



**Adobe**

# Guide de référence FormCalc

Juin 2008

**Adobe® LiveCycle® Designer ES**

Version 8.2

© 2008 Adobe Systems Incorporated. Tous droits réservés.

Adobe® LiveCycle® Designer ES (8.2) - Guide de référence FormCalc pour Microsoft® Windows®  
Edition 3.0, juin 2008

Si le présent guide est fourni avec un logiciel régi par un contrat d'utilisateur final, le guide, ainsi que le logiciel décrit, sont fournis sous licence et peuvent être utilisés ou copiés uniquement selon les clauses et conditions de la licence. A moins d'une autorisation expresse accordée par cette licence, aucune partie de ce guide ne peut être reproduite, stockée dans un système d'interrogation ou transmise, sous quelque forme ou par quelque moyen que ce soit (électronique, mécanique, par enregistrement ou autre) sans l'autorisation écrite préalable d'Adobe Systems Incorporated. Veuillez noter que le contenu du présent guide est protégé par la loi sur les droits d'auteur, même s'il n'est pas distribué avec un logiciel régi par un contrat de licence utilisateur.

Les informations contenues dans ce guide sont fournies à titre purement informatif ; elles sont susceptibles d'être modifiées sans préavis et ne doivent pas être interprétées comme étant un engagement de la part d'Adobe Systems Incorporated. Adobe Systems Incorporated n'accepte aucune responsabilité quant aux erreurs ou inexactitudes pouvant être contenues dans le présent guide.

Veuillez noter que les illustrations et images existantes que vous souhaitez éventuellement inclure dans votre projet sont susceptibles d'être protégées par les lois sur les droits d'auteur. L'inclusion non autorisée de tels éléments dans vos nouveaux travaux peut constituer une violation des droits du propriétaire. Veuillez vous assurer de détenir toute autorisation nécessaire auprès du détenteur des droits.

Toute référence à des noms de société, des logos de société et des noms d'utilisateur dans le matériel ou les formulaires d'exemple fournis avec cette documentation et/ou le présent logiciel n'est faite qu'à titre de démonstration et ne vise aucun organisme réel ou personne réelle.

Adobe, le logo Adobe, Acrobat, LiveCycle et Reader sont des marques ou des marques déposées d'Adobe Systems Incorporated aux Etats-Unis et/ou dans certains autres pays.

Microsoft et Windows sont des marques ou marques déposées de Microsoft Corporation aux Etats-Unis et/ou dans d'autres pays.

Toutes les autres marques citées sont la propriété de leurs détenteurs respectifs.

Ce produit contient les logiciels BSAFE et/ou TIPEM de RSA Security, Inc.

Ce produit contient des éléments logiciels développés par Apache Software Foundation (<http://www.apache.org/>).

Ce produit contient du code concédé sous licence par RSA Security.

Ce produit contient des éléments logiciels développés par JDOM Project (<http://www.jdom.org/>).

La technologie vidéo Macromedia Flash 8 est soutenue par la technologie vidéo On2 TrueMotion. © 1992-2005 On2 Technologies, Inc. Tous droits réservés. <http://www.on2.com>.

Ce produit contient des éléments logiciels développés par OpenSymphony Group (<http://www.opensymphony.com/>).

Certains composants du code proviennent de Nellymoser ([www.nellymoser.com](http://www.nellymoser.com)).

Ce produit inclut une technologie de compression audio MPEG Layer-3 concédée sous licence par Fraunhofer IIS et THOMSON multimedia (<http://www.iis.fhg.de/amm/>).

Ce produit contient des éléments logiciels développés par L2FProd.com (<http://www.L2FProd.com/>).

La bibliothèque JBoss est concédée sous licence par GNU Library General Public License. Une copie est incluse à votre logiciel.

La bibliothèque BeanShell est concédée sous licence par GNU Library General Public License. Une copie est incluse à votre logiciel.

Ce produit contient des éléments logiciels développés par The Werken Company.

Ce produit contient des éléments logiciels développés par IronSmith Project (<http://www.ironsmith.org/>).

La bibliothèque OpenOffice est concédée sous licence par GNU Library General Public License. Une copie est incluse à votre logiciel.

Adobe Systems Incorporated, 345 Park Avenue, San Jose, California 95110, USA.

Avis aux utilisateurs du gouvernement des Etats-Unis. Le logiciel et la documentation sont des « articles commerciaux », conformément à la définition du terme « Commercial Items » dans l'article 48 C.F.R. §2.101 du Code de la réglementation fédérale (Code Of Federal Regulations), qui consistent en du « logiciel informatique commercial » et de la « documentation logicielle commerciale », tels que mentionnés dans les articles 48 C.F.R. §12.212 et 48 C.F.R. §227.7202. Conformément aux articles 48 C.F.R. §12.212 et 48 C.F.R. §§227.7202-1 à 227.7202-4, le logiciel commercial et la documentation logicielle commerciale sont fournis sous licence aux utilisateurs du gouvernement des Etats-Unis (a) uniquement à titre d'articles commerciaux (b) et leur confèrent seulement les droits octroyés à tous les autres utilisateurs selon les conditions mentionnées aux présentes. Droits non publiés réservés dans le cadre des lois sur les droits d'auteur en vigueur aux Etats-Unis. Adobe Systems Incorporated, 345 Park Avenue, San Jose, CA 95110-2704, USA. A l'attention des utilisateurs finaux du gouvernement des Etats-Unis, Adobe s'engage à respecter toutes les lois sur l'égalité des chances, y compris, si approprié, les dispositions de l'Executive Order 11246, comme modifié, la section 402 de l'Acte d'assistance à la réhabilitation des vétérans du Vietnam (Vietnam Era Veterans Readjustment Assistance Act) de 1974 (38 USC 4212) et la section 503 de l'Acte de réhabilitation (Rehabilitation Act) de 1973, comme modifié, ainsi que les règlements de l'article 41 C.F.R., sections 60-1 à 60-60, 60-250 et 60-741. Les règlements et la clause d'action affirmative contenus dans la phrase précédente doivent être inclus comme référence.

# Table des matières

<b>Préface .....</b>	<b>6</b>
Contenu du guide .....	6
A qui s'adresse ce guide ? .....	6
Documentation connexe .....	6
<b>1 Introduction à FormCalc .....</b>	<b>7</b>
A propos de FormCalc .....	7
A propos des scripts dans LiveCycle Designer ES .....	7
Liste alphabétique des fonctions .....	7
<b>2 Section de référence sur le langage .....</b>	<b>12</b>
Blocs fonctionnels .....	12
A propos des blocs fonctionnels .....	12
Littéraux .....	12
Opérateurs .....	14
Commentaires .....	15
Mots-clés .....	16
Identificateurs .....	16
Termineurs de ligne .....	17
Espace blanc .....	17
Expressions .....	17
A propos des expressions .....	17
Simple .....	18
Affectation .....	20
OU logique .....	20
ET logique .....	21
Unaire .....	21
Egalité et inégalité .....	22
Expression relationnelle .....	22
Expressions if .....	23
Expressions while .....	24
Expressions for .....	25
Expressions foreach .....	26
Expressions break .....	26
Expressions continue .....	27
Variables .....	27
Syntaxe de référence .....	28
Appel des propriétés et des méthodes .....	33
Appels de fonctions intégrées .....	34
<b>3 Liste alphabétique des fonctions .....</b>	<b>35</b>

<b>4</b>	<b>Fonctions arithmétiques</b>	<b>39</b>
	A propos des fonctions arithmétiques	39
	Abs	39
	Avg	40
	Ceil	40
	Count	41
	Floor	41
	Max	42
	Min	43
	Mod	44
	Round	45
	Sum	46
<b>5</b>	<b>Fonctions de date et d'heure</b>	<b>47</b>
	A propos des fonctions de date et d'heure	47
	Structuration des dates et des heures	47
	Paramètres régionaux	47
	Epoque	51
	Formats de date	51
	Formats d'heure	52
	Formats d'image de date et d'heure	53
	Date	57
	Date2Num	57
	DateFmt	58
	IsoDate2Num	59
	IsoTime2Num	59
	LocalDateFmt	60
	LocalTimeFmt	60
	Num2Date	61
	Num2GMTTime	62
	Num2Time	63
	Time	64
	Time2Num	65
	TimeFmt	66
<b>6</b>	<b>Fonctions financières</b>	<b>67</b>
	A propos des fonctions financières	67
	Apr	67
	CTerm	68
	FV	69
	IPmt	70
	NPV	71
	Pmt	72
	PPmt	73
	PV	74
	Rate	75
	Term	76

<b>7</b>	<b>Fonctions logiques</b>	<b>78</b>
	A propos des fonctions logiques	78
	Choose	78
	Exists	79
	HasValue	80
	Oneof	80
	Within	81
<b>8</b>	<b>Fonctions diverses</b>	<b>82</b>
	A propos des fonctions diverses	82
	Eval	82
	Null	83
	Ref	83
	UnitType	84
	UnitValue	85
<b>9</b>	<b>Fonctions de chaîne</b>	<b>86</b>
	A propos des fonctions de chaîne	86
	At	87
	Concat	87
	Decode	88
	Encode	89
	Format	90
	Left	91
	Len	91
	Lower	92
	Ltrim	93
	Parse	93
	Replace	94
	Right	95
	Rtrim	95
	Space	96
	Str	96
	Stuff	97
	Substr	98
	Uuid	99
	Upper	99
	WordNum	100
<b>10</b>	<b>Fonctions URL</b>	<b>102</b>
	A propos des fonctions URL	102
	Get	102
	Post	103
	Put	104
	<b>Index</b>	<b>106</b>

# Préface

---

Adobe® LiveCycle® Designer ES offre un ensemble d'outils qui permet aux développeurs de formulaires de créer des documents professionnels intelligents. Les développeurs peuvent créer des formulaires en y intégrant des fonctions de calcul et des scripts pour que leurs destinataires bénéficient d'un confort d'utilisation inégalé. Par exemple, vous avez la possibilité d'utiliser des calculs simples pour assurer la mise à jour automatique du total d'un bon de commande ou bien des scripts complexes pour adapter la présentation d'un formulaire à la langue et au pays de l'utilisateur.

Pour faciliter la création de fonctions de calcul, LiveCycle Designer ES fournit FormCalc aux utilisateurs. FormCalc est un langage de calcul simple créé par Adobe, qui s'inspire des tableurs traditionnels. FormCalc est simple à utiliser et accessible aux développeurs qui n'ont aucune expérience dans la rédaction de scripts. De nombreuses règles et conventions sont communes aux autres langages de script pour que les développeurs expérimentés mettent à profit leurs connaissances lorsqu'ils utilisent FormCalc.

## Contenu du guide

Ce guide est conçu pour les développeurs de formulaires utilisant LiveCycle Designer ES, qui souhaitent intégrer des fonctions de calcul FormCalc dans leurs formulaires. Le présent guide est un ouvrage de référence sur les fonctions FormCalc, lesquelles sont regroupées en plusieurs chapitres correspondant à leur catégorie. Il propose également une introduction au langage FormCalc et à la création de blocs d'expressions FormCalc.

## A qui s'adresse ce guide ?

Ce guide s'adresse aux développeurs de formulaires qui souhaitent utiliser le langage FormCalc pour enrichir leurs conceptions de formulaire créées dans LiveCycle Designer ES avec des fonctions de calcul.

## Documentation connexe

Pour plus de détails sur l'utilisation des fonctions de calcul FormCalc dans des formulaires, voir la rubrique *Création de calculs et de scripts* de l'aide de *LiveCycle Designer ES*.

Si vous avez besoin de plus d'informations techniques à propos de FormCalc, reportez-vous à *Adobe XML Forms Architecture (XFA) Specification, version 2.4*, disponible sur le site [http://partners.adobe.com/public/developer/xml/index\\_arch.html](http://partners.adobe.com/public/developer/xml/index_arch.html).

## Introduction à FormCalc

### A propos de FormCalc

FormCalc est un langage de calcul simple mais puissant, fondé sur les tableurs traditionnels. Il permet de créer des formulaires, rapidement et efficacement, sans exiger la maîtrise des techniques ou des langages de script traditionnels. A l'aide de quelques fonctions intégrées, les nouveaux utilisateurs de FormCalc peuvent ainsi créer rapidement des formulaires pour éviter aux utilisateurs finaux d'effectuer certaines tâches fastidieuses, telles que les calculs, les validations et autres vérifications. De cette façon, un développeur de formulaires est capable d'intégrer une certaine intelligence à un formulaire au moment de la conception, de façon à ce que le formulaire interactif créé réagisse selon les données entrées.

FormCalc proposent de nombreuses fonctions portant sur des domaines variés, tels que les mathématiques, les dates et heures, les chaînes, la finance, la logique et le Web. Ces catégories représentent les types de données habituellement rencontrées dans les formulaires ; les fonctions permettent de manipuler rapidement et facilement les données à des fins utiles.

### A propos des scripts dans LiveCycle Designer ES

Dans LiveCycle Designer ES, FormCalc est le langage de script par défaut dans tous les emplacements de script. Le langage JavaScript™ constitue l'autre choix possible. Les scripts s'appliquent aux différents événements qui accompagnent chaque objet d'un formulaire. Vous pouvez combiner les langages FormCalc et JavaScript dans vos formulaires interactifs. Toutefois, si vous utilisez un processus serveur, tel que LiveCycle Forms ES, pour créer des formulaires qui seront affichés dans un navigateur Web, les scripts FormCalc de certains événements rattachés aux objets des formulaires ne sont pas reflétés dans les formulaires HTML. Cette fonctionnalité permet d'éviter des erreurs de navigateur Internet lorsque les utilisateurs manipulent les formulaires remplis.

### Liste alphabétique des fonctions

Le tableau suivant contient toutes les fonctions FormCalc disponibles, la description de chaque fonction et la catégorie à laquelle elle appartient.

Fonction	Description	Type
<a href="#">« Abs » à la page 39</a>	Retourne la valeur absolue d'une valeur ou d'une expression numérique.	Arithmétique
<a href="#">« Apr » à la page 67</a>	Renvoie le taux annuel d'un prêt.	Financière
<a href="#">« At » à la page 87</a>	Repère la position du premier caractère d'une chaîne dans une autre chaîne.	Chaîne

<b>Fonction</b>	<b>Description</b>	<b>Type</b>
<a href="#">« Avg » à la page 40</a>	Evalue un ensemble de valeurs et/ou d'expressions numériques et renvoie la moyenne des éléments non nuls de cet ensemble.	Arithmétique
<a href="#">« Ceil » à la page 40</a>	Renvoie le nombre entier supérieur ou égal au nombre fourni.	Arithmétique
<a href="#">« Choose » à la page 78</a>	Sélectionne une valeur dans un ensemble de paramètres donné.	Logique
<a href="#">« Concat » à la page 87</a>	Renvoie la concaténation des chaînes indiquées.	Chaîne
<a href="#">« Count » à la page 41</a>	Evalue un ensemble de valeurs et/ou d'expressions et renvoie le nombre d'éléments non nuls contenus dans cet ensemble.	Arithmétique
<a href="#">« CTerm » à la page 68</a>	Renvoie le nombre de périodes nécessaires pour qu'un investissement, dont le taux d'intérêt est fixe mais composé, atteigne une valeur capitalisée.	Financière
<a href="#">« Date » à la page 57</a>	Renvoie la date active du système sous la forme du nombre de jours écoulés depuis l' <a href="#">époque</a> .	Date et heure
<a href="#">« Date2Num » à la page 57</a>	Renvoie le nombre de jours écoulés depuis l' <a href="#">époque</a> , à partir d'une chaîne de date.	Date et heure
<a href="#">« DateFmt » à la page 58</a>	Renvoie une chaîne de format de date, à partir d'un style de format de date.	Date et heure
<a href="#">« Decode » à la page 88</a>	Renvoie la version décodée d'une chaîne donnée.	Chaîne
<a href="#">« Encode » à la page 89</a>	Renvoie la version codée d'une chaîne donnée.	Chaîne
<a href="#">« Eval » à la page 82</a>	Renvoie la valeur d'un calcul de formulaire donné.	Divers
<a href="#">« Exists » à la page 79</a>	Détermine si le paramètre donné est un mécanisme d'accès à un objet existant.	Logique
<a href="#">« Floor » à la page 41</a>	Renvoie le plus grand nombre entier inférieur ou égal à la valeur donnée.	Arithmétique
<a href="#">« Format » à la page 90</a>	Formate les données fournies selon la chaîne de format d'image indiquée.	Chaîne
<a href="#">« FV » à la page 69</a>	Renvoie la valeur capitalisée de paiements fixes effectués à intervalles réguliers, compte tenu d'un taux d'intérêt fixe.	Financière
<a href="#">« Get » à la page 102</a>	Télécharge le contenu de l'adresse URL indiquée.	URL
<a href="#">« HasValue » à la page 80</a>	Détermine si le paramètre indiqué est un mécanisme d'accès dont la valeur n'est ni nulle, ni vide, ni un blanc.	Logique
<a href="#">« IPmt » à la page 70</a>	Renvoie le montant de l'intérêt payé pour un prêt au cours d'une période donnée.	Financière



<b>Fonction</b>	<b>Description</b>	<b>Type</b>
<a href="#">« IsoDate2Num » à la page 59</a>	Renvoie le nombre de jours écoulés depuis l' <a href="#">époque</a> , à partir d'une chaîne de date valide.	Date et heure
<a href="#">« IsoTime2Num » à la page 59</a>	Renvoie le nombre de millisecondes écoulées depuis l' <a href="#">époque</a> , à partir d'une chaîne d'heure valide.	Date et heure
<a href="#">« Left » à la page 91</a>	Extrait un nombre spécifique de caractères d'une chaîne, en commençant par le premier caractère à gauche.	Chaîne
<a href="#">« Len » à la page 91</a>	Renvoie le nombre de caractères d'une chaîne donnée.	Chaîne
<a href="#">« LocalDateFmt » à la page 60</a>	Renvoie une chaîne de format de date localisée, à partir d'un style de format de date.	Date et heure
<a href="#">« LocalTimeFmt » à la page 60</a>	Renvoie une chaîne de format d'heure localisée, à partir d'un style de format d'heure.	Date et heure
<a href="#">« Lower » à la page 92</a>	Convertit en minuscules toutes les majuscules d'une chaîne donnée.	Chaîne
<a href="#">« Ltrim » à la page 93</a>	Renvoie une chaîne sans aucun caractère d'espace blanc à gauche.	Chaîne
<a href="#">« Max » à la page 42</a>	Renvoie la valeur maximale des éléments non nuls d'un ensemble donné de nombres.	Arithmétique
<a href="#">« Min » à la page 43</a>	Renvoie la valeur minimale des éléments non nuls d'un ensemble donné de nombres.	Arithmétique
<a href="#">« Mod » à la page 44</a>	Renvoie le reste d'un nombre divisé par un autre.	Arithmétique
<a href="#">« NPV » à la page 71</a>	Renvoie la valeur nette actualisée d'un investissement compte tenu d'une série de mouvements futurs de l'encaisse et d'un taux d'actualisation.	Financière
<a href="#">« Null » à la page 83</a>	Renvoie la valeur nulle. La valeur nulle signifie aucune valeur.	Divers
<a href="#">« Num2Date » à la page 61</a>	Renvoie une chaîne de date, à partir du nombre de jours écoulés depuis l' <a href="#">époque</a> .	Date et heure
<a href="#">« Num2GMTTime » à la page 62</a>	Renvoie une chaîne d'heure UT, à partir du nombre de millisecondes écoulées depuis l' <a href="#">époque</a> .	Date et heure
<a href="#">« Num2Time » à la page 63</a>	Renvoie une chaîne d'heure, à partir du nombre de millisecondes écoulées depuis l' <a href="#">époque</a> .	Date et heure
<a href="#">« Oneof » à la page 80</a>	Renvoie « vrai » (1) si une valeur se trouve dans un ensemble donné, et « faux » (0) dans le cas contraire.	Logique
<a href="#">« Parse » à la page 93</a>	Analyse les données fournies selon le format d'image donné.	Chaîne
<a href="#">« Pmt » à la page 72</a>	Renvoie le remboursement d'un prêt basé sur des versements fixes et sur un taux d'intérêt fixe.	Financière
<a href="#">« Post » à la page 103</a>	Place à l'adresse URL indiquée les données fournies.	URL

<b>Fonction</b>	<b>Description</b>	<b>Type</b>
<a href="#">« PPmt » à la page 73</a>	Renvoie le montant du principal payé pour un prêt au cours d'une période.	Financière
<a href="#">« Put » à la page 104</a>	Télécharge les données fournies vers l'adresse URL indiquée.	URL
<a href="#">« PV » à la page 74</a>	Renvoie la valeur actualisée d'un investissement réalisé par versements fixes périodiques avec un taux d'intérêt fixe.	Financière
<a href="#">« Rate » à la page 75</a>	Renvoie le taux d'intérêt composé par période nécessaire pour qu'un investissement d'une valeur actualisée atteigne une valeur capitalisée au cours d'une période donnée.	Financière
<a href="#">« Ref » à la page 83</a>	Renvoie une référence à un objet existant.	Divers
<a href="#">« Replace » à la page 94</a>	Remplace toutes les occurrences d'une chaîne par une autre, dans la chaîne indiquée.	Chaîne
<a href="#">« Right » à la page 95</a>	Extrait un nombre spécifique de caractères dans une chaîne donnée, en commençant par le dernier caractère à droite.	Chaîne
<a href="#">« Round » à la page 45</a>	Evalue une valeur ou expression numérique donnée et renvoie un nombre arrondi comportant le nombre de décimales défini.	Arithmétique
<a href="#">« Rtrim » à la page 95</a>	Renvoie une chaîne sans aucun caractère d'espace blanc à droite.	Chaîne
<a href="#">« Space » à la page 96</a>	Renvoie une chaîne composée d'un nombre donné d'espaces blancs.	Chaîne
<a href="#">« Str » à la page 96</a>	Convertit un nombre en une chaîne de caractères. FormCalc formate le résultat selon la largeur indiquée et l'arrondit au nombre de décimales défini.	Chaîne
<a href="#">« Stuff » à la page 97</a>	Insère une chaîne dans une autre chaîne.	Chaîne
<a href="#">« Substr » à la page 98</a>	Extrait une partie d'une chaîne donnée.	Chaîne
<a href="#">« Sum » à la page 46</a>	Renvoie la somme des éléments non nuls d'un ensemble donné de nombres.	Arithmétique
<a href="#">« Term » à la page 76</a>	Renvoie le nombre de périodes nécessaires pour atteindre la valeur capitalisée donnée, compte tenu de paiements fixes périodiques versés sur un compte portant intérêt.	Financière
<a href="#">« Time » à la page 64</a>	Renvoie l'heure actuelle du système sous la forme du nombre de millisecondes écoulées depuis l' <a href="#">époque</a> .	Date et heure
<a href="#">« Time2Num » à la page 65</a>	Renvoie le nombre de millisecondes écoulées depuis l' <a href="#">époque</a> , à partir d'une chaîne d'heure.	Date et heure

<b>Fonction</b>	<b>Description</b>	<b>Type</b>
<a href="#">« TimeFmt » à la page 66</a>	Renvoie un format d'heure, à partir d'un style de format d'heure.	Date et heure
<a href="#">« UnitType » à la page 84</a>	Renvoie les unités d'une étendue d'unité. Une étendue d'unité est une chaîne composée d'un nombre suivi d'un nom d'unité.	Divers
<a href="#">« UnitValue » à la page 85</a>	Retourne la valeur numérique d'une mesure avec l'étendue d'unité qui lui est associée, après une conversion d'unité facultative.	Divers
<a href="#">« Upper » à la page 99</a>	Convertit en majuscules toutes les minuscules d'une chaîne.	Chaîne
<a href="#">« Uuid » à la page 99</a>	Renvoie une chaîne UUID qui sert de méthode d'identification.	Chaîne
<a href="#">« Within » à la page 81</a>	Renvoie « vrai » (1) si une valeur de contrôle se trouve dans la plage indiquée, et « faux » (0) dans le cas contraire.	Logique
<a href="#">« WordNum » à la page 100</a>	Renvoie le texte équivalent à un nombre donné.	Chaîne

### Blocs fonctionnels

#### A propos des blocs fonctionnels

Le langage FormCalc consiste en un certain nombre de blocs fonctionnels qui composent les expressions FormCalc. Chaque expression FormCalc est une séquence composée d'une combinaison de blocs fonctionnels.

