使用 ACTIONSCRIPT® 3.0 組件

上次更新 **2011/5/17**

法律聲明

目錄

第 1 章 簡介

適用對象 1
系統需求 1
關於說明文件 1
印刷慣例 2
本手冊使用的詞彙 2
其它資源 2

第 2 章 關於 ActionScript 3.0 組件

使用組件的好處 3
組件類型 4
加入和刪除文件中的組件 5
尋找組件的版本 7
ActionScript 3.0 事件處理模型 7
簡單的應用程式 8

第 3 章 使用組件

組件的架構 15
使用組件檔案 16
除錯組件應用程式 18
設定參數和屬性 18
元件庫 19
調整組件大小 20
即時預覽 20
處理事件 21
使用顯示清單 22
使用 FocusManager 24
使用 List 架構的組件 25
使用 DataProvider 25
使用 CellRenderer 32
使組件具備輔助功能 38

第 4 章 使用 UI 組件

使用 Button 組件 40
使用 CheckBox 組件 42
使用 ColorPicker 組件 44
使用 ComboBox 組件 46
使用 DataGrid 組件 49
使用 Label 組件 54

使用 List 組件 56
使用 NumericStepper 組件 60
使用 ProgressBar 組件 62
使用 RadioButton 組件 67
使用 ScrollPane 組件 70
使用 Slider 組件 73
使用 TextArea 組件 75
使用 TextInput 組件 78
使用 TileList 組件 80
使用 UILoader 組件 83
使用 UIScrollBar 組件 84

第 5 章 自訂使用者介面組件

關於自訂 UI 組件 87
設定樣式 87
關於外觀元素 89
自訂 Button 組件 92
自訂 CheckBox 組件 94
自訂 ColorPicker 組件 95
自訂 ComboBox 組件 96
自訂 DataGrid 組件 98
自訂 Label 組件 103
自訂 List 組件 104
自訂 NumericStepper 組件 106
自訂 ProgressBar 組件 108
自訂 RadioButton 組件 109
自訂 ScrollPane 組件 111
自訂 Slider 組件 111
自訂 TextArea 組件 113
自訂 TextInput 組件 114
自訂 TileList 組件 115
自訂 UILoader 組件 117
自訂 UIScrollBar 組件 117

第 6 章 使用 FLVPlayback 組件

使用 FLVPlayback 組件 119
自訂 FLVPlayback 組件 134
使用 SMIL 檔 143

第 7 章 使用 FLVPlayback 註解功能組件

使用 FLVPlaybackCaptioning 組件 151
使用 Timed Text 註解 153
搭配使用提示點和註解功能 158

使用註解功能播放多個 FLV 檔 160
自訂 FLVPlaybackCaptioning 組件 160

第 1 章 簡介

Adobe® Flash® CS5 Professional 是標準編寫工具，可用來建立具備高度震撼力的 Web 經驗。在提供這些網頁效果的多樣化網際網路應用程式中，組件是其中的建構區塊。「組件」是具有參數的影片片段，您可以使用 Adobe® ActionScript® 方法、屬性和事件來自訂組件，無論是在 Flash 的編寫期間或是執行階段都可以。組件的用途是讓開發人員重複使用並共享程式碼，以及將設計人員不需使用 ActionScript 就能使用與自訂的複雜的功能包裝起來，方便使用。

組件可以讓您輕易且快速地建置穩定並擁有一致外觀和行為的應用程式。本手冊將說明如何使用 Adobe ActionScript 3.0 組件來建置應用程式。「Adobe® ActionScript® 3.0 語言和組件參考」中有每項組件的應用程式設計介面 (Application Programming Interface，API) 說明。

您可以使用 Adobe® 所建立的組件、下載其他開發人員所建立的組件，或自行建立個人的組件。

適用對象

本手冊的適用對象包括想要建立 Flash 應用程式並運用組件加速開發工作的開發人員。您應該已經熟悉運用 Flash 開發應用程式，以及編寫 ActionScript。

如果比較缺乏編寫 ActionScript 的經驗，您可以將組件新增到文件中，然後在 「屬性」檢測器或 「組件檢測器」中設定其參數，並使用 「行為」面板處理其事件。例如，您可以將 「前往網頁」行為指令附加到 Button 組件，即可在按下按鈕時在網頁瀏覽器中開啟一個 URL，而不需要編寫任何 ActionScript 程式碼。

如果您是想要建立更穩定應用程式的程式設計人員，您可以用動態方式建立組件、在執行階段使用 ActionScript 設定屬性和呼叫方法，並使用事件偵聽程式模型來處理事件。

如需詳細資訊，請參閱第 15 頁 「使用組件」。

系統需求

除了 Flash 的系統需求以外，Flash 組件並不需要任何的系統需求。

任何使用 Flash CS3 或更新版本組件的 SWF 檔，都必須使用 Adobe® Flash® Player 9.0.28.0 或更新版本來進行檢視，而且必須發佈為 ActionScript 3.0 格式 (您可以在 「檔案 > 發佈設定」的 「Flash」索引標籤中完成這項設定)。

關於說明文件

本文件詳細說明了如何利用組件開發 Flash 應用程式。它會假設您已具備 Flash 和 ActionScript 3.0 的一般知識。關於 Flash 和相關產品的特定說明文件則會個別提供。

本文件也具備 PDF 檔和線上說明版本。若要檢視線上說明，請啟動 Flash 並選取 「說明 > Flash 說明 > 使用 Adobe ActionScript 3.0 組件」。

如需關於 Flash 的相關資訊，請參閱下列文件：

- 使用 Flash
- ActionScript 3.0 開發人員指南
- 適用於 Adobe Flash Platform 的 ActionScript 3.0 參考

印刷慣例

本手冊的印刷慣例如下：

- 斜體字代表會被取代的值 (例如，位於資料夾路徑的值)。
- 程式碼字體代表 ActionScript 程式碼，包括方法和屬性名稱。
- 程式碼斜體字代表應該取代的程式碼項目 (例如：一個 ActionScript 參數)。
- 粗體字代表由您輸入的值。

本手冊使用的詞彙

本手冊使用了以下術語：

執行階段 當程式碼在 Flash Player 中執行時。

編寫期間 當您在 Flash 編寫環境中工作時。

其它資源

除了這些手冊的參考內容外，Adobe 也會定期在 「Adobe 開發人員中心」和 「Adobe 設計中心」上提供更新文章、設計參考和範例。

您可以在 www.adobe.com/go/learn_fl_samples_tw 上找到其它的組件樣本。

Adobe 開發人員中心
「Adobe 開發人員中心」是有關 ActionScript 最新資訊、實際應用程式開發文章以及重要合併課題資訊的參考資源。請造訪「開發人員中心」，網址為 www.adobe.com/go/flash_devcenter_tw。

Adobe 設計中心
學習最新的數位設計與移動圖像。瀏覽一流名家的作品、瞭解新的設計趨勢以及利用教學課程、關鍵工作流程與高階技術磨練您的技巧。每個月請檢查兩次最新的教學課程和文章，以及創意展示部分。請造訪 「設計中心」，網址為 www.adobe.com/go/fl_designcenter_tw。

第 2 章 關於 ActionScript 3.0 組件

Adobe® Flash® Professional CS5 組件是具有參數的影片片段，您可以透過這些參數來修改組件的外觀及行為。組件可以用來簡化使用者介面控制項 (如 RadioButton 或 CheckBox)，或是加入內容 (如 List 或 DataGrid)。

組件可以讓您輕易且快速地建置穩定並擁有一致行為和外觀的 Flash 應用程式。您可以使用 Flash 組件來操作這些控制項，不需要自行建立按鈕、下拉式清單方塊和清單。只需將它們從 「組件」面板拖入您的應用程式文件即可。您還可以輕鬆地自訂組件的外觀感覺，以配合應用程式設計的需要。

雖然不需要深入瞭解 ActionScript 就可以完成這些動作，您仍能使用 ActionScript 3.0 修改組件的行為或實作新的行為。每個組件都有一組獨一無二的 ActionScript 方法、屬性和事件，構成其 「應用程式設計介面」 (API)。API 可以讓您在應用程式執行階段建立與操作組件。

API 也可以讓您建立專用的自訂組件。您可以從 Adobe Exchange 下載由 Flash 社群成員建置的組件，網址為 www.adobe.com/go/flash_exchange_tw。如需有關建立組件的詳細資訊，請參閱 www.adobe.com/go/learn_fl_creating_components_tw。

ActionScript 3.0 組件架構包括類別 (所有組件的基礎)、外觀元素和樣式 (可以讓您自訂外觀)、事件處理模型、焦點管理、輔助功能介面，還有其它更多項目。

備註：Adobe Flash CS5 包括 ActionScript 2.0 組件以及 ActionScript 3.0 組件。但是，這兩組組件不能混合使用，一個應用程式只能使用其中一組組件。Flash CS5 會根據您開啟的是 ActionScript 2.0 或 ActionScript 3.0 檔案，而使用相對的 ActionScript 2.0 組件或 ActionScript 3.0 組件。建立新的 Flash 文件時，您必須指定 「Flash 檔案 (ActionScript 3.0)」或 「Flash 檔案 (ActionScript 2.0)」。當開啟現有的文件時，Flash 會檢查 「發佈設定」來決定要使用哪一組組件。如需有關 ActionScript 2.0 組件的詳細資訊，請參閱 「使用 Adobe® ActionScript® 2.0 組件」。

如需 Flash ActionScript 3.0 組件的完整清單，請參閱第 4 頁 「組件類型」。

使用組件的好處

組件可以讓您將設計應用程式的流程與編寫程式的流程區分開來。組件讓程式開發人員能夠建立各種功能，供設計人員在應用程式中運用。開發人員可以將經常使用的功能封裝成組件，設計人員可以變更組件的參數來自訂組件的大小、位置和行為。他們也可以編輯組件的圖像元素或外觀元素來變更它的外觀。

組件之間可以共用樣式、外觀元素和焦點管理之類的核心功能。加入應用程式的第一個組件，它的核心功能大約會佔用 20 KB 的大小。當您加入其它的組件時，後續加入的組件會共用最初的記憶體配置，避免繼續增加應用程式的大小。

本節將概要說明使用 ActionScript 3.0 組件的一些好處。

ActionScript 3.0 的強大功能 提供強大的物件導向程式設計語言，這是 Flash Player 功能的一項重大進展。這種語言是在重複使用程式碼的基礎上為建置多樣化網際網路應用程式而設計的。ActionScript 3.0 是依據 ECMAScript 國際標準化 Script 編寫語言，且遵循 ECMAScript (ECMA-262) 第 3 版語言規格。如需 ActionScript 3.0 的完整簡介，請參閱 「ActionScript 3.0 開發人員指南」。如需有關語言的參考資訊，請參閱 適用於 Adobe Flash Platform 的 ActionScript 3.0 參考。

FLA 架構的使用者介面組件 提供在編寫期間自訂外觀元素的簡便方式。這些組件也提供樣式 (包括外觀元素樣式) 可以讓您在執行階段自訂組件的各種外觀及載入外觀元素。如需詳細資訊，請參閱第 87 頁 「自訂使用者介面組件」以及 ActionScript 3.0 參考。

新增的 FVLPlayback 組件已經加入 FLVPlaybackCaptioning 組件，提供了全螢幕支援、更佳的即時預覽、外觀元素，可以讓您加入顏色和 Alpha 設定並增強 FLV 下載和版面配置的功能。

屬性檢測器和組件檢測器 可以讓您在 Flash 編寫期間變更組件參數。如需詳細資訊，請參閱第 16 頁 「使用組件檔案」以及第 18 頁 「設定參數和屬性」。

新增的組件集合對話方塊 包含 ComboBox、List 和 TileList 組件，可以讓您透過使用者介面填入它們的 dataProvider 屬性。如需詳細資訊，請參閱第 25 頁 「建立 DataProvider」。

ActionScript 3.0 事件模型 可以讓應用程式偵聽事件並叫用事件處理常式進行回應。如需詳細資訊，請參閱：第 7 頁 「ActionScript 3.0 事件處理模型」以及第 21 頁 「處理事件」。

Manager 類別 提供在應用程式中處理焦點和管理樣式的簡便方式。如需詳細資訊，請參閱 ActionScript 3.0 參考。

UIComponent 基底類別 對組件提供核心方法、屬性和事件以延伸這些組件。所有 ActionScript 3.0 使用者介面組件都是繼承自 UIComponent 類別。如需詳細資訊，請參閱 ActionScript 3.0 參考。

使用 SWC 在 UI FLA 架構的組件中，可提供 ActionScript 定義做為組件時間軸內的資源以加快編譯速度。

能輕易延伸的類別階層 (使用 ActionScript 3.0)，讓您能夠建立唯一的命名空間、視需要匯入類別，並輕鬆劃分子類別以擴充組件。

如需詳細資訊，請參閱 ActionScript 3.0 參考。

備註：Flash CS5 同時支援 FLA 架構和 SWC 架構的組件。如需詳細資訊，請參閱第 15 頁 「組件的架構」。

組件類型

當您安裝 Flash CS5 時，會一併安裝 Flash 組件。

ActionScript 3.0 組件包括下列使用者介面 (UI) 組件：

Button	List	TextArea
CheckBox	NumericStepper	TextInput
ColorPicker	RadioButton	TileList
ComboBox	ProgressBar	UILoader
DataGrid	ScrollPane	UIScrollBar
Label	Slider	

除了使用者介面組件以外，Flash ActionScript 3.0 組件還包括下列組件和支援的類別：

- FLVPlayback 組件 (fl.video.FLVPlayback)，屬於 SWC 架構的組件。

  FLVPlayback 組件可以讓您輕鬆地在 Flash 應用程式中加入視訊播放程式，以便透過 HTTP，從 Adobe® Flash® 視訊串流服務 (FVSS) 或 Adobe 的 Macromedia® Flash® Media Server (FMS) 播放漸進式串流視訊。如需詳細資訊，請參閱第 119 頁 「使用 FLVPlayback 組件」。
- FLVPlayback 自訂使用者介面組件，屬於 FLA 架構的組件，可以同時使用於 FLVPlayback 組件的 ActionScript 2.0 和 ActionScript 3.0 版本。如需詳細資訊，請參閱第 119 頁 「使用 FLVPlayback 組件」。
- FLVPlayback 註解功能組件，提供封閉式的 FLVPlayback 註解功能。請參閱第 151 頁 「使用 FLVPlayback 註解功能組件」。

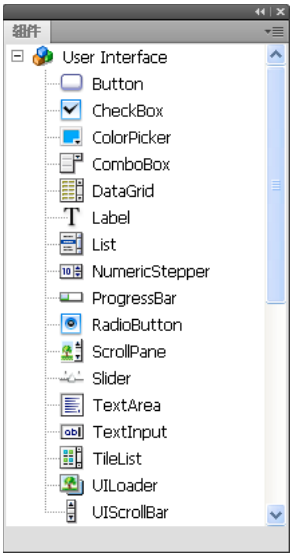
  如需 ActionScript 3.0 組件及其支援類別的完整清單，請參閱 ActionScript 3.0 參考。

檢視 **Flash** 組件：
您可以執行下列步驟來檢視 「組件」面板中的 Flash ActionScript 3.0 組件。

1 啟動 Flash。

2 建立新的 Flash 檔案 (ActionScript 3.0) 或開啟 「發佈設定」是指定為 ActionScript 3.0 的現有 Flash 文件。

3 如果 「組件」面板尚未開啟，請選取 「視窗 > 組件」加以開啟。



含有 「User Interface」組件的 「組件」面板

您也可以從 Adobe Exchange 下載其它組件，網址為 www.adobe.com/go/flash_exchange_tw。若要安裝從 Exchange 下載的組件，請下載並安裝位於 www.adobe.com/go/exchange_tw 的 Adobe® Extension Manager。按一下 「Adobe Exchange 首頁」連結，並尋找 「Extension Manager」連結。

任何組件都能出現在 Flash 的 「組件」面板中。請依照以下步驟在 Windows® 或 Macintosh® 電腦上安裝組件。

在 **Windows** 或 **Macintosh** 電腦上安裝組件：

1 退出 Flash。

2 將含有組件的 SWC 或 FLA 檔放到硬碟的下列資料夾中：

- 在 Windows 中：

  C:\Program Files\Adobe\Adobe Flash CS5\ 語言 \Configuration\Components

- 在 Macintosh 中：

  Macintosh HD:Applications:Adobe Flash CS5:Configuration:Components

3 啟動 Flash。

4 如果 「組件」面板尚未開啟，請選取 「視窗 > 組件」，檢視 「組件」面板中的組件。

如需有關組件檔案的詳細資訊，請參閱第 16 頁 「使用組件檔案」。

加入和刪除文件中的組件

當您將 FLA 架構的組件從 「組件」面板拖曳到 「舞台」時，Flash 會將一段可編輯的影片片段匯入元件庫中。當您將 SWC 架構的組件拖曳到 「舞台」時，Flash 則會將一段已編譯的片段匯入元件庫中。將組件匯入元件庫之後，您就可以將它的實體從 「元件庫」面板或 「組件」面板拖曳到 「舞台」上。

在編寫期間加入組件

您可以使用從 「組件」面板拖曳組件的方式，將組件加入到文件。您可以在 「屬性」檢測器或在 「組件檢測器」的 「參數」索引標籤中，設定每一個組件實體的屬性。

1 選取 「視窗 > 組件」。

2 按兩下 「組件」面板中的組件或是將組件拖曳到 「舞台」上。

3 在 「舞台」上選取組件。

4 如果沒有看到 「屬性」檢測器，請選取 「視窗 > 屬性 > 屬性」。

5 在 「屬性」檢測器中，為組件實體輸入實體名稱。

6 選取 「視窗 > 組件檢測器」，再選取 「參數」索引標籤以指定實體的參數。

如需詳細資訊，請參閱第 18 頁 「設定參數和屬性」。

7 視需要編輯寬度 (W:) 和高度 (H:) 的值，以變更組件的大小。

如需有關變更特定組件類型大小的詳細資訊，請參閱第 87 頁 「自訂使用者介面組件」。

8 選取 「控制 > 測試影片」或按 Control+Enter 鍵編譯文件並查看設定的結果。

您也可以變更組件的顏色和文字格式，方式是編輯組件的外觀元素，設定組件的樣式屬性或自訂它的外觀。如需有關這些主題的詳細資訊，請參閱第 87 頁 「自訂使用者介面組件」。

如果您在編寫期間將組件拖曳到 「舞台」上，就可以使用實體名稱 (例如 myButton) 來參考該組件。

使用 ActionScript 在階行階段加入組件

如果要使用 ActionScript 在執行階段將組件加入文件中，您必須在編譯 SWF 檔時先將組件放入應用程式的元件庫 (「視窗 > 元件庫」)。若要將組件加入元件庫中，請將組件從 「組件」面板拖曳到 「元件庫」面板。如需有關元件庫的詳細資訊，請參閱第 19 頁 「元件庫」。

您必須同時匯入組件的類別檔案，讓應用程式可以使用它的 API。組件類別檔案是安裝在 「套件」內，其中含有一個或多個類別。如果要匯入組件類別，請使用 import 陳述式並指定套件名稱和類別名稱。例如，要匯入 Button 類別可以使用下列 import 陳述式：

```
import fl.controls.Button;
```

如需組件所在的套件位置資訊，請參閱 ActionScript 3.0 參考。如需組件來源檔案位置的詳細資訊，請參閱第 16 頁 「使用組件檔案」。

如果要建立組件的實體，您必須叫用組件的 ActionScript 建構函式方法。例如，下列陳述式會建立名為 aButton 的 Button 實體：

```
var aButton:Button = new Button();
```

最後的步驟是呼叫靜態 addChild() 方法，將組件實體加入 「舞台」或應用程式容器中。例如，下列陳述式會加入 aButton 實體：

```
addChild(aButton);
```

在這個階段，您可以使用組件的 API 動態指定 「舞台」上組件的大小和位置、偵聽事件並設定屬性以修改它的行為。如需有關特定組件之 API 的詳細資訊，請參閱 ActionScript 3.0 參考。

如需有關 addChild() 方法的詳細資訊，請參閱第 22 頁 「使用顯示清單」。

刪除組件

如果要在編寫期間刪除 「舞台」上的組件實體，只需要選取它並按 Delete 鍵。這個動作將移除 「舞台」上的實體，但不會將組件從應用程式中移除。

如果要將已放入「舞台」或元件庫的組件從 Flash 文件中刪除，您必須從元件庫中刪除組件及其相關資源。只從「舞台」刪除組件是不夠的。如果沒有將組件從元件庫中移除，在編譯後它仍然會存在於應用程式中。

1 在「元件庫」面板中，選取組件的符號。

2 按一下「元件庫」面板底部的「刪除」按鈕，或從「元件庫」面板選單中選取「刪除」。

重複這些步驟，刪除與這個組件相關的任何資源。

如需在應用程式執行時從組件容器中移除組件的詳細資訊，請參閱第 23 頁「移除顯示清單中的組件」。

尋找組件的版本

Flash ActionScript 3.0 組件有一個 version 屬性，當需要提供給 Adobe Technical Support 或是需要知道正在使用的組件版本時，您可以顯示這個屬性。

顯示使用者介面組件的版本號碼：

1 建立新的 Flash 檔案 (ActionScript 3.0) 文件。

2 將組件拖曳至「舞台」上並指定它的實體名稱。例如，拖曳 ComboBox 到「舞台」上並將它命名為 **aCb**。

3 按 **F9** 鍵或選取「視窗 > 動作」開啟「動作」面板。

4 在主要「時間軸」上按一下「影格 1」，並將下列程式碼加入「動作」面板：

```
trace(aCb.version);
```

版本號碼 (如下圖所示) 應該會出現在「輸出」面板中。

對於 FLVPlayback 和 FLVPlaybackCaptioning 組件部分，您必須參考類別名稱而不是實體名稱，因為版本號碼是儲存在類別常數中。

顯示 **FLVPlayback** 和 **FLVPlaybackCaptioning** 組件的版本號碼：

1 建立新的 Flash 檔案 (ActionScript 3.0) 文件。

2 將 FLVPlayback 和 FLVPlaybackCaptioning 組件拖曳到「元件庫」面板。

3 按 **F9** 鍵或選取「視窗 > 動作」開啟「動作」面板。

4 在主要「時間軸」上按一下「影格 1」，並將下列程式碼加入「動作」面板。

```
import fl.video.*;
trace("FLVPlayback.VERSION: " + FLVPlayback.VERSION);
trace("FLVPLaybackCaptioning.VERSION: " + FLVPlaybackCaptioning.VERSION);
```

版本號碼會出現在「輸出」面板中。

ActionScript 3.0 事件處理模型

ActionScript 3.0 所引用的單一事件處理模型是用來取代舊版 ActionScript 所使用的事件處理機制。這個新的事件模型是以「文件物件模型第 3 層事件規格」(Document Object Model (DOM) Level 3 Events Specification) 為基礎。

新的規格對熟悉使用 ActionScript 2.0 addListener() 方法的開發人員而言，將有助於他們瞭解 ActionScript 2.0 事件偵聽程式模型與 ActionScript 3.0 事件模型之間的差異。下列清單將說明這兩個事件模型之間主要的不同點：

- 如果要在 ActionScript 2.0 中加入事件偵聽程式，在特定情況下請使用 addListener()，其它情況下則使用 addEventListener()，而在 ActionScript 3.0 中，請一律使用 addEventListener()。

