある場合)、この属性は必須です。指定された名前のウィンドウが存在する場合はそのウィンドウが表示され、残りのパラメータは無視されます。既存のウィンドウ名を指定しない場合は、ページ内で一意の名前を指定する必要があります。
title	ウィンドウのタイトルバーに表示するテキストです。HTML マークアップを使用してタイトルの外観を制御することもできます。
URL	ウィンドウ本体のコンテンツをどの URL から取得するかを指定します。この属性で URL パラメータを使用すると、ページにデータを渡すことができます。ColdFusion は標準のページパス解決ルールを使用してページを探します。 **メモ :** この属性で指定したページに ColdFusion AJAX 機能を使用するタグ (cfform タグ、cfgrid タグ、cfpod タグなど) が含まれている場合は、この関数を含むページで cfajaximport タグ内に各タグを指定する必要があります。詳細については、cfajaximport を参照してください。
configuration	ウィンドウ設定パラメータを含むオブジェクトです。詳細については、「使用方法」を参照してください。
refreshonshow	デフォルト値は false です。

戻り値
この関数は値を返しません。

使用方法
この関数は cfwindow タグに相当します。

この関数を呼び出すページで cfwindow タグも使用していない場合は、そのページで cfajaximport タグを指定し、tags 属性に cfwindow を指定する必要があります。これにより、そのページには、ウィンドウを作成するために必要な JavaScript が含まれます。たとえば、他の ColdFusion AJAX 機能を使用するために JavaScript をインポートする必要がない場合は、次のように記述します。

```
<cfajaximport tags="cfwindow">
```

configuration パラメータはウィンドウの特性を定義します。このパラメータでは、次のエントリの一部またはすべてを指定できます。

エントリ	デフォルト	説明
callbackhandler		ウィンドウ本体がロードされたときに呼び出される関数です。この関数には引数を渡せません。
center	false	このウィンドウをブラウザウィンドウの中央に配置するかどうかを指定するブール値です。 • true の場合、x および y 属性の値は無視されます。 • この属性が false に設定されていて、x および y 属性が指定されていない場合、ウィンドウは中央に配置されます。
closable	true	ユーザーがウィンドウを閉じることを許可するかどうかを指定するブール値です。true の場合は、ウィンドウ上に X 印のウィンドウを閉じるアイコンが表示されます。
draggable	true	ユーザーがウィンドウをドラッグすることを許可するかどうかを指定するブール値です。ウィンドウをドラッグするには、タイトルバーの上でマウスをクリックして、ボタンを押したままドラッグします。ウィンドウにタイトルがない場合はドラッグできません。
errorHandler		ウィンドウ本体のロード時にエラーが発生した場合に呼び出される関数です。この関数は次の 2 つの引数を取る必要があります。 • HTTP ステータスコード (エラーが HTTP のエラーでない場合は -1) • エラーメッセージ
height	300	ウィンドウの高さです (単位 : ピクセル)。使用可能なスペースを超える値を指定すると、使用可能なスペースがウィンドウによってすべて占有され、サイズを変更するためのハンドルが表示されません。
initshow	false	このウィンドウを含むページが最初に表示されたときに、このウィンドウを表示するかどうかを指定するブール値です。この値が false の場合は、ColdFusion.Window.show JavaScript 関数を使用してウィンドウを表示します。
minheight	0	ユーザーがウィンドウのサイズを変更した場合の最小の高さです (単位 : ピクセル)。 このパラメータと resizable="false" パラメータを指定すると、エラーになります。
minwidth	0	ユーザーがウィンドウのサイズを変更した場合の最小の幅です (単位 : ピクセル)。 このパラメータと resizable="false" パラメータを指定すると、エラーになります。
modal	false	このウィンドウがモーダルであるかどうか (つまり、このウィンドウが表示されている間、ユーザーがメインウィンドウを操作できないようにするかどうか) を指定するブール値です。true の場合、ユーザーはメインウィンドウを操作できません。
resizable	true	ユーザーがウィンドウのサイズを変更することを許可するかどうかを指定するブール値です。
width	500	ウィンドウの幅です (単位 : ピクセル)。使用可能なスペースを超える値を指定すると、使用可能なスペースがウィンドウによってすべて占有され、サイズを変更するためのハンドルが表示されません。
x		ブラウザウィンドウを基準とした、ウィンドウの左上隅の X (水平) 座標です。 center 属性の値が true で、y 属性の値が設定されていない場合、この属性は無視されます。
y		ブラウザウィンドウを基準とした、ウィンドウの左上隅の Y (垂直) 座標です。 center 属性の値が true で、x 属性の値が設定されていない場合、この属性は無視されます。

注意 : configuration オブジェクトのエントリ名はすべて小文字にする必要があります。

例

次の CFML アプリケーションは、ウィンドウを作成した後、hello1.cfm ファイルからウィンドウのコンテンツを取得します。

```
<cfajaximport tags="cfwindow">

<cfform name="test">
    <cfinput type="button" name="x" value="Create Window"
        onClick="ColdFusion.Window.create('Window1', 'This is a CF window',
        'http://localhost:8500/My_stuff/AjaxUI/Book/hello1.cfm',
        {x:100,y:100,height:300,width:400,modal:false,closable:false,
        draggable:true,resizable:true,center:true,initshow:true,
        minheight:200,minwidth:200 })">
</cfform>
```

hello1.cfm ファイルの内容は、次のようなに簡単な文字列でもかまいません。

```
Hello from hello1.cfm
```

ColdFusion.Window.getWindowObject

説明
指定されたウィンドウに対して、基盤となるオブジェクトを取得します。

関数のシンタックス
```
ColdFusion.Window.getWindowObject(name)
```

関連項目
cfwindow、ColdFusion.Window.create、ColdFusion.Window.onHide、ColdFusion.Window.onShow、ColdFusion.Window.show、ColdFusion.Tree.getTreeObject、『ColdFusion アプリケーションの開発』の Using pop-up windows

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	そのオブジェクトを使用する cfwindow タグの name 属性の値です。

戻り値
Ext.BasicDialog タイプのオブジェクト

使用方法
この関数は、HTML の cfwindow コントロールの基盤になる Ext JavaScript Library の Ext.BasicDialog オブジェクトを取得するときに使用します。この未処理オブジェクトを使用することで、表示されるウィンドウに変更を加えることができます。このオブジェクトの詳細および管理方法については、Ext JavaScript ライブラリのマニュアルを参照してください。

ColdFusion.Window.getWindowObject

説明
指定されたウィンドウに対して、基盤となるオブジェクトを取得します。

関数のシンタックス

```
ColdFusion.Window.getWindowObject(name)
```

関連項目

cfwindow、ColdFusion.Window.create、ColdFusion.Window.hide、ColdFusion.Window.onHide、ColdFusion.Window.onShow、ColdFusion.Window.show、ColdFusion.Tree.getTreeObject、『ColdFusion アプリケーションの開発』の Using pop-up windows

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	そのオブジェクトを使用する cfwindow タグの name 属性の値です。

戻り値

Ext.BasicDialog タイプのオブジェクト

使用方法

この関数は、HTML の cfwindow コントロールの基盤になる Ext JavaScript Library の Ext.BasicDialog オブジェクトを取得するときに使用します。この未処理オブジェクトを使用することで、表示されるウィンドウに変更を加えることができます。このオブジェクトの詳細および管理方法については、Ext JavaScript ライブラリのマニュアルを参照してください。

ColdFusion.Window.destroy

説明

ウィンドウインスタンスを破棄します。

関数のシンタックス

```
ColdFusion.Window.destroy(windowName[, destroyElement])
```

関連項目

cfwindow、ColdFusion.Window.create、ColdFusion.Window.getWindowObject、ColdFusion.Window.onHide、ColdFusion.Window.onShow、ColdFusion.Window.show、ColdFusion.Tree.getTreeObject、『ColdFusion アプリケーションの開発』の Using pop-up windows

履歴

ColdFusion 8 アップデート 1：この関数が追加されました。

パラメータ

パラメータ	説明
windowName	破棄するウィンドウの名前です。
destroyElement	ウィンドウに関連付けられている HTML 要素も一緒に破棄するかどうかを指定するブール値です。デフォルト値は false です。

戻り値
この関数は値を返しません。

使用方法
この関数は、ウィンドウインスタンスを破棄するために使用します。

ColdFusion.Window.hide

説明
現在表示されているウィンドウを非表示にします。

関数のシンタックス
```
ColdFusion.Window.hide(name)
```

関連項目
cfwindow、ColdFusion.Window.create、ColdFusion.Window.getWindowObject、ColdFusion.Window.onHide、ColdFusion.Window.onShow、ColdFusion.Window.show、ColdFusion.Tree.getTreeObject、『ColdFusion アプリケーションの開発』の Using pop-up windows

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	非表示にするウィンドウの name 属性です。

戻り値
この関数は値を返しません。

使用方法
このタグは、ウィンドウが既に非表示にされている場合には効果がありません。

例
次のコードでは、ボタンをクリックしてウィンドウを表示または非表示にできます。

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
</head>
<body>

<cfwindow name="testWindow" initshow=true title="test window" closable=true>
    Window contents
</cfwindow>

<cfform>
    <cfinput name="hidebutton" type="button" value="Hide Window"
        onclick="javascript:ColdFusion.Window.hide('testWindow');"/>
    <cfinput name="showbutton" type="button" value="Show Window"
        onclick="javascript:ColdFusion.Window.show('testWindow');"/>
</cfform>
</body>
</html>
```

ColdFusion.Window.onHide

説明
特定のウィンドウを非表示にするたびに実行される関数を指定します。

関数のシンタックス
```
ColdFusion.Window.onHide(windowName, handler)
```

関連項目
cfwindow、ColdFusion.Window.create、ColdFusion.Window.getWindowObject、ColdFusion.Window.hide、ColdFusion.Window.onShow、ColdFusion.Window.show、ColdFusion.Tree.getTreeObject、『ColdFusion アプリケーションの開発』の Using pop-up windows

履歴
ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
windowName	ウィンドウの名前です。ハンドラ関数は、このウィンドウを非表示にするたびに実行されます。
handler	ウィンドウを非表示にするときに実行する JavaScript 関数です。

戻り値
この関数は値を返しません。

使用方法
handler パラメータで指定した関数は、必要に応じて、ウィンドウ名が含まれたパラメータを 1 つ取ることができます。

例
次の例では、ColdFusion.Window.onHide 関数を使用して、ウィンドウを非表示にするボタンをクリックしたときに、そのウィンドウに関する情報が含まれた警告を表示します。

```
<head>
    <script language="javascript">
        function onhide(name) {
            alert("window hidden = " + name);
        }

        function test() {
            ColdFusion.Window.onHide("testWindow", onhide);
            ColdFusion.Window.hide("testWindow");
        }
    </script>
</head>
<body>

<cfwindow name="testWindow" initshow=true title="test window"
closable=true>
    Window contents
</cfwindow>

<cfform>
    <cfinput name="button" value="Hide Window" onclick="javascript:test()" type="button"/>
</cfform>
</body>
</html>
```

ColdFusion.Window.onShow

説明

特定のウィンドウを表示するたびに実行される関数を指定します (ウィンドウを作成し、initShow 属性または設定のエントリ値を true に指定する場合など)。

関数のシンタックス

```
ColdFusion.Window.onShow(windowName, handler)
```

関連項目

cfwindow、ColdFusion.Window.create、ColdFusion.Window.getWindowObject、ColdFusion.Window.onHide、ColdFusion.Window.hide、ColdFusion.Window.show、ColdFusion.Tree.getTreeObject、『ColdFusion アプリケーションの開発』の Using pop-up windows

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
windowName	ウィンドウの名前です。ハンドラ関数は、このウィンドウを表示するたびに実行されます。
handler	ウィンドウを表示するときに実行する JavaScript 関数です。

戻り値

この関数は値を返しません。

使用方法

handler パラメータで指定した関数は、必要に応じて、ウィンドウ名が含まれたパラメータを 1 つ取ることができます。

この関数で使用するパラメータでは、該当するウィンドウが表示された場合にのみ、ウィンドウのデータを取得します。cfajaxproxy タグを使用してデータを提供する CFC 関数の JavaScript プロキシを作成し、ColdFusion.Window.onShow 関数には、そのプロキシ関数を呼び出してウィンドウのコンテンツを新しいデータで更新する関数を指定することもできます。

例

次の例では、ColdFusion.Window.onShow 関数を使用して、ウィンドウを表示するボタンをクリックしたときに、そのウィンドウに関する情報が含まれた警告を表示します。

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <script language="javascript">
        function onshow(name) {
            alert("window shown = " + name);
        }

        function test() {
            ColdFusion.Window.onShow("testWindow", onshow);
            ColdFusion.Window.show("testWindow");
        }
    </script>
</head>
<body>

<cfwindow name="testWindow" initshow=false title="test window"
    closable=true>
    Window contents
</cfwindow>

<cfform>
    <cfinput name="button" value="show Window" onclick="javascript:test()" type="button"/>
</cfform>
</body>
</html>
```

ColdFusion.Window.show

説明

現在非表示になっているウィンドウを表示します。

関数のシンタックス

```
ColdFusion.Window.show(name)
```

関連項目

cfwindow、ColdFusion.Window.create、ColdFusion.Window.getWindowObject、ColdFusion.Window.hide、ColdFusion.Window.onHide、ColdFusion.Window.onShow、ColdFusion.Tree.getTreeObject、『ColdFusion アプリケーションの開発』の Using pop-up windows

履歴

ColdFusion 8: この関数が追加されました。

パラメータ

パラメータ	説明
name	表示するウィンドウの name 属性です。

戻り値
この関数は値を返しません。

使用方法
この関数では、initShow 属性またはパラメータ値を false に指定して作成したウィンドウ、あるいは ColdFusion.Window.hide 関数を呼び出して非表示にしたウィンドウが表示されます。ユーザーがタイトルバーの上にある X のアイコンをクリックして閉じたウィンドウは表示されません。

この関数は、ウィンドウが既に表示されている場合には効果がありません。

例
ColdFusion.Window.hide の例を参照してください。

JavaScript 関数

今回のリリースで追加された Ajax JavaScript 関数は、次のとおりです。

ColdFusion.Autosuggest.getAutosuggestObject

説明
基盤となる YUI AutoComplete オブジェクトにアクセスできるので、オブジェクトをより詳細に管理できます。たとえば、オブジェクトにイベントを関連付けたりできます。

戻り値
基盤となる AutoComplete オブジェクト

関数のシンタックス
ColdFusion.Autosuggest.getAutosuggestObject (Id)

パラメータ
- Id: 入力候補オブジェクトの名前です。

例

```
<html>
      <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
      <head>
            <cfajaximport tags="cfinput-autosuggest">
            <script>
                  var init = function()
                  {
                        autosuggestobj = ColdFusion.Autosuggest.getAutosuggestObject('state');
                        autosuggestobj.itemSelectEvent.subscribe(foo);
                  }
                  var foo = function(event,args)
                  {
                        var msg  = "";
                        msg = msg + "Event: " + event + "\n\n";
                        msg = msg + "Selected Item: " + args[2] + "\n\n";
                        msg = msg + "Index: " + args[1]._nItemIndex + "\n\n";
                        alert(msg);
                  }
                  var getStates = function(){
                        return
["California","Connecticut","Colorado","Illinois","Alabama","Iowa","Utah","Alaska"];
                  }
            </script>
      </head>
      <body>
            <h3>Attaching an event handler to the autosuggest object</h3>
            <cfform name="mycfform" method="post" >
                  State:<BR>
                  <cfinput
                        type="text"
                        name="state"
                        autosuggest="javascript:getStates({cfautosuggestvalue})"
                        autosuggestMinLength=1
                        autosuggestBindDelay=1>
                  <cfset ajaxOnLoad("init")>
            </cfform>
      </body>
</html>
```

ColdFusion.Layout.disableSourceBind

説明

ソースバインドを無効にします。

関数のシンタックス

ColdFusion.Layout.disableSourceBind(Id)

パラメータ

- Id: レイアウト領域の名前です。

使用方法

たとえば、Coldfusion.navigate を使用して、タブまたはアコーディオンパネルに内容を挿入するとします。cflayoutarea で source 属性が定義されている場合は、(ColdFusion.navigate から取得するのではなく) ソースバインドの呼び出しで取得した内容のインスタンスを使用できます。

このようなインスタンスでは、ソースバインドを無効にして、Coldfusion.navigate で内容を取得したい場合が考えられます。

例

layout.cfm では、Tab1_Src.cfm、Tab2_Src.cfm、および Tab3_Src.cfm のテンプレートを使用します。layout.cfm を実行すると、次のことがわかります。

- [Navigate] をクリックすると、navigate.cfm ではなく tab2_src.cfm の内容が挿入されます。
- [Disable Source Bind] をクリックすると、navigate.cfm の内容が tab2_src に挿入されます。
- [Enable Source Bind] をクリックしてから、tab2_src をクリックすると、tab2_src の内容が再び挿入されます。

Tab1_Src.cfm

```
<br><cfdump var="#CGI#" keys="15" label="[CGI scope]"><br>
```

Tab2_Src.cfm

```
<br><cfdump var="#server#" label="[Server scope]"><br>
```

Tab3_Src.cfm

```
<br><cfdump var="#server.coldfusion#" label="[Showing key coldfusion in server scope]"><br>
```

Tab4_Src.cfm

```
<br><cfdump var="#server.os#" label="[Showing key OS in server scope]"><br>
```

layout.cfm

```
<script>
      var navigateToTab = function(layoutId,tabId){
            alert("Navigating to " + tabId);
            ColdFusion.Layout.selectTab(layoutId,tabId);
            ColdFusion.navigate('navigate.cfm',tabId);
      }
      var disableBind = function(tabId){
            alert("Disabling binding on source for " + tabId);
            ColdFusion.Layout.disableSourceBind(tabId);
      }
      var enableBind = function(tabId){
            alert("Enabling binding on source for " + tabId);
            ColdFusion.Layout.enableSourceBind(tabId);
      }

</script>
<cflayout type="tab" name="layout1">
      <cflayoutarea
            name = "tab1"
            overflow = "auto"
            refreshonactivate = "yes"
            title = "Tab 1"
            source = "tab1_src.cfm"/>
      <cflayoutarea
            name = "tab2"
            overflow = "auto"
```

```
            refreshonactivate = "false"
            title = "Tab 2"
            source = "tab2_src.cfm"
            bindonload=false
            />
      <cflayoutarea
            name = "tab3"
            overflow = "auto"
            refreshonactivate = "yes"
            title = "Tab 3"
            source = "tab3_src.cfm"
            />
</cflayout>
<cfform name="myform">
      <cfinput type="button" name="disable" value="Disable Source Bind"
onClick="javascript:disableBind('tab2')">
     <cfinput type="button" name="b" value="Navigate" onClick="javascript:navigateToTab('layout1','tab2')">
      <cfinput type="button" name="disable" value="Enable Source Bind"
onClick="javascript:enableBind('tab2')">
</cfform>
```

ColdFusion.Layout.enableSourceBind

説明
ソースバインドが無効になっている場合は有効にします。

関数のシンタックス
ColdFusion.Layout.enableSourceBind(Id)

パラメータ
- Id: レイアウト領域の名前です。

使用方法
ColdFusion.Layout.disableSourceBind の「使用方法」を参照してください。

例
ColdFusion.Layout.disableSourceBind の例を参照してください。

ColdFusion.FileUpload.getSelectedFiles

説明
アップロード対象として選択されたファイルの名前とサイズが含まれるオブジェクトの配列を返します。ファイルサイズは、バイト単位で返されます。

また、この関数は、ファイルアップロードステータスを YES|NO|Error で返します。

関数のシンタックス
ColdFusion.FileUpload.getSelectedFiles(Id)

パラメータ
- Id: cffileupload コントロールの名前です。

使用方法

実際のシナリオでは、通常、アップローダは別のコントロールと一緒に使用します。たとえば、名前、電子メール、アップローダの 3 つのフィールドがあるフォームなどがあります。仮に、ファイルをアップロードしたが、[送信] をクリックし忘れてしまった、またはファイルを選択してフォームを送信したが、[アップロード] をクリックし忘れてしまったということがあるとします。

この関数を使用すると、アップロード対象として選択したファイルがあることや、次の詳細情報をユーザーに通知することができます。

- FILENAME: アップロード対象として選択されたファイルの名前です。
- SIZE: ファイルのサイズです (バイト単位)。
- UPLOADSTATUS: YES|NO|Error

例

次の例は、ユーザーが [Submit] をクリックすると、アップロード対象として選択されたファイルに関する通知が行われるシナリオを示しています。

```
<html>
<head>
    <script language="javascript">
        var formatNumber = function(num){
            if(num < 1024) return num + "  bytes";
            if(num < (1024 * 1024)) return (num/1024).toFixed(2) + " KB";
            if(num < (1024 * 1024  *1024)) return (num/(1024 * 1024)).toFixed(2) + " MB";
            return (num/(1024 * 1024 * 1024)).toFixed(2) + " GB";
        }
        var getSelectedList = function(id){
            var files = ColdFusion.FileUpload.getSelectedFiles(id);
            var fileslist = "";
            if(files.length)
                fileslist = "You have selected The following files for upload: \n\n";
            for(var i=0;i < files.length; i++){
                fileslist = fileslist + files[i].FILENAME + "  (" + formatNumber(files[i].SIZE) + ")"
                if(i != files.length-1)
                    fileslist = fileslist + "\r\n";
            }
            if(files.length)
            {
                alert(fileslist);
            }
        }
    </script>
</head>
<body>
<br>
<cfform name="frmUpload" method="POST">
    First Name: <cfinput type="text" name="fname" value=""><br>
    Last Name: <cfinput type="text" name="lname" value=""><br><br>
    <cffileupload
        url="uploadAll.cfm"
        name="myuploader1"
        hideUploadButton=false
        onUploadComplete="foo"
        /><br><br>
<cfinput type="button" name="submitForm2" value="Submit" onClick="getSelectedList('myuploader1')">
</cfform>
</body>
</html>
```

Coldfusion.fileUpload.setUrl

説明
ファイルアップロードコントロールの URL をダイナミックに設定する場合に使用します。

戻り値
なし

関数のシンタックス
ColdFusion.fileUpload.setUrl(id, url)

パラメータ
- Id: アップロードコントロールの名前です。
- Url: URL には、絶対 URL、相対 URL、または完全修飾 URL を設定できます。

例
```
<script language="javascript">
    var uploadDone = function(result){
        alert("File uploaded");
    }

    var setUploadUrl = function(id)
    {
        var selectedFiles = ColdFusion.FileUpload.getSelectedFiles(id);
        var uploadUrl = "/manual/ajaxui/cffileupload/setUrl/includes/_uploadall.cfm";
        alert("Upload URL : " + uploadUrl);
        if(selectedFiles.length){
            ColdFusion.FileUpload.setURL(id,uploadUrl);
            ColdFusion.FileUpload.startUpload(id);
        }
    }
    var callbackhandler = function(obj)
    {
        var fileName = obj["FILENAME"];
        var status = obj["STATUS"];
        var message = obj["MESSAGE"];
        var msg = "In callbackhandler()" + "\n\n" +
                  "FILENAME: " + fileName + "\n\n" +
                  "STATUS: " + status + "\n\n" +
```

```
                    "MESSAGE: " + message
        alert(msg);
    }
    var errorhandler = function()
    {
        alert("In errorhandler()");
    }
    var uploadcompleted = function()
    {
        alert("All files have been uploaded successfully");
    }
</script>
<cfform name="frmUpload">
    <br>
    <cffileupload name="uploader" hideuploadbutton="true" onComplete="uploadDone" onError="errorhandler"
onUploadComplete="uploadcompleted">
    <br>
    <cfinput type="button" name="submit" value="Click to set URL and Upload Files"
onClick="setUploadUrl('uploader')">
</cfform>
```

ColdFusion.grid.getSelectedRows

説明
グリッドの選択した行のデータを取得する場合に使用します。

戻り値
行データが含まれたオブジェクトの配列

関数のシンタックス
ColdFusion.grid.getSelectedRows(id)

パラメータ
- Id: cfgrid を使用して定義したグリッドの名前です。

関連項目
FileUpload

使用方法
ColdFusion.grid.clearSelectedRows の例を参照してください。

例
ColdFusion.grid.clearSelectedRows の例を参照してください。

ColdFusion.grid.clearSelectedRows

説明
グリッドの選択した行をクリアする場合に使用します。

戻り値
なし

関数のシンタックス
ColdFusion.grid.clearSelectedRows(id)

パラメータ
- Id: cfgrid を使用して定義したグリッドの名前です。

使用方法
次の例を参照してください。

例
Employee.cfm

```
<html>
<head>
    <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
    <cfajaxproxy cfc="emp" jsclassname="emputils">
    <script language="javascript">
        var emp = new emputils();
        var deleteAllSelectedRows = function(grid)
        {
            emp.setHTTPMethod("POST");
            emp.deleteEmployees(getAllSelectedRows(grid,false));
            ColdFusion.Grid.refresh(grid);
        }
        var getAllSelectedRows = function(grid,showalert)
        {
            obj = ColdFusion.Grid.getSelectedRows(grid);
            jsonbj = ColdFusion.JSON.encode(obj);
            if(showalert)
                alert(jsonbj);
            return obj;
        }
        var clearAllSelectedRows = function(grid)
        {
            ColdFusion.Grid.clearSelectedRows(grid);
        }
    </script>
</head>
<body>
<cfform>
    <cfgrid
        format="html"
        name="empListing"
        selectmode="edit"

bind="cfc:emp.getEmployees({cfgridpage},{cfgridpagesize},{cfgridsortcolumn},{cfgridsortdirection})"
        onchange="cfc:emp.editEmployees({cfgridaction},{cfgridrow},{cfgridchanged})"
        autowidth="true"
        multirowselect=true
        delete="true"
```

```
        insert="true"
        title="Employee database"
        pagesize="25"
        >
        <cfgridcolumn name="EMP_ID" header="EMP_ID" select="false" display="false">
        <cfgridcolumn name="FIRSTNAME" header="First Name" select="true" />
        <cfgridcolumn name="LASTNAME" header="Last Name" select="true" />
        <cfgridcolumn name="DEPARTMENT" header="Department" select="true" />
        <cfgridcolumn name="EMAIL" header="Email" select="true" />
    </cfgrid>
    <br>
    <cfinput type="button" onClick="javascript:getAllSelectedRows('empListing',true)" name="getRows"
value="Get Selected Rows">
    <cfinput type="button" onClick="javascript:clearAllSelectedRows('empListing')" name="clearRows"
value="Clear Selected Rows">
    <cfinput type="button" onClick="javascript:deleteAllSelectedRows('empListing')" name="deleteRows"
value="Delete Selected Rows">
</cfform>
</body>
</html>
```

Employee.cfc

```
<cfcomponent>
    <cfscript>
        empQuery = new query(name="emps", datasource="cfdocexamples");
        remote any function
getEmployees(page,pageSize,gridsortcolumn="EMP_ID",gridsortdirection="ASC",empName)
        {
            var orderBy = "EMP_ID";
            var mysql = "SELECT Emp_ID, FirstName,  LastName, EMail, Department, Email FROM Employees";
            if(isdefined("arguments.empName") and trim(arguments.empName) neq ""){
                mysql = mysql & " WHERE " & "firstname = '#arguments.empName#'";
            }
            if(arguments.gridsortcolumn eq ""){
                mysql = mysql & " ORDER BY " & orderBy;
            }
            mysql = mysql & " " & gridsortdirection;
            return QueryConvertForGrid(empQuery.execute(sql=mysql).getResult(), page, pageSize);
        }
        remote void function editEmployees(gridaction,gridrow,gridchanged)
        {
            switch(gridaction)
            {
                case "I":
                {
                    var eFName = gridrow["FIRSTNAME"];
                    var eLNAme = gridrow["LASTNAME"];
                    var eDept = gridrow["DEPARTMENT"];
                    var eEmail = gridrow["EMAIL"];
                    var insertSql = "insert into Employees(FirstName,LastName,Department,Email) values ("
& "'" & eFName & "', '" & eLName & "', '" & eDept & "', '" & eEmail & "')";
                    empQuery.execute(sql=insertSql);
                    break;
                }
                case "U":
                {
                    var empId = gridrow["EMP_ID"];
                    var changedCol = structkeylist(gridchanged);
                    var updateSql =  "UPDATE Employees SET " & changedCol & "='" &  gridchanged[changedCol]
& "' WHERE emp_id=" & empId;
                    empQuery.execute(sql=updateSql);
                    break;
```

```
                }
                case "D":
                {
                    deleteEmployees(gridrow);
                }
            }
        }
        remote void function deleteEmployees(empdata)
        {
            var i = 1;
            var emp = {};
            if(isArray(empdata) and not ArrayIsEmpty(empdata)){
                for(emp in empdata){
                    if(isStruct(emp) and structkeyexists(emp,"emp_id")){
                        empid = emp["emp_id"];
                        writelog("deleting " & empid);
                        //var deleteSql =  "delete from Employees where emp_id=" & empid;
                        //empQuery.execute(sql=deleteSql);
                    }
                }
            }
        }

    </cfscript>
</cfcomponent>
```

この例では、multirowselect=true を設定して、複数のレコードを削除したりするバッチ操作をグリッドデータに対して実行しています。

この例はバッチ操作であるため、誤ってデータを削除しないように、deleteemployees 関数内の 2 行をコメントアウトしています。削除を確認する場合は、このコードのコメントアウトを外します。

フォームにある deleteAllSelectedRows ボタンは、外部からどのようにレコードを削除できるかを示します。つまり、グリッドに組み込まれた削除ボタンを使用せずに削除します。同じ方法を使用して、その他のバッチ操作 (複数ファイルの別フォルダへの移動など) やバッチ更新を実行することもできます。

注意：Employee.cfm の deleteAllSelectedRows メソッドに示すとおり、プロキシオブジェクトの httpMethod に POST を設定して、[Request URL Too Large] エラーが発生しないように注意してください。

ColdFusion.Map.show

説明
マップが非表示になっている場合は表示します。

関数のシンタックス
ColdFusion.Map.show(Id)

パラメータ
- Id: マップの名前です。

例

```
<script>
    function showMap(mapId)
      {
                ColdFusion.Map.show(mapId);
    }

    function hideMap(mapId)
      {
                ColdFusion.Map.hide(mapId);
    }
</script>
<a href="##" id="a1" onclick="return showMap('mainMap')">Show Map</a> | <a href="##" id="a1" onclick="return
hideMap('mainMap')">Hide Map</a>
<cfmap
      zoomlevel = "12"
    name = "mainMap"
    showcentermarker= "true"
    centeraddress = "The Key Learning centre, Oxford, UK"
      title="Venue Address"
      hideborder=false
      collapsible=true
      initShow=false/>
```

ColdFusion.Map.hide

説明

マップが表示されている場合は非表示にします。

関数のシンタックス

ColdFusion.Map.hide(Id)

パラメータ

- Id: マップの名前です。

例

ColdFusion.Map.show の例を参照してください。

ColdFusion.Map.refresh

説明

マップをリロードします。

関数のシンタックス

ColdFusion.Map.refresh (Id)

パラメータ

- Id: マップの名前です。

使用方法

マップが、非表示の折り畳み可能な spry パネルまたは div の中に埋め込まれている場合 (つまり、マップコンテナは表示されているが、実際のマップは非表示である場合)、この関数を使用して、マップを強制的に表示できます。

例

```
<script type="text/javascript" src="/CFIDE/scripts/ajax/spry/includes_minified/SpryCollapsiblePanel.js"
></script>
<link type="text/css" href="/CFIDE/scripts/ajax/spry/widgets/collapsiblepanel/SpryCollapsiblePanel.css"
rel="stylesheet">
<div id="cp" class="CollapsiblePanel" style="width:500px;">
      <div class="CollapsiblePanelTab" tabindex="0">SHOW MAP</div>
    <div class="CollapsiblePanelContent">
    <cfmap
            width="500"
            height="200"
            zoomlevel="12"
            name="mainMap"
            markercolor="333444"
            showscale="false"
            typecontrol="none"
            showcentermarker="true"
            centeraddress="The Key Learning centre, Oxford, UK"
            >
      </cfmap>
      </div>
</div>
<script type="text/javascript">
      var myTabClick = function()
      {
            !cpanel.isOpen() ? cpanel.open() : cpanel.close();
            cpanel.focus();
            ColdFusion.Map.refresh('mainMap');
      }
      var cpanel = new Spry.Widget.CollapsiblePanel("cp", {contentIsOpen:false});
      cpanel.onTabClick = myTabClick;
</script>
```

ColdFusion.Grid.getTopToolbar

説明

上部のツールバーを取得します。ツールバーは、アイコンやボタンなどのコントロールの追加に使用します。

関数のシンタックス

ColdFusion.getTopToolbar(Id)

パラメータ

- Id: グリッドの名前です。

例

ColdFusion.Grid.refreshBottomToolbar の例を参照してください。

ColdFusion.Grid.getBottomToolbar

説明
下部のツールバーを取得します。ツールバーは、アイコンやボタンなどのコントロールの追加に使用します。

関数のシンタックス
ColdFusion.Grid.getBottomToolbar(Id)

パラメータ
- Id: グリッドの名前です。

例
ColdFusion.Grid.refreshBottomToolbar の例を参照してください。

ColdFusion.Grid.showTopToolbar

説明
上部のツールバーを表示します。ツールバーは、アイコンやボタンなどのコントロールの追加に使用します。

関数のシンタックス
ColdFusion.Grid.showTopToolbar(Id)

パラメータ
- Id: グリッドの名前です。

例
ColdFusion.Grid.refreshBottomToolbar の例を参照してください。

ColdFusion.Grid.hideTopToolbar

説明
上部のツールバーを非表示にします。ツールバーは、アイコンやボタンなどのコントロールの追加に使用します。

関数のシンタックス
ColdFusion.Grid.hideTopToolbar(Id)

パラメータ
- Id: グリッドの名前です。

例
ColdFusion.Grid.refreshBottomToolbar の例を参照してください。

ColdFusion.Grid.showBottomToolbar

説明
下部のツールバーを表示します。ツールバーは、アイコンやボタンなどのコントロールの追加に使用します。

関数のシンタックス
ColdFusion.Grid.showBottomToolbar(Id)

パラメータ
- Id: グリッドの名前です。

例
ColdFusion.Grid.refreshBottomToolbar の例を参照してください。

ColdFusion.Grid.hideBottomToolbar

説明
下部のツールバーを非表示にします。ツールバーは、アイコンやボタンなどのコントロールの追加に使用します。

関数のシンタックス
ColdFusion.Grid.hideBottomToolbar(Id)

パラメータ
- Id: グリッドの名前です。

例
ColdFusion.Grid.refreshBottomToolbar の例を参照してください。

ColdFusion.Grid.refreshTopToolbar

説明
上部のツールバーを更新します。ツールバーは、アイコンやボタンなどのコントロールの追加に使用します。この関数は、内部で JavaScript 関数 ColdFusion.Grid.showTopToolbar を呼び出します。

関数のシンタックス
ColdFusion.Grid.refreshTopToolbar(Id)

パラメータ
- Id: グリッドの名前です。

例
ColdFusion.Grid.refreshBottomToolbar の例を参照してください。

ColdFusion.Grid.refreshBottomToolbar

説明

下部のツールバーを更新します。ツールバーは、アイコンやボタンなどのコントロールの追加に使用します。この関数は、内部で JavaScript 関数 ColdFusion.Grid.showBottomToolbar を呼び出します。

関数のシンタックス

ColdFusion.Grid.refresheBottomToolbar(Id)

パラメータ

- Id: グリッドコントロールの名前です。

例

grid.cfc

```
<cfcomponent>
      <cfscript>
          remote any function getEmployees(page,pageSize,gridsortcolumn="EMP_ID",gridsortdirection="ASC"){
                  var startRow = (page-1)*pageSize;
                  var endRow = page*pageSize;

                  if(!isdefined("arguments.gridsortcolumn") or isdefined("arguments.gridsortcolumn") and
trim(arguments.gridsortcolumn) eq "")
                        gridsortcolumn = "EMP_ID";
                 if(!isdefined("arguments.gridsortdirection") or isdefined("arguments.gridsortdirection")
and arguments.gridsortdirection eq "")
                        gridsortdirection = "ASC";
                  var mysql = "SELECT Emp_ID, FirstName,  EMail, Department FROM Employees";
                  if(isdefined("arguments.gridsortcolumn") and arguments.gridsortcolumn neq "")
                        mysql = mysql & " ORDER BY " & gridsortcolumn;
                  if(isdefined("arguments.gridsortdirection") and arguments.gridsortdirection neq "")
                        mysql = mysql & " " & gridsortdirection ;
                  rs1 = new query(name="team", datasource="cfdocexamples", sql=mysql).execute();
                        return QueryConvertForGrid(rs1.getResult(), page, pageSize);
            }


            remote any function editEmployees(gridaction,gridrow,gridchanged){
                        writelog("edit employee info");
            }

      </cfscript>
</cfcomponent>
```

grid.cfm

```
<script>
      var refreshToolbar = function(id,type){
         type == "top" ? ColdFusion.Grid.refreshTopToolbar(id) : ColdFusion.Grid.refreshBottomToolbar(id);
      }

      var hideToolbar = function(id,type){
            type == "top" ? ColdFusion.Grid.hideTopToolbar(id) : ColdFusion.Grid.hideBottomToolbar(id);
      }

      var showToolbar = function(id,type){
            (type == "top") ? ColdFusion.Grid.showTopToolbar(id) : ColdFusion.Grid.showBottomToolbar(id);
      }

      var handleToolbar = function(id,type){
            if(type == "top"){
                  tbar = ColdFusion.Grid.getTopToolbar(id);
                  tbar.addButton({
                        text: "Add User Account",
                        tooltip: "Add a user account",
                        handler: addUserAccount
                  });
            }
            else{
                  bbar = ColdFusion.Grid.getBottomToolbar(id);
                  bbar.add(new Ext.Toolbar.Separator());
                  bbar.addButton({
                        text: "Delete User Account",
                        tooltip: "Delete a user account",
                        handler: deleteUserAccount
                  });
            }
      }

      var GetUserInfo = function(){
            alert("Retrieving user account");
      }

      var addUserAccount = function(){
            alert("Adding new user account")
      }
      var deleteUserAccount = function(){
            alert("Deleting user account")
      }
</script>
<cfform>
      <br>
      <cfinput type="button" onClick="showToolbar('empGrid','top')" name="btn1" value="Show Top Toolbar">
     <cfinput type="button" onClick="handleToolbar('empGrid','top')" name="btn2" value="Add button to Top
Toolbar">
      <cfinput type="button" onClick="refreshToolbar('empGrid','top')" name="btn3" value="Refresh Top
Toolbar">
      <cfinput type="button" onClick="hideToolbar('empGrid','top')" name="btn4" value="Hide Top Toolbar">
      <br><br>

      <cfgrid
                  format="html"
                  name="empGrid"
                  width="800"
                  pagesize=5
                  sort=true
                  title="Employee database"
                  collapsible="true"
```

```
                insert="yes"
                delete="yes"

bind="cfc:grid.getEmployees({cfgridpage},{cfgridpagesize},{cfgridsortcolumn},{cfgridsortdirection})"
                onChange="cfc:grid.editEmployees({cfgridaction},{cfgridrow},{cfgridchanged})"
                selectMode="edit"
                >
                <cfgridcolumn name="Emp_ID" display=false header="ID" />
                <cfgridcolumn name="FirstName" display=true header="First Name"/>
                <cfgridcolumn name="Email" display=true header="Email"/>
                <cfgridcolumn name="Department" display=true header="Department" />
      </cfgrid>

      <br><br>
      <cfinput type="button" onClick="hideToolbar('empGrid','bottom')" name="btn5" value="Hide Bottom
Toolbar">
      <cfinput type="button" onClick="showToolbar('empGrid','bottom')" name="btn6" value="Show Bottom
Toolbar">
      <cfinput type="button" onClick="handleToolbar('empGrid','bottom')" name="btn7" value="Add button to
Bottom Toolbar">
     <cfinput type="button" onClick="refreshToolbar('empGrid','bottom')" name="btn8" value="Refresh Bottom
Toolbar">
</cfform>
```

第 6 章：CFC として実装されるスクリプト関数

スクリプト関数は ColdFusion 9 で追加された機能です。これらの関数は ColdFusion コンポーネントとして実装されます。これらの関数を使用すると、cfmail、cfpdf、cfquery、cfhttp、cfstoredproc、および cfftp タグに相当する機能を CFScript で利用できるようになります。

関連項目

1501 ページの「CFC として実装されるスクリプト関数」

関数へのアクセス

スクリプト関数は、"<ColdFusion のルートディレクトリ >¥CustomTags¥com¥adobe¥coldfusion" にあります。

ColdFusion Administrator に表示されるデフォルトのカスタムタグマッピング ([拡張機能]-[カスタムタグのパス] に表示されるカスタムタグのマッピング) は削除しないでください。

デフォルトの場所または Web ルートに配置されていれば、スクリプト関数は問題なく動作します。これらの関数が他の場所にある場合は、新しい場所 (C:¥com など) をポイントする /com マッピングを ColdFusion Administrator で追加する必要があります。

注意：サービスアクションで設定された属性の値 (mail.send(body="test mail") など) は一時的にのみ使用されます。アクションの完了後にはアクセスできません。暗黙的な getter を使用して属性にアクセスするとエラーになりますが、暗黙の setter または init メソッドの呼び出しを使用して設定した属性は保持されており、暗黙の getter を使用してアクセスできます。

関数一覧

次の表に、スクリプト関数とそれに相当する ColdFusion タグを示します。

関数	相当する ColdFusion タグ
ftp	cfftp
http	cfhttp
mail	cfmail
pdf	cfpdf
query	cfquery
storedproc	cfstoredproc

ftp

説明

CFScript で File Transfer Protocol (FTP) 操作を実装する場合に使用します。

シンタックス

モード	シンタックス
サービスの作成	new ftp() または createObject("component", "ftp")
属性の初期化	次のいずれかを使用します。 • ftpService=new ftp(*attribute-value pair*) • ftpService.setAttributes(**attribute-value pair**) • ftpService.setA*ttributeName*(attribute_value) • ftpService.*action_method*(*attribute-value_pair*)
サービスアクションの実行	ftpService.*action_method*(*attribute-value_pair*)

プロパティ

actionparam	buffersize	connection	passive
password	port	proxyserver	retrycount
server	stoponerror	timeout	username
fingerprint	key	passphrase	secure
ASCIIExtensionList	directory	existing	failifexists
item	localfile	name	new
remotefile	result	transfermode	allosize

cfftp タグでサポートされるすべての属性を、属性と値のペアとして使用できます。次に例を示します。

```
<cfftp userName="myUserName">
```

この属性は次のように使用できます。

```
ftpService.setUserName("myUserName");
```

詳細については、cfftp タグの「属性」セクションを参照してください。

関連項目

cfftp、関数一覧

履歴

ColdFusion 9: この関数が追加されました。

メソッド

次の FTP アクションをメソッドとして使用できます。引数とシンタックスはすべてのメソッドでほぼ共通です。

open	close	quote	site
allo	acct	changeDir	createDir
listDir	removeDir	getFile	putFile
rename	remove	getCurrentDir	getCurrentUrl
existDir	existsFile	exists	

説明	これらのメソッドはすべて、cfftp タグでサポートされる FTP アクションに対応しています。各メソッドの詳細については、cfftp タグの該当セクションを参照してください。