- [« Littéraux » à la page 12](#)
- [« Opérateurs » à la page 14](#)
- [« Commentaires » à la page 15](#)
- [« Mots-clés » à la page 16](#)
- [« Identificateurs » à la page 16](#)
- [« Termineurs de ligne » à la page 17](#)
- [« Espace blanc » à la page 17](#)

### Littéraux

#### A propos des littéraux

Les littéraux sont des valeurs constantes qui forment la base de toutes les valeurs transmises à FormCalc pour traitement. Il existe deux grands types de littéraux : les nombres et les chaînes.

#### Littéraux numériques

Un littéral numérique est une séquence composée surtout de chiffres ; elle comprend un ou plusieurs des caractères suivants : un entier, un point décimal, une fraction, un indicateur d'exposant (« e » ou « E ») et une valeur d'exposant avec signe facultatif. Voici quelques exemples de littéraux numériques :

- -12
- 1.5362
- 0.875
- 5.56e-2
- 1.234E10

Dans un littéral numérique, il est possible d'omettre la partie de l'entier ou de la fraction, mais non les deux. En outre, dans la partie de la fraction, on peut aussi omettre le point décimal ou la valeur de l'exposant, mais non les deux.

Tous les littéraux numériques sont convertis en interne en valeurs binaires 64 bits IEEE. Cependant, puisque les valeurs IEEE ne peuvent représenter qu'une quantité finie de nombres, certaines valeurs n'ont pas de représentation en fraction binaire. Cette situation est comparable au fait que certaines valeurs, comme  $1/3$ , n'ont pas de représentation précise en fraction décimale (la valeur décimale aurait besoin d'un nombre infini de décimales pour être tout à fait exacte).

Ces valeurs qui ne possèdent pas d'équivalent en fraction binaire sont généralement les littéraux numériques comptant plus de 16 chiffres significatifs avant l'exposant. FormCalc arrondit ces valeurs à la valeur 64 bits IEEE représentable la plus proche selon les normes de l'IEEE. Par exemple, la valeur :

```
123456789.012345678
```

est arrondie à la valeur (la plus près) :

```
123456789.01234567
```

Autre exemple, le littéral numérique :

```
999999999999999999
```

est arrondi à la valeur (la plus près) :

```
1000000000000000000
```

Cette façon de faire entraîne parfois des résultats surprenants. FormCalc possède une fonction, [Round](#), qui renvoie le nombre indiqué arrondi au nombre indiqué de décimales. Lorsque le nombre indiqué est exactement à mi-chemin entre deux nombres représentables, il est arrondi au plus loin de zéro. Autrement dit, le nombre est arrondi à la hausse s'il est positif et à la baisse s'il est négatif. Considérez l'exemple suivant :

```
Round(0.124, 2)
```

renvoie 0.12,

and

```
Round(.125, 2)
```

renvoie 0.13.

D'après cette convention, vous pourriez penser que :

```
Round(0.045, 2)
```

renvoie 0.05.

Cependant, la norme IEEE 754 stipule que le littéral numérique 0.045 reçoit la valeur approximative 0.044999999999999999. Cette approximation est plus près de 0.04 que de 0.05. C'est pourquoi :

```
Round(0.045, 2)
```

renvoie 0.04.

Cette façon de faire est conforme à la norme IEEE 754.

Les valeurs 64 bits IEEE reconnaissent les représentations comme NaN ("not a number", n'est pas un nombre), +Inf (infini positif) et -Inf (infini négatif). FormCalc ne les gère pas ; toute expression dont l'évaluation a pour résultat NaN, +Inf ou -Inf entraîne une exception d'erreur, transmise au reste de l'expression.

### Littéraux chaînes

Un littéral chaîne est une séquence constituée de tout caractère Unicode, précédée et suivie de guillemets. Par exemple :

```
"The cat jumped over the fence."  
"Number 15, Main street, California, U.S.A"
```

Le littéral chaîne "" définit une séquence vide (sans caractères) appelée chaîne vide.

Pour incorporer un guillemet anglais double (") dans un littéral chaîne, vous devez insérer deux guillemets anglais double. Par exemple :

```
"The message reads: ""Warning: Insufficient Memory"""
```

Tous les caractères Unicode ont une séquence d'échappement de 6 caractères équivalente, composée de \u suivie de quatre chiffres hexadécimaux. Dans tout littéral chaîne, il est possible d'exprimer n'importe quel caractère, y compris les caractères de commande, au moyen de la séquence d'échappement Unicode équivalente. Par exemple :

```
"\u0047\u0066\u0066\u0069\u0073\u0068\u0021"  

"\u000d" (carriage return)  

"\u000a" (newline character)
```

## Opérateurs

FormCalc inclut un certain nombre d'opérateurs : unaires, multiplicatifs, additifs, relationnels, logiques, d'égalité et d'affectation.

Plusieurs opérateurs FormCalc ont un mot-clé mnémotechnique équivalent. Ces mots-clés sont utiles lorsque des expressions FormCalc sont incorporées dans du texte source HTML et XML, où les symboles " inférieur à " (<), " supérieur à " (>), et " et commercial " (&) ont des significations prédéfinies, et doivent donc être obtenus par des séquences d'échappement. Le tableau suivant dresse la liste de tous les opérateurs FormCalc accompagnés des formes symboliques et, le cas échéant, mnémotechniques.

Type d'opérateur	Représentations
Addition	+
Division	/
Egalité	== eq <> ne
ET logique	& and
OU logique	ou
Multiplication	*
Expression relationnelle	< lt (inférieur à) > gt (supérieur à) <= le (inférieur ou égal à) >= ge (supérieur ou égal à)
Soustraction	-
Unaire	- + not

## Commentaires

Les commentaires sont des sections de code que FormCalc n'exécute pas. Les commentaires contiennent normalement des renseignements ou des directives qui expliquent l'utilité d'un certain fragment de code. Au moment de l'exécution, FormCalc ne tient pas compte des informations stockées en commentaires.

Vous pouvez indiquer un commentaire en entrant soit un point-virgule (;), soit une paire de barres obliques (//). Dans FormCalc, un commentaire s'étend depuis son début jusqu'au prochain terminateur de ligne.

Nom du caractère	Représentations
Commentaire	; //

Par exemple :

```
// This is a type of comment
First_Name="Tony"
Initial="C" ;This is another type of comment
Last_Name="Blue"
```

### Commentaires de tous les calculs FormCalc pour un événement

L'ajout de commentaires à tous les calculs FormCalc pour un événement particulier entraîne une erreur lors de l'affichage de votre formulaire dans le panneau Aperçu PDF ou lors de l'affichage du PDF final. Chaque calcul FormCalc doit renvoyer une valeur, et FormCalc ne considère pas les commentaires comme étant des valeurs.

Pour empêcher que le code des commentaires FormCalc ne renvoie une erreur, effectuez l'une des opérations suivantes :

- Supprimez le code de commentaires de l'événement.
- Ajoutez une expression qui renvoie une valeur au code de FormCalc pour l'événement.

Pour empêcher que la valeur de l'expression n'entraîne des résultats indésirables sur votre formulaire, utilisez l'un des types d'expressions suivants :

- Une simple expression faite d'un caractère unique, comme suit :

```
// First_Name="Tony"
// Initial="C"
// Last_Name="Blue"
//
// The simple expression below sets the value of the event to zero.
0
```

- Une expression d'affectation conservant la valeur de l'objet. Utilisez ce type d'expression si le code de commentaire de FormCalc se trouve dans l'événement de calcul afin d'empêcher que la valeur réelle de l'objet ne soit modifiée, comme suit :

```
// First_Name="Tony"
// Initial="C"
// Last_Name="Blue"
//
// The assignment expression below sets the value of the current
// field equal to itself.
$.rawValue = $.rawValue
```

## Mots-clés

Les mots de passe FormCalc sont réservés et ne font pas la distinction entre majuscules et minuscules. Ces mots-clés sont réservés aux parties d'expressions, aux littéraux numériques spéciaux et aux opérateurs.

Le tableau suivant dresse la liste des mots-clés FormCalc. N'utilisez aucun de ces mots lorsque vous nommez un objet de votre conception de formulaire.

and	endif	in	step
break	endwhile	infinity	then
continue	eq	le	this
do	exit	lt	throw
downto	for	nan	upto
else	foreach	ne	var
elseif	func	not	while
end	ge	null	
endfor	gt	or	
endfunc	if	return	

## Identificateurs

Un identificateur est une séquence de caractères de longueur illimitée qui représente le nom d'une fonction ou d'une méthode. Un identificateur commence toujours par l'un des caractères suivants :

- N'importe quel caractère alphabétique (d'après le classement des lettres Unicode)
- Trait de soulignement ( \_ )
- Signe du dollar ( \$ )
- Point d'exclamation ( ! )

Les identificateurs FormCalc distinguent les lettres majuscules et les lettres minuscules. Autrement dit, les identificateurs dont les caractères ne diffèrent que par les majuscules ou les minuscules sont considérés comme distincts.

Nom du caractère	Représentations
Identificateur	A..Z,a..z \$ ! _

Voici quelques exemples d'identificateurs valides :

```
GetAddr  
$primary  
_item  
!dbresult
```



## Terminateurs de ligne

Les terminateurs de ligne sont utilisés pour séparer des lignes et améliorer la lisibilité.

Le tableau suivant dresse la liste des terminateurs de ligne valides FormCalc :

Nom du caractère	Caractères Unicode
Retour chariot	#xD U+000D
Changement de ligne	#xA &#x000D; &#D;

## Espace blanc

Les caractères d'espace blanc sont ceux qui séparent les objets et les opérations mathématiques. Ces caractères servent strictement à améliorer la lisibilité et n'ont aucune importance au moment du traitement dans FormCalc.

Nom du caractère	Caractère Unicode
Saut de page	#xC
Tabulation horizontale	#x9
Barre espace	#x20
Tabulation verticale	#xB

## Expressions

### A propos des expressions

Les littéraux, opérateurs, commentaires, mots-clés, identificateurs, terminateurs de ligne et espaces blancs constituent ensemble une liste d'expressions, même lorsque la liste ne contient qu'une seule expression. En général, chaque expression de la liste renvoie une valeur ; la valeur de la liste dans son ensemble est celle de la dernière expression de la liste.

Considérez, par exemple, les deux champs suivants d'une conception de formulaire :

Nom de champ	Calculs	Renvoi
Field1	5 + Abs(Price) "Hello World" 10 * 3 + 5 * 4	50
Field2	10 * 3 + 5 * 4	50

La valeur de `Field1` et de `Field2` après l'évaluation de l'expression de chaque champ est 50.

FormCalc classe comme suit les différents types d'expressions qui forment une liste d'expressions :

- [« Simple » à la page 18](#)
- [« Affectation » à la page 20](#)
- [« OU logique » à la page 20](#)
- [« ET logique » à la page 21](#)
- [« Unaire » à la page 21](#)
- [« Egalité et inégalité » à la page 22](#)
- [« Expression relationnelle » à la page 22](#)
- [« Expressions if » à la page 23](#)
- [« Expressions while » à la page 24](#)
- [« Expressions for » à la page 25](#)
- [« Expressions foreach » à la page 26](#)
- [« Expressions break » à la page 26](#)
- [« Expressions continue » à la page 27](#)

## Simple

### A propos des expressions simples

Dans leur forme leur plus simple, les expressions FormCalc sont des groupes d'opérateurs, mots-clés et littéraux assemblés de façon logique. Voici des exemples d'expressions simples :

```
2
"abc"
2 - 3 * 10 / 2 + 7
```

Chaque expression FormCalc renvoie une seule valeur, après exécution des opérations dans l'ordre traditionnel, même si cet ordre n'est pas toujours évident à déterminer d'après la syntaxe de l'expression. Ainsi, les ensembles d'expressions suivants, lorsqu'on les applique à des objets de conception de formulaire, produisent les mêmes résultats :

Expression	Equivalent à	Renvoie
"abc"	"abc"	abc
2 - 3 * 10 / 2 + 7	2 - (3 * (10 / 2)) + 7	-6
10 * 3 + 5 * 4	(10 * 3) + (5 * 4)	50
0 and 1 or 2 > 1	(0 and 1) or (2 >1)	1 (vrai)
2 < 3 not 1 == 1	(2 < 3) not (1 == 1)	0 (faux)

Comme le suggère le tableau précédent, tous les opérateurs FormCalc possèdent un certain niveau de priorité lorsqu'ils apparaissent dans des expressions. Le tableau suivant illustre la hiérarchie des opérateurs :

Priorité	Opérateur
La plus élevée	=
	(Unaire) -, +, not
	*, /
	+, -
	<, <=, >, >=, lt, le, gt, ge
	==, <>, eq, ne
	&, and
La plus basse	, or

### Promotion d'opérandes

Dans le cas où une ou plusieurs opérandes d'une opération ne correspondent pas au type prévu pour cette opération, FormCalc promeut les opérandes pour qu'elles correspondent au type requis. La façon dont se fait cette promotion dépend du type d'opérande exigé par l'opération.

### Opérations numériques

Lorsque des opérandes non numériques se retrouvent dans des opérations numériques, elles sont d'abord promues à leur équivalent numérique. Si l'opérande non numérique n'est pas convertie correctement en valeur numérique, sa valeur devient zéro (0). La promotion d'opérandes dont la valeur est nulle en nombres donne toujours la valeur zéro.

Le tableau suivant donne quelques exemples de promotion d'opérandes non numériques.

Expression	Equivalent à	Renvoi
(5 - "abc") * 3	(5 - 0) * 3	15
"100" / 10e1	100 / 10e1	1
5 + null + 3	5 + 0 + 3	8

### Opération booléenne

Lorsque des opérandes non booléennes se retrouvent dans des opérations booléennes, elles sont d'abord promues à leur équivalent booléen. Si l'opérande non booléenne n'est pas convertie correctement en une valeur différente de zéro, sa valeur est Vrai (1) ; sinon, sa valeur est Faux (0). La promotion d'opérandes dont la valeur est nulle en valeur booléenne donne toujours la valeur Faux (0). Par exemple, l'expression :

```
"abc" | 2
```

renvoie 1. Autrement dit, Faux | Vrai = Vrai, alors que

```
if ("abc") then
    10
else
    20
endif
```

renvoie 20.

## Opérations de chaîne

Lorsque des opérandes non chaînes se retrouvent dans des opérations de chaîne, elles sont d'abord promues en chaînes tirées de leur valeur. La promotion d'opérandes dont la valeur est nulle en chaînes donne toujours la chaîne vide. Par exemple, l'expression :

```
concat("The total is ", 2, " dollars and ", 57, " cents.")
```

renvoie "The total is 2 dollars and 57 cents."

**Remarque :** si durant l'évaluation d'une expression, une étape intermédiaire génère NaN, +Inf ou -Inf, FormCalc génère une exception d'erreur et propage cette erreur dans le reste de l'expression. La valeur de l'expression est alors toujours 0. Par exemple :

```
3 / 0 + 1
```

renvoie 0.

## Affectation

Une expression d'affectation attribue la valeur d'une expression simple à la propriété identifiée par une syntaxe de référence donnée. Par exemple :

```
$template.purchase_order.name.first = "Tony"
```

Cette expression définit la valeur de la conception de formulaire « first » sur Tony.

Pour plus de détails sur l'utilisation de la syntaxe de référence, voir [« Syntaxe de référence » à la page 28](#).

Pour plus de détails sur l'utilisation de la syntaxe de référence, voir [« A propos des références aux objets dans les calculs et les scripts » à la page 642](#).

## OU logique

Une expression OU logique renvoie soit Vrai (1) si au moins l'une de ses opérandes est Vrai (1), soit Faux (0) si les deux opérandes sont Faux (0). Si les deux opérandes sont nulles, l'expression renvoie Nul.

Expression	Représentation par des caractères
OU logique	 ou

Voici quelques exemples d'utilisation de l'expression OU logique :

Expression	Renvoie
1 or 0	1 (vrai)
0   0	0 (faux)
0 or 1   0 or 0	1 (vrai)

## ET logique

Une expression ET logique renvoie soit Vrai (1) si les deux opérandes sont Vrai (1), soit Faux (0) si au moins l'une de ses opérandes est Faux (0). Si les deux opérandes sont nulles, l'expression renvoie Nul.

Expression	Représentation par des caractères
ET logique	& and

Voici quelques exemples d'utilisation de l'expression ET logique :

Expression	Renvoie
1 and 0	0 (faux)
0 & 0	1 (vrai)
0 and 1 & 0 and 0	0 (faux)

## Unaire

Une expression unaire renvoie des résultats différents selon l'opérateur unaire utilisé.

Expression	Représentation par des caractères	Renvoie
Unaire	-	La négation arithmétique de l'opérande, ou nul si l'opérande est nulle.
	+	La valeur arithmétique de l'opérande (inchangée), ou nul si l'opérande est nulle.
	not	La négation logique de l'opérande.

**Remarque :** la négation arithmétique d'une opérande nulle donne le résultat nul, alors que la négation logique d'une opérande nulle renvoie le résultat booléen Vrai. Pour comprendre ces résultats, suivez le raisonnement suivant : Si « nul » signifie « rien », alors « pas rien » signifie « quelque chose ».

Voici quelques exemples d'utilisation de l'expression unaire :

Expression	Renvoie
-(17)	-17
-(-17)	17
+(17)	17
+(-17)	-17
not("true")	1 (vrai)
not(1)	0 (faux)

## Egalité et inégalité

Les expressions d'égalité et d'inégalité renvoient le résultat d'une comparaison d'égalité de ses opérandes.

Expression	Représentation par des caractères	Renvoi
Egalité	== eq	Vrai (1) lorsque les deux opérandes comparées sont identiques, et Faux (0) si elles sont différentes.
Inégalité	<> ne	Vrai (1) lorsque les deux opérandes comparées sont différentes, et Faux (0) si elles sont identiques.

Les cas spéciaux suivants s'appliquent aussi à l'utilisation des opérateurs d'égalité :

- Si l'une des opérande est nulle, une comparaison de valeur nulle est exécutée. Si les deux opérandes sont nulles, le résultat est " identique " ; si l'une des opérandes n'est pas nulle, le résultat est " différent ".
- Si les deux opérandes sont des références, elles sont considérées identiques si elles se rapportent toutes deux au même objet, et différentes si elles ne se rapportent pas au même objet.
- Si les deux opérandes ont pour valeur une chaîne, une comparaison lexicale des chaînes, prenant en compte le paramètre régional, est exécutée sur les opérandes. Sinon, si elles ne sont pas nulles, les opérandes sont promues en valeurs numériques et une comparaison numérique a lieu.

Voici quelques exemples d'utilisation des expressions d'égalité et d'inégalité :

Expression	Renvoi
3 == 3	1 (vrai)
3 <> 4	1 (vrai)
"abc" eq "def"	0 (faux)
"def" ne "abc"	1 (vrai)
5 + 5 == 10	1 (vrai)
5 + 5 <> "10"	0 (faux)

## Expression relationnelle

Une expression relationnelle renvoie le résultat booléen de la comparaison relationnelle des opérandes.

Expression	Représentation par des caractères	Renvoi
Expression relationnelle	< lt	Vrai (1) lorsque la première opérande est inférieure à la seconde ; faux (0) lorsque la première opérande est supérieure à la seconde.
	> gt	Vrai (1) lorsque la première opérande est supérieure à la seconde ; faux (0) lorsque la première opérande est inférieure à la seconde.

Expression	Représentation par des caractères	Renvoie
	<= le	Vrai (1) lorsque la première opérande est inférieure ou égale à la seconde ; faux (0) lorsque la première opérande est supérieure à la seconde.
	>= ge	Vrai (1) lorsque la première opérande est supérieure ou égale à la seconde ; faux (0) lorsque la première opérande est inférieure à la seconde.

Les cas spéciaux suivants s'appliquent aussi à l'utilisation des opérateurs relationnels :

- Si l'une des opérandes est nulle, une comparaison de valeur nulle est exécutée. Les opérandes dont la valeur est nulle sont considérées identiques si les deux opérandes sont nulles et que l'opérateur relationnel est " inférieur ou égal à " ou " supérieur ou égal à " ; sinon, elles sont considérées différentes.
- Si les deux opérandes ont pour valeur une chaîne, une comparaison lexicale des chaînes, prenant en compte le paramètre régional, est exécutée sur les opérandes. Sinon, si elles ne sont pas nulles, les opérandes sont promues en valeurs numériques et une comparaison numérique a lieu.

Voici quelques exemples d'utilisation de l'expression relationnelle :

Expression	Renvoie
3 < 3	0 (faux)
3 > 4	0 (faux)
"abc" <= "def"	1 (vrai)
"def" > "abc"	1 (vrai)
12 >= 12	1 (vrai)
"true" < "false"	0 (faux)

## Expressions if

Une expression (ou instruction) if évalue si une expression simple est vraie ; le résultat renvoyé est une liste d'expressions correspondant à la valeur Vrai. Si l'expression simple initiale est fausse (0), FormCalc vérifie les conditions elseif ou else et renvoie, le cas échéant, les résultats de leurs listes d'expressions.

Expression	Syntaxe	Renvoie
if	<pre>                     if ( simple expression ) then                         list of expressions                     elseif ( simple expression ) then                         list of expressions                     else                         list of expressions                     endif                 </pre>	Résultat de la liste d'expressions associée à toute condition correcte énoncée dans l'expression if.  <b>Remarque :</b> vous n'êtes pas tenu d'insérer des instructions elseif(...) ou else dans l'expression if. Cependant, vous devez signifier la fin de l'expression par endif.

Voici quelques exemples d'utilisation de l'expression if :

Expression	Renvoie
<pre>if ( 1 &lt; 2 ) then   1 endif</pre>	1
<pre>if ( "abc" &gt; "def" ) then   1 and 0 else   0 endif</pre>	0
<pre>if ( Field1 &lt; Field2 ) then   Field3 = 0 elseif ( Field1 &gt; Field2 ) then   Field3 = 40 elseif ( Field1 = Field2 ) then   Field3 = 10 endif</pre>	Varie selon la valeur de Field1 et de Field2. Par exemple, si Field1 est 20 et Field2 est 10, cette expression définit Field3 à 40.

## Expressions while

Une expression while est une instruction itérative, aussi appelée instruction de boucle, qui évalue une expression simple donnée. Si le résultat de l'évaluation est Vrai (1), FormCalc examine continuellement la condition do et renvoie les résultats des listes d'expressions. Si le résultat est Faux (0), le contrôle passe alors à l'instruction suivante.