- ActionScript 2.0 不使用事件流程，也就是只有廣播事件的物件才能呼叫 addListener() 方法，而在 ActionScript 3.0 中，只要是屬於事件流程的任何物件都可以呼叫 addEventListener() 方法。
- 在 ActionScript 2.0 中，事件偵聽程式可以是函數、方法或物件，而在 ActionScript 3.0 中，只有函數或方法可以做為事件偵聽程式。
- ActionScript 3.0 已不再支援 on(event) 語法，因此您不能將 ActionScript 事件程式碼附加至影片片段。您只能使用 addEventListener() 加入事件偵聽程式。

  下列範例會偵聽 Button 組件 aButton 上的 MouseEvent.CLICK 事件，藉以說明基本的 ActionScript 3.0 的事件處理模型：

```
aButton.addEventListener(MouseEvent.CLICK, clickHandler);
function clickHandler(event:MouseEvent):void {
trace("clickHandler detected an event of type: " + event.type);
trace("the event occurred on: " + event.target.name);
}
```

  如需有關 ActionScript 3.0 事件處理的詳細資訊，請參閱 「ActionScript 3.0 程式設計」。如需有關組件之 ActionScript 3.0 事件處理的詳細資訊，請參閱第 21 頁 「處理事件」。

簡單的應用程式

本節將使用 Flash 組件和 Flash 編寫工具，逐步引導您完成建立一個簡單的 ActionScript 3.0 應用程式所需的步驟。範例以兩種檔案形式提供，一種是 FLA 檔，另一種則是外部 ActionScript 類別檔案；前者在「時間軸」上含有 ActionScript 程式碼，後者則具有只將組件包含在元件庫中的 FLA 檔。一般而言，外部類別檔案是用於開發大型應用程式，可以在類別和應用程式之間共用程式碼，讓應用程式的維護工作更容易。如需有關使用 ActionScript 3.0 撰寫程式的詳細資訊，請參閱「ActionScript 3.0 程式設計」。

應用程式設計

我們的第一個 ActionScript 組件應用程式範例是標準 「Hello World」應用程式的變化，因此它的設計方式是相當簡單的：

- 應用程式的名稱是 Greetings。
- 它會使用 TextArea 起先顯示一段問候語 Hello World。
- 它會使用 ColorPicker 讓您可以變更文字的顏色。
- 它會使用三個 RadioButton 讓您可以將文字的大小設定為小、較大或最大。
- 它會使用 ComboBox 讓您可以從下拉式清單選取不同的問候語。
- 應用程式會使用 「組件」面板中的組件，同時也會利用 ActionScript 程式碼建立應用程式元素。

  完成定義後，就可以開始建置應用程式。

建立 Greetings 應用程式

下列建立 Greetings 應用程式的步驟是使用 Flash 編寫工具建立 FLA 檔、將組件放到「舞台」上，然後將 ActionScript 程式碼加入到時間軸。

在 **FLA** 檔中建立 **Greetings** 應用程式：

1. 請選取 「檔案 > 新增」。
2. 在 「新增文件」對話方塊中選取 「Flash 檔案 (ActionScript 3.0)」，然後按一下 「確定」。

   新的 Flash 視窗隨即開啟。

3 選取 「檔案 > 儲存檔案」、將 Flash 檔案命名為 **Greetings.fla**，再按一下 「儲存」按鈕。

4 在 「Flash 組件」面板中，選取 TextArea 組件並將它拖曳到 「舞台」上。

5 當 「舞台」上的 TextArea 已選取的情況下，在 「屬性」視窗中輸入 **aTa** 做為實體名稱，然後輸入下列資訊：

- 輸入 **230** 做為 W 值 (寬度)。
- 輸入 **44** 做為 H 值 (高度)。
- 輸入 **165** 做為 X 值 (水平位置)。
- 輸入 **57** 做為 Y 值 (垂直位置)。
- 在 「參數」索引標籤中，輸入 **Hello World!** 做為 text 參數。

6 將 ColorPicker 組件拖曳到 「舞台」上放在 TextArea 的左邊，然後將它的實體名稱設定為 **txtCp**。在 「屬性」檢測器中輸入下列資訊：

- 輸入 **96** 做為 X 值。
- 輸入 **72** 做為 Y 值。

7 逐一將三個 RadioButton 組件拖曳到 「舞台」上，並將它們的實體名稱設定為 **smallRb**、**largerRb** 和 **largestRb**。在 「屬性」檢測器中為它們輸入下列資訊：

- 分別輸入 **100** 和 **22** 做為這三個組件的 W 值和 H 值。
- 輸入 **155** 做為 X 值。
- 分別輸入 **120**、**148** 和 **175** 做為 smallRb、largerRb 和 largestRb 的 Y 值。
- 輸入 **fontRbGrp** 做為這三個組件的 groupName 參數。
- 在 「參數」索引標籤中，分別輸入 **Small**、**Larger** 和 **Largest** 做為這三個組件的標籤。

8 將 ComboBox 組件拖曳到 「舞台」上，並賦予它一個實體名稱 **msgCb**。在 「屬性」檢測器中為它輸入下列資訊：

- 輸入 **130** 做為 W 值。
- 輸入 **265** 做為 X 值。
- 輸入 **120** 做為 Y 值。
- 在 「參數」索引標籤中，輸入 **Greetings** 做為 prompt 參數。
- 按兩下 dataProvider 參數的文字欄位以開啟 「值」對話方塊。
- 按一下加號，以 **Hello World!** 取代標籤值。
- 重複上一個步驟，加入標籤值 **Have a nice day!** 和 **Top of the Morning!**。
- 按一下 「確定」，關閉 「值」對話方塊。

9 儲存檔案。

10 如果沒有開啟 「動作」面板，請按 **F9** 或從 「視窗」選單中選取 「動作」將它開啟。在主要 「時間軸」上按一下 「影格 1」，並在 「動作」面板中輸入下列程式碼：

```
import flash.events.Event;
import fl.events.ComponentEvent;
import fl.events.ColorPickerEvent;
import fl.controls.RadioButtonGroup;

var rbGrp:RadioButtonGroup = RadioButtonGroup.getGroup("fontRbGrp");
rbGrp.addEventListener(MouseEvent.CLICK, rbHandler);
txtCp.addEventListener(ColorPickerEvent.CHANGE,cpHandler);
msgCb.addEventListener(Event.CHANGE, cbHandler);
```

前三行程式碼會匯入應用程式所使用的事件類別。當使用者與其中一個組件進行互動時就會產生事件。底下五行程式碼會為應用程式要偵聽的事件註冊事件處理常式。當使用者按一下 RadioButton 時就會產生它的 click 事件。當使用者在 ColorPicker 中選取不同的顏色時就會產生 change 事件。當使用者從下拉式清單選擇不同的問候語時就會產生 ComboBox 的 change 事件。

第四行程式碼會匯入 RadioButtonGroup 類別讓應用程式可以將事件偵聽程式指定給 RadioButton 的群組，而不是個別將偵聽程式指定給每一個按鈕。

11 將下列程式碼行加入到 「動作」面板以建立 tf TextFormat 物件，讓應用程式可以使用它來變更 TextArea 中文字的 size 和 color 樣式屬性。

```
var tf:TextFormat = new TextFormat();
```

12 加入下列程式碼以建立 rbHandler 事件處理函數。這個函數會在使用者按一下其中一個 RadioButton 組件後處理產生的 click 事件。

```
function rbHandler(event:MouseEvent):void {
    switch(event.target.selection.name) {
        case "smallRb":
            tf.size = 14;
            break;
        case "largerRb":
            tf.size = 18;
            break;
        case "largestRb":
            tf.size = 24;
            break;
        }
        aTa.setStyle("textFormat", tf);
}
```

這個函數會使用 switch 陳述式檢查 event 物件的 target 屬性，判斷觸發事件的是哪一個 RadioButton。currentTarget 屬性含有觸發事件之物件的名稱。應用程式會根據使用者按下的 RadioButton 將 TextArea 中的文字大小變更為 14、18 或 24 點。

13 加入下列程式碼來實作 cpHandler() 函數，處理 ColorPicker 中值的變更動作：

```
function cpHandler(event:ColorPickerEvent):void {
    tf.color = event.target.selectedColor;
    aTa.setStyle("textFormat", tf);
}
```

這個函數會將 tf TextFormat 物件的 color 屬性設定為 ColorPicker 中選取的顏色，然後呼叫 setStyle() 將該顏色套用到 aTa TextArea 實體中的文字。

14 加入下列程式碼來實作 cbHandler() 函數，以處理對 ComboBox 中選取範圍所做的變更：

```
function cbHandler(event:Event):void {
    aTa.text = event.target.selectedItem.label;
}
```

這個函數只會以 ComboBox 中選取的文字 event.target.selectedItem.label 取代 TextArea 中的文字。

15 選取 「控制 > 測試影片」或按 Control+Enter 鍵，編譯程式碼並測試 Greetings 應用程式。

下一節將說明如何使用外部 ActionScript 類別和 FLA 檔 (在元件庫中只含有必要的組件) 建置相同的應用程式。

使用外部類別檔案建立 **Greetings2** 應用程式：

1 請選取 「檔案 > 新增」。

2 在 「新增文件」對話方塊中選取 「Flash 檔案 (ActionScript 3.0)」，然後按一下 「確定」。

新的 Flash 視窗隨即開啟。

3 選取 「檔案 > 儲存檔案」、將 Flash 檔案命名為 **Greetings2.fla**，再按一下 「儲存」按鈕。

4 將下列組件從 「組件」面板拖曳到元件庫：

- ColorPicker
- ComboBox
- RadioButton
- TextArea

編譯過的 SWF 檔將會使用每一項資源，因此您必須將它們加入到元件庫中。請將組件拖曳到 「元件庫」面板的底部。當您將這些組件加入元件庫時，也會自動加入其它的資源 (例如 List、TextInput 和 UIScrollBox)。

5 在 「屬性」視窗的 「文件類別」中，輸入 **Greetings2**。

如果 Flash 顯示 「找不到文件類別的定義」警告訊息，請忽略它。您將會在下列步驟中定義 Greetings2 類別。這個類別會定義應用程式主要的功能。

6 儲存 Greetings2.fla 檔案。

7 選取 「檔案 > 新增」。

8 在 「新增文件」對話方塊中選取 「ActionScript 檔案」，然後按一下 「確定」。

會開啟新的 Script 視窗。

9 在 Script 視窗中輸入下列程式碼：

```
package {
    import flash.display.Sprite;
    import flash.events.Event;
    import flash.events.MouseEvent;
    import flash.text.TextFormat;
    import fl.events.ComponentEvent;
    import fl.events.ColorPickerEvent;
    import fl.controls.ColorPicker;
    import fl.controls.ComboBox;
    import fl.controls.RadioButtonGroup;
    import fl.controls.RadioButton;
    import fl.controls.TextArea;
    public class Greetings2 extends Sprite {
        private var aTa:TextArea;
        private var msgCb:ComboBox;
        private var smallRb:RadioButton;
        private var largerRb:RadioButton;
        private var largestRb:RadioButton;
        private var rbGrp:RadioButtonGroup;
        private var txtCp:ColorPicker;
        private var tf:TextFormat = new TextFormat();
        public function Greetings2() {
```

此 Script 會定義一個名為 Greetings2 的 ActionScript 3.0 類別。Script 會執行下列動作：

- 匯入檔案中將使用的類別。一般的作法是在程式碼中參考到不同類別時，才會加入這些 import 陳述式，但為簡潔起見，範例在這一個步驟中就會將它們全部匯入。
- 宣告變數，表示將加入到程式碼的不同組件物件類型。另一個變數則會建立 tf TextFormat 物件。
- 定義類別的建構函數 Greetings2()。下列步驟會繼續將程式碼行加入這個函數，也會將其它方法加入到類別。

10 選取 「檔案 > 儲存檔案」，將檔案命名為 **Greetings2.as**，然後按一下 「儲存」按鈕。

11 將下列程式碼行加入到 Greeting2() 函數：

```
    createUI();
    setUpHandlers();
}
```

函數現在看起來會像下面這樣：

```
public function Greetings2() {
    createUI();
    setUpHandlers();
}
```

12 將下列各行程式碼加入到 Greeting2() 方法的右大括弧後面：

```
private function createUI() {
    bldTxtArea();
    bldColorPicker();
    bldComboBox();
    bldRadioButtons();
}
private function bldTxtArea() {
    aTa = new TextArea();
    aTa.setSize(230, 44);
    aTa.text = "Hello World!";
    aTa.move(165, 57);
    addChild(aTa);
}
private function bldColorPicker() {
    txtCp = new ColorPicker();
    txtCp.move(96, 72);
    addChild(txtCp);
}
private function bldComboBox() {
    msgCb = new ComboBox();
    msgCb.width = 130;
    msgCb.move(265, 120);
    msgCb.prompt = "Greetings";
    msgCb.addItem({data:"Hello.", label:"English"});
    msgCb.addItem({data:"Bonjour.", label:"Français"});
    msgCb.addItem({data:"¡Hola!", label:"Español"});
    addChild(msgCb);
}
private function bldRadioButtons() {
    rbGrp = new RadioButtonGroup("fontRbGrp");
    smallRb = new RadioButton();
    smallRb.setSize(100, 22);
```

```
    smallRb.move(155, 120);
    smallRb.group = rbGrp; //"fontRbGrp";
    smallRb.label = "Small";
    smallRb.name = "smallRb";
    addChild(smallRb);
    largerRb = new RadioButton();
    largerRb.setSize(100, 22);
    largerRb.move(155, 148);
    largerRb.group = rbGrp;
    largerRb.label = "Larger";
    largerRb.name = "largerRb";
    addChild(largerRb);
    largestRb = new RadioButton();
    largestRb.setSize(100, 22);
    largestRb.move(155, 175);
    largestRb.group = rbGrp;
    largestRb.label = "Largest";
    largestRb.name = "largestRb";
    addChild(largestRb);
}
```

這些程式碼行會執行下列動作：

- 實體化應用程式中使用的組件。
- 設定每個組件的大小、位置和屬性。
- 使用 addChild() 方法，將各個組件加入「舞台」中。

13 在 bldRadioButtons() 方法的右大括弧之後，加入下列 setUpHandlers() 方法的程式碼：

```
private function setUpHandlers():void {
    rbGrp.addEventListener(MouseEvent.CLICK, rbHandler);
    txtCp.addEventListener(ColorPickerEvent.CHANGE,cpHandler);
    msgCb.addEventListener(Event.CHANGE, cbHandler);
}
private function rbHandler(event:MouseEvent):void {
    switch(event.target.selection.name) {
        case "smallRb":
            tf.size = 14;
            break;
        case "largerRb":
            tf.size = 18;
            break;
        case "largestRb":
            tf.size = 24;
            break;
    }
    aTa.setStyle("textFormat", tf);
}
private function cpHandler(event:ColorPickerEvent):void {
    tf.color = event.target.selectedColor;
    aTa.setStyle("textFormat", tf);
}
private function cbHandler(event:Event):void {
    aTa.text = event.target.selectedItem.data;
}
}
}
```

這些函數會定義組件的事件偵聽程式。

14 選取 「檔案 > 儲存檔案」進行存檔。

15 選取 「控制 > 測試影片」或按 Control+Enter 鍵，編譯程式碼並測試 Greetings2 應用程式。

撰寫並執行後續的範例

撰寫並執行 Greetings 應用程式後，您應該已對如何執行本手冊中的其它程式碼範例有了基本的認識。每個範例中相關的 ActionScript 3.0 程式碼都會以反白標示並附上討論，您可以剪下本手冊中的每個範例內容，然後將它貼入 FLA 檔、編譯並執行。

第 3 章 使用組件

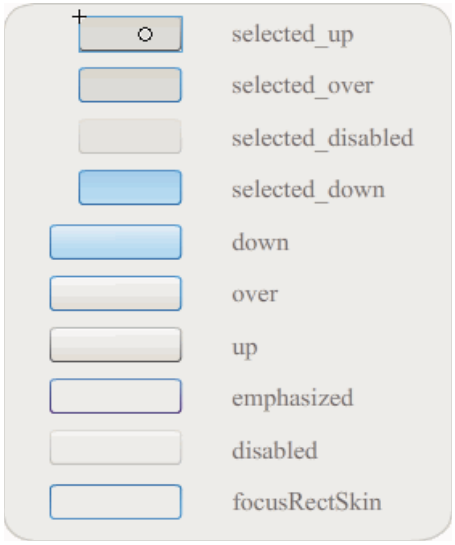
組件的架構

只有 Adobe® Flash Player 9.0.28.0 版和更新版本才能支援 Adobe® ActionScript® 3.0 組件。這些組件與使用 Flash CS4 之前版本所建立的組件不相容。如需有關使用 Adobe® ActionScript® 2.0 組件的詳細資訊，請參閱 「使用 Adobe® ActionScript® 2.0 組件」和 「Adobe® ActionScript® 2.0 組件語言參考」。

Adobe ActionScript 3.0 使用者介面 (UI) 組件的執行方式和 FLA 架構的組件一樣，但 Flash CS5 同時支援 SWC 和 FLA 架構的組件。例如，FLVPlayback 和 FLVPlaybackCaptioning 組件就是屬於 SWC 架構的組件。您可以將其中一種組件類型放入 「Components」資料夾，讓它出現在 「組件」面板中。這兩種組件類型的建立方式各有不同，因此在本章中將會分別說明這兩種組件。

ActionScript 3.0 FLA 架構的組件

ActionScript 3.0 的 「User Interface」組件是屬於 FLA 架構 (.fla) 的檔案，您可以按兩下 「舞台」上的組件，存取其內建外觀元素進行編輯。組件的外觀元素和其它資源都位於 「時間軸」的 「影格 2」上。當您按兩下組件時，Flash 會自動跳到 「影格 2」並開啟組件之外觀元素的面板。下圖顯示的面板具有會顯示 Button 組件的外觀元素。



Button 組件的外觀元素

如需有關組件外觀元素和自訂組件的詳細資訊，請參閱第 87 頁 「自訂使用者介面組件」和第 134 頁 「自訂 FLVPlayback 組件」。

為了加快應用程式的編譯速度並避免與 ActionScript 3.0 設定產生衝突，Flash CS5 FLA 架構的 UI 組件中也含有一個 SWC，其中包含組件的已編譯 ActionScript 程式碼。ComponentShim SWC 是置於 「舞台」上 「影格 2」的每個 「User Interface」組件中，如此才可以使用預先編譯的定義。組件必須位於 「舞台」上或在元件庫中，而且已在其 「連結」屬性中選取 「匯出在第一個影格」選項後，才可以使用 ActionScript。如果要使用 ActionScript 建立組件，還必須使用 import 陳述式匯入類別才能存取它。如需有關 import 陳述式的詳細資訊，請參閱 適用於 Adobe Flash Platform 的 ActionScript 3.0 參考。

SWC 架構的組件

SWC 架構的組件也含有 FLA 檔和 ActionScript 類別檔，但已編譯並匯出為 SWC。SWC 檔案是一組預先編譯的 Flash 元件和 ActionScript 程式碼套件，讓您不必重新編譯不會再變更的元件和程式碼。

FLVPlayback 和 FLVPlaybackCaptioning 組件就是屬於 SWC 架構的組件。它們是使用外部 (非內建) 的外觀元素。您可以變更 FLVPlayback 組件預設的外觀元素，方式是從預先設計的外觀元素集合中選取、自訂 「組件」面板中的 UI 控制項 (BackButton、BufferingBar 等等) 或建立自訂的外觀元素。如需詳細資訊，請參閱第 134 頁 「自訂 FLVPlayback 組件」。

在 Flash 中，您可以將影片片段轉換成已編譯的片段，如下列步驟所示：

編譯影片片段

- 在 「元件庫」面板中的影片片段上按一下右鍵 (Windows)，或 Control+ 按一下 (Macintosh)，然後選取 「轉換成編譯的影片」。

  編譯後的影片片段和編譯前的影片片段具有相同的行為模式，不過編譯後的影片片段的顯示和發佈速度比一般影片片段要快許多。編譯後的影片片段不能再進行編輯，不過它們的屬性會出現在 「屬性」檢測器和 「組件檢測器」中。

  SWC 組件包含已編譯的影片片段、組件的預先編譯 ActionScript 定義和其它用來描述組件的檔案。如果是建立自己的組件，您可以將它匯出成 SWC 檔案後再發佈。

匯出 **SWC** 檔案

- 選取 「元件庫」面板中的影片片段，並按一下右鍵 (Windows)，或是 Control+ 按一下 (Macintosh) 影片片段，然後選取 「匯出 SWC 檔案」。

  備註：Flash CS4 或更新版本 SWC 檔案的格式與 Flex SWC 格式相容，因此可以在這兩種產品間交換 SWC 檔案 (可能需要先修改)。

如需建立 SWC 架構之組件的詳細資訊，請參閱 www.adobe.com/go/learn_fl_creating_components_tw。

ActionScript 3.0 組件 API

每一個 ActionScript 3.0 組件都是建立在 ActionScript 3.0 類別上，位於套件資料夾且名稱的格式為 fl.packagename.classname。例如，Button 組件是 Button 類別的實體而套件名稱是 fl.controls.Button。在應用程式中匯入組件類別時必須參考套件名稱。您可以使用下列陳述式匯入 Button 類別：

```
import fl.controls.Button;
```

如需有關組件類別檔案位置的詳細資訊，請參閱第 16 頁 「使用組件檔案」。

組件的類別定義了方法、屬性、事件和樣式，可以讓您在應用程式中與它進行互動。ActionScript 3.0 UI 組件是 Sprite 和 UIComponent 類別的子類別，而且繼承了它們的屬性、方法和事件。Sprite 類別是基本的顯示清單建構區塊，它類似於 MovieClip 但不使用時間軸。UIComponent 類別是所有互動式和非互動式視覺組件的基底類別。適用於 Adobe Flash Platform 的 ActionScript 3.0 參考中說明了每個組件的繼承路徑，以及其屬性、方法、事件和樣式。

所有的 ActionScript 3.0 組件都是使用 ActionScript 3.0 事件處理模型。如需有關事件處理的詳細資訊，請參閱第 21 頁 「處理事件」以及 「ActionScript 3.0 程式設計」。

使用組件檔案

本節將說明儲存組件檔案的位置、如何尋找 ActionScript 來源檔案以及如何加入和移除 「組件」面板中的組件。

儲存組件檔案的位置

Flash 組件儲存在應用程式層級的 Configuration 資料夾中。

備註：如需關於這些資料夾的資訊，請參閱 「使用 Flash」中的 「隨著 Flash 安裝的 Configuration 資料夾」。

組件安裝在下列位置：

- Windows 2000 或 Windows XP：C:\Program Files\Adobe\Adobe Flash CS5\ 語言 \Configuration\Components
- Mac OS X：Macintosh HD:Applications:Adobe Flash CS5:Configuration:Components

  在 Components 資料夾內，「User Interface」 (UI) 組件是儲存在 User Interface.fla 檔案中，而 FLVPlayback (FLVPlaybackAS3.swc) 和 FLVPlaybackCaptioning 組件則是位於 Video 資料夾內。

  您也可以將組件儲存在下列使用者相關的位置：

- Windows 2000 或 Windows XP：C:\Documents and Settings\ 使用者名稱 \Local Settings\Application Data\Adobe\Adobe Flash CS5\zh_tw\Configuration\Components
- Windows Vista：C:\Users\ 使用者名稱 \Local Settings\Application Data\Adobe\Adobe Flash CS5\zh_tw\Configuration\Components

  備註：在 Windows 中，會根據預設隱藏 Application Data 資料夾。若要顯示隱藏的資料夾和檔案，請選取 「我的電腦」開啟 「Windows 檔案總管」，選取 「工具 > 資料夾選項」，然後選取 「檢視」索引標籤。在 「檢視」索引標籤下，選取 「顯示隱藏的檔案和資料夾」選項按鈕。