戻り値	すべてのメソッドは、次のプロパティが設定されたコンポーネントを返します。 • prefix: result 属性または cfftp スコープに相当します • result: action="listdir" の場合にのみ設定されます
シンタックス	ftpService.*methodName*(*attribute-value pair*)
引数	cfftp タグでサポートされるすべての属性。

- setAttributes

説明	ftp 関数の属性を設定します。
戻り値	なし
シンタックス	ftpService.*setAttributes* (*attribute-value pair*)
引数	cfftp タグでサポートされるすべての属性。

- getAttributes

説明	ftp 関数に対して設定された属性を取得します。
戻り値	すべてまたは一部の属性値を含む構造体を返します。
シンタックス	ftpService.get*Attributes* (*attributelist*)
引数	属性のカンマ区切りリスト。リストが指定されていない場合は、定義されているすべての属性が返されます。

- clear

説明	ftp 関数に対して追加されたすべての属性を削除します。
戻り値	なし
シンタックス	ftpService.clear()
引数	なし

使用方法

この関数は cfftp タグに対応します。詳細については、cfftp タグの「使用方法」セクションを参照してください。

例

```
<cfscript>
    /* Create a new ftp Service*/
    ftpService = new ftp();
    /* Set attributes using implicit setters */
    ftpService.setUsername("myUsername");
    ftpService.setPassword("myPassword");
    ftpService.setServer("myFtpServer");
    ftpService.setStopOnError("true");
    ftpService.setConnection("conn");
    /* Open connection to ftp server */
    WriteOutput("<h4>Open a connection</h4>");
    result = ftpService.open();
    WriteOutput("<p>Did it succeed? " & result.getPrefix().succeeded & "<br></p>");
    /* Get current directory */
    WriteOutput("<h4>Get current directory</h4>");
    result = ftpService.getcurrentdir();
    WriteOutput("<p>Current Directory: " & "'" & result.getPrefix().returnvalue & "'" & "<br></p>");
    /* List contents of the current directory */
    WriteOutput("<h4>List directory contents</h4>");
    result = ftpService.listdir(directory = "/",name="listDirs");
    displayListing(result.getResult());
    /* Move a file to the ftp server */
    WriteOutput("<h4>Move File to Remote Server</h4>");
    lFile = "C:\temp\artifacts.xml";
    rFile = "artifacts.xml";
    result = ftpService.putFile(transferMode="binary", localfile=lFile, remoteFile=rFile);
    WriteOutput("<p>Did it succeed? " & result.getPrefix().succeeded & "<br></p>");
    /* Close connection to the ftp server */
    WriteOutput("<h4>Close the connection</h4>");
    ftpService.close(connection="conn");
    WriteOutput("<p>Did it succeed? " & result.getPrefix().succeeded & "<br></p>");
</cfscript>
<cffunction name="displayListing" hint="display ftp files">
    <cfargument name="filesToList" required="true">
    <cftable query = "filesToList" HTMLTable = "Yes" colHeaders = "Yes" border="1" maxrows="10">
        <cfcol header = "<b>Name</b>" text = "#name#">
        <cfcol header = "<b>Path</b>" text = "#path#">
        <cfcol header = "<b>URL</b>" text = "#url#">
        <cfcol header = "<b>Length</b>" text = "#length#">
        <cfcol header = "<b>LastModified</b>"
         text = "#DateFormat(lastmodified)#">
        <cfcol header = "<b>IsDirectory</b>" text = "#isdirectory#">
    </cftable>
</cffunction>
```

http

説明

HTTP リクエストを生成し、サーバーからのレスポンスを処理する場合に、CFScript で使用します。

シンタックス

モード	シンタックス
サービスの作成	new http() または createObject("component", "http")
属性の初期化	次のいずれかを使用します。 • httpService=new http(**attribute-value_pair**) • httpService.setAttributes(**attribute-value_pair**) • httpService.set*AttributeName*(*attribute_value*) • httpService.send(*attribute-value_pair*)
サービスアクションの実行	httpService.send(**attribute-value_pair**)

プロパティ

url	charset	clientcert	clientcertpassword
columns	delimiter	file	firstrowasheaders
getasbinary	method	multipart	multiparttype
name	password	path	port
proxyserver	proxyport	proxyuser	proxypassword
redirect	resolveurl	result	textqualifier
throwonerror	timeout	useragent	username

cfhttp タグでサポートされるすべての属性を、属性と値のペアとして使用できます。次に例を示します。

```
<cfhttp name="onerow">
```

この属性は次のように使用できます。

```
httpService.setName("onerow");
```

属性の詳細については、cfhttp タグの「属性」セクションを参照してください。

関連項目

cfhttp、関数一覧

履歴

ColdFusion 9: この関数が追加されました。

メソッド

- addParam

説明	cfhttpparam タグを追加する場合に使用します。たとえば、CFScript で http POST 操作を指定できます。HTTP リクエストを作成するためのパラメータを指定します。
シンタックス	*httpService*.addParam(*attribute-value pair*)
戻り値	なし
引数	cfhttpparam タグでサポートされるすべての属性を、属性と値のペアとして使用できます。

- send

説明	HTTP リクエストを生成し、サーバーからのレスポンスを処理する場合に使用します。
戻り値	次のメソッドを呼び出せるコンポーネント。 • GetResult(): 名前属性が指定されている場合に、サーバーから返されたクエリーオブジェクトにアクセスします。 • GetPrefix():cfhttp スコープにアクセスします。これは cfhttp タグの result 属性に相当します。
シンタックス	*httpService*.send(*attribute_value pair*)
引数	cfhttpparam タグでサポートされるすべての属性。

- setAttributes

説明	http 関数の属性を設定します。
戻り値	なし
シンタックス	httpService.*setAttributes* (*attribute-value pair*)
引数	cfhttp タグでサポートされるすべての引数。

- getAttributes

説明	http 関数に対して設定された属性を取得します。
戻り値	すべてまたは一部のサービスタグ属性値を含む構造体を返します。
シンタックス	httpService.get*Attributes*(*attribute_list*)
引数	属性のカンマ区切りリスト。リストが指定されていない場合は、定義されているすべての属性が返されます。

- clearAttributes

説明	http 関数に対して追加されたすべての属性を削除します。
戻り値	なし
シンタックス	*httpService*.clearAttributes(*attribute_list*)
引数	属性のカンマ区切りリスト。

- clearParams

説明	addParam メソッドによって追加された cfhttpparam タグを削除します。
戻り値	なし
シンタックス	httpService.clearParams()
引数	なし

- clear

説明	addParam メソッドを使用して追加されたすべての属性と cfhttpparam タグを削除します。
戻り値	なし
シンタックス	httpService.clear()
引数	なし

使用方法

この関数は cfhttp タグに対応します。使用方法の詳細については、『CFML リファレンス』の cfhttp の「使用方法」セクションを参照してください。

例

```
<!--- Get Video --->
<!---
<cfset videoName = "<video path>\hello.wmv">
<cfset videoFileName = "hello.wmv">
--->
<!--- Set User Account Data --->
<!---
<cfset clientKey = "enter client key from google"/>
<cfset devKey = "ebtdev key from google>"/>
--->
<cfscript>
    /* youtube uplaod url */
    youTubeUploadURL = "http://uploads.gdata.youtube.com/feeds/api/users/default/uploads";
    /* video to upload */
    videoName = ExpandPath('./hello.wmv');
    videoFileName = "hello.wmv";
    /* set user account data */
    clientKey = "enter client key from google";
    devKey = "ewnter dev key from google";
    /* create new http service */
    httpService = new http();
    /* set attributes using implicit setters */
    httpService.setMethod("post");
    httpService.setCharset("utf-8");
    httpService.setUrl("https://www.google.com/accounts/ClientLogin");
    /* add httpparams using addParam() */
    httpService.addParam(type="formfield",name="accountType",value="HOSTED_OR_GOOGLE");
    httpService.addParam(type="formfield",name="Email",value="enter gmail id");
    httpService.addParam(type="formfield",name="Passwd",value="enter password");
    httpService.addParam(type="formfield",name="service",value="youtube");
    httpService.addParam(type="formfield",name="source",value="youtubecode");
    /* make the http call to the URL using send() */
    result = httpService.send().getPrefix();
    /* process the filecontent returned */
    content = listtoarray(result.filecontent,chr(10));
    for(i=1;i lte arraylen(content);i++)
    {
        item = content[i];
        authdata[listFirst(item, "=")] = listRest(item, "=");
    }
</cfscript>
<!--- Create ATOM XML and save to a file to be sent with video --->
<cfsavecontent variable="meta">
    <cfoutput>
    <entry xmlns="http://www.w3.org/2005/Atom"
        xmlns:media="http://search.yahoo.com/mrss/"
        xmlns:yt="http://gdata.youtube.com/schemas/2007">
        <media:group>
        <media:title type="plain">WithOutQuotes</media:title>
        <media:description type="plain">Test Description</media:description>
        <media:category
            scheme="http://gdata.youtube.com/schemas/2007/categories.cat">People
        </media:category>
        <media:keywords>yourvideo</media:keywords>
        </media:group>
    </entry>
```

```
    </cfoutput>
</cfsavecontent>
<cfscript>
    tmpfile = expandPath("./meta.xml");
    FileWrite(tmpfile,trim(meta));
    /* use the httpService created above  */
    httpService.setUrl("http://uploads.gdata.youtube.com/feeds/api/users/default/uploads");
    httpService.setTimeOut(450);
    httpService.setMultipartType("related");
    /* clear params first */
    httpService.clearParams();
    /* add httpparams using addParam() */
    httpService.addParam(type="header",name="Authorization", value="GoogleLogin auth=#authdata.auth#");
    httpService.addParam(type="header",name="X-GData-Client", value="#variables.clientkey#");
    httpService.addParam(type="header",name="X-GData-Key", value="key=#variables.devkey#");
    httpService.addParam(type="header",name="Slug",value="#videoFileName#");

httpService.addParam(type="file",name="API_XML_Request",file="#tmpfile#",mimetype="application/atom+xml")
;
    httpService.addParam(type="file",name="file",file="#videoName#",mimetype="video/*");
    /* make the http call to the URL using send() */
    result = httpService.send().getPrefix();
    if(result.statuscode contains "201")
    {
        WriteOutput("Your video has been successfully uploaded to YouTube");
    }
    else
    {
        WriteOutput("There was a problem uploading the video. Status code returned was " &
result.statuscode);
    }
</cfscript>
```

mail

説明

SMTP サーバーを使用して電子メールメッセージを送信する場合に使用します。オプションで、メッセージにクエリー出力を追加することもできます。

シンタックス

モード	シンタックス
サービスの作成	new mail() または createObject("component", "mail")
属性の初期化	次のいずれかを使用します。 • mailService=new mail(**attribute-value_pair**) • mailService.setAttributes(**attribute-value_pair**) • mailService.set*AttributeName*(*attribute_value*) • mailService.send(*attribute-value_pair*)
サービスアクションの実行	mailService.send(**attribute-value_pair**)

プロパティ

from	to	subject	bcc
cc	charset	debug	failto
group	groupcasesensitive	mailerid	maxrows
mimeattach	password	port	priority
query	replyto	server	spoolenable
startrow	timeout	type	username
useSSL	useTLS	wraptext	remove
body			

cfmail タグでサポートされるすべての属性を、属性と値のペアとして使用できます。次に例を示します。

```
<cfmail from="#form.mailFrom#">
```

この属性は次のように使用できます。

```
mailerService.setFrom(form.mailFrom);
```

関連項目

cfmail、関数一覧

履歴

ColdFusion 9: この関数が追加されました。

メソッド

- addParam

説明	cfmailparam タグを追加する場合に使用します。たとえば、電子メールメッセージにファイルを添付したり、ヘッダーを追加できます。
シンタックス	*mailService*.addParam(*attribute-value pair*)
戻り値	なし
引数	cfmailparam タグでサポートされるすべての属性を、属性と値のペアとして使用できます。

- addPart

説明	cfmailpart タグを追加する場合に使用します。たとえば、マルチパート電子メールメッセージの 1 つのパートを追加できます。
シンタックス	*mailService*.addPart(*attribute-value pair*)
戻り値	なし
引数	cfmailpart タグでサポートされるすべての属性を、属性と値のペアとして使用できます。

- send

説明	メールサービスを呼び出して電子メールメッセージを送信する場合に使用します。
戻り値	なし
シンタックス	*mailService*.send(*attribute-value pair*)
引数	cfmail タグでサポートされるすべての属性。

- setAttributes

説明	mail 関数の属性を設定します。

戻り値	なし
シンタックス	mailService.*setAttributes* (*attribute-value pair*)
引数	cfmail タグでサポートされるすべての属性。

- getAttributes

説明	mail 関数に対して設定された属性を取得します。
戻り値	すべてまたは一部の属性値を含む構造体を返します。
シンタックス	mailService.get*Attributes* (*attributelist*)
引数	属性のカンマ区切りリスト。リストが指定されていない場合は、定義されているすべての属性が返されます。

- clearAttributes

説明	mail 関数に対して追加されたすべての属性を削除します。
戻り値	なし
シンタックス	*mailService*.clearAttributes(*attribute_list*)
引数	属性のカンマ区切りリスト。

- clearParams

説明	addParam メソッドによって追加された cfmailparam タグを削除します。
戻り値	なし
シンタックス	mailService.clearParams()
引数	なし

- clearParts

説明	addPart メソッドによって追加された cfmailpart タグを削除します。
戻り値	なし
シンタックス	mailService.clearProcResults()
引数	なし

- clear

説明	addParam および addPart. メソッドによって追加されたすべての属性、cfmailparam タグ、および cfmailpart タグを削除します。
戻り値	なし
シンタックス	mailService.clear()
引数	なし

使用方法

この関数は cfmail タグに対応します。使用方法の詳細については、cfmail タグの「使用方法」セクションを参照してください。

例

```
<h3>Sending mail in cfscript</h3>
<cfscript>
    /* create mailer service */
    mailerService = new mail();
    if(IsDefined("form.mailto"))
    {
    if(form.mailto is not "" AND form.mailfrom is not "" AND form.Subject is not "" and form.attachment is
not "")
    {
        savecontent variable="mailBody"{
        WriteOutput("This message was sent by an automatic mailer built with cfmail:= = = = = = = = = = =
= = = = = = = = = = = = = = = = = =" & "<br><br>" & form.body);
             }
        /* set mail attributes using implicit setters provided */
        mailerService.setTo(form.mailto);
        mailerService.setFrom(form.mailFrom);
        mailerService.setSubject(form.subject);
        mailerService.setType("html");
        /* add mailparams */
        mailerService.addParam(file=expandpath(form.attachment),type="text/plain",remove=false);
        /* send mail using send(). Attribute values specified in an end action like "send" will not persist
after the action is performed */
            mailerService.send(body=mailBody);
            writeoutput("<h3>Thank you</h3>" & "<p>Thank you, " & mailfrom & "<br>" & "Your message, " &
subject & ", has been sent to " & mailto & "</p>");
    }
    }
</cfscript>
<p>
<form action = "mail1.cfm" method="POST">
    <table>
    <tr>
    <td>TO</td>
    <td><input type = "Text" name = "MailTo"></td>
    </tr>
    <tr>
    <td>FROM</td>
    <td><input type = "Text" name = "MailFrom"></td>
    </tr>
    <tr>
    <td>SUBJECT</td>
    <td><input type = "Text" name = "Subject"></td>
    </tr>
    <tr>
    <td>ATTACHMENT</td>
    <td><input type = "file" name = "attachment"></td>
    </tr>
    </table>
    <hr>
    MESSAGE BODY:
    <br>
    <textarea name ="body" cols="40" rows="5" wrap="virtual"></textarea>
    <!--- Establish required fields. --->
    <input type = "hidden" name = "MailTo_required" value = "You must enter a recipient">
    <input type = "hidden" name = "MailFrom_required" value = "You must enter a sender">
    <input type = "hidden" name = "Subject_required" value = "You must enter a subject">
    <input type = "hidden" name = "Body_required" value = "You must enter some text">
    <input type = "hidden" name = "attachment_required" value = "You must select a file">
    <p><input type = "Submit" name = ""></p>
</p>
</form>
```

pdf

説明

既存の PDF ドキュメントを CFScript で操作するためのサービスを提供します。

シンタックス

モード	シンタックス
サービスの作成	new pdf() または createObject("component", "pdf")
属性の初期化	次のいずれかを使用します。 • pdfService=new pdf(*attribute-value pair*) • pdfService.setAttributes(*attribute-value pair*) • pdfService.set*AttributeName(attribute_value*) • pdfService.*action_method*(*attribute-value pair*)
サービスアクションの実行	pdfService.*action_method*(*attribute-value pair*)

プロパティ

addQuads	algo	align	ascending
bottomMargin	compressTiffs	copyFrom	ddxfile
destination	directory	encodeAll	encrypt
flatten	foreground	format	height
hires	honourSpaces	hScale	image
imagePrefix	info	inputFiles	isBase64
jpgDpi	keepBookmark	leftMargin	maxBreadth
maxLength	maxScale	name	newOwnerPassword
newUserPassword	noAttachments	noBookmarks	noComments
noJavascripts	noLinks	noMetadata	noThumbnails
numberFormat	opacity	order	outputFiles
overridePage	overwrite	package	pages
password	permissions	position	resolution
rightMargin	rotation	saveOption	scale
showOnPrint	source	stopOnError	text
topMargin	transparent	type	useStructure
version	vscale	width	

cfpdf タグでサポートされるすべての属性を、属性と値のペアとして使用できます。次に例を示します。

```
<cfpdf action="getInfo" source="myBook.pdf" name="PDFInfo">
```

この属性は次のように使用できます。

```
pdfInfo = pdfService.getPdfInfo(source="myBook.pdf", name="pdfinfo");
```

詳細については、cfpdf タグの「属性」セクションを参照してください。

メソッド

- addParam

説明	cfpdfparam タグを追加する場合に CFScript で使用します。action="merge" である場合にのみ使用できます。
戻り値	なし
シンタックス	pdfService.addParam(attribute-value pair)
引数	cfpdfparam タグでサポートされるすべての属性を、属性と値のペアとして使用できます。

- 次の PDF アクションをメソッドとして使用できます。引数とシンタックスはすべてのメソッドでほぼ共通です。

addWatermark	removeWatermark	deletePages	getPDFInfo
setPDFInfo	merge	processDDX	protect
read	write	thumbnail	transform
optimize	extractImage	extractText	addHeader
addFooter	removeHeaderFooter		

注意：リストにある setPDFInfo と getPDFInfo のアクションは、cfpdf のアクションと異なります。それぞれ、cfpdf action="setinfo" および cfpdf action="getinfo" に対応します。

説明	これらのメソッドはすべて、cfpdf タグで指定される PDF アクションに対応しています。各メソッドの詳細については、cfpdf の該当セクションを参照してください。
戻り値	アクションによって異なります。name 属性が指定されている場合は、PDF 操作の結果が返されます。その以外の場合は、空の文字列が返されます。 たとえば、次のコードを使用すると "book.pdf" の PDF 情報を含む構造体が返されます。 pdfinfo = pdfService.getPDFInfo(source="book.pdf",name="var") PDF の操作には cfpdf タグが使用されます。そのため、name 属性を指定する必要があります。 "var" はページ変数スコープに存在しないので、"var" に直接アクセスすることはできません。
シンタックス	*serviceName.methodName*(*attribute-value pair*)
引数	指定されたアクションに対して cfpdf タグでサポートされるすべての属性がサポートされます。

- setAttributes

説明	pdf 関数の属性を設定します。
戻り値	なし
シンタックス	pdfService.*setAttributes* (*attribute-value pair*)
引数	cfpdf タグでサポートされるすべての属性。

- getAttributes

説明	pdf 関数に対して設定された属性を取得します。
戻り値	すべてまたは一部の属性値を含む構造体を返します。
シンタックス	pdfService.get*Attributes* (*attributelist*)
引数	属性のカンマ区切りリスト。リストが指定されていない場合は、定義されているすべての属性が返されます。

- clearAttributes

説明	pdf 関数に対して追加されたすべての属性を削除します。
戻り値	なし

シンタックス	*pdfService*.clearAttributes(*attribute_list*)
引数	削除する属性のカンマ区切りリスト。

- clearParams

説明	addParam メソッドによって追加されたパラメータを削除します。
戻り値	なし
シンタックス	pdfService.clearParams()
引数	なし

- clear

説明	addParam メソッドによって追加されたすべての属性とパラメータを削除します。
戻り値	なし
シンタックス	pdfService.clear()
引数	なし

関連項目
cfpdf、関数一覧

履歴
ColdFusion 9: この関数が追加されました。

使用方法
この関数は cfpdf タグに対応します。使用方法の詳細については、cfpdf の「使用方法」セクションを参照してください。

例

```
<h3>PDF Thumbnail</h3>
<cfscript>
    // Create a variable for the name of the PDF document.
    mypdf = "book";
    thumbnailsDirectory = ExpandPath(".") & "\" & "#mypdf#_thumbnails";
    //create new PDF service
    pdfService = new pdf();
    //set attributes using implicit setters
    pdfService.setSource(expandpath('./#mypdf#.pdf'));
    //Use the getPdfInfo action to retrieve the total page count for the PDF document.
    PDFInfo = pdfService.getPdfInfo(name="pdfinfo");
    pageCount = PDFInfo.TotalPages;
    WriteOutput("pageCount=" & pageCount);
    //Generate a thumbnail image for each page in the PDF source document,
    //create a directory (if it does not exist) in the web root that is
    //a concatenation of the PDF source name and the word "thumbnails", and
    //save the thumbnail images in that directory.
    pdfService.thumbnail(destination=thumbnailsDirectory, scale=60, overwrite=true);
    //Loop through the images in the thumbnail directory and generate a link
    //from each image to the corresponding page in the PDF document.
    for(i="1";i lte pageCount;i++)
    {
        //Click the thumbnail image to navigate to the page in the PDF document.
        WriteOutput("<a href='#mypdf#.pdf##page=#i#' target='_blank'><img
src='#mypdf#_thumbnails/#mypdf#_page_#i#.jpg'></a>");
    }
</cfscript
```

query

説明

CFScript を使用してデータソースに SQL ステートメントを渡すクエリーを実行する場合に使用します。

シンタックス

モード	シンタックス
サービスの作成	new query() または createObject("component", "query")
属性の初期化	次のいずれかを使用します。 • queryService=new query(*attribute-value_pair*) • queryService.setAttributes(*attribute-value_pair*) • queryService.set*AttributeName*(*attribute_value*) • queryService.execute(*attribute-value_pair*)
サービスアクションの実行	queryService.execute(*attribute-value_pair*)

プロパティ

name	blockfactor	cachedafter	cachedwithin
dataSource	dbtype	debug	maxRows
password	result	timeout	username
sql			

cfquery タグでサポートされるすべての属性を、属性と値のペアとして使用できます。次に例を示します。

```
<cfquery Name="myName"> </cfquery>
```

この属性は次のように使用できます。

```
queryService.setName("myName");
```

関連項目

cfquery、関数一覧

履歴

ColdFusion 9: この関数が追加されました。

メソッド

- addParam

説明	cfqueryparam タグを追加して次のタスクを実行する場合に CFScript で使用します。 • クエリーパラメータのデータ型を確認する • バインド変数をサポートする DBMS を使用する場合に、ColdFusion が SQL ステートメントでバインド変数を処理できるようにする
シンタックス	*serviceName*.addParam(*attribute-value pair*)
戻り値	なし
引数	cfqueryparam タグでサポートされるすべての属性を、属性と値のペアとして使用できます。

- execute

説明	SQL ステートメントを実行する場合に使用します。
戻り値	次のプロパティが設定されたコンポーネント。 • Result: 結果セットを返す SQL クエリーの場合 ("SELECT" SQL クエリーなど)。 • Prefix: cfquery タグの result 属性に相当します。
シンタックス	*queryService*.execute(*attribute-value pair*)
引数	cfquery タグでサポートされるすべての属性。

- setAttributes

説明	query 関数の属性を設定します。
戻り値	なし
シンタックス	queryService.*setAttributes* (*attribute-value pair*)
引数	cfquery タグでサポートされるすべての属性。

- getAttributes

説明	query 関数に対して設定された属性を取得します。
戻り値	すべてまたは一部の属性値を含む構造体を返します。
シンタックス	queryService.get*Attributes* (*attributelist*)
引数	属性のカンマ区切りリスト。リストが指定されていない場合は、定義されているすべての属性が返されます。

- clearAttributes

説明	query 関数に対して追加されたすべての属性を削除します。
戻り値	なし
シンタックス	*queryService*.clearAttributes(*attribute_list*)
引数	属性のカンマ区切りリスト。

- clearParams

説明	addParam メソッドによって追加された queryparams を削除します。
戻り値	なし
シンタックス	queryService.clearParams()
引数	なし

- clear

説明	addParam メソッドによって追加されたすべての属性と queryparms を削除します。
戻り値	なし
シンタックス	queryService.clear()
引数	なし

使用方法

この関数は cfquery タグに対応します。使用方法については、cfquery の「使用方法」を参照してください。

例

```
<cfscript>
    /*
    This example shows how to create a query service in cfscript, set/get attributes using implict
setters/getters, and also
    how to execute the query and access the resultset
    */
    param MaxRows="10";
    param StartRow="1";
    /*
    Query database for information if cached database information has
    not been updated in the last six hours; otherwise, use cached data.
    */
    /* create a quey service */
    queryService = new query();
    /* set properties using implict setters */
    queryService.setDatasource("cfdocexamples");
    queryService.setName("GetParks");
    queryService.setcachedwithin(CreateTimeSpan(0, 6, 0, 0));
    /* Add sql queryparams using named and positional notation */
    queryService.addParam(name="state",value="MD",cfsqltype="cf_sql_varchar");
    queryService.addParam(value="National Capital Region",cfsqltype="cf_sql_varchar");
    /* invoke execute() on the query object to execute the query and return a component with properties
result and prefix (which can be accessed as implcit getters) */
    result = queryService.execute(sql="SELECT PARKNAME, REGION, STATE FROM Parks WHERE STATE = :state and
REGION = ? ORDER BY ParkName, State ");
    GetParks = result.getResult();
    /* getPrefix() returns information like recordcount,sql etc (typically whatever one gets if one uses
the result attribute of the cfquery tag */
    metaInfo = result.getPrefix();
</cfscript>
<cfoutput>
<h4>Found #metaInfo.recordcount# records for '#metainfo.sqlparameters[2]#' in the state
'#metainfo.sqlparameters[1]#' </h4>
</cfoutput>
<!--- Build HTML table to display query. ------------------------->
<table cellpadding="1" cellspacing="1">
    <tr>
        <td bgcolor="f0f0f0">
             
        </td>
        <td bgcolor="f0f0f0">
            <b><i>Park Name</i></b>
        </td>
        <td bgcolor="f0f0f0">
            <b><i>Region</i></b>
        </td>
        <td bgcolor="f0f0f0">
            <b><i>State</i></b>
        </td>
    </tr>
<!--- Output the query and define the startrow and maxrows parameters.
Use the query variable CurrentCount to keep track of the row you are displaying. ------>
<cfoutput query="GetParks" startrow="#StartRow#" maxrows="#MaxRows#">
    <tr>
        <td valign="top" bgcolor="ffffed">
            <b>#GetParks.CurrentRow#</b>
        </td>
        <td valign="top">
            <font size="-1">#ParkName#</font>
```

```
        </td>
        <td valign="top">
            <font size="-1">#Region#</font>
        </td>
        <td valign="top">
            <font size="-1">#State#</font>
        </td>
    </tr>
</cfoutput>
<!--- If the total number of records is less than or equal to the total number of rows,
then offer a link to the same page, with the startrow value incremented by maxrows
(in the case of this example, incremented by 10). --------->
    <tr>
        <td colspan="4">
        <cfif (StartRow + MaxRows) LTE GetParks.RecordCount>
            <cfoutput><a href="#CGI.SCRIPT_NAME#?startrow=#Evaluate(StartRow + MaxRows)#">
            See next #MaxRows# rows</a></cfoutput>
        </cfif>
        </td>
    </tr>
```

storedproc

説明

CFScript を使用してサーバーデータベースでストアドプロシージャを実行する場合に使用します。このタグを使用すると、データベース接続情報やストアドプロシージャを指定できます。

シンタックス

モード	**シンタックス**
サービスの作成	new storedProc() または createObject("component", "storedproc")
属性の初期化	次のいずれかを使用します。 • storedProcService=new storedproc(*attribute-value_pair*) • storedprocService.setAttributes(*attribute-value_pair*) • storedProcService.set*AttributeName*(*attribute_value*) • storedProcService.execute(*attribute-value_pair*)
サービスアクションの実行	storedProcService.execute(**attribute-value_pair**)

プロパティ

datasource	procedure	debug	cachedafter
cachedwithin	blockfactor	password	result
returncode	username		

cfstoredproc タグでサポートされるすべての属性が、属性と値のペアとしてサポートされます。次に例を示します。

```
<cfstoredproc procedure= "sp_proc">
```

この属性は次のように使用できます。

```
spService.setProcedure("sp_proc");
```

cfstoredproc タグの属性の詳細については、cfstoredproc の「属性」セクションを参照してください。

関連項目

cfstoredproc、関数一覧

履歴

ColdFusion 9: この関数が追加されました。

メソッド

- addParam

説明	cfprocparam タグを追加する場合に使用します。
シンタックス	*storedprocService*.addParam(*attribute-value pair*)
戻り値	なし
引数	cfprocparam タグでサポートされるすべての属性を、属性と値のペアとして使用できます。

- addProcResult

説明	cfprocresult タグを追加して、ストアドプロシージャによって返される結果セットにクエリーオブジェクトを関連付ける場合に使用します。
シンタックス	*storedprocService*.addProcResult(*attribute-value pair*)
戻り値	なし
引数	cfprocresult タグでサポートされるすべての属性を、属性と値のペアとして使用できます。

- execute

説明	ストアドプロシージャを実行する場合に使用します。
戻り値	次のメソッドを呼び出せるコンポーネント。 • getProcResultSets(): プロシージャによって返される結果セットにアクセスします。 • getProcOutVariables(): プロシージャによって返される OUT または INOUT 変数にアクセスします。
シンタックス	storedprocService.execute(attribute-value pair)
引数	cfstoredproc タグでサポートされるすべての属性。

- setAttributes

説明	storedproc 関数の属性を設定します。
戻り値	なし
シンタックス	storedProcService.*setAttributes* (*attribute-value pair*)
引数	cfstoredproc タグでサポートされるすべての属性。

- getAttributes

説明	storedproc 関数に対して設定された属性を取得します。
戻り値	すべてまたは一部の属性値を含む構造体を返します。

シンタックス	storedProcService.get*Attributes* (*attributelist*)
引数	属性のカンマ区切りリスト。リストが指定されていない場合は、定義されているすべての属性が返されます。

- clearAttributes

説明	storedProc 関数に対して追加されたすべての属性を削除します。
戻り値	なし
シンタックス	*storedProcService*.clearAttributes(*attribute_list*)
引数	属性のカンマ区切りリスト。

- clearParams

説明	addParam メソッドによって追加された cfprocparam タグを削除します。
戻り値	なし
シンタックス	storedProcService.clearParams()
引数	なし

- clearProcResults

説明	addProcResults メソッドによって追加された cfprocresult タグを削除します。
戻り値	なし
シンタックス	storedProcService.clearProcResults()
引数	なし

- clear

説明	addProcResults および addParam メソッドによって追加されたすべての属性とパラメータを削除します。
戻り値	なし
シンタックス	storedProcService.clear()
引数	なし

使用方法

この関数は cfstoredproc タグに対応します。使用方法の詳細については、cfstoredproc の「使用方法」セクションを参照してください。

例

```
<cfscript>
//If submitting a new book, insert the record and display confirmation
if(isDefined("form.title"))
{
    //create a new storedproc service
    spService = new storedproc();
    //set attributes using implicit setters
    spService.setDatasource("books");
    spService.setProcedure("Insert_Book");
    //add procparams using addParam
    spService.addParam(cfsqltype="cf_sql_varchar", type="in",value=form.title);
    spService.addParam(cfsqltype="cf_sql_numeric",type="in",value=form.price);
    spService.addParam(cfsqltype="cf_sql_date", type="in",value=form.publishDate);
    spService.addParam(cfsqltype="cf_sql_numeric",type="out",variable="bookId");
    //add procresults using addProcResult
    spService.addProcResult(name="rs1",resultset=1);
    //execute the stored procedure
    result = spService.execute();
    //getprocOutVariables() returns any OUT or INOUT varibles added using addParams()
    bookId = result.getprocOutVariables().bookId;
    //getProcResultSets() returns resultsets added using addProcresult()
    listOfBooks = result.getProcResultSets().rs1;
    WriteOutput("<h3>List of Books</h3>");
    writeDump(listOfBooks);
    //output data
    WriteOutput("<h3>" & "'" & form.title & "'" & " inserted into database. The ID is " & bookId & ".</h3>");
}
</cfscript>
<cfform action="#CGI.SCRIPT_NAME#" method="post">
    <h3>Insert a new book</h3>
    <table>
    <tr>
    <td>Title:</td>
    <td><cfinput type="text" size="20" required="yes" name="title"/></td>
    </tr>
    <tr>
    <td>Price:</td>
    <td><cfinput type="text" size="20" required="yes" name="price" validate="float" /></td>
    </tr>
    <tr>
    <td>Publish Date:</td>
    <td>
    <cfinput type="datefield" name="publishdate" mask="mm/dd/yyyy" size="20" ></td>
    </tr>
    </table>
    <input type="submit" value="Insert Book"/>
</cfform>
```

CFC として実装されるスクリプト関数

関数一覧

次の表に、スクリプト関数とそれに相当する ColdFusion タグを示します。

関数	相当する ColdFusion タグ
dbinfo	cfdbinfo
imap	cfimap
pop	cfpop
ldap	cfldap
feed	cffeed

dbinfo

説明

CFScript でデータソースに関する情報を取得する場合に使用します。この情報には、データベースの詳細、テーブル、クエリー、プロシージャ、外部キー、インデックスの情報や、データベース、ドライバ、JDBC のバージョン情報などが含まれます。

シンタックス

モード	シンタックス

サービスの作成	new dbinfo();
	または
	createObject("component", "dbinfo");

サービスアクションの実行	dbinfoService.*action_method*(*attribute-value_pair*);
属性の初期化	1512 ページの「属性の初期化」を参照してください。
CFC のプロパティの取得	1512 ページの「CFC のプロパティの取得」を参照してください。
返されたデータの使用	data=dbinfoService.*action_method*(*attribute-value_pair*); writedump(data);

プロパティ

datasource	dbname	name	password
pattern	table	username	

cfdbinfo タグでサポートされるすべての属性を、属性と値のペアとして使用できます。次に例を示します。

```
<cfdbinfo userName="myUserName">
```

この属性は次のように使用できます。

```
dbinfoService.setUserName("myUserName");
```

詳細については、『ColdFusion 9 CFML リファレンス』の cfdbinfo タグの「属性」セクションを参照してください。

関連項目

1501 ページの「関数一覧」

履歴

ColdFusion 9 アップデート 1: この関数が追加されました。

メソッド

次の dbinfo タイプをメソッドとして使用できます。引数とシンタックスはすべてのメソッドでほぼ共通です。

dbnames	tables	columns	version
procedures	foriegnkeys	index	

説明	これらのメソッドはすべて、cfdbinfo タグでサポートされる情報のタイプに対応しています。各メソッドの詳細については、『ColdFusion 9 CFML リファレンス』の cfdbinfo タグの該当セクションを参照してください。
戻り値	すべてのメソッドは、クエリーオブジェクトを返します。
シンタックス	dbinfoService.*methodName*(*attribute-value pair*);
引数	cfdbinfo タグでサポートされるすべての属性。

- setAttributes、getAttributes、clearAttributes、clear、setProperties、getProperties および clearProperties の詳細については、1511 ページの「すべての関数で共通のメソッド」を参照してください。

使用方法

この関数は cfdbinfo タグに対応します。詳細については、『ColdFusion 9 CFML リファレンス』の cfdbinfo タグの「使用方法」セクションを参照してください。

例

```
<cfscript>

    d = new dbinfo(datasource=" cfartgallery ").dbnames(datasource="ajax");
    writedump(d);
    d = new dbinfo(datasource=" ajax").dbnames();
    writedump(d);
</cfscript>
```

imap

説明

CFScript で、複数のフォルダ内のメールを取得および管理するように IMAP サーバーに問い合わせる場合に使用します。

シンタックス

モード	**シンタックス**

サービスの作成	new imap();
	または
	createObject("component", "imap");

属性の初期化	1512 ページの「属性の初期化」を参照してください。
サービスアクションの実行	imapService.methodName(**attribute-value_pair**)
CFC のプロパティの取得	1512 ページの「CFC のプロパティの取得」を参照してください。
返されたデータの使用	imapResult=imapService.*action_method*(**attribute-value_pair**);

プロパティ

attachmentpath	connection	folder	generateuniquefilenames
maxrows	messagenumber	name	newfolder
password	port	recurse	secure
server	startrow	stoponerror	timeout
uid	username		

cfimap タグでサポートされるすべての属性を、属性と値のペアとして使用できます。次に例を示します。

```
<cfimap action="open" connection = "myconnection">
```

この属性は次のように使用できます。

```
imapService = new
imap(server="myimpaserver",username="myusername",password="mypassword",port="myport",secure="yes");
imapService.open();
```

注意： 初期化のとき、または open メソッドを呼び出すときのいずれかで、server や username、password、port、secure などの接続プロパティを指定すると、暗黙的に接続が作成されます。そのため、それ以降のアクションではプロパティを指定する必要はありません。サンドボックスセキュリティをオンにしている場合、attachmentPath プロパティで参照するディレクトリには、必要な権限が与えられている必要があります。デフォルトでは、テンポラリディレクトリが使用されます。

属性の詳細については、『ColdFusion 9 CFML リファレンス』の cfimap タグの「属性」セクションを参照してください。

関連項目

1501 ページの「関数一覧」

履歴

ColdFusion 9 アップデート 1: この関数が追加されました。

メソッド

次の imap アクションをメソッドとして使用できます。引数とシンタックスはすべてのメソッドでほぼ共通です。

getAll	delete	open	close
markRead	createFolder	deleteFolder	renameFolder
listAllFolders	moveMail	getHeaderOnly	

説明	これらのメソッドはすべて、cfimap タグでサポートされる情報のタイプに対応しています。各メソッドの詳細については、『ColdFusion 9 CFML リファレンス』の cfimap の該当セクションを参照してください。
戻り値	getAll、getHeaderOnly および listAllFolders メソッドの場合は、クエリーオブジェクト。それ以外の場合は、何も返されません。
シンタックス	imapService.*methodName*(*attribute-value pair*);
引数	cfimap タグでサポートされるすべての属性。

- setAttributes、getAttributes、clearAttributes、clear、setProperties、getProperties および clearProperties の詳細については、1511 ページの「すべての関数で共通のメソッド」を参照してください。

使用方法

この関数は cfimap タグに対応します。詳細については、『ColdFusion 9 CFML リファレンス』の cfimap の「使用方法」セクションを参照してください。

例

```
<cfscript>

    m = new imap();
    m.setAttributes(server="#REQUEST.server#",username="#REQUEST.username#",
                    password="#REQUEST.password#",secure="#REQUEST.secure#",
                    connection="#REQUEST.connectionname#",stoponerror="#REQUEST.stoponerror#");
    m.open();
    master = m.getAll(connection = "#REQUEST.connectionname#",name = "queryname", stoponerror =
"#REQUEST.stoponerror#" );
writedump(master);

</cfscript>
```

pop

説明

CFScript で POP メールサーバーから電子メールメッセージを取得または削除する場合に使用します。

シンタックス

モード	シンタックス

サービスの作成	new pop();
	または
	createObject("component", "pop");

属性の初期化	1512 ページの「属性の初期化」を参照してください。
サービスアクションの実行	popService.*action_method*(**attribute-value_pair**);
CFC のプロパティの取得	1512 ページの「CFC のプロパティの取得」を参照してください。
返されたデータの使用	popresult = popService.action_method (*attribute-value* pair);。action_method が getAll または getHeaderOnly の場合、popresult はクエリーオブジェクトです。 その他のメソッドの場合は、何も返されません。

プロパティ

server		attachmentPath	debug
generateUniqueFilenames	maxRows	messageNumber	name
password	port	startRow	timeout
uid	username		

cfpop タグでサポートされるすべての属性を、属性と値のペアとして使用できます。次に例を示します。

```
<cfpop server = "#form.popserver# " action = "getHeaderOnly" name = "GetHeaders">
```

この属性は次のように使用できます。

```
popHeaders = popService.getHeaderOnly(server="#form.popserver#");
```

注意：name は、cfpop では必須の属性ですが、CFScript ではオプションです。

関連項目

1501 ページの「関数一覧」

履歴
ColdFusion 9 アップデート 1: この関数が追加されました。

メソッド
次の pop アクションをメソッドとして使用できます。引数とシンタックスはすべてのメソッドでほぼ共通です。

getHeaderOnly　　getAll　　delete

説明	これらのメソッドはすべて、cfpop タグでサポートされる情報のタイプに対応しています。各メソッドの詳細については、『ColdFusion 9 CFML リファレンス』の cfpop の該当セクションを参照してください。
戻り値	delete 以外のすべてのメソッドは、クエリーオブジェクトを返します。
シンタックス	popService.*methodName*(*attribute-value pair*)
引数	cfpop タグでサポートされるすべての属性。

- setAttributes、getAttributes、clearAttributes、clear、setProperties、getProperties および clearProperties の詳細については、1511 ページの「すべての関数で共通のメソッド」を参照してください。

使用方法
この関数は cfpop タグに対応します。使用方法の詳細については、『ColdFusion 9 CFML リファレンス』の cfpop の「使用方法」セクションを参照してください。

例
```
<cfscript>

    p = createObject("component","pop");
    p.setAttributes(server="#popServer#",username="failoveruser",password="#popPassword#");
    r = p.GetAll(name="results",maxRows = "2");

    writeoutput("getAll Passed<br>");

    r = p.GetAll(messageNumber = "2");
    writeoutput(#r.FROM# & "<br>");


    r= p.GETHEADERONLY(messageNumber = "1");
    writeoutput(#r.subject# & "<br>");


</cfscript>
```

ldap

説明
CFScript で Netscape Directory Server などの LDAP (Lightweight Directory Access Protocol) ディレクトリサーバーへのインターフェイスを提供する場合に使用します。

シンタックス

モード	シンタックス

サービスの作成	new ldap();
	または
	createObject("component", "ldap");

属性の初期化	1512 ページの「属性の初期化」を参照してください。
サービスアクションの実行	ldapService.*action_method*(*attribute-value pair*);
CFC のプロパティの取得	1512 ページの「CFC のプロパティの取得」を参照してください。
データの使用	ldapresult = ldapService.query(*attribute-value pair*) その他のメソッドの場合は、何も返されません。

プロパティ

	server	attributes	delimiter
dn	filter	maxRows	modifyType
name	password	port	rebind
referral	returnAsBinary	scope	secure
separator	sort	sortcontrol	start
startRow	timeout	userName	

cfldap タグでサポートされるすべての属性を、属性と値のペアとして使用できます。次に例を示します。

```
<cfldap action="add" server="ldap.uconn.edu">
```