Une expression while est particulièrement adaptée aux situations où une répétition conditionnelle est nécessaire. En revanche, les situations demandant l'utilisation d'une répétition non conditionnelle sont souvent traitées plus efficacement avec une expression for.

Expression	Syntaxe	Renvoie
While	<pre>while ( simple expression ) do   expression list endwhile</pre>	Le résultat de la liste d'expressions associée à la condition do.

Dans l'exemple suivant, les valeurs des éléments sont ajoutées à une liste déroulante à partir d'un fichier XML. La méthode addItem est utilisée pour l'ensemble des éléments XML répertoriés sous list1 qui ne sont pas égaux à 3 :

```
var List = ref(xfa.record.lists.list1)
var i = 0
while ( List.nodes.item(i+1).value ne "3" ) do
$.addItem (List.nodes.item(i).value, List.nodes.item(i+1).value)
i = i + 2
endwhile
```



## Expressions for

Une expression for est une boucle ou une instruction itérative conditionnelle.

Une expression for est particulièrement adaptée aux situations de boucles où une répétition non conditionnelle est nécessaire. En revanche, les situations demandant l'utilisation d'une répétition conditionnelle sont souvent traitées plus efficacement avec une expression while.

La valeur de l'expression for correspond à la valeur de la dernière liste d'évaluation évaluée ou à Faux (0).

La condition for initialise une variable FormCalc qui contrôle l'action de boucle.

Dans la variante upto, la valeur de la variable de boucle effectue une itération de l'expression start à l'expression end, en utilisant pour cela des incréments de l'expression step. Si vous n'indiquez aucune valeur pour l'expression step, l'incrément step prend la valeur 1 par défaut.

Dans la variante downto, la valeur de la variable de boucle effectue une itération de l'expression start à l'expression end, en utilisant pour cela des décréments de l'expression step. Si vous n'indiquez aucune valeur pour l'expression step, les décréments step prennent la valeur -1 par défaut.

Les itérations de la boucle sont contrôlées par la valeur de l'expression end. Avant chaque itération, l'expression end est évaluée et comparée à la variable de boucle. Si le résultat est Vrai (1), la liste d'expressions est évaluée. Après chaque évaluation, l'expression step est évaluée et ajoutée à la variable de boucle.

Avant chaque itération, l'expression end est évaluée et comparée à la variable de boucle. De plus, après chaque évaluation de la condition do, l'expression step est évaluée et ajoutée à la variable de boucle.

Une boucle for se termine lorsque l'expression start a dépassé l'expression end. L'expression start peut dépasser l'expression end vers le haut, si vous utilisez la variante upto, ou vers le bas, si vous utilisez la variante downto.

Expression	Syntaxe	Renvoie
Afin	<pre>for variable = start expression   (upto   downto ) end expression   (step step expression ) do   expression list endfor</pre> <p><b>Remarque :</b> les expressions start, end et step doivent toutes être des expressions simples.</p>	Le résultat de la liste d'expressions associée à la condition do.

Dans l'exemple suivant, les valeurs des éléments sont ajoutées à une liste déroulante à partir d'un fichier XML. La méthode addItem est utilisée pour l'ensemble des éléments XML répertoriés sous list1 :

```
var List = ref(xfa.record.lists.list1)
for i=0 upto List.nodes.length - 1 step 2 do
$.addItem (List.nodes.item(i).value, "")
endfor
```

## Expressions foreach

Une expression foreach effectue une itération sur la liste d'expressions pour chaque valeur de sa liste d'arguments.

La valeur de l'expression foreach correspond à la valeur de la dernière liste d'expressions évaluée, ou à zéro (0) si aucune boucle n'a été indiquée.

La condition in, exécutée une seule fois (après déclaration de la variable de boucle), contrôle l'itération de la boucle. Avant chaque itération, des valeurs consécutives issues de la liste d'arguments sont affectées à la variable de boucle. La liste d'arguments ne peut pas être vide.

Expression	Syntaxe	Renvoie
foreach	<pre>foreach variable in( <i>argument list</i> )do   expression list endfor</pre> <p><b>Remarque :</b> utilisez une virgule (,) pour séparer les expressions simples de la liste d'arguments.</p>	La valeur de la dernière liste d'expressions évaluée ou zéro (0) si aucune boucle n'a été indiquée.

Dans l'exemple suivant, seules les valeurs des éléments XML « display » sont ajoutées à la liste déroulante foreach.

```
foreach Item in (xfa.record.lists.list1.display[*]) do  
$.addItem(Item, "")  
endfor
```

## Expressions break

Une expression break provoque une sortie immédiate à partir de la boucle d'expression while, for ou foreach la plus centrale. Le contrôle est alors transmis à l'expression suivant la boucle terminée.

La valeur de l'expression break est toujours égale à zéro (0).

Expression	Syntaxe	Renvoie
Saut	break	Transmet le contrôle à l'expression suivant la boucle terminée.

Dans l'exemple suivant, une condition if est placée dans la boucle while pour vérifier si la valeur actuelle est égale à « Display data for 2 ». Si l'expression if est Vrai, l'expression break est exécutée et interrompt la boucle.

```
var List = ref(xfa.record.lists.list1)  
var i=0  
while (List.nodes.item(i+1).value ne "3") do  
$.addItem(List.nodes.item(i).value, List.nodes.item(i+1).value)  
i = i + 2  
if (List.nodes.item(i) eq "Display data for 2" then  
break  
endif  
endwhile
```

## Expressions continue

Une expression continue déclenche l'itération suivante de la boucle while, for ou foreach la plus centrale.

La valeur de l'expression continue est toujours égale à zéro (0).

Expression	Syntaxe	Renvoi
Continuer	continue	Lorsqu'elle est utilisée dans une expression while, le contrôle est transmis à la condition while. Lorsqu'elle est utilisée dans une expression for, le contrôle est transmis à l'expression step.

L'objectif de l'exemple suivant est de remplir la liste déroulante à l'aide des valeurs d'un fichier XML. Si la valeur de l'élément XML actuel est « Display data for 3 », l'expression break entraîne la sortie de la boucle while. Si la valeur de l'élément XML actuel est « Display data for 2 », le script ajoute 2 à la variable i (le compteur) et la boucle passe immédiatement à son cycle suivant. Les deux dernières lignes ne sont pas prises en compte lorsque la valeur de l'élément XML actuel est « Display data for 2 ».

```
var List = ref(xfa.record.lists.list1)
var i = 0
while (List.nodes.item(i+1).value ne "5") do
if (List.nodes.item(i) eq "Display data for 3") then
break
endif
if (List.nodes.item(i) eq "Display data for 2" then
i=i+2
continue
endif
$.addItem(List.nodes.item(i).value,List.nodes.item(i+1).value)
i=i+2
endwhile
```

## Variables

Dans vos calculs, FormCalc permet de créer et de manipuler des variables pour stocker les données. Le nom que vous affectez à chaque variable que vous créez doit être un identificateur unique.

Par exemple, les expressions FormCalc suivantes définissent la variable userName et la valeur d'un champ de texte qui doit correspondre à la valeur de userName.

```
var userName = "Tony Blue"
TextField1.rawValue = userName
```

Vous pouvez référencer les variables que vous définissez dans l'onglet Variables de la boîte de dialogue Propriétés du formulaire de la même façon. Les expressions FormCalc suivantes utilisent la fonction Concat pour définir la valeur d'un champ de texte à l'aide des variables de formulaire salutation et name.

```
TextField1.rawValue = Concat(salutation, name)
```

**Remarque :** une variable créée avec FormCalc a priorité sur une variable d'un nom semblable définie dans le panneau Variable de la boîte de dialogue Propriétés du formulaire.

## Syntaxe de référence

FormCalc donne accès aux propriétés et aux valeurs des objets de conception de formulaire au moyen d'une syntaxe de référence. L'exemple suivant illustre l'affectation et la récupération des valeurs d'un objet :

```
Invoice_Total.rawValue = Invoice_SubTotal.rawValue * (8 / 100)
```

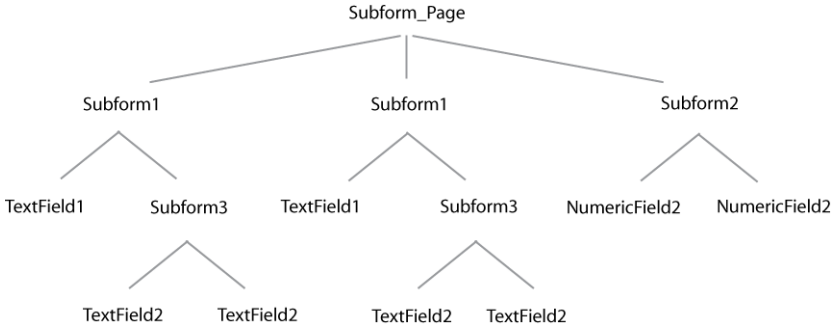
Dans le cas présent, la syntaxe de référence `Invoice_Total` affecte la valeur de `Invoice_SubTotal * (8 / 100)` au champ `Invoice_Total`.

Dans le contexte des conceptions de formulaire, une syntaxe de référence complète permet l'accès à tous les objets.

Pour faciliter l'accès aux propriétés et aux valeurs et limiter votre travail, FormCalc offre des raccourcis pratiques pour créer des références. Le tableau suivant décrit les raccourcis de la syntaxe de référence pour FormCalc.

Notation	Description
\$	Fait référence au champ ou à l'objet actuel, comme l'illustre cet exemple : <pre>\$ = "Tony Blue"</pre> L'exemple ci-dessus définit la valeur du champ ou de l'objet actuel sur <code>Tony Blue</code> .
\$data	Représente la racine du modèle de données <code>xfa.datasets.data</code> . Par exemple : <pre>\$data.purchaseOrder.total</pre> est équivalent à <pre>xfa.datasets.data.purchaseOrder.total</pre>
\$event	Représente l'événement d'objet de formulaire actif. Par exemple : <pre>\$event.name</pre> est équivalent à <pre>xfa.event.name</pre> Pour plus de détails, voir <a href="#">A propos du modèle d'événement</a> .
\$form	Représente la racine du modèle de formulaire <code>xfa.form</code> . Par exemple : <pre>\$form.purchaseOrder.tax</pre> est équivalent à <pre>xfa.form.purchaseOrder.tax</pre>
\$host	Représente l'objet hôte. Par exemple : <pre>\$host.messageBox("Hello world")</pre> est équivalent à <pre>xfa.host.messageBox("Hello world")</pre> Pour plus de détails, voir <a href="#">A propos de l'utilisation d'une application hôte</a> .

Notation	Description
\$layout	Représente la racine du modèle de disposition <code>xfa.layout</code> . Par exemple : <code>\$layout.ready</code> est équivalent à <code>xfa.layout.ready</code>
\$record	Représente l'enregistrement actuel d'une collection de données, provenant d'un fichier XML par exemple. Par exemple : <code>\$record.header.txtOrderedByCity</code> référence le noeud <code>txtOrderedByCity</code> dans le noeud <code>header</code> des données XML actuelles.
\$template	Représente la racine du modèle de modèle <code>xfa.template</code> . Par exemple : <code>\$template.purchaseOrder.item</code> est équivalent à <code>xfa.template.purchaseOrder.item</code>
!	Représente la racine du modèle de données <code>xfa.datasets</code> . Par exemple : <code>!data</code> est équivalent à <code>xfa.datasets.data</code>
*	Sélectionne tous les objets de formulaire dans un conteneur donné (tel qu'un sous-formulaire), quel que soit le nom, ou sélectionne tous les objets ayant un nom similaire.  Par exemple, l'expression suivante sélectionne tous les objets nommés <code>item</code> sur un formulaire : <code>xfa.form.form1.item[*]</code>  <b>Remarque :</b> l'utilisation d'un astérisque (*) est admise dans le langage JavaScript avec la méthode <a href="#">resolveNode</a> <code>resolveNode</code> . For more information about the <code>resolveNode</code> method, see <i>LiveCycle Designer ES Help</i> , or see the <i>LiveCycle Designer ES Scripting Reference</i> at <a href="http://www.adobe.com/go/learn_lc_scriptingReference">http://www.adobe.com/go/learn_lc_scriptingReference</a> .

Notation	Description
..	<p>Vous pouvez utiliser deux points à un endroit quelconque de votre syntaxe de référence pour rechercher des objets faisant partie de n'importe quel sous-conteneur du conteneur actuel (tel qu'un sous-formulaire). Par exemple, l'expression <code>Subform_Page..Subform2</code> signifie rechercher le nœud <code>Subform_Page</code> (comme d'habitude) et trouver un descendant de <code>Subform_Page</code> appelé <code>Subform2</code>.</p>  <pre> graph TD     SP[Subform_Page] --&gt; S1_1[Subform1]     SP --&gt; S1_2[Subform1]     SP --&gt; S2[Subform2]     S1_1 --&gt; T1_1[TextField1]     S1_1 --&gt; S3_1[Subform3]     S1_2 --&gt; T1_2[TextField1]     S1_2 --&gt; S3_2[Subform3]     S2 --&gt; NF1[NumericField2]     S2 --&gt; NF2[NumericField2]     S3_1 --&gt; T2_1[TextField2]     S3_1 --&gt; T2_2[TextField2]     S3_2 --&gt; T2_3[TextField2]     S3_2 --&gt; T2_4[TextField2]     </pre> <p>En se fondant sur l'arborescence ci-dessus,  <code>Subform_Page..TextField2</code>  est équivalent à  <code>Subform_Page.Subform1 [0] .Subform3 .TextField2 [0]</code>  puisque <code>TextField2 [0]</code> est le premier nœud <code>Subform1</code> que FormCalc rencontre lors de sa recherche. Deuxième exemple :  <code>Subform_Page..Subform3 [*]</code>  renvoie la totalité des quatre objets <code>TextField2</code>.</p> <p><b>Remarque :</b> vous pouvez utiliser la syntaxe double point (..) avec JavaScript en faisant appel à la méthode <code>resolveNode</code>. For more information about the <code>resolveNode</code> method, see <i>LiveCycle Designer ES Help</i>, or see the <i>LiveCycle Designer ES Scripting Reference</i> at <a href="http://www.adobe.com/go/learn_lc_scriptingReference">http://www.adobe.com/go/learn_lc_scriptingReference</a>.</p>

Notation	Description
#	<p>La notation # (signe dièse) est utilisée pour représenter l'un des éléments suivants dans une syntaxe de référence :</p> <ul style="list-style-type: none"><li>● Un objet sans nom. Ainsi, la syntaxe de référence suivante accède à un sous-formulaire sans nom : <code>xfa.form.form1.#subform</code></li><li>● Une propriété dans une syntaxe de référence si une propriété et un objet possèdent le même nom. La syntaxe de référence suivante accède, par exemple, à la propriété <code>name</code> d'un sous-formulaire si ce dernier contient également un champ appelé <code>name</code> : <code>xfa.form.form1.#subform.#name</code></li></ul> <p><b>Remarque :</b> l'utilisation du signe dièse (#) est admise dans le langage JavaScript avec la méthode <a href="#">resolveNode</a> <code>resolveNode</code>. For more information about the <code>resolveNode</code> method, see <i>LiveCycle Designer ES Help</i>, or see <i>LiveCycle Designer ES Scripting Reference</i> at <a href="http://www.adobe.com/go/learn_lc_scriptingReference">http://www.adobe.com/go/learn_lc_scriptingReference</a>.</p>

Notation	Description
[ ]	<p>Le crochet ( [ ] ) représente la valeur d'occurrence d'un objet. Pour créer une référence à une valeur d'occurrence, placez des crochets ( [ ] ) après un nom d'objet et insérez entre les crochets l'une des valeurs suivantes :</p> <ul style="list-style-type: none"><li>• [ n ], où n est un numéro d'index d'occurrence absolu commençant à 0. Un numéro d'occurrence hors limite ne renvoie aucune valeur. Par exemple : <pre>xfa.form.form1.#subform.Quantity[3]</pre>refers to the fourth occurrence of the Quantity object.</li><li>• [ +/- n ], où n indique une occurrence relative à l'occurrence de l'objet porteur de la référence. Les valeurs positives retournent des nombres d'occurrences plus élevés alors que les valeurs négatives retournent des nombres d'occurrences plus bas. Par exemple, <pre>xfa.form.form1.#subform.Quantity[+2]</pre><p>Cette référence retourne l'occurrence de Quantity dont le nombre d'occurrences est deux de plus que le nombre d'occurrences du conteneur qui constitue la référence. Si, par exemple, cette référence était jointe à l'objet Quantity[2], la référence serait la même que <pre>xfa.template.Quantity[4]</pre></p><p>Si le numéro d'index calculé est hors limite, la référence renvoie une erreur.</p><p>Cette syntaxe sert surtout à localiser l'occurrence précédente ou suivante d'un objet en particulier. Par exemple, chaque occurrence de l'objet Quantity (sauf la première) peut utiliser Quantity[-1] pour obtenir la valeur de l'objet Quantity précédent.</p></li></ul> <li>• [*] indique plusieurs occurrences d'un objet. Le premier objet nommé est trouvé, et les objets portant le même nom et ayant le même parent immédiat sont retournés. Le résultat de cette notation est une collection d'objets. Par exemple, <pre>xfa.form.form1.#subform.Quantity[*]</pre><p>Cette expression fait référence à tous les objets portant le nom Quantity et ayant le même parent immédiat que la première occurrence Quantity trouvée par la référence.</p></li> <p><b>Remarque :</b> dans les formulaires en arabe, hébreu, thaï et vietnamien, la syntaxe de référence se trouve toujours à droite (même pour les langues s'écrivant de droite à gauche).</p>



Notation	Description
<p>[ ] (Suite)</p>	<div style="text-align: center;"> </div> <p>L'arborescence servant de référence, les expressions suivantes retournent les résultats indiqués :</p> <ul style="list-style-type: none"> <li>• <code>Subform_Page.Subform1 [*]</code> renvoie les deux objets <code>Subform1</code>.</li> <li>• <code>Subform_Page.Subform1.Subform3.TextField2 [*]</code> renvoie les deux objets <code>TextField2</code>. <code>Subform_Page.Subform1</code> a pour résultat le premier objet <code>Subform1</code> à gauche et <code>TextField2 [*]</code> est évalué relativement à l'objet <code>Subform3</code>.</li> <li>• <code>Subform_Page.Subform1 [*].TextField1</code> renvoie les deux instances <code>TextField1</code>. <code>Subform_Page.Subform1 [*]</code> a pour résultat les deux objets <code>Subform1</code> et <code>TextField1</code> est évalué relativement aux objets <code>Subform1</code>.</li> <li>• <code>Subform_Page.Subform1 [*].Subform3.TextField2 [1]</code> renvoie le deuxième et le quatrième objets <code>TextField2</code> à partir de la gauche. <code>Subform_Page.Subform1 [*]</code> a pour résultat les deux objets <code>Subform1</code> et <code>TextField2 [1]</code> est évalué relativement aux objets <code>Subform3</code>.</li> <li>• <code>Subform_Page.Subform1 [*].Subform3 [*]</code> renvoie les deux instances de l'objet <code>Subform3</code>.</li> <li>• <code>Subform_Page.*</code> renvoie les deux objets <code>Subform1</code> et l'objet <code>Subform2</code>.</li> <li>• <code>Subform_Page.Subform2.*</code> renvoie les deux instances de l'objet <code>NumericField2</code>.</li> </ul> <p><b>Remarque :</b> l'utilisation des crochets ([ ]) est admise dans le langage JavaScript avec la méthode <a href="#">resolveNode</a> <code>resolveNode</code>. For more information about the <code>resolveNode</code> method, see <i>LiveCycle Designer ES Help</i>, or see <i>LiveCycle Designer ES Scripting Reference</i> at <a href="http://www.adobe.com/go/learn_lc_scriptingReference">http://www.adobe.com/go/learn_lc_scriptingReference</a>.</p>

## Appel des propriétés et des méthodes

LiveCycle Designer ES définit différentes méthodes et propriétés pour tous les objets de conception de formulaire. FormCalc donne accès à ces propriétés et méthodes et permet de les utiliser pour changer l'aspect et le comportement des objets sur votre formulaire. Tout comme l'appel de fonction, l'appel de méthode et de propriété se fait par la transmission d'arguments selon un ordre spécifique. Le nombre et le type d'arguments de chaque méthode et propriété sont propres à chaque type d'objet.

**Remarque :** en effet, différents objets de conception de formulaire correspondent à différentes méthodes et propriétés. Pour une liste complète des propriétés et méthodes prises en charge par les

objets, voir la *source de référence sur les scripts LiveCycle Designer ES* à l'adresse [http://www.adobe.com/go/learn\\_lc\\_scriptingReference\\_fr](http://www.adobe.com/go/learn_lc_scriptingReference_fr).

## Appels de fonctions intégrées

FormCalc prend en charge un vaste ensemble de fonctions intégrées avec une multitude de possibilités. Le nom de ces fonctions ne fait pas la distinction entre les majuscules et les minuscules, mais contrairement aux mots-clés, FormCalc ne réserve pas le nom des fonctions. Autrement dit, dans les formulaires, les calculs d'objets dont les noms coïncident avec ceux des fonctions FormCalc ne génèrent pas de conflits.

Selon la fonction utilisée, vous pouvez être amené à indiquer un ensemble d'arguments pour qu'une valeur soit renvoyée. Beaucoup de fonctions comportent des arguments facultatifs : selon la situation, vous devez décider des arguments à utiliser.

FormCalc évalue tous les arguments de fonction dans l'ordre, de gauche à droite. Si vous tentez de transmettre à une fonction un nombre d'arguments inférieur à celui requis, la fonction génère une exception d'erreur.

Chaque fonction exige que ses arguments soient entrés selon un format précis, sous la forme d'un littéral numérique ou d'un littéral chaîne. Si la valeur d'un argument ne correspond pas à ce qu'attend la fonction, FormCalc convertit la valeur. Par exemple :

```
Len (35)
```

La fonction [Len](#) exige un littéral chaîne. Dans ce cas, FormCalc convertit l'argument, du nombre 35 à la chaîne « 35 », et la fonction retourne 2.

Cependant, la conversion d'un littéral chaîne en littéral numérique est plus complexe. Par exemple :

```
Abs ("abc")
```

La fonction [Abs](#) exige un littéral numérique. FormCalc convertit la valeur de tous les littéraux chaînes à la valeur 0. Cette conversion peut générer des problèmes dans les fonctions où la valeur 0 entraîne une erreur, comme dans le cas de la fonction [Apr](#).

Les arguments de certaines fonctions doivent obligatoirement être des entiers. Dans ce cas, les arguments transmis sont toujours convertis en entiers, la partie fractionnaire étant tronquée.

Voici le résumé des principales propriétés des fonctions intégrées :

- Le nom des fonctions intégrées ne fait pas la distinction entre les minuscules et les majuscules.
- Même si les fonctions intégrées sont prédéfinies, leurs noms ne constituent pas des mots réservés. Autrement dit, la fonction intégrée [Max](#), par exemple, n'entre jamais en conflit avec un objet, une propriété d'objet ou une méthode d'objet nommée.
- Beaucoup de fonctions intégrées exigent la saisie d'un certain nombre d'arguments, parfois suivis d'arguments facultatifs.
- Quelques fonctions intégrées, telles que [Avg](#), [Count](#), [Max](#), [Min](#), [Sum](#) et [Concat](#), acceptent un nombre indéfini d'arguments.