- Mac OS X：Macintosh HD:Users:< 使用者名稱 >:Library:Application Support:Adobe Flash CS5:Configuration:Components

儲存組件來源檔案的位置

組件的 ActionScript (.as) 類別檔案 (或 「來源檔案」) 是安裝在 Windows 2000 或 Windows XP 的下列應用程式資料夾內：

User Interface 組件　C:\Program Files\Adobe\Adobe Flash CS5\zh_tw\Configuration\Component Source\ActionScript 3.0\User Interface\fl

FLVPlayback　C:\Program Files\Adobe\Adobe Flash CS5\zh_tw\Configuration\Component Source\ActionScript 3.0\FLVPlayback\fl\video

FLVPlaybackCaptioning　C:\Program Files\Adobe\Adobe Flash CS5\zh_tw\Configuration\Component Source\ActionScript 3.0\FLVPlaybackCaptioning\fl\video

在 Mac OS X 上，組件來源檔案的位置是：

User Interface 組件　Macintosh HD:Applications:Adobe Flash CS5:Configuration:Component Source:ActionScript 3.0:User Interface:fl

FLVPlayback　Macintosh HD:Applications:Adobe Flash CS5:Configuration:Component Source:ActionScript 3.0:FLVPlayback:fl:video

FLVPlaybackCaptioning　Macintosh HD:Applications:Adobe Flash CS5:Configuration:Component Source:ActionScript 3.0:FLVPlaybackCaptioning:fl:video

組件來源檔案和類別路徑

由於 ActionScript 3.0 組件是已經完成程式碼編譯的組件，因此不能在 Classpath 變數中指定 ActionScript 類別檔案的位置。如果沒有在 Classpath 中指定它們的位置，將會增加編譯應用程式所需的時間。不過，如果 Flash 在 「類別路徑」設定中找到組件類別檔案，類別檔案將永遠優先於組件的已編譯程式碼。

在 「類別路徑」設定中加入組件來源檔案位置的時機是在對含有組件的應用程式進行除錯的期間。如需詳細資訊，請參閱 第 18 頁 「除錯組件應用程式」。

修改組件檔案

在更新、加入或移除 SWC 架構的組件或加入新 FLA 架構的組件到 Flash 之後，您必須將它們重新載入到 「組件」面板才可以使用它們。您可以重新啟動 Flash 或選取 「組件」面板選單中的 「重新載入」將組件重新載入。如此 Flash 就可以選取您已經加入 Components 資料夾的任何組件。

在執行 **Flash** 時重新載入組件面板中的組件：

- 從 「組件」面板選單中選取 「重新載入」。

從組件面板移除組件：

- 從 Components 資料夾移除 FLA、SWC 或 MXP 檔案，然後重新啟動 Flash 或選取 「組件」面板選單中的 「重新載入」。MXP 檔案是指從 Adobe Exchange 下載的組件檔案。

  您可以在 Flash 正在執行時移除及取代 SWC 架構的組件，然後重新載入以反映變更，但如果變更或刪除的是 FLA 架構的組件，則必須等到終止並重新啟動 Flash 後才會反映變更。不過，您可以使用 「重新載入」命令，新增 FLA 架構的組件並將它們載入。

  Adobe 建議您對想要變更的任何 Flash 組件檔案 (.fla 或 .as) 都先製作一份副本。必要時就可以將它還原。

除錯組件應用程式

ActionScript 3.0 組件含有本身的來源程式碼，可以減少編譯應用程式所需的編譯時間。然而，Flash 除錯程式無法檢查已編譯影片片段內部的程式碼。因此，如果要對應用程式的組件來源程式碼進行除錯，您必須在 「類別路徑」設定中加入組件來源檔案。

組件套件資料夾的位置是相對於組件類型來源檔案的位置。如果要參考所有 UI 組件的 ActionScript 3.0 來源檔案，請在 「使用者介面」套件的 「類別路徑」中加入下列位置：

- $(AppConfig)/Component Source/ActionScript 3.0/User Interface

備註：這個動作將會覆寫所有 UI 組件的已編譯程式碼，並增加編譯應用程式所需的時間。如果因為任何原因而變更組件的來源檔案，結果可能會使組件產生不同的行為。

如果要設定 「類別路徑」，請選取 「編輯」選單中的 「偏好設定」，然後選取 「類別」清單中的 「ActionScript」並按一下 「ActionScript 3.0 設定」按鈕。如果要加入新的項目，請按一下視窗上方的加號以顯示目前的設定。

$(AppConfig) 變數會從您安裝 Flash CS5 的位置參考 Flash CS5 的 Configuration 資料夾。一般而言，路徑看起來會像這樣：

- Windows 2000 或 Windows XP：C:\Program Files\Adobe\Adobe Flash CS5\ 語言 \Configuration\
- Mac OS X：Macintosh HD:Applications:Adobe Flash CS5:Configuration

備註：如果必須變更組件來源檔案，Adobe 強烈建議您先將原始來源檔案複製到不同的位置，然後再將這個位置加入到您的 「類別路徑」。

如需有關組件來源檔案位置的詳細資訊，請參閱第 17 頁 「儲存組件來源檔案的位置」。

設定參數和屬性

每個組件都有參數，您可以設定這些參數以變更組件的外觀和行為指令。參數是組件的類別的屬性，並且會出現在 「屬性」檢測器和 「組件檢測器」中。最常使用的屬性會顯示為編寫參數；其它參數則必須以 ActionScript 設定。所有在編寫時可以設定的參數，也都可以用 ActionScript 來設定。用 ActionScript 設定的參數會覆寫任何在編寫時所設定的值。

大部分 ActionScript 3.0「User Interface」組件都會繼承 UIComponent 類別以及基底類別的屬性和方法。例如，Button 和 CheckBox 類別會同時繼承 UIComponent 類別和 BaseButton 類別的屬性。組件所繼承的屬性以及它本身的類別屬性都可以供您存取。例如，ProgressBar 組件會繼承 UIComponent 的 ProgressBar.enabled 屬性，而它本身也有 ProgressBar.percentComplete 屬性。您可以同時存取這些屬性與 ProgressBar 組件的實體進行互動。如需有關組件屬性的詳細資訊，請參閱它在 ActionScript 3.0 參考中的類別項目。

您可以使用 「屬性」檢測器或 「組件檢測器」來設定組件實體的參數。

在屬性檢測器中輸入組件的實體名稱：

1 選取 「視窗 > 屬性 > 屬性」。

2 選取 「舞台」上的組件實體。

3 在 < 實體名稱 > 方塊 (位於 「影片片段」下拉式清單下方) 中輸入組件實體的名稱。或是按一下 「參數」索引標籤，並在「組件」一字下方的文字方塊中輸入名稱。輸入您要設定的參數值。

針對實體名稱增加字尾是一種指示組件類型的良好做法：如此會使閱讀您的 ActionScript 程式碼更為容易。例如，**licenseSb** 實體名稱會識別本身為捲軸的組件，而它會捲動 **licenseTa** 文字區域中的授權合約。

在組件檢測器中輸入組件實體的參數：

1 選取 「視窗 > 組件檢測器」。

2 選取 「舞台」上的組件實體。

3 按一下 「參數」索引標籤並為列出的參數輸入值。

組件檢測器中的組件參數

在 ActionScript 中設定組件屬性

在 ActionScript 中，會使用點 (.) 運算子 (「點語法」) 來存取 「舞台」上屬於物件或實體的屬性或方法。點語法運算式開始時要先寫實體的名稱，接著加上一個點，然後以所要指定的元素來結束。例如，下列 ActionScript 程式碼會設定 CheckBox 實體 aCh 的 width 屬性，使它的寬度變成 50 個像素：

```
aCh.width = 50;
```

下列 if 陳述式會檢查使用者是否已經選取核取方塊：

```
if (aCh.selected == true) {
    displayImg(redCar);
}
```

元件庫

當您第一次將組件加入到文件時，Flash 會以影片片段的格式將它匯入 「元件庫」面板。您也可以直接將 「組件」面板中的組件拖曳到 「元件庫」面板，然後將它的實體加入 「舞台」中。無論是哪一種作法，您都必須先將組件加入元件庫，才可以存取它的類別元素。

如果您將組件加入元件庫並使用 ActionScript 建立它的實體，則必須先使用 import 陳述式匯入它的類別。在 import 陳述式中，必須同時指定組件的套件名稱和它的類別名稱。例如，下列陳述式會匯入 Button 類別：

```
import fl.controls.Button;
```

當您將組件放入元件庫時，Flash 也會匯入組件的資源資料夾，其中包含組件在不同狀態下的外觀元素。組件的「外觀元素」是由元件集合所組成，這些元件會構成組件在應用程式中的圖像顯示。單一外觀元素是表示組件的特定狀態的圖像表示方式或影片片段。

Component Assets 資料夾中的內容可以讓您在需要時變更組件的外觀元素。如需詳細資訊，請參閱第 87 頁「自訂使用者介面組件」。

將組件加入元件庫之後，您就可以將它的圖示從「組件」面板或「元件庫」面板拖曳到「舞台」，將更多的組件實體加入文件中。

調整組件大小

請使用「自由變形」工具或 setSize() 方法調整組件實體的大小。您可以從任何組件實體呼叫 setSize() 方法 (請參閱 UIComponent.setSize()) 以調整其大小。下列程式碼會將 List 組件實體的大小調整為 200 像素寬和 300 像素高：

```
aList.setSize(200, 300);
```

組件不會自動調整大小來配合其標籤。如果已加入文件中的組件實體不夠大，無法完整顯示其標籤，則會裁切標籤文字。您必須調整組件大小來配合其標籤。

如需有關調整大小的詳細資訊，請參閱 適用於 Adobe Flash Platform 的 ActionScript 3.0 參考中的個別項目。

即時預覽

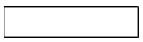
預設為啟用的「即時預覽」功能，可讓您在「舞台」上預覽將出現在發佈 Flash 內容中的組件。這些組件會以概略的大小出現。

開啟或關閉即時預覽：

- 選取「控制 > 啟用即時預覽」。選項旁邊的核取標記代表功能已經啟動。

對於不同的組件，即時預覽會反映不同的參數。如需有關在即時預覽中所反映組件參數的詳細資訊，請參閱 適用於 Adobe Flash Platform 的 ActionScript 3.0 參考中的每個組件項目。

啟用「即時預覽」的 Button 組件。



停用「即時預覽」的 Button 組件。

「即時預覽」中的組件不具功能。若要測試組件功能，必須使用「控制 > 測試影片」命令。

處理事件

每個組件會在使用者與它進行互動時廣播事件。例如，當使用者按一下 Button 後，它會送出 MouseEvent.CLICK 事件，而當使用者選取 「清單」中的項目時，該 「清單」也會送出事件。CHANGE 事件。當組件發生重大事項時也會產生事件，例如完成載入 UILoader 實體的內容時就會產生 Event.COMPLETE 事件。若要處理事件，您可以撰寫 ActionScript 程式碼，在事件發生時加以執行。

組件的事件包括任何組件繼承之類別所產生的事件。這就表示所有 ActionScript 3.0 「User Interface」組件都會繼承 UIComponent 類別的事件，因為它是 ActionScript 3.0 「User Interface」組件的基底類別。若要查看組件廣播的事件清單，請參閱 適用於 Adobe Flash Platform 的 ActionScript 3.0 參考中的 Events 區段。

如需 ActionScript 3.0 事件處理的完整說明，請參閱 「ActionScript 3.0 程式設計」。

關於事件偵聽程式

下列重點適用於處理 ActionScript 3.0 組件的事件：

- 所有事件都由組件類別的實體進行廣播。這個組件實體就是 「廣播程式」。
- 註冊事件 「偵聽程式」的方式是為組件實體呼叫 addEventListener() 方法。例如，下列程式碼行會為 Button 實體 aButton 加入 MouseEvent.CLICK 事件的偵聽程式：

  ```
  aButton.addEventListener(MouseEvent.CLICK, clickHandler);
  ```

  addEventListener() 方法的第二個參數會註冊事件發生時所要呼叫的 clickHandler 函數名稱。這個函數也稱為 「回呼函數」。
- 您可以將多個偵聽程式註冊到一個組件實體。

  ```
  aButton.addEventListener(MouseEvent.CLICK, clickHandler1);
  aButton.addEventListener(MouseEvent.CLICK, clickHandler2);
  ```
- 您可以將一個偵聽程式註冊到多個組件實體。

  ```
  aButton.addEventListener(MouseEvent.CLICK, clickHandler1);
  bButton.addEventListener(MouseEvent.CLICK, clickHandler1);
  ```
- 事件處理常式函數傳遞目標的事件物件中包含事件類型以及廣播事件之實體的相關資訊。如需詳細資訊，請參閱第 21 頁 「關於事件物件」。
- 在終止應用程式或是使用 removeEventListener() 方法明確地將偵聽程式移除之前，偵聽程式仍會保持在作用中。例如，下列程式碼行會移除 aButton 的 MouseEvent.CLICK 事件偵聽程式：

  ```
  aButton.removeEventListener(MouseEvent.CLICK, clickHandler);
  ```

關於事件物件

事件物件是繼承自 Event 物件類別，而且其屬性中包含已發生的事件資訊，包括提供必要事件資訊的 target 和 type 屬性：

屬性	說明
type	字串，表示事件的類型。
target	廣播事件之組件實體的參考。

當事件具有其他屬性時，這些屬性會列在 適用於 Adobe Flash Platform 的 ActionScript 3.0 參考中的事件類別說明。

事件物件是在事件發生時自動產生的，並且會傳遞給事件處理常式函數。

您可以使用函數內的事件物件來存取已廣播事件的名稱，或是廣播事件的組件實體名稱。您可以從實體名稱存取其它組件屬性。例如，下列程式碼會使用 evtObj 事件物件的 target 屬性來存取 aButton 的 label 屬性，並且將它顯示在 「輸出」面板中：

```
import fl.controls.Button;
import flash.events.MouseEvent;

var aButton:Button = new Button();
aButton.label = "Submit";
addChild(aButton);
aButton.addEventListener(MouseEvent.CLICK, clickHandler);

function clickHandler(evtObj:MouseEvent){
trace("The " + evtObj.target.label + " button was clicked");
}
```

使用顯示清單

所有 ActionScript 3.0 組件都是繼承自 DisplayObject 類別，因此，存取它的方法和屬性就可以和顯示清單進行互動。「顯示清單」是已顯示物件和應用程式中視覺元素的階層架構。這個階層架構包括下列元素：

- 舞台，即最上層容器
- 顯示物件，包括形狀、影片片段和文字欄位的其中一項
- 顯示物件容器，可以包含子顯示物件的特殊顯示物件類型。

顯示清單中的物件順序決定了它們在父容器中的深度。物件的深度代表該物件在 「舞台」上或其顯示容器中由上到下或由前到後的位置。物件重疊時就可以看出深度的順序，但物件不重疊時仍然可以看出深度的順序。顯示清單中的每個物件都有一個在「舞台」上的對應深度。如果要將物件放到最前面，或是移到其它物件後面以變更物件的深度，您必須變更它在顯示清單中的位置。顯示清單中預設的物件順序是之前將它們放入 「舞台」時的順序。顯示清單中的位置 0 表示位在深度順序底部的物件。

將組件加入到顯示清單

您可以呼叫容器的 addChild() 或 addChildAt() 方法，將物件加入 DisplayObjectContainer 物件中。如果是在 「舞台」上，您可以先建立物件，然後在編寫期間將它加入到它的顯示清單中；如果是加入組件，則可以將它從 「組件」面板拖曳到 「舞台」上。如果要使用 ActionScript 將物件加入到容器，請先使用 new 運算子叫用物件的建構函式來建立它的實體，然後呼叫 addChild() 或 addChildAt() 方法將它放到 「舞台」上和顯示清單中。addChild() 方法會將物件放入顯示清單中的下一個位置上，而 addChildAt() 則是指定要加入物件的位置。如果指定的位置已經有物件佔用，這個位置上以及它上方的物件將往上移動 1 個位置。DisplayObjectContainer 物件的 numChildren 屬性則包含它所含有的顯示物件數目。從顯示清單擷取物件的方式是，呼叫 getChildAt() 方法並指定位置，或是在知道物件的名稱時呼叫 getChildByName() 方法。

備註：如果想透過顯示清單中的名稱來存取物件，在使用 ActionScript 加入組件時就必須將名稱指定給它的 name 屬性。

下列範例會從顯示清單中顯示三個組件的名稱和位置。首先，將 NumericStepper、Button 和 ComboBox 拖曳到 「舞台」上使它們互相重疊，並設定 **aNs**、**aButton** 和 **aCb** 的實體名稱。然後，在 「時間軸」之 「影格 1」的 「動作」面板中，加入下列程式碼：

```
var i:int = 0;
while(i < numChildren) {
    trace(getChildAt(i).name + " is at position: " + i++);
}
```

您應該會在 「輸出」面板中看到下列結果：

```
aNs is at position: 0
aButton is at position: 1
aCb is at position: 2
```

移動顯示清單中的組件

您可以呼叫 addChildAt() 方法並提供物件的名稱以及您要放入物件的位置做為方法的參數，變更物件在顯示清單和顯示深度中的位置。例如，將下列程式碼加入上一個範例就會將 NumericStepper 放到頂端，再循環執行一次則會顯示組件在顯示清單中的新位置：

```
this.addChildAt(aNs, numChildren - 1);
i = 0;
while(i < numChildren) {
    trace(getChildAt(i).name + " is at position: " + i++);
}
```

您應該會在「輸出」面板中看到下列結果：

```
aNs is at position: 0
aButton is at position: 1
aCb is at position: 2
aButton is at position: 0
aCb is at position: 1
aNs is at position: 2
```

NumericStepper 也應該會出現在螢幕上其它組件的前面。

請注意，numChildren 是顯示清單中物件的數目 (從 1 到 n)，而清單中的第一個位置是 0。因此當清單中有三個物件時，第三個物件的索引位置即是 2。也就是 numChildren - 1 可以用來表示顯示清單的最後一個位置，以顯示深度而言，則是最頂端的物件。

移除顯示清單中的組件

您可以使用 removeChild() 和 removeChildAt() 方法，從顯示物件容器和組件的顯示清單中將組件移除。下列範例會將三個 Button 組件重疊放到「舞台」上，並為每個組件加入事件偵聽程式。當您按一下每個 Button 後，事件處理常式就會將它從顯示清單和「舞台」上移除。

1. 建立新的 Flash 檔案 (ActionScript 3.0) 文件。
2. 將 Button 從「組件」面板拖曳到「元件庫」面板。
3. 開啟「動作」面板，選取主要「時間軸」中的「影格 1」，然後加入下列程式碼：

```
import fl.controls.Button;

var i:int = 0;
while(i++ < 3) {
    makeButton(i);
}
function removeButton(event:MouseEvent):void {
    removeChildAt(numChildren -1);
}
function makeButton(num) {
    var aButton:Button = new Button();
    aButton.name = "Button" + num;
    aButton.label = aButton.name;
    aButton.move(200, 200);
    addChild(aButton);
    aButton.addEventListener(MouseEvent.CLICK, removeButton);
}
```

如需顯示清單的完整說明，請參閱「ActionScript 3.0 程式設計」中的「顯示程式設計」。

使用 FocusManager

當使用者在 Flash 應用程式中按下 Tab 鍵進行瀏覽，或在應用程式中以滑鼠選擇時，FocusManager 類別會決定哪個組件將成為輸入焦點。除非您正在建立組件，否則就不需要將 FocusManager 實體加入應用程式中，或是撰寫任何程式碼來啟動 FocusManager。

如果 RadioButton 物件成為焦點，FocusManager 會檢查該物件及所有具有相同 groupName 值的物件，並設定物件的焦點，將 selected 屬性設定為 true。

每個強制回應 Window 組件都含有 FocusManager 的實體，因此該視窗上的控制項會變成本身的 tab 集。如此便能防止使用者因為按下 Tab 鍵，不小心瀏覽到其它視窗中的組件。

FocusManager 會使用容器中元素的深度階層 (或 z 順序) 做為預設的瀏覽配置或「定位鍵迴圈」。使用者通常使用 Tab 鍵來瀏覽定位鍵迴圈，將焦點從成為焦點的第一個組件移到最後一個組件，然後再回到第一個組件。深度階層的設定主要以組件拖曳到「舞台」的順序為準。不過，您也可以使用「修改 > 排列 > 移至最前面 / 移至最後面」命令來決定最後的 z 順序。如需深度階層的詳細資訊，請參閱第 22 頁 「使用顯示清單」。

您可以呼叫 setFocus() 方法，讓應用程式中的組件實體成為焦點。例如，下列範例會為目前的容器 (this) 建立 FocusManager 實體，並讓 Button 實體 aButton 成為焦點。

```
var fm:FocusManager = new FocusManager(this);
fm.setFocus(aButton);
```

您可以透過呼叫 getFocus() 方法判斷定位鍵迴圈中成為焦點的組件，也可以透過呼叫 getNextFocusManagerComponent() 方法判斷定位鍵迴圈中將成為焦點的下一個組件。在下列範例中，CheckBox、RadioButton 和 Button 位於「舞台」上，而且每個組件都有 MouseEvent.CLICK 和 FocusEvent.MOUSE_FOCUS_CHANGE 事件的偵聽程式。發生 MouseEvent.CLICK 事件時，由於使用者已按一下組件，因此 showFocus() 函數會呼叫 getNextFocusManagerComponent() 方法來判斷定位鍵迴圈中將成為焦點的下一個組件。接著，它會呼叫 setFocus() 方法，讓該組件成為焦點。發生 FocusEvent.MOUSE_FOCUS_CHANGE 事件時，fc() 函數會顯示發生此事件的組件名稱。當使用者按一下定位鍵迴圈中的組件，而該組件並非下一個組件時，就會觸發這個事件。

```
// This example assumes a CheckBox (aCh), a RadioButton (aRb) and a Button
// (aButton) have been placed on the Stage.

import fl.managers.FocusManager;
import flash.display.InteractiveObject;

var fm:FocusManager = new FocusManager(this);

aCh.addEventListener(MouseEvent.CLICK, showFocus);
aRb.addEventListener(MouseEvent.CLICK, showFocus);
aButton.addEventListener(MouseEvent.CLICK, showFocus);
aCh.addEventListener(FocusEvent.MOUSE_FOCUS_CHANGE, fc);
aRb.addEventListener(FocusEvent.MOUSE_FOCUS_CHANGE, fc);
aButton.addEventListener(FocusEvent.MOUSE_FOCUS_CHANGE, fc);

function showFocus(event:MouseEvent):void {
    var nextComponent:InteractiveObject = fm.getNextFocusManagerComponent();
    trace("Next component in tab loop is: " + nextComponent.name);
    fm.setFocus(nextComponent);
}

function fc(fe:FocusEvent):void {
    trace("Focus Change: " + fe.target.name);
}
```

若要建立在使用者按下 Enter 鍵 (Windows) 或 Return 鍵 (Macintosh) 時成為焦點的按鈕，請將 FocusManager.defaultButton 屬性設定為想要設成預設 Button 的 Button 實體，如下列程式碼所示：

```
import fl.managers.FocusManager;

var fm:FocusManager = new FocusManager(this);
fm.defaultButton = okButton;
```

FocusManager 類別會覆寫預設的 Flash Player 焦點矩形，並繪製自訂的圓角焦點矩形。

如需有關在 Flash 應用程式中建立焦點配置的詳細資訊，請參閱 Flash Professional 的 ActionScript 3.0 參考中的 FocusManager 類別。如果要建立自訂焦點管理員，您必須建立會實作 IFocusManager 介面的類別。如需詳細資訊，請參閱 Flash Professional 的 ActionScript 3.0 參考中的 IFocusManager。