この属性は次のように使用できます。

```
ldapService.add(server="ldap.uconn.edu");
```

詳細については、『ColdFusion 9 CFML リファレンス』の cfldap タグの「属性」セクションを参照してください。

メソッド

次の ldap アクションをメソッドとして使用できます。引数とシンタックスはすべてのメソッドでほぼ共通です。

query	add	modify	modifyDN
delete			

説明	これらのメソッドはすべて、cfldap タグでサポートされるアクションに対応しています。各メソッドの詳細については、『ColdFusion 9 CFML リファレンス』の cfldap の該当セクションを参照してください。
戻り値	メソッドが query の場合、クエリーオブジェクトを返します。それ以外の場合は、何も返しません。
シンタックス	ldapService.*methodName*(*attribute-value pair*)
引数	cfldap タグでサポートされるすべての属性。

- setAttributes の詳細については、1511 ページの「すべての関数で共通のメソッド」を参照してください。
- getAttributes、clearAttributes、clear、setProperties、getProperties および clearProperties の詳細については、1511 ページの「すべての関数で共通のメソッド」を参照してください。
- setLdapAttributes

説明	attributes プロパティを設定します。
戻り値	なし

シンタックス	ldapService.*setLdapAttributes*(*attribute-value*);
引数	attributes プロパティの値を含む文字列

- getLdapAttributes

説明	attributes プロパティを取得します。
戻り値	attributes プロパティの値を含む文字列
シンタックス	myattributes = ldapService.*getLdapAttributes*();

関連項目
1501 ページの「関数一覧」

履歴
ColdFusion 9 アップデート 1: この関数が追加されました。

使用方法
この関数は cfldap タグに対応します。使用方法の詳細については、cfldap の「使用方法」セクションを参照してください。

例
```
<cfscript>

    l = new ldap();
l.setLdapAttributes("objectclass=top, person, organizationalPerson, inetOrgPerson;cn=Joe Smith; sn=Smith;
mail=spenella@allaire.com; telephonenumber=(617) 761 - 2128");
l.setUsername("uid=admin,ou=system");
l.setPassword("administrator");
l.setPort(port);
l.setServer(ldapserver);

    l.setdn("ou=People+o=aribus.com,dc=example,dc=com");

    l.add();-

l.clearAttributes();result = l.query(name="apache",
            attributes="dn,cn,o,ou,c,mail,telephonenumber",
            start="dc=example,dc=com",
            scope="SUBTREE",
            filter="(&(cn=Joe Smith)(ou=people))";

    writeoutput("<b>Adding and Querying a LDAP entry : </b>" & "CN = " & result.CN & " DN = " & result.DN
& "<br> ");
    l.clearAttributes();
    l.delete(
                DN="ou=People+o=aribus.com,dc=example,dc=com",
                );


</cfscript>
```

feed

説明

CFScript で RSS または Atom フィードの読み込みまたは作成を行う場合に使用します。このサービスでは、RSS バージョン 0.90、0.91、0.92、0.93、0.94、1.0、2.0、および Atom 0.3 または 1.0 を読み込みます。また、RSS 2.0 または Atom 1.0 のフィードを作成できます。

シンタックス

モード	シンタックス
サービスの作成	new feed() または createObject("component" "feed")
属性の初期化	1512 ページの「属性の初期化」を参照してください。
サービスアクションの実行	feedService.*action_method*(*attribute-value_pair*)
CFC のプロパティの取得	1512 ページの「CFC のプロパティの取得」を参照してください。
返されたデータの使用	• feedresult = feedService.read(*attribute-value_pair*)。feedresult は、name、query、properties および xmlVar のキーを含む構造体です。 • feedresult = feedService.create(attribute-value_pair)。feedresult は xmlVar を含む文字列です。

プロパティ

columnMap	enclosureDir	escapeChar	ignoreEnclosureError
name (CFScript ではオプション)	outputFile	overwrite	overwriteEnclosure
properties (CFScript ではオプション)	proxyPassword	proxyPort	proxyServer
proxyUser	query (CFScript ではオプション)	source	timeout
useragent	xmlVar (CFScript ではオプション)		

cffeed タグでサポートされるすべての属性を、属性と値のペアとして使用できます。次に例を示します。

```
<cffeed action="read" source="http://googleblog.blogspot.com/atom.xml"
query="feedQuery" properties="feedMetadata" >
```

この属性は次のように使用できます。

```
feedservice.read(source="http://googleblog.blogspot.com/atom.xml",
query="feedQuery", properties="feedMetadata");
```

関連項目

1501 ページの「関数一覧」

履歴

ColdFusion 9 アップデート 1: この関数が追加されました。

メソッド

- create

説明	RSS 2.0 または Atom 1.0 フィードの XML ドキュメントを作成して、変数に保存するかファイルに書き出します (両方を行うこともできます)。
戻り値	xmlVar を表す文字列
シンタックス	feedService.create (*attribute-value pair*);
引数	cffeed タグでサポートされるすべての属性。

- read

説明	URL または XML ファイルから RSS または Atom フィードを読み込んで解析し、構造体またはクエリーに保存します。別の構造体にあるフィードメタデータを取得することもできます。
戻り値	次のキーを含む構造体 • name • query • properties • xmlVar
シンタックス	feedService.read (*attribute-value pair*);
引数	cffeed タグでサポートされるすべての属性。

- setAttributes、getAttributes、clearAttributes、clear、setProperties、getProperties および clearProperties の詳細については、1511 ページの「すべての関数で共通のメソッド」を参照してください。
- getFeedProperties

説明	properties プロパティの値を返します。
戻り値	構造体またはエラー (プロパティが設定されていない場合)
シンタックス	feedService.getFeedProeprties()
引数	なし

- setFeedProperties

説明	properties プロパティの値を設定します。
戻り値	なし
シンタックス	feedService.setFeedProperties()
引数	properties 構造体

使用方法

このサービスは cffeed タグに対応します。使用方法については、『ColdFusion 9 CFML リファレンス』の cffeed の「使用方法」セクションを参照してください。

例

```
<cfscript>
        f = new feed();
    r = f.read(source=feedpath);

        writeoutput("Name : " & r.name.title & "<br>");
        writeoutput("Properties : " & r.properties.version & "<br>");
        writeoutput("Query : " & r.query.recordcount & "<br>");
        writeoutput("XMLVar : " & r.xmlvar.length() & "<br>");
</cfscript>
```

すべての関数で共通のメソッド

次のメソッドは、すべてのスクリプト関数で共通です。

- setAttributes

説明	関数の属性を設定します。
戻り値	なし
シンタックス	*service_name*.*setAttributes* (*attribute-value pair*);
引数	対応するタグでサポートされるすべての属性。

- getAttributes

説明	関数に対して設定された属性を取得します。
戻り値	すべてまたは一部の属性値を含む構造体を返します。
シンタックス	*service_name*.get*Attributes* (*attributelist*);
引数	属性のカンマ区切りリスト。リストが指定されていない場合は、定義されているすべての属性が返されます。

- clearAttributes

説明	関数に対して追加されたすべての属性を削除します。
戻り値	なし
シンタックス	**service_name**.clearAttributes(*attribute_list*);
引数	属性のカンマ区切りリスト。

- clear

説明	関数に対して追加されたすべての属性を削除します。
戻り値	なし
シンタックス	*service_name*.clear();
引数	なし

- clearProperties

説明	関数に対して追加されたすべてのプロパティを削除します。
戻り値	なし
シンタックス	*service_name*.clearProperties(*attribute_list*);

引数	指定がない場合は、すべてのプロパティが削除されます。

- setProperties

説明	関数のプロパティを設定します。
戻り値	なし
シンタックス	*service_name.setproperties* (*attribute-value pair*);
引数	対応するタグでサポートされるすべての属性。

- getProperties

説明	関数に対して設定されたプロパティを取得します。
戻り値	すべてまたは一部の属性値を含む構造体を返します。
シンタックス	*service_name*.getproperties (*attributelist*);
引数	属性のカンマ区切りリスト。リストが指定されていない場合は、定義されているすべての属性が返されます。

属性の初期化

次のいずれかの方法で、属性を初期化できます。

- ***service_name***=new dbinfo(***attribute-value pair***)
- ***service_name***=new dbinfo().init(attribute-value pair)
- ***service_name***.setAttributes(**attribute-value pair**)
- ***service_name***.setA***ttributeName***(attribute_value)
- ***service_name.action_method***(***attribute-value_pair***)
- ***service_name***.setProperties (***attribute_value***)

CFC のプロパティの取得

次のいずれかの方法で、CFC のプロパティを取得します。

- ***service_name***.getAttributeName(***attributelist***)
- ***service_name***.getProperties (***attributelist***)
- ***service_name***.getAttributes(***attributelist***)

第 7 章：ColdFusion Flash フォームスタイルリファレンス

フォームまたはフォーム要素を Flash 形式で表示するときに、ColdFusion フォームタグでスタイルを指定できます。

注意：Inh の列は、vbox のフォームコントロールなどのチャイルドコントロールにスタイルが継承されるかどうかを示します。

すべてのコントロールに有効なスタイル

次の type 属性を持つ cfformitem タグでは、style 属性を使用しません。以下に説明するスタイルは、このタグ以外のすべての ColdFusion Flash 形式フォームタグに有効です。

- html
- space

以下に説明するスタイルは、この例外以外のどのタグに使用してもエラーは発生しません。一方、一部のタグに使用した場合に効果がないスタイルも多数あります。

スタイル	Inh	説明
backgroundAlpha	N	backgroundImage によって定義される SWF ファイルまたはイメージの alpha (透明度) レベルを指定します。有効な値は、0 (透明) ～ 100 (不透明) です。デフォルト値は 100 です。
backgroundColor	Y	形式は色です。コントロールの背景色を指定します。cfform コントロールタグで指定しても、効果はありません。このタグの場合は、background-color スタイルを使用して色を制御します。type 属性が button、img、submit、radio、または checkbox に設定された cfinput タグの場合も無視されます。これらの場合は、ボタン表面などのグラフィックによって背景が塗りつぶされるためです。
backgroundDisabledColor	Y	形式は色です。無効時のコンポーネントの背景色を指定します。デフォルト値は ##EFEEEF (ライトグレー) です。
backgroundSize	N	backgroundImage で指定されたイメージを別のサイズ比率に拡大縮小します。デフォルト値は auto で、イメージの元のサイズが維持されます。値を 100% に設定すると、イメージは画面いっぱいになるように拡大されます。値にはパーセント記号を付けます。
barColor	Y	形式は色です。外側のバーの色を指定します。
borderCapColor	Y	形式は色です。スキンの左外側および右外側の色を指定します。
borderColor	Y	形式は色です。3 次元ボーダーの black セクション、または 2 次元ボーダーの color セクションを指定します。
borderSides	N	境界ボックスの境界の側面を指定します。borderStyle="solid" の場合に限り、使用されます。表示するボーダーの側面をスペース区切り文字列で示します。順序は重要ではありません。デフォルト値は left top right bottom です。

スタイル	Inh	説明
borderStyle	Y	境界ボックスの表示スタイルです。指定できる値は次のとおりです。 • inset (デフォルト) • none • outset • solid
borderThickness	N	境界ボックスの太さを指定します。borderStyle="solid" の場合に限り、使用されます。デフォルト値は 1 です。
color	Y	形式は色です。コンポーネントのラベルのテキスト色を指定します。
cornerRadius	N	コンポーネントの角に付いている丸みの半径です。デフォルト値は 0 です。
disabledColor	Y	形式は色です。無効時のコンポーネントの色を指定します。
dropShadow	N	形式はブール値です。コンポーネントのドロップシャドウを表示するかどうかを制御します。デフォルト値は false です。このスタイルは、borderStyle="solid" の場合に限り、使用されます。ドロップシャドウをコンテナに表示するには、backgroundColor または backgroundImage を設定します。これらのスタイルを設定しない場合、コンテナのデフォルトの背景は透明なので、ドロップシャドウはコンテナの背後に表示されます。
errorColor	Y	形式は色です。エラーテキストの色を指定します。
fillColors	N	形式は色です。コントロールの背景に付ける色を指定します。フラットな外観のコントロールを作成するには、どちらの値にも同じカラーを渡します。デフォルト値は ##E6EEEE、##FFFFFF です。
fontFamily	Y	使用するフォントをカンマで区切ったリストです。望ましい順に示されます。任意のフォントファミリー名を使用できます。汎用フォント名を指定した場合、適切なデバイスフォントに置換されます。Flash では、クライアントシステムにインストールされているフォントのみを使用できます。
fontSize	Y	形式は長さです。テキストのサイズを指定します。
fontStyle	Y	テキストがイタリックかどうかを指定します。認識される値は normal および italic です。デフォルト値は normal です。
fontWeight	Y	テキストがボールドかどうかを指定します。認識される値は normal および bold です。デフォルト値は normal です。
highlightColor	Y	形式は色です。強調表示時のコントロールの色を指定します。
horizontalGap	N	形式は長さです。水平方向の子の間隔をピクセル数で指定します。
leading	N	テキスト行の垂直方向の追加間隔を指定します。デフォルト値は no leading です。
marginLeft	N	形式は長さです。コンテナの左ボーダーとコンテンツ領域との間隔をピクセル数で指定します。
marginRight	N	形式は長さです。コンテナの右ボーダーとコンテンツ領域との間隔をピクセル数で指定します。
scrollTrackColor	Y	形式は色です。スクロールバーのスクロールトラックを指定します。デフォルト値は ##EFEEEF (ライトグレー) です。
selectedFillColors	N	形式は色です。選択した状態のときにコントロールの背景に付ける 2 色を指定します。フラットな外観のコントロールを作成するには、どちらの値にも同じカラーを渡します。デフォルト値は定義されていません。この色は themeColor に基づいて決定されます。
textAlign	Y	コンテナ内でテキストを整列します。認識される値は、left、right、および center です。デフォルト値は right です。
textDecoration	N	テキストにアンダーラインを付けるかどうかを指定します。認識される値は none および underline です。デフォルト値は none です。

スタイル	Inh	説明
textIndent	Y	形式は長さです。テキストの先頭行の、コンテナの左側からのオフセットを指定します。デフォルト値は 0 です。
themeColor	Y	形式は色です。コンポーネントの背景色を指定します。指定できる値は次のとおりです。 • haloGreen • haloBlue • haloOrange • haloSilver
verticalGap	N	形式は長さです。垂直方向の子の間隔をピクセル数で指定します。

cfform に有効なスタイル

以下のスタイルは、cfform タグに適用されます。

スタイル	Inh	説明
background-color		形式は色です。フォームの背景色を指定します。
indicatorGap	Y	形式は長さです。ラベルと子コンポーネントとの間隔をピクセル数で指定します。デフォルト値は 14 です。
labelWidth	Y	形式は長さです。フォームラベルの幅を指定します。デフォルト値は、フォーム内で最も長いラベルの長さです。
marginBottom	N	形式は長さです。コンテナの下部ボーダーとコンテンツ領域との間隔をピクセル数で指定します。デフォルト値は 16 です。
marginTop	N	形式は長さです。コンテナの上部ボーダーとコンテンツ領域との間隔をピクセル数で指定します。デフォルト値は 16 です。
verticalGap	N	形式は長さです。垂直方向の子の間隔をピクセル数で指定します。デフォルト値は 8 です。

type 属性が horizontal または vertical である cfformgroup に有効なスタイル

以下のスタイルは、type 属性が horizontal または vertical である cfformgroup タグに適用されます。

スタイル	Inh	説明
horizontalAlign	N	子の水平方向の整列を指定します。指定できる値は、left、center、および right です。デフォルト値は left です。
horizontalGap	N	形式は長さです。水平方向の子の間隔をピクセル数で指定します。デフォルト値は 6 です。
indicatorGap	Y	形式は長さです。ラベルと子コンポーネントとの間隔をピクセル数で指定します。デフォルト値は 14 です。
labelWidth	Y	形式は長さです。フォームラベルの幅を指定します。デフォルト値は、フォーム内で最も長いラベルの長さです。

スタイル	Inh	説明
marginBottom	N	形式は長さです。コンテナの下部ボーダーとコンテンツ領域との間隔をピクセル数で指定します。デフォルト値は 0 です。
marginTop	N	形式は長さです。コンテナの上部ボーダーとコンテンツ領域との間隔をピクセル数で指定します。デフォルト値は 0 です。
verticalGap	N	形式は長さです。垂直方向の子の間隔をピクセル数で指定します。デフォルト値は 6 です。

ボックススタイルの cfformgroup 要素に有効なスタイル

以下のスタイルは、次の type 属性がある cfformgroup タグに適用されます。一部のタイプには追加属性があり、後のセクションに記載しています。

- hbox
- vbox
- hdividedbox
- vdividedbox
- panel
- tile
- page

スタイル	Inh	説明
horizontalAlign	N	このコンテナに含まれる子の水平方向の配置です。デフォルト値は left です。指定できる値は、left、center、および right です。
horizontalGap	N	形式は長さです。水平方向の子の間隔をピクセル数で指定します。デフォルト値は 8 (tile の場合は 6) です。
marginBottom	N	形式は長さです。コンテナの下部ボーダーとコンテンツ領域との間隔をピクセル数で指定します。デフォルト値は 0 です。
marginTop	N	形式は長さです。コンテナの上部ボーダーとコンテンツ領域との間隔をピクセル数で指定します。デフォルト値は 0 です。
verticalAlign	N	コンテナ内での垂直方向の子の配置を指定します。デフォルト値は top です。指定できる値は、top、middle、および bottom です。
verticalGap	N	形式は長さです。垂直方向の子の間隔をピクセル数で指定します。デフォルト値は 8 (tile の場合は 6) です。

type 属性が hdividedbox または vdividedbox である cfformgroup に固有のスタイル

次の追加スタイルは、type="hdividedbox" または type="vdividedbox" である cfformgroup タグに適用されます。

スタイル	Inh	説明
dividerAffordance	N	形式は長さです。ユーザーがマウスポインタを使って選択できる区切りの領域の幅 (hdividedbox の場合) または高さ (vdividedbox の場合) をピクセル単位で指定します。デフォルト値は 6 です。
dividerColor	Y	形式は色です。up ステートのときの区切りの色を指定します。デフォルト値は ##AAAAAA です。
dividerThickness	N	形式は長さです。区切りの太さをピクセル単位で指定します。デフォルト値は 4 です。

type 属性が panel である cfformgroup に固有のスタイル

次の追加スタイルは、type="panel" である cfformgroup タグに適用されます。

スタイル	Inh	説明
cornerRadius	N	形式は長さです。ウィンドウフレームの角の丸みを指定します。デフォルト値は 8 です。
dropShadow	N	パネルにドロップシャドウを表示するかどうかを示すブール値を指定します。デフォルト値は true です。
footerColors	Y	形式は色です。フッタ (ControlBar) の背景に使用する 2 色を示す、カンマ区切りのリストを指定します。最初の色は、上部の色です。2 番目の色は、下部の色です。デフォルト値は ##F4F5F7、##E1E5EB です。
headerColors	Y	形式は色です。ヘッダに使用する 2 色を示す、カンマ区切りのリストです。最初の色は、上部の色です。2 番目の色は、下部の色です。デフォルト値は ##E1E5EB、##F4F5F7 です。
headerHeight	N	形式は長さです。ヘッダの高さを指定します。デフォルト値は 28 です。
panelBorderStyle	N	コンテナ下部の 2 つの角に適用する境界スタイルを指定します。コンテナ上部の 2 つの角には常に丸みが付きます。指定できる値は、default (四角い角) および roundCorners (丸みがある角) です。上部の角を四角くする場合は、cornerRadius を 0 に設定します。デフォルト値は default です。
shadowDirection	N	ドロップシャドウの方向です。指定できる値は、left、center、および right です。デフォルト値は center です。
shadowDistance	N	ドロップシャドウの距離です。負の値を指定すると、ドロップシャドウはパネルの手前側に移動します。デフォルト値は 2 です。

type 属性が accordion である cfformgroup に有効なスタイル

次のスタイルは、type="accordion" である cfformgroup タグに適用されます。

スタイル	Inh	説明
headerHeight	N	形式は長さです。アコーディオン形式コンテナのボタンの高さをピクセル単位で指定します。デフォルト値は 22 です。
marginBottom	N	形式は長さです。コンテナの下部ボーダーとコンテンツ領域との間隔をピクセル数で指定します。デフォルト値は -1 です。
marginTop	N	形式は長さです。コンテナの上部ボーダーとコンテンツ領域との間隔をピクセル数で指定します。デフォルト値は -1 です。
openDuration	N	形式は時間です。ある子パネルから別の子パネルへの移動時間をミリ秒単位で指定します。デフォルト値は 250 です。
verticalGap	N	形式は長さです。垂直方向の子の間隔をピクセル数で指定します。デフォルト値は -1 です。

type 属性が tabnavigator である cfformgroup に有効なスタイル

次のスタイルは、type="tabnavigator" である cfformgroup タグに適用されます。

スタイル	Inh	説明
horizontalAlign	N	子の水平方向の整列を指定します。デフォルト値は left です。指定できる値は、left、center、および right です。タブナビゲータコンテナ内の各タブの優先幅はラベルテキストのサイズであるため、整列の表示サイズを変えるには、tabWidth スタイルを使用してタブの幅のサイズを優先幅を超えるまで大きくします。
horizontalGap	N	形式は長さです。水平方向の子の間隔をピクセル数で指定します。デフォルト値は 6 です。
tabHeight	N	形式は長さです。デフォルトのタブの高さをピクセル単位で指定します。デフォルト値は 22 です。
tabWidth	N	形式は長さです。タブの幅をピクセル単位で指定します。未定義の場合、デフォルトのタブの幅はラベルテキストに基づいて自動的に計算されます。コンテナの幅がラベルテキストの幅より小さい場合、ラベルは切り捨てられます。タブラベルが切り捨てられた場合、ユーザーがマウスポインタをそのタブ上に動かすと、ラベルテキスト全文を示すツールチップが表示されます。明示的なタブ幅を指定すると、表示可能なスペースに収まりきらない場合でもラベルが自動的に縮小されることはありません。

type 属性が hrule または vrule である cfformitem に有効なスタイル

次のスタイルは、type="hrule" または type="vrule" である formitem タグに適用されます。

スタイル	Inh	説明
color	Y	形式は色です。線の色を指定します。次のルールに従います。 • strokeWidth が 1 の場合、線全体の色を指定します。 • strokeWidth が 2 (デフォルト) の場合、上部の線の色を指定します。 • strokeWidth が 3 以上の場合、矩形の上端および左端の色を指定します。 デフォルト値は ##C4CCCC です。
shadowColor	Y	形式は色です。線の影の色を次のように指定します。 • strokeWidth が 1 の場合、色の指定はありません。 • strokeWidth が 2 (デフォルト) の場合、下部の線の色を指定します。 • strokeWidth が 3 以上の場合、矩形の下端および右端の色を指定します。 デフォルト値は ##D4D0C8 です。
strokeWidth	Y	罫線の太さを次のようにピクセル単位で指定します。 • strokeWidth が 1 の場合、罫線は 1 ピクセル幅の線になります。 • strokeWidth が 2 (デフォルト) の場合、1 ピクセル幅の水平線が 2 本隣り合います。 • strokeWidth が 3 以上の場合、罫線は 1 ピクセル幅の枠線で作られた空白の矩形になります。 デフォルト値は 2 です。

type 属性が radio、checkbox、button、image、または submit である cfinput に有効なスタイル

以下のスタイルは、type 属性が次の各値である cfinput タグに適用されます。

- button
- checkbox
- image
- radio
- submit

説明に記載されているように、これらの入力タイプのサブセットにのみスタイルが適用される場合もあります。

スタイル	Inh	説明
borderThickness	N	ボーダーの " 外枠 " の太さを指定します。この値が 0 の場合、ボーダーはありません。この値が 3 以上の場合、ボタンの周囲に強調表示された " 外枠 " が作成されます。デフォルト値は 3 です。
cornerRadius	N	角の丸みを指定します。デフォルト値は 5 です。
horizontalGap	N	labelPlacement = "left" または "right" の場合の、img 入力内のラベルとイメージの間隔を指定します。デフォルト値は 2 です。
repeatDelay	N	形式は時間です。最初の buttonDown イベント後、repeatInterval で buttonDown イベントが繰り返されるまでの待機時間をミリ秒単位で指定します。デフォルト値は 500 です。
repeatInterval	N	形式は時間です。ボタンを押したままにした場合の buttonDown イベントの実行間隔をミリ秒単位で指定します。デフォルト値は 35 です。
symbolBackgroundColor	Y	形式は色です。チェックボックスおよびラジオボタンの背景色を指定します。デフォルト値は ##FFFFFF (白) です。
symbolBackgroundDisabledColor	Y	形式は色です。無効時のチェックボックスおよびラジオボタンの背景色を指定します。デフォルト値は ##EFEEEF (ライトグレー) です。
symbolBackgroundPressedColor	Y	形式は色です。有効時のチェックボックスおよびラジオボタンの背景色を指定します。デフォルト値は ##FFFFFF (白) です。
symbolColor	Y	形式は色です。チェックボックスのチェックマークまたはラジオボタンのドットの色を指定します。デフォルト値は ##000000 (黒) です。
symbolDisabledColor	Y	形式は色です。無効時のチェックボックスのチェックマークまたはラジオボタンのドットの色を指定します。デフォルト値は ##848384 (ダークグレー) です。
texRollOverColor	Y	形式は色です。マウスポインタをコントロールの上に動かしたときのラベルテキストの色を指定します。デフォルト値は ##2B333C です。
textSelectColor	Y	形式は色です。コントロールを選択したときのラベルテキストの色を指定します。デフォルト値は ##000000 です。
verticalGap	N	labelPlacement = "top" または "bottom" の場合の、img 入力内のラベルとイメージの間隔を指定します。デフォルト値は 2 です。

cftextarea タグ、および type 属性が text、password、または hidden である cfinput に有効なスタイル

以下のスタイルは、次のタグとタグ属性の組み合わせに適用されます。

- textarea
- cfinput type="hidden"
- cfinput type="password"
- cfinput type="text"

スタイル	Inh	説明
disabledColor	Y	形式は色です。無効時のテキスト領域の色を指定します。

size 属性の値が 1 である cfselect に有効なスタイル

以下のスタイルは、size 属性が 1 の場合、つまりコントロールがドロップダウンリスト (コンボボックス) を使用してオプションを一度に 1 つずつ表示するときに、cfselect タグに適用されます。

スタイル	Inh	説明
alternatingRowColors	Y	形式は反復パターンの行の色を示すカンマ区切りのリストです。値は 2 色以上の色のリストです。backgroundColor スタイルを指定しない場合にのみ使用します。
closeDuration	N	ドロップダウンリストを開くまでの時間をミリ秒単位で指定します。デフォルト値は 250 です。
openDuration	N	ドロップダウンリストを開くまでの時間をミリ秒単位で指定します。デフォルト値は 250 です。
rollOverColor	Y	形式は色です。ユーザーがアイテムをロールオーバーするときの背景色を指定します。デフォルト値は ##0EFFD6 です。
selectionColor	Y	形式は色です。ユーザーがアイテムを選択したときの背景色を指定します。デフォルト値は ##0DFFC1 です。

size 属性が 2 以上である cfselect に有効なスタイル

以下のスタイルは、size 属性が 2 以上の場合、つまり、コントロールが一度に 2 つ以上のオプションを表示するリストボックスであるときに、cfselect タグに適用されます。

スタイル	Inh	説明
alternatingRowColors	Y	形式は反復パターンの行の色を示すカンマ区切りのリストです。値は 2 色以上の色のリストです。
marginBottom	N	形式は長さです。行の下端とその行内のテキストの下端との間隔をピクセル数で指定します。デフォルト値は 0 です。
marginTop	N	形式は長さです。行の上端とその行内のテキストの上端との間隔をピクセル数で指定します。デフォルト値は 0 です。
rollOverColor	Y	形式は色です。ユーザーがマウスポインタをリンクの上に動かしたときの背景色を指定します。デフォルト値は ##0EFFD6 です。
selectionColor	Y	形式は色です。ユーザーがリンクを選択したときの背景色を指定します。デフォルト値は ##0DFFC1 です。

スタイル	Inh	説明
selectionDuration	N	選択アニメーションの表示期間をミリ秒単位で指定します。デフォルト値は 250 です。0 にすると、アニメーションが無効になります。
textRollOverColor	Y	形式は色です。ユーザーがマウスポインタを選択対象の上に動かしたときのテキストの色を指定します。デフォルト値は ##02B33C です。
textSelectedColor	Y	形式は色です。選択したときのテキストの色を指定します。デフォルト値は ##005F33 です。

cfcalendar タグ、および type 属性が dateField である cfinput に有効なスタイル

以下のスタイルは、cfcalendar タグおよび type が dateField である cfinput タグに適用されます。

スタイル	Inh	説明
headerColors	Y	形式は色です。DateChooser コントロール上部のバンドの色を指定します。2 つの値をカンマで区切って指定します。塗りつぶしのバンドの場合、両方の値に同じ色を使用します。デフォルト値は ##E6EEEE、##FFFFFF です。
rollOverColor	Y	形式は色です。ユーザーがマウスポインタを DateField の上に動かしたときの背景色を指定します。デフォルト値は ##E3FFD6 です。
selectionColor	Y	形式は色です。ユーザーが DateField を選択したときの背景色を指定します。デフォルト値は ##CDFFC1 です。
todayColor	Y	形式は色です。今日の日付の色を指定します。デフォルト値は ##2B333C です。

cfgrid タグに有効なスタイル

以下のスタイルは、cfgrid タグに適用されます。

スタイル	Inh	説明
horizontalAlign	N	このコンテナに含まれる子の水平方向の配置です。デフォルト値は left です。指定できる値は、left、center、および right です。
horizontalGap	N	水平方向の子の間隔をピクセル数で指定します。デフォルト値は 8 です。
marginBottom	N	コンテナの下部ボーダーとコンテンツ領域との間隔をピクセル数で指定します。デフォルト値は 0 です。
marginTop	N	コンテナの上部ボーダーとコンテンツ領域との間隔をピクセル数で指定します。デフォルト値は 0 です。
verticalAlign	N	コンテナ内での垂直方向の子の配置を指定します。デフォルト値は top です。指定できる値は、top、middle、および bottom です。
verticalGap	N	垂直方向の子の間隔をピクセル数で指定します。デフォルト値は 8 です。

cftree タグに有効なスタイル

以下のスタイルは、cftree タグに適用されます。

スタイル	Inh	説明
alternatingRowColors	Y	型は配列です。反復パターンの行の色を指定します。値は、2 つ以上の色の配列です。
depthColors	Y	型は配列です。ツリーコントロールで使用する色の配列を降順で指定します。
indentation	N	各ツリーレベルのインデントをピクセル数で指定します。デフォルト値は 8 です。
openDuration	N	形式は時間です。開くまたは閉じるまでの時間の長さをミリ秒単位で指定します。デフォルト値は 250 です。
rollOverColor	Y	形式は色です。ユーザーがマウスポインタをリンクの上に動かしたときの背景色を指定します。デフォルト値は ##E3FFD6 です。
selectionColor	Y	形式は色です。ユーザーがリンクを選択したときの背景色を指定します。デフォルト値は ##CDFFC1 です。
selectionDuration	N	選択アニメーションの表示期間をミリ秒単位で指定します。デフォルト値は 250 です。0 にすると、アニメーションが無効になります。
textRollOverColor	Y	形式は色です。ユーザーがマウスポインタをエントリの上に動かしたときのテキストの色を指定します。デフォルト値は ##02B33C です。
textSelectedColor	Y	形式は色です。ユーザーがエントリを選択したときのテキストの色を指定します。デフォルト値は ##005F33 です。

第 8 章：Application.cfc リファレンス

Application.cfc では、ColdFusion アプリケーションイベントを処理するためのメソッドを実装し、変数を設定してアプリケーションの特性を指定します。

アプリケーション変数

Application.cfc の This スコープには、cfapplication タグで設定した属性に対応する複数のビルトイン変数が含まれています。CFC メソッドを定義する前に、CFC 初期化コードでこれらの変数の値を設定します。任意のメソッドの変数にアクセスできます。

注意：Windows では大文字と小文字は区別されませんが、"Application.cfc" のファイル名は常に大文字の A で始める必要があります。application.cfc および Application.cfc はいずれも予約語です。

注意：アプリケーションに Application.cfc があり、Application.cfm または onRequestend.cfm ページがある場合、ColdFusion は CFM ページを無視します。

次の表では、アプリケーションの動作を制御するために設定できる変数について簡単に説明します。詳細については、cfapplication タグを参照してください。

変数	デフォルト	説明
name	名前なし	アプリケーション名です。この変数を設定しない場合、または空の文字列に設定した場合、CFC は、ColdFusion J2EE サーブレットコンテキストである、名前のない Application スコープに適用されます。名前のないスコープの詳細については、『ColdFusion アプリケーションの開発』の Integrating JSP and servlets in a ColdFusion application を参照してください。
applicationTimeout	Administrator の値	アプリケーション (すべての Application スコープ変数も含む) の有効期間を示す実数値 (日数)。この変数の値を生成するには、CFML CreateTimeSpan 関数を使用します。
authcookie.disableupdate	False	cfcookie または cfheader タグを使用した cfauthorization Cookie の更新を無効にします。
authcookie.timeout	-1	認証 Cookie の有効期間（日数）です。
chartStyleDirectory		アプリケーション固有のクライアントサイドのチャート作成スタイルディレクトリです。
clientManagement	Administrator の値	アプリケーションで Client スコープ変数をサポートするかどうかを示します。
clientStorage	Administrator の値	Client 変数を格納する場所を示します。Cookie、レジストリ、またはデータソース名を指定できます。
customtagpaths	Administrator の値	ColdFusion カスタムタグのパスを含みます。これは、絶対パスのカンマ区切りリストです。 この変数を使用するには、Administrator の**サーバー**／**設定**ページで「アプリケーションごとの設定の有効化」オプションを選択します。 ここで定義する設定は、現在のアプリケーションの Administrator の**サーバーの設定**／**マッピング**ページで定義されているカスタムタグのパスよりも優先されます。
googleMapKey		Web ページに Google マップを埋め込むために必要な Google Map API キーです。
datasource		クエリーによってデータを取得するデータソースの名前です。
loginStorage	cookie	ログイン情報を Cookie スコープと Session スコープのいずれに格納するかを示します。

変数	デフォルト	説明
mappings	Administrator の値	ColdFusion マッピングを含む構造体です。構造体の各エレメントは、1 つのキーと 1 つの値で構成されます。論理パスはキー、絶対パスは値です。 この変数を使用するには、Administrator の [サーバーの設定]-[設定] ページで [アプリケーションごとの設定の有効化] オプションを選択します。 ここで定義するマッピングは、現在のアプリケーションの Administrator の [サーバーの設定]-[マッピング] ページで定義されているマッピングよりも優先されます。
sessioncookie.httponly	True	セッション Cookie を httponly に設定する必要があるかどうかを指定します。つまり、Http リクエストにのみアクセスできます。
sessioncookie.secure	False	セッション Cookie をセキュアに設定する必要があるかどうかを指定します。つまり、すべてのタイプの接続で返されるか、またはセキュリティ保護されている（https）接続でのみ返されるかを指定します。
sessioncookie.domain		Cookie を設定するドメインです。これは、アプリケーションにアクセスするドメインと完全に一致する必要があります。
sessioncookie.timeout	30 年	セッション Cookie の有効期間（日数）です。
sessioncookie.disableupdate	False	cfcookie または cfheader タグを使用した cfid および cftoken Cookie の更新を無効にします。
serverSideFormValidation	yes	フォームの送信時に cfform フィールドで検証を有効にするかどうかを指定します。
sessionManagement	no	アプリケーションで Session スコープ変数をサポートするかどうかを示します。
sessionTimeout	Administrator の値	ユーザーセッション (すべての Session 変数も含む) の有効期間を示す実数値 (日数)。この変数の値を生成するには、CFML CreateTimeSpan 関数を使用します。
setClientCookies	True	クライアントブラウザに CFID Cookie および CFTOKEN Cookie を送信するかどうかを示します。
setDomainCookies	False	CFID Cookie および CFTOKEN Cookie を (1 つのホストではなく) ドメイン全体に設定するかどうかを示します。
scriptProtect	Administrator の値	変数をクロスサイトスクリプティング攻撃から 保護するかどうかを示します。
secureJSON	Administrator の値	リモートの呼び出しへの応答として ColdFusion 関数が JSON 形式で返す値の前に、セキュリティの接頭辞を追加するかどうかを指定するブール値です。 デフォルト値は、Administrator の [サーバーの設定]-[設定] ページの [シリアル化 JSON への接頭辞付加] 設定の値 (デフォルトは false) です。cffunction タグ内でこの値を上書きすることができます。 詳細については、『ColdFusion アプリケーションの開発』の Improving security を参照してください。
secureJSONPrefix	Administrator の値	secureJSON の設定が true の場合に、リモート呼び出しへの応答として ColdFusion 関数が JSON 形式で返す値の前に挿入するセキュリティの接頭辞です。 デフォルト値は、Administrator の [サーバーの設定]-[設定] ページの [シリアル化 JSON への接頭辞付加] 設定の値 (デフォルトは JavaScript のコメント文字である //) です。 詳細については、『ColdFusion アプリケーションの開発』の Improving security を参照してください。

変数	デフォルト	説明
welcomeFileList		ファイル名のカンマ区切りリストです。ファイルが見つからない場合に、onMissingTemplate メソッドを呼び出さないようにします。この変数を使用して、次のすべての条件を満たす場合は onMissingTemplate ハンドラを呼び出さないようにします。 • Web サーバーにウェルカムファイルリスト (例 : web.xml ファイル) があり、ディレクトリで終わるパスが URL に指定された場合に実行される index.cfm などの CFML ページが記載されている。 • Web サーバーは、ページが存在するかどうかを最初に判断せずに、ウェルカムファイルリストの CFML ページのリクエストを ColdFusion に送信する。 • ウェルカムファイルリストのファイルが存在しないディレクトリについて、ディレクトリ参照をサポートする。 この変数を指定するのは、"Application.cfc" ファイルで onMissingTemplate ハンドラが指定されている場合のみです。ハンドラには web.xml ウェルカムファイルリストと同様のファイルリストが必要です。 **メモ :** Apache などの多くの純粋な Web サーバーでは、welcomeFileList 変数を使用する必要はありません。統合型の Web サーバーやアプリケーションサーバーの多くでは、welcomeFileList 変数を使用する必要があります。
smtpServersettings		server、username および password の値を含む構造体です。値が指定されていない場合は、Administrator の値が使用されます。
timeout		有効期間です。 <cfsetting requesttimeout=""> を使用して設定したタイムアウト値は、Application.cfc で this.timeout="" を使用して設定したタイムアウト値より優先されます。
debuggingIPAddresses		デバッグを必要とする IP アドレスのリストです。
enablerobustexception		デフォルトの Administrator 設定を上書きします。コンパイル時の例外は報告されません。

同じ名前を持つフォームフィールド

フォームフィールドに同じ名前が設定されているとします。この場合、ColdFusion では、フォームフィールドをリストではなく、配列として変換します。これを行うには、Application.cfc で this.sameformfieldsasarray = "true" を指定します。デフォルト値は false です。

ColdFusion 9.0.1 で行われた機能強化

Application.cfc を使用すると、データソースの認証情報をデータソースに指定できます。今回のリリースでは、データソース設定は文字列または構造体のいずれかです。文字列の場合、データソース名と認証情報は、ColdFusion Administrator で定義されているデータソースから取得されます。

データソースに構造体の値を使用すると、認証情報を指定できます。キー名は、次のとおりです。

- name: データソース名
- username: データソースのユーザー名
- password: データソースのパスワード

例

```
<this.datasource={name='cfartgallery', username="user", password="passwd"}>
```

または

```
<this.datasource="cfartgallery">
```

注意：ORM のデフォルトのデータソースにも同じ規則が使用されます。ORM のデフォルトのデータソースでは、ormsettings でデータソースの認証情報を指定できます。

Amazon S3 統合用の次のアプリケーション固有の属性が追加されました。

- accessKeyId：Amazon S3 アカウントの ID です。
- awsSecretKey：S3 アカウントの秘密鍵です。
- defaultLocation：Amazon S3 のバケット作成のデフォルトの場所です。S3 ストレージのバケットは、US、EU または US-WEST のいずれかのリージョンを選択できます。

  Application.cfc で指定する defaultLocation で、作成するバケットのデフォルトの場所を定義します。デフォルト値は US です。

例

```
<cfscript>
this.s3.accessKeyId = "key_ID";
this.s3.awsSecretKey = "secret_key";
this.s3.defaultLocation="location";
</cfscript>
```

アプリケーション固有のメモリ内ファイルシステム

アプリケーション別のメモリ内ファイルシステムを使用できます。これにより、仮想ファイルシステムでアプリケーションの分離を実現できます。つまり、あるアプリケーションによってメモリ内ファイルシステムに作成されたファイルには、別のアプリケーションからアクセスできません。

この設定は、Application.cfc で次のように指定できます。

変数	説明
this.inmemoryfilesystem.enabled	アプリケーション用のメモリ内ファイルシステムを有効にするには、値を true に設定します。 これはデフォルトの設定です。
this.inmemoryfilesystem.size	メモリ内ファイルシステムのメモリ制限（MB 単位）を指定します。 また、この値は ColdFusion Administrator でも指定できます（サーバーの設定／設定／メモリ内仮想ファイルシステムのアプリケーションごとのメモリ制限）。 小さい方の値が優先されます。

メソッドの概要

次の表では、Application.cfc で実装できるアプリケーションイベントメソッドについて簡単に説明します。

メソッド名	メソッドを実行するタイミング
onAbort	cfabort タグを実行したときに実行されます。
onApplicationEnd	アプリケーションの終了時に実行します。つまり、アプリケーションのタイムアウト時またはサーバーの停止時です。
onApplicationStart	アプリケーションの最初の起動時に実行します。つまり、ページに対する最初のリクエストが処理されるときか、イベントゲートウェイインスタンス、Web サービス、または Flash Remoting リクエストによって CFC メソッドが最初に起動されるときに実行します。
onCFCRequest	アプリケーションに対して HTTP 呼び出しまたは AMF 呼び出しが行われたときに実行します。
onError	try/catch ブロックで検出されない例外の発生時に実行します。
onMissingTemplate	ColdFusion が、存在しないページに対するリクエストを受け取ったときに実行します。

メソッド名	メソッドを実行するタイミング
onRequest	onRequestStart メソッドの終了時に実行します。このメソッドでリクエストコンテンツにフィルタを設定することができます。
onRequestEnd	そのリクエストに含まれるすべてのページの処理完了時に実行します。
onRequestStart	リクエストの開始時に実行します。
onSessionEnd	セッションの終了時に実行します。
onSessionStart	セッションの開始時に実行します。
onServerStart	ColdFusion サーバーの起動時に実行します。

これらのメソッドに対するパラメータはすべて定位置パラメータです。これらのパラメータには任意の名前を使用できます。

リクエストの実行時に、ColdFusion は CFC メソッドを次の順序で実行します。

1. onApplicationStart (そのアプリケーションでまだ実行されていない場合)
2. onSessionStart (そのセッションでまだ実行されていない場合)
3. onRequestStart
4. onRequest ／ onCFCRequest
5. onRequestEnd

onApplicationEnd、onSessionEnd、および onError の各 CFC は、それぞれ特定のイベントによって実行されます。

onAbort

説明
cfabort タグを実行したときに実行されます。

注意： cfabort で showError 属性が指定されている場合は、OnAbort の代わりに onError メソッドが実行されます。

注意： cfabort、cflocation または cfcontent タグを使用した場合は、OnRequestEnd ではなく OnAbort メソッドが呼び出されます。