Pour consulter la liste complète des fonctions FormCalc, voir « [Liste alphabétique des fonctions](#) » à la [page 35](#).

# 3

## Liste alphabétique des fonctions

Le tableau suivant contient toutes les fonctions FormCalc disponibles, la description de chaque fonction et la catégorie à laquelle elle appartient.

Fonction	Description	Type
<a href="#">« Abs » à la page 39</a>	Retourne la valeur absolue d'une valeur ou d'une expression numérique.	Arithmétique
<a href="#">« Apr » à la page 67</a>	Revoie le taux annuel d'un prêt.	Financière
<a href="#">« At » à la page 87</a>	Repère la position du premier caractère d'une chaîne dans une autre chaîne.	Chaîne
<a href="#">« Avg » à la page 40</a>	Evalue un ensemble de valeurs et/ou d'expressions numériques et renvoie la moyenne des éléments non nuls de cet ensemble.	Arithmétique
<a href="#">« Ceil » à la page 40</a>	Revoie le nombre entier supérieur ou égal au nombre fourni.	Arithmétique
<a href="#">« Choose » à la page 78</a>	Sélectionne une valeur dans un ensemble de paramètres donné.	Logique
<a href="#">« Concat » à la page 87</a>	Revoie la concaténation des chaînes indiquées.	Chaîne
<a href="#">« Count » à la page 41</a>	Evalue un ensemble de valeurs et/ou d'expressions et renvoie le nombre d'éléments non nuls contenus dans cet ensemble.	Arithmétique
<a href="#">« CTerm » à la page 68</a>	Revoie le nombre de périodes nécessaires pour qu'un investissement, dont le taux d'intérêt est fixe mais composé, atteigne une valeur capitalisée.	Financière
<a href="#">« Date » à la page 57</a>	Revoie la date active du système sous la forme du nombre de jours écoulés depuis l' <a href="#">époque</a> .	Date et heure
<a href="#">« Date2Num » à la page 57</a>	Revoie le nombre de jours écoulés depuis l' <a href="#">époque</a> , à partir d'une chaîne de date.	Date et heure
<a href="#">« DateFmt » à la page 58</a>	Revoie une chaîne de format de date, à partir d'un style de format de date.	Date et heure
<a href="#">« Decode » à la page 88</a>	Revoie la version décodée d'une chaîne donnée.	Chaîne
<a href="#">« Encode » à la page 89</a>	Revoie la version codée d'une chaîne donnée.	Chaîne
<a href="#">« Eval » à la page 82</a>	Revoie la valeur d'un calcul de formulaire donné.	Divers
<a href="#">« Exists » à la page 79</a>	Détermine si le paramètre donné est une syntaxe de référence à un objet existant.	Logique
<a href="#">« Floor » à la page 41</a>	Revoie le plus grand nombre entier inférieur ou égal à la valeur donnée.	Arithmétique

<b>Fonction</b>	<b>Description</b>	<b>Type</b>
<a href="#">« Format » à la page 90</a>	Formate les données fournies selon la chaîne de format d'image indiquée.	Chaîne
<a href="#">« FV » à la page 69</a>	Renvoie la valeur capitalisée de paiements fixes effectués à intervalles réguliers, compte tenu d'un taux d'intérêt fixe.	Financière
<a href="#">« Get » à la page 102</a>	Télécharge le contenu de l'adresse URL indiquée.	URL
<a href="#">« HasValue » à la page 80</a>	Détermine si le paramètre indiqué est un mécanisme d'accès dont la valeur n'est ni nulle, ni vide, ni un blanc.	Logique
<a href="#">« IPmt » à la page 70</a>	Renvoie le montant de l'intérêt payé pour un prêt au cours d'une période donnée.	Financière
<a href="#">« IsoDate2Num » à la page 59</a>	Renvoie le nombre de jours écoulés depuis l' <a href="#">époque</a> , à partir d'une chaîne de date valide.	Date et heure
<a href="#">« IsoTime2Num » à la page 59</a>	Renvoie le nombre de millisecondes écoulées depuis l' <a href="#">époque</a> , à partir d'une chaîne d'heure valide.	Date et heure
<a href="#">« Left » à la page 91</a>	Extrait un nombre spécifique de caractères d'une chaîne, en commençant par le premier caractère à gauche.	Chaîne
<a href="#">« Len » à la page 91</a>	Renvoie le nombre de caractères d'une chaîne donnée.	Chaîne
<a href="#">« LocalDateFmt » à la page 60</a>	Renvoie une chaîne de format de date localisée, à partir d'un style de format de date.	Date et heure
<a href="#">« LocalTimeFmt » à la page 60</a>	Renvoie une chaîne de format d'heure localisée, à partir d'un style de format d'heure.	Date et heure
<a href="#">« Lower » à la page 92</a>	Convertit en minuscules toutes les majuscules d'une chaîne donnée.	Chaîne
<a href="#">« Ltrim » à la page 93</a>	Renvoie une chaîne sans aucun caractère d'espace blanc à gauche.	Chaîne
<a href="#">« Max » à la page 42</a>	Renvoie la valeur maximale des éléments non nuls d'un ensemble donné de nombres.	Arithmétique
<a href="#">« Min » à la page 43</a>	Renvoie la valeur minimale des éléments non nuls d'un ensemble donné de nombres.	Arithmétique
<a href="#">« Mod » à la page 44</a>	Renvoie le reste d'un nombre divisé par un autre.	Arithmétique
<a href="#">« NPV » à la page 71</a>	Renvoie la valeur nette actualisée d'un investissement, compte tenu d'une série de mouvements futurs de l'encaisse et d'un taux d'actualisation.	Financière
<a href="#">« Null » à la page 83</a>	Renvoie la valeur nulle. La valeur nulle signifie aucune valeur.	Divers
<a href="#">« Num2Date » à la page 61</a>	Renvoie une chaîne de date, à partir du nombre de jours écoulés depuis l' <a href="#">époque</a> .	Date et heure

<b>Fonction</b>	<b>Description</b>	<b>Type</b>
<a href="#">« Num2GMTTime » à la page 62</a>	Renvoie une chaîne d'heure UT, à partir du nombre de millisecondes écoulées depuis l' <a href="#">époque</a> .	Date et heure
<a href="#">« Num2Time » à la page 63</a>	Renvoie une chaîne d'heure, à partir du nombre de millisecondes écoulées depuis l' <a href="#">époque</a> .	Date et heure
<a href="#">« Oneof » à la page 80</a>	Retourne vrai (1) si une valeur se trouve dans un ensemble donné, et faux (0) dans le cas contraire.	Logique
<a href="#">« Parse » à la page 93</a>	Analyse les données fournies selon le format d'image donné.	Chaîne
<a href="#">« Pmt » à la page 72</a>	Renvoie le remboursement d'un prêt basé sur des versements fixes et sur un taux d'intérêt fixe.	Financière
<a href="#">« Post » à la page 103</a>	Place à l'adresse URL indiquée les données fournies.	URL
<a href="#">« PPmt » à la page 73</a>	Renvoie le montant du principal payé pour un prêt au cours d'une période.	Financière
<a href="#">« Put » à la page 104</a>	Télécharge les données fournies vers l'adresse URL indiquée.	URL
<a href="#">« PV » à la page 74</a>	Renvoie la valeur actualisée d'un investissement réalisé par versements fixes périodiques avec un taux d'intérêt fixe.	Financière
<a href="#">« Rate » à la page 75</a>	Renvoie le taux d'intérêt composé par période nécessaire pour qu'un investissement d'une valeur actualisée atteigne une valeur capitalisée au cours d'une période donnée.	Financière
<a href="#">« Ref » à la page 83</a>	Renvoie une référence à un objet existant.	Divers
<a href="#">« Replace » à la page 94</a>	Remplace toutes les occurrences d'une chaîne par une autre, dans la chaîne indiquée.	Chaîne
<a href="#">« Right » à la page 95</a>	Extrait un nombre spécifique de caractères dans une chaîne donnée, en commençant par le dernier caractère à droite.	Chaîne
<a href="#">« Round » à la page 45</a>	Evalue une valeur ou expression numérique donnée et renvoie un nombre arrondi comportant le nombre de décimales défini.	Arithmétique
<a href="#">« Rtrim » à la page 95</a>	Renvoie une chaîne sans aucun caractère d'espace blanc à droite.	Chaîne
<a href="#">« Space » à la page 96</a>	Renvoie une chaîne composée d'un nombre donné d'espaces blancs.	Chaîne
<a href="#">« Str » à la page 96</a>	Convertit un nombre en une chaîne de caractères. FormCalc formate le résultat selon la largeur indiquée et l'arrondit au nombre de décimales défini.	Chaîne
<a href="#">« Stuff » à la page 97</a>	Insère une chaîne dans une autre chaîne.	Chaîne

<b>Fonction</b>	<b>Description</b>	<b>Type</b>
<a href="#">« Substr » à la page 98</a>	Extrait une partie d'une chaîne donnée.	Chaîne
<a href="#">« Sum » à la page 46</a>	Renvoie la somme des éléments non nuls d'un ensemble donné de nombres.	Arithmétique
<a href="#">« Term » à la page 76</a>	Renvoie le nombre de périodes nécessaires pour atteindre la valeur capitalisée donnée, compte tenu de paiements fixes périodiques versés sur un compte portant intérêt.	Financière
<a href="#">« Time » à la page 64</a>	Renvoie l'heure actuelle du système sous la forme du nombre de millisecondes écoulées depuis l' <a href="#">époque</a> .	Date et heure
<a href="#">« Time2Num » à la page 65</a>	Renvoie le nombre de millisecondes écoulées depuis l' <a href="#">époque</a> , à partir d'une chaîne d'heure.	Date et heure
<a href="#">« TimeFmt » à la page 66</a>	Renvoie un format d'heure, à partir d'un style de format d'heure.	Date et heure
<a href="#">« UnitType » à la page 84</a>	Renvoie les unités d'une étendue d'unité. Une étendue d'unité est une chaîne composée d'un nombre suivi d'un nom d'unité.	Divers
<a href="#">« UnitValue » à la page 85</a>	Retourne la valeur numérique d'une mesure avec l'étendue d'unité qui lui est associée, après une conversion d'unité facultative.	Divers
<a href="#">« Upper » à la page 99</a>	Convertit en majuscules toutes les minuscules d'une chaîne.	Chaîne
<a href="#">« Uuid » à la page 99</a>	Renvoie une chaîne UUID qui sert de méthode d'identification.	Chaîne
<a href="#">« Within » à la page 81</a>	Retourne vrai (1) si une valeur de contrôle se trouve dans une plage donnée, et faux (0) dans le cas contraire.	Logique
<a href="#">« WordNum » à la page 100</a>	Renvoie le texte équivalent à un nombre donné.	Chaîne

# 4 Fonctions arithmétiques

## A propos des fonctions arithmétiques

Ces fonctions exécutent différentes opérations mathématiques.

### Fonctions

- [« Abs » à la page 39](#)
- [« Avg » à la page 40](#)
- [« Ceil » à la page 40](#)
- [« Count » à la page 41](#)
- [« Floor » à la page 41](#)
- [« Max » à la page 42](#)
- [« Min » à la page 43](#)
- [« Mod » à la page 44](#)
- [« Round » à la page 45](#)
- [« Sum » à la page 46](#)

## Abs

Revoie la valeur absolue d'une valeur ou expression numérique, ou renvoie « nul » si la valeur ou expression est nulle.

### Syntaxe

Abs (n1)

### Paramètres

Paramètre	Description
n1	Valeur ou expression numérique à évaluer.

**Remarque :** FormCalc respecte la norme internationale IEEE-754 en matière de gestion des valeurs numériques à virgule flottante. Pour plus de détails, voir [« Littéraux numériques » à la page 12.](#)

### Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction Abs :

Expression	Renvoie
Abs (1.03)	1.03
Abs (-1.03)	1.03
Abs (0)	0

## Avg

Évalue un ensemble de valeurs et/ou d'expressions numériques et renvoie la moyenne des éléments non nuls de cet ensemble.

### Syntaxe

`Avg(n1 [, n2 ...])`

### Paramètres

Paramètre	Description
n1	Première valeur ou expression numérique de l'ensemble.
n2 (facultatif)	Autres valeurs ou expressions numériques.

**Remarque :** FormCalc respecte la norme internationale IEEE-754 en matière de gestion des valeurs numériques à virgule flottante. Pour plus de détails, voir « [Littéraux numériques](#) » à la [page 12](#).

### Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction Avg :

Expression	Renvoie
<code>Avg(0, 32, 16)</code>	16
<code>Avg(2.5, 17, null)</code>	9.75
<code>Avg(Price[0], Price[1], Price[2], Price[3])</code>	Moyenne des quatre premières occurrences non nulles de <code>Price</code> .
<code>Avg(Quantity[*])</code>	Moyenne de toutes les occurrences non nulles de <code>Quantity</code> .

## Ceil

Renvoie le nombre entier supérieur ou égal au nombre fourni, ou renvoie « nul » si le paramètre est nul.

### Syntaxe

`Ceil(n)`

### Paramètres

Paramètre	Description
n	N'importe quelle valeur ou expression numérique. La fonction renvoie la valeur 0 si n n'est pas une valeur ou expression numérique.

**Remarque :** FormCalc respecte la norme internationale IEEE-754 en matière de gestion des valeurs numériques à virgule flottante. Pour plus de détails, voir « [Littéraux numériques](#) » à la [page 12](#).



## Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction `Ceil` :

Expression	Renvoie
<code>Ceil (2.5875)</code>	3
<code>Ceil (-5.9)</code>	-5
<code>Ceil ("abc")</code>	0
<code>Ceil (A)</code>	100 si la valeur de A est égale à 99,999.

## Count

Evalue un ensemble de valeurs et/ou expressions, et renvoie le nombre d'éléments non nuls contenus dans cet ensemble.

### Syntaxe

`Count (n1 [, n2 ...])`

### Paramètres

Paramètre	Description
n1	Valeur ou expression numérique.
n2 (facultatif)	Autres valeurs et/ou expressions numériques.

**Remarque :** FormCalc respecte la norme internationale IEEE-754 en matière de gestion des valeurs numériques à virgule flottante. Pour plus de détails, voir « [Littéraux numériques](#) » à la [page 12](#).

### Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction `Count` :

Expression	Renvoie
<code>Count ("Tony", "Blue", 41)</code>	3
<code>Count (Customers [*])</code>	Nombre d'occurrences non nulles de Customers.
<code>Count (Coverage [2], "Home", "Auto")</code>	3, pourvu que la troisième occurrence de Coverage ne soit pas nulle.

## Floor

Renvoie le plus grand nombre entier inférieur ou égal à la valeur donnée.

### Syntaxe

`Floor (n)`

### Paramètres

Paramètre	Description
n	N'importe quelle valeur ou expression numérique.

**Remarque :** FormCalc respecte la norme internationale IEEE-754 en matière de gestion des valeurs numériques à virgule flottante. Pour plus de détails, voir « [Littéraux numériques](#) » à la [page 12](#).

### Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction `Floor` :

Expression	Renvoie
<code>Floor(21.3409873)</code>	21
<code>Floor(5.999965342)</code>	5
<code>Floor(3.2 * 15)</code>	48

## Max

Renvoie la valeur maximale des éléments non nuls d'un ensemble donné de nombres.

### Syntaxe

`Max(n1 [, n2 ...])`

### Paramètres

Paramètre	Description
n1	Valeur ou expression numérique.
n2 (facultatif)	Autres valeurs et/ou expressions numériques.

**Remarque :** FormCalc respecte la norme internationale IEEE-754 en matière de gestion des valeurs numériques à virgule flottante. Pour plus de détails, voir « [Littéraux numériques](#) » à la [page 12](#).

### Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction `Max` :

Expression	Renvoie
<code>Max(234, 15, 107)</code>	234
<code>Max("abc", 15, "Tony Blue")</code>	15
<code>Max("abc")</code>	0

Expression	Renvoie
<code>Max(Field1[*], Field2[0])</code>	Évalue les occurrences non nulles de <code>Field1</code> ainsi que la première occurrence de <code>Field2</code> , et renvoie la valeur la plus élevée.
<code>Max(Min(Field1[*], Field2[0]), Field3, Field4)</code>	La première expression évalue les occurrences non nulles de <code>Field1</code> ainsi que la première occurrence de <code>Field2</code> , et renvoie la valeur la plus basse. Le résultat final est le maximum de cette valeur comparativement aux valeurs de <code>Field3</code> et de <code>Field4</code> .  Voir aussi « <a href="#">Min</a> » à la page 43.

## Min

Renvoie la valeur minimale des éléments non nuls d'un ensemble donné de nombres.

### Syntaxe

`Min(n1 [, n2 ...])`

### Paramètres

Paramètre	Description
<code>n1</code>	Valeur ou expression numérique.
<code>n2</code> (facultatif)	Autres valeurs et/ou expressions numériques.

**Remarque :** FormCalc respecte la norme internationale IEEE-754 en matière de gestion des valeurs numériques à virgule flottante. Pour plus de détails, voir « [Littéraux numériques](#) » à la page 12.

### Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction `Min` :

Expression	Renvoie
<code>Min(234, 15, 107)</code>	15
<code>Min("abc", 15, "Tony Blue")</code>	15
<code>Min("abc")</code>	0

Expression	Renvoie
<code>Min(Field1[*], Field2[0])</code>	Évalue les occurrences non nulles de <code>Sales_July</code> ainsi que la première occurrence de <code>Sales_August</code> , et renvoie la valeur la plus basse.
<code>Min(Max(Field1[*], Field2[0]), Field3, Field4)</code>	La première expression évalue les occurrences non nulles de <code>Field1</code> ainsi que la première occurrence de <code>Field2</code> , et renvoie la valeur la plus élevée. Le résultat final est le minimum de cette valeur comparativement aux valeurs de <code>Field3</code> et de <code>Field4</code> .  Voir aussi « <a href="#">Max</a> » à la page 42.

## Mod

Renvoie le reste d'un nombre divisé par un autre. Il s'agit du reste de la division du dividende par le diviseur. Le signe du reste est toujours le même que celui du dividende.

### Syntaxe

`Mod(n1, n2)`

### Paramètres

Paramètre	Description
<code>n1</code>	Le dividende : valeur ou expression numérique.
<code>n2</code>	Le diviseur : valeur ou expression numérique.

Si `n1` et/ou `n2` ne sont pas des valeurs ou des expressions numériques, la fonction renvoie 0.

**Remarque :** FormCalc respecte la norme internationale IEEE-754 en matière de gestion des valeurs numériques à virgule flottante. Pour plus de détails, voir « [Littéraux numériques](#) » à la page 12.

### Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction `Mod` :

Expression	Renvoie
<code>Mod(64, -3)</code>	1
<code>Mod(-13, 3)</code>	-1
<code>Mod("abc", 2)</code>	0

Expression	Renvoi
Mod (X [0] , Y [9] )	La première occurrence de X sert de dividende et la dixième occurrence de Y est utilisée comme diviseur.
Mod (Round (Value [4] , 2) , Max (Value [*] ) )	La cinquième occurrence de Value arrondie à deux décimales sert de dividende et la valeur la plus élevée parmi toutes les occurrences non nulles de Value est utilisée comme diviseur.  Voir aussi « <a href="#">Max</a> » à la page 42 et « <a href="#">Round</a> » à la page 45.

## Round

Évalue une valeur ou expression numérique donnée et renvoie un nombre arrondi comportant un nombre de décimales défini.

### Syntaxe

Round (n1 [, n2] )

### Paramètres

Paramètre	Description
n1	Valeur ou expression numérique à évaluer.
n2 (facultatif)	Nombre de décimales avec lesquelles n1 est évalué (maximum de 12). Si vous omettez la valeur n2, ou si n2 n'est pas valide, la fonction ne retournera aucune décimale.

**Remarque :** FormCalc respecte la norme internationale IEEE-754 en matière de gestion des valeurs numériques à virgule flottante. Pour plus de détails, voir « [Littéraux numériques](#) » à la page 12.

### Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction Round :

Expression	Renvoi
Round (12.389764537, 4)	12.3898
Round (20/3, 2)	6.67
Round (8.9897, "abc")	9
Round (FV (400, 0.10/12, 30*12), 2)	904195.17. Cette fonction prend la valeur obtenue par la fonction FV et l'arrondit à deux décimales.  Voir aussi « <a href="#">FV</a> » à la page 69.
Round (Total_Price, 2)	Arrondit la valeur de Total_Price à deux décimales.

## Sum

Renvoie la somme des éléments non nuls d'un ensemble donné de nombres.

### Syntaxe

`Sum(n1 [, n2 ...])`

### Paramètres

Paramètre	Description
n1	Valeur ou expression numérique.
n2 (facultatif)	Autres valeurs et/ou expressions numériques.

**Remarque :** FormCalc respecte la norme internationale IEEE-754 en matière de gestion des valeurs numériques à virgule flottante. Pour plus de détails, voir [« Littéraux numériques » à la page 12](#).

### Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction Sum :

Expression	Renvoie
<code>Sum(2, 4, 6, 8)</code>	20
<code>Sum(-2, 4, -6, 8)</code>	4
<code>Sum(4, 16, "abc", 19)</code>	39
<code>Sum(Amount [2], Amount [5])</code>	Totaux des troisième et sixième occurrences de Amount.
<code>Sum(Round(20/3, 2), Max(Amount [*]), Min(Amount [*]))</code>	Effectue le total de la valeur de 20/3 arrondie à deux décimales, et des occurrences non nulles les plus et moins élevées de Amount.  Voir aussi <a href="#">« Max » à la page 42</a> , <a href="#">« Min » à la page 43</a> et <a href="#">« Round » à la page 45</a> .

## A propos des fonctions de date et d'heure

Les fonctions décrites dans la présente section portent spécifiquement sur la création et la gestion des valeurs de date et d'heure.

### Fonctions

- [« Date » à la page 57](#)
- [« Date2Num » à la page 57](#)
- [« DateFmt » à la page 58](#)
- [« IsoDate2Num » à la page 59](#)
- [« IsoTime2Num » à la page 59](#)
- [« LocalDateFmt » à la page 60](#)
- [« LocalTimeFmt » à la page 60](#)
- [« Num2Date » à la page 61](#)
- [« Num2GMTTime » à la page 62](#)
- [« Num2Time » à la page 63](#)
- [« Time » à la page 64](#)
- [« Time2Num » à la page 65](#)
- [« TimeFmt » à la page 66](#)

## Structuration des dates et des heures

### Paramètres régionaux

Un *paramètre régional* est un terme standard utilisé, dans le développement de normes internationales, pour identifier une nation

(une langue, un pays ou une région). En ce qui concerne FormCalc, un *paramètre régional* définit le format de date et d'heure ainsi que le format numérique et monétaire d'une nation ou d'une région, pour permettre aux utilisateurs de se servir des formats auxquels ils sont habitués.

Chaque paramètre régional comprend une chaîne de caractères unique appelée *identificateur de paramètre régional*. La composition de ces chaînes est gérée par le groupe IETF (Internet Engineering Task Force) de l'Organisation mondiale de normalisation (ISO), organe de la Société Internet ([www.isoc.org](http://www.isoc.org)).

Les identificateurs de paramètre régional se composent d'une partie langue, d'une partie pays ou région ou des deux. Le tableau suivant présente la liste des paramètres régionaux valides avec la présente version de LiveCycle Designer ES.