使用 List 架構的組件

List、DataGrid 和 TileList 組件都是繼承自 SelectableList 基底類別。因此，可以將這些組件視為 List 架構的組件。ComboBox 是由文字方塊和 List 組成，因此，它也是一個 List 架構的組件。

List 是由列組成。DataGrid 和 TileList 是由可以分成多個欄的列所組成。列和欄的交集處是儲存格。List 的列只有單一欄，每一列就是一個儲存格。儲存格有下列兩個要點：

- 儲存格容納的資料值稱為項目。「項目」是在 List 中用來儲存資訊單元的 ActionScript 物件。您可以把 List 想像成一個陣列，陣列中的每一個索引空間就是一個項目。List 中的項目通常是具有 label 屬性 (用於顯示) 和 data 屬性 (用於儲存資料) 的物件。「資料提供者」是 List 中項目的資料模型。資料提供者可以讓您填入 List 架構的組件，只需將它指定給組件的 dataProvider 屬性即可。
- 儲存格可以容納不同類型的資料，從文字到影像、影片片段或是您可以建立的任何類別都可以容納。因此，在繪製或顯示儲存格時必須使用適合其內容的方法。所以，List 架構的組件會使用 「儲存格輸出器」來顯示它的儲存格。而在 DataGrid 方面，每一欄就是一個 DataGridColumn 物件，它同樣也具有 cellRenderer 屬性，可以用來配合它的內容以顯示每一欄。

  所有 List 架構的組件都具有 cellRenderer 和 dataProvider 屬性，您可以設定這兩個屬性以載入和顯示這些組件的儲存格。如需使用這些屬性和 List 架構之組件的詳細資訊，請參閱第 25 頁 「使用 DataProvider」和第 32 頁 「使用 CellRenderer」。

使用 DataProvider

DataProvider 是您可以用來提供資料給 ComboBox、DataGrid、List 和 TileList 組件的資料來源。這些組件類別都具有 dataProvider 屬性，您可以用來指定 DataProvider 物件需使用資料來填入組件的儲存格。一般而言，資料提供者是屬於 Array 或 XML 物件之類的資料集合。

建立 DataProvider

對於 ComboBox、List 和 TileList 組件，您可以在編寫環境中使用 dataProvider 參數來建立 DataProvider。DataGrid 組件在 「屬性」檢測器中並無 dataProvider 參數，因為它可能包含多個欄，而使它的資料提供者更形複雜。您也可以使用 ActionScript 建立這些組件以及 DataGrid 的 DataProvider。

使用 dataProvider 參數

您可以在 「屬性」檢測器或 「組件檢測器」中，按一下 「參數」索引標籤上的 dataProvider 參數，為 ComboBox、List 和 TileList 組件建立簡單的資料提供者。

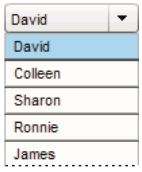
如果您按兩下值儲存格 (一開始顯示的是空白 「陣列」)，將會開啟 「值」對話方塊，讓您可以輸入多個標籤和資料值來建立資料提供者。

dataProvider 的「值」對話方塊

按一下加號可以將項目加入到 dataProvider。按一下減號可以刪除項目。按一下向上鍵可以將清單中選取的項目往上移動一個位置；按一下向下鍵可以將清單中選取的項目往下移動一個位置。下圖顯示可以建立孩童姓名和生日清單的「值」對話方塊。

包含資料的「值」對話方塊

您建立的 Array 是由一對標籤和值欄位組成。標籤欄位是 label 和 data，而值欄位則是孩童的姓名和他們的生日。標籤欄位可以辨識 List 中顯示的內容，在這個範例中是孩童的姓名。最後的 ComboBox 看起來會像這樣：



由 DataProvider 填入的 ComboBox

完成加入資料後，請按一下「確定」關閉對話方塊。dataProvider 參數中的 Array 現在已經填入您建立的項目。

allowMultipleSelection	false
dataProvider	[{label:David,data:1995/11/19},{label:Colleen,data:1993/4/20},{label:Sharon,data:1997/9/6},
enabled	false
horizontalLineScrollSize	1
horizontalPageScrollSize	0
horizontalScrollPolicy	auto
verticalLineScrollSize	1

包含資料的 dataProvider 參數

使用 ActionScript 存取組件的 dataProvider 屬性就可以存取您建立的標籤和資料值。

使用 ActionScript 建立 DataProvider

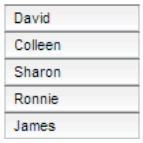
您可以在 Array 或 XML 物件中建立資料，並提供該物件做為 DataProvider 建構函式的 value 參數，以便建立 DataProvider。

備註：在 ActionScript 3.0 中，不可以直接將 Array 或 XML 物件指定給 dataProvider 屬性，因為屬性已經定義為 DataProvider 物件而且只能接受 DataProvider 類型的物件。

下列範例會填入 List 組件，它的列只有單一欄，包括幾個孩童的姓名和生日。此範例會定義 items Array 中的清單，在清單建立 DataProvider 實體 (new DataProvider(items)) 時提供它做為參數，然後將它指定給 List 組件的 dataProvider 屬性。

```
import fl.controls.List;
import fl.data.DataProvider;

var aList:List = new List();
var items:Array = [
{label:"David", data:"11/19/1995"},
{label:"Colleen", data:"4/20/1993"},
{label:"Sharon", data:"9/06/1997"},
{label:"Ronnie", data:"7/6/1993"},
{label:"James", data:"2/15/1994"},
];
aList.dataProvider = new DataProvider(items);
addChild(aList);
aList.move(150,150);
```

Array 是由一對標籤和值欄位組成。標籤欄位是 label 和 data，而值欄位則是孩童的姓名和他們的生日。標籤欄位可以辨識 List 中顯示的內容，在這個範例中是孩童的姓名。最後的 List 看起來會像這樣：

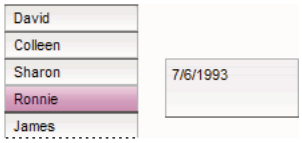


由 DataProvider 填入的 List

使用者在按一下以選取清單中的項目並造成 change 事件後就可以使用資料欄位的值。下列範例會將 TextArea (aTa) 和事件處理常式 (changeHandler) 加入上一個範例，在使用者選取 List 中的名稱時顯示孩童的生日。

```
import fl.controls.List;
import fl.controls.TextArea;
import flash.events.Event;
import fl.data.DataProvider;

var aList:List = new List();
var aTa:TextArea = new TextArea();
var items:Array = [
{label:"David", data:"1/19/1995"},
{label:"Colleen", data:"4/20/1993"},
{label:"Sharon", data:"9/06/1994"},
{label:"Ronnie", data:"7/6/1993"},
{label:"James", data:"2/15/1994"},
];
aList.dataProvider = new DataProvider(items);

addChild(aList);
addChild(aTa);

aList.move(150,150);
aTa.move(150, 260);

aList.addEventListener(Event.CHANGE, changeHandler);

function changeHandler(event:Event):void {
    aTa.text = event.target.selectedItem.data;
};
```

當使用者選取 List 中的孩童姓名後，TextArea 中就會顯示孩童的生日，如下圖所示。這項作業是由 changeHandler() 函數完成，它會將 TextArea (aTa.text) 的 text 屬性設定成已選取項目 (event.target.selectedItem.data) 中資料欄位的值。event.target 屬性是觸發這個事件的物件，在這個範例中就是 List。



從 List 的 DataProvider 顯示資料欄位

DataProvider 可以接受文字以外的資料。下列範例會將 MovieClips 加入 DataProvider 以提供資料給 TileList。範例會在建立彩色方塊 MovieClip 之後呼叫 addItem() 來加入每個項目，藉以建立 DataProvider。

```
import fl.data.DataProvider;
import flash.display.DisplayObject;

var aBox:MovieClip = new MovieClip();
var i:uint = 0;
var colors:Array = new Array(0x00000, 0xFF0000, 0x0000CC, 0x00CC00, 0xFFFF00);
var colorNames:Array = new Array("Midnight", "Cranberry", "Sky", "Forest", "July");
var dp:DataProvider = new DataProvider();
for(i=0; i < colors.length; i++) {
    drawBox(aBox, colors[i]);// draw box w next color in array
    dp.addItem( {label:colorNames[i], source:aBox} );
}
aTl.dataProvider = dp;
aTl.columnWidth = 110;
aTl.rowHeight = 130;
aTl.setSize(280,150);
aTl.move(150, 150);
aTl.setStyle("contentPadding", 5);

function drawBox(box:MovieClip,color:uint):void {
            box.graphics.beginFill(color, 1.0);
            box.graphics.drawRect(0, 0, 100, 100);
            box.graphics.endFill();
```

您也可以使用 XML 資料 (不使用陣列) 填入 DataProvider 物件。例如，下列程式碼會將資料儲存在名為 employeesXML 的 XML 物件中，然後傳遞該物件做為 DataProvider() 建構函數的 value 參數：

```
import fl.controls.DataGrid;
import fl.data.DataProvider;

var aDg:DataGrid = new DataGrid();
addChild(aDg);

var employeesXML:XML =
    <employees>
        <employee Name="Edna" ID="22" />
        <employee Name="Stu" ID="23" />
    </employees>;

var myDP:DataProvider = new DataProvider(employeesXML);

aDg.columns = ["Name", "ID"];
aDg.dataProvider = myDP;
```

您可以提供資料做為 XML 資料的特質 (如同上一段程式碼一般)，或是做為 XML 資料的屬性 (如下列程式碼所示)：

```
var employeesXML:XML =
    <employees>
        <employee>
            <Name>Edna</Name>
            <ID>22</ID>
        </employee>
        <employee>
            <Name>Stu</Name>
            <ID>23</ID>
        </employee>
    </employees>;
```

DataProvider 也有一組可以讓您對它進行存取和操作的方法和屬性。您可以使用 DataProvider API 加入、移除、取代、排序和合併 DataProvider 中的項目。

操作 DataProvider

您可以使用 addItem() 和 addItemAt() 方法將項目加入 DataProvider。下列範例會加入使用者在可編輯 ComboBox 的文字欄位中輸入的項目。範例中假設已經將 ComboBox 拖曳到「舞台」上，並為其指定實體名稱 aCb。

```
import fl.data.DataProvider;
import fl.events.ComponentEvent;

var items:Array = [
{label:"Roger"},
{label:"Carolyn"},
{label:"Darrell"},
{label:"Rebecca"},
{label:"Natalie"},
{label:"Mitchell"},
];
aCb.dataProvider = new DataProvider(items);

aCb.addEventListener(ComponentEvent.ENTER, newItemHandler);

function newItemHandler(event:ComponentEvent):void {
    var newRow:int = event.target.length + 1;
        event.target.addItemAt({label:event.target.selectedLabel},
        event.target.length);
}
```

您也可以透過組件的 DataProvider 來取代和移除組件中的項目。下列範例會實作兩個單獨的 List 組件：listA 和 listB，並提供一個已命名為 Sync 的 Button。當使用者按一下 Button 後，範例會使用 replaceItemAt() 方法，以 listA 中的項目取代 listB 中的項目。如果 listA 的長度大於 listB，範例會呼叫 addItem() 方法將額外的項目加入到 listB。如果 listB 的長度大於 listA，範例會呼叫 removeItemAt() 方法移除 ListB 中的額外項目。

```
// Requires the List and Button components to be in the library

import fl.controls.List;
import fl.controls.Button;
import flash.events.Event;
import fl.data.DataProvider;

var listA:List = new List();
var listB:List = new List();
var syncButton:Button = new Button();
syncButton.label = "Sync";

var itemsA:Array = [
{label:"David"},
{label:"Colleen"},
{label:"Sharon"},
{label:"Ronnie"},
{label:"James"},
];
var itemsB:Array = [
{label:"Roger"},
{label:"Carolyn"},
{label:"Darrell"},
{label:"Rebecca"},
{label:"Natalie"},
{label:"Mitchell"},
];
listA.dataProvider = new DataProvider(itemsA);
listB.dataProvider = new DataProvider(itemsB);

addChild(listA);
```

```
addChild(listB);
addChild(syncButton);

listA.move(100, 100);
listB.move(250, 100);
syncButton.move(175, 220);

syncButton.addEventListener(MouseEvent.CLICK, syncHandler);

function syncHandler(event:MouseEvent):void {
    var i:uint = 0;
    if(listA.length > listB.length) { //if listA is longer, add items to B
        while(i < listB.length) {
            listB.dataProvider.replaceItemAt(listA.dataProvider.getItemAt(i), i);
            ++i;
        }
        while(i < listA.length) {
            listB.dataProvider.addItem(listA.dataProvider.getItemAt(i++));
        }
    } else if(listA.length == listB.length) { //if listA and listB are equal length
        while(i < listB.length) {
            listB.dataProvider.replaceItemAt(listA.dataProvider.getItemAt(i), i);
            ++i;
        }
    } else {         //if listB is longer, remove extra items from B
        while(i < listA.length) {
            listB.dataProvider.replaceItemAt(listA.dataProvider.getItemAt(i), i);
            ++i;
        }
        while(i < listB.length) {
            listB.dataProvider.removeItemAt(i++);
        }
    }
}
```

您也可以使用 merge()、sort() 和 sortOn() 方法來合併及排序 DataProvider。下列範例會使用兩個壘球隊的部分名冊填入兩個 DataGrid 實體 (aDg 和 bDg)。範例中會加入標示 Merge 的 Button，當使用者按一下後，事件處理常式 (mrgHandler) 就會將 bDg 的名冊與 aDg 的名冊合併，然後對合併後 DataGrid 的「名稱」欄進行排序。

```
import fl.data.DataProvider;
import fl.controls.DataGrid;
import fl.controls.Button;

var aDg:DataGrid = new DataGrid();
var bDg:DataGrid = new DataGrid();
var mrgButton:Button = new Button();
addChild(aDg);
addChild(bDg);
addChild(mrgButton);
bldRosterGrid(aDg);
bldRosterGrid(bDg);
var aRoster:Array = new Array();
var bRoster:Array = new Array();
aRoster = [
        {Name:"Wilma Carter", Bats:"R", Throws:"R", Year:"So", Home: "Redlands, CA"},
        {Name:"Sue Pennypacker", Bats:"L", Throws:"R", Year:"Fr", Home: "Athens, GA"},
        {Name:"Jill Smithfield", Bats:"R", Throws:"L", Year:"Sr", Home: "Spokane, WA"},
        {Name:"Shirley Goth", Bats:"R", Throws:"R", Year:"Sr", Home: "Carson, NV"}
];
bRoster = [
        {Name:"Angelina Davis", Bats:"R", Throws:"R", Year:"So", Home: "Odessa, TX"},
        {Name:"Maria Santiago", Bats:"L", Throws:"L", Year:"Sr", Home: "Tacoma, WA"},
```

```
        {Name:"Debbie Ferguson", Bats:"R", Throws:"R", Year: "Jr", Home: "Bend, OR"}
];
aDg.dataProvider = new DataProvider(aRoster);
bDg.dataProvider = new DataProvider(bRoster);
aDg.move(50,50);
aDg.rowCount = aDg.length;
bDg.move(50,200);
bDg.rowCount = bDg.length;
mrgButton.label = "Merge";
mrgButton.move(200, 315);
mrgButton.addEventListener(MouseEvent.CLICK, mrgHandler);

function bldRosterGrid(dg:DataGrid){
    dg.setSize(400, 300);
    dg.columns = ["Name", "Bats", "Throws", "Year", "Home"];
    dg.columns[0].width = 120;
    dg.columns[1].width = 50;
    dg.columns[2].width = 50;
    dg.columns[3].width = 40;
    dg.columns[4].width = 120;
};

function mrgHandler(event:MouseEvent):void {
    aDg.dataProvider.merge(bDg.dataProvider);
    aDg.dataProvider.sortOn("Name");
}
```

如需詳細資訊，請參閱「DataProvider 類別」(位於「ActionScript 3.0 參考)。

使用 CellRenderer

CellRenderer 是屬於 List 架構的組件類別 (例如 List、DataGrid、TileList 和 ComboBox)，它是用來操作和顯示各列中的自訂儲存格內容。自訂儲存格可以包含文字、預先建置的組件 (例如 CheckBox) 或是您可以建立的任何顯示物件類別。如果要使用自訂 CellRenderer 顯示資料，您可以擴充 CellRenderer 類別或實作 ICellRenderer 介面來建立您自己的自訂 CellRenderer 類別。

List、DataGrid、TileList 和 ComboBox 類別是 SelectableList 類別的子類別。SelectableList 類別包括 cellRenderer 樣式。這個樣式會定義組件用來顯示儲存格的顯示物件。

您可以呼叫 List 物件的 setRendererStyle() 方法，調整 CellRenderer 所使用的樣式格式 (請參閱第 32 頁「格式化儲存格」)。或者，您也可以定義自訂類別做為 CellRenderer 使用 (請參閱第 33 頁「定義自訂 CellRenderer 類別」)。

格式化儲存格

CellRenderer 類別包括一些樣式，可以讓您控制儲存格的格式。

下列樣式可以讓您定義不同狀態下儲存格所使用的外觀元素 (停用、往下、重疊和往上)：

- disabledSkin 和 selectedDisabledSkin
- downSkin 和 selectedDownSkin
- overSkin 和 selectedOverSkin
- upSkin 和 selectedUpSkin

  下列樣式會套用至文字格式：

- disabledTextFormat

- textFormat
- textPadding

您可以呼叫 List 物件的 setRendererStyle() 方法或 CellRenderer 物件的 setStyle() 方法來設定這些樣式。您可以呼叫 List 物件的 getRendererStyle() 方法或 CellRenderer 物件的 getStyle() 方法來取得這些樣式。您也可以透過 List 物件的 rendererStyles 屬性或 CellRenderer 物件的 getStyleDefinition() 方法，存取用來定義所有輸出器樣式的物件 (做為物件的具名屬性)。

您可以呼叫 clearRendererStyle() 方法，將樣式重新設定成它的預設值。

如果要取得或設定清單中列的高度，請使用 List 物件的 rowHeight 屬性。

定義自訂 CellRenderer 類別

建立可以擴充 CellRenderer 類別的類別來定義自訂 CellRenderer

例如，下列程式碼會包括兩個類別。ListSample 類別會將 List 組件實體化，並使用另一個 CustomRenderer 類別來定義 List 組件所使用的儲存格輸出器。CustomRenderer 類別會擴充 CellRenderer 類別。

1 選取 「檔案 > 新增」。

2 在顯示的 「新增文件」對話方塊中選取 「Flash 檔案 (ActionScript 3.0)」，然後按一下 「確定」按鈕。

3 選取 「視窗 > 組件」，顯示 「組件」面板。

4 在 「組件」面板中，將 List 組件拖曳到 「舞台」上。

5 如果 Flash 沒有顯示 「屬性」檢測器，請選取 「視窗 > 屬性 > 屬性」。

6 選取 List 組件，然後設定 「屬性」檢測器中的屬性：

- 實體名稱：myList
- 寬 (寬度)：200
- 高 (高度)：300
- X：20
- Y：20

7 選取 「時間軸」中 「圖層 1」的 「影格 1」，然後選取 「視窗 > 動作」。

8 在 「動作」面板中輸入下列 Script：

```
myList.setStyle("cellRenderer", CustomCellRenderer);
myList.addItem({label:"Burger -- $5.95"});
myList.addItem({label:"Fries -- $1.95"});
```

9 選取 「檔案 > 儲存」。輸入檔案的名稱，然後按一下 「確定」按鈕。

10 選取 「檔案 > 新增」。

11 在顯示的 「新增文件」對話方塊中選取 「ActionScript 檔案」，然後按一下 「確定」按鈕。

12 在 Script 視窗中，輸入下列程式碼以定義 CustomCellRenderer 類別：

```
package {
    import fl.controls.listClasses.CellRenderer;
    import flash.text.TextFormat;
    import flash.filters.BevelFilter;
    public class CustomCellRenderer extends CellRenderer {
        public function CustomCellRenderer() {
            var format:TextFormat = new TextFormat("Verdana", 12);
            setStyle("textFormat", format);
            this.filters = [new BevelFilter()];
        }
    }
}
```

13 選取「檔案 > 儲存」。將檔案命名為 CustomCellRenderer.as，將它放入與 FLA 檔相同的目錄，然後按一下「確定」按鈕。

14 選取「控制 > 測試影片」。

使用會實作 ICellRenderer 介面的類別來定義自訂 CellRenderer

您也可以使用會繼承 DisplayObject 類別和實作 ICellRenderer 介面的任何類別來定義 CellRenderer。例如，下列程式碼會定義兩個類別。ListSample2 類別會將 List 物件加入到顯示清單，並將它的 CellRenderer 定義成可以使用 CustomRenderer 類別。CustomRenderer 類別會擴充 CheckBox 類別 (可擴充 DisplayObject 類別) 並實作 ICellRenderer 介面。請注意，CustomRenderer 類別會定義 data 和 listData 屬性的 getter 和 setter 方法，而這些屬性和方法已經在 ICellRenderer 介面中定義過。在 ICellRenderer 介面中定義的其它屬性和方法 (selected 屬性和 setSize() 方法) 也已經在 CheckBox 類別中定義過：

1 選取「檔案 > 新增」。

2 在顯示的「新增文件」對話方塊中選取「Flash 檔案 (ActionScript 3.0)」，然後按一下「確定」按鈕。

3 選取「視窗 > 組件」，顯示「組件」面板。

4 在「組件」面板中，將 List 組件拖曳到「舞台」上。

5 如果 Flash 沒有顯示「屬性」檢測器，請選取「視窗 > 屬性 > 屬性」。

6 選取 List 組件，然後設定「屬性」檢測器中的屬性：

- 實體名稱：myList
- 寬 (寬度)：100
- 高 (高度)：300
- X：20
- Y：20

7 選取「時間軸」中「圖層 1」的「影格 1」，然後選取「視窗 > 動作」。

8 在「動作」面板中輸入下列 Script：

```
myList.setStyle("cellRenderer", CustomCellRenderer);
myList.addItem({name:"Burger", price:"$5.95"});
myList.addItem({name:"Fries", price:"$1.95"});
```

9 選取「檔案 > 儲存」。輸入檔案的名稱，然後按一下「確定」按鈕。

10 選取「檔案 > 新增」。

11 在顯示的「新增文件」對話方塊中選取「ActionScript 檔案」，然後按一下「確定」按鈕。

12 在 Script 視窗中，輸入下列程式碼以定義 CustomCellRenderer 類別：

```
package
{
    import fl.controls.CheckBox;
    import fl.controls.listClasses.ICellRenderer;
    import fl.controls.listClasses.ListData;
    public class CustomCellRenderer extends CheckBox implements ICellRenderer {
        private var _listData:ListData;
        private var _data:Object;
        public function CustomCellRenderer() {
        }
        public function set data(d:Object):void {
            _data = d;
            label = d.label;
        }
        public function get data():Object {
            return _data;
        }
        public function set listData(ld:ListData):void {
            _listData = ld;
        }
        public function get listData():ListData {
            return _listData;
        }
    }
}
```

13 選取「檔案 > 儲存」。將檔案命名為 CustomCellRenderer.as，將它放入與 FLA 檔相同的目錄，然後按一下「確定」按鈕。

14 選取「控制 > 測試影片」。

使用元件來定義 CellRenderer

您也可以使用元件庫中的元件來定義 CellRenderer。您必須將元件匯出給 ActionScript 使用，且元件庫元件的類別名稱必須有一個會實作 ICellRenderer 介面或擴充 CellRenderer 類別 (或是它的其中一個子類別) 的關聯類別檔案。