戻り値
なし

シンタックス

```
<cffunction name="onAbort" returnType="boolean">
    <cfargument type="string" name="targetPage" required=true/>
    ...
    <cfreturn BooleanValue />
</cffunction>
```

パラメータ

パラメータ	説明
targetPage	Web ルートからリクエストされた CFML ページまでのパスです。

例
Application.cfc

```
<cfcomponent>
<cffunction name="onAbort"
                            access="public"
                            returntype="void"
                            hint="Hanldes Aborted request">

                                <cfargument type="String"
                                            name="targetPage"
                                            required=true/>
                                <cfoutput> Target Page: #targetPage#</cfoutput>
                                <!--- do stuff --->
</cffunction>
</cfcomponent>
```

Test.cfm

```
<cfabort>
```

onApplicationEnd

説明

アプリケーションのタイムアウト時またはサーバーのシャットダウン時に実行されます。

シンタックス

```
<cffunction name="onApplicationEnd" returnType="void">
    <cfargument name="ApplicationScope" required=true/>
    ...
</cffunction>
```

関連項目

onApplicationStart、メソッドの概要、『ColdFusion アプリケーションの開発』の Managing the application with Application.cfc

パラメータ

ColdFusion は、次のパラメータをこのメソッドに渡します。

パラメータ	説明
ApplicationScope	Application スコープです。

戻り値

このメソッドは値を返しません。cfreturn タグを使用しないでください。

使用方法

このメソッドは、アプリケーションのシャットダウン時に必要となる任意のクリーンアップ処理に使用できます。たとえば、メモリ内にあるデータをデータベースに保存することや、アプリケーションの終了をファイルに記録することができます。このメソッドはリクエストに関連付けられていないので、ユーザーページにデータを表示することはできません。このメソッドで例外が発生した場合でもアプリケーションは終了します。

このメソッドを明示的に呼び出した場合、ColdFusion はアプリケーションを終了しません。メソッドコードを実行しますが、メソッドの実行中は Application スコープをロックしません。

Application スコープにアクセスするには、**ApplicationScope** パラメータを使用します。Application スコープを直接参照することはできません。たとえば、Application.myVariable ではなく、Arguments.ApplicationScope.myVariable を使用します。このメソッドは Server スコープに直接アクセスできますが、Session スコープや Request スコープにはアクセスできません。

注意：アプリケーションがアクティブでないままタイムアウト期間が経過すると、このアプリケーションはタイムアウトになります。アプリケーションが終了しても、セッションは終了しないので onSessionEnd メソッドは呼び出されません。詳細については、onSessionEnd を参照してください。

例

```
<cffunction name="onApplicationEnd">
    <cfargument name="ApplicationScope" required=true/>
    <cflog file="#This.Name#" type="Information"
        text="Application #Arguments.ApplicationScope.applicationname# Ended" >
</cffunction>
```

onApplicationStart

説明

ColdFusion がアプリケーションに対する最初のリクエストを受け取ったときに実行されます。

シンタックス

```
<cffunction name="onApplicationStart" returnType="boolean">
    ...
    <cfreturn Boolean>
</cffunction>
```

関連項目

onApplicationEnd、メソッドの概要、『ColdFusion アプリケーションの開発』の Managing the application with Application.cfc

戻り値

ブール値です。アプリケーションのスタートアップコードが正常に実行された場合は true、正常に実行されなかった場合は false を返します。cffunction タグの returntype 属性を省略した場合、true を明示的に返す必要はありません。

使用方法

このメソッドは、アプリケーションの初期化コードに使用できます。たとえば、Application スコープ変数を設定するとき、必要なデータソースやその他のリソースが利用可能かどうかを判別するとき、またはアプリケーションの開始をログに記録するときに使用します。このメソッドで Application 変数を指定した場合、Application スコープをロックする必要はありません。通常どおりに Application スコープ変数を参照できます。たとえば、Application.myVariable を使用します。

このメソッドでは、リクエストされたページを呼び出す onRequest メソッドが "Application.cfc" ファイルに含まれている場合に限り、リクエストされたページの Variables スコープにアクセスできます。

このメソッドを明示的に呼び出した場合、ColdFusion はアプリケーションを開始しません。メソッドコードを実行しますが、メソッドの実行中は Application スコープをロックしません。

検出されない例外がこのメソッドで発生した場合、または false が返された場合、アプリケーションは開始されず、ColdFusion はアプリケーション内のページを処理しません。この場合、ユーザーがそのアプリケーション内のページを次回リクエストしたときに、ColdFusion は onApplicationStart メソッドを実行します。

例
次の例では、データベースが利用可能かどうかをテストします。データベースを利用できない場合は、その旨が報告されてエラーがログに記録され、アプリケーションは起動しません。データベースを利用できる場合は、このメソッドが 2 つの Application スコープ変数を初期化します。

```
<cffunction name="onApplicationStart">
    <cftry>
        <!--- Test whether the DB is accessible by selecting some data. --->
        <cfquery name="testDB" dataSource="cfdocexamples" maxrows="2">
            SELECT Emp_ID FROM employee
        </cfquery>
        <!--- If you get a database error, report an error to the user, log the
                error information, and do not start the application. --->
        <cfcatch type="database">
            <cfoutput>
                This application encountered an error<br>
                Please contact support.
            </cfoutput>
            <cflog file="#This.Name#" type="error"
                text="cfdocexamples DB not available. message: #cfcatch.message#
    Detail: #cfcatch.detail# Native Error: #cfcatch.NativeErrorCode#" >
            <cfreturn False>
        </cfcatch>
    </cftry>
    <cflog file="#This.Name#" type="Information" text="Application Started">
    <!--- You do not have to lock code in the onApplicationStart method that sets
            Application scope variables. --->
    <cfscript>
        Application.availableResources=0;
        Application.counter1=1;
    </cfscript>
    <cfreturn True>
</cffunction>
```

onCFCRequest

説明
CFC リクエストに基づいてアプリケーションに対する HTTP 呼び出しまたは AMF 呼び出しを代行受信します。

シンタックス

```
<cffunction name="oncfcRequest" returnType="void">
          <cfargument type="string" name="cfcname">
          <cfargument type="string" name="method">
          <cfargument type="struct" name="args">
</cffunction>
```

関連項目
メソッドの概要、『ColdFusion アプリケーションの開発』の Handling errors in Application.cfc

パラメータ
ColdFusion は、次のパラメータをこのメソッドに渡します。

パラメータ	説明
cfcname	CFC への完全修飾ドット区切りパスです。
method	呼び出すメソッドの名前です。
args	メソッドを呼び出すときに使用する引数 (構造体) です。

使用方法

onRequest は ColdFusion テンプレートに対するリクエストのみを処理するのに対して、この関数は Ajax、Web サービス、および Flash Remoting に対するリクエストを制御します。

例

Web ルートに onCFCRequest フォルダを作成します。このディレクトリに test.cfc と Application.cfm を配置し、次の URL を使用して CFC への HTTP 呼び出しを実行します。

http://localhost:8500/onCFCRequest/test.cfc?method=foo&arg1=1&arg2=2&arg3=3

この URL を実行すると、メソッド onCFCRequest が呼び出され、関数名 foo が引数 arg1、arg2、および arg3 とともに渡されます。

その後、次の例に示すように test.cfc を呼び出せるようになります。

```
<!--- Application.cfc --->
<cfcomponent>
    <cfset this.name = "oncfcrequest">
        <cffunction name="onCFCRequest">
        <cfargument type="string" name="cfcname" required=true>
        <cfargument type="string" name="method" required=true>
        <cfargument type="struct" name="args" required=true>
        <cflog text="oncfcRequest()">
        <cfdump var="#arguments#" output="console" format="text">
        <cfinvoke
            component = "arguments.cfcname"
            method = "arguments.method"
            returnVariable = "result"
            argumentCollection = "#arguments.args#">
            <cfdump var="#result#" output="console" format="text">
        </cffunction>
        <cffunction name="onRequest" output="yes" access="remote">
        <cfargument type="string" name="targetpage">
        <cflog text="onRequest()">
        </cffunction>
</cfcomponent>
<!--- test.cfc --->
<cfcomponent>
    <cffunction name="foo">
        <cfargument name="arg1" type="string" >
        <cfargument name="arg2" type="string" >
        <cfargument name="arg3" type="string" >
        <cfreturn arguments>
    </cffunction>
</cfcomponent>
```

onError

説明
検出されない例外がアプリケーションで発生したときに実行されます。

シンタックス
```
<cffunction name="onError" returnType="void">
    <cfargument name="Exception" required=true/>
    <cfargument name="EventName" type="String" required=true/>
    ...
</cffunction>
```

関連項目
メソッドの概要、『ColdFusion アプリケーションの開発』の Handling errors in Application.cfc

パラメータ
ColdFusion は、次のパラメータをこのメソッドに渡します。

パラメータ	説明
Exception	ColdFusion の例外オブジェクトです。このオブジェクトの構造体の詳細については、cfcatch の説明にある cfcatch 変数の説明を参照してください。
EventName	例外を生成したイベントハンドラの名前です。リクエストの処理中にエラーが発生し、onRequest メソッドを実装していない場合、EventName は空の文字列になります。

戻り値
このメソッドは値を返しません。cfreturn タグを使用しないでください。

使用方法
このメソッドを使用して、アプリケーション固有の方法でエラーを処理します。このメソッドは、ColdFusion Administrator または cferror タグで設定されたエラーハンドラより優先されます。try/catch ブロックは無効になりません。

onError メソッドで出力を表示できるかどうかは、次に示すようにエラーの発生場所により異なります。

- onApplicationStart、onSessionStart、onRequestStart、onRequest、onRequestEnd の各イベントメソッドの実行中またはリクエストの処理中にエラーが発生した場合、onError メソッドではユーザーにエラーメッセージを表示することができます。
- onApplicationEnd または onSessionEnd のイベントメソッドの実行中にエラーが発生した場合は、使用できるページコンテキストがないため、onError メソッドではユーザーにエラーメッセージを表示できません。ただし、エラーメッセージをログに記録することはできます。

onSessionStart などのスコープ固有のイベントメソッドで onError イベントハンドラが起動された場合、そのスコープ以下のスコープのレベルでのそれ以上の処理は中止されます。onSessionStart メソッドで onError イベントが起動された場合は、たとえば、そのセッションでのそれ以上の処理は中止されますが、アプリケーションでの処理は続行されます。

onError メソッドの処理中に例外が発生した場合、または onError メソッドで cfthrow タグを使用する場合は、ColdFusion の標準のエラー処理メカニズムによって例外が処理されます。このメカニズムとは、Application.cfc 初期化コードの cferror タグで指定された任意のエラーハンドラ、ColdFusion Administrator で指定されたサイト全体のエラーハンドラ、および ColdFusion のデフォルトのエラーページです。したがって、onError メソッドをフィルタとして使用して、選択したエラーを処理することができます。また、ColdFusion のその他のエラー処理テクニックを使用して、残りのエラーを処理することができます。

例

```
<cffunction name="onError">
    <cfargument name="Exception" required=true/>
    <cfargument type="String" name="EventName" required=true/>
    <!--- Log all errors. --->
    <cflog file="#This.Name#" type="error"
            text="Event Name: #Arguments.Eventname#" >
    <cflog file="#This.Name#" type="error"
            text="Message: #Arguments.Exception.message#">
    <cflog file="#This.Name#" type="error"
        text="Root Cause Message: #Arguments.Exception.rootcause.message#">
    <!--- Display an error message if there is a page context. --->
    <cfif NOT (Arguments.EventName IS "onSessionEnd") OR
            (Arguments.EventName IS "onApplicationEnd")>
        <cfoutput>
            <h2>An unexpected error occurred.</h2>
            <p>Please provide the following information to technical support:</p>
            <p>Error Event: #Arguments.EventName#</p>
            <p>Error details:<br>
            <cfdump var=#Arguments.Exception#></p>
        </cfoutput>
    </cfif>
</cffunction>
```

onMissingTemplate

説明

リクエストで存在しない CFML ページが指定されたときに実行します。

シンタックス

```
<cffunction name="onMissingTemplate" returnType="boolean">
    <cfargument type="string" name="targetPage" required=true/>
    ...
    <cfreturn BooleanValue />
</cffunction>
```

関連項目

メソッドの概要、『ColdFusion アプリケーションの開発』の Handling errors in Application.cfc

パラメータ

ColdFusion は、次のパラメータをこのメソッドに渡します。

パラメータ	説明
targetPage	Web ルートからリクエストされた CFML ページまでのパスです。

戻り値

ブール値です。true、または値を返さない場合は、イベントが処理されたことを示します。false の場合は、イベントが処理されなかったことを示します。

使用方法

ColdFusion は、ファイルが見つからない場合、つまり、存在しない CFML ページが URL に指定されている場合に、このメソッドを呼び出します。

onMissingTemplate 関数は、イベントが処理されたことを示す場合は true、イベントが処理されなかったことを示す場合は false を返す必要があります。この関数が値を返さない場合は、true と見なされます。この関数が false を返した場合、ColdFusion は標準のエラーハンドラを呼び出します。onMissingTemplate 関数内でエラーが発生した場合、エラーハンドラは呼び出されません。したがって、見つからないテンプレートのハンドラで try/catch ブロックを使用する必要があります。catch ブロックでエラーを処理できない場合は、この関数の戻り値を false に設定して、標準のエラーハンドラでエラーをレポートできるようにする必要があります。

onMissingTemplate 関数が呼び出された場合は、必要に応じて、onApplicationStart イベントハンドラと onSessionStart イベントハンドラが最初に呼び出されます。ただし、onRequestStart、onRequest、および onRequestEnd の各ハンドラは呼び出されず、リクエストの処理は onMissingTemplate ハンドラが返されると終了します。

onMissingTemplate 関数では、Application スコープ、Session スコープ、Client スコープなど、すべての標準スコープを使用できます (有効化されている場合)。

ページのコンテンツを onMissingTemplate 関数に含めるには、cfinclude タグを使用します。他のメソッド (cflocation、GetPageContext().forward()、GetPageContext().include() などのタグや関数) を使用して他のページのコンテンツを含めたり、リダイレクトしないでください。

次のすべての条件を満たす場合は、This.welcomeFileList 変数を使用して、この関数が実行されないようにします。

- Web サーバーで、CFML ファイル (index.cfm など) が少なくとも 1 つ含まれたウェルカムファイルリストを使用し、ユーザーがディレクトリ名で終わる URL を入力した場合に CFML ファイルへのアクセスを試みる。
- Web サーバーは、CFML ページが存在するかどうかを最初に判断せずに、ウェルカムファイルリストの CFML ページに対するリクエストを ColdFusion に送信する。
- リストのファイルが存在しない Web ディレクトリのユーザーによる参照を許可する必要がある。

詳細については、アプリケーション変数の welcomeFileList を参照してください。

例

```
<!--- The web.xml welcome-file-list includes index.cfm.
    To allow web browsing, specify index.cfm in This.welcomFileList. --->
<cfset This.welcomeFileList="index.cfm">

<cffunction name="onMissingTemplate">
    <cfargument name="targetPage" type="string" required=true/>
    <!--- Use a try block to catch errors. --->
    <cftry>
        <!--- Log all errors. --->
        <cflog type="error" text="Missing template: #Arguments.targetPage#">
        <!--- Display an error message. --->
        <cfoutput>
            <h3>#Arguments.targetPage# could not be found.</h3>
            <p>You requested a non-existent ColdFusion page.<br />
                Please check the URL.</p>
        </cfoutput>
        <cfreturn true />
        <!--- If an error occurs, return false and the default error
            handler will run. --->
        <cfcatch>
            <cfreturn false />
        </cfcatch>
    </cftry>
</cffunction>
```

onRequest

説明

onRequestStart イベントハンドラの後、リクエストの開始時に実行されます。このメソッドを実装した場合、このメソッドではリクエストされたページを明示的に呼び出して処理する**必要があります**。

シンタックス

```
<cffunction name="onRequest" returnType="void">
    <cfargument name="targetPage" type="String" required=true/>
    ...
    <cfinclude template="#Arguments.targetPage#">
    ...
</cffunction>
```

関連項目

onRequestStart、onRequestEnd、メソッドの概要、『ColdFusion アプリケーションの開発』の Managing requests in Application.cfc

パラメータ

ColdFusion は、次のパラメータをこのメソッドに渡します。

パラメータ	説明
targetPage	Web ルートからリクエストされたページまでのパスです。

戻り値

このメソッドは値を返しません。cfreturn タグを使用しないでください。

使用方法

このイベントハンドラは、CFML ページリクエスト (ブラウザを使用してリクエストされた .cfm ページ) 用のオプションのリクエストフィルタメカニズムを提供します。このメカニズムを使用して、対象ページへのリクエストを遮断し、リクエストされたページを実行するデフォルトの動作を無効にします。次のルールは、onRequest メソッドの使用対象および使用方法を指定します。

- このメソッドは、次の条件が当てはまる場合にのみ実装します。
    - この Application.cfc の影響を受けるディレクトリおよびサブディレクトリに CFM ファイルが格納されているが、それらのディレクトリおよびサブディレクトリに、Web サービスまたは AJAX バインドとしてアクセスされる CFC ファイルや、Flash Remoting またはイベントゲートウェイを使用してアクセスすることを目的とした CFC ファイルが格納されていない場合。
    - リクエストを遮断して特別な方法で処理する必要がある場合。
- このメソッドを実装しない場合、ColdFusion は対象ページ (または、Web サービスの CFC、Flash Remoting、イベントゲートウェイイベント) を自動的に呼び出します。
- このメソッドを実装した場合、このメソッドで通常は cfinclude タグを使用することにより、対象ページを明示的に呼び出す**必要があります**。
- Web サービスを実装して Flash Remoting やイベントゲートウェイ対するリクエストを処理する .cfc ファイルに影響を与える "Application.cfc" ファイルでは、onRequest メソッドを実装**しないでください**。このメソッドを実装した場合、ColdFusion はリクエストを実行しません。
- このメソッドのコードは、対象ページに対する呼び出しよりも前に置くと onRequestStart メソッドと同じ関数を実行することが可能で、Variables スコープを対象ページと共有します。

- このメソッドのコードは、対象ページに対する呼び出しよりも後に置くと onRequestEnd メソッドと同じ関数を実行することが可能で、Variables スコープを対象ページと共有します。
- このメソッドを実装すると、onRequestStart メソッドおよび onRequestEnd メソッドを実装することもできます。

このメソッドを使用して、すべてのリクエストに必要な前処理を実行することもできます。一般的な使用の例としては、リクエストページコンテンツのフィルタリングと修正 (不要な空白の削除など)、または使用可能なパラメータに基づいて表示する正しいページを判別する切り替えメカニズムの作成などがあります。

例

```
<cffunction name="onRequest">
    <cfargument name="targetPage" type="String" required=true/>
    <cfset var content="">
    <cfsavecontent variable="content">
        <cfinclude template="#Arguments.targetPage#">
    </cfsavecontent>
    <cfoutput>
        #replace(content, "report", "MyCompany Quarterly Report", "all")#
    </cfoutput>
</cffunction>
```

onRequestEnd

説明

他のすべての CFML コードの後、リクエストの最後に実行されます。

シンタックス

```
<cffunction name="onRequestEnd" returnType="void">
    <cfargument type="String" name="targetPage" required=true/>
    ...
</cffunction>
```

関連項目

onRequestStart、onRequest、メソッドの概要、『ColdFusion アプリケーションの開発』の Managing requests in Application.cfc

パラメータ

ColdFusion は、次のパラメータをこのメソッドに渡します。

パラメータ	説明
targetPage	Web ルートからリクエストされたページまでのパスです。

戻り値

このメソッドは値を返しません。cfreturn タグを使用しないでください。

使用方法

このメソッドには、onRequestEnd.cfm ページと同じ目的があります (使用するアプリケーション用の "Application.cfc" ファイルがある場合、onRequestEnd.cfm ページは使用できません)。このメソッドは、リクエストが終了する前に稼動します。したがって、ページコンテキストへのアクセス、および出力の生成が可能です。

このメソッドは、パフォーマンスの測定値を収集するときや、ダイナミックなフッタ情報を表示するときに役立つ場合があります。

このメソッドでは、リクエストされたページを呼び出す onRequest メソッドが "Application.cfc" ファイルに含まれている場合に限り、リクエストされたページの Variables スコープにアクセスできます。"Application.cfc" ファイルに onRequest メソッドがない場合でも、Request スコープ変数を使用して、リクエストされたページとの間でデータを共有することができます。

このメソッドを明示的に呼び出した場合、ColdFusion はリクエストを終了しませんが、このメソッドコードを実行します。

例

次の例では、ユーザーがログインしたかどうかに基づいて表示される 2 つのフッタページの一方を表示します。

Application.cfc の onRequestEnd メソッドには、次のコードが含まれています。

```
<cffunction name="onRequestEnd">
    <cfargument type="String" name="targetPage" required=true/>
    <cfset theAuthuser=getauthuser()>
    <cfif theAuthUser NEQ "">
        <cfinclude template="authuserfooter.cfm">
    <cfelse>
        <cfinclude template="noauthuserfooter.cfm">
    </cfif>
</cffunction>
```

シンプルな authuserfooter.cfm ページは、次のコードで構成されています。

```
<cfoutput>
    <h3>Thank you for shopping at our store, #theAuthUser#!</h3>
</cfoutput>
```

シンプルな noauthuserfooter.cfm ページは、次のコードで構成されています。

```
<cfoutput>
    <h3>Remember, only registered users get all our benefits!</h3>
</cfoutput>
```

この例をテストするには、ユーザーログインのためのコードを実装します。あるいは、onRequestStart の Application.cfc メソッドで次の行を使用して、または使用せずにこの例を試します。

```
<cfloginuser name="Robert Smith" password="secret" roles="customer">
```

onRequestStart

説明

リクエストの開始時に実行されます。

シンタックス

```
<cffunction name="onRequestStart" returnType="boolean">
    <cfargument type="String" name="targetPage" required=true/>
    ...
    <cfreturn Boolean>
</cffunction>
```

関連項目

onRequest、onRequestEnd、メソッドの概要、『ColdFusion アプリケーションの開発』の Managing requests in Application.cfc

パラメータ
ColdFusion は、次のパラメータをこのメソッドに渡します。

パラメータ	説明
targetPage	Web ルートからリクエストされたページまでのパスです。

戻り値
ブール値です。ColdFusion でリクエストが処理されないようにする場合は false を返します。cffunction タグの returntype 属性を省略した場合、true を明示的に返す必要はありません。

使用方法
このメソッドはリクエストの開始時に実行されます。これはユーザーの承認 (ログイン処理) や、リクエスト固有の変数の初期化 (パフォーマンス統計の収集など) に適しています。

このメソッドで例外が発生した場合 (cfthrow タグを使用する場合など)、ColdFusion はそのエラーを処理し、リクエストをそれ以上は処理しません。

このメソッドを明示的に呼び出した場合、ColdFusion はリクエストを開始しませんが、メソッドコードを実行します。

このメソッドでは、リクエストされたページを呼び出す onRequest メソッドが "Application.cfc" ファイルに含まれている場合に限り、リクエストされたページの Variables スコープにアクセスできます。"Application.cfc" ファイルに onRequest メソッドがない場合でも、Request スコープ変数を使用して、リクエストされたページとの間でデータを共有することができます。

例
この例では、ColdFusion Dreamweaver の Login ウィザードで生成した認証コードを使用して、ユーザーのログインを確認します。このウィザードでは、Application.cfm にのみ適したコードを生成します。このコードを Application.cfc で使用するには、生成された Application.cfm を削除します。

```
<cffunction name="onRequestStart">
    <cfargument name="requestname" required=true/>
    <!--- Authentication code, generated by the Dreamweaver Login wizard.
    <cfinclude template="mm_wizard_application_include.cfm">
    <!--- Regular maintenance is done late at night. During those hours, tell
             people to come back later, and do not process the request further. --->
    <cfscript>
        if ((Hour(now()) gt 1) and (Hour(now()) lt 3)) {
            WriteOutput("The system is undergoing periodic maintenance.
                Please return after 3:00 AM Eastern time.");
            return false;
        } else {
            this.start=now();
            return true;
        }
    </cfscript>
</cffunction>
```

onSessionEnd

説明
セッションの終了時に実行されます。

シンタックス

```
<cffunction name="onSessionEnd" returnType="void">
    <cfargument name="SessionScope" required=True/>
    <cfargument name="ApplicationScope" required=False/>
    ...
</cffunction>
```

関連項目

onSessionStart、メソッドの概要、『ColdFusion アプリケーションの開発』の Managing sessions in Application.cfc

パラメータ

ColdFusion は、次のパラメータをこのメソッドに渡します。

パラメータ	説明
SessionScope	Session スコープです。
ApplicationScope	Application スコープです。

戻り値

このメソッドは値を返しません。cfreturn タグを使用しないでください。

使用方法

このメソッドは、セッション終了時に任意のクリーンアップ処理を実行するために使用します。アクティブでない状態のままセッションタイムアウト期間が経過すると、セッションは終了します。たとえば、ショッピングカートの内容やユーザーが注文を完了したかどうかなどのセッション関連データをデータベースに保存したり、その他の必要な処理をユーザーのステータスに応じて実行したりすることができます。セッションの終了またはその他のセッション関連情報を診断目的でファイルに記録する必要がある場合もあります。

このメソッドを明示的に呼び出した場合、ColdFusion はセッションを終了しません。メソッドコードを実行しますが、セッションをロックしません。

このメソッドはリクエストに関連付けられていないので、ユーザーページにデータを表示することはできません。

共有スコープ変数には次のようにアクセスすることができます。

- Session スコープにアクセスするには、**SessionScope** パラメータを使用します。Session スコープを直接参照することはできません。たとえば、Session.myVariable ではなく、Arguments.SessionScope.myVariable を使用します。
- Application スコープにアクセスするには、**ApplicationScope** パラメータを使用する必要があります。Application スコープを直接参照することはできません。たとえば、Application.myVariable ではなく、Arguments.ApplicationScope.myVariable を使用します。例に示すように、Application スコープ内の変数を参照する場合、名前付きのロックを使用します。
- Server スコープには直接アクセスできます。たとえば、Server.myVariable を使用します。
- Request スコープにはアクセスできません。

アプリケーションが終了しても、セッションは終了しないので onSessionEnd メソッドは呼び出されません。ただし、アクティブなアプリケーションがない場合、onSessionEnd は実行されません。

例

次のメソッドでは、Application スコープのセッションカウント変数を減算し、セッションの長さをログに記録します。

```
<cffunction name="onSessionEnd">
    <cfargument name = "SessionScope" required=true/>
    <cfargument name = "AppScope" required=true/>
    <cfset var sessionLength = TimeFormat(Now() - SessionScope.started,
        "H:mm:ss")>
    <cflock name="AppLock" timeout="5" type="Exclusive">
        <cfset Arguments.AppScope.sessions = Arguments.AppScope.sessions - 1>
    </cflock>
    <cflog file="#This.Name#" type="Information"
        text="Session #Arguments.SessionScope.sessionid# ended. Length: #sessionLength# Active sessions:
#Arguments.AppScope.sessions#">
</cffunction>
```

onSessionStart

説明
セッションの開始時に実行されます。

シンタックス
```
<cffunction name="onSessionStart" returnType="void">
    ...
</cffunction>
```

関連項目
onSessionEnd、メソッドの概要、『ColdFusion アプリケーションの開発』の Managing sessions in Application.cfc

戻り値
このメソッドは値を返しません。cfreturn タグを使用しないでください。

使用方法
このメソッドは、ショッピングカートなどの Session スコープデータを初期化する場合、またはアクティブなセッションの数を追跡するなどのセッション固有の Application スコープ変数を設定する場合に役立ちます。このメソッドを使用して Session スコープの変数を設定するために Session スコープをロックする必要はありません。

このメソッドを明示的に呼び出した場合、ColdFusion はセッションを開始しません。メソッドコードを実行しますが、Session スコープをロックしません。

このメソッドでは、リクエストされたページを呼び出す onRequest メソッドが "Application.cfc" ファイルに含まれている場合に限り、リクエストされたページの Variables スコープにアクセスできます。

例
次の onSessionStart の例では、Session スコープ変数を初期化し、アクティブなセッションの Application スコープのカウンタをインクリメントします。

```
<cffunction name="onSessionStart">
    <cfscript>
        Session.started = now();
        Session.shoppingCart = StructNew();
        Session.shoppingCart.items =0;
    </cfscript>
        <cflock scope="Application" timeout="5" type="Exclusive">
            <cfset Application.sessions = Application.sessions + 1>
    </cflock>
</cffunction>
```

onServerStart

ColdFusion では、サーバーの起動時にのみ実行される onServerStart メソッドを持つ CFC がサポートされています。onServerStart メソッドはパラメータを取りません。また、CFC で唯一の関数です。この関数は、アプリケーションのインスタンス化、ロギングの設定、スケジューラの設定など、アプリケーションに依存しないタスクを使用する場合に便利です。

デフォルトでは、ColdFusion は cf_webroot/Server.cfc で onServerStart メソッドを検索します。別のファイルパスを指定するには :

1. ColdFusion Administrator を起動します。
2. ColdFusion Administrator の [サーバーの設定]-[設定] をクリックします。
3. [設定] ページで Web ルート下の絶対ファイルパス (c:¥Server.cfc など) を指定します。Web ルート下のドット区切りのパス (a.b.Server など) を使用することもできます。

注意 : 絶対パスを使用する場合、ファイル名の末尾を .cfc にする必要があります。相対パスまたはドット区切りパスを使用する場合は、末尾に .cfc 接尾辞を付けないでください。

onServerStart メソッドを有効または無効にするには、[設定] ページでオプションを選択します。デフォルトでは、メソッドは無効になっています。

onServerStart メソッドに対しては、タイムアウト制限 (秒単位) を指定することもできます。タイムアウト制限を指定すると、サーバー起動時にこのメソッドを実行できる期間が決定されます。この設定は、server.cfc で指定できます。

onServerStart メソッドでは、ほとんどの CFML 機能を使用できますが、完全なサーバー起動を必要とする機能は使用できません。たとえば、このメソッドでは、同じサーバー上の場所を指定する URL を含む cfhttp タグは使用できません。メソッドで Application または Request スコープ変数を使用することもできません。

デフォルトでは、すべてのエラー (serverCFC エラーを含む) は <ColdFusion のホーム >/WEB-INF/cfusion/logs ディレクトリ (スタンドアローンの場合)、および < アプリケーションサーバーのルートディレクトリ >/logs ディレクトリ (J2EE 設定の場合) にロギングされます。

ColdFusion Administrator の [デバッグとロギング]-[ロギングの設定] でログディレクトリ設定を指定することで、ロギングに別の場所を指定することもできます。

server.log ファイルには、サーバーの起動情報が保存されます。そのため、server.CFC の起動エラーはこのファイルに記録されますが、エラーの詳細については、exception.log ファイルを参照する必要があります。また、サーバーの起動情報は < アプリケーションサーバーのルートディレクトリ >/logs ディレクトリにロギングされます。

WebSphere の場合は、SystemOut.log ファイルにロギングされます。

第 9 章：ColdFusion イベントゲートウェイリファレンス

Java による ColdFusion カスタム CFX の構築では、Java インターフェイスを使用できます。

注意：CFML 関数 GetGatewayHelper および SendGatewayMessage もゲートウェイアプリケーション開発に適用されます。

ゲートウェイ開発のインターフェイスとクラス

ColdFusion イベントゲートウェイシステムは、coldfusion.eventgateway パッケージで定義されています。ゲートウェイ開発者は、2 つのインターフェイスを実装し、複数のクラスを使用します。これらのインターフェイスとクラスは次のとおりです。

関連項目

1542 ページの「ゲートウェイインターフェイス」

1551 ページの「GatewayHelper インターフェイス」

1552 ページの「GatewayServices クラス」

1556 ページの「CFEvent クラス」

1569 ページの「Logger クラス」

ゲートウェイインターフェイス

`coldfusion.eventgateway.Gateway`

ColdFusion イベントゲートウェイを実装するためのインターフェイスです。

このインターフェイスを実装するクラスにより、ColdFusion アプリケーションで使用できる ColdFusion イベントゲートウェイのタイプが定義されます。このクラスは、次のメソッドを実装します。

シグネチャ	説明
GatewayName([String id[, StringconfigFile]])	ゲートウェイコンストラクタです。
String getGatewayID()	ゲートウェイ ID を返します。
GatewayHelper getHelper()	このゲートウェイタイプについて GatewayHelper クラスのインスタンスを返します。ゲートウェイに GatewayHelper クラスがない場合は null を返します。
int getStatus()	イベントゲートウェイのステータスを取得します。
String outgoingMessage(coldfusion.eventgateway.CFEvent cfmessage)	ColdFusion によって送信されたメッセージを処理の対象とし、メッセージがメッセージレシーバーに送信されるように処理します。
void restart()	実行中のイベントゲートウェイを再起動します。

シグネチャ	説明
void setCFCListeners(String[] listeners)	イベントゲートウェイからの着信メッセージをリスンする CFC を識別します。
void setGatewayID(String id)	ゲートウェイインスタンスを一意に識別するゲートウェイ ID を設定します。
void start()	イベントゲートウェイを起動します。
void stop()	イベントゲートウェイを停止します。

コンストラクタ

説明
ゲートウェイをインスタンス化します。

カテゴリ
イベントゲートウェイの開発

シンタックス
```
public void gatewayName()
public void gatewayName(String id)
public void gatewayName(String id, String configFile)
```

関連項目
setGatewayID、『ColdFusion アプリケーションの開発』の Class constructor

パラメータ

パラメータ	説明
id	ゲートウェイインスタンスの識別子です。
configFile	ゲートウェイ設定ファイルへの絶対パスです。

使用方法
ゲートウェイに設定ファイルが必要な場合は、これら 2 つのパラメータがあるコンストラクタを使用します。それ以外の場合は、デフォルトのコンストラクタまたはパラメータが 1 つのバージョンのいずれかを使用できます。ColdFusion では、常に setGatewayID メソッドを使用して ID を設定します。

例
次の例は、ColdFusion SocketGateway クラスに実装された、2 つの引数を取るコンストラクタを示しています。

```
public SocketGateway(String id, String configpath) {
        propsFilePath=configpath;
        try {
            FileInputStream propsFile = new FileInputStream(propsFilePath);
            properties.load(propsFile);
            propsFile.close();
            this.loadProperties();
        }
        catch (FileNotFoundException f) {
            // do nothing. use default value for port.
        }
        catch (IOException e) {
            e.printStackTrace();
        }
        gatewayID = id;
        gatewayService = GatewayServices.getGatewayServices();
    }
```

getGatewayID

説明
ゲートウェイインスタンスを識別するゲートウェイ ID を返します。

カテゴリ
イベントゲートウェイの開発

シンタックス

```
public String getGatewayID()
```

関連項目
setGatewayID、『ColdFusion アプリケーションの開発』の Providing Gateway class service and information routines

使用方法
このメソッドは、setGatewayID メソッドによって設定された文字列値を返します。

例
次の例は、ColdFusion SocketGateway クラスの getGatewayID メソッドです。

```
public String getGatewayID()
    {
        return gatewayID;
    }
```

getHelper

説明
ゲートウェイタイプについて gatewayHelper クラスが存在する場合、その gatewayHelper クラスを返します。

カテゴリ
イベントゲートウェイの開発

シンタックス

```
public GatewayHelper getHelper()
```

関連項目

GatewayHelper インターフェイス、『ColdFusion アプリケーションの開発』の Providing Gateway class service and information routines

戻り値

coldfusion.eventgateway.GatewayHelper クラスインスタンス。ゲートウェイに GatewayHelper クラスがない場合は null。

使用方法

ColdFusion は、ColdFusion アプリケーションが CFML GetGatewayHelper 関数を呼び出すときにこのメソッドを呼び出します。ColdFusion アプリケーションは、gatewayHelper オブジェクトのメソッドを使用して、インスタントメッセージバディ管理のメソッドなどのゲートウェイ固有のユーティリティメソッドを呼び出します。

例

次の例は、ColdFusion SocketGateway クラスの getHelper メソッドです。

```
public GatewayHelper getHelper()
    {
        // SocketHelper class implements the GatewayHelper interface
        return new SocketHelper();
    }
```

getStatus

説明

ゲートウェイステータスを返します。

カテゴリ

イベントゲートウェイの開発

シンタックス

```
public int getStatus()
```

関連項目

『ColdFusion アプリケーションの開発』の Providing Gateway class service and information routines

戻り値

ステータスの整数値。ゲートウェイインターフェイスは、次のステータス定数を定義します。

- STARTING
- RUNNING
- STOPPING
- STOPPED
- FAILED

例
次の例は、ColdFusion SocketGateway クラスの getStatus メソッドです。

```
public int getStatus()
    {
        return status;
    }
```

outgoingMessage

説明
メッセージを ColdFusion からメッセージレシーバーに送信します。

カテゴリ
イベントゲートウェイの開発

シンタックス

```
public String outgoingMessage(coldfusion.eventgateway.CFEvent message)
```

関連項目
『ColdFusion アプリケーションの開発』の Responding to a ColdFusion function or listener CFC

パラメータ

パラメータ	説明
message	送信するメッセージを含む coldfusion.eventgateway.CFEvent インスタンスです。

戻り値
メッセージ ID やステータスインジケータなどのゲートウェイ固有の文字列です。

使用方法
このメソッドは、ColdFusion によって送信されたメッセージを処理の対象とし、ゲートウェイタイプに応じた必要な処理を加えて、メッセージを (通常は外部の) メッセージレシーバーに送信します。リスナー CFC のリスナーメソッドがメッセージを返すとき、または ColdFusion アプリケーションが SendGatewayMessage 関数を呼び出すときに、ColdFusion はこのメソッドを呼び出します。ColdFusion は、このメソッドによって返された String を、CFML SendGatewayMessage 関数の戻り値として渡します。

例
次の例は、ColdFusion SocketGateway クラスの outgoingMessage メソッドです。

```
public String outgoingMessage(coldfusion.eventgateway.CFEvent cfmsg)
{
    String retcode="ok";
    // Get the table of data returned from the event handler
    Map data = cfmsg.getData();
    String message = (String) data.get("MESSAGE");
    // find the right socket to write to from the socketRegistry hashtable
    if (cfmsg.getOriginatorID() != null && message != null)
    {
        SocketServerThread st =
            ((SocketServerThread)socketRegistry.get(cfmsg.getOriginatorID()));
        if(st != null)
            st.writeOutput(message);
        else
        {
            log.error("Cannot send outgoing message. OriginatorID '" +
                cfmsg.getOriginatorID() + "' is not a valid socket id.");
            retcode="failed";
        }
    }
    else if (data.get("OriginatorID") != null && message != null)
    {
        SocketServerThread st =
            ((SocketServerThread)socketRegistry.get(data.get("OriginatorID")));
        if(st != null)
            st.writeOutput(message);
        else
        {
            log.error("Cannot send outgoing message. OriginatorID '" +
                data.get("OriginatorID") + "' is not a valid socket id.");
            retcode="failed";
        }
    }
    else
    {
        log.error("Cannot send outgoing message. OriginatorID/MESSAGE is not
            available.");
        retcode="failed";
    }
    return retcode;
}
```

restart

説明
実行中のゲートウェイをいったん停止してから起動します。

カテゴリ
イベントゲートウェイの開発

シンタックス

```
public void restart()
```

関連項目
start、stop

使用方法
ほとんどの場合、start メソッドが後に続く stop メッドの呼び出しとして、このメソッドを実装します。ただし、ゲートウェイのタイプに応じて restart メソッドを最適化できる場合もあります。

例
次の例は、ColdFusion SocketGateway クラスの restart メソッドです。

```
public void restart()
{
    stop();
    start();
}
```

setCFCListeners

説明
ゲートウェイがメッセージを送信する先のリスナー CFC の配列を設定します。

カテゴリ
イベントゲートウェイの開発

シンタックス
```
public void setCFCListeners(String[] listeners)
```

関連項目
コンストラクタ、getGatewayID、setCFCPath、『ColdFusion アプリケーションの開発』の Providing Gateway class service and information routines

パラメータ

パラメータ	説明
listeners	ゲートウェイがイベントの取得時にメッセージを転送する先の CFC への絶対ファイルパスの配列です。

使用方法
ColdFusion は、ゲートウェイライセンスの開始時に、ColdFusion Administrator でインスタンスのリスナーリスト内の名前を使ってメソッドを呼び出します。ゲートウェイの実行中に ColdFusion Administrator のリスナーリストが変更された場合も、ColdFusion はこのメソッドを呼び出すことができます。

例
次の例は、ColdFusion SocketGateway クラスの setCFCListeners メソッドです。

```
public void setCFCListeners(String[] listeners)
{
    ArrayList aListeners = new ArrayList();
    for(int i = 0; i<listeners.length; i++)
    {
        aListeners.add(listeners[i]);
    }
        // Try not to pull the rug out from underneath a running message
    synchronized (cfcListeners)
    {
        cfcListeners = aListeners;
    }
}
```

setGatewayID

説明
ゲートウェイインスタンスを一意に識別するゲートウェイ ID を設定します。

カテゴリ
イベントゲートウェイの開発

シンタックス
```
public void setGatewayID(String id)
```

関連項目
コンストラクタ、getGatewayID、『ColdFusion アプリケーションの開発』の Providing Gateway class service and information routines

パラメータ

パラメータ	説明
ID	このゲートウェイインスタンスの識別子です。

使用方法
このメソッドは、getGatewayID メソッドによって返される文字列値を設定します。ColdFusion はこのメソッドを呼び出し、ゲートウェイコンストラクタでゲートウェイ ID が設定される場合であっても、イベントゲートウェイの開始前に、ColdFusion Administrator のゲートウェイインスタンス設定で指定された値を持つゲートウェイ ID を設定します。

例
次の例は、ColdFusion SocketGateway クラスの setGatewayID メソッドです。

```
public void setGatewayID(String id)
    {
        gatewayID = id;
    }
```

start

説明
ゲートウェイの実行を開始します。

カテゴリ
イベントゲートウェイの開発

シンタックス
```
public void start()
```

関連項目
restart、stop、『ColdFusion アプリケーションの開発』の Starting, stopping, and restarting the event gateway

使用方法
必要な初期化を実行することにより、ゲートウェイを開始します。このメソッドは、ゲートウェイのイベントソースを監視するリスナースレッドを開始します。ColdFusion Administrator は、ゲートウェイインスタンスを開始するときにこの関数を呼び出します。

このメソッドは、getStatus メソッドで返されるステータス情報を更新し、ゲートウェイを開始することやゲートウェイを実行中であることを示します。

ColdFusion Administrator の [ゲートウェイタイプ] ページでは、ゲートウェイスタートアップのタイムアウトの指定、およびスタートアップのタイムアウト時にゲートウェイを中断するかどうかの指定ができます。中断のオプションを選択した場合、タイムアウト期間内に start メソッドが情報を返さないときは、ColdFusion はこの関数を呼び出すスレッドを中断します。

例
次の例は、ColdFusion SocketGateway クラスの restart メソッドです。

```
public void start()
    {
        status = STARTING;
        listening=true;
        // Start up event generator thread
        Runnable r = new Runnable()
        {
            public void run()
            {
                socketServer();
            }
        };
        Thread t = new Thread(r);
        t.start();
        status = RUNNING;
    }
```

stop

説明
実行中のゲートウェイを停止します。

カテゴリ
イベントゲートウェイの開発

シンタックス
```
public void stop()
```

関連項目
restart、start、『ColdFusion アプリケーションの開発』の Starting, stopping, and restarting the event gateway