<b>Langue</b>	<b>Pays ou région</b>	<b>Code ISO</b>
Allemand	Allemagne	de_DE
Allemand	Autriche	de_AT
Allemand	Luxembourg	de_LU
Allemand	Suisse	de_CH
Anglais	Afrique du Sud	en_ZA
Anglais	Anglais (Etats-Unis)	en_US
Anglais	Australie	en_AU
Anglais	Canada	en_CA
Anglais	Inde	en_IN
Anglais	Irlande	en_IE
Anglais	Nouvelle-Zélande	en_NZ
Anglais	Royaume-Uni	en_GB
Anglais	Royaume-Uni (Europe)	en_GB_EURO
Arabe	Algérie	ar_DZ
Arabe	Arabie Saoudite	ar_SA
Arabe	Bahreïn	ar_BH
Arabe	Egypte	ar_EG
Arabe	Emirats arabes unis	ar_AE
Arabe	Iraq	ar_IQ
Arabe	Jordanie	ar_JO
Arabe	Koweït	ar_KW
Arabe	Liban	ar_LB
Arabe	Libye	ar_LY
Arabe	Maroc	ar_MA
Arabe	Oman	ar_OM
Arabe	Qatar	ar_QA
Arabe	Soudan	ar_SD
Arabe	Syrie	ar_SY
Arabe	Tunisie	ar_TN
Arabe	Yémen	ar_YE



<b>Langue</b>	<b>Pays ou région</b>	<b>Code ISO</b>
Bulgare	Bulgarie	bg_BG
Chinois	Chinois simplifié (République populaire de Chine)	zh_CN
Chinois	Chinois traditionnel	zh_TW
Chinois	Hong Kong	zh_HK
Coréen	Corée (caractères Hanja)	ko_KR_HANI
Coréen	République de Corée	ko_KR
Croate	Croatie	hr_HR
Danois	Danemark	da_DK
Espagnol	Argentine	es_AR
Espagnol	Bolivie	es_BO
Espagnol	Chili	es_CL
Espagnol	Colombie	es_CO
Espagnol	Costa Rica	es_CR
Espagnol	El Salvador	es_SV
Espagnol	Equateur	es_EC
Espagnol	Espagne	es_ES
Espagnol	Guatemala	es_GT
Espagnol	Honduras	es_HN
Espagnol	Mexique	es_MX
Espagnol	Nicaragua	es_NI
Espagnol	Panama	es_PA
Espagnol	Paraguay	es_PY
Espagnol	Pérou	es_PE
Espagnol	Porto Rico	es_PR
Espagnol	République dominicaine	es_DO
Espagnol	Uruguay	es_UY
Espagnol	Venezuela	es_VE
Estonien	Estonie	et_EE
Finnois	Finlande	fi_FI
Français	Belgique	fr_BE
Français	Canada	fr_CA

<b>Langue</b>	<b>Pays ou région</b>	<b>Code ISO</b>
Français	France	fr_FR
Français	Luxembourg	fr_LU
Français	Suisse	fr_CH
Grec	Grèce	el_GR
Hébreu	Israël	he_IL
Hongrois	Hongrie	hu_HU
Indonésien	Indonésie	id_ID
Italien	Italie	it_IT
Italien	Suisse	it_CH
Japonais	Japon	ja_JP
Letton	Lettonie	lv_LV
Lituanien	Lituanie	lt_LT
Malais	Malaisie	ms_MY
Néerlandais	Belgique	nl_BE
Néerlandais	Pays-Bas	nl_NL
Norvégien (Bokmal)	Norvège	nb_NO
Norvégien (Nynorsk)	Norvège	nn_NO
Polonais	Pologne	pl_PL
Portugais	Brésil	pt_BR
Portugais	Portugal	pt_PT
Roumain	Roumanie	ro_RO
Russe	Russie	ru_RU
Serbo-croate	Bosnie-Herzégovine	sh_BA
Serbo-croate	Croatie	sh_HR
Serbo-croate	Serbie-et-Monténégro	sh_CS
Slovaque	Slovaquie	sk_SK
Slovène	Slovénie	sl_SI
Suédois	Suède	sv_SE
Tchèque	République tchèque	cs_CAZ
Thaï	Royaume de Thaïlande	th_TH_TH
Thaï	Thaïlande	th_TH

Langue	Pays ou région	Code ISO
Turc	Turquie	tr_TR
Ukrainien	Ukraine	uk_UA
Vietnamien	Vietnam	vi_VN

Habituellement, les deux éléments d'un paramètre régional sont importants. Par exemple, le nom des jours de la semaine et des mois en anglais pour le Canada et en anglais pour la Grande-Bretagne ont le même format, mais les dates sont formatées différemment. Ainsi, l'indication de la langue anglaise comme paramètre régional n'est pas suffisante. De même, l'indication du seul pays comme paramètre régional n'est pas suffisante. Par exemple, le Canada possède deux formats de date, l'un pour l'anglais et l'autre pour le français.

En général, chaque application fonctionne dans un environnement où se trouve un paramètre régional. Ce paramètre régional est *ambient*. Dans certaines circonstances, une application peut fonctionner sur un système, ou dans un environnement, dans lequel ne se trouve aucun paramètre régional. Dans ces rares cas, le paramètre régional *ambient* par défaut est l'anglais des Etats-Unis (en-US). Ce paramètre régional est celui *par défaut*.

## Epoque

On associe aux valeurs de date et d'heure une origine, dite *époque*, qui sert de point de départ chronologique. Toute valeur de date ou d'heure située avant cette époque n'est pas valide.

L'unité de valeur pour toutes les fonctions de date est le nombre de jours calculé depuis l'époque. L'unité de valeur pour toutes les fonctions d'heure est le nombre de millisecondes calculé depuis l'époque.

LiveCycle Designer ES définit le jour 1 de l'époque pour toutes les fonctions de date comme étant le 1er janvier 1900, et la milliseconde 1 de l'époque pour toutes les fonctions d'heure comme étant minuit, 00:00:00, en temps universel coordonné (UTC), aussi connu sous le nom de « temps moyen de Greenwich » (GMT). Cette définition signifie que les utilisateurs des fuseaux horaires situés à l'Est de l'UTC peuvent recevoir des valeurs d'heure négatives.

## Formats de date

Un *format de date* est une notation abrégée déterminant la façon dont la date s'affiche. Elle comprend différents signes de ponctuation et symboles représentant le formatage utilisé par la date. Le tableau suivant fournit des exemples de formats de date :

Format de date	Exemple
MM/JJ/AA	11/11/78
JJ/MM/AA	25/07/85
MMMM JJ, AAAA	Mars 10, 1964

Le format de date est géré par une norme ISO. Chaque pays ou région spécifie ses propres formats de date. Les quatre catégories générales de formats de date correspondent aux formats court, moyen, long et complet. Le tableau ci-dessous indique quelques exemples de formats de date, dans différents paramètres régionaux, pour chacune des quatre catégories.

Identificateur de paramètre régional et description	Format de date (catégorie)	Exemple
en_GB Anglais (Royaume-Uni)	JJ/MM/AA (Court)	08/18/92 08/04/05
fr_CA Français (Canada)	AA-MM-JJ (Moyen)	92-08-18
de_DE Allemand (Allemagne)	J. MMMM AAAA (Long)	17. Juni 1989
fr_FR Français (France)	EEEE, ' le ' J MMMM AAAA (Complet)	Lundi, le 29 Octobre, 1990

## Formats d'heure

Un *format d'heure* est une notation abrégée déterminant la façon dont l'heure est affichée. Elle comprend des signes de ponctuation, des littéraux et des symboles. Le tableau suivant propose des exemples de formats d'heure.

Format d'heure	Exemple
h:MM A	7:15 PM
HH:MM:SS	21:35:26
HH:MM:SS 'o'clock' A Z	14:20:10 o'clock PM EDT

Le format d'heure est géré par une norme ISO. Chaque nation définit ses formats d'heure par défaut, court, moyen, long, et complet. Le paramètre régional identifie le format d'heure conforme aux normes de cette nation.

Le tableau ci-dessous indique quelques exemples de formats d'heure, dans différents paramètres régionaux, pour chacune des quatre catégories.

Identificateur de paramètre régional et description	Format d'heure (Catégorie)	Exemple
en_GB Anglais (Royaume-Uni)	HH:MM (Court)	14:13
fr_CA Français (Canada)	HH:MM:SS (Moyen)	12:15:50
de_DE Allemand (Allemagne)	HH:MM:SS z (Long)	14:13:13 -0400
fr_FR Français (France)	HH ' h ' MM Z (Complet)	14 h 13 GMT-04:00

## Formats d'image de date et d'heure

Vous devez utiliser les symboles suivants pour créer des formats de date et heure pour les champs de date/heure. Certains symboles de dates sont uniquement utilisés dans les paramètres régionaux correspondant aux langues chinoise, japonaise et coréenne. Ces symboles sont également indiqués ci-dessous.

**Remarque :** la virgule (,), le tiret (-), le deux-points (:), la barre oblique (/), le point (.) et l'espace ( ) sont considérés comme des valeurs littérales qui peuvent être incluses n'importe où dans le format. Pour inclure du texte dans un format, délimitez la chaîne de texte par des guillemets simples (''). Par exemple, 'Le paiement est exigible au' MM-JJ-AA peut être spécifié comme format d'affichage.

<b>Symbole de date</b>	<b>Description</b>	<b>Valeur formatée pour un paramètre régional Anglais (Etats-Unis) dont la valeur d'entrée tributaire du format du paramètre régional est 1/1/08 (Janvier 1, 2008)</b>
D	Jour du mois à 1 ou 2 chiffres (1-31)	1
DD	Jour du mois à 2 chiffres avec zéro de remplissage (01-31)	01
J	Jour de l'année à 1, 2 ou 3 chiffres (1-366)	1
JJJ	Jour de l'année à 3 chiffres avec zéros de remplissage (001-366)	001
M	Mois de l'année à 1 ou 2 chiffres (1-12)	1
MM	Mois de l'année à 2 chiffres avec zéro de remplissage (01-12)	01
MMM	Nom abrégé du mois	Jan
MMMM	Nom complet du mois	Janvier
E	Jour de la semaine à 1 chiffre (1-7), où 1=Dimanche	3 (le 1er janvier 2008 correspond à un mardi)
EEE	Nom abrégé du jour de la semaine	Mar (le 1er janvier 2008 correspond à un mardi)
EEEE	Nom complet du jour de la semaine	Mardi (le 1er janvier 2008 correspond à un mardi)
YY	Année à 2 chiffres, où les nombres inférieurs à 30 se situent après l'an 2000 et où les nombres égaux et supérieurs à 30 se situent avant l'an 2000. Par exemple, 00=2000, 29=2029, 30=1930 et 99=1999.	08
YYYY	Année à 4 chiffres	2008

<b>Symbole de date</b>	<b>Description</b>	<b>Valeur formatée pour un paramètre régional Anglais (Etats-Unis) dont la valeur d'entrée tribulaire du format du paramètre régional est 1/1/08 (Janvier 1, 2008)</b>
G	Nom de l'ère (av. J.-C. ou ap. J.-C.)	ap. J.-C.
w	Semaine du mois à 1 chiffre (0-5), où la semaine 1 est le premier groupe de 4 jours qui se termine un samedi	1
WW	Semaine de l'année à 2 chiffres (01-53) ISO-8601, où la semaine 1 représente la semaine qui contient le 4 janvier	01

Vous disposez de plusieurs autres modèles de dates correspondant aux paramètres régionaux chinois, japonais et coréen.

Les ères japonaises sont représentables à l'aide de plusieurs symboles différents. Les quatre derniers symboles d'ère constituent d'autres symboles possibles pour représenter les ères japonaises.

<b>Symbole de date CJK</b>	<b>Description</b>
DDD	Jour du mois du paramètre régional, exprimé en valeur numérique idéographique
DDDD	Jour du mois du paramètre régional, exprimé en valeur numérique idéographique (règle des dixièmes)
YYY	Année du paramètre régional, exprimée en valeur numérique idéographique
YYYYY	Année du paramètre régional, exprimée en valeur numérique idéographique (règle des dixièmes)
g	Autre nom possible désignant l'ère du paramètre régional. Pour l'ère actuelle japonaise, Heisei, ce modèle affiche la lettre H ASCII (U+48)
gg	Autre nom possible désignant l'ère du paramètre régional. Pour l'ère actuelle japonaise, ce modèle affiche l'idéogramme représenté par le symbole Unicode (U+5E73)
ggg	Autre nom possible désignant l'ère du paramètre régional. Pour l'ère actuelle japonaise, ce modèle affiche les idéogrammes représentés par les symboles Unicode (U+5E73 U+6210)
g	Autre nom possible désignant l'ère du paramètre régional. Pour l'ère actuelle japonaise, ce modèle affiche la lettre H à pleine largeur (U+FF28)
g g	Autre nom possible désignant l'ère du paramètre régional. Pour l'ère actuelle japonaise, ce modèle affiche l'idéogramme représenté par le symbole Unicode (U+337B)

Symbole de l'heure	Description	Valeur d'entrée tributaire du format du paramètre régional	Valeur formatée du paramètre régional Anglais (Etats-Unis)
h	Heure au format du jour (AM/PM) à 1 ou à 2 chiffres (1-12)	12:08 AM ou 2:08 PM	12 ou 2
hh	Heure au format du jour (AM/PM) à 2 chiffres avec zéro de remplissage (01-12)	12:08 AM ou 2:08 PM	12 ou 02
k	Heure au format du jour (AM/PM) à 0 ou à 2 chiffres (1-11)	12:08 AM ou 2:08 PM	0 ou 2
kk	Heure au format du jour (AM/PM) à 2 chiffres (00-11)	12:08 AM ou 2:08 PM	00 ou 02
H	Heure du jour à 1 ou 2 chiffres (0-23)	12:08 AM ou 2:08 PM	0 ou 14
HH	Heure du jour à 2 chiffres avec zéro de remplissage (00-23)	12:08 AM ou 2:08 PM	00 ou 14
K	Heure du jour à 1 ou 2 chiffres (1-24)	12:08 AM ou 2:08 PM	24 ou 14
KK	Heure du jour à 2 chiffres avec zéro de remplissage (01-24)	12:08 AM ou 2:08 PM	24 ou 14
M	Minute de l'heure à 1 ou 2 chiffres (0-59)  <b>Remarque :</b> vous devez utiliser ce symbole dans un symbole d'heure.	2:08 PM	8
MM	Minute de l'heure à 2 chiffres avec zéro de remplissage (00-59)  <b>Remarque :</b> vous devez utiliser ce symbole dans un symbole d'heure.	2:08 PM	08
S	Seconde de la minute à 1 ou 2 chiffres (0-59)  <b>Remarque :</b> vous devez utiliser ce symbole dans un symbole d'heure et de minutes.	2:08:09 PM	9
SS	Seconde de la minute à 2 chiffres avec zéro de remplissage (00-59)  <b>Remarque :</b> vous devez utiliser ce symbole dans un symbole d'heure et de minutes.	2:08:09 PM	09

Symbole de l'heure	Description	Valeur d'entrée tributaire du format du paramètre régional	Valeur formatée du paramètre régional Anglais (Etats-Unis)
FFF	Millième de la seconde à 3 chiffres (000-999)  <b>Remarque :</b> vous devez utiliser ce symbole dans un symbole d'heure, de minute et de seconde.	2:08:09 PM	09
A	Partie de la journée comprise entre minuit et midi (AM) ou midi et minuit (PM)	2:08:09 PM	PM
z	Format de fuseau horaire ISO-8601 (par exemple, z, +0500, -0030, -01, +0100)  <b>Remarque :</b> vous devez utiliser ce symbole dans un symbole d'heure.	2:08:09 PM	-0400
zz	Format de fuseau horaire ISO-8601 secondaire (par exemple, z, +05:00, -00:30, -01, +01:00)  <b>Remarque :</b> vous devez utiliser ce symbole dans un symbole d'heure.	2:08:09 PM	-04:00
Z	Nom abrégé de fuseau horaire, par exemple, TMG, TMG+05:00, TMG-00:30, HNE, HAP  <b>Remarque :</b> vous devez utiliser ce symbole dans un symbole d'heure.	2:08:09 PM	EDT

### Symboles réservés

Les symboles qui suivent ont une signification particulière et ne peuvent pas être utilisés comme texte littéral.

Symbole	Description
?	Lorsqu'il est utilisé, ce symbole représente n'importe quel caractère. Lorsqu'il est fusionné pour l'affichage, il devient un espace.
*	Lorsqu'il est utilisé, ce symbole représente 0 ou le caractère d'espace Unicode. Lorsqu'il est fusionné pour l'affichage, il devient un espace.
+	Lorsqu'il est utilisé, ce symbole représente un ou plusieurs caractères d'espace Unicode. Lorsqu'il est fusionné pour l'affichage, il devient un espace.



## Date

Retourne la date active du système représentée sous forme du nombre de jours écoulés depuis la date considérée.

### Syntaxe

Date ( )

### Paramètres

Aucun

### Exemples

L'expression suivante constitue un exemple illustrant l'utilisation de la fonction Date :

Expression	Renvoie
Date ( )	37875 (le nombre de jours depuis une origine, dite époque, jusqu'au 12 septembre 2003)

## Date2Num

Retourne le nombre de jours écoulés depuis la date considérée, à partir d'une chaîne de date.

### Syntaxe

Date2Num ( d [, f [, k ] ] )

### Paramètres

Paramètre	Description
d	Chaîne de date respectant le format indiqué par f qui par ailleurs se conforme au paramètre régional indiqué par k.
f (facultatif)	Chaîne de format de date. Si f est omis, le format de date par défaut MMM D, YYYY est utilisé.
k (facultatif)	Chaîne d'identification du paramètre régional qui se conforme aux normes de dénomination de ce dernier. Si k est omis (ou n'est pas valide), le paramètre régional ambiant est utilisé.

La fonction renvoie la valeur 0 si l'une des conditions suivantes est vraie :

- Le format de la date fournie ne correspond pas au format indiqué dans la fonction.
- Le paramètre régional ou le format de date fourni dans la fonction n'est pas valide.

Les informations fournies sont insuffisantes pour déterminer un jour unique calculé depuis l'époque (autrement dit, une information relative à la date est manquante ou incomplète).

## Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction `Date2Num` :

Expression	Renvoi
<code>Date2Num("Mar 15, 1996")</code>	35138
<code>Date2Num("1/1/1900", "D/M/YYYY")</code>	1
<code>Date2Num("03/15/96", "MM/DD/YY")</code>	35138
<code>Date2Num("Aug 1,1996", "MMM D, YYYY")</code>	35277
<code>Date2Num("96-08-20", "YY-MM-DD", "fr_FR")</code>	35296
<code>Date2Num("1/3/00", "D/M/YY") - Date2Num("1/2/00", "D/M/YY")</code>	29

## DateFmt

Renvoie une chaîne de format de date, à partir d'un style de format de date.

### Syntaxe

`DateFmt([n [, k]])`

### Paramètres

Paramètre	Description
<code>n</code> (facultatif)	Entier qui identifie le style de format d'heure propre au paramètre régional, de la façon suivante : <ul style="list-style-type: none"> <li>● 1 (style court)</li> <li>● 2 (style moyen)</li> <li>● 3 (style long)</li> <li>● 4 (style complet)</li> </ul> Si <code>n</code> est omis (ou n'est pas valide), la valeur de style par défaut est 0.
<code>k</code> (facultatif)	Chaîne d'identification du paramètre régional qui se conforme aux normes de dénomination de ce dernier. Si <code>k</code> est omis (ou n'est pas valide), le paramètre régional ambiant est utilisé.

## Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction `DateFmt` :

Expression	Renvoi
<code>DateFmt(1)</code>	M/D/YY (si le paramètre régional <code>en_US</code> est défini)
<code>DateFmt(2, "fr_CA")</code>	YY-MM-DD
<code>DateFmt(3, "de_DE")</code>	D. MMMM YYYY
<code>DateFmt(4, "fr_FR")</code>	EEEE D' MMMM YYYY

## IsoDate2Num

Renvoie le nombre de jours écoulés depuis l'époque, à partir d'une chaîne de date valide.

### Syntaxe

IsoDate2Num (d)

### Paramètres

Paramètre	Description
d	Chaîne de date valide.

### Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction IsoDate2Num :

Expression	Renvoie
IsoDate2Num ("1900")	1
IsoDate2Num ("1900-01")	1
IsoDate2Num ("1900-01-01")	1
IsoDate2Num ("19960315T20:20:20")	35138
IsoDate2Num ("2000-03-01") - IsoDate2Num ("20000201")	29

## IsoTime2Num

Retourne le nombre de millisecondes écoulées depuis l'époque considérée, à partir d'une chaîne d'heure valide.

### Syntaxe

IsoTime2Num (d)

### Paramètres

Paramètre	Description
d	Chaîne d'heure valide.

### Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction IsoTime2Num :

Expression	Renvoie
IsoTime2Num ("00:00:00Z")	1, pour un utilisateur situé dans le fuseau horaire de l'heure normale de l'Est.
IsoTime2Num ("13")	64800001, pour un utilisateur situé à Boston, aux États-Unis.
IsoTime2Num ("13:13:13")	76393001, pour un utilisateur situé en Californie.
IsoTime2Num ("19111111T131313+01")	43993001, pour un utilisateur situé dans le fuseau horaire de l'heure normale de l'Est.

## LocalDateFmt

Renvoie une chaîne de format de date localisée, à partir d'un style de format de date.

### Syntaxe

`LocalDateFmt ([n [, k ]])`

### Paramètres

Paramètre	Description
n (facultatif)	Entier qui identifie le style de format de date propre au paramètre régional, de la façon suivante : <ul style="list-style-type: none"><li>• 1 (style court)</li><li>• 2 (style moyen)</li><li>• 3 (style long)</li><li>• 4 (style complet)</li></ul> Si n est omis (ou n'est pas valide), la valeur de style par défaut est 0.
k (facultatif)	Chaîne d'identification du paramètre régional qui se conforme aux normes de dénomination de ce dernier. Si k est omis (ou n'est pas valide), le paramètre régional ambiant est utilisé.

### Exemples

Les expressions suivantes constituent des exemples de la fonction `LocalDateFmt` :

Expression	Renvoie
<code>LocalDateFmt (1, "de_DE")</code>	tt.MM.uu
<code>LocalDateFmt (2, "fr_CA")</code>	aa-MM-jj
<code>LocalDateFmt (3, "de_CH")</code>	t. MMMM jjjj
<code>LocalDateFmt (4, "fr_FR")</code>	EEEE j MMMM aaaa

## LocalTimeFmt

Renvoie une chaîne de format d'heure localisée, à partir d'un style de format d'heure.

### Syntaxe

`LocalTimeFmt ([n [, k ]])`

### Paramètres

Paramètre	Description
n (facultatif)	Entier qui identifie le style de format d'heure propre au paramètre régional, de la façon suivante : <ul style="list-style-type: none"> <li>● 1 (style court)</li> <li>● 2 (style moyen)</li> <li>● 3 (style long)</li> <li>● 4 (style complet)</li> </ul> Si n est omis (ou n'est pas valide), la valeur de style par défaut est 0.
k (facultatif)	Chaîne d'identification du paramètre régional qui se conforme aux normes de dénomination de ce dernier. Si k est omis (ou n'est pas valide), le paramètre régional ambiant est utilisé.

### Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction LocalTimeFmt :

Expression	Renvoie
LocalTimeFmt (1, "de_DE")	HH:mm
LocalTimeFmt (2, "fr_CA")	HH:mm:ss
LocalTimeFmt (3, "de_CH")	HH:mm:ss z
LocalTimeFmt (4, "fr_FR")	HH' h 'mm z

## Num2Date

Retourne une chaîne de date, déterminant un nombre de jours écoulés depuis l'époque considérée.

### Syntaxe

Num2Date (n [, f [, k ]])

### Paramètres

Paramètre	Description
n	Entier représentant le nombre de jours. Si n n'est pas valide, la fonction renvoie une erreur.
f (facultatif)	Chaîne de format de date. Si vous omettez la valeur f, la fonction utilise le format de date par défaut MMM D, YYYY.
k (facultatif)	Chaîne d'identification du paramètre régional qui se conforme aux normes de dénomination de ce dernier. Si vous omettez la valeur k, ou si k n'est pas valide, la fonction utilise le paramètre régional ambiant.

La fonction renvoie la valeur 0 si l'une des conditions suivantes est vraie :

- Le format de la date fournie ne correspond pas au format indiqué dans la fonction.
- Le paramètre régional ou le format de date fourni dans la fonction n'est pas valide.

Les informations fournies sont insuffisantes pour déterminer un jour unique calculé depuis l'époque (autrement dit, une information relative à la date est manquante ou incomplète).

### Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction Num2Date :

Expression	Renvoie
Num2Date (1, "DD/MM/YYYY")	01/01/1900
Num2Date (35139, "DD-MMM-YYYY", "de_DE")	16-Mrz-1996
Num2Date (Date2Num ("Mar 15, 2000") - Date2Num ("98-03-15", "YY-MM-DD", "fr_CA"))	Jan 1, 1902

## Num2GMTIME

Retourne une chaîne d'heure UT, à partir du nombre de millisecondes écoulées depuis l'époque considérée.

### Syntaxe

Num2GMTIME (n [, f [, k ]])

### Paramètres

Paramètre	Description
n	Entier représentant le nombre de millisecondes. Si n n'est pas valide, la fonction renvoie une erreur.
f (facultatif)	Chaîne de format d'heure. Si vous omettez la valeur f, la fonction utilise le format d'heure par défaut H:MM:SS A.
k (facultatif)	Chaîne d'identification du paramètre régional qui se conforme aux normes de dénomination de ce dernier. Si vous omettez la valeur k, ou si k n'est pas valide, la fonction utilise le paramètre régional ambiant.

La fonction renvoie la valeur 0 si l'une des conditions suivantes est vraie :

- Le format de l'heure fournie ne correspond pas au format indiqué dans la fonction.
- Le paramètre régional ou le format d'heure fourni dans la fonction n'est pas valide.

Les informations fournies sont insuffisantes pour déterminer une heure unique calculée depuis l'époque (autrement dit, une information relative à l'heure est manquante ou incomplète).

### Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction Num2GMTTime :

Expression	Renvoi
Num2GMTTime (1, "HH:MM:SS")	00:00:00
Num2GMTTime (65593001, "HH:MM:SS Z")	18:13:13 GMT
Num2GMTTime (43993001, TimeFmt (4, "de_DE"), "de_DE")	12.13 Uhr GMT

## Num2Time

Retourne une chaîne d'heure, à partir du nombre de millisecondes écoulées depuis l'époque considérée.

### Syntaxe

Num2Time (n [, f [, k ]])

### Paramètres

Paramètre	Description
n	Entier représentant le nombre de millisecondes. Si n n'est pas valide, la fonction renvoie une erreur.
f (facultatif)	Chaîne de format d'heure. Si vous omettez la valeur f, la fonction utilise le format d'heure par défaut H:MM:SS A.
k (facultatif)	Chaîne d'identification du paramètre régional qui se conforme aux normes de dénomination de ce dernier. Si vous omettez la valeur k, ou si k n'est pas valide, la fonction utilise le paramètre régional ambiant.

La fonction renvoie la valeur 0 si l'une des conditions suivantes est vraie :

- Le format de l'heure fournie ne correspond pas au format indiqué dans la fonction.
- Le paramètre régional ou le format d'heure fourni dans la fonction n'est pas valide.

Les informations fournies sont insuffisantes pour déterminer une heure unique calculée depuis l'époque (autrement dit, une information relative à l'heure est manquante ou incomplète).

## Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction Num2Time :

Expression	Renvoie
Num2Time (1, "HH:MM:SS")	00:00:00 à Greenwich, en Angleterre, et 09:00:00 à Tokyo.
Num2Time (65593001, "HH:MM:SS Z")	13:13:13 EST à Boston, aux États-Unis.
Num2Time (65593001, "HH:MM:SS Z", "de_DE")	13:13:13 GMT-05:00 pour un utilisateur suisse allemand situé à Boston, aux États-Unis.
Num2Time (43993001, TimeFmt (4, "de_DE"), "de_DE")	13.13 Uhr GMT+01:00 pour un utilisateur situé à Zurich, en Suisse.
Num2Time (43993001, "HH:MM:SSzz")	13:13+01:00 pour un utilisateur situé à Zurich, en Suisse.

## Time

Retourne l'heure courante du système représentée sous forme du nombre de millisecondes écoulées depuis l'époque considérée.

### Syntaxe

Time ()

### Paramètres

Non

### Exemples

L'expression suivante constitue un exemple d'utilisation de la fonction Time :

Expression	Renvoie
Time ()	71533235 à précisément 15:52:15 le 15 septembre 2003 pour un utilisateur situé dans le fuseau horaire de l'heure normale de l'Est (HNE).



# Time2Num

Retourne le nombre de millisecondes écoulées depuis l'époque considérée, à partir d'une chaîne d'heure.

## Syntaxe

Time2Num(*d* [, *f* [, *k* ]])

## Paramètres

Paramètre	Description
<i>d</i>	Chaîne d'heure respectant le format indiqué par <i>f</i> qui par ailleurs se conforme au paramètre régional indiqué par <i>k</i> .
<i>f</i> (facultatif)	Chaîne de format d'heure. Si vous omettez la valeur <i>f</i> , la fonction utilise le format d'heure par défaut <code>H:MM:SS A</code> .
<i>k</i> (facultatif)	Chaîne d'identification du paramètre régional qui se conforme aux normes de dénomination de ce dernier. Si vous omettez la valeur <i>k</i> , ou si <i>k</i> n'est pas valide, la fonction utilise le paramètre régional ambiant.

La fonction renvoie la valeur 0 si l'une des conditions suivantes est vraie :

- Le format de l'heure fournie ne correspond pas au format indiqué dans la fonction.
- Le paramètre régional ou le format d'heure fourni dans la fonction n'est pas valide.

Les informations fournies sont insuffisantes pour déterminer une heure unique calculée depuis l'époque (autrement dit, une information relative à l'heure est manquante ou incomplète).

## Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction Time2Num :

Expression	Renvoie
Time2Num("00:00:00 GMT", "HH:MM:SS Z")	1
Time2Num("1:13:13 PM")	76393001 pour un utilisateur situé en Californie qui suit l'heure normale du Pacifique, et 76033001 lorsque le même utilisateur suit l'heure avancée du Pacifique.
Time2Num("13:13:13", "HH:MM:SS") - Time2Num("13:13:13 GMT", "HH:MM:SS Z") / (60 * 60 * 1000)	8 pour un utilisateur situé à Vancouver et 5 pour un utilisateur situé à Ottawa, à l'heure normale. A l'heure avancée, les valeurs renvoyées sont 7 et 4, respectivement.
Time2Num("13:13:13 GMT", "HH:MM:SS Z", "fr_FR")	47593001

## TimeFmt

Renvoie un format d'heure, à partir d'un style de format d'heure.

### Syntaxe

TimeFmt ([n [, k ]])

### Paramètres

Paramètre	Description
n (facultatif)	Entier qui identifie le style de format d'heure propre au paramètre régional, de la façon suivante : <ul style="list-style-type: none"><li>• 1 (style court)</li><li>• 2 (style moyen)</li><li>• 3 (style long)</li><li>• 4 (style complet)</li></ul> Si vous omettez la valeur n, ou si n n'est pas valide, la fonction utilise la valeur de style par défaut.
k (facultatif)	Chaîne d'identification du paramètre régional qui se conforme aux normes de dénomination de ce dernier. Si k est omis (ou n'est pas valide), le paramètre régional ambiant est utilisé.

### Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction TimeFmt :

Expression	Renvoie
TimeFmt (1)	h:MM A (if en_US locale is set)
TimeFmt (2, "fr_CA")	HH:MM:SS
TimeFmt (3, "fr_FR")	HH:MM:SS Z
TimeFmt (4, "de_DE")	H.MM' Uhr 'Z

## A propos des fonctions financières

Ces fonctions effectuent différents calculs d'intérêt, de principal et d'évaluation liés au secteur financier.

### Fonctions

- [« Apr » à la page 67](#)
- [« CTerm » à la page 68](#)
- [« FV » à la page 69](#)
- [« IPmt » à la page 70](#)
- [« NPV » à la page 71](#)
- [« Pmt » à la page 72](#)
- [« PPmt » à la page 73](#)
- [« PV » à la page 74](#)
- [« Rate » à la page 75](#)
- [« Term » à la page 76](#)

## Apr

Renvoie le taux annuel d'un prêt.

**Remarque :** les méthodes de calcul du taux d'intérêt varient d'un pays à l'autre. Cette fonction calcule le taux d'intérêt d'après les normes américaines.

### Syntaxe

`Apr (n1, n2, n3)`

### Paramètres

Paramètre	Description
n1	Valeur ou expression numérique représentant le montant du principal du prêt.
n2	Valeur ou expression numérique représentant le montant du versement envers le remboursement du prêt.
n3	Valeur ou expression numérique représentant le nombre de périodes correspondant à la durée du prêt.

Si un paramètre est nul, la fonction renvoie `null`. Si un paramètre est négatif ou équivalent à 0, la fonction renvoie une erreur.

**Remarque :** FormCalc respecte la norme internationale IEEE-754 en matière de gestion des valeurs numériques à virgule flottante. Pour plus de détails, voir [« Littéraux numériques » à la page 12](#).

## Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction `Apr` :

Expression	Renvoie
<code>Apr(35000, 269.50, 360)</code>	0.08515404566 pour un prêt de 35 000 \$ remboursé à raison de 269,50 \$ par mois pendant 30 ans.
<code>Apr(210000 * 0.75, 850 + 110, 25 * 26)</code>	0.07161332404
<code>Apr(-20000, 250, 120)</code>	Erreur
<code>Apr(P_Value, Payment, Time)</code>	Cet exemple utilise des variables au lieu de valeurs ou expressions numériques.

## CTerm

Renvoie le nombre de périodes nécessaires pour qu'un investissement, dont le taux d'intérêt est fixe mais composé, atteigne une valeur capitalisée.

**Remarque :** les méthodes de calcul du taux d'intérêt varient d'un pays à l'autre. Cette fonction calcule le taux d'intérêt d'après les normes américaines.

### Syntaxe

`CTerm(n1, n2, n3)`

### Paramètres

Paramètre	Description
n1	Valeur ou expression numérique représentant le taux d'intérêt par période.
n2	Valeur ou expression numérique représentant la valeur capitalisée de l'investissement.
n3	Valeur ou expression numérique représentant le montant de l'investissement de départ.

Si un paramètre est nul, la fonction renvoie null. Si un paramètre est négatif ou équivalent à 0, la fonction renvoie une erreur.

**Remarque :** FormCalc respecte la norme internationale IEEE-754 en matière de gestion des valeurs numériques à virgule flottante. Pour plus de détails, voir « [Littéraux numériques](#) » à la [page 12](#).

## Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction CTerm :

Expression	Renvoi
CTerm(0.02, 1000, 100)	116.2767474515
CTerm(0.10, 500000, 12000)	39.13224648502
CTerm(0.0275 + 0.0025, 1000000, 55000 * 0.10)	176.02226044975
CTerm(Int_Rate, Target_Amount, P_Value)	Cet exemple utilise des variables au lieu de valeurs ou expressions numériques.

## FV

Renvoie la valeur capitalisée de paiements fixes effectués à intervalles réguliers, compte tenu d'un taux d'intérêt fixe.

**Remarque :** les méthodes de calcul du taux d'intérêt varient d'un pays à l'autre. Cette fonction calcule le taux d'intérêt d'après les normes américaines.

### Syntaxe

FV(n1, n2, n3)

### Paramètres

Paramètre	Description
n1	Valeur ou expression numérique représentant le montant du versement.
n2	Valeur ou expression numérique représentant l'intérêt par période de l'investissement.
n3	Valeur ou expression numérique représentant le nombre total de périodes de versement.

La fonction renvoie une erreur si l'une des conditions suivantes est vraie :

- n1 ou n3 est négatif ou équivalent à 0.
- n2 est négatif.

Si un paramètre est nul, la fonction renvoie nul.

**Remarque :** FormCalc respecte la norme internationale IEEE-754 en matière de gestion des valeurs numériques à virgule flottante. Pour plus de détails, voir « [Littéraux numériques](#) » à la [page 12](#).

## Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction FV :

Expression	Renvoie
<code>FV(400, 0.10 / 12, 30 * 12)</code>	904195.16991842445. Valeur, après 30 ans, d'un investissement de 400 \$ par mois rapportant 10 % annuellement.
<code>FV(1000, 0.075 / 4, 10 * 4)</code>	58791.96145535981. Valeur, après 10 ans, d'un investissement de 1000 \$ par mois rapportant 7,5% par trimestre.
<code>FV(Payment [0], Int_Rate / 4, Time)</code>	Cet exemple utilise des variables au lieu de valeurs ou expressions numériques.

## IPmt

Renvoie le montant de l'intérêt payé pour un prêt au cours d'une période donnée.

**Remarque :** les méthodes de calcul du taux d'intérêt varient d'un pays à l'autre. Cette fonction calcule le taux d'intérêt d'après les normes américaines.

### Syntaxe

`IPmt (n1, n2, n3, n4, n5)`

### Paramètres

Paramètre	Description
n1	Valeur ou expression numérique représentant le montant du principal du prêt.
n2	Valeur ou expression numérique représentant le taux d'intérêt annuel de l'investissement.
n3	Valeur ou expression numérique représentant le montant du versement mensuel.
n4	Valeur ou expression numérique représentant le premier mois où un paiement est effectué.
n5	Valeur ou expression numérique représentant le nombre de mois à prendre en compte dans le calcul.

La fonction renvoie une erreur si l'une des conditions suivantes est vraie :

- n1, n2 ou n3 est négatif ou équivalent à 0.
- n4 ou n5 est négatif.

Si un paramètre est nul, la fonction renvoie null. Si le montant du versement (n3) est inférieur aux intérêts mensuels, la fonction renvoie 0.

**Remarque :** FormCalc respecte la norme internationale IEEE-754 en matière de gestion des valeurs numériques à virgule flottante. Pour plus de détails, voir « [Littéraux numériques](#) » à la [page 12](#).

## Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction IPmt :

Expression	Renvoi
IPmt (30000, 0.085, 295.50, 7, 3)	624.8839283142. Montant des intérêts versés sur un prêt de 30 000 \$ contracté à 8,5 %, pour les trois mois situés entre les septième et dixième mois du terme du prêt.
IPmt (160000, 0.0475, 980, 24, 12)	7103.80833569485. Montant du principal remboursé pendant la troisième année du prêt.
IPmt (15000, 0.065, 65.50, 15, 1)	0, puisque le versement mensuel est inférieur aux intérêts courus chaque mois.

## NPV

Revoie la valeur nette actualisée d'un investissement, compte tenu d'une série de mouvements futurs de l'encaisse et d'un taux d'actualisation.

**Remarque :** les méthodes de calcul du taux d'intérêt varient d'un pays à l'autre. Cette fonction calcule le taux d'intérêt d'après les normes américaines.

### Syntaxe

NPV (n1, n2 [, ...])

### Paramètres

Paramètre	Description
n1	Valeur ou expression numérique représentant le taux d'actualisation au cours d'une période.
n2	Valeur ou expression numérique représentant une valeur de mouvement d'encaisse, qui doit survenir à la fin de la période. Il est important que les valeurs indiquées dans n2 et au-delà soient dans l'ordre adéquat.

La fonction renvoie une erreur si la valeur de n1 est négative ou égale à 0. Si l'un des paramètres est nul, la fonction renvoie « null ».

**Remarque :** FormCalc respecte la norme internationale IEEE-754 en matière de gestion des valeurs numériques à virgule flottante. Pour plus de détails, voir « [Littéraux numériques](#) » à la [page 12](#).

## Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction NPV :

Expression	Renvoie
NPV(0.065, 5000)	4694.83568075117, qui est la valeur nette actualisée d'un investissement rapportant 6,5 % par année et qui générera 5 000 \$.
NPV(0.10, 500, 1500, 4000, 10000)	11529.60863329007, qui est la valeur nette actualisée d'un investissement rapportant 10 % par année et qui générera 500 \$, 1 500 \$, 4 000 \$ et 10 000 \$ pour chacune des quatre prochaines années.
NPV(0.0275 / 12, 50, 60, 40, 100, 25)	273.14193838457, qui est la valeur nette actualisée d'un investissement rapportant 2,75 % par année et qui générera 50 \$, 60 \$, 40 \$, 100 \$ et 25 \$ pour chacun des cinq prochains mois.

## Pmt

Renvoie le remboursement d'un prêt basé sur des versements fixes et sur un taux d'intérêt fixe.

**Remarque :** les méthodes de calcul du taux d'intérêt varient d'un pays à l'autre. Cette fonction calcule le taux d'intérêt d'après les normes américaines.

### Syntaxe

Pmt (n1, n2, n3)

### Paramètres

Paramètre	Description
n1	Valeur ou expression numérique représentant le montant du principal du prêt.
n2	Valeur ou expression numérique représentant le taux d'intérêt par période de l'investissement.
n3	Valeur ou expression numérique représentant le nombre total de périodes de versement.

La fonction renvoie une erreur si l'un des paramètres est négatif ou égal à 0. Si l'un des paramètres est nul, la fonction renvoie « null ».

**Remarque :** FormCalc respecte la norme internationale IEEE-754 en matière de gestion des valeurs numériques à virgule flottante. Pour plus de détails, voir « [Littéraux numériques](#) » à la [page 12](#).



## Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction `Pmt` :

Expression	Renvoie
<code>Pmt(150000, 0.0475 / 12, 25 * 12)</code>	855.17604207164, qui est le versement mensuel d'un prêt de 150 000 \$ assorti d'un taux d'intérêt annuel de 4,75 %, remboursable en 25 ans.
<code>Pmt(25000, 0.085, 12)</code>	3403.82145169876, qui est le versement annuel d'un prêt de 25 000 \$ assorti d'un taux d'intérêt annuel de 8,5 %, remboursable en 12 ans.

## PPmt

Renvoie le montant du principal payé pour un prêt au cours d'une période.

**Remarque :** les méthodes de calcul du taux d'intérêt varient d'un pays à l'autre. Cette fonction calcule le taux d'intérêt d'après les normes américaines.

### Syntaxe

`PPmt(n1, n2, n3, n4, n5)`

### Paramètres

Paramètre	Description
n1	Valeur ou expression numérique représentant le montant du principal du prêt.
n2	Valeur ou expression numérique représentant le taux d'intérêt annuel.
n3	Valeur ou expression numérique représentant le montant du versement mensuel.
n4	Valeur ou expression numérique représentant le premier mois où un paiement est effectué.
n5	Valeur ou expression numérique représentant le nombre de mois à prendre en compte dans le calcul.

La fonction renvoie une erreur si l'une des conditions suivantes est vraie :

- n1, n2 ou n3 est négatif ou équivalent à 0.
- n4 ou n5 est négatif.

Si un paramètre est nul, la fonction renvoie null. Si le montant du versement (n3) est inférieur aux intérêts mensuels, la fonction renvoie 0.

**Remarque :** FormCalc respecte la norme internationale IEEE-754 en matière de gestion des valeurs numériques à virgule flottante. Pour plus de détails, voir « [Littéraux numériques](#) » à la [page 12](#).

## Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction `PPmt` :

Expression	Renvoie
<code>PPmt(30000, 0.085, 295.50, 7, 3)</code>	261.6160716858, qui est le montant du principal remboursé sur un prêt de 30 000 \$ contracté à 8,5 %, pour les trois mois situés entre les septième et dixième mois du terme du prêt.
<code>PPmt(160000, 0.0475, 980, 24, 12)</code>	4656.19166430515, qui est le montant du principal remboursé pendant la troisième année du prêt.
<code>PPmt(15000, 0.065, 65.50, 15, 1)</code>	0, puisque dans le présent cas le versement mensuel est inférieur aux intérêts courus chaque mois, et que donc aucune partie du principal n'a été remboursée.

## PV

Renvoie la valeur actualisée d'un investissement réalisé par versements fixes périodiques avec un taux d'intérêt fixe.

**Remarque :** les méthodes de calcul du taux d'intérêt varient d'un pays à l'autre. Cette fonction calcule le taux d'intérêt d'après les normes américaines.

### Syntaxe

`PV(n1, n2, n3)`

### Paramètres

Paramètre	Description
n1	Valeur ou expression numérique représentant le montant du versement.
n2	Valeur ou expression numérique représentant l'intérêt par période de l'investissement.
n3	Valeur ou expression numérique représentant le nombre total de périodes de versement.

La fonction renvoie une erreur si la valeur de `n1` ou de `n3` est négative ou égale à 0. Si l'un des paramètres est nul, la fonction renvoie « null ».

**Remarque :** FormCalc respecte la norme internationale IEEE-754 en matière de gestion des valeurs numériques à virgule flottante. Pour plus de détails, voir « [Littéraux numériques](#) » à la [page 12](#).

## Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction PV :

Expression	Renvoie
PV(400, 0.10 / 12, 30 * 12)	45580.32799074439. Valeur, après 30 ans, d'un investissement de 400 \$ par mois rapportant 10 % annuellement.
PV(1000, 0.075 / 4, 10 * 4)	58791.96145535981. Valeur, après dix ans, d'un investissement de 1000 \$ par mois rapportant 7,5 % par trimestre.
PV(Payment [0], Int_Rate / 4, Time)	Cet exemple utilise des variables au lieu de valeurs ou expressions numériques.

## Rate

Renvoie le taux d'intérêt composé par période nécessaire pour qu'un investissement d'une valeur actualisée atteigne une valeur capitalisée au cours d'une période donnée.

**Remarque :** les méthodes de calcul du taux d'intérêt varient d'un pays à l'autre. Cette fonction calcule le taux d'intérêt d'après les normes américaines.

### Syntaxe

Rate (n1, n2, n3)

### Paramètres

Paramètre	Description
n1	Valeur ou expression numérique représentant la valeur capitalisée de l'investissement.
n2	Valeur ou expression numérique représentant la valeur actualisée de l'investissement.
n3	Valeur ou expression numérique représentant le nombre total de périodes de versement.

La fonction renvoie une erreur si l'un des paramètres est négatif ou égal à 0. Si l'un des paramètres est nul, la fonction renvoie « null ».

**Remarque :** FormCalc respecte la norme internationale IEEE-754 en matière de gestion des valeurs numériques à virgule flottante. Pour plus de détails, voir « [Littéraux numériques](#) » à la [page 12](#).

## Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction `Rate` :

Expression	Renvoie
<code>Rate(12000, 8000, 5)</code>	0.0844717712 (ou 8,45 %), qui est le taux d'intérêt par période nécessaire pour que la valeur actuelle de 8 000 \$ atteigne 12 000 \$ en cinq périodes.
<code>Rate(10000, 0.25 * 5000, 4 * 12)</code>	0.04427378243 (ou 4,43 %), qui est le taux d'intérêt mensuel nécessaire pour que la valeur actuelle atteigne 10 000 \$ en quatre ans.
<code>Rate(Target_Value, Pres_Value[*], Term * 12)</code>	Cet exemple utilise des variables au lieu de valeurs ou expressions numériques.

## Term

Renvoie le nombre de périodes nécessaires pour atteindre la valeur capitalisée donnée, compte tenu de paiements fixes périodiques versés sur un compte portant intérêt.

**Remarque :** les méthodes de calcul du taux d'intérêt varient d'un pays à l'autre. Cette fonction calcule le taux d'intérêt d'après les normes américaines.

### Syntaxe

`Term(n1, n2, n3)`

### Paramètres

Paramètre	Description
n1	Valeur ou expression numérique représentant le montant du versement effectué à la fin de chaque période.
n2	Valeur ou expression numérique représentant le taux d'intérêt par période de l'investissement.
n3	Valeur ou expression numérique représentant la valeur capitalisée de l'investissement.

La fonction renvoie une erreur si l'un des paramètres est négatif ou égal à 0. Si l'un des paramètres est nul, la fonction renvoie « null ».

**Remarque :** FormCalc respecte la norme internationale IEEE-754 en matière de gestion des valeurs numériques à virgule flottante. Pour plus de détails, voir « [Littéraux numériques](#) » à la [page 12](#).

## Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction `Term` :

Expression	Renvoi
<code>Term(475, .05, 1500)</code>	3.00477517728 (soit environ 3), qui est le nombre de périodes nécessaires pour qu'un versement de 475 \$ atteigne 1 500 \$, avec un taux d'intérêt de 5 % par période.
<code>Term(2500, 0.0275 + 0.0025, 5000)</code>	1.97128786369, qui est le nombre de périodes nécessaires pour que des versements de 2500 \$ atteignent 5000 \$, avec un taux d'intérêt de 3% par période.
<code>Rate(Inv_Value[0], Int_Rate + 0.0050, Target_Value)</code>	Cet exemple utilise des variables au lieu de valeurs ou expressions numériques. Dans le présent cas, la première occurrence de la variable <code>Inv_Value</code> sert de montant de versement, un demi point de pourcentage est ajouté à la variable <code>Int_Rate</code> qui représente le taux d'intérêt, et la variable <code>Target_Value</code> correspond à la valeur capitalisée de l'investissement.

## A propos des fonctions logiques

Ces fonctions sont utiles pour tester et analyser des informations dans le but d'obtenir un résultat vrai ou faux.

### Fonctions

- [« Choose » à la page 78](#)
- [« Exists » à la page 79](#)
- [« HasValue » à la page 80](#)
- [« Oneof » à la page 80](#)
- [« Within » à la page 81](#)

## Choose

Sélectionne une valeur dans un ensemble de paramètres donné.

### Syntaxe

`Choose (n, s1 [, s2 ...])`

### Paramètres

Paramètre	Description
n	Position de la valeur que vous souhaitez sélectionner dans l'ensemble. Si cette valeur n'est pas un nombre entier, la fonction arrondit n à la valeur entière la plus proche.  Cette fonction renvoie une chaîne vide si l'une ou l'autre des conditions suivantes est vraie : <ul style="list-style-type: none"><li>• n est inférieur à 1.</li><li>• n est supérieur au nombre d'éléments contenus dans la liste.</li></ul> Si n est nul, la fonction renvoie « null ».
s1	Première valeur de l'ensemble de valeurs.
s2 (facultatif)	Autres valeurs de l'ensemble.

## Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction `Choose` :

Expression	Renvoie
<code>Choose(3, "Taxes", "Price", "Person", "Teller")</code>	Person
<code>Choose(2, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1)</code>	9
<code>Choose(Item_Num[0], Items[*])</code>	Renvoie la valeur de l'ensemble <code>Items</code> qui correspond à la position définie par la première occurrence de <code>Item_Num</code> .
<code>Choose(20/3, "A", "B", "C", "D", "E", "F", "G", "H")</code>	F

## Exists

Détermine si le paramètre donné est une syntaxe de référence à un objet existant.

### Syntaxe

`Exists(v)`

### Paramètres

Paramètre	Description
<code>v</code>	Expression de syntaxe de référence valide. Si <code>v</code> n'est pas une syntaxe de référence, la fonction renvoie faux (0).

### Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction `Exists` :

Expression	Renvoie
<code>Exists(Item)</code>	Vrai (1) si l'objet <code>Item</code> existe et Faux (0) dans le cas contraire.
<code>Exists("hello world")</code>	Faux (0). La chaîne n'est pas une syntaxe de référence.
<code>Exists(Invoice.Border.Edge[1].Color)</code>	Vrai (1) si l'objet <code>Invoice</code> existe et possède la propriété <code>Border</code> , laquelle est définie au moins par une propriété <code>Edge</code> , qui dispose quant à elle de la propriété <code>Color</code> . Sinon, la fonction renvoie Faux (0).

## HasValue

Détermine si le paramètre indiqué est une syntaxe de référence dont la valeur n'est ni nulle, ni vide, ni un blanc.

### Syntaxe

HasValue (v)

### Paramètres

Paramètre	Description
v	Expression de syntaxe de référence valide. Si v n'est pas une syntaxe de référence, la fonction renvoie faux (0).

### Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction HasValue.

Expression	Renvoie
HasValue (2)	Vrai (1)
HasValue (" ")	Faux (0).
HasValue (Amount [*])	Erreur
HasValue (Amount [0])	Evalue la première occurrence de Amount et renvoie Vrai (1) si sa valeur n'est ni nulle, ni vide, ni un blanc.

## Oneof

Détermine si la valeur fournie fait partie d'un ensemble.

### Syntaxe

Oneof (s1, s2 [, s3 ...])

### Paramètres

Paramètre	Description
s1	Position de la valeur que vous souhaitez sélectionner dans l'ensemble. Si cette valeur n'est pas un nombre entier, la fonction arrondit s1 à la valeur entière la plus proche.
s2	Première valeur de l'ensemble de valeurs.
s3 (facultatif)	Autres valeurs de l'ensemble.



## Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction `Oneof` :

Expression	Renvoi
<code>Oneof(3, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1)</code>	Vrai (1)
<code>Oneof("John", "Bill", "Gary", "Joan", "John", "Lisa")</code>	Vrai (1)
<code>Oneof(3, 1, 25)</code>	Faux (0).
<code>Oneof("loan", Fields[*])</code>	Détermine si une occurrence de <code>Fields</code> a la valeur <code>loan</code> .

## Within

Détermine si la valeur fournie est dans une plage donnée.

### Syntaxe

`Within(s1, s2, s3)`

### Paramètres

Paramètre	Description
<code>s1</code>	Valeur à tester. Si <code>s1</code> est un nombre, la comparaison d'ordonnement est numérique. Si <code>s1</code> n'est pas un nombre, la comparaison d'ordonnement utilise la séquence de classement du paramètre régional actuel. Pour plus de détails, voir <a href="#">« Paramètres régionaux » à la page 47</a> Paramètres régionaux. Si <code>s1</code> est nul, la fonction renvoie « null ».
<code>s2</code>	Limite inférieure de la plage de test.
<code>s3</code>	Limite supérieure de la plage de test.

## Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction `Within` :

Expression	Renvoi
<code>Within("C", "A", "D")</code>	Vrai (1)
<code>Within(1.5, 0, 2)</code>	Vrai (1)
<code>Within(-1, 0, 2)</code>	Faux (0).
<code>Within(\$, 1, 10)</code>	Vrai (1) si la valeur actuelle se situe entre 1 et 10.

## A propos des fonctions diverses

Les fonctions décrites dans la présente section ne font partie d'aucune catégorie de fonctions particulière, et sont utiles dans de multiples applications.

### Fonctions

- [« Eval » à la page 82](#)
- [« Null » à la page 83](#)
- [« Ref » à la page 83](#)
- [« UnitType » à la page 84](#)
- [« UnitValue » à la page 85](#)

## Eval

Renvoie la valeur d'un calcul de formulaire donné.

### Syntaxe

`Eval (s)`

### Paramètres

Paramètre	Description
s	Chaîne valide représentant une expression ou une liste d'expressions.  <b>Remarque :</b> la fonction <code>Eval</code> ne peut pas faire référence à des variables ou à des fonctions définies par l'utilisateur. Par exemple : <pre>var s = "var t = concat(s, "hello")" eval(s)</pre> Dans ce cas, la fonction <code>Eval</code> ne reconnaît pas <code>s</code> et renvoie une erreur. Toutes les fonctions suivantes faisant référence à la variable <code>s</code> échouent également.

### Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction `Eval` :

Expression	Renvoie
<code>eval("10*3+5*4")</code>	50
<code>eval("hello")</code>	error

## Null

Renvoie la valeur nulle. La valeur nulle signifie aucune valeur.

### Définition

Null ()

### Paramètres

Non

### Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction Null :

Expression	Renvoie
Null ()	null
Null () + 5	5
Quantity = Null ()	Affecte la valeur null à l'objet Quantity.
Concat ("ABC", Null (), "DEF")	ABCDEF Voir aussi <a href="#">« Concat » à la page 87.</a>

## Ref

Renvoie une référence à un objet existant.

### Définition

Ref (v)

### Paramètres

Paramètres	Description
v	Chaîne valide représentant une syntaxe de référence, une propriété, une méthode ou une fonction.  <b>Remarque :</b> si un paramètre est nul, la fonction renvoie null. Pour tous les autres paramètres, la fonction génère une erreur.

### Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction Ref :

Expressions	Renvoie
Ref ("10*3+5*4")	10*3+5*4
Ref ("hello")	hello

## UnitType

Renvoie les unités d'une étendue d'unité. Une étendue d'unité est une chaîne composée d'un nombre suivi d'un nom d'unité.

### Syntaxe

UnitType (*s*)

### Paramètres

Paramètre	Description
<i>s</i>	Chaîne valide contenant une valeur numérique et une unité de mesure valide (unitspan). Unités de mesure reconnues : <ul style="list-style-type: none"><li>● in, inches (pouces)</li><li>● mm, millimètres</li><li>● cm, centimètres</li><li>● pt, picas, points</li><li>● mp, millipoints</li></ul> Si <i>s</i> n'est pas valide, la fonction renvoie la valeur <i>in</i> .

### Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction UnitType :

Expression	Résultats
UnitType("36 in")	in
UnitType("2.54centimeters")	cm
UnitType("picas")	pt
UnitType("2.cm")	cm
UnitType("2.zero cm")	in
UnitType("kilometers")	in
UnitType(Size[0])	Renvoie la valeur de la mesure de la première occurrence de Size.

## UnitValue

Renvoie la valeur numérique d'une mesure avec l'étendue d'unité qui lui est associée, après une conversion d'unité facultative. Une étendue d'unité est une chaîne composée d'un nombre suivi d'une unité de mesure valide.

### Syntaxe

`UnitValue(s1 [, s2 ])`

### Paramètres

Paramètres	Description
s1	Chaîne valide contenant une valeur numérique et une unité de mesure valide (unit <span>span</span> ). Unités de mesure reconnues : <ul style="list-style-type: none"><li>• in, inches (pouces)</li><li>• mm, millimètres</li><li>• cm, centimètres</li><li>• pt, picas, points</li><li>• mp, millipoints</li></ul>
s2 (facultatif)	Chaîne contenant une unité de mesure valide. La fonction convertit l'étendue d'unité indiquée dans s1 sous la nouvelle unité de mesure spécifiée par ce paramètre.  Si vous ne donnez pas de valeur à s2, la fonction utilise l'unité de mesure indiquée dans s1. Si s2 n'est pas valide, la fonction convertit s1 en pouces.

### Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction `UnitValue` :

Expression	Renvoie
<code>UnitValue("2in")</code>	2
<code>UnitValue("2in", "cm")</code>	5.08
<code>UnitValue("6", "pt")</code>	432
<code>UnitValue("A", "cm")</code>	0
<code>UnitValue(Size[2], "mp")</code>	Renvoie la valeur de la mesure de la troisième occurrence de <code>Size</code> convertie en millipoints.
<code>UnitValue("5.08cm", "kilograms")</code>	2

## A propos des fonctions de chaîne

Les fonctions décrites dans la présente section portent sur la manipulation, l'évaluation et la création de valeurs de chaîne.

### Fonctions

- [« At » à la page 87](#)
- [« Concat » à la page 87](#)
- [« Decode » à la page 88](#)
- [« Encode » à la page 89](#)
- [« Format » à la page 90](#)
- [« Left » à la page 91](#)
- [« Len » à la page 91](#)
- [« Lower » à la page 92](#)
- [« Ltrim » à la page 93](#)
- [« Parse » à la page 93](#)
- [« Replace » à la page 94](#)
- [« Right » à la page 95](#)
- [« Rtrim » à la page 95](#)
- [« Space » à la page 96](#)
- [« Str » à la page 96](#)
- [« Stuff » à la page 97](#)
- [« Substr » à la page 98](#)
- [« Uuid » à la page 99](#)
- [« Upper » à la page 99](#)
- [« WordNum » à la page 100](#)

## At

Repère la position du premier caractère d'une chaîne dans une autre chaîne.

### Syntaxe

At (s1, s2)

### Paramètres

Paramètre	Description
s1	Chaîne source.
s2	Chaîne de recherche. Si s2 ne fait pas partie de s1, la fonction renvoie 0. Si s2 est vide, la fonction renvoie 1.

### Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction At :

Expression	Renvoie
At ("ABCDEFGH", "AB")	1
At ("ABCDEFGH", "F")	6
At (23412931298471, 29)	5 , la première occurrence de 29 à l'intérieur de la chaîne source.
At (Ltrim(Cust_Info[0]), "555")	Emplacement de la chaîne 555 à l'intérieur de la première occurrence de Cust_Info. Voir aussi <a href="#">« Ltrim » à la page 93</a> .

## Concat

Renvoie la concaténation des chaînes indiquées.

### Syntaxe

Concat (s1 [, s2 ...])

### Paramètres

Paramètre	Description
s1	Première chaîne de l'ensemble.
s2 (facultatif)	Autres chaînes à ajouter à l'ensemble.

## Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction Concat :

Expression	Renvoie
Concat ("ABC", "DEF")	ABCDEF
Concat ("Tony", Space (1), "Blue")	Tony Blue Voir aussi <a href="#">« Space » à la page 96.</a>
Concat ("You owe ", WordNum (1154.67, 2), ".")	You owe One Thousand One Hundred Fifty-four Dollars And Sixty-seven Cents. Voir aussi <a href="#">« WordNum » à la page 100.</a>

## Decode

Renvoie la version décodée d'une chaîne donnée.

### Syntaxe

Decode (s1 [, s2 ])

### Paramètres

Paramètre	Description
s1	Chaîne à décoder.
s2 (facultatif)	Chaîne identifiant le type de décodage à effectuer. Chaînes de décodage valides : <ul style="list-style-type: none"> <li>• url (décodage URL)</li> <li>• html (décodage HTML)</li> <li>• xml (décodage XML)</li> </ul> Si vous omettez la valeur s2, la fonction utilise le décodage URL.

## Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction Decode :

Expression	Renvoie
Decode ("&AElig;&Aacute;&Acirc;&Aacute;&Acirc;", "html")	ÆÁÃÄ
Decode ("~!@#\$\$%^&*()_+ `{&quot;} [] &lt;&gt;;? , . / ; &apos; : ", "xml")	~!@#\$\$%^&*()_+ `{""} [] <>?, ./;':



## Encode

Renvoie la version codée d'une chaîne donnée.

### Syntaxe

Encode (*s1* [, *s2* ])

### Paramètres

Paramètre	Description
s1	Chaîne à coder.
s2 (facultatif)	Chaîne identifiant le type de codage à effectuer. Chaînes de codage valides : <ul style="list-style-type: none"><li>• url (codage URL)</li><li>• html (codage HTML)</li><li>• xml (codage XML)</li></ul> Si vous omettez la valeur <i>s2</i> , la fonction utilise le codage URL.

### Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction Encode :

Expression	Renvoie
Encode (" "hello, world! " ", "url")	%22hello,%20world!%22
Encode ("ÁÂÃÄÅÆ", "html")	&#xc1;&#xc2;&#xc3;&#xc4;&#xc5;&#xc6;

# Format

Formate les données fournies selon la chaîne de format d'image indiquée.

## Syntaxe

Format (s1, s2 [, s3 ...])

## Paramètres

Paramètre	Description
s1	Chaîne de format d'image, qui peut être un format de date ou d'heure prenant en compte le paramètre régional. Voir <a href="#">« Paramètres régionaux » à la page 249</a> .
s2	<p>Données source à formater.</p> <p>Pour les formats d'image de date, les données source doivent être soit une chaîne de date/heure ISO, soit une chaîne de date ISO dans l'un ou l'autre des formats suivants :</p> <ul style="list-style-type: none"> <li>• AAAA[MM[JJ]]</li> <li>• AAAA[-MM[-JJ]]</li> </ul> <p>Pour les formats d'image de date, les données source doivent être soit une chaîne de date/heure ISO, soit une chaîne de date ISO dans l'un ou l'autre des formats suivants :</p> <ul style="list-style-type: none"> <li>• HH[MM[SS[.FFF][z]]]</li> <li>• HH[MM[SS[.FFF][+HH[MM]]]]</li> <li>• HH[MM[SS[.FFF][-HH[MM]]]]</li> <li>• HH[:MM[:SS[.FFF][z]]]</li> <li>• HH[:MM[:SS[.FFF][-HH[:MM]]]]</li> <li>• HH[:MM[:SS[.FFF][+HH[:MM]]]]</li> </ul> <p>Pour les formats d'image de date-heure, les données source doivent être une chaîne de date-heure ISO.</p> <p>Pour les formats d'image numériques, les données source doivent être numériques.</p> <p>Pour les formats d'image textuels, les données source doivent être textuelles.</p> <p>Pour les formats d'image composés, le nombre d'arguments des données source doit correspondre au nombre de sous-éléments de l'image.</p>
s3 (facultatif)	Autres données source à formater.

## Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction Format :

Expression	Renvoi
Format ("MMM D, YYYY", "20020901")	Sep 1, 2002
Format ("\$\$9,999,999.99", 1234567.89)	\$1,234,567.89 aux États-Unis et1 234 567,89euros en France.

## Left

Extrait un nombre spécifique de caractères d'une chaîne, en commençant par le premier caractère à gauche.

### Syntaxe

Left (*s*, *n*)

### Paramètres

Paramètre	Description
<i>s</i>	Chaîne dans laquelle l'extraction est effectuée.
<i>n</i>	Nombre de caractères à extraire. Si le nombre de caractères à extraire est supérieur à la taille de la chaîne, la fonction renvoie toute la chaîne. Si le nombre de caractères à extraire est 0 ou moins, la fonction renvoie une chaîne vide.

### Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction `Left` :

Expression	Renvoie
Left("ABCDEFGH", 3)	ABC
Left("Tony Blue", 5)	"Tony "
Left(Telephone[0], 3)	Les trois premiers caractères de la première occurrence de Telephone.
Left(Rtrim>Last_Name), 3)	Les trois premiers caractères de Last_Name. Voir aussi <a href="#">« Rtrim » à la page 95</a> .

## Len

Renvoie le nombre de caractères d'une chaîne donnée.

### Syntaxe

Len (*s*)

### Paramètres

Paramètre	Description
<i>s</i>	Chaîne à évaluer.

## Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction `Len` :

Expression	Renvoi
<code>Len ("ABDCEFGH")</code>	8
<code>Len (4)</code>	1
<code>Len (Str (4.532, 6, 4))</code>	6 Voir aussi <a href="#">« Str » à la page 96</a> .
<code>Len (Amount [ * ])</code>	Nombre de caractères de la première occurrence de <code>Amount</code> .

## Lower

Convertit en minuscules toutes les majuscules d'une chaîne donnée.

### Syntaxe

`Lower (s, [, k ])`

### Paramètres

Paramètre	Description
<code>s</code>	Chaîne à convertir.
<code>k</code> (facultatif)	Chaîne représentant un paramètre régional valide. Si vous omettez la valeur <code>k</code> , la fonction utilise le paramètre régional ambiant. Voir aussi <a href="#">« Paramètres régionaux » à la page 47</a> Paramètres régionaux. <b>Remarque :</b> cette fonction convertit uniquement les caractères Unicode U+41 à U+5A (du jeu de caractères ASCII), ainsi que les caractères U+FF21 à U+FF3A (du jeu de caractères pleine largeur).

## Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction `Lower` :

Expression	Renvoi
<code>Lower ("ABC")</code>	abc
<code>Lower ("21 Main St.")</code>	21 main st.
<code>Lower (15)</code>	15
<code>Lower (Address [0])</code>	Cet exemple convertit en minuscules la première occurrence de <code>Address</code> .

## Ltrim

Renvoie une chaîne sans aucun caractère d'espace blanc à gauche.

Les caractères d'espace incluent l'espace ASCII, la tabulation horizontale, le changement de ligne, la tabulation verticale, le changement de page, le retour chariot et les caractères d'espace Unicode (catégorie Unicode Zs).

### Syntaxe

Ltrim(*s*)

### Paramètres

Paramètre	Description
<i>s</i>	Chaîne à dégrossir.

### Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction Ltrim :

Expression	Renvoie
Ltrim("            ABCD")	"ABCD"
Ltrim(Rtrim("            Tony Blue            "))	"Tony Blue" Voir aussi <a href="#">« Rtrim » à la page 95</a> .
Ltrim(Address[0])	Supprime tous les espaces situés à gauche de la première occurrence de Address.

## Parse

Analyse les données fournies selon le format d'image donné.

Une analyse réussie des données génère l'un des résultats suivants :

- Format d'image de date : Chaîne de date ISO au format AAAA-MM-JJ.
- Formats d'image d'heure : Chaîne d'heure ISO au format HH:MM:SS.
- Format d'image de date-heure : Chaîne de date-heure ISO au format AAAA-MM-JJTHH:MM:SS.
- Format d'image numérique : Un nombre.
- Images de texte : Du texte.

### Syntaxe

Parse(*s1*, *s2*)

### Paramètres

Paramètre	Description
s1	Chaîne de format d'image de date ou d'heure valide. Pour plus de détails sur les formats de date et d'heure, voir « <a href="#">Structuration des dates et des heures</a> » à la page 47. Pour plus de détails sur les formats de date et d'heure, voir <a href="#">A propos des fonctions de date et d'heure</a> .
s2	Données de la chaîne à analyser.

### Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction Parse :

Expression	Renvoie
Parse("MMM D, YYYY", "Sep 1, 2002")	2002-09-01
Parse("\$9,999,999.99", "\$1,234,567.89")	1234567.89 aux États-Unis.

## Replace

Remplace toutes les occurrences d'une chaîne par une autre, dans la chaîne indiquée.