下列範例會使用元件庫元件來定義自訂 CellRenderer。

1 選取「檔案 > 新增」。

2 在顯示的「新增文件」對話方塊中選取「Flash 檔案 (ActionScript 3.0)」，然後按一下「確定」按鈕。

3 選取「視窗 > 組件」，顯示「組件」面板。

4 在「組件」面板中，將 List 組件拖曳到「舞台」上。

5 如果 Flash 沒有顯示「屬性」檢測器，請選取「視窗 > 屬性 > 屬性」。

6 選取 List 組件，然後設定「屬性」檢測器中的屬性：

- 實體名稱：myList
- 寬 (寬度)：100
- 高 (高度)：400
- X：20
- Y：20

7 按一下「參數」面板，然後按兩下 dataProvider 列的第二欄。

8 在顯示的「值」對話方塊中，按兩次加號按鈕以加入兩個資料元素 (標籤已設定為 label0 和 label1)，然後按一下「確定」按鈕。

9 使用「文字」工具，在「舞台」上繪製一個文字欄位。

10 選取該文字欄位，然後設定 「屬性」檢測器中的屬性：

- 文字類型：動態文字
- 實體名稱：textField
- 寬 (寬度)：100
- 字體大小：24
- X：0
- Y：0

11 選取文字欄位，然後選取 「修改 > 轉換成元件」。

12 在 「轉換成元件」對話方塊中進行下列設定，然後按一下 「確定」。

- 名稱：MyCellRenderer
- 類型：影片片段
- 匯出給 ActionScript 使用：選取
- 匯出在第一個影格：選取
- 類別：MyCellRenderer
- 基底類別：flash.display.MovieClip

  如果 Flash 顯示 「ActionScript 類別警告」，請按一下警告方塊中的 「確定」按鈕。

13 刪除 「舞台」上新影片片段元件的實體。

14 選取 「時間軸」中 「圖層 1」的 「影格 1」，然後選取 「視窗 > 動作」。

15 在 「動作」面板中輸入下列 Script：

```
myList.setStyle("cellRenderer", MyCellRenderer);
```

16 選取 「檔案 > 儲存」。輸入檔案的名稱，然後按一下 「確定」按鈕。

17 選取 「檔案 > 新增」。

18 在顯示的 「新增文件」對話方塊中選取 「ActionScript 檔案」，然後按一下 「確定」按鈕。

19 在 Script 視窗中，輸入下列程式碼以定義 MyCellRenderer 類別：

```
package {
    import flash.display.MovieClip;
    import flash.filters.GlowFilter;
    import flash.text.TextField;
    import fl.controls.listClasses.ICellRenderer;
    import fl.controls.listClasses.ListData;
    import flash.utils.setInterval;
    public class MyCellRenderer extends MovieClip implements ICellRenderer {
        private var _listData:ListData;
        private var _data:Object;
        private var _selected:Boolean;
        private var glowFilter:GlowFilter;
        public function MyCellRenderer() {
            glowFilter = new GlowFilter(0xFFFF00);
            setInterval(toggleFilter, 200);
        }
        public function set data(d:Object):void {
            _data = d;
            textField.text = d.label;
        }
        public function get data():Object {
```

```
            return _data;
        }
        public function set listData(ld:ListData):void {
            _listData = ld;
        }
        public function get listData():ListData {
            return _listData;
        }
        public function set selected(s:Boolean):void {
            _selected = s;
        }
        public function get selected():Boolean {
            return _selected;
        }
        public function setSize(width:Number, height:Number):void {
        }
        public function setStyle(style:String, value:Object):void {
        }
        public function setMouseState(state:String):void{
        }
        private function toggleFilter():void {
            if (textField.filters.length == 0) {
                textField.filters = [glowFilter];
            } else {
                textField.filters = [];
            }
        }
    }
}
```

20 選取「檔案 > 儲存」。將檔案命名為 MyCellRenderer.as，將它放入與 FLA 檔相同的目錄，然後按一下「確定」按鈕。

21 選取「控制 > 測試影片」。

CellRenderer 屬性

data 屬性是一個物件，它包含了所有為 CellRenderer 設定的屬性。例如，下列類別會定義可以擴充 Checkbox 類別的自訂 CellRenderer；請注意 data 屬性的 setter 函數會將 data.label 的值傳遞給繼承自 CheckBox 類別的 label 屬性：

```
    public class CustomRenderer extends CheckBox implements ICellRenderer {
        private var _listData:ListData;
        private var _data:Object;
        public function CustomRenderer() {
        }
        public function set data(d:Object):void {
            _data = d;
            label = d.label;
        }
        public function get data():Object {
            return _data;
        }
        public function set listData(ld:ListData):void {
            _listData = ld;
        }
        public function get listData():ListData {
            return _listData;
        }
    }
}
```

selected 屬性定義清單中是否已經選取了儲存格。

將 CellRenderer 套用至 DataGrid 物件的欄

DataGrid 物件可以擁有多個欄，您也可以為每一欄指定不同的儲存格輸出器。DataGrid 的每一欄都是由 DataGridColumn 物件表示，DataGridColumn 類別也具有 cellRenderer 屬性，您可以定義每一欄的 CellRenderer。

定義可編輯儲存格的 CellRenderer

DataGridCellEditor 類別會定義用於 DataGrid 物件中可編輯儲存格的輸出器。當 DataGrid 物件的 editable 屬性設定為 true 且使用者按一下要編輯的儲存格後，它就會成為儲存格的輸出器。如果要定義可編輯儲存格的 CellRenderer，請在 DataGrid 物件的 columns 陣列中設定每個元素的 itemEditor 屬性。

使用影像、SWF 檔或影片片段做為 CellRenderer

ImageCell 類別是 CellRenderer 的子類別，它可以定義物件在儲存格的主要內容是影像、SWF 檔或影片片段時用於顯示儲存格。ImageCell 類別包括下列可以定義儲存格外觀的樣式：

- imagePadding— 用來分隔儲存格邊緣與影像邊緣的邊框間距，以像素為單位
- selectedSkin— 用來表示已選取狀態的外觀元素
- textOverlayAlpha— 重疊在儲存格標籤之後的不透明部分
- textPadding— 用來分隔儲存格邊緣與文字邊緣的邊框間距，以像素為單位

ImageCell 類別是 TileList 類別預設的 CellRenderer。

使組件具備輔助功能

您可以透過會以語音說明螢幕內容的螢幕朗讀程式，讓視力不佳的使用者可以存取 Flash 應用程式的螢幕內容。如需如何將 Flash 應用程式設定成可以支援螢幕朗讀程式的詳細資訊，請參閱 「使用 Flash」中的第 18 章 「建立輔助功能內容」。

如果要讓 ActionScript 3.0 組件可以支援螢幕朗讀程式，您必須同時匯入它的輔助功能類別並呼叫類別的 enableAccessibility() 方法。您可以讓下列 ActionScript 3.0 組件支援螢幕朗讀程式：

組件	**Accessibility 類別**
Button	ButtonAccImpl
CheckBox	CheckBoxAccImpl
ComboBox	ComboBoxAccImpl
List	ListAccImpl
RadioButton	RadioButtonAccImpl
TileList	TileListAccImpl

組件輔助功能類別是位於 fl.accessibility 套件內。例如，要讓 CheckBox 可以支援螢幕朗讀程式，請在應用程式中加入下列陳述式：

```
import fl.accessibility.CheckBoxAccImpl;

CheckBoxAccImpl.enableAccessibility();
```

不論您為組件建立了多少個實體，都只需要啟用組件的輔助功能一次。

備註：啟用輔助功能會在編譯期間加入必要的類別，但只會增加一點檔案大小。

大部分組件都可以透過鍵盤來瀏覽。如需啟用輔助功能組件以及使用鍵盤瀏覽的詳細資訊，請參閱第 40 頁 「使用 UI 組件」中的 「使用者互動」小節，以及 適用於 Adobe Flash Platform 的 ActionScript 3.0 參考中的輔助功能類別。

第 4 章 使用 UI 組件

本章說明如何使用 Flash 隨附的 ActionScript 3.0 使用者介面 (UI) 組件。

使用 Button 組件

此 Button 組件是可調整大小的矩形按鈕，使用者可以透過滑鼠或空白鍵按下此按鈕，啟動應用程式中的動作。您可以將自訂的圖示增加到 Button，也可以將 Button 的行為從按下方式變更為切換方式。切換 Button 會在按一下時保持按下狀態，再按一下時又會回復一般狀態。

Button 是許多表單與網路應用程式的基本項目。如果希望使用者觸發某個事件，您就可以使用按鈕。例如，大部分表單都有「送出」按鈕。您也可以在簡報中增加「上一張」和「下一張」按鈕。

與 Button 組件的使用者互動

您可以在應用程式中啟用或停用按鈕。處於停用狀態的按鈕不接受滑鼠或鍵盤輸入。如果您按一下或用 Tab 鍵移到啟用的按鈕上，它就會成為焦點。當 Button 實體成為焦點時，您可以使用下列按鍵加以控制：

按鍵	說明
Shift+Tab	將焦點移到上一個物件。
空白鍵	按下或放開按鈕，並且觸發 click 事件。
Tab 鍵	將焦點移到下一個物件。
Enter/Return	如果按鈕設定為 FocusManager 的預設 Button，則會將焦點移至下一個物件。

如需有關控制焦點的詳細資訊，請參閱 Flash Professional 的 ActionScript 3.0 參考中的「IFocusManager 介面」和「FocusManager 類別」，以及第 24 頁「使用 FocusManager」。

進行編寫時，每個 Button 實體的即時預覽會反映您在「屬性」檢測器或「組件檢測器」中對參數的變更。

備註：如果圖示比按鈕大，就會延伸超出按鈕的邊框。

若要在應用程式中指定某個按鈕為預設按鈕 (也就是當使用者按下 Enter 鍵時，便會接收 click 事件的按鈕)，請設定 FocusManager.defaultButton。例如，下列程式碼會將預設按鈕設定為 Button 實體 submitButton。

```
FocusManager.defaultButton = submitButton;
```

將 Button 組件加入應用程式時，您可以加入下列 ActionScript 程式碼，讓螢幕朗讀程式能夠存取此組件：

```
import fl.accessibility.ButtonAccImpl;

ButtonAccImpl.enableAccessibility();
```

不論您為組件建立了多少個實體，都只需要啟用組件的輔助功能一次。

Button 組件參數

您可以在「屬性」檢測器 (「視窗 > 屬性 > 屬性」) 或「組件檢測器」(「視窗 > 組件檢測器」) 中，為每個 Button 實體：emphasized、label、labelPlacement、selected 和 toggle。這些參數都具有相對應的 ActionScript 同名屬性。當您指定這些參數的值時，就是設定屬性在應用程式中的起始狀態。在 ActionScript 中設定屬性會覆寫您設定給參數的值。如需有關這些參數可能值的詳細資訊，請參閱 適用於 Adobe Flash Platform 的 ActionScript 3.0 參考中的 Button 類別。

建立具有 Button 組件的應用程式

下列程序說明如何在編寫時將 Button 組件加入應用程式。在這個範例中，當您按一下 Button 時，就會變更 ColorPicker 組件的狀態。

1 建立新的 Flash (ActionScript 3.0) 文件。

2 將 Button 組件從 「組件」面板拖曳到 「舞台」，然後在 「屬性」檢測器中為組件輸入下列值：

- 輸入實體名稱 **aButton**。
- 輸入 **Show** 做為 label 參數的值。

3 在 「舞台」上新增 ColorPicker，並賦予實體名稱 **aCp**。

4 開啟 「動作」面板，選取主要 「時間軸」中的 「影格 1」，然後輸入下列 ActionScript 程式碼：

```
aCp.visible = false;

aButton.addEventListener(MouseEvent.CLICK, clickHandler);

function clickHandler(event:MouseEvent):void {

    switch(event.currentTarget.label) {
        case "Show":
            aCp.visible = true;
            aButton.label = "Disable";
            break;
        case "Disable":
            aCp.enabled = false;
            aButton.label = "Enable";
            break;
        case "Enable":
            aCp.enabled = true;
            aButton.label = "Hide";
            break;
        case "Hide":
            aCp.visible = false;
            aButton.label = "Show";
            break;
    }
}
```

第二行程式碼會將 clickHandler() 函數註冊成為 MouseEvent.CLICK 事件的事件處理常式函數。當使用者按一下該 Button 時便會發生此事件，並導致 clickHandler() 函數根據 Button 的值採取下列其中一個動作：

- Show 會讓 ColorPicker 變成可見狀態，並將 Button 的標籤變更為 Disable。
- Disable 會停用 ColorPicker，並將 Button 的標籤變更為 Enable。
- Enable 會啟用 ColorPicker，並將 Button 的標籤變更為 Hide。
- Hide 會讓 ColorPicker 變成隱藏狀態，並將 Button 的標籤變更為 Show。

5 選取 「控制 > 測試影片」，執行應用程式。

建立具有 Button 組件的應用程式

下列程序會使用 ActionScript 建立切換 Button，並且當您按一下此 Button 時，會在 「輸出」面板中顯示事件類型。此範例會叫用類別的建構函式以建立 Button 實體，並透過呼叫 addChild() 方法將實體加入 「舞台」。

1 建立新的 Flash (ActionScript 3.0) 文件。

2 將 Button 組件從 「組件」面板拖曳到目前文件的 「元件庫」面板。

此舉會將組件加入元件庫中，但不會使其在應用程式中顯示。

3 開啟「動作」面板，選取主要「時間軸」中的「影格 1」，然後輸入下列程式碼以建立 Button 實體：

```
import fl.controls.Button;

var aButton:Button = new Button();
addChild(aButton);
aButton.label = "Click me";
aButton.toggle =true;
aButton.move(50, 50);
```

move() 方法會將該按鈕放置於「舞台」上 50 (x 座標), 50 (y 座標) 的位置。

4 接著，加入下列 ActionScript 以建立事件偵聽程式和事件處理常式函數：

```
aButton.addEventListener(MouseEvent.CLICK, clickHandler);

function clickHandler(event:MouseEvent):void {
    trace("Event type: " + event.type);
}
```

5 選取「控制 > 測試影片」。

當您按一下按鈕時，Flash 會在「輸出」面板中顯示「Event type: click」訊息。

使用 CheckBox 組件

此 CheckBox 是一種可供選取或取消選取的方塊。選取之後，方塊中就會出現一個核取標記。您可以將文字標籤加入 CheckBox，放置在 CheckBox 的左邊、右邊、上方或下方。

如果您需要蒐集一組非互斥的 true 或 false 值，就可以使用 CheckBox。例如，對於蒐集您要購買之車款相關資訊的應用程式，就會使用 CheckBox 供您選取車款特徵。

與 CheckBox 的使用者互動

您可以在應用程式中啟用或停用 CheckBox。啟用 CheckBox 之後，如果使用者按一下這個方塊或它的標籤，CheckBox 就會成為輸入焦點並且顯示當按下時的外觀。如果使用者在按下滑鼠按鈕期間，將指標移到 CheckBox 或其標籤的範圍區域之外，該組件的外觀就會回復原始狀態並且保留輸入焦點。CheckBox 的狀態要等到滑鼠在該組件上方放開之後才會變更。此外，CheckBox 還有選取和取消選取兩種停用狀態，兩者分別使用 selectedDisabledSkin 和 disabledSkin，不允許滑鼠或鍵盤互動。

停用 CheckBox 之後，不論使用者的動作為何，都只會顯示它的停用狀態。處於停用狀態的 CheckBox 不接受滑鼠或鍵盤輸入。

當使用者按一下 CheckBox 或用 Tab 鍵移到實體上方時，它就會成為焦點。當 CheckBox 實體成為焦點時，您可以使用下列按鍵加以控制：

按鍵	說明
Shift+Tab	將焦點移到上一個元素。
空白鍵	選取或取消選取組件，並且觸發 change 事件。
Tab 鍵	將焦點移到下一個元素。

如需有關控制焦點的詳細資訊，請參閱第 24 頁「使用 FocusManager」以及 Flash Professional 的 ActionScript 3.0 參考中的「FocusManager 類別」。

進行編寫時，每個 CheckBox 實體的即時預覽會反映您在「屬性」檢測器或「組件檢測器」中對參數的變更。

將 CheckBox 組件加入應用程式時，您可以加入下列 ActionScript 程式碼，讓螢幕朗讀程式能夠存取此組件：

```
import fl.accessibility.CheckBoxAccImpl;

CheckBoxAccImpl.enableAccessibility();
```

不論您有多少個組件實體，都只需要啟用組件的輔助功能一次。

CheckBox 組件參數

您可以在 「屬性」檢測器或 「組件檢測器」中，為每個 CheckBox 組件實體設定下列編寫參數：label、labelPlacement 和 selected。這些參數都具有相對應的 ActionScript 同名屬性。如需有關這些參數可能值的詳細資訊，請參閱 適用於 Adobe Flash Platform 的 ActionScript 3.0 參考中的 CheckBox 類別。

使用 CheckBox 建立應用程式

下列程序說明如何使用貸款應用程式表單的一部分摘錄，在編寫期間將 CheckBox 組件加入應用程式。這份表單會詢問申請人是否自己擁有房屋，並提供 CheckBox 供使用者回答 「是」。假設符合上述條件，此表單便會顯示兩個選項按鈕，供使用者指定房屋的相對價值。

使用 CheckBox 組件建立應用程式

1 建立新的 Flash (ActionScript 3.0) 文件。

2 將 CheckBox 組件從 「組件」面板拖曳到 「舞台」。

3 在 「屬性」檢測器中，執行下列步驟：

- 輸入 **homeCh** 做為實體名稱。
- 輸入 **140** 做為寬度 (W) 值。
- 輸入 **Own your home?** 做為 label 參數的值。

4 將兩個 RadioButton 組件從 「組件」面板拖曳到 「舞台」，放置於 CheckBox 的下方及右邊。在 「屬性」檢測器中，為這些組件輸入下列值：

- 輸入 **underRb** 和 **overRb** 做為實體名稱。
- 輸入 **120** 做為這兩個 RadioButton 的 W (寬度) 參數值。
- 輸入 **Under $500,000?** 做為 underRb 的 label 參數值。
- 輸入 **Over $500,000?** 做為 overRb 的 label 參數值。
- 輸入 **valueGrp** 做為這兩個 RadioButton 的 groupName 參數值。

5 開啟 「動作」面板，選取主要 「時間軸」中的 「影格 1」，然後輸入下列 ActionScript 程式碼：

```
homeCh.addEventListener(MouseEvent.CLICK, clickHandler);
underRb.enabled = false;
overRb.enabled = false;

function clickHandler(event:MouseEvent):void {
    underRb.enabled = event.target.selected;
    overRb.enabled = event.target.selected;
}
```

此程式碼會建立 CLICK 事件的事件處理常式，以便在選取 homeCh CheckBox 的情況下啟用 underRb 和 overRb RadioButton；如果未選取 homeCh，則會將它們停用。如需詳細資訊，請參閱 Flash Professional 的 ActionScript 3.0 參考中的 「MouseEvent 類別」。

6 選取 「控制 > 測試影片」。

下列範例會重製上一個應用程式，但不同的是使用 ActionScript 建立 CheckBox 和 RadioButton。

使用 ActionScript 建立 Checkbox

1 建立新的 Flash (ActionScript 3.0) 文件。

2 將 CheckBox 組件和 RadioButton 組件從「組件」面板拖曳到目前文件的「元件庫」面板。如果「元件庫」面板尚未開啟，請按 Ctrl+L 或選取「視窗 > 元件庫」，開啟「元件庫」面板。

這麼做能讓您的應用程式可以使用這些組件，而不需將這些組件放在「舞台」。

3 開啟「動作」面板，選取主要「時間軸」中的「影格 1」，然後輸入下列程式碼以建立並放置組件實體：

```
import fl.controls.CheckBox;
import fl.controls.RadioButton;

var homeCh:CheckBox = new CheckBox();
var underRb:RadioButton = new RadioButton();
var overRb:RadioButton = new RadioButton();
addChild(homeCh);
addChild(underRb);
addChild(overRb);
underRb.groupName = "valueGrp";
overRb.groupName = "valueGrp";
homeCh.move(200, 100);
homeCh.width = 120;
homeCh.label = "Own your home?";
underRb.move(220, 130);
underRb.enabled = false;
underRb.width = 120;
underRb.label = "Under $500,000?";
overRb.move(220, 150);
overRb.enabled = false;
overRb.width = 120;
overRb.label = "Over $500,000?";
```

此程式碼會使用 CheckBox() 和 RadioButton() 建構函式建立組件，並使用 addChild() 方法將這些組件放置於「舞台」。它會將 move() 方法用於指定組件在「舞台」上的位置。

4 接著，加入下列 ActionScript 以建立事件偵聽程式和事件處理常式函數：

```
homeCh.addEventListener(MouseEvent.CLICK, clickHandler);

function clickHandler(event:MouseEvent):void {
    underRb.enabled = event.target.selected;
    overRb.enabled = event.target.selected;
}
```

此程式碼會建立 CLICK 事件的事件處理常式，以便在選取 homeCh CheckBox 的情況下啟用 underRb 和 overRb 選項按鈕；如果未選取 homeCh，則會將它們停用。如需詳細資訊，請參閱 Flash Professional 的 ActionScript 3.0 參考中的「MouseEvent 類別」。

5 選取「控制 > 測試影片」。

使用 ColorPicker 組件

此 ColorPicker 組件可以讓使用者從色票清單中選取顏色。ColorPicker 的預設模式會在方形按鈕中顯示單一顏色。當使用者按一下該按鈕時，可使用的顏色清單就會伴隨文字欄位出現在色票面板中，而文字欄位會顯示目前選取顏色的十六進位值。

您只需將 ColoPicker 的 colors 屬性設定為希望顯示的顏色值，即可設定組件所要顯示的顏色。

與 ColorPicker 組件的使用者互動

ColorPicker 可以讓使用者選取顏色，並將該顏色套用到應用程式中的另一個物件。例如，若您要讓使用者能夠將應用程式的元素 (背景顏色或文字顏色等) 個人化，便可加入 ColorPicker 並套用使用者選取的顏色。

使用者可以按一下面板中某個顏色的色票，或是在文字欄位中輸入該顏色的十六進位值，藉以選擇該顏色。一旦使用者選取某個顏色之後，您便可以使用 ColorPicker 的 selectedColor 屬性，將該顏色套用到應用程式中的文字或另一個物件。

如果使用者將指標移動到 ColorPicker 實體上方或使用 Tab 鍵瀏覽至該實體，它就會成為焦點。當 ColorPicker 的色票面板開啟時，您可以使用下列按鍵加以控制：

按鍵	說明
首頁	將色票面板中的選取項目移至第一個顏色。
向上鍵	將色票面板中的選取項目上移一列。
向下鍵	將色票面板中的選取項目下移一列。
向右鍵	將色票面板中的選取項目向右移動一個顏色。
向左鍵	將色票面板中的選取項目向左移動一個顏色。
End	將色票面板中的選取項目移至最後一個顏色。

ColorPicker 組件參數

您可以在 「屬性」檢測器或 「組件檢測器」中，為每個 ColorPicker 實體設定下列編寫參數：selectedColor 和 showTextField。這些參數都具有相對應的 ActionScript 同名屬性。如需有關這些參數可能值的詳細資訊，請參閱 Flash Professional 的 ActionScript 3.0 參考中的 ColorPicker 類別。