使用方法
必要なクリーンアップ操作を実行することにより、ゲートウェイを停止します。このメソッドは、任意のリスナースレッド、またはゲートウェイのイベントソースを監視するスレッドを停止し、その他のリソースを解放します。ColdFusion Administrator は、ゲートウェイインスタンスを停止するときにこの関数を呼び出します。

このメソッドは、getStatus メソッドで返されるステータス情報を更新し、ゲートウェイを停止することやゲートウェイを停止したことを示します。

例
次の例は、ColdFusion SocketGateway クラスの stop メソッドです。

```
public void stop()
    {
        status = STOPPING;
        listening=false;
        Enumeration e = socketRegistry.elements();
        while (e.hasMoreElements()) {
            try
            {
                ((SocketServerThread)e.nextElement()).socket.close();
            }
            catch (IOException e1) {
                e1.printStackTrace();
            }
        }
        if (serverSocket != null) {
            try
            {
                serverSocket.close();
            }
            catch (IOException e1) {
            }
            serverSocket = null;
        }
        status = STOPPED;
    }
```

GatewayHelper インターフェイス

coldfusion.eventgateway.GatewayHelper

ColdFusion には、coldfusion.eventgateway.GatewayHelper Java マーカーインターフェイスが含まれています。メソッドはありません。このインターフェイスを実装することで、ColdFusion アプリケーションまたはリスナー CFC にゲートウェイ固有のユーティリティメソッドを提供するクラスを定義します。たとえば、インスタントメッセージングイベント

ゲートウェイは、ヘルパクラスを使用して、バディリスト管理メソッドをアプリケーションに提供することができます。Gateway クラスは、ヘルパクラスまたは null (インターフェイスを実装しない場合) を返す getHelper メソッドを実装する必要があります。

GatewayHelper クラスの詳細については、GatewayHelper class を参照してください。

GatewayServices クラス

```
coldfusion.eventgateway.GatewayServices
```

Gateway クラスは coldfusion.eventgateway.GatewayServices クラスを使用して、ColdFusion イベントゲートウェイサービスを操作します。このクラスには、次のメソッドがあります。

シグネチャ	**説明**
GatewayServices getGatewayServices()	GatewayServices オブジェクトを返すスタティックメソッド。
boolean addEvent(CFEvent **msg**)	リスナー CFC に送信するために、CFEvent インスタンスを ColdFusion に送信します。
coldfusion.eventgateway.LoggergetLogger([String *logfile*])	イベントゲートウェイがファイルに情報を記録するときに使用できる ColdFusion Logger オブジェクトを返します。
int getMaxQueueSize()	ColdFusion Administrator で設定されている、ColdFusion イベントキューの最大サイズを返します。
int getMaxQueueSize()	すべてのゲートウェイのすべてのメッセージを処理する ColdFusion イベントキューの現在のサイズを返します。

getGatewayServices

説明
GatewayServices オブジェクトを返すスタティックメソッド。ゲートウェイコードは、必要に応じていつでもこのメソッドを呼び出すことができます。

カテゴリ
イベントゲートウェイの開発

シンタックス

```
GatewayServices getGatewayServices()
```

関連項目
『ColdFusion アプリケーションの開発』の GatewayServices class

戻り値
GatewayServices オブジェクト

使用方法
Gateway コンストラクタは、このメソッドを呼び出して、GatewayServices クラスおよびそのメソッドへの有用なリファレンスを取得することができます。

例
次の Socket ゲートウェイコンストラクタコードは、GatewayServices 変数を設定します。

```
public SocketGateway(String id)
    {
        gatewayID = id;
        gatewayService = GatewayServices.getGatewayServices();
    }
```

次のような GatewayServices メソッドの呼び出しで戻り値を使用します。

```
boolean sent = gatewayService.addEvent(event);
```

addEvent

説明
リスナー CFC に送信するために、CFEvent インスタンスを ColdFusion に送信します。

カテゴリ
イベントゲートウェイの開発

シンタックス

```
boolean addEvent(CFEvent msg)
```

関連項目
getMaxQueueSize、getMaxQueueSize、『ColdFusion アプリケーションの開発』の Responding to incoming messages

パラメータ

パラメータ	説明
msg	リスナー CFC に配達するためにキューに入れられたメッセージを含む CFEvent オブジェクトです。

戻り値
イベントが配信のためにゲートウェイサービスキューに追加された場合は true、それ以外の場合は false。したがって、true が返されても、メッセージが配信されたことを示しているわけではありません。

使用方法
イベントゲートウェイは、このメソッドを使用して、着信メッセージを処理用アプリケーションに送信する必要があります。

例
ColdFusion SocketGateway コードからの次の例では、イベントをすべてのリスナー CFC に送信します。

```
for (int i = 0; i < listeners.length; i++) {
    String path = listeners[i];
    CFEvent event = new CFEvent(gatewayID);
    Hashtable mydata = new Hashtable();
    mydata.put("MESSAGE", theInput);
    event.setData(mydata);
    event.setGatewayType("SocketGateway");
    event.setOriginatorID(theKey);
    event.setCfcMethod(cfcEntryPoint);
    event.setCfcTimeOut(10);
    if (path != null)
        event.setCfcPath(path);
    boolean sent = gatewayService.addEvent(event);
    if (!sent)
        log.error("SocketGateway(" + gatewayID + ") Unable to put message on
            event queue. Message not sent from " + gatewayID + ", thread " +
            theKey + ".Message was " + theInput);
}
```

getLogger

説明
イベントゲートウェイがファイルに情報を記録するときに使用できる ColdFusion Logger オブジェクトを返します。

カテゴリ
イベントゲートウェイの開発

シンタックス
```
coldfusion.eventgateway.Logger getLogger([String logfile])
```

関連項目
Logger クラス、『ColdFusion アプリケーションの開発』の Logging events and using log files

パラメータ

パラメータ	説明
logfile	ColdFusion ログディレクトリにあるログファイルの名前 (拡張子なし) です。ColdFusion は、この名前に .log という拡張子を自動的に付けます。このファイルが存在しない場合、ColdFusion は最初のメッセージを記録するときにこのファイルを作成します。ColdFusion のデフォルトのログファイルは eventgateway.log です。

戻り値
ColdFusion Logger オブジェクト

使用方法
Logger クラスには、debug、info、warn、error、および fatal の 5 つのメソッドがあります。これらのメソッドは、ログメッセージで設定される厳格度レベルに対応しています。各メソッドでは、メッセージ文字列、Throwable クラスオブジェクト、またはこれら両方を取ります。

Throwable オブジェクトをこれらのメソッドに渡すと、ColdFusion は exceptions.log ファイルに例外情報を書き込みます。

例

ColdFusion の DirectoryWatcherGateway の例では、Logger オブジェクトを取得するための次の行がコンストラクタにあります。

```
// We create our own log file, which will be named "watcher.log"
logger = gatewayService.getLogger("watcher");
```

次のコードでは、設定ファイルから情報をロードするルーチンの始めから、このオブジェクトを使用して初期化を記録します。

```
// Load the properties file to get our settings
protected void loadconfig() throws ServiceRuntimeException
{
    // load config
    logger.info("DirectoryWatcher (" + gatewayID + ") Initializing
        DirectoryWatcher gateway with configuration file " + config);
.
.
.
```

getMaxQueueSize

説明

ColdFusion Administrator で設定されている、ColdFusion イベントキューの最大サイズを返します。

カテゴリ

イベントゲートウェイの開発

シンタックス

```
int getMaxQueueSize()
```

関連項目

addEvent、getQueueSize

戻り値

ゲートウェイサービスキューに保持できるメッセージの最大数 (整数)

使用方法

キューの長さがこの値に達すると、addEvent メソッドはそのメッセージをキューに追加しなくなります。このメソッドおよび getQueueSize メソッドを使用して、イベントキューイングの度合いを制御したり、ゲートウェイ全体の問題を診断したりすることができます。

例

次の例では、gatewayService.addEvent メソッドがリスナー CFC に配信するメッセージをキューに入れることができない場合に、キューのサイズ、キューの最大サイズ、およびその他の情報を記録します。エラーメッセージ文字列を作成するための内部メソッドを使用します。

```
boolean sent = gatewayService.addEvent(cfmsg);
if (!sent)
{
    logger.error(RB.getString(this, "IMGateway.cantAddToQueue",
        gatewayType, gatewayID, ((path != null) ? path : "default"),
        Integer.ToString(gatewayService.getQueueSize()),
        Integer.ToString(gatewayService.getMaxQueueSize())));
}
```

getQueueSize

説明
すべてのゲートウェイのすべてのメッセージを処理する ColdFusion イベントキューの現在のサイズを返します。

カテゴリ
イベントゲートウェイの開発

シンタックス
```
int getQueueSize()
```

関連項目
addEvent、getMaxQueueSize

戻り値
CFC への配信を待機している、ゲートウェイメッセージキュー内のメッセージの数 (整数)

使用方法
このメソッドおよび getMaxQueueSize メソッドを使用して、イベントキューイングの度合いを制御したり、ゲートウェイ全体の問題を診断したりすることができます。

例
次の例では、gatewayService.addEvent メソッドがリスナー CFC に配信するメッセージをキューに入れることができない場合に、キューのサイズ、キューの最大サイズ、およびその他の情報を記録します。エラーメッセージ文字列を作成するための内部メソッドを使用します。

```
boolean sent = gatewayService.addEvent(cfmsg);
if (!sent)
{
    logger.error(RB.getString(this, "IMGateway.cantAddToQueue",
        gatewayType, gatewayID, ((path != null) ? path : "default"),
        Integer.ToString(gatewayService.getQueueSize()),
        Integer.ToString(gatewayService.getMaxQueueSize())));
}
```

CFEvent クラス

```
coldfusion.gateway.CFEvent
```

Gateway クラスは、CFEvent インスタンスを送受信して、ColdFusion リスナー CFC またはアプリケーションとの間で通信を実行します。CFEvent インスタンスは CFML CFEvent 構造体 に対応しています。この構造体では、ColdFusion アプリケーションコードがゲートウェイに送信するメッセージ構造体を ColdFusion アプリケーションリスナー CFC メソッドが受信して保持します。

- Gateway は、GatewayServices.addEvent メソッドで CFEvent インスタンスを送信することにより、ColdFusion にメッセージを通知します。
- Gateway は、ColdFusion がゲートウェイの outgoingMessage メソッドを呼び出すときに CFEvent インスタンスを受信します。

CFEvent クラスは、java.util.Hashtable クラスを拡張し、次のメソッドを備えています。

メソッド	説明
CFEvent(String *gatewayID*)	CFEvent のコンストラクタ。
String getGatewayID()	CFEvent コンストラクタで設定されたゲートウェイ ID を返します。
void setCFCMethod(String *method*) String getCFCMethod()	着信メッセージを受信する CFC メソッドの名前を設定または取得します。
void setCFCPath(String *path*) String getCFCPath()	イベントを処理するアプリケーションリスナー CFC へのパスを設定または取得します。
void setCFCTimeout(String *seconds*) String getCFCTimeout()	リスナー CFC がイベントリクエストを処理するためのタイムアウト時間 (秒単位) を設定または取得します。
void setData(Map *data*) Map getData()	メッセージの内容およびその他のゲートウェイ固有の情報が含まれているイベントデータ構造体を設定または取得します。
void setGatewayType(String *type*) String getGatewayType()	SMS などのイベントゲートウェイタイプ識別子を設定または取得します。
void setOriginatorID(String *id*) String getOriginatorID()	メッセージの発信元のゲートウェイ固有またはプロトコル固有の ID を設定または取得します。

CFEvent

説明
CFEvent のコンストラクタ。

カテゴリ
イベントゲートウェイの開発

シンタックス
```
CFEvent(String gatewayID)
```

関連項目
getGatewayID、CFML CFEvent 構造体、『ColdFusion アプリケーションの開発』の CFEvent class

パラメータ

パラメータ	説明
gatewayID	ゲートウェイの ID です。このパラメータはメッセージのソースを示します。Gateway コンストラクタで渡される値、または Gateway setGatewayID メソッドを使って設定される値を指定する必要があります。SMS ゲートウェイ ID は、21 文字以下にする必要があります。

使用方法

このメソッドでは、CFC リスナーメソッドに配信するために gatewayServices.addEvent メソッドの ColdFusion ゲートウェイサービスに送信する、イベントゲートウェイメッセージのコンテナを作成します。

例

次の例では、ColdFusion の非同期 CFML ゲートウェイのコードに基づいて、ゲートウェイが受信したメッセージを CFC に送信します。

```
public String outgoingMessage(coldfusion.eventgateway.CFEvent cfmsg)
{
    // Get the data
    Map data = cfmsg.getData();
    boolean status = true;
    if (data != null)
    {
        // create an event
        CFEvent event = new coldfusion.eventgateway.CFEvent(gatewayID);
        //set the event field values
        event.setGatewayType("CFMLGateway");
        event.setOriginatorID("CFMLGateway");
        event.setData(data);
        // send it to the event service
        status = gatewayService.addEvent(event);
    }
        return new Boolean(status).ToString();
}
```

getCFCMethod

説明

メッセージを処理する CFC メソッドの名前を取得します。

カテゴリ

イベントゲートウェイの開発

シンタックス

```
String getCFCMethod()
```

関連項目

getCFCPath、getCFCTimeout、setCFCMethod、CFML CFEvent 構造体、『ColdFusion アプリケーションの開発』の CFEvent class

戻り値
着信メッセージの場合、setCFCMethod メソッドによる設定どおりに、ゲートウェイサービスがリスナー CFC で呼び出すメソッドの名前。setCFCMethod が呼び出されていない場合、ColdFusion ゲートウェイサービスがデフォルトで使用する onIncomingMessge ではなく、null を返します。ゲートウェイが着信メッセージ上にフィールドを設定した場合、着信メッセージへの応答として CFC が返す発信メッセージにもこのフィールドの CFC メソッド名が含まれます。

使用方法
ほとんどのイベントゲートウェイでは、このメソッドを使用する必要はありません。このメソッドが役立つのは、ゲートウェイが複数の CFC メソッドにメッセージを送信し、どのメソッドが応答しているかを判別する必要がある場合です。

getCFCPath

説明
このメッセージを処理するリスナー CFC へのパスを取得します。

カテゴリ
イベントゲートウェイの開発

シンタックス
```
String getCFCPath()
```

関連項目
getCFCMethod、getCFCTimeout、setCFCPath、CFML CFEvent 構造体、『ColdFusion アプリケーションの開発』の CFEvent class

戻り値
setCFCPath メソッドで設定された、このイベントを処理するアプリケーションリスナー CFC への絶対パス。setCFCPath メソッドが呼び出されていない場合、ColdFusion Administrator で指定してゲートウェイサービスがデフォルトで使用するパスではなく、null を返します。ゲートウェイが着信メッセージ上にフィールドを設定した場合、着信メッセージへの応答として CFC が返す発信メッセージにもこのフィールドの CFC メソッド名が含まれます。

使用方法
ほとんどのイベントゲートウェイでは、このメソッドを使用する必要はありません。このメソッドが役立つのは、ゲートウェイが複数の CFC メソッドにメッセージを送信し、どの CFC が応答しているかを判別する必要がある場合です。

getCFCTimeout

説明
リスナー CFC がイベントリクエストを処理する場合のタイムアウト時間 (秒単位) を取得します。

カテゴリ
イベントゲートウェイの開発

シンタックス
```
String getCFCTimeout()
```

関連項目
getCFCMethod、getCFCPath、setCFCTimeout、CFML CFEvent 構造体、『ColdFusion アプリケーションの開発』の CFEvent class

戻り値
setCFCTimeout メソッドで設定されたリスナー CFC のタイムアウト時間 (秒単位)、または null。

使用方法
ほとんどのゲートウェイでは、この関数を使用する必要はありません。

ColdFusion がイベントを処理するためにリスナー CFC メソッドを呼び出したが、指定されたタイムアウト時間内に CFC がイベントを処理しなかった場合、ColdFusion はこのリクエストを終了し、"application.log" ファイルにエラーを記録します。ColdFusion は、ColdFusion Administrator の [サーバーの設定] ページで設定された [リクエストタイムアウト] の値をデフォルトで使用します。

getData

説明
メッセージの内容およびその他のゲートウェイ固有の情報を含むデータ Map を返します。

カテゴリ
イベントゲートウェイの開発

シンタックス
```
Map getData()
```

関連項目
setData、CFML CFEvent 構造体、『ColdFusion アプリケーションの開発』の CFEvent class

戻り値
イベントデータ構造体、または null。この構造体には、ゲートウェイによって渡されるメッセージの内容、およびゲートウェイ固有のその他の情報が含まれています。

使用方法
このデータ Map の内容は、イベントゲートウェイのタイプに応じて異なります。一般的なフィールドには、メッセージの内容、発信元 ID、宛先 ID などがあり、ゲートウェイ (ColdFusion SMS ゲートウェイなど) が複数のコマンドをサポートする場合はコマンドも含まれます。

注意：返される Map オブジェクトには、大文字と小文字を区別しないキーがあります。

例
次の SocketGateway ゲートウェイの例の outgoingMessage メソッドでは、発信メッセージの CFEvent データフィールドからメッセージの内容を取得します。CFEvent オブジェクトが OriginatorID フィールドを含まない場合、このオブジェクトはデータフィールドから発信元 ID を取得しようとします。

```
public String outgoingMessage(coldfusion.eventgateway.CFEvent cfmsg)
{
    String retcode="ok";
    // Get the table of data returned from the event handler
    Map data = cfmsg.getData();
    String message = (String) data.get("MESSAGE");
    // find the right socket to write to from the socketRegistry hashtable
    if (cfmsg.getOriginatorID() != null)
        ((SocketServerThread)socketRegistry.get(cfmsg.getOriginatorID())).
            writeOutput(message);
    else if (data.get("OriginatorID") != null)
        ((SocketServerThread)socketRegistry.get(data.get("OriginatorID"))).
            writeOutput(message);
    else {
        System.out.println("cannot send outgoing message. OriginatorID is not
            available.");
            retcode="failed";
    }
    return retcode;
}
```

getGatewayID

説明
CFEvent オブジェクトのゲートウェイ ID フィールドを返します。

カテゴリ
イベントゲートウェイの開発

シンタックス
```
String getGatewayID(CFEvent event)
```

関連項目
CFEvent、CFML CFEvent 構造体、『ColdFusion アプリケーションの開発』の CFEvent class

戻り値
CFEvent オブジェクトのゲートウェイ ID、または null。

使用方法
ほとんどのゲートウェイでは、このメソッドを使用する必要はありません。ゲートウェイ ID は CFEvent コンストラクタで設定され、イベントを処理するゲートウェイに通常は対応します。

getGatewayType

説明
CFEvent オブジェクトのゲートウェイタイプフィールドを返します。

カテゴリ
イベントゲートウェイの開発

シンタックス

```
String getGatewayType()
```

関連項目

setGatewayType、CFML CFEvent 構造体、『ColdFusion アプリケーションの開発』の CFEvent class

戻り値

CFEvent オブジェクトのゲートウェイタイプ、または null。

使用方法

ほとんどのゲートウェイでは、このメソッドを使用する必要はありません。

getOriginatorID

説明

着信メッセージの発信元を識別します。発信メッセージの宛先にこのフィールドを使用するゲートウェイタイプもあります。

カテゴリ

イベントゲートウェイの開発

シンタックス

```
String getOriginatorID()
```

関連項目

setOriginatorID、CFML CFEvent 構造体、『ColdFusion アプリケーションの開発』の CFEvent class

戻り値

メッセージ発信元のプロトコル固有の識別子、または null。

例

SocketGateway ゲートウェイの例の outgoingMessage メソッドで getOriginatorID メソッドを使用して、発信メッセージの宛先を識別します。こうすると、応答を発信元に返すリスナー CFC は、戻り変数で宛先を明示的に設定する必要はありません。フィールドが空の場合、CFML SendGatewayMessage 関数によって送信されたメッセージには宛先のデータがあるため、ゲートウェイは CFevent データフィールドから宛先を取得しようとします。

```
public String outgoingMessage(coldfusion.eventgateway.CFEvent cfmsg)
{
    String retcode="ok";
    // Get the table of data returned from the event handler
    Map data = cfmsg.getData();
    String message = (String) data.get("MESSAGE");
    // find the right socket to write to from the socketRegistry hashtable
    if (cfmsg.getOriginatorID() != null)
        ((SocketServerThread)socketRegistry.get(cfmsg.getOriginatorID())).
            writeOutput(message);
    else if (data.get("OriginatorID") != null)
        ((SocketServerThread)socketRegistry.get(data.get("OriginatorID"))).
            writeOutput(message);
    else
    {
        System.out.println("cannot send outgoing message. OriginatorID is not
                available.");
        retcode="failed";
    }
    return retcode;
}
```

setCFCMethod

説明
着信メッセージを処理する CFC メソッドの名前を設定します。

カテゴリ
イベントゲートウェイの開発

シンタックス
```
void setCFCMethod(String method)
```

関連項目
getCFCMethod、setCFCPath、setCFCTimeout、1573 ページの「CFML CFEvent 構造体」、『ColdFusion アプリケーションの開発』の CFEvent class

パラメータ

パラメータ	説明
method	ColdFusion がこのイベントを処理するために呼び出すリスナー CFC のメソッド。このメソッドをゲートウェイで使用しない場合、ColdFusion は onIncomingMessage メソッドを呼び出します。

使用方法
単一の CFC リスナーメソッドを使用するゲートウェイでは、CFC リスナーメソッドの名前が onIncomingMessage である場合、このメソッドを使用する必要はありません。一貫性を保つため、単一のリスナーを持つすべてのイベントゲートウェイではこのデフォルトの設定を変更しないことをお勧めします。

異なるメッセージタイプに対して異なるリスナーメソッドを使用する、ColdFusion XMPP ゲートウェイなどのゲートウェイでは、このメソッドを使用して目的のメソッドを識別します。

例

次のコードの例は、ColdFusion XMPP ゲートウェイ着信メッセージハンドラからのものです。この例では、CFEvent オブジェクトを作成し、メッセージタイプに基づいてテストを処理するメソッドを設定します。

```
CFEvent cfmsg = new CFEvent(gatewayID);
cfmsg.setOriginatorID(sender);
cfmsg.setGatewayType(gatewayType);
if(messageType == IMessage.IM)
    {
    // default for normal messages
    cfmsg.setCfcMethod(onIncomingMessageFunction);
    }
    //if the message is an authorization request
else if(messageType == IMessage.AUTH_REQUEST)
    {
    cfmsg.setCfcMethod(onAddBuddyRequestFunction);
    message = "Requesting authorization to add '" + recipient + "' to '"
    + sender + "' buddy list and view '" + recipient + "' presence.";
    } // Code snipped here for brevity.
```

setCFCPath

説明

このイベントを処理するリスナー CFC を指定します。

カテゴリ

イベントゲートウェイの開発

シンタックス

```
void setCFCPath(String path)
```

関連項目

getCFCPath、setCFCMethod、setCFCTimeout、『ColdFusion アプリケーションの開発』の CFEvent class

パラメータ

パラメータ	説明
path	イベントを処理するアプリケーションリスナー CFC の絶対パス。このメソッドをゲートウェイで呼び出さない場合、ColdFusion は ColdFusion Administrator の [イベントゲートウェイ] ページでイベントゲートウェイインスタンス用に設定された最初のパスを使用します。

使用方法

デフォルトでは、ColdFusion は ColdFusion Administrator の [イベントゲートウェイ] ページでイベントゲートウェイインスタンス用に設定された最初のパスにある CFC にメッセージを配信します。

使用するアプリケーションが複数のリスナー CFC をサポートする場合、このメソッドを使用して、各リスナー CFC を設定し、gatewayService.addEvent メソッドを呼び出してイベントを CFC に送信します。

例

次のコードの例は、ソケットから入力値を取り入れて CFC リスナーメソッドに送信する、Socket ゲートウェイ processInput メソッドに基づいています。リスナー変数にはリスナー CFC の配列が含まれています。リスナー変数は、ゲートウェイの開始時に ColdFusion が呼び出すゲートウェイの setCFCListeners メソッドによって設定されます。

```
for (int i = 0; i < listeners.length; i++)
{
    String path = listeners[i];
    CFEvent event = new CFEvent(gatewayID);
    Hashtable mydata = new Hashtable();
    mydata.put("MESSAGE", theInput);
    event.setData(mydata);
    event.setGatewayType("SocketGateway");
    event.setOriginatorID(theKey);
    event.setCFCMethod(cfcEntryPoint);
    event.setCFCTimeout(10);
    if (path != null)
        event.setCFCPath(path);boolean sent = gatewayService.addEvent(event);
}
```

setCFCTimeout

説明
タイムアウトの値を秒単位で設定します。ここで設定した時間内に、リスナー CFC がイベントリクエストを処理して結果を返さなかった場合、ColdFusion ゲートウェイサービスはリクエストを終了します。

カテゴリ
イベントゲートウェイの開発

シンタックス
```
void setCFTimeout(String timeout)
```

関連項目
getCFCTimeout、setCFCMethod、setCFCPath、『ColdFusion アプリケーションの開発』の CFEvent class

パラメータ

パラメータ	説明
timeout	CFC のタイムアウト時間 (秒単位) です。

使用方法
ColdFusion がイベントを処理するためにリスナー CFC メソッドを呼び出したが、指定されたタイムアウト時間内に CFC が結果を返さなかった場合、ColdFusion はこのリクエストを終了し、"application.log" ファイルにエラーを記録します。

このメソッドを使用しない場合、ColdFusion は ColdFusion Administrator の [サーバーの設定] ページで設定した [リクエストタイムアウト] の値を使用します。

ColdFusion HTML の標準のリクエストよりも長いか短いタイムアウト時間が必要なメッセージの場合、このメソッドを使用します。

例
次のコードの例は、ソケットから入力値を取り入れて CFC リスナーメソッドに送信する、Socket ゲートウェイ processInput メソッドに基づいています。この例では、CFC のタイムアウト時間を 10 秒に設定します。

```
for (int i = 0; i < listeners.length; i++)
{
    String path = listeners[i];
    CFEvent event = new CFEvent(gatewayID);
    Hashtable mydata = new Hashtable();
    mydata.put("MESSAGE", theInput);
    event.setData(mydata);
    event.setGatewayType("SocketGateway");
    event.setOriginatorID(theKey);
    event.setCfcMethod(cfcEntryPoint);
    event.setCfcTimeOut(10);
    if (path != null)
        event.setCfcPath(path);
    boolean sent = gatewayService.addEvent(event);
}
```

setData

説明
メッセージの内容など、ゲートウェイ固有のデータを Java Map として CFEvent に追加します。

カテゴリ
イベントゲートウェイの開発

シンタックス
```
void setData(Map data)
```

関連項目
getData、CFML CFEvent 構造体、『ColdFusion アプリケーションの開発』の CFEvent class

パラメータ

パラメータ	説明
data	着信メッセージおよびその他のゲートウェイ固有のイベントデータです。

使用方法
フィールドの数とその内容は、イベントゲートウェイのタイプに応じて異なります。Map キーは文字列である必要があります。

ColdFusion は大文字と小文字を区別しないため、setData メソッドで渡される Map を大文字と小文字を区別しない Map に変換します。したがって、大文字小文字のみが異なる名前を持つエントリをデータに作成しないでください。

例
次のコードは、着信メッセージを処理する JMS ゲートウェイの例からのルーチンを示します。JMS メッセージの ID と内容をデータ HashMap に挿入し、それを setData メソッドで使用します。

```
public void handleMessage(String msg, String topicName, String msgID) {
    coldfusion.eventgateway.Logger log = getGatewayServices().getLogger();
    Map data = new HashMap();
    CFEvent cfMsg = new CFEvent(getGatewayID());
    data.put("msg", msg);
    data.put("id", msgID);
    cfMsg.setData(data);
    cfMsg.setOriginatorID(topicName);
    cfMsg.setGatewayType("JMS");
    if (sendMessage(cfMsg)) {
            log.info("Added message '" + msgID + "' to queue.");
    } else {
            log.error("Failed to add message '" + msgID + "' to queue.");
    }
}
```

setGatewayType

説明
イベントゲートウェイのタイプを識別します。

カテゴリ
イベントゲートウェイの開発

シンタックス
```
void setGatewayType(String gatewayType)
```

関連項目
getGatewayType、CFML CFEvent 構造体、『ColdFusion アプリケーションの開発』の CFEvent class

パラメータ

パラメータ	説明
gatewayType	ゲートウェイタイプの識別子です。

使用方法
一貫性を保つため、ColdFusion Administrator でイベントゲートウェイタイプを追加するときは、このメソッドと [タイプ名] フィールドで同じ名前を使用します。複数のインスタントメッセージングプロバイダを処理するインスタントメッセージングアプリケーションの CFC など、複数のゲートウェイタイプを処理するゲートウェイアプリケーションの CFC では、このフィールドを使用して、プロトコルのタイプとゲートウェイタイプ固有のアクションを指定します。

例
次のコードは、着信メッセージを処理する JMS ゲートウェイの例からのルーチンを示します。ゲートウェイタイプを JMS に設定します。

```
public void handleMessage(String msg, String topicName, String msgID) {
    coldfusion.eventgateway.Logger log = getGatewayServices().getLogger();
    Map data = new HashMap();
    CFEvent cfMsg = new CFEvent(getGatewayID());
    data.put("msg", msg);
    data.put("id", msgID);
    cfMsg.setData(data);
    cfMsg.setOriginatorID(topicName);
    cfMsg.setGatewayType("JMS");
    if (sendMessage(cfMsg)) {
            log.info("Added message '" + msgID + "' to queue.");
    } else {
            log.error("Failed to add message '" + msgID + "' to queue.");
    }
}
```

setOriginatorID

説明
着信メッセージの発信元を識別します。

カテゴリ
イベントゲートウェイの開発

シンタックス
```
void setOriginatorID(String originatorID)
```

関連項目
getOriginatorID、CFML CFEvent 構造体、『ColdFusion アプリケーションの開発』の CFEvent class

パラメータ

パラメータ	説明
originatorID	メッセージ発信元のゲートウェイ固有またはプロトコル固有の ID です。

例
次のコードは、着信メッセージを処理する JMS ゲートウェイの例からのルーチンを示します。発信元 ID をゲートウェイが処理する JMS トピックの名前に設定します。

```
public void handleMessage(String msg, String topicName, String msgID) {
    coldfusion.eventgateway.Logger log = getGatewayServices().getLogger();
    Map data = new HashMap();
    CFEvent cfMsg = new CFEvent(getGatewayID());
    data.put("msg", msg);
    data.put("id", msgID);
    cfMsg.setData(data);
    cfMsg.setOriginatorID(topicName);
    cfMsg.setGatewayType("JMS");
    if (sendMessage(cfMsg)) {
            log.info("Added message '" + msgID + "' to queue.");
    } else {
            log.error("Failed to add message '" + msgID + "' to queue.");
    }
}
```

Logger クラス

```
coldfusion.eventgateway.Logger
```

注意：このクラスは、coldfusion.eventgateway パッケージではなく、coldfusion.log パッケージにあります。パッケージには、その他すべてのイベントゲートウェイ関連のインターフェイスとクラスが含まれています。

Logger クラスは、ColdFusion のログディレクトリ内のファイルにメッセージを記録しますこのディレクトリは、ColdFusion Administrator の [ロギングの設定] ページで設定します。coldfusion.eventgateway.GatewayServices.getLogger() メソッドは、Logger クラスのインスタンスを返します。Logger クラスには、次のメソッドがあります。

シグネチャ	説明
debug	デバッグメッセージをログファイルに書き込みます。
error	エラーメッセージをログファイルに書き込みます。
fatal	重大エラーをログファイルに書き込みます。
info	情報メッセージをログファイルに書き込みます。
warn	警告メッセージをログファイルに書き込みます。

debug

説明
厳格度が debug であるログエントリを ColdFusion Logger に書き込みます。このエントリには、厳格度、スレッド ID、日付、時刻、およびテキストメッセージが含まれています。

カテゴリ
イベントゲートウェイの開発

シンタックス
```
debug(String message)
debug(Throwable th)
debug(String message, Throwable th)
```

関連項目
error、fatal、info、warn、getLogger、『ColdFusion アプリケーションの開発』の Logging events and using log files

パラメータ

パラメータ	説明
message	ログエントリに含めるメッセージです。
th	Throwable オブジェクト (通常は例外) です。ColdFusion は、ColdFusion のログディレクトリにある "exception.log" ファイルに例外情報を記録します。

使用方法
このメソッドを使用して、デバッグメッセージを ColdFusion のロギングサブシステムに送信します。

デフォルトでは、ColdFusion はデバッグメッセージをログファイルに書き込み**ません**。デバッグメッセージがログファイルに含まれるようにするには、<ColdFusion のルートディレクトリ >¥lib¥neo-logging.xml (サーバー設定の場合) または <ColdFusion のルートディレクトリ >¥WEB-INF¥cfusion¥lib¥neo-logging.xml (J2EE 設定の場合) の priority エントリを変更します。変更するのは次のエントリです。

```
<var name='priority'>
    <string>information</string>
</var>
```

このエントリを次のように変更します。

```
<var name='priority'>
    <string>debug</string>
</var>
```

priority エントリを debug にすると、ColdFusion は厳格度が "debug" であるメッセージを、Logger インスタンスを返す getLogger メソッドで指定されたログファイル (またはデフォルトのログファイル) に書き込みます。

例

ColdFusion のインスタントメッセージングゲートウェイは次の行を使用して、priority エントリが debug の場合に限り、着信する管理メッセージまたはエラーに関する情報を記録します。

```
// code to process incoming administrative messages or errors
logger.debug(gatewayType + "Gateway (" + gatewayID + ") admin message: " +
            msg.getMessage());
```

error

説明

厳格度が error であるログエントリを ColdFusion Logger に書き込みます。このエントリには、厳格度、スレッド ID、日付、時刻、およびテキストメッセージが含まれています。

カテゴリ

イベントゲートウェイの開発

シンタックス

```
error(String message)
error(Throwable th)
error(String message, Throwable th)
```

関連項目

debug、fatal、info、warn、getLogger、『ColdFusion アプリケーションの開発』の Logging events and using log files

パラメータ

パラメータ	説明
message	ログエントリに含めるメッセージです。
th	Throwable オブジェクト (通常は例外) です。ColdFusion は、ColdFusion のログディレクトリにある "exception.log" ファイルに例外情報を記録します。

使用方法
このメソッドを使用して、エラーメッセージを ColdFusion のロギングサブシステムに送信します。ColdFusion は、厳格度が "error" であるメッセージを、Logger インスタンスを返す getLogger メソッドで指定されたログファイル (またはデフォルトのログファイル) に書き込みます。

例
ColdFusion の SocketGateway クラスの例では、outgoingMessage メソッドに次のコードが含まれています。このコードでは、メッセージの発信元 ID が開いているソケットに対応しない場合、エラーメッセージを書き込みます。

```
SocketServerThread st =
        ((SocketServerThread)socketRegistry.get(cfmsg.getOriginatorID()));
    if(st != null)
        st.writeOutput(message);
    else {
        log.error("Cannot send outgoing message. OriginatorID '" +
            cfmsg.getOriginatorID() + "' is not a valid socket id.");
        retcode="failed";
    }
```

fatal

説明
厳格度が fatal であるログエントリを ColdFusion Logger に書き込みます。このエントリには、厳格度、スレッド ID、日付、時刻、およびテキストメッセージが含まれています。

カテゴリ
イベントゲートウェイの開発

シンタックス
```
fatal(String message)
fatal(Throwable th)
fatal(String message, Throwable th)
```

関連項目
debug、error、info、warn、getLogger、『ColdFusion アプリケーションの開発』の Logging events and using log files

パラメータ

パラメータ	説明
message	ログエントリに含めるメッセージです。
th	Throwable オブジェクト (通常は例外) です。ColdFusion は、ColdFusion のログディレクトリにある "exception.log" ファイルに例外情報を記録します。

使用方法
このメソッドを使用して、重大エラーメッセージを ColdFusion のロギングサブシステムに送信します。ColdFusion は、厳格度が "fatal" であるメッセージを、Logger インスタンスを返す getLogger メソッドで指定されたログファイル (またはデフォルトのログファイル) に書き込みます。

info

説明
厳格度が information であるログエントリを ColdFusion Logger に書き込みます。このエントリには、厳格度、スレッド ID、日付、時刻、およびテキストメッセージが含まれています。

カテゴリ
イベントゲートウェイの開発

シンタックス
```
info(String message)
info(Throwable th)
info(String message, Throwable th)
```

関連項目
debug、error、fatal、warn、getLogger、『ColdFusion アプリケーションの開発』の Logging events and using log files

パラメータ

パラメータ	説明
message	ログエントリに含めるメッセージです。
th	Throwable オブジェクト (通常は例外) です。ColdFusion は、ColdFusion のログディレクトリにある "exception.log" ファイルに例外情報を記録します。通常はこのメソッドでは使用しません。

使用方法
このメソッドを使用して、情報メッセージを ColdFusion のロギングサブシステムに送信します。ColdFusion は、厳格度が "information" であるメッセージを、Logger インスタンスを返す getLogger メソッドで指定されたログファイル (またはデフォルトのログファイル) に書き込みます。

通常、厳格度が "information" のメッセージはすべてログに書き込まれます。そのため、デバッグメッセージや頻繁に発生するイベントに対しては、この厳格度を使用しないでください。

例
ColdFusion の DirectoryWatcherGateway クラスの例では、ゲートウェイの設定ファイルをロードする loadconfig メソッドの先頭に次の行があります。この行では、ゲートウェイ ID と設定ファイルを含むメッセージを書き込みます。

```
logger.info("DirectoryWatcher (" + gatewayID + ") Initializing
        DirectoryWatcher gateway with configuration file " + config);
```

warn

説明
厳格度が warning であるログエントリを ColdFusion Logger に書き込みます。このエントリには、厳格度、スレッド ID、日付、時刻、およびテキストメッセージが含まれています。

カテゴリ
イベントゲートウェイの開発

シンタックス

```
warn(String message)
warn(Throwable th)
warn(String message, Throwable th)
```

関連項目

debug、error、fatal、info、getLogger、『ColdFusion アプリケーションの開発』の Logging events and using log files

パラメータ

パラメータ	説明
message	ログエントリに含めるメッセージです。
th	Throwable オブジェクト (通常は例外) です。ColdFusion は、ColdFusion のログディレクトリにある "exception.log" ファイルに例外情報を記録します。

使用方法

このメソッドを使用して、警告メッセージを ColdFusion のロギングサブシステムに送信します。ColdFusion は、厳格度が "warning" であるメッセージを、Logger インスタンスを返す getLogger メソッドで指定されたログファイル (またはデフォルトのログファイル) に書き込みます。

例

ColdFusion の SocketWatcherGateway クラスの例では、設定ファイルをロードするコンストラクタに次のコードが含まれています。このコードでは、設定ファイルをロードできない場合、その例外情報を文字列に変換して、ゲートウェイ ID を含む警告および例外情報を記録します。また、このコードは例外を warn メソッドに渡します。

```
propsFilePath=configpath;
try {
    FileInputStream propsFile = new FileInputStream(propsFilePath);
    properties.load(propsFile);
    propsFile.close();
    this.loadProperties();
    }
catch (IOException e) {
    // do nothing. use default value for port.
    log.warn("SocketGateway(" + gatewayID + ") Unable to read configuration file
        " + propsFilePath + ": " + e.ToString() + ".Using default port.", e);
    }
```

CFML CFEvent 構造体

CFML リスナー CFC メソッドでは、ゲートウェイ開発者が使用する CFEvent クラスに対応する CFEvent 構造体の形式でメッセージを受信します。この構造体には次のフィールドがあります。すべてのゲートウェイで使用されるわけではないフィールドもあります。Data フィールド以外のすべてのフィールドでは、テキスト値または数値を使用します。Data フィールドでは構造体を使用します。

フィールド	説明
GatewayID	イベントを送信したイベントゲートウェイまたは発信メッセージを処理するイベントゲートウェイです。この値は、ColdFusion Administrator の [ゲートウェイ] ページで設定したイベントゲートウェイインスタンスの ID です。アプリケーションで SendGatewayMessage 関数を呼び出してイベントゲートウェイに応答する場合は、この ID を関数の最初のパラメータとして使用します。
Data	メッセージなどのイベントデータを含む構造体。Data 構造体の内容は、イベントゲートウェイタイプによって異なります。このフィールドは、SendGatewayMessage 関数の 2 番目のパラメータに対応します。
OriginatorID	メッセージの発信元。この値は、プロトコルやイベントゲートウェイタイプによって異なります。一部のイベントゲートウェイが、応答の宛先を識別するために応答メッセージでこの値を必要とすることがあります。メッセージの送信者を識別します。
GatewayType	SMS などの、イベントゲートウェイのタイプ。複数のイベントゲートウェイタイプからのメッセージを処理するアプリケーションでは、このフィールドを使用できます。この値は、イベント Gateway クラスによって指定されるゲートウェイタイプ名です。ColdFusion Administrator のゲートウェイタイプ名と一致しているとは限りません。
CFCPath	リスナー CFC の場所です。リスナー CFC はこのフィールドを使用する必要はありません。
CFCMethod	ColdFusion がイベントを処理するために呼び出すリスナーメソッドです。リスナー CFC はこのフィールドを使用する必要はありません。
CFCTimeout	リスナー CFC がイベントリクエストを処理するためのタイムアウト時間 (秒単位)。リスナー CFC はこのフィールドを使用する必要はありません。

IM ゲートウェイメソッドとコマンド

XMPP ゲートウェイおよび IBM Sametime ゲートウェイは、CFC メソッドを実装してメッセージを受信し、gatewayHelper オブジェクトメソッドを使用してゲートウェイを管理し、発信メッセージコマンドを使用してメッセージを送信します。次のセクションでは、これらのメソッドとコマンドについて説明します。

関連項目

1574 ページの「IM ゲートウェイ CFC 着信メッセージメソッド」

1582 ページの「IM ゲートウェイメッセージ送信コマンド」

1583 ページの「IM ゲートウェイ GatewayHelper クラスのメソッド」

IM ゲートウェイ CFC 着信メッセージメソッド

XMPP または Lotus Sametime インスタントメッセージングゲートウェイからの着信メッセージを処理するために、次の CFC メソッドを書き込みます。

注意：メソッド名にはデフォルトのゲートウェイ設定が適用されます。ColdFusion では、このメソッド名を変更して、ゲートウェイ設定ファイル内のイベントタイプを無効にすることができます。

メソッド	メッセージタイプ
onAddBuddyRequest	他の IM ユーザーからの、ゲートウェイ ID をそのユーザーのバディとして追加するように要求するリクエストです。
onAddBuddyResponse	「あなたを自分のバディリストに追加したい」というリクエストを他のユーザーに送信した結果、そのユーザーから返された応答。「自分をあなたのバディリストから削除したい」というバディからのリクエストにも使用されます。