### Syntaxe

Replace(s1, s2 [, s3 ])

### Paramètres

Paramètre	Description
s1	Chaîne source.
s2	Chaîne à remplacer.
s3 (facultatif)	Chaîne à utiliser pour le remplacement. Si vous omettez la valeur s3, ou si s3 est nul, la fonction utilise une chaîne vide.

### Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction Replace :

Expression	Renvoie
Replace("Tony Blue", "Tony", "Chris")	Chris Blue
Replace("ABCDEFGH", "D")	ABCEFGH
Replace("ABCDEFGH", "d")	ABCDEFGH
Replace(Comments[0], "recieve", "receive")	Corrige les utilisations mal orthographiées du mot receive dans la première occurrence de Comments.

## Right

Extrait un nombre spécifique de caractères dans une chaîne donnée, en commençant par le dernier caractère à droite.

### Syntaxe

`Right ( s , n )`

### Paramètres

Paramètre	Description
s	Chaîne à extraire.
n	Nombre de caractères à extraire. Si n est supérieur à la taille de la chaîne, la fonction renvoie toute la chaîne. Si n est 0 ou moins, la fonction renvoie une chaîne vide.

### Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction `Right` :

Expression	Renvoie
<code>Right ("ABCDEFGH", 3)</code>	FGH
<code>Right ("Tony Blue", 5)</code>	" Blue"
<code>Right (Telephone [0], 7)</code>	Les sept derniers caractères de la première occurrence de Telephone.
<code>Right (Rtrim (CreditCard_Num), 4)</code>	Les quatre derniers caractères de CreditCard_Num. Voir aussi <a href="#">« Rtrim » à la page 95.</a>

## Rtrim

Renvoie une chaîne sans aucun caractère d'espace blanc à droite.

Les caractères d'espace incluent l'espace ASCII, la tabulation horizontale, le changement de ligne, la tabulation verticale, le changement de page, le retour chariot et les caractères d'espace Unicode (catégorie Unicode Zs).

### Syntaxe

`Rtrim ( s )`

### Paramètres

Paramètre	Description
s	Chaîne à dégrossir.

## Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction `Rtrim` :

Expression	Renvoi
<code>Rtrim("ABCD ")</code>	"ABCD"
<code>Rtrim("Tony Blue ")</code>	"Tony Blue"
<code>Rtrim(Address [0])</code>	Supprime tous les espaces situés à droite de la première occurrence de <code>Address</code> .

## Space

Renvoie une chaîne composée d'un nombre donné d'espaces blancs.

### Syntaxe

`Space ( n )`

### Paramètres

Paramètre	Description
<code>n</code>	Nombre d'espaces.

## Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction `Space` :

Expression	Renvoi
<code>Space (5)</code>	" "
<code>Space (Max (Amount [*]))</code>	Chaîne vide contenant autant de caractères que la valeur de l'occurrence la plus élevée de <code>Amount</code> . Voir aussi « <a href="#">Max</a> » à la page 42.
<code>Concat ("Tony", Space (1), "Blue")</code>	Tony Blue

## Str

Convertit un nombre en une chaîne de caractères. FormCalc formate le résultat selon la largeur indiquée et l'arrondit au nombre de décimales défini.

### Syntaxe

`Str (n1 [, n2 [, n3 ]])`



### Paramètres

Paramètre	Description
n1	Nombre à convertir.
n2 (facultatif)	Largeur maximale de la chaîne. Si vous omettez la valeur n2, la fonction utilise la valeur 10 comme largeur par défaut. Si la chaîne obtenue est plus longue que n2, la fonction renvoie une chaîne d'astérisques (*) de la largeur indiquée par n2.
n3 (facultatif)	Nombre de chiffres après le point décimal. Si vous omettez la valeur n3, la fonction utilise 0 par défaut.

### Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction `Str` :

Expression	Renvoie
<code>Str(2.456)</code>	" 2"
<code>Str(4.532, 6, 4)</code>	4.5320
<code>Str(234.458, 4)</code>	" 234"
<code>Str(31.2345, 4, 2)</code>	****
<code>Str(Max(Amount [*]), 6, 2)</code>	Convertit l'occurrence la plus élevée de <code>Amount</code> en une chaîne de six caractères comprenant deux décimales. Voir aussi <a href="#">« Max » à la page 42.</a>

## Stuff

Insère une chaîne dans une autre chaîne.

### Syntaxe

`Stuff(s1, n1, n2 [, s2 ])`

### Paramètres

Paramètre	Description
s1	Chaîne source.
n1	Position, dans s1, où la nouvelle chaîne s2 doit être insérée. Si n1 est inférieur à un, la fonction utilise la position du premier caractère. Si n1 est supérieur à la longueur de s1, la fonction utilise la position du dernier caractère.
n2	Nombre de caractères à supprimer de la chaîne s1, en commençant à la position du caractère n1. Si n2 est inférieur ou égal à 0, la fonction ne supprime aucun caractère.
s2 (facultatif)	Chaîne à insérer dans s1. Si vous omettez la valeur s2, la fonction utilise une chaîne vide.

## Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction `Stuff` :

Expression	Renvoi
<code>Stuff("TonyBlue", 5, 0, " ")</code>	Tony Blue
<code>Stuff("ABCDEFGH", 4, 2)</code>	ABCFGH
<code>Stuff(Address[0], Len(Address[0]), 0, "Street")</code>	Le mot <code>Street</code> est ajouté à la fin de la première occurrence de <code>Address</code> .  Voir aussi <a href="#">« Len » à la page 91</a> .
<code>Stuff("members-list@myweb.com", 0, 0, "cc:")</code>	cc:members-list@myweb.com

## Substr

Extrait une partie d'une chaîne donnée.

### Syntaxe

`Substr(s1, n1, n2 )`

### Paramètres

Paramètre	Description
<code>s1</code>	Chaîne source.
<code>n1</code>	Position, dans la chaîne <code>s1</code> , marquant le début de l'extraction.  Si <code>n1</code> est inférieur à un, la fonction utilise la position du premier caractère. Si <code>n1</code> est supérieur à la longueur de <code>s1</code> , la fonction utilise la position du dernier caractère.
<code>n2</code>	Nombre de caractères à extraire.  Si <code>n2</code> est inférieur ou égal à 0, la fonction renvoie une chaîne vide. Si <code>n1 + n2</code> est supérieur à la longueur de <code>s1</code> , la fonction renvoie la sous-chaîne commençant à la position <code>n1</code> jusqu'à la fin de <code>s1</code> .

### Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction `Substr` :

Expression	Renvoi
<code>Substr("ABCDEFGF", 3, 4)</code>	CDEF
<code>Substr(3214, 2, 1)</code>	2
<code>Substr&gt;Last_Name[0], 1, 3)</code>	Les trois premiers caractères de la première occurrence de <code>Last_Name</code> .
<code>Substr("ABCDEFGF", 5, 0)</code>	" "
<code>Substr("21 Waterloo St.", 4, 5)</code>	Water

## Uuid

Renvoie une chaîne UUID qui sert de méthode d'identification.

### Syntaxe

Uuid( [n ] )

### Paramètres

Paramètre	Description
n	Numéro identifiant le format de la chaîne UUID. Numéros valides : <ul style="list-style-type: none"><li>0 (valeur par défaut) : la chaîne UUID contient uniquement des valeurs hexadécimales.</li><li>1: la chaîne UUID contient des tirets séparant les séquences de valeurs hexadécimales à des positions fixes.</li></ul> Si vous omettez la valeur n, la fonction utilise la valeur par défaut.

### Exemples

Les expressions suivantes constituent des exemples de la fonction Uuid :

Expression	Renvoie
Uuid()	Une valeur comme 3c3400001037be8996c400a0c9c86dd5
Uuid(0)	Une valeur comme 3c3400001037be8996c400a0c9c86dd5
Uuid(1)	Une valeur comme 1a3ac000-3dde-f352-96c4-00a0c9c86dd5
Uuid(7)	Une valeur comme 1a3ac000-3dde-f352-96c4-00a0c9c86dd5

## Upper

Convertit en majuscules toutes les minuscules d'une chaîne.

### Syntaxe

Upper( s [, k ] )

### Paramètres

Paramètre	Description
s	Chaîne à convertir.
k (facultatif)	Chaîne représentant un paramètre régional valide. Si vous omettez la valeur k, la fonction utilise le paramètre régional ambiant. Voir aussi « <a href="#">Paramètres régionaux</a> » à la page 47Paramètres régionaux. <b>Remarque :</b> cette fonction convertit uniquement les caractères Unicode U+61 à U+7A (du jeu de caractères ASCII), ainsi que les caractères U+FF41 à U+FF5A (du jeu de caractères pleine largeur).

## Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction `Upper` :

Expression	Renvoie
<code>Upper ("abc")</code>	ABC
<code>Upper ("21 Main St.")</code>	21 MAIN ST.
<code>Upper (15)</code>	15
<code>Upper (Address [0])</code>	Cet exemple convertit en majuscules la première occurrence de Address.

## WordNum

Renvoie le texte équivalent à un nombre donné.

### Syntaxe

`WordNum (n1 [, n2 [, k ]])`

### Paramètres

Paramètre	Description
n1	Nombre à convertir. Si l'un des énoncés suivants est vrai, la fonction renvoie des astérisques (*) pour indiquer une erreur : <ul style="list-style-type: none"><li>n1 n'est pas un nombre.</li><li>La valeur entière de n1 est négative.</li><li>La valeur entière de n1 est supérieure à 922 337 203 685 477 550.</li></ul>
n2 (facultatif)	Numéro identifiant l'option de formatage. Numéros valides : <ul style="list-style-type: none"><li>0 (valeur par défaut) : le nombre est converti en texte représentant la partie entière du nombre.</li><li>1: le nombre est converti en texte représentant la valeur monétaire, sans décimales.</li><li>2: le nombre est converti en texte représentant la valeur monétaire, avec les décimales.</li></ul> Si vous omettez la valeur n2, la fonction utilise la valeur par défaut (0).
k (facultatif)	Chaîne représentant un paramètre régional valide. Si vous omettez la valeur k, la fonction utilise le paramètre régional ambiant. Voir aussi « <a href="#">Paramètres régionaux</a> » à la page 47. <b>Remarque :</b> à compter de la présente version, vous ne pouvez plus indiquer un identificateur de paramètre régional autre que l'anglais pour cette fonction.

## Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction `WordNum` :

<b>Expression</b>	<b>Renvoie</b>
<code>WordNum(123.45)</code>	One Hundred and Twenty-three Dollars
<code>WordNum(123.45, 1)</code>	One Hundred and Twenty-three Dollars
<code>WordNum(1154.67, 2)</code>	One Thousand One Hundred Fifty-four Dollars And Sixty-seven Cents
<code>WordNum(43, 2)</code>	Forty-three Dollars And Zero Cents
<code>WordNum(Amount [0], 2)</code>	Cet exemple utilise la première occurrence de <code>Amount</code> comme nombre à convertir.

## A propos des fonctions URL

Ces fonctions portent sur l'envoi et la réception d'informations (type de contenu et codage des données compris) à destination ou en provenance de n'importe quel emplacement URL accessible.

### Fonctions

- [« Get » à la page 102](#)
- [« Post » à la page 103](#)
- [« Put » à la page 104](#)

## Get

Télécharge le contenu de l'adresse URL indiquée.

**Remarque :** la fonction `Get` ne s'applique qu'aux formulaires certifiés. Adobe Acrobat® et Adobe Reader® ne sont pas capables d'identifier les formulaires certifiés tant que l'événement `initialize` n'a pas été lancé. Pour tirer parti de la fonction `Get` sur les formulaires certifiés avant de procéder au rendu du formulaire, utilisez l'événement `docReady`.

### Syntaxe

`Get (s)`

### Paramètres

Paramètre	Description
<code>s</code>	Adresse URL à télécharger. Si la fonction ne parvient pas à télécharger le contenu de l'adresse URL, elle renvoie une erreur.

### Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction `Get` :

Expression	Renvoie
<code>Get ("http://www.myweb.com/data/mydata.xml")</code>	Données XML extraites du fichier indiqué.
<code>Get ("ftp://ftp.gnu.org/gnu/GPL")</code>	Contenu de la Licence Publique Générale GNU.
<code>Get ("http://intranet?sql=SELECT+*+FROM+projects+FOR+XML+AUTO,+ELEMENTS")</code>	Résultats d'une requête SQL adressée au site Web indiqué.

## Post

Place à l'adresse URL indiquée les données fournies.

**Remarque :** la fonction `Post` ne s'applique qu'aux formulaires certifiés. Acrobat et Adobe Reader ne sont pas capables d'identifier les formulaires certifiés tant que l'événement `initialize` n'a pas été lancé. Pour tirer parti de la fonction `Post` sur les formulaires certifiés avant de procéder au rendu du formulaire, utilisez l'événement `docReady`.

### Syntaxe

`Post (s1, s2 [, s3 [, s4 [, s5 ]]])`

### Paramètres

Paramètre	Description
s1	Adresse URL à laquelle les données sont inscrites.
s2	Données à publier. Si la fonction ne parvient pas à publier les données, elle renvoie une erreur.
s3 (facultatif)	Chaîne contenant le type de contenu des données à publier. Types de contenu valides : <ul style="list-style-type: none"><li>● application/flux d'octets (valeur par défaut)</li><li>● texte/html</li><li>● texte/xml</li><li>● texte/en clair</li><li>● texte en plusieurs parties/données de formulaire</li><li>● application/formulaire x-www codé url</li><li>● tout autre type MIME valide</li></ul> Si vous ne donnez pas de valeur à s3, la fonction affecte au type de contenu la valeur par défaut. L'application doit s'assurer que les données à publier utilisent le format adéquat en fonction du type de contenu indiqué.
s4 (facultatif)	Chaîne contenant le nom de la page de codes utilisée pour coder les données. Noms de pages de codes valides : <ul style="list-style-type: none"><li>● UTF-8 (valeur par défaut)</li><li>● UTF-16</li><li>● ISO-8859-1</li><li>● Tout codage de caractères reconnu par l'IANA (Internet Assigned Numbers Authority)</li></ul> Si vous ne donnez pas de valeur à s4, la fonction affecte à la page de codes la valeur par défaut. L'application doit s'assurer que le codage des données à publier correspond à la page de codes indiquée.
s5 (facultatif)	Chaîne contenant tout en-tête HTTP supplémentaire à inclure à la publication des données. Si vous ne donnez pas de valeur à s5, la fonction n'ajoute pas d'en-tête HTTP supplémentaire à la publication. Pour effectuer une publication sur un serveur SOAP, vous devez généralement utiliser un en-tête SOAPAction.

## Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction `Post` :

Expression	Renvoie
<pre>Post ("http://tools_build/scripts/jfecho.cgi", "user=joe&amp;passwd=xxxxx&amp;date=27/08/2002", "application/x-www-form-urlencoded")</pre>	Publie des données de connexion codées URL sur le serveur et renvoie l'accusé de réception de ce serveur.
<pre>Post ("http://www.nanonull.com/TimeService/ TimeService.asmx/getLocalTime", "&lt;?xml version='1.0' encoding='UTF-8'?'&gt;&lt;soap:Envelope&gt;&lt;soap:Body&gt; &lt;getLocalTime/&gt;&lt;/soap:Body&gt; &lt;/soap:Envelope&gt;", "text/xml", "utf-8", "http://www.Nanonull.com/TimeService/getLocalTime")</pre>	Publie une requête SOAP pour l'heure locale sur un serveur et attend une réponse XML.

## Put

Télécharge les données fournies vers l'adresse URL indiquée.

**Remarque :** la fonction `Put` ne s'applique qu'aux formulaires certifiés. Acrobat et Adobe Reader ne sont pas capables d'identifier les formulaires certifiés tant que l'événement `initialize` n'a pas été lancé. Pour tirer parti de la fonction `Put` sur les formulaires certifiés avant de procéder au rendu du formulaire, utilisez l'événement `docReady`.

### Syntaxe

`Put (s1, s2 [, s3 ])`

### Paramètres

Paramètre	Description
s1	Adresse URL de destination du téléchargement.
s2	Données à télécharger. Si la fonction ne parvient pas à télécharger les données, elle renvoie une erreur.
s3 (facultatif)	Chaîne contenant le nom de la page de codes utilisée pour coder les données. Noms de pages de codes valides : <ul style="list-style-type: none"> <li>• UTF-8 (valeur par défaut)</li> <li>• UTF-16</li> <li>• ISO8859-1</li> <li>• Tout codage de caractères reconnu par l'IANA (Internet Assigned Numbers Authority)</li> </ul> Si vous ne donnez pas de valeur à s3, la fonction affecte à la page de codes la valeur par défaut. L'application doit s'assurer que le codage des données à télécharger correspond à la page de codes indiquée.



## Exemples

Les expressions suivantes constituent des exemples d'utilisation de la fonction `Put` :

Expression	Renvoi
<pre>Put("ftp://www.example.com/pub/fubu.xml", "&lt;?xml version='1.0' encoding='UTF-8'?&gt;&lt;msg&gt;hello world!&lt;/msg&gt;")</pre>	Rien si le serveur FTP a autorisé l'utilisateur à télécharger certaines données XML vers le fichier <code>pub/fubu.xml</code> sur le serveur. Dans les autres cas, la fonction renvoie une erreur.

# Index

## A

Abs, fonction arithmétique 39  
Appel de fonction, FormCalc 34  
Apr, fonction financière 67  
At, fonction de chaîne 87  
Avg, fonction arithmétique 40

## C

Caractère  
  casse, conversion 92, 99  
  extraction à partir d'une chaîne 91, 95  
  position de départ 87  
  suppression d'un espace blanc d'une chaîne 93, 95  
Ceil, fonction arithmétique 40  
Chaîne vide 13  
Champ de date/heure, objet  
  FormCalc, fonctions 47  
  symbole de création de formats 53  
Choose, fonction logique 78  
Commentaire FormCalc 15  
Concat, fonction de chaîne 87  
Conversion  
  casse de caractère 92, 99  
  chaînes d'heure en nombres 65  
  nombre en chaîne 96  
  nombre en texte 100  
Count, fonction arithmétique 41  
Cterm, fonction financière 68

## D

Date, fonction 57  
Date2Num, fonction 57  
DateFmt, fonction 58  
Decode, fonction de chaîne 88

## E

Encode, fonction de chaîne 89  
Epoque, FormCalc 51  
Espace blanc  
  à propos 17  
  suppression dans une chaîne 93, 95  
Espace blanc, chaîne 96  
Eval, fonction diverse 82  
Exists, fonction logique 79  
Expression  
  FormCalc 17  
Expression break, FormCalc 26  
Expression continue, FormCalc 27  
Expression d'affectation, FormCalc 20  
Expression d'égalité, FormCalc 22  
Expression d'inégalité, FormCalc 22  
Expression for, FormCalc 25

Expression foreach, FormCalc 26  
Expression if, FormCalc 23  
Expression logique, FormCalc 20, 21  
Expression relationnelle, FormCalc 22  
Expression simple, FormCalc 18  
Expression unaire, FormCalc 21  
Expression while, FormCalc 24

## F

Floor, fonction arithmétique 41  
Fonction alphabétique, liste FormCalc 35  
Fonction arithmétique, FormCalc 39  
Fonction d'heure. *Voir* FormCalc, fonctions  
Fonction de chaîne 86  
Fonction financière, FormCalc 67  
Fonction logique, FormCalc 78  
Fonction mathématique, FormCalc 39  
Fonction URL, FormCalc 102  
Format d'heure  
  à propos 52  
  chaîne 60, 66  
  FormCalc 53  
Format d'image  
  analyse 93  
  application 90  
  date et heure 53  
Format de date  
  à propos 51  
  chaîne 58, 60  
  FormCalc 53  
Format de date et heure, symboles 53  
Format, fonction de chaîne 90  
Formats  
  date et heure 53  
FormCalc  
  à propos 7  
  appel de fonction, FormCalc 34  
  commentaire 15  
  espace blanc 17  
  expression 17  
  expression logique 20, 21  
  fonction intégrée 34  
  fonction logique 78  
  identificateur 16  
  littéral 12  
  mot de passe restreint 16  
  opérateur 14  
  paramètre régional de langue 77  
  raccourcis de la syntaxe de référence 28  
  terminateur de ligne 17  
  variable 27  
FormCalc, expression conditionnelle 23, 24, 25, 26, 27

FormCalc, fonctions  
  arithmétiques 39  
  date et heure 47  
  financières 67  
  liste alphabétique 7, 35  
  URL 102

Fusion de chaînes 87  
FV, fonction financière 69

## G

Get, fonction URL 102

## H

HasValue, fonction logique 80

## I

Identificateur, FormCalc 16, 47  
Identification unique 99  
Ipmt, fonction financière 70  
IsoDate2Num, fonction 59  
IsoTime2Num, fonction 59

## L

Langue  
  à propos 47  
Left, fonction de chaîne 91  
Len, fonction de chaîne 91  
Littéral chaîne, FormCalc 13  
Littéral numérique, FormCalc 12  
Littéral, FormCalc 12  
LocalDateFmt, fonction 60  
LocalTimeFmt, fonction 60  
Lower, fonction de chaîne 92  
Ltrim, fonction de chaîne 93

## M

Max, fonction arithmétique 42  
Min, fonction arithmétique 43  
Mod, fonction arithmétique 44  
Mot-clé, restriction FormCalc 16

## N

Nombre  
  conversion en chaîne 96  
  conversion en texte 100  
NPV, fonction financière 71  
Null, fonction diverse 83  
Num2Date, fonction 61  
Num2GMTTime, fonction 62  
Num2Time, fonction 63

## O

oneof, fonction logique 80  
Opérande, promotion 19  
Opérateur, FormCalc 14  
Opération booléenne 19

Opération de chaîne 19  
Opération numérique 19

## P

Paramètre régional 47  
  à propos 47  
  *Voir aussi* langue  
Paramètre régional ambiant 51  
Paramètre régional par défaut 51  
Parse, fonction de chaîne 93  
Pmt, fonction financière 72  
Post, fonction URL 103  
PPmt, fonction financière 73  
Publication de données vers des URL 103  
Put, fonction URL 104  
PV, fonction financière 74

## R

Raccourci, syntaxe de référence 28  
Rate, fonction financière 75  
Ref, fonction diverse 83  
Référence à un tableau 32  
Référence, syntaxe 28  
Replace, fonction de chaîne 94  
Reste 44  
Right, fonction de chaîne 95  
Round, fonction arithmétique 45  
Rtrim, fonction de chaîne 95

## S

Script, à propos de 7  
Séquence d'échappement Unicode 13  
Space, fonction de chaîne 96  
Str, fonction de chaîne 96  
Stuff, fonction de chaîne 97  
Substr, fonction de chaîne 98  
Sum, fonction arithmétique 46  
Suppression des espaces blanc 93, 95  
Syntaxe de référence  
  à propos 28  
  raccourci 28  
  raccourcis pour FormCalc 28

## T

Téléchargement de données vers des URL 104  
Téléchargement du contenu de l'URL 102  
Term, fonction financière 76  
Terminateur de ligne, FormCalc 17  
Time, fonction 64  
Time2Num, fonction 65  
TimeFmt, fonction 66

## U

Unicode, séquence d'échappement 13  
UnitType, fonction diverse 84  
UnitValue, fonction diverse 85  
Upper, fonction de chaîne 99

UUID 99  
Uuid, fonction de chaîne 99

## V

Valeurs nulles 41  
Variable  
    FormCalc 27

## W

Within, fonction logique 81  
WordNum, fonction de chaîne 100