建立具有 ColorPicker 組件的應用程式

下列範例會在編寫期間，將 ColorPicker 組件加入應用程式。在此範例中，每一次變更 ColorPicker 中的顏色時，changeHandler() 函數都會呼叫 drawBox() 函數，繪製具有您在 ColorPicker 中選取之顏色的新方塊。

1 建立新的 Flash (ActionScript 3.0) 文件。

2 將 ColorPicker 從 「組件」面板拖曳到 「舞台」，並賦予實體名稱 **aCp**。

3 開啟 「動作」面板，選取主要 「時間軸」中的 「影格 1」，然後輸入下列 ActionScript 程式碼：

```
import fl.events.ColorPickerEvent;

var aBox:MovieClip = new MovieClip();
drawBox(aBox, 0xFF0000);//draw a red box
addChild(aBox);

aCp.addEventListener(ColorPickerEvent.CHANGE,changeHandler);

function changeHandler(event:ColorPickerEvent):void {
    drawBox(aBox, event.target.selectedColor);
}

function drawBox(box:MovieClip,color:uint):void {
            box.graphics.beginFill(color, 1);
            box.graphics.drawRect(100, 150, 100, 100);
            box.graphics.endFill();
}
```

4 選取 「控制 > 測試影片」。

5 按一下 ColorPicker 並選取顏色，為方塊上色。

使用 ActionScript 建立 ColorPicker

此範例會使用 ColorPicker() 建構函式和 addChild()，在「舞台」上建立 ColorPicker。接著將 colors 屬性設定為紅色的顏色值 (0xFF0000)、綠色的顏色值 (0x00FF00) 與藍色的顏色值 (0x0000FF)，以指定 ColorPicker 將要顯示的顏色。另外還會建立 TextArea。每當您從 ColorPicker 選取不同的顏色時，TextArea 中的文字顏色隨即變更為相符的顏色。

1 建立新的 Flash (ActionScript 3.0) 文件。

2 將 ColorPicker 組件從 「組件」面板拖曳到 「元件庫」面板。

3 將 TextArea 組件從 「組件」面板拖曳到 「元件庫」面板。

4 開啟 「動作」面板，選取主要 「時間軸」中的 「影格 1」，然後輸入下列 ActionScript 程式碼：

```
import fl.controls.ColorPicker;
import fl.controls.TextArea;
import fl.events.ColorPickerEvent;

var aCp:ColorPicker = new ColorPicker();
var aTa:TextArea = new TextArea();
var aTf:TextFormat = new TextFormat();

aCp.move(100, 100);
aCp.colors = [0xff0000, 0x00ff00, 0x0000ff];
aCp.addEventListener(ColorPickerEvent.CHANGE, changeHandler);

aTa.text = "Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Vivamus quis nisl vel tortor nonummy
vulputate. Quisque sit amet eros sed purus euismod tempor. Morbi tempor. Class aptent taciti sociosqu ad
litora torquent per conubia nostra, per inceptos hymenaeos. Curabitur diam. Suspendisse at purus in ipsum
volutpat viverra. Nulla pellentesque libero id libero.";
aTa.setSize(200, 200);
aTa.move(200,100);

addChild(aCp);
addChild(aTa);

function changeHandler(event:ColorPickerEvent):void {
    if(TextFormat(aTa.getStyle("textFormat"))){
        aTf = TextFormat(aTa.getStyle("textFormat"));
    }
    aTf.color = event.target.selectedColor;
    aTa.setStyle("textFormat", aTf);
}
```

5 選取 「控制 > 測試影片」。

使用 ComboBox 組件

此 ComboBox 組件可以讓使用者從下拉式清單選取單一項目。ComboBox 可以是靜態的或是可供編輯。可編輯的 ComboBox 能讓使用者直接在清單頂端的文字欄位輸入文字。如果下拉式清單碰到文件底部，它會以向上開啟方式取代向下開啟。ComboBox 由下列三個子組件構成：BaseButton、TextInput 和 List 組件。

在可編輯的 ComboBox 中，只有按鈕才是作用區域 (而非文字方塊)。但是對於靜態 ComboBox，按鈕和文字方塊都是作用區域。開啟或關閉下拉式清單便會使作用區域產生回應。

當使用者透過滑鼠或鍵盤選取清單中的項目時，選取項目的標籤便會複製到 ComboBox 頂端的文字欄位。

與 ComboBox 組件的使用者互動

若有任何表單或應用程式需要從清單中選取單一選項，您就可以使用 ComboBox 組件。例如，您可以在客戶地址表單中提供「州」的下拉式清單。對於較複雜的情況，您可以使用可編輯的 ComboBox。例如，在提供行車指引的應用程式中，使用可編輯的 ComboBox 讓使用者輸入出發點及目的地的地址。下拉式清單會包含使用者先前輸入的地址。

如果 ComboBox 可以編輯，表示 editable 屬性為 true，則下列按鍵會從文字輸入方塊移除焦點，並保留先前的值。Enter 鍵則是例外，因為如果使用者輸入文字，此鍵便會先套用新的值。

按鍵	說明
Shift+Tab	將焦點移到上一個項目。如果已經選取新的項目，則會傳送 change 事件。
Tab 鍵	將焦點移到下一個項目。如果已經選取新的項目，則會傳送 change 事件。
向下鍵	將選取範圍向下移動一個項目。
End	將選取範圍移到清單底部。
Escape	關閉下拉式清單，使 ComboBox 成為焦點。
Enter	關閉下拉式清單，使 ComboBox 成為焦點。當 ComboBox 為可編輯組件，而且使用者輸入文字時，Enter 會將值設定為輸入的文字。
首頁	將選取範圍移到清單頂端。
Page Up	將選取範圍向上移動一頁。
Page Down	將選取範圍向下移動一頁。

將 ComboBox 組件加入應用程式時，您可以加入下列 ActionScript 程式碼，讓螢幕朗讀程式能夠存取此組件：

```
import fl.accessibility.ComboBoxAccImpl;

ComboBoxAccImpl.enableAccessibility();
```

不論您有多少個組件實體，都只需要啟用組件的輔助功能一次。

ComboBox 組件參數

您可以在「屬性」檢測器或「組件檢測器」中，為每個 ComboBox 實體設定下列參數：dataProvider、editable、prompt 和 rowCount。這些參數都具有相對應的 ActionScript 同名屬性。如需有關這些參數可能值的詳細資訊，請參閱 Flash Professional 的 ActionScript 3.0 參考中的 ComboBox 類別。如需有關 dataProvider 參數用法的詳細資訊，請參閱第 25 頁「使用 dataProvider 參數」。

建立具有 ComboBox 組件的應用程式

下列程序說明如何在編寫時將 ComboBox 組件加入應用程式。ComboBox 為可編輯，而且如果您在文字欄位中輸入 **Add**，範例就會在下拉式清單中新增一個項目。

1. 建立新的 Flash (ActionScript 3.0) 文件。
2. 將 ComboBox 拖曳到「舞台」，並賦予實體名稱 **aCb**。在「參數」索引標籤中，將 editable 參數設定為 true。
3. 開啟「動作」面板，選取主要「時間軸」中的「影格 1」，然後輸入下列程式碼：

```
import fl.data.DataProvider;
import fl.events.ComponentEvent;

var items:Array = [
{label:"screen1", data:"screenData1"},
{label:"screen2", data:"screenData2"},
{label:"screen3", data:"screenData3"},
{label:"screen4", data:"screenData4"},
{label:"screen5", data:"screenData5"},
];
aCb.dataProvider = new DataProvider(items);

aCb.addEventListener(ComponentEvent.ENTER, onAddItem);

function onAddItem(event:ComponentEvent):void {
    var newRow:int = 0;
    if (event.target.text == "Add") {
        newRow = event.target.length + 1;
            event.target.addItemAt({label:"screen" + newRow, data:"screenData" + newRow},
            event.target.length);
    }
}
```

4 選取「控制 > 測試影片」。

使用 ActionScript 建立 ComboBox

下列範例會使用 ActionScript 建立 ComboBox，並在其中填入位於加州舊金山地區的大學清單。接著設定 ComboBox 的 width 屬性以配合提示文字的寬度，並將 dropdownWidth 屬性設定為稍寬一些，以配合最長的大學名稱。

此範例在 Array 實體中建立大學清單，並使用 label 屬性儲存學校名稱，使用 data 屬性儲存每一所學校網站的 URL。它會設定 ComboBox 的 dataProvider 屬性，以指定由 Array 提供資料給組件。

當使用者從清單中選取一所大學時，便會觸發 Event.CHANGE 事件並呼叫 changeHandler() 函數，然後將 data 屬性載入至 URL 要求，以存取該學校的網站。

請注意，最後一行程式碼將 ComboBox 實體的 selectedIndex 屬性設定為 -1，以便於清單關閉時重現提示；否則，該提示將由所選取的學校名稱取代。

1 建立新的 Flash (ActionScript 3.0) 文件。

2 將 ComboBox 組件從「組件」面板拖曳到「元件庫」面板。

3 開啟「動作」面板，選取主要「時間軸」中的「影格 1」，然後輸入下列 ActionScript 程式碼：

```
import fl.controls.ComboBox;
import fl.data.DataProvider;
import flash.net.navigateToURL;

var sfUniversities:Array = new Array(
    {label:"University of California, Berkeley",
                data:"http://www.berkeley.edu/"},
    {label:"University of San Francisco",
                data:"http://www.usfca.edu/"},
    {label:"San Francisco State University",
                data:"http://www.sfsu.edu/"},
    {label:"California State University, East Bay",
                data:"http://www.csuhayward.edu/"},
    {label:"Stanford University", data:"http://www.stanford.edu/"},
    {label:"University of Santa Clara", data:"http://www.scu.edu/"},
    {label:"San Jose State University", data:"http://www.sjsu.edu/"}
);

var aCb:ComboBox = new ComboBox();
aCb.dropdownWidth = 210;
aCb.width = 200;
aCb.move(150, 50);
aCb.prompt = "San Francisco Area Universities";
aCb.dataProvider = new DataProvider(sfUniversities);
aCb.addEventListener(Event.CHANGE, changeHandler);

addChild(aCb);

function changeHandler(event:Event):void {
    var request:URLRequest = new URLRequest();
    request.url = ComboBox(event.target).selectedItem.data;
    navigateToURL(request);
    aCb.selectedIndex = -1;
}
```

4 選取「控制 > 測試影片」。

您可以在 Flash 編寫環境中實作及執行此範例，但如果嘗試以按一下 ComboBox 中項目的方式存取大學網站，就會收到警告訊息。若要在網際網路上存取功能完整的 ComboBox，請存取下列位置：

http://www.helpexamples.com/peter/bayAreaColleges/bayAreaColleges.html

使用 DataGrid 組件

此 DataGrid 組件讓您能以列與欄的格線形式顯示資料，它會繪製陣列或外部 XML 檔中的資料，讓您可以剖析至 DataProvider 的陣列內部。DataGrid 組件包含垂直和水平捲動功能、事件支援 (包括對可編輯儲存格的支援) 功能，以及排序功能。

您可以調整大小並自訂如字體、顏色及格線中各欄邊框之類的特性。您可以將自訂影片片段當做格線中任何一欄的「儲存格輸出器」(儲存格輸出器顯示儲存格的內容)。您可以關閉捲軸並使用 DataGrid 方法，建立頁面檢視樣式顯示。如需有關自訂的詳細資訊，請參閱 Flash Professional 的 ActionScript 3.0 參考中的「DataGridColumn 類別」。

更多說明主題

建立、填入 DataGrid 組件和調整組件大小

自訂和排序 DataGrid 組件

在 DataGrid 組件中篩選資料和設定格式

與 DataGrid 組件的使用者互動

您可以使用滑鼠和鍵盤與 DataGrid 組件互動。

如果 sortableColumns 屬性和該欄的 sortable 屬性都是 true，按一下欄標題便會根據該欄的值來排序資料。您可以將各欄的 sortable 屬性設定為 false，停用該欄的排序功能。

如果 resizableColumns 屬性為 true，您便可藉由拖曳標題列上的欄分隔線來調整欄的大小。

在可編輯的儲存格中按一下，就能讓該儲存格成為焦點；而在不可編輯的儲存格中按一下，則不會對焦點產生任何影響。當個別儲存格的 DataGrid.editable 和 DataGridColumn.editable 兩個屬性都是 true 時，該儲存格便為可編輯的儲存格。

如需詳細資訊，請參閱 ActionScript 3.0 參考中的 「DataGrid 類別」和 「DataGridColumn 類別」。

利用按一下或 Tab 鍵定位的方式讓 DataGrid 實體成為焦點後，您就能使用下列按鍵加以控制：

按鍵	說明
向下鍵	編輯儲存格時，插入點會移到儲存格文字的結尾。如果儲存格不可編輯，向下鍵則以如同 List 組件的方式來處理選取範圍。
向上鍵	編輯儲存格時，插入點會移到儲存格文字的開頭。如果儲存格不可編輯，向上鍵則以如同 List 組件的方式來處理選取範圍。
Shift+ 向上 / 向下鍵	如果 DataGrid 不可編輯，而且 allowMultipleSelection 為 true，就會選取相鄰的列。以相反的按鍵逆向操作時，會取消選取所選取的列，直到跳過一開始選取的列為止 (此時，繼續逆向操作又會開始選取列)。
Shift+ 按一下	如果 allowMultipleSelection 為 true，便會選取介於所選取列與目前跳脫字元位置 (反白標示的儲存格) 之間的所有列。
Ctrl+ 按一下	如果 allowMultipleSelection 為 true，則會選取其它列 (不一定相鄰)。
向右鍵	編輯儲存格時，插入點會向右移一個字元。如果儲存格不可編輯，向右鍵不會有任何反應。
向左鍵	編輯儲存格時，插入點會向左移一個字元。如果儲存格不可編輯，向左鍵不會有任何反應。
首頁	選取 DataGrid 中的第一列。
End	選取 DataGrid 中的最後一列。
PageUp	選取 DataGrid 頁面中的第一列。整個頁面包含了 DataGrid 不需要捲動就能顯示的所有列數。
PageDown	選取 DataGrid 頁面中的最後一列。整個頁面包含了 DataGrid 不需要捲動就能顯示的所有列數。
Return/Enter/Shift+Enter	當儲存格可以編輯時，這些按鍵可以認可變更，並將插入點移到同一欄中下一列的儲存格 (向上或向下，視位移切換而定)。
Shift+Tab/Tab	如果 DataGrid 可以編輯，便會將焦點移到上一個 / 下一個項目，直到抵達該欄結尾為止，然後再移到上一列 / 下一列，直到抵達第一個或最後一個儲存格為止。如果已經選取第一個儲存格，按下 Shift+Tab 便會將焦點移到前一個控制項。如果已經選取最後一個儲存格，按下 Tab 便會將焦點移到下一個控制項。 如果 DataGrid 不可編輯，則會將焦點移到上一個 / 下一個控制項。

您可以使用 DataGrid 組件做為各種以資料驅動之應用程式類型的基礎。您可以輕鬆地顯示格式化表格資料檢視，也可以使用儲存格輸出器功能來建立更複雜又可以編輯的使用者介面片段。以下是 DataGrid 組件的實際用途：

- 網路郵件用戶端
- 搜尋結果網頁
- 試算表應用程式，例如貸款計算程式和報稅應用程式

當您設計具有 DataGrid 組件的應用程式時，瞭解 List 組件的設計原理將會對您相當有幫助，因為 DataGrid 類別會擴充 SelectableList 類別。如需有關 SelectableList 類別和 List 組件的詳細資訊，請參閱 「ActionScript 3.0 語言和組件參考」中的 ActionScript 3.0 參考中的 「SelectableList 類別」和 「List 類別」。

將 DataGrid 組件加入應用程式時，您可以加入下列 ActionScript 程式碼，讓螢幕朗讀程式能夠存取此組件：

```
import fl.accessibility.DataGridAccImpl;
DataGridAccImpl.enableAccessibility();
```

不論組件有多少個實體，您只需要啟用組件的輔助功能一次。如需詳細資訊，請參閱 「使用 Flash」中的第 18 章 「建立輔助功能內容」。

DataGrid 組件參數

您可以在 「屬性」檢測器或 「組件檢測器」中，為每個 DataGrid 組件實體設定下列編寫參數：allowMultipleSelection、editable、headerHeight、horizontalLineScrollSize、horizontalPageScrollSize、horizontalScrolllPolicy、resizableColumns、rowHeight、showHeaders、verticalLineScrollSize、verticalPageScrollSize 和 verticalScrollPolicy。這些參數都具有相對應的 ActionScript 同名屬性。如需有關這些參數可能值的詳細資訊，請參閱 Flash Professional 的 ActionScript 3.0 參考中的 DataGrid 類別。

建立具有 DataGrid 組件的應用程式

若要建立具有 DataGrid 組件的應用程式，您必須先確定資料的來源。一般而言，資料都來自 Array，您可以設定 dataProvider 屬性，將 Array 拉入格線中。您也可以使用 DataGrid 和 DataGridColumn 類別的方法，將資料加入格線。

將本機資料提供者與 **DataGrid** 組件搭配使用：
此範例會建立 DataGrid，顯示壘球隊的名冊。名冊定義於 Array (aRoster) 中，然後指定給 DataGrid 的 dataProvider 屬性。

1. 在 Flash 中，選取 「檔案 > 新增」，然後選取 「Flash 檔案 (ActionScript 3.0)」。
2. 將 DataGrid 組件從 「組件」面板拖曳到 「舞台」。
3. 在 「屬性」檢測器中，輸入實體名稱 **aDg**。
4. 開啟 「動作」面板，選取主要 「時間軸」中的 「影格 1」，然後輸入下列 ActionScript 程式碼：

```
import fl.data.DataProvider;

bldRosterGrid(aDg);
var aRoster:Array = new Array();
aRoster = [
        {Name:"Wilma Carter", Bats:"R", Throws:"R", Year:"So", Home: "Redlands, CA"},
        {Name:"Sue Pennypacker", Bats:"L", Throws:"R", Year:"Fr", Home: "Athens, GA"},
        {Name:"Jill Smithfield", Bats:"R", Throws:"L", Year:"Sr", Home: "Spokane, WA"},
        {Name:"Shirley Goth", Bats:"R", Throws:"R", Year:"Sr", Home: "Carson, NV"},
        {Name:"Jennifer Dunbar", Bats:"R", Throws:"R", Year:"Fr", Home: "Seaside, CA"},
        {Name:"Patty Crawford", Bats:"L", Throws:"L", Year:"Jr", Home: "Whittier, CA"},
        {Name:"Angelina Davis", Bats:"R", Throws:"R", Year:"So", Home: "Odessa, TX"},
        {Name:"Maria Santiago", Bats:"L", Throws:"L", Year:"Sr", Home: "Tacoma, WA"},
        {Name:"Debbie Ferguson", Bats:"R", Throws:"R", Year: "Jr", Home: "Bend, OR"},
        {Name:"Karen Bronson", Bats:"R", Throws:"R", Year: "Sr", Home: "Billings, MO"},
        {Name:"Sylvia Munson", Bats:"R", Throws:"R", Year: "Jr", Home: "Pasadena, CA"},
        {Name:"Carla Gomez", Bats:"R", Throws:"L", Year: "Sr", Home: "Corona, CA"},
        {Name:"Betty Kay", Bats:"R", Throws:"R", Year: "Fr", Home: "Palo Alto, CA"},
];
aDg.dataProvider = new DataProvider(aRoster);
aDg.rowCount = aDg.length;

function bldRosterGrid(dg:DataGrid){
    dg.setSize(400, 300);
    dg.columns = ["Name", "Bats", "Throws", "Year", "Home"];
    dg.columns[0].width = 120;
    dg.columns[1].width = 50;
    dg.columns[2].width = 50;
    dg.columns[3].width = 40;
    dg.columns[4].width = 120;
    dg.move(50,50);
};
```

bldRosterGrid() 函數會設定 DataGrid 的大小，並設定欄的順序及大小。

5 選取 「控制 > 測試影片」。

為應用程式中的 **DataGrid** 組件指定欄並加入排序功能

請注意，您可以按一下任何一個欄標題，根據該欄的值，以遞減的順序來排序 DataGrid 的內容。

下列範例使用 addColumn() 方法，將 DataGridColumn 實體加入至 DataGrid。這些欄代表選手名稱與其得分。此範例也會設定 sortOptions 屬性，以指定各欄的排序選項：Name 欄為 Array.CASEINSENSITIVE，而 Score 欄則為 Array.NUMERIC。DataGrid 的長度將設定為列數，且寬度設定為 200，以適當地調整大小。

1 在 Flash 中，選取 「檔案 > 新增」，然後選取 「Flash 檔案 (ActionScript 3.0)」。

2 將 DataGrid 組件從 「組件」面板拖曳到 「舞台」。

3 在 「屬性」檢測器中，輸入實體名稱 **aDg**。

4 開啟 「動作」面板，選取主要 「時間軸」中的 「影格 1」，然後輸入下列 ActionScript 程式碼：

```
import fl.controls.dataGridClasses.DataGridColumn;
import fl.events.DataGridEvent;
import fl.data.DataProvider;
// Create columns to enable sorting of data.
var nameDGC:DataGridColumn = new DataGridColumn("name");
nameDGC.sortOptions = Array.CASEINSENSITIVE;
var scoreDGC:DataGridColumn = new DataGridColumn("score");
scoreDGC.sortOptions = Array.NUMERIC;
aDg.addColumn(nameDGC);
aDg.addColumn(scoreDGC);
var aDP_array:Array = new Array({name:"clark", score:3135}, {name:"Bruce", score:403}, {name:"Peter",
score:25})
aDg.dataProvider = new DataProvider(aDP_array);
aDg.rowCount = aDg.length;
aDg.width = 200;
```

5 選取「控制 > 測試影片」。

使用 **ActionScript** 建立 **DataGrid** 組件實體

此範例使用 ActionScript 建立 DataGrid，並在其中填入由選手名稱及得分構成的 Array。

1 建立新的 Flash (ActionScript 3.0) 文件。

2 將 DataGrid 組件從「組件」面板拖曳到目前文件的「元件庫」面板。

此舉會將組件加入元件庫中，但不會使其在應用程式中顯示。

3 開啟「動作」面板，選取主要「時間軸」中的「影格 1」，然後輸入下列 ActionScript 程式碼：

```
import fl.controls.DataGrid;
import fl.data.DataProvider;

var aDg:DataGrid = new DataGrid();
addChild(aDg);
aDg.columns = [ "Name", "Score" ];
aDg.setSize(140, 100);
aDg.move(10, 40);
```

此程式碼會建立 DataGrid 實體，然後調整格線大小及位置。

4 建立陣列、將資料加入陣列中，並且指定陣列為 DataGrid 的資料提供者：

```
var aDP_array:Array = new Array();
aDP_array.push({Name:"Clark", Score:3135});
aDP_array.push({Name:"Bruce", Score:403});
aDP_array.push({Name:"Peter", Score:25});
aDg.dataProvider = new DataProvider(aDP_array);
aDg.rowCount = aDg.length;
```

5 選取「控制 > 測試影片」。

將 **XML** 檔載入 **DataGrid**

下列範例使用 DataGridColumn 類別，建立 DataGrid 的欄。它會傳遞 XML 物件做為 DataProvider() 建構函式的 value 參數，藉以填入 DataGrid。

1 使用文字編輯器以下列資料建立 XML 檔案，並儲存為 team.xml（儲存的位置為儲存 FLA 檔案相同的資料夾）。

```
<team>
    <player name="Player A" avg="0.293" />
    <player name="Player B" avg="0.214" />
    <player name="Player C" avg="0.317" />
</team>
```