メソッド	メッセージタイプ
onBuddyStatus	オンラインステータス情報メッセージです。
onIMServerMessage	IM サーバーからのエラーおよび管理に関するメッセージです。
onIncomingMessage	インスタントメッセージです。

onAddBuddyRequest

説明
ユーザーに対してそのバディの 1 人としてゲートウェイユーザー名を追加するように要求する着信リクエストを処理します。

シンタックス
```
onAddBuddyRequest(CFEvent)
```

関連項目
onIncomingMessage、onAddBuddyResponse、onBuddyStatus、onIMServerMessage

パラメータ
このメソッドにはパラメータが 1 つ必要です。つまり次のフィールドを持つ CFEvent 構造体が必要です。

フィールド	説明
gatewayType	ゲートウェイタイプです。XMPP または SAMETIME のいずれかです。
gatewayID	ColdFusion Administrator で設定されたゲートウェイインスタンスの ID です。
originatorID	メッセージ発信元の IM ID です。
cfcMethod	CFC メソッドです。デフォルトでは onAddBuddyRequest です。
data.MESSAGE	リクエストとともに送信されたメッセージです。
data.SENDER	送信者の ID です。originatorID フィールドの値と同じです。
data.RECIPIENT	ゲートウェイの設定ファイルで指定された、受信者の ID です。
data.TIMESTAMP	メッセージが送信された日時です。

戻り値
この関数は、応答メッセージを送信するための値を必要に応じて返すことができます。返される構造体には、次のフィールドが含まれている必要があります。

フィールド	説明
command	次のいずれかです。 • accept ユーザーをバディとして追加するリクエストを受け入れます。ColdFusion は、ステータス情報を取得できるユーザーのアクセス許可リストにこのユーザーを追加します。 • decline ユーザーをバディとして追加するリクエストを拒否します。ColdFusion は、ステータス情報を取得できるユーザーのアクセス拒否リストにこのユーザーを追加します。 • noact 何も実行しません。ColdFusion はリクエスタに応答しません。
buddyID	メッセージの送信先 ID です。通常は CFEvent.data.SENDER フィールドの値です。noact コマンドと一緒には使用しません。
reason	アクションの理由を示すテキストメッセージです。noact コマンドと一緒には使用しません。

例

次の例では、リクエストされたバディの名前をデータソースで検索します。固有のエントリが見つかった場合は、そのバディを追加し、Application スコープの buddyStatus 構造体にあるバディのステータス情報を更新します。名前が見つからなかった場合は、そのバディを拒否します。そのバディ名について複数のエントリがデータベース内にある場合は、応答しないようにゲートウェイに指示します。この例では、すべてのアクションを記録します。

```
<cffunction name="onAddBuddyRequest">
    <cfargument name="CFEvent" type="struct" required="YES">
    <cfquery name="buddysearch" datasource="cfdocexamples">
        SELECT IM_ID
        FROM Employees
        WHERE IM_ID = '#CFEvent.Data.SENDER#'
    </cfquery>
    <cflock scope="APPLICATION" timeout="10" type="EXCLUSIVE">
        <cfscript>
            // If the name is in the DB once, accept; if it is missing, decline.
            // If it is in the DB multiple times, take no action.
            if (buddysearch.RecordCount IS 0) {
                action="decline";
                reason="Invalid ID";
            }
            else if (buddysearch.RecordCount IS 1) {
                action="accept";
                reason="Valid ID";
                //Add the buddy to the buddy status structure only if accepted.
                if (NOT StructKeyExists(Application,
                        "buddyStatus")) {
                    Application.buddyStatus=StructNew();
                }
                if (NOT StructKeyExists(Application.buddyStatus,
                        CFEvent.Data.SENDER)) {
                    Application.buddyStatus[#CFEvent.Data.SENDER#]=StructNew();
                }
                Application.buddyStatus[#CFEvent.Data.SENDER#].status=
                    "Accepted Buddy Request";
                Application.buddyStatus[#CFEvent.Data.SENDER#].timeStamp=
```

```
                CFEvent.Data.TIMESTAMP;
            Application.buddyStatus[#CFEvent.Data.SENDER#].message=
                CFEvent.Data.MESSAGE;
        }
        else {
            action="noact";
        }
    </cfscript>
</cflock>
<!--- Log the request and decision information. --->
<cflog file="#CFEvent.GatewayID#Status"
        text="onAddBuddyRequest; SENDER: #CFEvent.Data.SENDER# MESSAGE:
#CFEvent.Data.MESSAGE# TIMESTAMP: #CFEvent.Data.TIMESTAMP# ACTION: #action#">
<!--- Return the action decision. --->
<cfset retValue = structNew()>
<cfset retValue.command = action>
<cfset retValue.BuddyID = CFEvent.DATA.SENDER>
<cfset retValue.Reason = reason>
<cfreturn retValue>
</cffunction>
```

onAddBuddyResponse

説明

バディリストへの追加を要求するゲートウェイからのリクエストに対する他のユーザーからの着信応答を処理します。バディリストから削除するように要求するバディからのリクエストも受信します。

シンタックス

```
onAddBuddyResponse(CFEvent)
```

関連項目

onIncomingMessage、onAddBuddyRequest、onBuddyStatus、onIMServerMessage

パラメータ

このメソッドにはパラメータが 1 つ必要です。つまり次のフィールドを持つ CFEvent 構造体が必要です。

フィールド	説明
gatewayType	ゲートウェイタイプです。XMPP または SAMETIME のいずれかです。
gatewayID	ColdFusion Administrator で設定されたゲートウェイインスタンスの ID です。
originatorID	メッセージ発信元の IM ID です。
cfcMethod	CFC メソッドです。デフォルトでは onAddBuddyResponse です。
data.MESSAGE	次のいずれかです。 • accept リクエストは受け入れられました。 • decline リクエストが拒否されたか、またはバディがリストから削除されることを求めています。
data.SENDER	送信者の ID です。originatorID と同じです。
data.RECIPIENT	ゲートウェイの設定ファイルで指定された、受信者の ID です。
data.TIMESTAMP	メッセージが送信された日時です。

戻り値
この関数は値を返しません。

例
次の例では、メッセージの送信者が追加バディリクエストを受け入れた場合、バディのステータスを Application スコープの buddyStatus 構造体に追加します。この関数はすべての応答を記録します。

```
<cffunction name="onAddBuddyResponse">
    <cfargument name="CFEvent" type="struct" required="YES">
    <cflock scope="APPLICATION" timeout="10" type="EXCLUSIVE">
        <cfscript>
            //Do the following only if the buddy accepted the request.
            if (NOT StructKeyExists(Application, "buddyStatus")) {
                Application.buddyStatus=StructNew();
                }
            if (#CFEVENT.Data.MESSAGE# IS "accept") {
                //Create a new entry in the buddyStatus record for the buddy.
                if (NOT StructKeyExists(Application.buddyStatus,
                        CFEvent.Data.SENDER)) {
                    Application.buddyStatus[#CFEvent.Data.SENDER#]=StructNew();
                }
                //Set the buddy status information to indicate buddy was added.
                Application.buddyStatus[#CFEvent.Data.SENDER#].status=
                    "Buddy accepted us";
                Application.buddyStatus[#CFEvent.Data.SENDER#].timeStamp=
                    CFEvent.Data.TIMESTAMP;
                Application.buddyStatus[#CFEvent.Data.SENDER#].message=
                    CFEvent.Data.MESSAGE;
            }
        </cfscript>
    </cflock>
    <!--- Log the information for all responses. --->
    <cflog file="#CFEvent.GatewayID#Status"
        text="onAddBuddyResponse; BUDDY: #CFEvent.Data.SENDER# RESPONSE:
#CFEvent.Data.MESSAGE# TIMESTAMP: #CFEvent.Data.TIMESTAMP#">
</cffunction>
```

onBuddyStatus

説明
ゲートウェイのバディリスト上のユーザーに関するオンラインステータス (プレゼンス) の変更を示す着信メッセージを処理します。

シンタックス
```
onBuddyStatus(CFEvent)
```

関連項目
onIncomingMessage、onAddBuddyRequest、onAddBuddyResponse、onIMServerMessage

パラメータ
このメソッドにはパラメータが 1 つ必要です。つまり次のフィールドを持つ CFEvent 構造体が必要です。

フィールド	説明
gatewayType	ゲートウェイタイプです。XMPP または SAMETIME のいずれかです。
gatewayID	ColdFusion Administrator で設定されたゲートウェイインスタンスの ID です。
originatorID	メッセージ発信元の IM ID (バディ名) です。
cfcMethod	CFC メソッドです。デフォルトでは onIMServerMessage です。
data.BUDDYNAME	送信者のバディ名または ID です。originatorID と同じです。
data.BUDDYNICKNAME	バディの表示名またはニックネームです。
data.BUDDYSTATUS	バディのステータスです。次のいずれかです。 • ONLINE • OFFLINE • AWAY • DO NOT DISTURB • NOT AVAILABLE • FREE TO CHAT • IDLE **XMPP の場合のみ** • NOT AVAILABLE • FREE TO CHAT • IDLE **Sametime の場合のみ** • IDLE IMGatewayHelpergetCustomAwayMessage メソッドを使用して、ステータスの変更時にバディが送信するカスタムメッセージを取得します。
data.BUDDYGROUP	バディが属するグループです。
data.RECIPIENT	ゲートウェイの設定ファイルで指定された、受信者の ID です。
data.TIMESTAMP	メッセージが送信された日時です。

注意：gatewayHelper オブジェクトの addBuddy メソッドを使用してバディを追加するときに、バディのニックネームとグループを設定します。

戻り値
この関数は値を返しません。

例
次の例では、Application スコープ構造体を、バディのステータスを反映した最新のものとして維持します。バディのカスタム不在メッセージがある場合は、gatewayhelper オブジェクトの getBuddyStatus メソッドを使用して、そのカスタム不在メッセージも取得します。

```
<cffunction name="onBuddyStatus">
    <cfargument name="CFEvent" type="struct" required="YES">
    <!--- Get the gatewayhelper object and to get the info for this buddy. --->
    <!--- This is used to get the buddy's custom away message. --->
    <cfset helper = getGatewayHelper("MYIM")>
    <cfset mybuddyinfo=helper.getBuddyInfo(CFEvent.Data.BUDDYNAME)>

    <cflog file="#CFEvent.GatewayID#Status" type="Information"
        text="in OnbuddyStatus, sender is #CFEvent.OriginatorID#">
    <cflock scope="APPLICATION" timeout="10" type="EXCLUSIVE">
        <cfscript>
        // Create the status structures if they don't exist.
            if (NOT StructKeyExists(Application, "buddyStatus")) {
                Application.buddyStatus=StructNew();
            }
            if (NOT StructKeyExists(Application.buddyStatus,
                     CFEvent.Data.BUDDYNAME)) {
                Application.buddyStatus[#CFEvent.Data.BUDDYNAME#]=StructNew();
            }
            // Save the buddy status, timestamp, and custom away message
            Application.buddyStatus[#CFEvent.Data.BUDDYNAME#].status=
                CFEvent.Data.BUDDYSTATUS;
            Application.buddyStatus[#CFEvent.Data.BUDDYNAME#].timeStamp=
                CFEvent.Data.TIMESTAMP;
            // The following assumes that the buddy is in only one group.
            Application.buddyStatus[#CFEvent.Data.BUDDYNAME#].customAway=
                mybuddyinfo[1].BUDDYCUSTOMAWAYMESSAGE;
        </cfscript>
    </cflock>
    <!--- log the info, for debugging purposes only --->
    <cfset temp=Application.buddyStatus[#CFEvent.Data.BUDDYNAME#].status>
    <cflog file="#CFEvent.GatewayID#Status" type="Information" text=
        "Application.buddyStatus[#CFEvent.Data.BUDDYNAME#].status is #temp#">
    <cfset temp=Application.buddyStatus[#CFEvent.Data.BUDDYNAME#].timeStamp>
        <cflog file="#CFEvent.GatewayID#Status" type="Information" text=
            "Application.buddyStatus[#CFEvent.Data.BUDDYNAME#].timestamp is #temp#">
        <cflog file="#CFEvent.GatewayID#Status" type="Information" text=
            "Buddy Custom Away Message is mybuddyinfo[1].BUDDYCUSTOMAWAYMESSAGE#">
</cffunchtion>
```

onIMServerMessage

説明

IM サーバーからのエラーおよびステータスに関するメッセージを処理します。

シンタックス

```
onIMServerMessage(CFEvent)
```

関連項目

onIncomingMessage、onAddBuddyRequest、onAddBuddyResponse、onBuddyStatus

パラメータ

このメソッドにはパラメータが 1 つ必要です。つまり次のフィールドを持つ CFEvent 構造体が必要です。

フィールド	説明
gatewayType	ゲートウェイタイプです。XMPP または SAMETIME のいずれかです。
gatewayID	ColdFusion Administrator で設定されたゲートウェイインスタンスの ID です。
originatorID	メッセージ発信元の IM ID (バディ名) です。
cfcMethod	CFC メソッドです。デフォルトでは onIMServerMessage です。
data.MESSAGE	サーバーによって送信されたメッセージです。
data.SENDER	送信者の ID です。originatorID と同じです。
data.RECIPIENT	ゲートウェイの設定ファイルで指定された、受信者の ID です。
data.TIMESTAMP	メッセージが送信された日時です。

例

次の例では、IM サーバーがエラーまたはステータスに関するメッセージを送信するときに、送信者、メッセージ、およびタイムスタンプを記録します。

```
<cffunction name="onIMServerMessage">
    <!--- This function just logs the message. --->
    <cfargument name="CFEvent" type="struct" required="YES">
    <cflog file="#CFEvent.GatewayID#Status"
        text="onIMServerMEssage; SENDER: #CFEvent.OriginatorID#MESSAGE:
#CFEvent.Data.MESSAGE# TIMESTAMP: #CFEvent.Data.TIMESTAMP#">
</cffunction>
```

onIncomingMessage

説明

他のユーザーからの着信インスタントメッセージを処理します。必要に応じてメッセージの送信者に応答を返します。

シンタックス

```
onIncomingMessage(CFEvent)
```

関連項目

onAddBuddyRequest、onAddBuddyResponse、onBuddyStatus、onIMServerMessage、『ColdFusion アプリケーションの開発』の Handling incoming messages

パラメータ

このメソッドにはパラメータが 1 つ必要です。つまり次のフィールドを持つ CFEvent 構造体が必要です。

フィールド	説明
gatewayType	ゲートウェイタイプです。XMPP または SAMETIME のいずれかです。
gatewayID	ColdFusion Administrator で設定されたゲートウェイインスタンスの ID です。
originatorID	メッセージ発信元の IM ID です。
cfcMethod	CFC メソッドです。デフォルトでは onIncomingMessage です。
data.MESSAGE	受信されたメッセージです。

フィールド	説明
data.SENDER	送信者の ID です。originatorID と同じです。
data.RECIPIENT	ゲートウェイの設定ファイルで指定された、受信者の ID です。
data.TIMESTAMP	メッセージが送信された日時です。

戻り値

この関数は、応答メッセージを送信するための値を必要に応じて返すことができます。返される構造体には、次のフィールドが含まれている必要があります。

フィールド	説明
command	通常は省略されます。submit を指定することもできます。
buddyID	メッセージの送信先 ID です。通常は、入力パラメータの Data.SENDER フィールドの値です。
message	メッセージの内容です。

例

次の例は、メッセージを送信者にエコーバックするシンプルな onIncomingMessage メソッドを示しています。

```
<cffunction name="onIncomingMessage">
    <cfargument name="CFEvent" type="struct" required="YES">
    <cfset input_mesg = CFEvent.data.MESSAGE>
    <cfset retValue = structNew()>
    <cfset retValue.command = "submit">
    <cfset retValue.buddyID = CFEvent.originatorID>
    <cfset retValue.message = "Message Received:" & input_mesg>
    <cfreturn retValue>
</cffunction>
```

IM ゲートウェイメッセージ送信コマンド

SendGatewayMessage CFML 関数または CFC リスナーメソッドの戻り値を使用して、発信メッセージを送信します。ColdFusion IM ゲートウェイでは、次の発信メッセージコマンドを使用できます。

コマンド	説明
submit	(デフォルト) 標準メッセージを他の IM ユーザーに送信します。
accept	バディの追加リクエストを承認します。プレゼンス情報の取得を許可する ID のリストにそのバディを追加して、承認メッセージをそのバディ ID に送信します。
decline	バディの追加リクエストを拒否し、拒否メッセージをそのバディ ID に送信します。
noact	ゲートウェイに対して、何も実行しないことを伝えます。ゲートウェイは、何も実行しなかったことを示すメッセージをログに記録します。このメッセージには、ゲートウェイのタイプ、ゲートウェイ ID、およびバディ ID が含まれます。

ゲートウェイリスナー CFC 関数で返すメッセージや、CFML の SendGatewayMessage 関数の 2 番目のパラメータとして使用するメッセージは、次のフィールドを持つメッセージ構造体です。次の表では、それらのフィールドおよびフィールドが使用されるコマンドについて示し、各フィールドの使用方法を説明します。

フィールド	コマンド	説明
buddyID	すべて	宛先となるユーザーの ID です。
command	すべて	コマンドです。省略した場合、デフォルトの submit になります。
message	submit	宛先のユーザーに送信するテキストメッセージです。
reason	accept、decline	アクションに関する理由を示すテキスト、またはバディ追加をリクエストしたユーザーに送信するその他のメッセージです。

一般的に ColdFusion アプリケーションは、onAddBuddyRequest メソッドの戻り値に accept、decline、および noact コマンドを使用し、SendGatewayMessage CFML 関数で、および onIncomingMessage CFC メソッドの戻り値で submit コマンド (または、submit がデフォルトコマンドなのでコマンドを指定しない) を使用します。

IM ゲートウェイ GatewayHelper クラスのメソッド

CFML GetGatewayHelper 関数によって返される GatewayHelper クラスには、次のメソッドがあります。

addBuddy	getDenyList	getStatusAsString	removeDeny
addDeny	getName	getStatusTimeStamp	removePermit
addPermit	getNickName	isOnline	setNickName
getBuddyInfo	getPermitList	numberOfMessagesReceived	setPermitMode
getBuddyList	getPermitMode	numberOfMessagesSent	setStatus
getCustomAwayMessage	getProtocolName	removeBuddy	

addBuddy

説明

ゲートウェイユーザー ID のバディリストにバディを追加し、バディのオンラインプレゼンスステートを含むメッセージを IM サーバーがゲートウェイに送信するように求めます。

シンタックス

```
Boolean = addBuddy(name, nickname, group)
```

関連項目

getBuddyInfo、getBuddyList、removeBuddy、『ColdFusion アプリケーションの開発』の Using the GatewayHelper object

パラメータ

パラメータ	説明
name	ステータスに関する定期的なメッセージを受信する対象の個人についての、固有のインスタントメッセージングユーザー名です。
nickname	ユーザーを参照するためにアプリケーションで使用できるニックネームです。
group	使用するバディリスト内の、ユーザーを追加する先のグループの名前です。指定されたグループが存在しない場合は、そのグループが作成されます。group パラメータが空の文字列の場合、ゲートウェイは General グループを使用します。

戻り値
ID がゲートウェイのバディリストに追加された場合は true、それ以外の場合は false。

使用方法
このメソッドは、ゲートウェイの ID のバディリストにバディを追加し、バディのオンラインステータスに関するプレゼンス情報を自動的に取得するために、サブスクリプションリクエストをリモートバディに送信します。バディからの応答を待たないので、バディがサブスクリプションリクエストを拒否した場合でも true を返し、ゲートウェイはバディをリストに追加します。リスナー CFC onAddBuddyResponse メソッドを使用して、バディの応答を監視します。
CFEvent.data.MESSAGE フィールド値が拒否された場合、リスナーメソッドは gatewayHelper オブジェクトの removeBuddy メソッドを呼び出して、バディリストからそのバディを削除することができます。

例
『ColdFusion アプリケーションの開発』の GatewayHelper example を参照してください。この例では、すべての GatewayHelper クラスメソッドを使用しています。

addDeny

説明
指定されたユーザーをゲートウェイのユーザー ID のアクセス拒否リストに追加するように IM サーバーに指示します。
ゲートウェイのアクセス許可モード値が DENY_SOME である場合、指定されたユーザーはゲートウェイのプレゼンステートのメッセージを受信できません。

シンタックス
```
Boolean = addDeny(name, nickname, group)
```

関連項目
addPermit、getDenyList、getPermitList、getPermitMode、removeDeny、removePermit、setPermitMode、『ColdFusion アプリケーションの開発』の Using the GatewayHelper object

パラメータ

パラメータ	説明
name	ステータスメッセージへのアクセスを拒否する対象となる個人の固有のインスタントメッセージングユーザー名です。
nickname	ユーザーを参照するためにアプリケーションで使用できるニックネームです。空の文字列も指定できます。
group	ユーザーの追加先であるバディリスト内のグループの名前です。指定されたグループが存在しない場合は作成します。group パラメータが空の文字列の場合、ゲートウェイは General グループを使用します。

戻り値
ID がアクセス拒否リストに追加された場合は true、それ以外の場合は false。

注意： XMPP サーバーがアクセス許可管理をサポートしない場合、この関数は常に false を返します。

例
『ColdFusion アプリケーションの開発』の GatewayHelper example を参照してください。この例では、すべての GatewayHelper クラスメソッドを使用しています。

addPermit

説明
指定されたユーザーをゲートウェイのユーザー ID のアクセス許可リストに追加するように IM サーバーに指示します。ゲートウェイのアクセス許可モード値が PERMIT_SOME である場合、指定されたユーザーはゲートウェイのプレゼンスステートのメッセージを受信します。

シンタックス
```
Boolean = addPermit(name, nickname, group)
```

関連項目
addDeny、getDenyList、getPermitList、getPermitMode、removeDeny、removePermit、setPermitMode、『ColdFusion アプリケーションの開発』の Using the GatewayHelper object

パラメータ

パラメータ	説明
name	ステータスメッセージへのアクセスを拒否する対象となる個人の固有のインスタントメッセージングユーザー名です。
nickname	ユーザーを参照するためにアプリケーションで使用できるニックネームです。空の文字列も指定できます。
group	ユーザーの追加先であるバディリスト内のグループの名前です。指定されたグループが存在しない場合は作成します。group パラメータが空の文字列の場合、ゲートウェイは General グループを使用します。

戻り値
ID がアクセス許可リストに追加された場合は true、それ以外の場合は false。

注意：XMPP サーバーがアクセス許可管理をサポートしない場合、この関数は常に false を返します。

例
『ColdFusion アプリケーションの開発』の GatewayHelper example を参照してください。この例では、すべての GatewayHelper クラスメソッドを使用しています。

getBuddyInfo

説明
バディリスト、アクセス拒否リスト、およびアクセス許可リストから、指定したユーザーに関する情報を取得します。

シンタックス
```
array = getBuddyInfo(name)
```

関連項目
addBuddy、getBuddyList、removeBuddy、『ColdFusion アプリケーションの開発』の Using the GatewayHelper object

パラメータ

パラメータ	説明
name	情報を取得する対象となる個人の固有のインスタントメッセージングユーザー名です。

戻り値

検出された各情報レコードに関する 1 つの構造体がある構造体の配列。このメソッドでは、指定した名前を含む各リスト (バディ、許可、拒否) 内で、ユーザーが属するグループごとに 1 つのレコードを検出します。各構造体には次のフィールドがあります。一部の IM プロトコルに対して意味を持たないフィールドもあります。あるフィールドについて情報がない場合、そのフィールドは空白になります。

フィールド	説明
BUDDYNAME	ユーザーの固有の ID です。
BUDDYGROUP	ユーザーが属するグループです。
BUDDYNICKNAME	ユーザーに割り当てたニックネームです。
BUDDYPROTOCOL	インスタントメッセージングプロトコルです。JABBER (XMPP の場合) か SAMETIME、または空の文字列 (サーバーが値を返さなかった場合) です。
BUDDYSTATUS	ユーザーのプレゼンスステートです。次のいずれかです。 • ONLINE • OFFLINE • AWAY • DND (DO NOT DISTURB として表示) • NA (NOT AVAILABLE として表示) • FREE_TO_CHAT (FREE TO CHAT として表示) • IDLE **XMPP の場合のみ** • NA (NOT AVAILABLE として表示) • FREE_TO_CHAT (FREE TO CHAT として表示) • IDLE **Sametime の場合のみ** • IDLE
BUDDYSIGNONTIME	ユーザーが IM サーバーにサインオンした日時です。ユーザーが現在サインオンしていない場合は空です。XMPP および Sametime の場合は常に空の文字列です。
BUDDYSTATUSTIME	ユーザーのステータスが変更された最新の日時です。
BUDDYCUSTOMAWAYMESSAGE	現在のステータスを示すためにユーザーが設定したカスタム不在メッセージがある場合、そのカスタム不在メッセージです。
BUDDYOWNER	この ID に関連付けられたクライアントおよびプロトコルを示す文字列です。形式は、< クライアント >@< プロトコル > です。
BUDDYLISTTIYPE	このバディリストに適用されるリストのタイプです。次のいずれかです。 • BUDDY_LIST ゲートウェイで受信できるプレゼンスステータス情報を持つユーザーのリストです。 • DENY_LIST ゲートウェイ ID に関するプレゼンス情報を取得できないユーザーのリストです。 • PERMIT_LIST プレゼンス情報メッセージをゲートウェイ ID に送信できるユーザーのリストです。 • REVERSE_LIST メッセージの使用を許可しないユーザーのリストです。

フィールド	説明
BUDDYIDLETIME	バディステータスが IDLE の場合、そのバディがアイドルであった時間の長さです。XMPP または SameTime の場合は常に 0 です。
BUDDYISMOBILE	ユーザーがモバイルデバイスを使用しているかどうかを true または false で示します。XMPP または SameTime の場合は常に false です。
BUDDYWARNINGPERCENT	ユーザーの警告のパーセント値です。XMPP または SameTime の場合は常に 0 です。

例
『ColdFusion アプリケーションの開発』の GatewayHelper example を参照してください。この例では、すべての GatewayHelper クラスメソッドを使用しています。このメソッドを使用してバディカスタム不在メッセージを取得する場合の例については、onBuddyStatus を参照してください。

getBuddyList

説明
ゲートウェイのユーザー ID のバディリストを取得します。

シンタックス
```
array = getBuddyList()
```

関連項目
addBuddy、getBuddyInfo、removeBuddy、『ColdFusion アプリケーションの開発』の Using the GatewayHelper object

戻り値
ゲートウェイのバディリスト内のユーザーの ID (バディ名) の配列、このゲートウェイが通常通信するインスタントメッセージング ID のリスト。

例
『ColdFusion アプリケーションの開発』の GatewayHelper example を参照してください。この例では、すべての GatewayHelper クラスメソッドを使用しています。

getCustomAwayMessage

説明
gatewayHelper オブジェクトの setStatus メソッドによりゲートウェイのカスタム不在メッセージが設定されている場合、そのカスタム不在メッセージを返します。

シンタックス
```
string = getCustomAwayMessage()
```

関連項目
getStatusAsString、getStatusTimeStamp、isOnline、setStatus、『ColdFusion アプリケーションの開発』の Using the GatewayHelper object

戻り値
gatewayHelper オブジェクトの setStatus メソッドによりゲートウェイのカスタム不在メッセージが設定されている場合、そのカスタム不在メッセージです。

例
『ColdFusion アプリケーションの開発』の GatewayHelper example を参照してください。この例では、すべての GatewayHelper クラスメソッドを使用しています。

getDenyList

説明
アクセス許可モードが DENY_SOME に設定されている場合、ゲートウェイに関するステート情報を送信しないように IM サーバーが指示されたユーザーのリストを返します。

シンタックス
```
array = getDenyList()
```

関連項目
addDeny、addPermit、getPermitList、getPermitMode、removeDeny、removePermit、setPermitMode、『ColdFusion アプリケーションの開発』の Using the GatewayHelper object

戻り値
ゲートウェイのアクセス拒否リスト内のユーザーの ID (バディ名) の配列、IM サーバーがプレゼンスステータス情報を送信しない ID のリスト。

注意：XMPP サーバーがアクセス許可管理をサポートしない場合、この関数は常に false を返します。

例
『ColdFusion アプリケーションの開発』の GatewayHelper example を参照してください。この例では、すべての GatewayHelper クラスメソッドを使用しています。

getName

説明
ゲートウェイのユーザー名を返します。

シンタックス
```
string = getName()
```

関連項目
getProtocolName、numberOfMessagesReceived、numberOfMessagesSent、setNickName、『ColdFusion アプリケーションの開発』の Using the GatewayHelper object

戻り値
ゲートウェイ設定ファイルで指定した、ゲートウェイのユーザー名

例
『ColdFusion アプリケーションの開発』の GatewayHelper example を参照してください。この例では、すべての GatewayHelper クラスメソッドを使用しています。

getNickName

説明
gatewayHelper オブジェクトの setNickName メソッドによりゲートウェイのニックネーム (表示名) が設定されている場合、そのニックネームを返します。

シンタックス
```
string = getNickName()
```

関連項目
getName、getProtocolName、numberOfMessagesReceived、numberOfMessagesSent、setNickName、『ColdFusion アプリケーションの開発』の Using the GatewayHelper object

戻り値
ゲートウェイのニックネームがある場合はそのニックネーム、ない場合は空の文字列。

例
『ColdFusion アプリケーションの開発』の GatewayHelper example を参照してください。この例では、すべての GatewayHelper クラスメソッドを使用しています。

getPermitList

説明
ゲートウェイに関するステート情報を送信するように IM サーバーが指示されたユーザーのリストを返します。

シンタックス
```
array = getPermitList()
```

関連項目
addDeny、addPermit、getDenyList、getPermitMode、removeDeny、removePermit、setPermitMode、『ColdFusion アプリケーションの開発』の Using the GatewayHelper object

戻り値
ゲートウェイのアクセス許可リスト内のユーザーの ID (バディ名) の配列、アクセス許可モードが PERMIT_SOME に設定されている場合に IM サーバーがプレゼンスステータス情報を送信する ID のリスト。

注意：XMPP サーバーがアクセス許可管理をサポートしない場合、この関数は常に false を返します。

例
『ColdFusion アプリケーションの開発』の GatewayHelper example を参照してください。この例では、すべての GatewayHelper クラスメソッドを使用しています。

getPermitMode

説明
IM サーバーからゲートウェイのアクセス許可モードを取得します。このアクセス許可モードにより、すべてのユーザーがゲートウェイのオンラインステータス情報を取得できるかどうか、サーバーが、ステータス情報を取得するユーザーを制御するのにアクセス許可リストまたはアクセス拒否リストを使用するのかどうかが決定されます。

シンタックス
```
string = getPermitMode()
```

関連項目
addDeny、addPermit、getDenyList、getPermitList、removeDeny、removePermit、setPermitMode、『ColdFusion アプリケーションの開発』の Using the GatewayHelper object

戻り値
ゲートウェイのアクセス許可モード。次のいずれかです。

モード	説明
PERMIT_ALL	(デフォルト) すべてのユーザーがゲートウェイのオンラインプレゼンスおよびステートを把握することを許可します。
PERMIT_SOME	アクセス許可リスト内のユーザーのみがゲートウェイのオンラインプレゼンスおよびステートを把握することを許可します。
DENY_SOME	アクセス拒否リスト内のユーザーがゲートウェイのオンラインプレゼンスおよびステートを把握できないようにします。

注意：XMPP サーバーがアクセス許可管理をサポートしない場合、この関数は常に PERMIT_ALL を返します。

例
『ColdFusion アプリケーションの開発』の GatewayHelper example を参照してください。この例では、すべての GatewayHelper クラスメソッドを使用しています。

getProtocolName

説明
ゲートウェイのインスタントメッセージングプロトコルの名前を取得します。

シンタックス
```
string = getProtocolName()
```

関連項目
getName、getNickName、numberOfMessagesReceived、numberOfMessagesSent、setNickName、『ColdFusion アプリケーションの開発』の Using the GatewayHelper object

戻り値
ゲートウェイタイプによって決まるゲートウェイのプロトコル。次のいずれかです。

- JABBER (XMPP の場合)

- SAMETIME

例
『ColdFusion アプリケーションの開発』の GatewayHelper example を参照してください。この例では、すべての GatewayHelper クラスメソッドを使用しています。

getStatusAsString

説明
ゲートウェイのオンラインステータスをテキスト文字列として取得します。

シンタックス
```
string = getStatusAsString()
```

関連項目
getCustomAwayMessage、getStatusTimeStamp、isOnline、setStatus、『ColdFusion アプリケーションの開発』の Using the GatewayHelper object

戻り値
ゲートウェイのオンラインステータス。次のいずれかです。

- ONLINE
- OFFLINE
- AWAY
- DO NOT DISTURB

XMPP の場合のみ

- NOT AVAILABLE
- FREE TO CHAT

Sametime の場合のみ

- IDLE

使用方法
DO NOT DISTURB、NOT AVAILABLE、および FREE TO CHAT の各文字列は、setStatus メソッドで使用するステータス値とは異なります。このステータス名にはスペースを使用できません。

例
『ColdFusion アプリケーションの開発』の GatewayHelper example を参照してください。この例では、すべての GatewayHelper クラスメソッドを使用しています。

getStatusTimeStamp

説明
ゲートウェイがオンラインステータスを変更した日時を取得します。

シンタックス

```
date-time object = getStatusTimeStamp()
```

関連項目

getCustomAwayMessage、getStatusAsString、isOnline、setStatus、『ColdFusion アプリケーションの開発』の Using the GatewayHelper object

戻り値

通常、setStatus gatewayHelper オブジェクトメソッドを呼び出すことにより、ゲートウェイがオンラインステータスを変更した日時。

例

『ColdFusion アプリケーションの開発』の GatewayHelper example を参照してください。この例では、すべての GatewayHelper クラスメソッドを使用しています。

isOnline

説明

ゲートウェイがインスタントメッセージングサーバーに接続されているかどうかを判別します。

シンタックス

```
Boolean = isOnline()
```

関連項目

getCustomAwayMessage、getStatusAsString、getStatusTimeStamp、setStatus、『ColdFusion アプリケーションの開発』の Using the GatewayHelper object

戻り値

ゲートウェイが IM サーバーに接続している場合は true、接続していない場合は false。

例

『ColdFusion アプリケーションの開発』の GatewayHelper example を参照してください。この例では、すべての GatewayHelper クラスメソッドを使用しています。

numberOfMessagesReceived

説明

起動してから、ゲートウェイが受け取ったメッセージの数を取得します。

シンタックス

```
integer = numberOfMessagesReceived()
```

関連項目

getName、getNickName、getProtocolName、numberOfMessagesSent、setNickName、『ColdFusion アプリケーションの開発』の Using the GatewayHelper object

戻り値
起動してから、ゲートウェイが受け取ったメッセージの数

例
『ColdFusion アプリケーションの開発』の GatewayHelper example を参照してください。この例では、すべての GatewayHelper クラスメソッドを使用しています。

numberOfMessagesSent

説明
起動してから、ゲートウェイが送信したメッセージの数を取得します。

シンタックス
```
integer = numberOfMessagesSent()
```

関連項目
getName、getNickName、getProtocolName、numberOfMessagesReceived、setNickName、『ColdFusion アプリケーションの開発』の Using the GatewayHelper object

戻り値
起動してから、ゲートウェイが送信したメッセージの数

例
『ColdFusion アプリケーションの開発』の GatewayHelper example を参照してください。この例では、すべての GatewayHelper クラスメソッドを使用しています。

removeBuddy

説明
ゲートウェイのバディリスト内のグループから ID を削除し、そのバディのオンラインプレゼンスステートを含むゲートウェイメッセージを送信しないように IM サーバーに指示します。

シンタックス
```
Boolean = removeBuddy(name, group)
```

関連項目
addBuddy、getBuddyInfo、getBuddyList、removeDeny、removePermit、『ColdFusion アプリケーションの開発』の Using the GatewayHelper object

パラメータ

パラメータ	説明
name	バディリストから削除する対象の固有のインスタントメッセージングユーザー名です。
group	削除するユーザーが属するグループの名前です。group パラメータが空の文字列の場合、ゲートウェイは General グループを使用します。

戻り値
ID がグループから削除された場合は true、それ以外の場合は false。

使用方法
ユーザーがバディリスト内の複数のグループに属する場合は、各グループからバディを個別に削除します。その名前をすべてのグループから削除するまで、IM サーバーはステータス更新の送信を停止しません。

例
『ColdFusion アプリケーションの開発』の GatewayHelper example を参照してください。この例では、すべての GatewayHelper クラスメソッドを使用しています。

removeDeny

説明
ゲートウェイのアクセス拒否リスト内のグループから ID を削除します。ゲートウェイのアクセス許可モードが DENY_SOME の場合、指定されたユーザーはゲートウェイのプレゼンスステートのメッセージを受信できます。

シンタックス
```
Boolean = removeDeny(name, group)
```

関連項目
addDeny、addPermit、getDenyList、getPermitList、getPermitMode、removeBuddy、removePermit、setPermitMode、『ColdFusion アプリケーションの開発』の Using the GatewayHelper object

パラメータ

パラメータ	説明
name	アクセス拒否リストから削除する対象の固有のインスタントメッセージングユーザー名です。
group	削除するユーザーが属するグループの名前です。group パラメータが空の文字列の場合、ゲートウェイは General グループを使用します。

戻り値
ID がグループから削除された場合は true、それ以外の場合は false。

注意：XMPP サーバーがアクセス許可管理をサポートしない場合、この関数は常に false を返します。

使用方法
ユーザーがアクセス拒否リスト内の複数のグループに属する場合は、各グループからユーザーを個別に削除します。グループからユーザーの名前を削除すると、IM サーバーではステータス更新の送信が有効になります。

例
『ColdFusion アプリケーションの開発』の GatewayHelper example を参照してください。この例では、すべての GatewayHelper クラスメソッドを使用しています。

removePermit

説明
ゲートウェイのアクセス許可リスト内のグループから ID を削除します。ゲートウェイのアクセス許可モードが PERMIT_SOME の場合、指定されたユーザーはゲートウェイのプレゼンスステートのメッセージを受信できません。

シンタックス
```
Boolean = removePermit(name, group)
```

関連項目
addDeny、addPermit、getDenyList、getPermitList、getPermitMode、removeBuddy、removeDeny、setPermitMode、『ColdFusion アプリケーションの開発』の Using the GatewayHelper object

パラメータ

パラメータ	説明
name	アクセス許可リストから削除する対象の固有のインスタントメッセージングユーザー名です。
group	削除するユーザーが属するグループの名前です。group パラメータが空の文字列の場合、ゲートウェイは General グループを使用します。

戻り値
ID がグループから削除された場合は true、それ以外の場合は false。

注意：XMPP サーバーがアクセス許可管理をサポートしない場合、この関数は常に false を返します。

使用方法
ユーザーがアクセス許可リスト内の複数のグループに属する場合は、各グループからユーザーを個別に削除します。ただし、最初のグループからユーザーを削除すると、IM サーバーではステータス更新の送信が無効になります。

例
『ColdFusion アプリケーションの開発』の GatewayHelper example を参照してください。この例では、すべての GatewayHelper クラスメソッドを使用しています。

setNickName

説明
ゲートウェイのニックネーム (表示名) を設定します。

シンタックス
```
Boolean = setNickName(name)
```

関連項目
getName、getNickName、getProtocolName、numberOfMessagesReceived、numberOfMessagesSent、『ColdFusion アプリケーションの開発』の Using the GatewayHelper object

パラメータ

パラメータ	説明
name	このゲートウェイに関連付ける表示名です。この名前は、プロトコル固有である保証はありません。

戻り値
ニックネームが設定された場合は true、設定されない場合は false。

例
『ColdFusion アプリケーションの開発』の GatewayHelper example を参照してください。この例では、すべての GatewayHelper クラスメソッドを使用しています。

setPermitMode

説明
IM サーバーにゲートウェイのアクセス許可モードを設定します。このアクセス許可モードにより、すべてのユーザーがゲートウェイのオンラインステータス情報を取得できるかどうか、サーバーがステータス情報を取得するユーザーを制御するのにアクセス許可リストまたはアクセス拒否リストを使用するのかどうかが決まります。

シンタックス

```
Boolean = setPermitMode(permitMode)
```

関連項目
addDeny、addPermit、getDenyList、getPermitList、getPermitMode、removeDeny、removePermit、『ColdFusion アプリケーションの開発』の Using the GatewayHelper object

パラメータ

パラメータ	説明
permitMode	アクセス許可モードです。次のいずれかです。 • PERMIT_ALL すべてのユーザーがゲートウェイのオンラインプレゼンスおよびステートを把握することを許可します。この関数を呼び出さない場合のデフォルトです。 • PERMIT_SOME アクセス許可リスト内のユーザーのみがゲートウェイのオンラインプレゼンスおよびステートを把握することを許可します。 • DENY_SOME アクセス拒否リスト内のすべてのユーザーがゲートウェイのオンラインプレゼンスおよびステートを把握できないようにします。

戻り値
アクセス許可モードが設定された場合は true、設定されない場合は false。

注意：XMPP サーバーがアクセス許可管理をサポートしない場合、この関数は PERMIT_ALL 以外のすべての値に対して false を返します。

例
『ColdFusion アプリケーションの開発』の GatewayHelper example を参照してください。この例では、すべての GatewayHelper クラスメソッドを使用しています。

setStatus

説明
任意のカスタム不在メッセージなど、ゲートウェイのオンラインプレゼンスステータスを設定します。

シンタックス
```
Boolean = setStatus(status, customAwayMsg)
```

関連項目
getCustomAwayMessage、getStatusAsString、getStatusTimeStamp、isOnline、『ColdFusion アプリケーションの開発』の Using the GatewayHelper object

パラメータ

パラメータ	説明
status	ゲートウェイのオンラインプレゼンスステータス。次のいずれかです。 • ONLINE • AWAY • DND (Do Not Disturb) • NA (Not Available) • FREE_TO_CHAT • IDLE **XMPP の場合のみ** • NA (Not Available) • FREE_TO_CHAT • IDLE **Sametime の場合のみ** • IDLE
customAwayMsg	ステータスのカスタムメッセージを含むテキスト文字列です。カスタム不在メッセージが必要ない場合、空の文字列にすることもできます。

戻り値
処理が成功した場合は true、失敗した場合は false。プロトコルについて無効なステータスを渡すと、このメソッドは false を返します。

使用方法
オフラインにするために setStatus メソッドを使用しないでください。このメソッドは OFFLINE というパラメータを受け入れますが、ゲートウェイは直ちにオンラインにリセットされます。ゲートウェイをオフラインに設定するには、ColdFusion Administrator でゲートウェイインスタンスを停止するか、CFIDE.adminapi.eventgateway CFC の stopGatewayInstance メソッドを使用します。

例

『ColdFusion アプリケーションの開発』の GatewayHelper example を参照してください。この例では、すべての GatewayHelper クラスメソッドを使用しています。

SMS ゲートウェイ CFEvent の構造体とコマンド

このセクションでは、SMS ゲートウェイリスナー CFC および CFML SendGatewayMessage 関数で使用する次の構造体の詳細な内容について説明します。

関連項目

1598 ページの「SMS ゲートウェイ着信メッセージ CFEvent 構造体」

1599 ページの「SMS ゲートウェイメッセージ送信コマンド」

SMS ゲートウェイ着信メッセージ CFEvent 構造体

SMS ゲートウェイは、CFC リスナーメソッドに送信する CFEvent インスタンスに次の情報を挿入します。

フィールド	値
OriginatorID	PDU source_addr フィールドの内容。メッセージを送信したデバイスのアドレスです。
CfcMethod	リスナー CFC のメソッド名 . 設定ファイルの cfc-method エントリの値、または設定ファイルにこのエントリが設定されていない場合は onIncomingMessage。
Data.MESSAGE	PDU の short_message フィールドの内容。
Data.sourceAddress	このメッセージを送信したデバイスのアドレス。
Data.destAddress	メッセージの送信先のアドレス。ゲートウェイの設定ファイルの address-range 設定により指定された範囲内のアドレスです。
Data.esmClass	PDU esm_class フィールドの内容。メッセージタイプを識別します。バイト値を表す 0 ～ 255 の範囲の数値です。ここで、ビット 2 ～ 5 (0 でインデックス付け) はメッセージタイプを示すので、data.MESSAGE フィールドの内容は次のようになります (予約された値は省略されます)。 xx0000xx 標準メッセージ xx0001xx SMSC 配信受信 xx0010xx SME 配信承認 xx0100xx SME 手動 / ユーザー承認 xx0110xx 会話中断 (Korean CDMA の場合のみ) xx1000xx 中間配信通知 このフィールドの詳細については、SMPP の仕様を参照してください。
Data.protocol	PDU protocol_id フィールドの内容。GSM ネットワークから送信されるメッセージの場合にのみ意味があります。詳細については、GSM 03.40 の仕様を参照してください。
Data.priority	PDU priority_flag フィールドの内容。発信元の SME によって設定されるメッセージの priority レベルです。範囲は 0 ～ 3 で、0 は最下位レベル、3 は最上位レベルです。priority レベルの固有の意味は、発信元のネットワークにより異なります。詳細については、SMPP の仕様を参照してください。

フィールド	値
Data.registeredDelivery	PDU registered_delivery フィールドの内容。送信者がリクエストしたタイプ (配信受信または配信承認) を示します。0 ～ 32 の範囲の数値で、次の値の合計を表します。 0: SMS 配信受信のリクエストなし、または 1: 配信の成功または失敗に対する SMSC 配信受信のリクエスト、または 2: 配信の失敗のみに対する SMSC 配信受信のリクエスト および 0: SME 配信承認のリクエストなし、または 4: SME 配信承認のリクエスト、または 8: SME 手動 / ユーザー承認のリクエスト、または 12: 配信および手動 / ユーザー承認の両方のリクエスト および 0: 中間配信通知のリクエストなし、または 16: 中間配信通知のリクエスト
Data.DataCoding	PDU data_coding フィールドの内容。メッセージの内容の文字セットまたは文字以外のデータ型を示します。次のとおりです。 00000000 SMSC デフォルトアルファベット 00000001 IA5 (CCITT T.50)/ASCII (ANSI X3.4) 00000010 指定なしの Octet (8 ビットバイナリ) 00000011 Latin 1 (ISO-8859-1) 00000100 指定なしの Octet (8 ビットバイナリ) 00000101 JIS (X 0208-1990) 00000110 Cyrillic (ISO-8859-5) 00000111 Latin/Hebrew (ISO-8859-8) 00001000 UCS2 (ISO/IEC-10646) 00001001 ピクトグラムエンコーディング 00001010 ISO-2022-JP (Music コード) 00001101 拡張漢字 JIS(X 0212-1990) 00001110 KS C 5601 11xxxxxx GSM コントロールのみで使用 (GSM 03.38 の仕様を参照) 詳細については、SMPP の仕様を参照してください。
Data.messageLength	short_message フィールドの長さ。
GatewayType	常に SMS。

これらのフィールドの一部の意味に関する詳細、および SMS ゲートウェイリスナー CFC メソッドでの着信 SMS メッセージの処理方法の詳細については、『ColdFusion アプリケーションの開発』の Handling incoming messages を参照してください。

SMS ゲートウェイメッセージ送信コマンド

SMS (Short Message Service) タイプのゲートウェイを使用する ColdFusion アプリケーションは、発信メッセージを介してイベントゲートウェイに次のコマンドを送信できます。

関連項目

1600 ページの「submit コマンド」

1601 ページの「submitMulti コマンド」

1602 ページの「data コマンド」

submit コマンド

SMPP SUBMIT_SM PDU で単一の宛先アドレスにメッセージを送信するために、SendGatewayMessage 関数の **Data** パラメータで使用する構造体、または CFC リスナーメソッドの戻り変数には、次のフィールドが含まれています。これらのフィールドの詳細については、SMPP3.4 仕様の SUBMIT_MULTI PDU に関するドキュメントを参照してください。このドキュメントは、www.smsforum.net/ の SMS Forum からダウンロードできます。

必須フィールド

フィールド	内容
command	存在する場合、この値を submit に設定する必要があります。このフィールドを省略した場合、イベントゲートウェイは submit メッセージを送信します。
shortMessage または messagePayload	メッセージの内容です。いずれかのフィールドを指定する必要があります。両方は指定できません。SMPP の仕様では、shortMessage フィールドの最大サイズは 254 バイトに制限されています。一部のキャリアでは、このサイズをさらに制限している場合もあります。messagePayload フィールドには、最大で 64K バイトのデータを設定できます。これは、0x0424 で始まり、その後にデータ量の長さを指定する 2 バイトが続き、さらにメッセージのコンテンツが続く必要があります。
destAddress	必須です。メッセージの送信先アドレス。
sourceAddress	このアプリケーションのアドレス。このフィールドは省略できます。設定ファイルがアプリケーションを指定します。

オプションフィールド

SMS イベントゲートウェイ設定ファイルの次のオプションフィールドに使用するデフォルト値を設定できます。デフォルト値の詳細については、『ColdFusion アプリケーションの開発』の Configuring an SMS event gateway を参照してください。

destAddress_npi	destAddress_ton	serviceType	

次のオプションフィールドにはデフォルト値はありません。

alertOnMsgDelivery	EsmClass	priorityFlag	smDefaultMsgId
callbackNum	ItsReplyType	PrivacyIndicator	SmsSignal
callbackNumAtag	ItsSessionInfo	protocolId	SourceAddrSubunit
callbackNumPresInd	LanguageIndicator	registeredDelivery	SourcePort
dataCoding	MoreMsgsToSend	replaceIfPresent	SourceSubaddress
DestAddrSubunit	MsMsgWaitFacilities	SarMsgRefNum	UserMessageReference
DestinationPort	MsValidity	SarSegmentSeqnum	UserResponseCode
DestSubaddress	NumberOfMessages	SarTotalSegments	UssdServiceOp
DisplayTime	PayloadType	scheduleDeliveryTime	validityPeriod

例

次のリスナー CFC のサンプル onIncomingMessage メソッドの例は、submit コマンドを使用して、着信 SMS メッセージをメッセージの発信元にエコーします。

```
<cffunction name="onIncomingMessage" output="no">
    <cfargument name="CFEvent" type="struct" required="yes">
    <!--- Create a return structure that contains the message. --->
    <cfset retValue = structNew()>
    <cfset retValue.command = "submit">
    <cfset retValue.destAddress = arguments.CFEvent.originatorid>
    <cfset retValue.shortMessage = "Echo: " & CFEvent.Data.MESSAGE>
    <!--- Send the message back. --->
    <cfreturn retValue>
</cffunction>
```

submitMulti コマンド

SMPP SUBMIT_MULTI PDU を使って複数の受信者に 1 つのテキストメッセージを送信するために、SendGatewayMessage 関数の **Data** パラメータまたは CFC リスナーメソッドの戻り変数には、一般的に次のフィールドが含まれています。これらのフィールドの詳細については、SMPP3.4 仕様の SUBMIT_MULTI PDU に関するドキュメントを参照してください。このドキュメントは、www.smsforum.net/ の SMS Forum からダウンロードできます。

必須フィールド

フィールド	内容
command	submitMulti を指定する必要があります。
shortMessage または messagePayload	メッセージの内容です。いずれかのフィールドを指定する必要があります。両方は指定できません。SMPP の仕様では、shortMessage フィールドの最大サイズは 254 バイトに制限されています。一部のキャリアでは、このサイズをさらに制限している場合もあります。messagePayload フィールドには、最大で 64K バイトのデータを設定できます。これは、0x0424 で始まり、その後にデータ量の長さを指定する 2 バイトが続き、さらにメッセージのコンテンツが続く必要があります。
destAddress	宛先アドレスの ColdFusion 配列 (必須)。 これらのアドレスに対して個別の TON および NPI の値を指定することはできません。すべてのアドレスで同じ設定を使用する必要があります。
sourceAddress	このアプリケーションのアドレス。このフィールドは省略できます。設定ファイルがアプリケーションを指定します。

オプションフィールド

次のオプションフィールドでは、SMS イベントゲートウェイ設定ファイルのデフォルト値のセットを使用することができます。デフォルト値の詳細については、『ColdFusion アプリケーションの開発』の Configuring an SMS event gateway を参照してください。

destAddress_npi	destAddress_ton	serviceType	

次のオプションフィールドにはデフォルト値はありません。

alertOnMsgDelivery	DisplayTime	protocolId	SmsSignal
callbackNum	EsmClass	registeredDelivery	SourceAddrSubunit
callbackNumAtag	LanguageIndicator	replaceIfPresent	SourcePort

callbackNumPresInd	MsMsgWaitFacilities	SarMsgRefNum	SourceSubaddress
dataCoding	MsValidity	SarSegmentSeqnum	UserMessageReference
DestAddrSubunit	PayloadType	SarTotalSegments	validityPeriod
DestinationPort	priorityFlag	scheduleDeliveryTime	
DestSubaddress	PrivacyIndicator	smDefaultMsgId	

例

次の onIncomingMessage メソッドの例は、着信メッセージを発信元アドレスにエコーする応答を送信し、その応答のコピーを 2 番目のアドレスに送信します。

```
<cffunction name="onIncomingMessage" output="no">
    <cfargument name="CFEvent" type="struct" required="yes">
    <!--- Get the message. --->
    <cfset data=cfevent.DATA>
    <cfset message="#data.message#">
    <!--- Create the return structure. --->
    <cfset retValue = structNew()>
    <cfset retValue.command = "submitmulti">
    <cfset retValue.destAddresses=arraynew(1)>
    <!--- One destination is incoming message originator;
                    get the address from CFEvent originator ID. --->
    <cfset retValue.destAddresses[1] = arguments.CFEvent.originatorid>
    <cfset retValue.destAddresses[2] = "12345">
    <cfset retValue.shortMessage = "echo: " & message>
<cfreturn retValue>
</cffunction>
```

data コマンド

SMPP DATA_SM PDU を使って 1 つの宛先アドレスにバイナリデータを送信するには、SendGatewayMessage 関数の **Data** パラメータまたは CFC リスナーメソッドの戻り変数に、次のフィールドが設定されている必要があります。これらのフィールドの詳細については、SMPP3.4 仕様の SUBMIT_MULTI PDU に関するドキュメントを参照してください。このドキュメントは、www.smsforum.net/ の SMS Forum からダウンロードできます。

必須フィールド

フィールド	内容
command	data を指定する必要があります。
messagePayload	メッセージデータ。データをバイナリ形式に変換するには、ColdFusion ToBinary 関数を使用します。
destAddress	メッセージの送信先アドレス。
sourceAddress	このアプリケーションのアドレス。このフィールドは省略できます。設定ファイルがアプリケーションを指定します。

オプションフィールド

次のオプションフィールドでは、SMS イベントゲートウェイ設定ファイルのデフォルト値のセットを使用することができます。デフォルト値の詳細については、『ColdFusion アプリケーションの開発』の Configuring an SMS event gateway を参照してください。

destAddress_npi	destAddress_ton	serviceType	

次のオプションフィールドにはデフォルト値はありません。

alertOnMsgDelivery	DestTelematicsId	NetworkErrorCode	SetDpf
callbackNum	DisplayTime	NumberOfMessages	SmsSignal
callbackNumAtag	EsmClass	PayloadType	SourceAddrSubunit
callbackNumPresInd	ItsReplyType	PrivacyIndicator	SourceBearerType
dataCoding	ItsSessionInfo	QosTimeToLive	SourceNetworkType
DestAddrSubunit	LanguageIndicator	ReceiptedMessgeId	SourcePort
DestBearerType	MessageState	registeredDelivery	SourceSubaddress
DestNetworkType	MoreMsgsToSend	SarMsgRefNum	SourceTelematicsId
DestinationPort	MsMsgWaitFacilities	SarSegmentSeqnum	UserMessageReference
DestSubaddress	MsValidity	SarTotalSegments	UserResponseCode

例

次の onIncomingMessage メソッドの例は、着信メッセージをバイナリデータに変換し、メッセージのバイナリバージョンを発信元アドレスに送り返します。

```
<cffunction name="onIncomingMessage" output="no">
    <cfargument name="CFEvent" type="struct" required="yes">
    <!--- Get the message --->
    <cfset data=CFEvent.DATA>
    <cfset message="#data.message#">
    <!--- Create the return structure --->
    <cfset retValue = structNew()>
    <cfset retValue.command = "data">
    <!--- Sending to incoming message originator; get value from CFEvent. --->
    <cfset retValue.destAddress = arguments.CFEvent.originatorid>
    <cfset retValue.messagePayload = tobinary(tobase64("echo: " & message))>
    <cfreturn retValue>
</cffunction>
```

CFML イベントゲートウェイ SendGatewayMessage の data パラメータ

ColdFusion CFML ゲートウェイタイプにより、CFC メソッドを非同期で呼び出すことができます。SendGatewayMessage 関数の **data** パラメータで使用する構造体には、2 つのフィールドタイプを含めることができます。

- CFC で使用するための集計内容を任意の数のフィールドに含めることができます。
- いくつかのオプションフィールドで、ゲートウェイから CFC への情報の送信方法を設定できます。

  CFML ゲートウェイは次のオプションフィールドを検索し、存在する場合はそのフィールドを使用して、メッセージの送信方法を決定します。CFC メソッドに送信するデータに対して、これらのフィールド名を使用しないでください。

フィールド	用途
cfcpath	ColdFusion Administrator で指定されている CFC パスを上書きします。このフィールドを使用すれば、複数の CFC に対して、ColdFusion Administrator で単一のゲートウェイ設定を使用することができます。このフィールドでは CFEvent オブジェクトの CFCPath 変数を設定します。
method	CFC で起動するメソッドの名前を指定します。デフォルト値は onIncomingMessage です。このフィールドを使用すれば、複数のメソッドを持つ CFC に対して、ColdFusion Administrator で単一のゲートウェイ設定を使用することができます。このフィールドでは CFEvent オブジェクトの CFCMethod 変数を設定します。
originatorID	ColdFusion から CFC に送信する CFEvent オブジェクトの originatorID フィールドを設定します。デフォルト値は CFMLGateway です。
timeout	タイムアウトの値を秒単位で設定します。ここで設定した時間内に、リスナー CFC がイベントリクエストを処理して結果を返さなかった場合、ColdFusion ゲートウェイサービスはリクエストを終了します。デフォルト値は、ColdFusion Administrator の [サーバーの設定] ページで設定したリクエストタイムアウトの値です。デフォルトのタイムアウト時間よりリクエストの処理時間が長くなる可能性がある場合は、この値を設定します。このフィールドでは CFEvent オブジェクトの CFCTimeout 変数を設定します。

例

次の例は、logger.CFC ファイル内の a logevent メソッドにメッセージを送信する CFML ページで構成されています。CFML ページでは、呼び出す CFC とメソッドを指定し、OriginatorID を設定します。

```
 <h3>Sending an event using a generic CFML event gateway and specifying the CFC and method.</h3>
<cfscript>
    status = False;
    props = structNew();
    props.cfcpath="C:\CFusionMX7\gateway\cfc\MyCFCs\logger.cfc";
    props.method="logEvent";
    props.OriginatorID=CGI.SCRIPT_NAME;
    props.Message="Replace me with a variable with data to log";
    props.file="GenericCFCtest";
    props.type="warning";
    status = SendGatewayMessage("DefaultCFC", props);
    if (status IS True)
        WriteOutput('Event Message "#props.Message#" has been sent.');
</cfscript>
```

CFC メソッドは、OriginatorID および CFEvent パラメータのデータフィールドの message、file、type の各フィールドを使用して、ログファイルとメッセージを指定します。

```
 <cfcomponent>
    <cffunction name="logEvent" output="no">
        <cfargument name="CFEvent" type="struct" required="yes">
         <cfscript>
            if (NOT IsDefined("CFEvent.Data.file")) {
                CFEvent.Data.file="defaultEventLog"; }
            if (NOT IsDefined("CFEvent.Data.type")) {
                CFEvent.Data.type="information"; }
        </cfscript>
        <cflog text="Message from #CFEvent.originatorID#: #CFEvent.Data.message#"
            file="#CFEvent.data.file#" type="#CFEvent.Data.type#" >
    </cffunction>
</cfcomponent>
```

第 10 章：ColdFusion C++ CFX リファレンス

ColdFusion には、ColdFusion 拡張機能を構築するための CFXAPI クラスおよびメソッドが用意されています。

C++ クラスの概要

次の表に、CFXAPI クラスとそのメソッドを示します。

クラス	メソッド
CCFXException クラス	CCFXException::GetError CCFXException::GetDiagnostics
CCFXQuery クラス	CCFXQuery::AddRow CCFXQuery::GetColumns CCFXQuery::GetData CCFXQuery::GetName CCFXQuery::GetRowCount CCFXQuery::SetData
CCFXRequest クラス	CCFXRequest::AddQuery CCFXRequest::AttributeExists CCFXRequest::CreateStringSet CCFXRequest::Debug CCFXRequest::GetAttribute CCFXRequest::GetAttributeList CCFXRequest::GetCustomData CCFXRequest::GetQuery CCFXRequest::ReThrowException CCFXRequest::SetCustomData CCFXRequest::SetVariable CCFXRequest::ThrowException CCFXRequest::Write CCFXRequest::WriteDebug
CCFXStringSet クラス	CCFXStringSet::AddString CCFXStringSet::GetCount CCFXStringSet::GetIndexForString CCFXStringSet::GetString

非推奨のクラスメソッド

次の CFXAPI クラスおよびメソッドの使用は非推奨になっています。これらのクラスおよびメンバーは動作せず、今後のリリースではエラーが発生する可能性があります。

クラス	非推奨のメンバー	非推奨となった ColdFusion リリース
CCFXQuery クラス	CCFXQuery::SetQueryString CCFXQuery::SetTotalTime	ColdFusion MX ColdFusion MX
CCFXRequest クラス	CCFXRequest::GetSetting	ColdFusion MX

CCFXException クラス

これは、CFX (ColdFusion Extension) プロシージャの処理中に発生した例外を表す抽象クラスです。

このタイプの例外は、CCFXRequest クラス、CCFXQuery クラス、および CCFXStringSet クラスで発生する可能性があります。したがって、CFX (ColdFusion Extension) コードは、このタイプの例外が処理されるように作成する必要があります。詳細については、CCFXRequest::ThrowException および CCFXRequest::ReThrowException を参照してください。

クラスメソッド

virtual LPCSTR GetError()	CCFXException::GetError 関数は、通常のエラーメッセージを返します。
virtual LPCSTR GetDiagnostics()	CCFXException::GetDiagnostics 関数は、詳しいエラー情報を返します。

CCFXException::GetError

説明
処理時に発生する例外に関する基本的なユーザー出力を提供します。

CCFXException::GetDiagnostics

説明
処理時に発生する例外に関する詳細なユーザー出力を提供します。

例
次のコードブロックは、GetError および GetDiagnostics が ThrowException および ReThrowException とどのように連動するかを示しています。

```
// Write output back to the user here...
    pRequest->Write( "Hello from CFX_FOO2!" ) ;
    pRequest->ThrowException("User Error", "You goof'd...");

    // Output optional debug info
    if ( pRequest->Debug() )
    {
        pRequest->WriteDebug( "Debug info..." ) ;
    }

    // Catch ColdFusion exceptions & re-raise them
    catch( CCFXException* e )
    {
        // This is how you would pull the error information
        LPCTSTR strError = e->GetError();
        LPCTSTR strDiagnostic = e->GetDiagnostics();

        pRequest->ReThrowException( e ) ;
    }

    // Catch ALL other exceptions and throw them as
    // ColdFusion exceptions (DO NOT REMOVE! --
    // this prevents the server from crashing in
    // case of an unexpected exception)
    catch( ... )
    {
        pRequest->ThrowException(
            "Error occurred in tag CFX_FOO2",
            "Unexpected error occurred while processing tag." ) ;
}
```

CCFXQuery クラス

これは、ColdFusion Extension (CFX) によって使用または作成されたクエリーを表す抽象クラスです。クエリーには、何行にも拡張されたデータ列 (複数の場合あり) が含まれます。

クラスメソッド

virtual int AddRow()	CCFXQuery::AddRow はクエリーに行を追加します。
virtual CCFXStringSet* GetColumns	CCFXQuery::GetColumns はクエリーの列名のリストを取得します。
virtual LPCSTR GetData(int iRow, int iColumn)	CCFXQuery::GetData はクエリーの行と列からデータ要素を取得します。
virtual LPCSTR GetName()	CCFXQuery::GetName はクエリーの名前を取得します。
virtual int GetRowCount()	CCFXQuery::GetRowCount はクエリーの行数を取得します。

virtual void SetData(int iRow, int iColumn, LPCSTR lpszData)	CCFXQuery::SetData はクエリーの行と列に含まれるデータ要素を設定します。
virtual void SetQueryString(LPCSTR lpszQuery)	この関数は非推奨となっています。今後のリリースではこのタグは機能せずエラーを引き起こす可能性があります。
virtual void SetTotalTime(DWORD dwMilliseconds)	この関数は非推奨となっています。今後のリリースではこのタグは機能せずエラーを引き起こす可能性があります。

CCFXQuery::AddRow

シンタックス

```
int CCFXQuery::AddRow(void)
```

説明

クエリーに行を追加します。クエリーに行を追加するには、この関数を呼び出します。

戻り値

クエリーに追加された行のインデックス

例

次の例では、3 列 (City、State、Zip) のクエリーに 2 行を追加しています。

```
// First row
    int iRow ;
iRow = pQuery->AddRow() ;
    pQuery->SetData( iRow, iCity, "Minneapolis" ) ;
    pQuery->SetData( iRow, iState, "MN" ) ;
    pQuery->SetData( iRow, iZip, "55345" ) ;

    // Second row
    iRow = pQuery->AddRow() ;
    pQuery->SetData( iRow, iCity, "St. Paul" ) ;
    pQuery->SetData( iRow, iState, "MN" ) ;
    pQuery->SetData( iRow, iZip, "55105" ) ;
```

CCFXQuery::GetColumns

シンタックス

```
CCFXStringSet* CCFXQuery::GetColumns(void)
```

説明

クエリーに含まれている列名のリストを取得します。

戻り値

クエリーに含まれている列のリストを含んでいる CCFXStringSet クラスのオブジェクト。リクエストが完了すると、返される文字列セットに割り当てられていたメモリが自動的に解放されます。

例

次の例では、列のリストを取得し、リスト内を巡回して、各列の名前をユーザーに返します。

```
// Get the list of columns from the query
    CCFXStringSet* pColumns = pQuery->GetColumns() ;
    int nNumColumns = pColumns->GetCount() ;

    // Print the list of columns to the user
    pRequest->Write( "Columns in query: " ) ;
    for( int i=1; i<=nNumColumns; i++ )
    {
        pRequest->Write( pColumns->GetString( i ) ) ;
        pRequest->Write( " " ) ;
    }
```

CCFXQuery::GetData

シンタックス

```
LPCSTR CCFXQuery::GetData(int iRow, int iColumn)
```

説明

クエリーの行と列からデータ要素を取得します。行と列のインデックスは 1 から始まります。CCFXQuery::GetRowCount を呼び出すと、クエリーの行数がわかります。CCFXQuery::GetColumns を使って列のリストを取得することができ、返された文字列セットに対して CCFXStringSet::GetCount を呼び出すと、クエリーの列数を判断できます。

戻り値

リクエストされたデータ要素の値を返します。

パラメータ

パラメータ	説明
iRow	データを取得する行です (1 から始まります)。
iColumn	データを取得する列です (1 から始まります)。

例

次の例では、クエリーの要素に対して繰り返し作業を行い、簡単なスペース区切り形式でクエリー内のデータをユーザーに出力します。

```
int iRow, iCol ;
    int nNumCols = pQuery->GetColumns()->GetCount() ;
    int nNumRows = pQuery->GetRowCount() ;
    for ( iRow=1; iRow<=nNumRows; iRow++ )
    {
        for ( iCol=1; iCol<=nNumCols; iCol++ )
        {
        pRequest->Write( pQuery->GetData( iRow, iCol ) ) ;
        pRequest->Write( " " ) ;
        }
        pRequest->Write( "<BR>" ) ;
    }
```

CCFXQuery::GetName

シンタックス

```
LPCSTR CCFXQuery::GetName(void)
```

説明
クエリーの名前を返します。

例
次の例では、クエリーの名前を取得して、ユーザーに出力します。

```
CCFXQuery* pQuery = pRequest->GetQuery() ;
    pRequest->Write( "The query name is: " ) ;
    pRequest->Write( pQuery->GetName() ) ;
```

CCFXQuery::GetRowCount

シンタックス
```
int CCFXQuery::GetRowCount(void)
```

説明
クエリーの行数を返します。

例
次の例では、クエリーの行数を取得して、ユーザーに出力します。

```
CCFXQuery* pQuery = pRequest->GetQuery() ;
    char buffOutput[256] ;
    wsprintf( buffOutput,
        "The number of rows in the query is %ld.",
        pQuery->GetRowCount() ) ;
    pRequest->Write( buffOutput ) ;
```

CCFXQuery::SetData

シンタックス
```
void CCFXQuery::SetData(int iRow, int iColumn, LPCSTR lpszData)
```

説明
クエリーの行と列に含まれるデータ要素を設定します。行と列のインデックスは 1 から始まります。ある行について、setData を呼び出す場合は、その前に CCFXQuery::AddRow を呼び出し、その戻り値を setData を呼び出すための行インデックスとして使用します。

パラメータ

パラメータ	説明
iRow	設定するデータ要素の行です (1 から始まります)。
iColumn	設定するデータ要素の列です (1 から始まります)。
lpszData	データ要素の新しい値です。

例
次の例では、3 列 (City、State、Zip) のクエリーに 2 行を追加しています。

```
// First row
    int iRow ;
    iRow = pQuery->AddRow() ;
    pQuery->SetData( iCity, iRow, "Minneapolis" ) ;
    pQuery->SetData( iState, iRow, "MN" ) ;
    pQuery->SetData( iZip, iRow, "55345" ) ;

    // Second row
    iRow = pQuery->AddRow() ;
    pQuery->SetData( iCity, iRow, "St. Paul" ) ;
    pQuery->SetData( iState, iRow, "MN" ) ;
    pQuery->SetData( iZip, iRow, "55105" ) ;
```

CCFXRequest クラス

CFX (ColdFusion Extension) へのリクエストを表す抽象クラスです。このクラスのインスタンスは、拡張 DLL のメイン関数に渡されます。このクラスでは、次のようなアクションのカスタム拡張で使用できるインターフェイスが提供されます。

- 変数の読み取りと書き込み
- 出力の戻し
- クエリーの作成と使用
- 例外の発生

クラスメソッド

virtual BOOL AttributeExists(LPCSTR lpszName)	CCFXRequest::AttributeExists はタグに属性が渡されたかどうかをチェックします。
virtual LPCSTR GetAttribute(LPCSTR lpszName)	CCFXRequest::GetAttribute は渡された属性の値を取得します。
virtual CCFXStringSet* GetAttributeList()	CCFXRequest::GetAttributeList はタグに渡された属性名の配列を取得します。
virtual CCFXQuery* GetQuery()	CCFXRequest::GetQuery はタグに渡されたクエリーを取得します。
virtual LPCSTR GetSetting(LPCSTR lpszSettingName)	CCFXRequest::GetSetting このメソッドは非推奨となっています。今後のリリースではこのタグは機能せずエラーを引き起こす可能性があります。
virtual void Write(LPCSTR lpszOutput)	CCFXRequest::Write はテキスト出力をユーザーに返します。
virtual void SetVariable(LPCSTR lpszName, LPCSTR lpszValue)	CCFXRequest::SetVariable はこのタグを含むテンプレートに変数を設定します。
virtual CCFXQuery* AddQuery(LPCSTR lpszName, CCFXStringSet* pColumns)	CCFXRequest::AddQuery はこのタグを含むテンプレートにクエリーを追加します。
virtual BOOL Debug()	CCFXRequest::Debug はタグに Debug 属性があるかどうかをチェックします。
virtual void WriteDebug(LPCSTR lpszOutput)	CCFXRequest::WriteDebug はデバッグストリームにテキスト出力を書き込みます。
virtual CCFXStringSet* CreateStringSet()	CCFXRequest::CreateStringSet は、CCFXStringSet インスタンスを割り当て、返します。
virtual void ThrowException(LPCSTR lpszError, LPCSTR lpszDiagnostics)	CCFXRequest::ThrowException はエラーを生成して、このリクエストの処理を終了します。

virtual void ReThrowException(CCFXException* e)	CCFXRequest::ReThrowException は検出したエラーを再び返します。
virtual void SetCustomData(LPVOID lpvData)	CCFXRequest::SetCustomData はリクエストとともに渡すタグ固有のカスタムデータを設定します。
virtual LPVOID GetCustomData()	CCFXRequest::GetCustomData はリクエストに対するタグ固有のカスタムデータを取得します。

CCFXRequest::AddQuery

シンタックス

```
CCFXQuery* CCFXRequest::AddQuery(LPCSTR lpszName, CCFXStringSet* pColumns)
```

説明

呼び出しテンプレートにクエリーを追加します。このクエリーには、テンプレート内の CFML タグ (cfoutput や cftable など) を使ってアクセスできます。AddQuery を呼び出した後は、クエリーが空 (0 行) になります。クエリーにデータを挿入するには、CCFXQuery::AddRow 関数および CCFXQuery::SetData 関数を呼び出します。

戻り値

テンプレートに追加されたクエリーへのポインタ (CCFXQuery クラスのオブジェクト)。リクエストが完了すると、返されるクエリーに割り当てられていたメモリが自動的に解放されます。

パラメータ

パラメータ	説明
lpszName	テンプレートに追加するクエリーの名前です (固有でなければなりません)。
pColumns	クエリーで使用する列の名前のリストです。

例

次の例では、呼び出しテンプレートに People という名前のクエリーが追加されます。クエリーは、2 列 (FirstName と LastName) と 2 行から構成されています。

```
// Create a string set and add the column names to it
    CCFXStringSet* pColumns = pRequest->CreateStringSet() ;
    int iFirstName = pColumns->AddString( "FirstName" ) ;
    int iLastName = pColumns->AddString( "LastName" ) ;

    // Create a query that contains these columns
    CCFXQuery* pQuery = pRequest->AddQuery( "People", pColumns ) ;

    // Add data to the query
    int iRow ;
    iRow = pQuery->AddRow() ;
    pQuery->SetData( iRow, iFirstName, "John" ) ;
    pQuery->SetData( iRow, iLastName, "Smith" ) ;
    iRow = pQuery->AddRow() ;
    pQuery->SetData( iRow, iFirstName, "Jane" ) ;
    pQuery->SetData( iRow, iLastName, "Doe" ) ;
```

CCFXRequest::AttributeExists

シンタックス

```
BOOL CCFXRequest::AttributeExists(LPCSTR lpszName)
```

説明

タグにパラメータが渡されたかどうかをチェックします。パラメータが利用可能な場合は true を返し、それ以外の場合は false を返します。

パラメータ

パラメータ	説明
lpszName	チェックするパラメータの名前です (大文字と小文字は区別されません)。

例

次の例では、ユーザーが DESTINATION という名前の属性をタグに渡したかどうかをチェックし、この属性が渡されていない場合は例外を生成します。

```
if ( pRequest->AttributeExists("DESTINATION")==FALSE )
    {
        pRequest->ThrowException(
            "Missing DESTINATION parameter",
            "You must pass a DESTINATION parameter in "
            "order for this tag to work correctly." ) ;
    }
```

CCFXRequest::CreateStringSet

シンタックス

```
CCFXStringSet* CCFXRequest::CreateStringSet(void)
```

説明

インスタンスを割り当てて返します。文字列セットの作成には、new 演算子を直接使用するのではなく、常にこの関数を使用する必要があることに注意してください。

戻り値

CCFXStringSet クラス のオブジェクト。リクエストが完了すると、返された文字列セットに割り当てられていたメモリが自動的に解放されます。

例

次の例では、文字列セットを作成し、それに 3 つの文字列を追加しています。

```
CCFXStringSet* pColors = pRequest->CreateStringSet() ;
    pColors->AddString( "Red" ) ;
    pColors->AddString( "Green" ) ;
    pColors->AddString( "Blue" ) ;
```

CCFXRequest::Debug

シンタックス

```
BOOL CCFXRequest::Debug(void)
```

説明

タグに Debug 属性があるかどうかをチェックします。この関数を使用してリクエストにデバッグ情報を書き込むかどうかを判断します。詳細については、CCFXRequest::WriteDebug を参照してください。

戻り値

タグに Debug 属性がある場合は true、それ以外の場合は false

例

次の例では、Debug 属性があるかどうかをチェックし、存在する場合は簡単なデバッグメッセージを出力します。

```
if ( pRequest->Debug() )
    {
        pRequest->WriteDebug( "Top secret debug info" ) ;
    }
```

CCFXRequest::GetAttribute

シンタックス

```
LPCSTR CCFXRequest::GetAttribute(LPCSTR lpszName)
```

説明

渡された属性の値を取得します。属性がない場合は、空の文字列を返します (タグに属性が渡されたかどうかをテストするには、CCFXRequest::AttributeExists を使います)。

戻り値

タグに渡された属性の値を返します。その名前の属性がタグに渡されていない場合は、空の文字列が返されます。

パラメータ

パラメータ	説明
lpszName	取得する属性の名前です (大文字と小文字は区別しません)。

例

次の例では、DESTINATION という名前の属性を取得して、その値をユーザーに返しています。

```
LPCSTR lpszDestination = pRequest->GetAttribute("DESTINATION") ;
    pRequest->Write( "The destination is: " ) ;
    pRequest->Write( lpszDestination ) ;
```

CCFXRequest::GetAttributeList

シンタックス

```
CCFXStringSet* CCFXRequest::GetAttributeList(void)
```

説明
タグに渡された属性名の配列を取得します。1 つの属性の値を取得するには、CCFXRequest::GetAttribute を使用します。

戻り値
タグに渡された属性のリストを持つ、CCFXStringSet クラス クラスのオブジェクト。リクエストが完了すると、返された文字列セットに割り当てられていたメモリが自動的に解放されます。

例
次の例では、属性のリストを取得し、リストから繰り返し各属性とその値をユーザーに返します。

```
LPCSTR lpszName, lpszValue ;
    CCFXStringSet* pAttribs = pRequest->GetAttributeList() ;
    int nNumAttribs = pAttribs->GetCount() ;

    for( int i=1; i<=nNumAttribs; i++ )
    {
        lpszName = pAttribs->GetString( i ) ;
        lpszValue = pRequest->GetAttribute( lpszName ) ;
        pRequest->Write( lpszName ) ;
        pRequest->Write( " = " ) ;
        pRequest->Write( lpszValue ) ;
        pRequest->Write( "<BR>" ) ;
    }
```

CCFXRequest::GetCustomData

シンタックス
```
LPVOID CCFXRequest::GetCustomData(void)
```

説明
リクエストに対するタグ固有のカスタムデータを取得します。通常、実装したタグのサブルーチン内からこのメソッドを使用すると、リクエストの中からタグ固有のデータを抽出できます。

戻り値
カスタムデータを指すポインタ、またはリクエストの際に CCFXRequest::SetCustomData を使ってカスタムデータが設定されていない場合は NULL

例
次の例では、MYTAGDATA という、リクエストに固有のデータ構造体タイプがあると想定し、それを指すポインタを取得します。

```
void DoSomeGruntWork( CCFXRequest* pRequest )
    {
        MYTAGDATA* pTagData =
            (MYTAGDATA*)pRequest->GetCustomData() ;

        ... remainder of procedure ...
    }
```

CCFXRequest::GetQuery

シンタックス

```
CCFXQuery* CCFXRequest::GetQuery(void)
```

説明

タグに渡されたクエリーを取得します。カスタムタグにクエリーを渡すには、QUERY 属性を使用します。この属性を、cfquery タグまたは別のカスタムタグを使用して作成されたクエリーの名前に設定します。QUERY 属性はオプションであり、既存のデータセットを処理するタグのみで使用する必要があります。

戻り値

タグに渡されたクエリーを表す、CCFXQuery クラス のオブジェクト。タグにクエリーが渡されていない場合は NULL が返されます。リクエストが完了すると、返されるクエリーに割り当てられていたメモリが自動的に解放されます。

例

次の例では、タグに渡されたクエリーを取得します。クエリーが渡されていない場合は例外が発生します。

```
CCFXQuery* pQuery = pRequest->GetQuery() ;
    if ( pQuery == NULL )
    {
        pRequest->ThrowException(
                "Missing QUERY parameter",
                "You must pass a QUERY parameter in "
                "order for this tag to work correctly." ) ;
    }
```

CCFXRequest::ReThrowException

シンタックス

```
void CCFXRequest::ReThrowException(CCFXException* e)
```

説明

拡張プロシージャで検出した例外を再び返します。これは、DLL 拡張コードによって返された C++ 例外が ColdFusion まで伝播しないようにする関数です。拡張コードで発生するすべての C++ エラーを検出して再び返すか (CCFXException クラス の場合)、または CCFXRequest::ThrowException を使って新しい例外ポインタを生成して返します。

パラメータ

パラメータ	説明
e	検出された CCFXException エラーです。

例

次のコードは、ColdFusion Extension DLL プロシージャでの正しい例外処理方法を示しています。

```
try
    {
        ...Code that could throw an exception...
    }
    catch( CCFXException* e )
    {
        ...Do appropriate resource cleanup here...
        // Re-throw the exception
        pRequest->ReThrowException( e ) ;
    }
    catch( ... )
    {
        // Something nasty happened

        pRequest->ThrowException(
            "Unexpected error occurred in CFX tag", "" ) ;
    }
```

CCFXRequest::SetCustomData

シンタックス

```
void CCFXRequest::SetCustomData(LPVOID lpvData)
```

説明

リクエストとともに渡すタグ固有のカスタムデータを設定します。この関数は、カスタムタグ実装内のプロシージャに渡すリクエスト固有のデータを保管するために使用します。

パラメータ

パラメータ	説明
lpvData	カスタムデータを指すポインタです。

例

次の例では、MYTAGDATA という仮想タイプのリクエスト固有のデータ構造体を作成し、そのリクエスト内の構造体を指すポインタを後の使用のために保管しています。

```
void ProcessTagRequest( CCFXRequest* pRequest )
    try
    {
        MYTAGDATA tagData ;
        pRequest->SetCustomData( (LPVOID)&tagData ) ;

        ... remainder of procedure ...
    }
```

CCFXRequest::SetVariable

シンタックス

```
void CCFXRequest::SetVariable(LPCSTR lpszName, LPCSTR lpszValue)
```

説明

呼び出しテンプレートに変数を設定します。指定した変数名がテンプレートに既に存在している場合、その値は置き換えられます。既存の変数がない場合は、新しい変数が作成されます。SetVariable を使用して作成された変数の値は、他のテンプレート変数 (#MessageSent# など) と同じ方法でアクセスできます。

パラメータ

パラメータ	説明
lpszName	変数の名前です。
lpszValue	変数の値です。

例

次の例では、カスタムタグによって実行された操作が成功したかどうかに基づいて、MessageSent という名前の変数値が設定されます。

```
BOOL bMessageSent;
    ...attempt to send the message...
    if ( bMessageSent == TRUE )
    {
        pRequest->SetVariable( "MessageSent", "Yes" ) ;
    }
    else
    {
        pRequest->SetVariable( "MessageSent", "No" ) ;
    }
```

CCFXRequest::ThrowException

シンタックス

```
void CCFXRequest::ThrowException(LPCSTR lpszError, LPCSTR lpszDiagnostics)
```

説明

例外を返し、リクエストの処理を終了します。リクエスト処理を継続できないエラーが発生した場合は、この関数を呼び出します。この関数は、ほとんどの場合、拡張コードのリソースリークを防ぐために、CCFXRequest::ReThrowException と組み合わせて使用されます。

パラメータ

パラメータ	説明
lpszError	短いエラー識別子です。
lpszDiagnostics	エラー診断情報です。

例

次の例では、リクエスト処理中に、予測せぬエラーが発生したことを示す例外を返しています。

```
char buffError[512] ;
    wsprintf( buffError,
            "Unexpected Windows NT error number %ld "
            "occurred while processing request.", GetLastError() ) ;

    pRequest->ThrowException( "Error occurred", buffError ) ;
```

CCFXRequest::Write

シンタックス

```
void CCFXRequest::Write(LPCSTR lpszOutput)
```

説明

テキスト出力をユーザーに返します。

パラメータ

パラメータ	説明
lpszOutput	出力するテキストです。

例

次の例では、出力文字列を保持するバッファーを作成し、バッファーにデータを挿入し、ユーザーにその出力を返します。

```
CHAR buffOutput[1024] ;
    wsprintf( buffOutput, "The destination is: %s",
            pRequest->GetAttribute("DESTINATION") ) ;
    pRequest->Write( buffOutput ) ;
```

CCFXRequest::WriteDebug

シンタックス

```
void CCFXRequest::WriteDebug(LPCSTR lpszOutput)
```

説明

デバッグストリームにテキスト出力を書き込みます。このテキストは、タグに Debug 属性がある場合にのみ、エンドユーザーに対して表示されます。詳細については、CCFXRequest::Debug を参照してください。

パラメータ

パラメータ	説明
lpszOutput	出力するテキストです。

例

次の例では、Debug 属性があるかどうかをチェックし、存在する場合は簡単なデバッグメッセージを出力します。

```
if ( pRequest->Debug() )
    {
        pRequest->WriteDebug( "Top secret debug info" ) ;
    }
```

CCFXStringSet クラス

順番に並んだ文字列セットを表す抽象クラス。文字列セットに文字列を追加するとき、または文字列セットから文字列を取得するときには、数値インデックスを使用します (文字列のインデックス値は 1 から始まります)。文字列セットを作成するには、CCFXRequest::CreateStringSet を使用します。

クラスメソッド

virtual int AddString(LPCSTR lpszString)	CCFXStringSet::AddString はリストの最後に文字列を追加します。
virtual int GetCount()	CCFXStringSet::GetCount はリストに含まれている文字列の数を取得します。
virtual LPCSTR GetString(int iIndex)	CCFXStringSet::GetString は渡されたインデックスに位置する文字列を取得します。
virtual int GetIndexForString(LPCSTR lpszString)	CCFXStringSet::GetIndexForString は渡された文字列のインデックスを取得します。

CCFXStringSet::AddString

シンタックス

```
int CCFXStringSet::AddString(LPCSTR lpszString)
```

説明

リストの最後に文字列を追加します。

戻り値

追加された文字列のインデックスを返します。

パラメータ

パラメータ	説明
lpszString	リストに追加する文字列です。

例

次の例では、3 行の文字列を文字列セットに追加し、追加されたその項目のインデックスを保存します。

```
CCFXStringSet* pSet = pRequest->CreateStringSet() ;
    int iRed = pSet->AddString( "Red" ) ;
    int iGreen = pSet->AddString( "Green" ) ;
    int iBlue = pSet->AddString( "Blue" ) ;
```

CCFXStringSet::GetCount

シンタックス

```
int CCFXStringSet::GetCount(void)
```

説明

文字列セットに含まれている文字列の数を取得します。この値を CCFXStringSet::GetString とともに使用すると、セット内の文字列を巡回できます (リスト内の文字列のインデックス値は 1 から始まります)。