2 建立新的 Flash (ActionScript 3.0) 文件。

3 在「組件」面板中，按兩下 DataGrid，將組件加入至「舞台」。

4 在「屬性」檢測器中，輸入實體名稱 **aDg**。

5 開啟「動作」面板，選取主要「時間軸」中的「影格 1」，然後輸入下列 ActionScript 程式碼：

```
import fl.controls.dataGridClasses.DataGridColumn;
import fl.data.DataProvider;
import flash.net.*;
import flash.events.*;

var request:URLRequest = new URLRequest("team.xml");
var loader:URLLoader = new URLLoader;


loader.load(request);
loader.addEventListener(Event.COMPLETE, loaderCompleteHandler);

function loaderCompleteHandler(event:Event):void {

    var teamXML:XML = new XML(loader.data);

    var nameCol:DataGridColumn = new DataGridColumn("name");
    nameCol.headerText = "Name";
    nameCol.width = 120;
    var avgCol:DataGridColumn = new DataGridColumn("avg");
    avgCol.headerText = "Average";
    avgCol.width = 60;

    var myDP:DataProvider = new DataProvider(teamXML);

    aDg.columns = [nameCol, avgCol];
    aDg.width = 200;
    aDg.dataProvider = myDP;
    aDg.rowCount = aDg.length;
}
```

6 選取「控制 > 測試影片」。

使用 Label 組件

此 Label 組件會顯示單行文字，通常用來識別網頁上的其它元素或活動。您可以指定具有 HTML 格式的標籤 (Label)，以利用其文字格式化標籤的優點；也可以控制標籤的對齊和大小調整。Label 組件沒有邊框，不能成為焦點，也不能傳送任何事件。

進行編寫時，每個 Label 實體的即時預覽會反映您在「屬性」檢測器或「組件檢測器」中對參數的變更。Label 沒有邊框，所以必須透過設定文字參數才能看到即時預覽。

與 Label 組件的使用者互動

使用 Label 組件可為表單中的另一個組件建立文字標籤。例如，在接受使用者名稱的 TextInput 欄位左邊建立「名稱：」標籤。使用 Label 組件來取代純文字欄位是個不錯的做法，因為您可以利用樣式保持一致的外觀與操作。

如果想要旋轉 Label 組件，就必須內嵌字體；否則在測試影片時，這些字體就不會出現。

Label 組件參數

您可以在「屬性」檢測器或「組件檢測器」中，為每個 Label 組件實體設定下列編寫參數：autoSize、condenseWhite、selectable、text 和 wordWrap。這些參數都具有相對應的 ActionScript 同名屬性。如需有關這些參數可能值的詳細資訊，請參閱 Flash Professional 的 ActionScript 3.0 參考中的 Label 類別。

建立具有 Label 組件的應用程式

下列程序說明如何在編寫時將 Label 組件加入應用程式。在此範例中，標籤只會顯示「Expiration Date」文字而已。

1 建立新的 Flash (ActionScript 3.0) 文件。

2 將 Label 組件從「組件」面板拖曳到「舞台」，然後在「屬性」檢測器中為組件指定下列值：

- 輸入 **aLabel** 做為實體名稱。
- 輸入 **80** 做為 W 值。
- 輸入 **100** 做為 X 值。
- 輸入 **100** 做為 Y 值。
- 輸入 **Expiration Date** 做為 text 參數的值。

3 將 TextArea 組件拖曳到「舞台」，然後在「屬性」檢測器中為組件指定下列值：

- 輸入 **aTa** 做為實體名稱。
- 輸入 **22** 做為 H 值。
- 輸入 **200** 做為 X 值。
- 輸入 **100** 做為 Y 值。

4 開啟「動作」面板，選取主要「時間軸」中的「影格 1」，然後輸入下列 ActionScript 程式碼：

```
var today:Date = new Date();
var expDate:Date = addDays(today, 14);
aTa.text = expDate.toDateString();

function addDays(date:Date, days:Number):Date {
return addHours(date, days*24);
}

function addHours(date:Date, hrs:Number):Date {
return addMinutes(date, hrs*60);
}

function addMinutes(date:Date, mins:Number):Date {
return addSeconds(date, mins*60);
}

function addSeconds(date:Date, secs:Number):Date {
var mSecs:Number = secs * 1000;
var sum:Number = mSecs + date.getTime();
return new Date(sum);
}
```

5 選取「控制 > 測試影片」。

使用 ActionScript 建立 Label 組件實體

下列範例使用 ActionScript 建立 Label 組件，以利用 Label 識別 ColorPicker 組件的功能，並且透過 htmlText 屬性將格式套用到 Label 的文字。

1 建立新的 Flash (ActionScript 3.0) 文件。

2 將 Label 組件從 「組件」面板拖曳到目前文件的 「元件庫」面板。

3 將 ColorPicker 組件從 「組件」面板拖曳到目前文件的 「元件庫」面板。

4 開啟 「動作」面板，選取主要 「時間軸」中的 「影格 1」，然後輸入下列 ActionScript 程式碼：

```
import fl.controls.Label;
import fl.controls.ColorPicker;

var aLabel:Label = new Label();
var aCp:ColorPicker = new ColorPicker();

addChild(aLabel);
addChild(aCp);

aLabel.htmlText = '<font face="Arial" color="#FF0000" size="14">Fill:</font>';
aLabel.x = 200;
aLabel.y = 150;
aLabel.width = 25;
aLabel.height = 22;

aCp.x = 230;
aCp.y = 150;
```

5 選取 「控制 > 測試影片」。

使用 List 組件

此 List 組件是一個可以捲動的單選或複選清單方塊。清單也能顯示圖像 (包括其它組件)。若要加入項目供清單顯示，請按一下 label 或 data 參數欄位，開啟 「值」對話方塊來指定。您也可以使用 List.addItem() 和 List.addItemAt() 方法加入項目至清單。

List 組件使用以零為基底的索引；索引為 0 的項目就是第一個顯示的項目。使用 List 類別方法和屬性來加入、移除或取代清單項目時，您可能需要指定清單項目的索引。

與 List 組件的使用者互動

您可以設定清單，讓使用者進行單選或複選。例如，瀏覽電子商務網站的使用者會需要選擇想購買的項目。清單中有 30 個項目，使用者只要捲動清單，按一下上面的項目即可選取該項目。

您也可以設計一份使用自訂影片片段做為列的 List，為使用者提供更多資訊。例如，在電子郵件應用程式中，每個信箱都可以是 List 組件，而且每列都可以擁有指示優先順序和狀態的圖示。

如果您按一下或者用 Tab 鍵跳到 List 上，它就會成為焦點，接著您可使用下列按鍵加以控制：

按鍵	說明
英數字母按鍵	跳到以 Key.getAscii() 做為標籤中第一個字元的下一個項目。
Control	切換按鍵以進行多個非連續的選取與取消選取動作。
向下鍵	選取範圍向下移動一個項目。

按鍵	說明
首頁	選取範圍移到清單頂端。
Page Down	選取範圍向下移動一頁。
Page Up	選取範圍向上移動一頁。
Shift	可進行連續選取。
向上鍵	選取範圍向上移動一個項目。

備註：請注意，捲動的大小是以像素 (而非列) 為單位。

備註：Page Up 和 Page Down 鍵所使用的頁面大小比頁面所能容納的項目數目少一。例如，在一個十行的下拉式清單向下翻頁將會顯示項目 0-9、9-18、18-27，依此類推，每一頁有一個重疊項目。

如需有關控制焦點的詳細資訊，請參閱 Flash Professional 的 ActionScript 3.0 參考中的 「IFocusManager 介面」和「FocusManager 類別」，以及第 24 頁 「使用 FocusManager」。

進行編寫時，「舞台」上每個 List 實體的即時預覽會反映您在 「屬性」檢測器或 「組件檢測器」中對參數的變更。

將 List 組件加入應用程式時，您可以加入下列 ActionScript 程式碼，讓螢幕朗讀程式能夠存取此組件：

```
import fl.accessibility.ListAccImpl;

ListAccImpl.enableAccessibility();
```

不論組件有多少個實體，您只需要啟用組件的輔助功能一次。如需詳細資訊，請參閱 「使用 Flash」中的第 18 章 「建立輔助功能內容」。

List 組件參數

您可以在 「屬性」檢測器或 「組件檢測器」中，為每個 List 組件實體設定下列參數：allowMultipleSelection、dataProvider、horizontalLineScrollSize、horizontalPageScrollSize、horizontalScrollPolicy、multipleSelection、verticalLineScrollSize、verticalPageScrollSize 和 verticalScrollPolicy。這些參數都具有相對應的 ActionScript 同名屬性。如需有關這些參數可能值的詳細資訊，請參閱 Flash Professional 的 ActionScript 3.0 參考中的 List 類別。如需有關 dataProvider 參數用法的詳細資訊，請參閱第 25 頁 「使用 dataProvider 參數」。

建立具有 List 組件的應用程式

下列範例說明如何在編寫時將 List 組件加入應用程式。

將簡易清單 List 組件加入應用程式

在此範例中，List 是由識別車型的標籤以及包含價格的資料欄位所組成。

1. 建立新的 Flash (ActionScript 3.0) 文件。
2. 將 List 組件從 「組件」面板拖曳到 「舞台」。
3. 在 「屬性」檢測器中，執行下列步驟：
   - 輸入實體名稱 **aList**。
   - 指定 **200** 做為 W (寬度) 值。
4. 使用 「文字」工具在 aList 下方建立文字欄位，並且賦予實體名稱 **aTf**。
5. 開啟 「動作」面板，選取主要 「時間軸」中的 「影格 1」，然後輸入下列 ActionScript 程式碼：

```
import fl.controls.List;
import flash.text.TextField;

aTf.type = TextFieldType.DYNAMIC;
aTf.border = false;

// Create these items in the Property inspector when data and label
// parameters are available.
aList.addItem({label:"1956 Chevy (Cherry Red)", data:35000});
aList.addItem({label:"1966 Mustang (Classic)", data:27000});
aList.addItem({label:"1976 Volvo (Xcllnt Cond)", data:17000});
aList.allowMultipleSelection = true;

aList.addEventListener(Event.CHANGE, showData);

function showData(event:Event) {
    aTf.text = "This car is priced at: $" + event.target.selectedItem.data;
}
```

此程式碼會使用 addItem() 方法，將三個項目填入 aList，並為每個項目指定 label 值 (出現在清單中) 以及 data 值。當您選取 List 中的項目時，事件偵聽程式便會呼叫 showData() 函數，進而顯示所選取項目的 data 值。

6 選取 「控制 > 測試影片」，編譯並執行此應用程式。

透過資料提供者填入 **List** 實體

此範例所建立的 List 也包含車型及其價格。不過，填入 List 的方式是透過資料提供者，而非使用 addItem() 方法。

1 建立新的 Flash (ActionScript 3.0) 文件。

2 將 List 組件從 「組件」面板拖曳到 「舞台」。

3 在 「屬性」檢測器中，執行下列步驟：

- 輸入實體名稱 **aList**。
- 指定 **200** 做為 W (寬度) 值。

4 使用 「文字」工具在 aList 下方建立文字欄位，並且賦予實體名稱 **aTf**。

5 開啟 「動作」面板，選取主要 「時間軸」中的 「影格 1」，然後輸入下列 ActionScript 程式碼：

```
import fl.controls.List;
import fl.data.DataProvider;
import flash.text.TextField;

aTf.type = TextFieldType.DYNAMIC;
aTf.border = false;

var cars:Array = [
{label:"1956 Chevy (Cherry Red)", data:35000},
{label:"1966 Mustang (Classic)", data:27000},
{label:"1976 Volvo (Xcllnt Cond)", data:17000},
];
aList.dataProvider = new DataProvider(cars);
aList.allowMultipleSelection = true;

aList.addEventListener(Event.CHANGE, showData);

function showData(event:Event) {
    aTf.text = "This car is priced at: $" + event.target.selectedItem.data;
}
```

6 選取 「控制 > 測試影片」，查看 List 及其項目。

使用 **List** 組件控制 **MovieClip** 實體

下列範例會建立包含顏色名稱的 List，並且當您選取某個顏色時，該顏色就會套用到 MovieClip。

1 建立 Flash (ActionScript 3.0) 文件。

2 將 List 組件從 「組件」面板拖曳到 「舞台」，然後在 「屬性」檢測器中為組件指定下列值：

- 輸入 **aList** 做為實體名稱。
- 輸入 **60** 做為 H 值。
- 輸入 **100** 做為 X 值。
- 輸入 **150** 做為 Y 值。

3 開啟 「動作」面板，選取主要 「時間軸」中的 「影格 1」，然後輸入下列 ActionScript 程式碼：

```
aList.addItem({label:"Blue", data:0x0000CC});
aList.addItem({label:"Green", data:0x00CC00});
aList.addItem({label:"Yellow", data:0xFFFF00});
aList.addItem({label:"Orange", data:0xFF6600});
aList.addItem({label:"Black", data:0x000000});

var aBox:MovieClip = new MovieClip();
addChild(aBox);

aList.addEventListener(Event.CHANGE, changeHandler);
function changeHandler(event:Event) {
    drawBox(aBox, event.target.selectedItem.data);
};

function drawBox(box:MovieClip,color:uint):void {
            box.graphics.beginFill(color, 1.0);
            box.graphics.drawRect(225, 150, 100, 100);
            box.graphics.endFill();
}
```

4 選取 「控制 > 測試影片」，執行應用程式。

5 按一下 List 中的顏色，讓這些顏色在 MovieClip 中顯示。

使用 **ActionScript** 建立 **List** 組件實體：

此範例使用 ActionScript 建立一份簡易清單，並使用 addItem() 方法填入清單。

1 建立新的 Flash (ActionScript 3.0) 文件。

2 將 List 組件從 「組件」面板拖曳到 「元件庫」面板。

3 開啟 「動作」面板，選取主要 「時間軸」中的 「影格 1」，然後輸入下列 ActionScript 程式碼：

```
import fl.controls.List;

var aList:List = new List();
aList.addItem({label:"One", data:1});
aList.addItem({label:"Two", data:2});
aList.addItem({label:"Three", data:3});
aList.addItem({label:"Four", data:4});
aList.addItem({label:"Five", data:5});
aList.setSize(60, 40);
aList.move(200,200);
addChild(aList);
aList.addEventListener(Event.CHANGE, changeHandler);
function changeHandler(event:Event):void {
    trace(event.target.selectedItem.data);
}
```

4 選取 「控制 > 測試影片」，執行應用程式。

使用 NumericStepper 組件

此 NumericStepper 組件可以讓使用者依序逐步增減數字。這個組件是由小型上下箭頭旁文字方塊內的數字所構成。當使用者按下按鈕時，數字會依據 stepSize 參數所指定的單位遞增或遞減，直到使用者放開按鈕或達到最大值或最小值為止。您還可以編輯 NumericStepper 組件文字方塊內的文字。

每個 NumericStepper 實體的即時預覽會反映 「屬性」檢測器或 「組件檢測器」中的 value 參數設定。但是，即時預覽下無法用滑鼠或鍵盤與 NumericStepper 的方向鍵互動。

與 NumericStepper 組件的使用者互動

在您希望使用者選取數值的任何地方，NumericStepper 組件都能派上用場。例如，您可以在表單中使用 NumericStepper 組件，設定信用卡到期的年、月、日。您也可以使用 NumericStepper 組件讓使用者加大或縮小字體。

NumericStepper 組件只能處理數值資料。而且，如果要顯示超過兩位數字 (例如數字 5246 或 1.34)，編寫時必須調整步進器的大小。

您可以在應用程式中啟用或停用 NumericStepper。處於停用狀態的 NumericStepper 不接受滑鼠或鍵盤輸入。當 NumericStepper 啟用時，如果您按一下或用 Tab 鍵移到該組件，它就會成為焦點，且其內部焦點設定為文字方塊。在 NumericStepper 實體成為焦點後，您可以使用下列按鍵加以控制：

按鍵	說明
向下鍵	值變更一個單位。
向左鍵	將插入點移到文字方塊內的左邊。
向右鍵	將插入點移到文字方塊內的右邊。
Shift+Tab	將焦點移到上一個物件。
Tab 鍵	將焦點移到下一個物件。
向上鍵	值變更一個單位。

如需有關控制焦點的詳細資訊，請參閱 Flash Professional 的 ActionScript 3.0 參考中的 「FocusManager 類別」，以及 第 24 頁 「使用 FocusManager」。

NumericStepper 組件參數

您可以在 「屬性」檢測器或 「組件檢測器」中，為每個 NumericStepper 實體設定下列參數：maximum、minimum、stepSize 和 value。這些參數都具有相對應的 ActionScript 同名屬性。如需有關這些參數可能值的詳細資訊，請參閱 Flash Professional 的 ActionScript 3.0 參考中的 NumericStepper 類別。

建立具有 NumericStepper 的應用程式

下列程序說明如何在編寫時將 NumericStepper 組件加入應用程式。此範例會在 「舞台」上放置 NumericStepper 組件和 Label 組件，並且為 NumericStepper 實體建立 Event.CHANGE 事件的偵聽程式。當數值步進器的值變更時，範例即透過 Label 實體的 text 屬性顯示新的值。

1 將 NumericStepper 組件從 「組件」面板拖曳到 「舞台」。

2 在 「屬性」檢測器中，輸入實體名稱 **aNs**。

3 將 Label 組件從「組件」面板拖曳到「舞台」。

4 在「屬性」檢測器中，輸入實體名稱 **aLabel**。

5 開啟「動作」面板，選取主要「時間軸」中的「影格 1」，然後輸入下列 ActionScript 程式碼：

```
import flash.events.Event;

aLabel.text = "value = " + aNs.value;

aNs.addEventListener(Event.CHANGE, changeHandler);
function changeHandler(event:Event) :void {
    aLabel.text = "value = " + event.target.value;
};
```

此範例會將標籤的 text 屬性設定為 NumericStepper 的值。每當 NumericStepper 實體的值有所變更，changeHandler() 函數都將更新該標籤的 text 屬性。

6 選取「控制 > 測試影片」。

使用 **ActionScript** 建立 **NumericStepper**：

此範例使用 ActionScript 程式碼建立三個 NumericStepper，以供使用者分別輸入生日的月份、日期及年份。另外還會加入三個 Label，做為每個 NumericStepper 的提示文字和識別項。

1 建立新的 Flash (ActionScript 3.0) 文件。

2 將 Label 拖曳到「元件庫」面板。

3 將 NumericStepper 組件拖曳到「元件庫」面板。

4 開啟「動作」面板，選取主要「時間軸」中的「影格 1」，然後輸入下列 ActionScript 程式碼：

```
import fl.controls.Label;
import fl.controls.NumericStepper;

var dobPrompt:Label = new Label();
var moPrompt:Label = new Label();
var dayPrompt:Label = new Label();
var yrPrompt:Label = new Label();

var moNs:NumericStepper = new NumericStepper();
var dayNs:NumericStepper = new NumericStepper();
var yrNs:NumericStepper = new NumericStepper();

addChild(dobPrompt);
addChild(moPrompt);
addChild(dayPrompt);
addChild(yrPrompt);
addChild(moNs);
addChild(dayNs);
addChild(yrNs);

dobPrompt.setSize(65, 22);
dobPrompt.text = "Date of birth:";
dobPrompt.move(80, 150);

moNs.move(150, 150);
moNs.setSize(40, 22);
moNs.minimum = 1;
moNs.maximum = 12;
moNs.stepSize = 1;
moNs.value = 1;

moPrompt.setSize(25, 22);
```

```
moPrompt.text = "Mo.";
moPrompt.move(195, 150);

dayNs.move(225, 150);
dayNs.setSize(40, 22);
dayNs.minimum = 1;
dayNs.maximum = 31;
dayNs.stepSize = 1;
dayNs.value = 1;

dayPrompt.setSize(25, 22);
dayPrompt.text = "Day";
dayPrompt.move(270, 150);

yrNs.move(300, 150);
yrNs.setSize(55, 22);
yrNs.minimum = 1900;
yrNs.maximum = 2006;
yrNs.stepSize = 1;
yrNs.value = 1980;

yrPrompt.setSize(30, 22);
yrPrompt.text = "Year";
yrPrompt.move(360, 150);
```

5 選取 「控制 > 測試影片」，執行應用程式。

使用 ProgressBar 組件

此 ProgressBar 組件會顯示載入內容的進度，不斷提醒使用者內容龐大時可能會耽擱應用程式的執行。當您需要顯示載入影像或應用程式特定部分的進度時，ProgressBar 就非常有用。載入過程有兩種情況：確定式及非確定式。「確定式」的進度列代表工作進度隨著時間呈線性遞增，使用於已確知所要載入內容的數量時。「非確定式」的進度列則是使用於不確定所要載入內容的數量時。您也可以加入 Label 組件，顯示載入進度百分比。

ProgressBar 組件會使用 9 分割縮放，而且具有列外觀元素、軌道外觀元素和非確定式外觀元素。

與 ProgressBar 組件的使用者互動

ProgressBar 組件有三種使用模式。最常用的模式為事件模式與輪詢模式。這些模式指定的載入程序會發出 progress 和 complete 事件 (事件與輪詢模式)，或者顯露 bytesLoaded 和 bytesTotal 屬性 (輪詢模式)。您也可藉由設定 maximum、minimum 和 value 屬性，同時呼叫 ProgressBar.setProgress() 方法，以使用手動模式的 ProgressBar 組件。如果 indeterminate 屬性為 true，表示 ProgressBar 具有條紋填色及不明大小的載入來源，false 則表示具有純色填色及已知大小的載入來源。

若要設定 ProgressBar 的模式，請透過 「屬性」檢測器或 「組件檢測器」設定 mode 參數，或使用 ActionScript 設定 mode 屬性。

如果您在單一影格迴圈中使用 ProgressBar 顯示處理的狀態 (例如剖析 100,000 個項目)，ProgressBar 就不會反映更新情形，因為螢幕重繪並未發生。

ProgressBar 組件參數

您可以在 「屬性」檢測器或 「組件檢測器」中，為每個 ProgressBar 實體設定下列參數：direction、mode 和 source。這些參數都具有相對應的 ActionScript 同名屬性。

您可以撰寫 ActionScript 利用 ProgressBar 組件的屬性、方法和事件，來控制上列各種選項及其它選項。如需詳細資訊，請參閱 Flash Professional 的 ActionScript 3.0 參考中的 ProgressBar 類別。

建立具有 ProgressBar 的應用程式

下列程序說明如何在編寫時將 ProgressBar 組件加入應用程式。在此範例中，ProgressBar 使用事件模式。若為事件模式，載入內容時會發出 progress 和 complete 事件，這些是由 ProgressBar 所傳送用來指示進度的事件。當 progress 事件發生時，範例就會更新標籤，以顯示已下載內容的百分比。當 complete 事件發生時，範例則會顯示 "Loading complete" 和 bytesTotal 屬性的值 (代表檔案大小)。

1 建立新的 Flash (ActionScript 3.0) 文件。

2 將 ProgressBar 組件從 「組件」面板拖曳到 「舞台」。

- 在 「屬性」檢測器中，輸入實體名稱 **aPb**。
- 在 「參數」區段，輸入 **200** 做為 X 值。
- 輸入 **260** 做為 Y 值。
- 選取 event 做為 mode 參數的值。

3 將 Button 組件從 「組件」面板到 「舞台」。

- 在 「屬性」檢測器中，輸入 **loadButton** 做為實體名稱。
- 輸入 **220** 做為 X 參數的值。
- 輸入 **290** 做為 Y 參數的值。
- 輸入 **Load Sound** 做為 label 參數的值。