戻り値

文字列セットに含まれている文字列の数

例

次の例では、GetCount とともに CCFXStringSet::GetString を使用して文字列セットを巡回し、リストの内容をユーザーに返します。

```
int nNumItems = pStringSet->GetCount() ;
    for ( int i=1; i<=nNumItems; i++ )
    {
        pRequest->Write( pStringSet->GetString( i ) ) ;
        pRequest->Write( "<BR>" ) ;
    }
```

CCFXStringSet::GetIndexForString

シンタックス

```
int CCFXStringSet::GetIndexForString(LPCSTR lpszString)
```

説明

渡された文字列について検索を行います。この検索では大文字と小文字は区別されません。

戻り値

文字列が検出された場合、その文字列セット内のインデックス。見つからない場合は、定数 CFX_STRING_NOT_FOUND が返されます。

パラメータ

パラメータ	説明
lpszString	検索する文字列です。

例

次の例では、文字列を検索し、見つからない場合は例外を返します。

```
CCFXStringSet* pAttribs = pRequest->GetAttributeList() ;

    int iDestination = pAttribs->GetIndexForString("DESTINATION") ;
    if ( iDestination == CFX_STRING_NOT_FOUND )
    {
        pRequest->ThrowException(
            "DESTINATION attribute not found."
            "The DESTINATION attribute is required "
            "by this tag." ) ;
    }
```

CCFXStringSet::GetString

シンタックス

```
LPCSTR CCFXStringSet::GetString(int iIndex)
```

説明

渡されたインデックスにある文字列を取得します。インデックス値は 1 から始まります。

戻り値

渡されたインデックスにある文字列

パラメータ

パラメータ	説明
iIndex	取得する文字列のインデックスです。

例

次の例では、GetString とともに CCFXStringSet::GetCount を使用して、文字列セットを巡回し、リストの内容をユーザーに返しています。

```
int nNumItems = pStringSet->GetCount() ;
    for ( int i=1; i<=nNumItems; i++ )
    {
        pRequest->Write( pStringSet->GetString( i ) ) ;
        pRequest->Write( "<BR>" ) ;
    }
```

第 11 章：ColdFusion Java CFX リファレンス

ColdFusion には、Java で ColdFusion カスタム CFX を構築するための Java インターフェイスが用意されています。

クラスライブラリの概要

Java による ColdFusion カスタム CFX の構築では、次の Java インターフェイスを使用できます。

インターフェイス	メソッド
カスタムタグインターフェイス	processRequest
クエリーインターフェイス	addRow getColumnIndex getColumns getData getName getRowCount setData
リクエストインターフェイス	attributeExists debug getAttribute getAttributeList getIntAttribute getQuery getSetting
レスポンスインターフェイス	addQuery setVariable write writeDebug

カスタムタグインターフェイス

```
public abstract interface CustomTag
```

カスタムタグを実装するためのインターフェイスです。

このインターフェイスを実装するクラスは、Java CFX タグの CLASS 属性で指定できます。たとえば、このインターフェイスを実装する **MyCustomTag** というクラスでは、次の CFML コードを使用すると MyCustomTag.processRequest メソッドが呼び出されます。

```
<CFX_MyCustomTag>
```

Java CFX タグに他の属性を渡すこともできます。これらの属性の値は、processRequest メソッドにリクエストオブジェクトを渡すことで利用できます。

メソッド

戻り値	シンタックス	説明
void	processRequest(Request request, Response response)	CFX_mycustomtag タグから発生したリクエストを処理します。

processRequest

説明
Java CFX タグから発生したリクエストを処理します。

カテゴリ
カスタムタグインターフェイス

シンタックス
```
public void processRequest(Request request, Response response)
```

戻り値
Exception　リクエスト処理時に予期せぬエラーが発生した場合に返されます。

パラメータ

パラメータ	説明
request	このリクエストのパラメータです (属性、クエリーなど)。
response	リクエストに対するレスポンスを生成するインターフェイスです (出力、変数、クエリーなど)。

クエリーインターフェイス

```
public abstract interface Query
```

カスタムタグによって使用または作成されるクエリーへのインターフェイスです。クエリーには、名前の付いた列および行で構成されるテーブル式のデータが含まれています。

メソッド

戻り値	メソッド	説明
int	addRow()	クエリーに行を追加します。
int	getColumnIndex(String name)	その名前の列のインデックスを取得します。
String[]	getColumns()	クエリーに含まれている列名のリストを取得します。
文字列	getData(int iRow, int iCol)	クエリーの行と列からデータ要素を取得します。
文字列	getName()	クエリーの名前を取得します。
int	getRowCount()	クエリーの行数を取得します。
void	setData(int iRow, int iCol, String data)	クエリーの行と列に含まれるデータ要素を設定します。

addRow

説明
クエリーに行を追加します。クエリーに行を追加する場合は、このメソッドを呼び出します。

クエリーに追加された行のインデックスを返します。

カテゴリ
クエリーインターフェイス

シンタックス
```
public int addRow()
```

関連項目
setData、getData

例
次の例では、3 つの列 (**City**、**State**、**Zip**) を持つクエリーに 2 つの行を追加します。

```
// Define column indexes
int iCity = 1, iState = 2, iZip = 3 ;

// First row
int iRow = query.addRow() ;
query.setData( iRow, iCity, "Minneapolis" ) ;
query.setData( iRow, iState, "MN" ) ;
query.setData( iRow, iZip, "55345" ) ;
// Second row
iRow = query.addRow() ;
query.setData( iRow, iCity, "St. Paul" ) ;
query.setData( iRow, iState, "MN" ) ;
query.setData( iRow, iZip, "55105" ) ;
```

getColumnIndex

説明
その列のインデックスを返します。該当する列がない場合には 0 を返します。

カテゴリ
クエリーインターフェイス

シンタックス
```
public int getColumnIndex(String name)
```

関連項目
getColumns、getData

パラメータ

パラメータ	説明
name	インデックスを取得する列の名前です (ルックアップ関数では大文字と小文字を区別しません)。

例

次の例は、EMAIL 列のインデックスを取得し、このインデックスを使用して、EMAIL 列に格納されている住所のリストを出力します。

```
// Get the index of the EMAIL column
int iEMail = query.getColumnIndex( "EMAIL" ) ;

// Iterate over the query and output list of addresses
int nRows = query.getRowCount() ;
for( int iRow = 1; iRow <= nRows; iRow++ )
{
    response.write( query.getData( iRow, iEMail ) + "<BR>" ) ;
}
```

getColumns

説明

クエリーの列の名前を含んでいる文字列の配列を返します。

カテゴリ

クエリーインターフェイス

シンタックス

```
public String[] getColumns()
```

例

次の例では、列の配列を取得し、このリストを巡回して、各列の名前をユーザーに出力します。

```
// Get the list of columns from the query
String[] columns = query.getColumns() ;
int nNumColumns = columns.length ;

// Print the list of columns to the user
response.write( "Columns in query: " ) ;
for( int i=0; i<nNumColumns; i++ )
{
    response.write( columns[i] + " " ) ;
}
```

getData

説明

クエリーの行と列からデータ要素を取得します。行と列のインデックスは 1 から始まります。getRowCount を呼び出すと、クエリーの行数を調べることができます。getColumns を呼び出すと、クエリーの列数を調べることができます。

リクエストされたデータ要素の値を返します。

カテゴリ

クエリーインターフェイス

シンタックス

```
public String getData(int iRow, int iCol)
```

戻り値
IndexOutOfBoundsException。無効なインデックスがメソッドに渡された場合に返されます。

関連項目
setData、addRow

パラメータ

パラメータ	説明
iRow	データを取得する行です (1 から始まります)。
iCol	データを取得する列です (1 から始まります)。

例
次の例では、クエリーの行を巡回して、簡単なスペース区切り形式でデータをユーザーに出力します。

```
int iRow, iCol ;
int nNumCols = query.getColumns().length ;
int nNumRows = query.getRowCount() ;
for ( iRow = 1; iRow <= nNumRows; iRow++ )
{
    for ( iCol = 1; iCol <= nNumCols; iCol++ )
    {
        response.write( query.getData( iRow, iCol ) + " " ) ;
    }
    response.write( "<BR>" ) ;
}
```

getName

説明
クエリーの名前を返します。

カテゴリ
クエリーインターフェイス

シンタックス
```
public String getName()
```

例
次の例では、クエリーの名前を取得して、ユーザーに出力します。

```
Query query = request.getQuery() ;
    response.write( "The query name is: " + query.getName() ) ;
```

getRowCount

説明
クエリーの行数を取得します。

クエリーの行数を返します。

カテゴリ
クエリーインターフェイス

シンタックス
public int getRowCount()

例
次の例では、クエリーの行数を取得して、ユーザーに出力します。

```
Query query = request.getQuery() ;
    int rows = query.getRowCount() ;
    response.write( "The number of rows in the query is "
    + Integer.ToString(rows) ) ;
```

setData

説明
クエリーの行と列に含まれるデータ要素を設定します。行と列のインデックスは、1 から始まります。ある行について、setData を呼び出す場合は、その前に addRow を呼び出し、その戻り値を setData を呼び出すための行インデックスとして使用します。

カテゴリ
クエリーインターフェイス

シンタックス
```
public void setData(int iRow, int iCol, String data)
```

戻り値
IndexOutOfBoundsException。無効なインデックスがメソッドに渡された場合に返されます。

関連項目
getData、addRow

パラメータ

パラメータ	説明
iRow	設定するデータ要素の行です (1 から始まります)。
iCol	設定するデータ要素の列です (1 から始まります)。
data	データ要素の新しい値です。

例
次の例では、3 つの列 (**City**、**State**、**Zip**) を持つクエリーに 2 つの行を追加します。

```
// Define column indexes
int iCity = 1, iState = 2, iZip = 3 ;

// First row
int iRow = query.addRow() ;
query.setData( iRow, iCity, "Minneapolis" ) ;
query.setData( iRow, iState, "MN" ) ;
query.setData( iRow, iZip, "55345" ) ;


// Second row
iRow = query.addRow() ;
query.setData( iRow, iCity, "St. Paul" ) ;
query.setData( iRow, iState, "MN" ) ;
query.setData( iRow, iZip, "55105" ) ;
```

リクエストインターフェイス

```
public abstract interface Request
```

CustomTag に対して生成されるリクエストへのインターフェイスです。このインターフェイスには、クエリーを含め、タグに渡される属性を取得するメソッドや、グローバルタグ設定値を読み込むメソッドが含まれています。

メソッド

戻り値	シンタックス	説明
boolean	attributeExists(String name)	このタグに属性が渡されたかどうかをチェックします。
boolean	debug()	タグに debug 属性があるかどうかをチェックします。
文字列	getAttribute(String name)	渡された属性の値を取得します。
String[]	getAttributeList()	タグに渡された属性のリストを取得します。
int	getIntAttribute(String name)	渡された属性の整数値を取得します。
int	getIntAttribute(String name, int def)	渡された属性の整数値を取得します。属性がないか、または属性が無効な場合は、デフォルト値を返します。
Query	getQuery()	タグに渡されたクエリーを取得します。

attributeExists

説明

このタグに属性が渡されたかどうかをチェックします。

属性が利用可能な場合は true を返し、それ以外の場合には false を返します。

カテゴリ

リクエストインターフェイス

シンタックス

```
public boolean attributeExists(String name)
```

関連項目

getAttribute、getAttributeList

パラメータ

パラメータ	説明
name	チェックする属性の名前です (大文字と小文字は区別されません)。

例

次の例では、ユーザーが DESTINATION という名前の属性をタグに渡したかどうかをチェックします。属性が渡されていない場合は、例外を返します。

```
if ( ! request.attributeExists("DESTINATION") )
{
    throw new Exception(
    "Missing DESTINATION parameter",
    "You must pass a DESTINATION parameter in "
    "order for this tag to work correctly." ) ;
} ;
```

debug

説明

タグに debug 属性があるかどうかをチェックします。このメソッドを使用して、このリクエストのデバッグ情報を書き込むかどうかを判断します。詳細については、writeDebug を参照してください。

タグに debug 属性がある場合は true、それ以外の場合は false を返します。

カテゴリ

リクエストインターフェイス

シンタックス

```
public boolean debug()
```

関連項目

writeDebug

例

次の例では、debug 属性があるかどうかをチェックし、存在する場合は簡単なデバッグメッセージを出力します。

```
if ( request.debug() )
{
    response.writeDebug( "debug info" ) ;
}
```

getAttribute

説明

渡された属性の値を取得します。属性が存在しない場合は、空の文字列を返します。属性がタグに渡されたかどうかをテストするには、attributeExists を使用します。空の文字列ではなくデフォルト値を返すには、getAttribute(String,String) を使用します。

タグに渡された属性の値を返します。その名前の属性がタグに渡されていない場合は、空の文字列が返されます。

カテゴリ
リクエストインターフェイス

シンタックス
```
public String getAttribute(String name)
```

関連項目
attributeExists、getAttributeList、getIntAttribute

パラメータ

パラメータ	説明
name	取得する属性です (大文字と小文字は区別されません)。

例
次の例では、DESTINATION という名前の属性を取得して、その値をユーザーに返しています。

```
String strDestination = request.getAttribute("DESTINATION") ;
response.write( "The destination is: " + strDestination ) ;
```

getAttributeList

説明
タグに渡された属性のリストを取得します。1 つの属性の値を取得するには、getAttribute メソッドを使用する必要があります。

タグに渡された属性の名前を含んでいる文字列の配列を返します。

カテゴリ
リクエストインターフェイス

シンタックス
```
public String[] getAttributeList()
```

関連項目
attributeExists

例
次の例では、属性のリストを取得し、リストについて繰り返し作業し、各属性とその値をユーザーに出力します。

```
String[] attribs = request.getAttributeList() ;
int nNumAttribs = attribs.length ;

for( int i = 0; i < nNumAttribs; i++ )
{
    String strName = attribs[i] ;
    String strValue = request.getAttribute( strName ) ;
    response.write( strName + "=" + strValue + "<BR>" ) ;
}
```

getIntAttribute

説明
渡された属性の整数値を取得します。属性が存在しない場合は -1 を返します。属性がタグに渡されたかどうかをテストするには、attributeExists を使用します。例外や -1 ではなくデフォルト値を返すには、getIntAttribute(String,int) を使用します。

タグに渡された属性の値を返します。その名前の属性がタグに渡されていない場合は、-1 が返されます。

カテゴリ
リクエストインターフェイス

シンタックス
```
public int getIntAttribute(String name)
```

戻り値
NumberFormatException。属性が有効な数値でない場合に返されます。

関連項目
attributeExists、getAttributeList

パラメータ

パラメータ	説明
name	取得する属性です (大文字と小文字は区別されません)。

例
次の例では、PORT という名前の属性を取得して、その値をユーザーに返しています。

```
int nPort = request.getIntAttribute("PORT") ;
if ( nPort != -1 )
    response.write( "The port is: " + String.valueOf(nPort) ) ;
```

getQuery

説明
タグに渡されたクエリーを取得します。

カスタムタグにクエリーを渡すには、query 属性を使用します。この値は、(cfquery タグを使用して作成された) クエリーの名前に設定する必要があります。query 属性はオプションであり、既存のデータセットを処理するタグでのみ使用します。

タグに渡されたクエリーが返されます。タグにクエリーが渡されていない場合は null が返されます。

カテゴリ
リクエストインターフェイス

シンタックス
```
public Query getQuery()
```

例
次の例では、タグに渡されたクエリーを取得します。クエリーが渡されていない場合は例外が発生します。

```
Query query = request.getQuery() ;
if ( query == null )
{
    throw new Exception(
    "Missing QUERY parameter. " +
    "You must pass a QUERY parameter in "
    "order for this tag to work correctly." ) ;
}
```

getSetting

説明
グローバルカスタムタグ設定の値を取得します。カスタムタグ設定は、ColdFusion レジストリキーの CustomTags セクションに保管されています。

カスタムタグ設定の値を返します。その名前の設定値が存在しない場合、空の文字列が返されます。

カテゴリ
リクエストインターフェイス

シンタックス
public String getSetting(String **name**)

パラメータ

パラメータ	説明
name	取得する設定の名前です (大文字と小文字は区別されません)。

使用方法
Java で実装されたすべてのカスタムタグは、設定値を保管するためのレジストリキーを共有します。名前の重複を避けるため、設定値の名前の前にはカスタムタグクラスの名前を付けます。たとえば、次のコードでは、**MyCustomTag** という名前のカスタムタグクラスの **VerifyAddress** という設定の値を取得します。

```
String strVerify = request.getSetting("MyCustomTag.VerifyAddress") ;
if ( Boolean.valueOf(strVerify) )
{
    // Do address verification...
}
```

レスポンスインターフェイス

```
public abstract interface Response
```

カスタムタグから生成されたレスポンスへのインターフェイスです。このインターフェイスには、出力を記述し、クエリーを生成し、呼び出しページに変数を設定するメソッドが含まれています。

メソッド

戻り値	シンタックス	説明
Query	addQuery(String name, String[] columns)	呼び出しテンプレートにクエリーを追加します。
void	setVariable(String name, String value)	呼び出しテンプレートに変数を設定します。
void	write(String output)	ユーザーに対してテキストを出力します。
void	writeDebug(String output)	デバッグストリームにテキスト出力を書き込みます。

addQuery

説明
呼び出しテンプレートにクエリーを追加します。このテンプレートの CFML タグでクエリーにアクセスできます。addQuery を呼び出した後は、クエリーが空 (0 行) になります。クエリーにデータを挿入するには、Query メソッド addRow および setData を呼び出します。

テンプレートに追加されたクエリーを返します。

カテゴリ
レスポンスインターフェイス

シンタックス
```
public Query addQuery(String name, String[] columns)
```

戻り値
IllegalArgumentException　name パラメータが無効な CFML 変数名である場合に返されます。

関連項目
addRow、setData

パラメータ

パラメータ	説明
name	テンプレートに追加するクエリーの名前です。
columns	クエリーの列の名前です。

例
次の例では、呼び出しテンプレートに **People** という名前のクエリーが追加されます。クエリーは、2 列 (**FirstName** と **LastName**) と 2 行から構成されています。

```
// Create string array with column names (also track columns indexes)
String[] columns = { "FirstName", "LastName" } ;
int iFirstName = 1, iLastName = 2 ;

// Create a query which contains these columns
Query query = response.addQuery( "People", columns ) ;

// Add data to the query
int iRow = query.addRow() ;
query.setData( iRow, iFirstName, "John" ) ;
query.setData( iRow, iLastName, "Smith" ) ;
iRow = query.addRow() ;
query.setData( iRow, iFirstName, "Jane" ) ;
query.setData( iRow, iLastName, "Doe" ) ;
```

setVariable

説明

呼び出しテンプレートに変数を設定します。指定した変数名がテンプレートに既に存在している場合、その値は変更されます。指定した変数名がない場合は、新しい変数が作成されます。

カテゴリ

レスポンスインターフェイス

シンタックス

```
public void setVariable(String name, String value)
```

戻り値

IllegalArgumentException name パラメータが無効な CFML 変数名である場合に返されます。

パラメータ

パラメータ	説明
name	設定する変数の名前です。
value	変数に設定する値です。

例

次の例では、カスタムタグによって実行されたオペレーションの結果に基づいて、**MessageSent** という名前の変数値を設定します。

```
boolean bMessageSent ;

    ...attempt to send the message...

    if ( bMessageSent == true )
    {
        response.setVariable( "MessageSent", "Yes" ) ;
    }
    else
    {
        response.setVariable( "MessageSent", "No" ) ;
    }
```

write

説明
ユーザーに対してテキストを出力します。

カテゴリ
レスポンスインターフェイス

シンタックス
```
public void write(String output)
```

パラメータ

パラメータ	説明
output	出力するテキストです。

例
次の例は、DESTINATION 属性の値を出力します。

```
response.write( "DESTINATION = " +
    request.getAttribute("DESTINATION") ) ;
```

writeDebug

説明
デバッグストリームにテキスト出力を書き込みます。このテキストは、タグに debug 属性が含まれている場合のみエンドユーザーに表示されます。Request.debug メソッドを使用すると、この属性が含まれているかどうかをチェックすることができます。

カテゴリ
レスポンスインターフェイス

シンタックス
```
public void writeDebug(String output)
```

関連項目
debug

パラメータ

パラメータ	説明
output	出力するテキストです。

例
次の例では、debug 属性があるかどうかをチェックし、存在する場合は簡単なデバッグメッセージを出力します。

```
if ( request.debug() )
{
    response.writeDebug( "debug info" ) ;
}
```

デバッグクラスリファレンス

DebugRequest クラス、DebugResponse クラス、および DebugQuery クラスによってサポートされているコンストラクタとメソッドは次のとおりです。これらのクラスは、Request インターフェイス、Response インターフェイス、および Query インターフェイス以外のメソッドもサポートしています。

DebugRequest

```
// initialize a debug request with attributes
public DebugRequest( Hashtable attributes ) ;

// initialize a debug request with attributes and a query
public DebugRequest( Hashtable attributes, Query query ) ;

// initialize a debug request with attributes, a query, and settings
public DebugRequest( Hashtable attributes, Query query, Hashtable settings ) ;
```

DebugResponse

```
// initialize a debug response
public DebugResponse() ;

// print the results of processing
public void printResults() ;
```

DebugQuery

```
// initialize a query with name and columns
public DebugQuery( String name, String[] columns )
    throws IllegalArgumentException ;

// initialize a query with name, columns, and data
public DebugQuery( String name, String[] columns, String[][] data )
    throws IllegalArgumentException ;
```

第 12 章：WDDX JavaScript オブジェクト

ColdFusion アプリケーションでは、WDDX とともに JavaScript のオブジェクトと関数を使用します。

JavaScript オブジェクトの概要

JavaScript のオブジェクトと関数を次に示します。

クラス	関数
WddxSerializer オブジェクト	serialize serializeVariable serializeValue write
WddxRecordset オブジェクト	addColumn addRows getField getRowCount setField wddxSerialize

WDDX の JavaScript オブジェクトは "wddx.js" ファイル内で定義されています。このファイルは、CFIDE/scripts ディレクトリにインストールされています。

これらのオブジェクトを使用するには、オブジェクトを参照するコードの前に JavaScript タグを配置する必要があります。たとえば次のようになります。

```
<script type="text/javascript" src="/CFIDE/scripts/wddx.js"></script>
```

WddxSerializer オブジェクト

WddxSerializer オブジェクトには、JavaScript データ構造体をシリアル化する関数が含まれています。このオブジェクトの使用方法の詳細については、『ColdFusion アプリケーションの開発』の Using WDDX を参照してください。

関数

開発者が通常呼び出す関数は serialize だけです。

関数のシンタックス	説明
object.serialize(rootobj)	渡された WddxRecordset インスタンスの WDDX パケットを作成します。

関数のシンタックス	説明
object.serializeVariable(name, obj)	構造体のプロパティをシリアル化します。オブジェクトが文字列、数値、配列、ブール値、または日付でない場合、WddxSerializer はそのオブジェクトを構造体として扱います。
object.serializeValue(obj)	渡されたインスタンス内にある、条件を満たすすべてのデータを再帰的にシリアル化します。
object.write(str)	シリアル化されたデータストリームにデータを追加します。

serialize

説明
渡された WddxRecordset インスタンスの WDDX パケットを作成します。

シンタックス
```
object.serialize( rootobj )
```

パラメータ

パラメータ	説明
object	WddxSerializer オブジェクトのインスタンスの名前です。
rootobj	シリアル化する JavaScript データ構造体です。

戻り値
関数が成功した場合はシリアル化した WDDX パケットの文字列、エラーが発生した場合は null 値

使用方法
WddxRecordset インスタンス内のデータをシリアル化するために、この関数を呼び出します。

例
次の例は、WddxRecordset インスタンスをシリアル化するために呼び出す JavaScript 関数を示しています。この関数では、シリアル化したデータをフォームフィールドにコピーして表示します。

```
function serializeData(data, formField)
{
    wddxSerializer = new WddxSerializer();
    wddxPacket = wddxSerializer.serialize(data);
    if (wddxPacket != null)
    {
        formField.value = wddxPacket;
    }
    else
    {
        alert("Couldn't serialize data");
    }
}
```

serializeVariable

説明
構造体のプロパティをシリアル化します。オブジェクトが文字列、数値、配列、ブール値、または日付でない場合、WddxSerializer はそのオブジェクトを構造体として扱います。

シンタックス

```
object.serializeVariable( name, obj )
```

パラメータ

パラメータ	説明
object	WddxSerializer オブジェクトのインスタンスの名前です。
name	シリアル化するプロパティです。
obj	シリアル化する値のインスタンス名です。

戻り値

シリアル化に成功した場合は true、失敗した場合は false

これは内部関数であるため、通常は呼び出す必要はありません。

例

次の例は、WddxSerializer serializeValue 関数からの引用です。

```
...
// Some generic object; treat it as a structure
    this.write("<struct>");
    for (prop in obj)
    {
        bSuccess = this.serializeVariable(prop, obj[prop]);
        if (! bSuccess)
        {
            break;
        }
    }
    this.write("</struct>");
...
```

serializeValue

説明

渡されたインスタンス内にある、条件を満たすすべてのデータを再帰的にシリアル化します。条件を満たすデータには次のものがあります。

- 文字列
- 数値
- ブール値
- 日付
- 配列
- レコードセット
- 任意の JavaScript オブジェクト

この関数では、null 値は空の文字列としてシリアル化されます。

シンタックス

```
object.serializeValue( obj )
```

パラメータ

パラメータ	説明
object	WddxSerializer オブジェクトのインスタンスの名前です。
obj	シリアル化する WddxRecordset オブジェクトのインスタンスの名前です。

戻り値
obj のシリアル化に成功した場合は true、失敗した場合は false

使用方法
これは内部関数であるため、通常は呼び出す必要はありません。

例
次に、WddxSerializer の serialize 関数の例を示します。

```
...
this.wddxPacket = "";
this.write("<wddxPacket version='1.0'><header/><data>");
bSuccess = this.serializeValue(rootObj);
this.write("</data></wddxPacket>");
if (bSuccess)
{
    return this.wddxPacket;
}
else
{
    return null;
}
...
```

write

説明
シリアル化されたデータストリームにデータを追加します。

シンタックス
```
object.write( str )
```

パラメータ

パラメータ	説明
object	WddxSerializer オブジェクトのインスタンスの名前です。
str	シリアル化されたデータストリームにコピーする文字列です。

戻り値
更新後のシリアル化されたデータストリームの文字列

使用方法
これは内部関数であるため、通常は呼び出す必要はありません。

例

次の例は、WddxSerializer serializeValue 関数からの引用です。

```
...
else if (typeof(obj) == "number")
{
    // Number value
    this.write("<number>" + obj + "</number>");
}
else if (typeof(obj) == "boolean")
{
    // Boolean value
    this.write("<boolean value='" + obj + "'/>");
}
...
```

WddxRecordset オブジェクト

このオブジェクトには、WDDX レコードセットの構築時に必要に応じて呼び出す関数が含まれています。このオブジェクトの使用方法の詳細については、『ColdFusion アプリケーションの開発』の Using WDDX を参照してください。

関数

関数のシンタックス	説明
object.addColumn(name)	WddxRecordset インスタンスのすべての行に 1 列を追加します
object.addRows(n)	WddxRecordset インスタンスのすべての列に行を追加します
object.dump(escapeStrings)	WddxRecordset オブジェクトデータを表示します。
object.getField(row, col)	指定した行および列の位置にある要素を返します。
object.getRowCount()	WddxRecordset インスタンス内の行数を示します。
object.setField(row, col, value)	指定した行および列の位置にある要素を設定します。
object.wddxSerialize(serializer)	レコードセットをシリアル化します。

戻り値

WddxRecordset オブジェクトのデータの HTML テーブル

使用方法

レコードセットのデバッグおよびテストに便利です。ブール値のパラメータ escapeStrings によって、文字列値に含まれる <>& 文字を HTML で <>& としてエスケープするかどうかが決定されます。

例

```
<!--- Create a simple query --->
<cfquery name = "q" datasource ="cfdocexamples">
    SELECT Message_Id, Thread_id, Username, Posted
    FROM messages
</cfquery>
<!--- Load the wddx.js file, which includes the dump function --->
<script type="text/javascript" src="/CFIDE/scripts/wddx.js"></script>
<script>
// Use WDDX to move from CFML data to JS
<cfwddx action="cfml2js" input="#q#" topLevelVariable="qj">
// Dump the record set
document.write(qj.dump(true));
</script>
```

addColumn

説明

WddxRecordset インスタンスのすべての行に 1 列を追加します

シンタックス

```
object.addColumn( name )
```

パラメータ

パラメータ	説明
object	WddxRecordset オブジェクトのインスタンスの名前です。
name	追加する列の名前です。

戻り値

なし

使用方法

WDDX レコードセットのすべての行に 1 列を追加します。新しい列の値は、最初は null に設定されます。

例

次の例では、addColumn 関数を呼び出します。

```
// Create a new record set
rs = new WddxRecordset();

// Add a new column
rs.addColumn("NewColumn");

// Extend the record set by 3 rows
rs.addRows(3);

// Set an element in the first row
// newValue is a previously defined variable
rs.setField(0, "NewColumn", newValue);
```

addRows

説明
WddxRecordset インスタンスのすべての列に行を追加します

シンタックス
```
object.addRows( n )
```

パラメータ

パラメータ	説明
object	WddxRecordset オブジェクトのインスタンスの名前です。
n	追加する行の数を示す整数です。

戻り値
なし

使用方法
この関数を使用すると、指定した行数を WDDX レコードセットの各列に追加できます。行および列の値は、最初は null に設定されます。

例
次の例では、addRows 関数を呼び出します。

```
// Create a new record set
rs = new WddxRecordset();

// Add a new column
rs.addColumn("NewColumn");

// Extend the record set by 3 rows
rs.addRows(3);

// Set an element in the first row
// newValue is a previously defined variable
rs.setField(0, "NewColumn", newValue);
```

getField

説明
指定した行および列の位置にある要素を返します。

シンタックス
```
object.getField( row, col )
```

パラメータ

パラメータ	説明
object	WddxRecordset オブジェクトのインスタンスの名前です。
row	返す値の 0 から始まる行番号を指定する整数です。
col	返す値の列を指定する整数または文字列です。

戻り値
指定した行および列の位置にある値

使用方法
WDDX レコードセットの値にアクセスするために、この関数を呼び出します。

例
次の例では、getField 関数を呼び出します。ここで、変数 **r** は、WddxRecordset インスタンスへの参照です。

```
for (row = 0; row < nRows; ++row)
{
    o += "<tr>";
    for (i = 0; i < colNames.length; ++i)
    {
        o += "<td>" + r.getField(row, colNames[i]) + "</td>";
    }
    o += "</tr>";
}
```

getRowCount

説明
WddxRecordset インスタンス内の行数を示します。

シンタックス

```
object.getRowCount( )
```

パラメータ

パラメータ	説明
object	WddxRecordset オブジェクトのインスタンスの名前です。

戻り値
整数。WddxRecordset インスタンスの行数です。

使用方法
レコードセットの行数を調べるには、ループ構文の前でこの関数を呼び出します。

例
次の例では、getRowCount 関数を呼び出します。

```
function dumpWddxRecordset(r)
{
// Get row count
    nRows = r.getRowCount();
...
    for (row = 0; row < nRows; ++row)
...
```

setField

説明
指定した行および列の位置に要素を設定します。

シンタックス
```
object.setField( row, col, value )
```

パラメータ

パラメータ	説明
object	WddxRecordset オブジェクトのインスタンスの名前です。
row	設定する要素のある行を示す整数です。
col	設定する要素を含んでいる列を指定する整数または文字列です。
value	設定する値です。

戻り値
なし

使用方法
WddxRecordset インスタンスに値を設定するために、この関数を呼び出します。

例
次の例では、setField 関数を呼び出します。

```
// Create a new recordset
rs = new WddxRecordset();

// Add a new column
rs.addColumn("NewColumn");

// Extend the record set by 3 rows
rs.addRows(3);

// Set an element in the first row
// newValue is a previously defined variable
rs.setField(0, "NewColumn", newValue);
```

wddxSerialize

説明
レコードセットをシリアル化します。

シンタックス

```
object.wddxSerialize( serializer )
```

パラメータ

パラメータ	説明
object	WddxRecordset オブジェクトのインスタンスの名前です。
serializer	WddxSerializer インスタンスです。

戻り値

シリアル化に成功した場合は true、失敗した場合は false

使用方法

これは内部関数であるため、通常は呼び出す必要はありません。

例

次の例は、WddxSerializer serializeValue 関数からの引用です。

```
...
else if (typeof(obj) == "object")
{
if (obj == null)
{
// Null values become empty strings
this.write("<string></string>");
}
else if (typeof(obj.wddxSerialize) == "function")
{
// Object knows how to serialize itself
bSuccess = obj.wddxSerialize(this);
}
...
```

第 13 章：ColdFusion ActionScript 関数

ColdFusion には、特定のシンタックスやメソッドを含む CF.query と CF.http という 2 つのサーバーサイド ActionScript 関数があります。

CF.query

説明
ColdFusion データソースに対するクエリーを実行します。

戻り値
RecordSet オブジェクトを返します。

シンタックス

```
CF.query
        ({
            datasource:"data source name",
            sql:"SQL stmts",
            username:"username",
            password:"password",
            maxrows:number,
            timeout:milliseconds
        })
```

引数

引数	必須 / オプション	説明
datasource	必須	クエリーによってデータを取得するデータソースの名前です。
sql	必須	SQL ステートメントです。
username	オプション	ユーザー名です。データソース設定で指定されたユーザー名より優先されます。
password	オプション	パスワード。データソース設定で指定されたパスワードより優先されます。
maxrows	オプション	レコードセットに返される行の最大数です。
timeout	オプション	クエリーの処理がタイムアウトになったことを示すエラーが返されるまでの、クエリーの最大実行時間 (秒単位) です。名前付き引数でのみ使用できます。

使用方法
CF.query 関数のコードは、名前付き引数または定位置引数を使用して記述できます。サポートされているすべての引数は、次のように名前付き引数スタイルを使って呼び出すことができます。

```
CF.query({datasource:"datasource", sql:"sql stmt",
    username:"username", password:"password", maxrows:"maxrows",
    timeout:"timeout"});
```

注意：名前付き引数スタイルでは、関数の引数を中括弧 ({}) で囲みます。

定位置引数スタイルは略式のコーディングスタイルであり、すべての引数で使用できるわけではありません。CF.query 関数で定位置引数を使用する場合は、次のシンタックスを使用します。

```
CF.query(datasource, sql);
CF.query(datasource, sql, maxrows);
CF.query(datasource, sql, username, password);
CF.query(datasource, sql, username, password, maxrows);
```

注意：定位置引数には中括弧 ({}) を使用しないでください。

CF.query 関数によって返されたレコードセットは、RecordSet ActionScript クラス内のメソッドを使用して操作できます。RecordSet クラスで使用できる主なメソッドには次のものがあります。

- RecordSet.getColumnnames
- RecordSet.getLength
- RecordSet.getItemAt
- RecordSet.getItemID
- RecordSet.sortItemsBy
- RecordSet.getNumberAvailable
- RecordSet.filter
- RecordSet.sort

  サーバーサイド ActionScript の使用方法の詳細については、『ColdFusion アプリケーションの開発』の Using Server-Side ActionScript を参照してください。RecordSet ActionScript クラスの詳細については、『Flash Remoting の使用』を参照してください。

例

```
// Define a function to do a basic query
// Note use of positional arguments
function basicQuery()
{
    result = CF.query("myquery", "cust_data", "SELECT * from tblParks");
    return result;
}

// Example function declaration using named arguments
function basicQuery()
{
    result = CF.query({datasource:"cust_data", sql:"SELECT * from tblParks"});
    return result;
}

// Example of the CF.query function using maxrows argument
function basicQueryWithMaxRows()
{
    result = CF.query("cust_data", "SELECT * from tblParks", 25);
    return result;
}

// Example of the CF.query function with username and password
function basicQueryWithUser()
{
    result = CF.query("cust_data", "SELECT * from tblParks",
        "wsburroughs", "migraine1");
    return result;
}
```

CF.http

説明

ファイルに対して HTTP の POST および GET オペレーションを実行します。POST オペレーションでは、MIME ファイルタイプのサーバーへのアップロード、Cookie、フォームフィールド、URL、ファイル、または CGI 変数のサーバーへの直接送信ができます。

戻り値

データにアクセスするために参照するプロパティを含むオブジェクトを返します。

シンタックス

```
CF.http
    ({
            method:"get or post",
        url:"URL",
        username:"username",
        password:"password",
        resolveurl:"yes or no",
        params:arrayvar,
        path:"path",
        file:"filename"
    })
```

引数

引数	必須 / オプション	説明
method	必須	2 つの引数のいずれかを指定します。 • get: テキストファイルまたはバイナリファイルをダウンロードしたり、テキストファイルのコンテンツからクエリーを作成したりすることができます。 • post: 情報をサーバーページまたは CGI プログラムに送信します。params 引数が必要です。
url	必須	ファイルが置かれているサーバーのホスト名の完全な URL または IP アドレスです。URL には、プロトコル (http または https) とホスト名を指定する必要があります。
username	オプション	サーバーに必須の場合は、ユーザー名を指定します。
password	オプション	サーバーに必須の場合は、パスワードを指定します。

引数	必須 / オプション	説明
resolveurl	オプション	Get メソッドと Post メソッドで使用します。 • Yes または No を指定します (デフォルトは No)。 GET オペレーションおよび POST オペレーションの場合に Yes を指定すると、Filecontent プロパティに返されるページリファレンスでは、ポート番号などの内部の URL が完全に変換されます。その結果、リンクは変更されずに残ります。リンクを含む可能性のある次の HTML タグは変換されます。 - img src - a href - form action - applet code - script src - embed src - embed pluginspace - body background - frame src - bgsound src - object data - object classid - object codebase - object usemap
params	オプション	オブジェクトの配列として受け渡される HTTP パラメータです。次の種類のパラメータがサポートされます。 • name • type • value CF.http パラメータは、オブジェクトの配列として受け渡されます。POST オペレーションには params 引数が必須です。
path	オプション	ファイルを格納するディレクトリへのパスです。path 引数を使用するときには、file 引数は必須です。
file	オプション	アクセスされるファイルの名前です。GET オペレーションの場合は、url 引数で指定した名前がデフォルト値になります。path 引数にはパス情報を入力します。path 引数を使用する場合は、この引数は必須です。

使用方法

CF.http 関数のコードは、名前付き引数または定位置引数を使用して記述できます。サポートされているすべての引数は、次のように名前付き引数スタイルを使って呼び出すことができます。

```
CF.http({method:"method", url:"URL", username:"username", password:"password",
    resolveurl:"yes or no", params:arrayvar,
    path:"path", file:"filename"});
```

注意：名前付き引数スタイルでは、関数の引数を中括弧 ({}) で囲みます。

定位置引数は、略式のコーディングスタイルです。ただし、すべての引数が定位置引数スタイルに対応しているわけではありません。CF.http 関数で定位置引数を使用する場合は、次のシンタックスを使用します。

```
CF.http(url);
CF.http(method, url);
CF.http(method, url, username, password);
CF.http(method, url, params, username, password);
```

注意：定位置引数には中括弧 ({}) を使用しないでください。

次のパラメータは、CF.http 関数の params 引数内のオブジェクト配列としてだけ受け渡すことができます。

パラメータ	**説明**
name	渡すデータの変数名。
type	トランザクションタイプです。 • URL • FormField • Cookie • CGI • ファイル
value	渡す URL、フォームフィールド、Cookie、ファイル、または CGI 変数の値。

CF.http 関数は、一連のオブジェクトプロパティとしてデータを返します。詳しい説明を次の表に示します。

プロパティ	**説明**
Text	指定した URL 位置にテキストデータが含まれるかどうかを示すブール値です。
Charset	URL で指定されたドキュメントで使用されている文字セット。 通常は、HTTP サーバーによってこの情報が提供されるか、または HTTP プロトコルの Content-Type ヘッダフィールドの charset パラメータに文字セットが指定されています。たとえば、次の HTTP ヘッダは、文字エンコードが EUC-JP であることを示しています。 Content-Type: text/html; charset=EUC-JP
Header	レスポンスヘッダの生データ。たとえば、次のようになります。 HTTP/1.1 200 OK Date: Mon, 04 Mar 2002 17:27:44 GMT Server: Apache/1.3.22 (Unix) mod_perl/1.26 Set-Cookie: MM_cookie=207.22.48.162.4731015262864476; path=/; expires=Wed, 03-Mar-04 17:27:44 GMT; domain=adobe.com Connection: close Content-Type: text/html
Filecontent	テキストおよび MIME ファイルの内容。

プロパティ	説明
Mimetype	MIME タイプ。text/html、image/png、image/gif、video/mpeg、text/css、audio/basic などがあります。
responseHeader	レスポンスヘッダ。ヘッダキーが 1 つしかない場合、その値は単純なタイプとしてアクセスできます。ヘッダキーが複数ある場合、その値は responseHeader 構造の配列に格納されます。
Statuscode	HTTP エラーコードと、関連するエラー文字列。レスポンスヘッダで返される一般的な HTTP ステータスコードには次のものがあります。 400: Bad Request 401: Unauthorized 403: Forbidden 404: Not Found 405: Method Not Allowed

これらの属性には、get 関数を使用してアクセスします。

```
function basicGet()
{
url = "http://localhost:8100/";

// Invoke with just the url. This is an HTTP GET.
result = CF.http(url);
return result.get("Filecontent");
}
```

注意：サーバーサイド ActionScript の使用方法の詳細については、『ColdFusion アプリケーションの開発』の Using Server-Side ActionScript を参照してください。

例

以下の例では、CF.http 関数のさまざまな使用方法を示しています。

```
function postWithNamedArgs()
{
    // Set up the array of Post parameters.
    params = new Array();
    params[1] = {name:"arg1", type:"FormField", value:"value1"};
    params[2] = {name:"arg2", type:"URL", value:"value2"};
    params[3] = {name:"arg3", type:"CGI", value:"value3"};

    url = "http://localhost:8100/";

    path = application.getContext("/").getRealPath("/");
    file = "foo.txt";

    result = CF.http({method:"post", url:url, username:"karl", password:"salsa",
    resolveurl:true, params:params, path:path, file:file});

    if (result)
        return result.get("Statuscode");
    return null;
}

// Example of a basic HTTP GET operation
// Shows that HTTP GET is the default
function basicGet()
{
    url = "http://localhost:8100/";

    // Invoke with just the url. This is an HTTP GET.
```

```
    result = CF.http(url);
    return result.get("Filecontent");
}

// Example showing simple array created to pass params arguments
function postWithParams()
{
    // Set up the array of Post parameters. These are just like cfhttpparam tags.
    params = new Array();
    params[1] = {name:"arg2", type:"URL", value:"value2"};

    url = "http://localhost:8100/";

    // Invoke with the method, url, and params
    result = CF.http("post", url, params);
    return result.get("Filecontent");
}

// Example with username and params arguments
function postWithParamsAndUser()
{
    // Set up the array of Post parameters. These are just like cfhttpparam tags.
    params = new Array();
    params[1] = {name:"arg2", type:"URL", value:"value2"};

    url = "http://localhost:8100/";

    // Invoke with the method, url, params, username, and password
    result = CF.http("post", url, params, "karl", "salsa");
    return result.get("Filecontent");
}
```