4 將 Label 組件拖曳到 「舞台」，並且為它指定實體名稱為 **progLabel**。

- 輸入 **150** 做為 W 值。
- 輸入 **200** 做為 X 參數的值。
- 輸入 **230** 做為 Y 參數的值。
- 在 「參數」區段，清除 text 參數的值。

5 開啟 「動作」面板，選取主要 「時間軸」中的 「影格 1」，然後輸入下列 ActionScript 程式碼，載入 mp3 音效檔：

```
import fl.controls.ProgressBar;
import flash.events.ProgressEvent;
import flash.events.IOErrorEvent;

var aSound:Sound = new Sound();
aPb.source = aSound;
var url:String = "http://www.helpexamples.com/flash/sound/song1.mp3";
var request:URLRequest = new URLRequest(url);

aPb.addEventListener(ProgressEvent.PROGRESS, progressHandler);
aPb.addEventListener(Event.COMPLETE, completeHandler);
aSound.addEventListener(IOErrorEvent.IO_ERROR, ioErrorHandler);
loadButton.addEventListener(MouseEvent.CLICK, clickHandler);

function progressHandler(event:ProgressEvent):void {
        progLabel.text = ("Sound loading ... " + aPb.percentComplete);
}

function completeHandler(event:Event):void {
    trace("Loading complete");
    trace("Size of file: " + aSound.bytesTotal);
    aSound.close();
    loadButton.enabled = false;
}

function clickHandler(event:MouseEvent) {
    aSound.load(request);
}

function ioErrorHandler(event:IOErrorEvent):void {
    trace("Load failed due to: " + event.text);
}
```

6 選取「控制 > 測試影片」。

建立具有輪詢模式 **ProgressBar** 組件的應用程式

下列範例將 ProgressBar 設定為輪詢模式。輪詢模式會藉由偵聽當前所載入內容的 progress 事件來判斷進度，而進度的算法則是依據 bytesLoaded 和 bytesTotal 屬性。此範例會載入 Sound 物件、偵聽該物件的 progress 事件，並使用物件的 bytesLoaded 和 bytesTotal 屬性來計算載入百分比。載入百分比將顯示於標籤上及「輸出」面板。

1 建立新的 Flash (ActionScript 3.0) 文件。

2 將 ProgressBar 組件從「組件」面板拖曳到「舞台」，然後在「屬性」檢測器中輸入下列值：

- 輸入 **aPb** 做為實體名稱。
- 輸入 **185** 做為 X 值。
- 輸入 **225** 做為 Y 值。

3 將 Label 組件拖曳到「舞台」，然後在「屬性」檢測器中輸入下列值：

- 輸入 **progLabel** 做為實體名稱。
- 輸入 **180** 做為 X 值。
- 輸入 **180** 做為 Y 值。
- 在「參數」區段，清除 text 參數的值。

4 開啟「動作」面板，選取主要「時間軸」中的「影格 1」，然後輸入下列 ActionScript 程式碼，建立 Sound 物件 (aSound) 並呼叫 loadSound() 將聲音載入 Sound 物件中：

```
import fl.controls.ProgressBarMode;
import flash.events.ProgressEvent;
import flash.media.Sound;

var aSound:Sound = new Sound();
var url:String = "http://www.helpexamples.com/flash/sound/song1.mp3";
var request:URLRequest = new URLRequest(url);

aPb.mode = ProgressBarMode.POLLED;
aPb.source = aSound;
aSound.addEventListener(ProgressEvent.PROGRESS, loadListener);

aSound.load(request);

function loadListener(event:ProgressEvent) {
    var percentLoaded:int = event.target.bytesLoaded / event.target.bytesTotal * 100;
    progLabel.text = "Percent loaded: " + percentLoaded + "%";
    trace("Percent loaded: " + percentLoaded + "%");
}
```

5 選取「控制 > 測試影片」，執行應用程式。

建立具有手動模式 **ProgressBar** 組件的應用程式

下列範例將 ProgressBar 設定為手動模式。使用手動模式時，必須呼叫 setProgress() 方法，並傳遞目前值與最大值供其判斷進度的範圍，藉以手動設定進度。在手動模式中無法設定 source 屬性。此範例會使用最大值為 250 的 NumericStepper 組件來遞增 ProgressBar。一旦 NumericStepper 的值變更而觸發 CHANGE 事件，事件處理常式 nsChangeHander 隨即呼叫 setProgress() 方法將 ProgressBar 往前推進。另外還會根據最大值，顯示完成進度百分比。

1 建立新的 Flash (ActionScript 3.0) 文件。

2 將 ProgressBar 組件從「組件」面板拖曳到「舞台」，然後在「屬性」檢測器中為組件指定下列值：

- 輸入 **aPb** 做為實體名稱。
- 輸入 **180** 做為 X 值。
- 輸入 **175** 做為 Y 值。

3 將 NumericStepper 組件拖曳到「舞台」，然後在「屬性」檢測器中輸入下列值：

- 輸入 **aNs** 做為實體名稱。
- 輸入 **220** 做為 X 值。
- 輸入 **215** 做為 Y 值。
- 在「參數」區段，輸入 **250** 做為 maximum 參數的值、輸入 **0** 做為 minimum 參數的值、輸入 **1** 做為 stepSize 參數的值、輸入 **0** 做為 value 參數的值。

4 將 Label 組件拖曳到「舞台」，然後在「屬性」檢測器中輸入下列值：

- 輸入 **progLabel** 做為實體名稱。
- 輸入 **150** 做為 W 值。
- 輸入 **180** 做為 X 值。
- 輸入 **120** 做為 Y 值。
- 在「參數」索引標籤中，清除 text 參數的值 Label。

5 開啟「動作」面板，選取主要「時間軸」中的「影格 1」，然後輸入下列程式碼：

```
import fl.controls.ProgressBarDirection;
import fl.controls.ProgressBarMode;
import flash.events.Event;

aPb.direction = ProgressBarDirection.RIGHT;
aPb.mode = ProgressBarMode.MANUAL;
aPb.minimum = aNs.minimum;
aPb.maximum = aNs.maximum;
aPb.indeterminate = false;

aNs.addEventListener(Event.CHANGE, nsChangeHandler);

function nsChangeHandler(event:Event):void {
    aPb.value = aNs.value;
    aPb.setProgress(aPb.value, aPb.maximum);
progLabel.text = "Percent of progress = " + int(aPb.percentComplete) + "%";
}
```

6 選取「控制 > 測試影片」，執行應用程式。

7 按一下 NumericStepper 上的「向上箭頭」，讓 ProgressBar 往前進。

使用 **ActionScript** 建立 **ProgressBar**

此範例會使用 ActionScript 建立 ProgressBar。除了這點不同以外，功能與前述範例一樣，仍是建立手動模式 ProgressBar。

1 建立新的 Flash (ActionScript 3.0) 文件。

2 將 ProgressBar 組件拖曳到「元件庫」面板。

3 將 NumericStepper 組件拖曳到「元件庫」面板。

4 將 Label 組件拖曳到「元件庫」面板。

5 開啟「動作」面板，選取主要「時間軸」中的「影格 1」，然後輸入下列程式碼：

```
import fl.controls.ProgressBar;
import fl.controls.NumericStepper;
import fl.controls.Label;
import fl.controls.ProgressBarDirection;
import fl.controls.ProgressBarMode;
import flash.events.Event;

var aPb:ProgressBar = new ProgressBar();
var aNs:NumericStepper = new NumericStepper();
var progLabel:Label = new Label();

addChild(aPb);
addChild(aNs);
addChild(progLabel);

aPb.move(180,175);
aPb.direction = ProgressBarDirection.RIGHT;
aPb.mode = ProgressBarMode.MANUAL;

progLabel.setSize(150, 22);
progLabel.move(180, 150);
progLabel.text = "";

aNs.move(220, 215);
aNs.maximum = 250;
aNs.minimum = 0;
aNs.stepSize = 1;
aNs.value = 0;

aNs.addEventListener(Event.CHANGE, nsChangeHandler);

function nsChangeHandler(event:Event):void {
aPb.setProgress(aNs.value, aNs.maximum);
progLabel.text = "Percent of progress = " + int(aPb.percentComplete) + "%";
}
```

6 選取 「控制 > 測試影片」，執行應用程式。

7 按一下 NumericStepper 上的 「向上箭頭」，讓 ProgressBar 往前進。

使用 RadioButton 組件

此 RadioButton 組件可以讓您強制使用者在一組選擇中做出一個選擇。使用此組件時，群組中至少必須有兩個 RadioButton 實體。不論任何時候，都只能選取群組中的一個成員。選取群組中的某個選項按鈕後，將會取消選取群組中目前選取的選項按鈕。您應該設定 groupName 參數來指定選項按鈕屬於哪個群組。

選項按鈕是許多表單或網路應用程式的基本項目。如果您希望使用者從選項群組中選取一個選項，您就可以使用選項按鈕。例如，您可以在表單中使用選項按鈕詢問客戶要使用哪一張信用卡。

與 RadioButton 組件的使用者互動

選項按鈕可以啟用或停用。停用的選項按鈕不接受滑鼠或鍵盤輸入。當使用者按一下或用 Tab 鍵移到 RadioButton 組件群組時，只有已選取的選項按鈕會成為焦點。接著使用者便可以用下列按鍵來進行控制：

按鍵	說明
向上鍵 / 向左鍵	選取範圍移到選項按鈕群組中的上一個選項按鈕。
向下鍵 / 向右鍵	選取範圍移到選項按鈕群組中的下一個選項按鈕。
Tab 鍵	將焦點從選項按鈕群組移到下一個組件。

如需有關控制焦點的詳細資訊，請參閱 Flash Professional 的 ActionScript 3.0 參考中的 「IFocusManager 介面」和 「FocusManager 類別」，以及第 24 頁 「使用 FocusManager」。

進行編寫時，「舞台」上每個 RadioButton 實體的即時預覽會反映您在 「屬性」檢測器或 「組件檢測器」中對參數進行的變更。但是，選取範圍的互斥效果不會顯示在即時預覽中。如果您將同一個群組中兩個選項按鈕的 selected 參數都設定為 true，則這兩個選項按鈕都會呈現選取狀態；但是實際上，只有最後建立的實體會在執行階段呈現選取狀態。如需詳細資訊，請參閱第 68 頁 「RadioButton 組件參數」。

將 RadioButton 組件加入應用程式時，您可以加入下列 ActionScript 程式碼，讓螢幕朗讀程式能夠存取此組件：

```
import fl.accessibility.RadioButtonAccImpl;
RadioButtonAccImpl.enableAccessibility();
```

不論您有多少個組件實體，都只需要啟用組件的輔助功能一次。如需詳細資訊，請參閱 「使用 Flash」中的第 18 章 「建立輔助功能內容」。

RadioButton 組件參數

您可以在 「屬性」檢測器或 「組件檢測器」中，為每個 RadioButton 組件實體設定下列編寫參數：groupName、label、LabelPlacement、selected 和 value。這些參數都具有相對應的 ActionScript 同名屬性。如需有關這些參數可能值的詳細資訊，請參閱 Flash Professional 的 ActionScript 3.0 參考中的 RadioButton 類別。

您可以使用 RadioButton 類別的方法、屬性和事件撰寫 ActionScript，來設定 RadioButton 實體的其它選項。

建立具有 RadioButton 組件的應用程式

下列程序說明如何在編寫時將 RadioButton 組件加入應用程式。此範例使用 RadioButton 呈現是非題選項。來自 RadioButton 的資料會顯示在 TextArea 中。

1. 建立新的 Flash (ActionScript 3.0) 文件。
2. 將兩個 RadioButton 組件從 「組件」面板拖曳到 「舞台」。
3. 選取第一個選項按鈕。在 「屬性」檢測器中，賦予實體名稱 **yesRb** 以及群組名稱 **rbGroup**。
4. 選取第二個選項按鈕。在 「屬性」檢測器中，賦予實體名稱 **noRb** 以及群組名稱 **rbGroup**。
5. 將 TextArea 組件從 「組件」面板拖曳到 「舞台」，並賦予實體名稱 **aTa**。
6. 開啟 「動作」面板，選取主要 「時間軸」中的 「影格 1」，然後輸入下列 ActionScript 程式碼：

```
yesRb.label = "Yes";
yesRb.value = "For";
noRb.label = "No";
noRb.value = "Against";

yesRb.move(50, 100);
noRb.move(100, 100);
aTa.move(50, 30);
noRb.addEventListener(MouseEvent.CLICK, clickHandler);
yesRb.addEventListener(MouseEvent.CLICK, clickHandler);

function clickHandler(event:MouseEvent):void {
    aTa.text = event.target.value;
}
```

7 選取「控制 > 測試影片」，執行應用程式。

使用 ActionScript 建立 RadioButton

此範例使用 ActionScript 建立顏色各為紅色、藍色和綠色的三個 RadioButton，並繪製一個灰色方塊。每一個 RadioButton 的 value 屬性都會指定與該按鈕相關顏色的十六進位值。當使用者按一下其中一個 RadioButton 時，clickHandler() 函數就會呼叫 drawBox()，傳遞 RadioButton value 屬性的顏色而為該方塊上色。

1 建立新的 Flash (ActionScript 3.0) 文件。

2 將 RadioButton 組件拖曳到「元件庫」面板。

3 開啟「動作」面板，選取主要「時間軸」中的「影格 1」，然後輸入下列 ActionScript 程式碼：

```
import fl.controls.RadioButton;
import fl.controls.RadioButtonGroup;

var redRb:RadioButton = new RadioButton();
var blueRb:RadioButton = new RadioButton();
var greenRb:RadioButton = new RadioButton();
var rbGrp:RadioButtonGroup = new RadioButtonGroup("colorGrp");

var aBox:MovieClip = new MovieClip();
drawBox(aBox, 0xCCCCCC);

addChild(redRb);
addChild(blueRb);
addChild(greenRb);
addChild(aBox);

redRb.label = "Red";
redRb.value = 0xFF0000;
blueRb.label = "Blue";
blueRb.value = 0x0000FF;
greenRb.label = "Green";
greenRb.value = 0x00FF00;
redRb.group = blueRb.group = greenRb.group = rbGrp;
redRb.move(100, 260);
blueRb.move(150, 260);
greenRb.move(200, 260);

rbGrp.addEventListener(MouseEvent.CLICK, clickHandler);

function clickHandler(event:MouseEvent):void {
    drawBox(aBox, event.target.selection.value);
}

function drawBox(box:MovieClip,color:uint):void {
            box.graphics.beginFill(color, 1.0);
            box.graphics.drawRect(125, 150, 100, 100);
            box.graphics.endFill();
}
```

4 選取「控制 > 測試影片」，執行應用程式。

如需詳細資訊，請參閱 適用於 Adobe Flash Platform 的 ActionScript 3.0 參考中的 RadioButton 類別。

使用 ScrollPane 組件

您可以使用 ScrollPane 組件來顯示因為太大而無法載入顯示區域的任何內容。例如，假設您有一個大型影像，但是應用程式的可用空間太小而容納不下，您就可以將此影像載入到 ScrollPane 中。ScrollPane 可以接受影片片段、JPEG、PNG、GIF 和 SWF 檔。

ScrollPane 和 UILoader 這類組件都有 complete 事件，可讓您判斷內容載入完成的時間。如果您要設定 ScrollPane 或 UILoader 組件內容的屬性，請偵聽 complete 事件，並且在事件處理常式中設定屬性。例如，下列程式碼會建立 Event.COMPLETE 事件的偵聽程式，並且建立事件處理常式而將 ScrollPane 內容的 alpha 屬性設定為 .5：

```
function spComplete(event:Event):void{
aSp.content.alpha = .5;
}
aSp.addEventListener(Event.COMPLETE, spComplete);
```

如果您在載入內容至 ScrollPane 時指定位置，則必須將該位置 (X 和 Y 座標) 指定為 0, 0。例如，下列程式碼會正確地載入 ScrollPane，因為該方塊是在 0, 0 的位置繪製：

```
var box:MovieClip = new MovieClip();
box.graphics.beginFill(0xFF0000, 1);
box.graphics.drawRect(0, 0, 150, 300);
box.graphics.endFill();
aSp.source = box;//load ScrollPane
```

如需詳細資訊，請參閱 適用於 Adobe Flash Platform 的 ActionScript 3.0 參考中的 ScrollPane 類別。

與 ScrollPane 組件的使用者互動

您可以啟用或停用 ScrollPane。停用的 ScrollPane 不接受滑鼠或鍵盤輸入。當 ScrollPane 成為焦點時，使用者便可以用下列按鍵加以控制：

按鍵	說明
向下鍵	將內容向上垂直捲動一行。
向上鍵	將內容向下垂直捲動一行。
End	將內容移到 ScrollPane 底部。
向左鍵	將內容向右水平捲動一行。
向右鍵	將內容向左水平捲動一行。
首頁	將內容移到 ScrollPane 頂端。
End	將內容移到 ScrollPane 底部。
PageDown	將內容向上垂直捲動一頁。
PageUp	將內容向下垂直捲動一頁。

使用者可以利用滑鼠與 ScrollPane 的內容及垂直和水平捲軸進行互動。當 scrollDrag 屬性設定為 true 時，使用者便可以使用滑鼠拖曳內容。在內容上方出現手掌游標表示使用者可以拖曳該內容。和大部分其它控制項的不同之處在於，動作會在按下滑鼠按鈕時發生並一直持續，直到放開按鈕為止。如果內容包含有效的定位停駐點，您必須將 scrollDrag 設定為 false。否則，所有滑鼠點按內容的動作都會叫用捲軸拖曳作業。

ScrollPane 組件參數

您可以在 「屬性」檢測器或 「組件檢測器」中，為每個 ScrollPane 實體設定下列參數：horizontalLineScrollSize、horizontalPageScrollSize、horizontalScrollPolicy、scrollDrag、source、verticalLineScrollSize、verticalPageScrollSize 和 verticalScrollPolicy。這些參數都具有相對應的 ActionScript 同名屬性。如需有關這些參數可能值的詳細資訊，請參閱 適用於 Adobe Flash Platform 的 ActionScript 3.0 參考中的 ScrollPane 類別。

您可以撰寫 ActionScript 利用 ScrollPane 組件的屬性、方法和事件，來控制上列各種選項及其它選項。

建立具有 ScrollPane 組件的應用程式

下列程序說明如何在編寫時將 ScrollPane 組件加入應用程式。在此範例中，ScrollPane 會從 source 屬性指定的路徑載入圖片。

1 建立新的 Flash (ActionScript 3.0) 文件。

2 將 ScrollPane 組件從 「組件」面板拖曳到 「舞台」，並且賦予實體名稱 **aSp**。

3 開啟 「動作」面板，選取主要 「時間軸」中的 「影格 1」，然後輸入下列 ActionScript 程式碼：

```
import fl.events.ScrollEvent;

aSp.setSize(300, 200);

function scrollListener(event:ScrollEvent):void {
    trace("horizontalScPosition: " + aSp.horizontalScrollPosition +
     ", verticalScrollPosition = " + aSp.verticalScrollPosition);
};
aSp.addEventListener(ScrollEvent.SCROLL, scrollListener);

function completeListener(event:Event):void {
    trace(event.target.source + " has completed loading.");
};
// Add listener.
aSp.addEventListener(Event.COMPLETE, completeListener);

aSp.source = "http://www.helpexamples.com/flash/images/image1.jpg";
```

4 選取「控制 > 測試影片」，執行應用程式。

使用 ActionScript 建立 ScrollPane 實體

此範例會建立 ScrollPane 並設定其大小，再使用 source 屬性將影像載入至其中。另外還會建立兩個偵聽程式。前者將偵聽 scroll 事件，當使用者垂直或水平捲動時，會顯示影像的位置。後者則偵聽 complete 事件，並且在「輸出」面板中顯示訊息，告知影像已經完成載入。

此範例會使用 ActionScript 建立 ScrollPane，並且在其中放置 150 像素寬及 300 像素高的 MovieClip (紅色方塊)。

1 建立新的 Flash (ActionScript 3.0) 文件。

2 將 ScrollPane 組件從「組件」面板拖曳到「元件庫」面板。

3 將 DataGrid 組件從「組件」面板拖曳到「元件庫」面板。

4 開啟「動作」面板，選取主要「時間軸」中的「影格 1」，然後輸入下列 ActionScript 程式碼：

```
import fl.containers.ScrollPane;
import fl.controls.ScrollPolicy;
import fl.controls.DataGrid;
import fl.data.DataProvider;

var aSp:ScrollPane = new ScrollPane();
var aBox:MovieClip = new MovieClip();
drawBox(aBox, 0xFF0000);//draw a red box

aSp.source = aBox;
aSp.setSize(150, 200);
aSp.move(100, 100);

addChild(aSp);

function drawBox(box:MovieClip,color:uint):void {
            box.graphics.beginFill(color, 1);
            box.graphics.drawRect(0, 0, 150, 300);
            box.graphics.endFill();
}
```

5 選取「控制 > 測試影片」，執行應用程式。

使用 Slider 組件

此 Slider 組件可以讓使用者在滑動軌道（對應到值範圍）的兩端點之間滑動圖像「縮圖」，藉以選取某個值。例如，您可以使用滑動軸讓使用者選擇如數字或百分比這類的值。您也可以使用 ActionScript 讓滑動軸的值影響另一個物件的行為。例如，您可以讓滑動軸與圖片產生關聯，並根據滑動軸縮圖的相對位置 (或值) 縮放該圖片。

Slider 的目前值是由介於滑動軌道兩端點之間的縮圖相對位置，或者是 Slider 的最大值與最小值來決定。

Slider 允許在其最大值與最小值之間的連續範圍值，但是您也可以設定 snapInterval 參數，以指定最大值與最小值之間的間隔。Slider 可以沿著軌道於指定的間隔顯示刻度標記，這些標記都與滑動軸的指定值無關。

根據預設，滑動軸具有水平方向，但是您可以將 direction 參數的值設定為 vertical，為其指定垂直方向。滑動軸軌道會從一端延展至另一端，而且這些刻度標記會從左至右放置於軌道上方。

與 Slider 組件的使用者互動

當 Slider 實體成為焦點時，您可以使用下列按鍵加以控制：

按鍵	說明
向右鍵	增加水平滑動軸的相關聯值。
向上鍵	增加垂直滑動軸的相關聯值。
向左鍵	減少水平滑動軸的相關聯值。
向下鍵	減少垂直滑動軸的相關聯值。
Shift+Tab	將焦點移到上一個物件。
Tab 鍵	將焦點移到下一個物件。

如需有關控制焦點的詳細資訊，請參閱 Flash Professional 的 ActionScript 3.0 參考中的「IFocusManager 介面」和「FocusManager 類別」，以及第 24 頁「使用 FocusManager」。

進行編寫時，每個 Slider 實體的即時預覽會反映您在「屬性」檢測器或「組件檢測器」中對參數的變更。

Slider 組件參數

您可以在「屬性」檢測器或「組件檢測器」中，為每個 Slider 組件實體設定下列編寫參數：direction、liveDragging、maximum、minimum、snapInterval、tickInterval 和 value。這些參數都具有相對應的 ActionScript 同名屬性。如需有關這些參數可能值的詳細資訊，請參閱 適用於 Adobe Flash Platform 的 ActionScript 3.0 參考中的 Slider 類別。

建立具有 Slider 的應用程式

下列範例會建立 Slider 實體，讓使用者可以表達自己對於某些假設性事件的滿意程度。使用者可以將 Slider 往右移或往左移，指定較高或較低的滿意程度。

1 建立新的 Flash (ActionScript 3.0) 文件。

2 將 Label 組件從「組件」面板拖曳到「舞台」中央。

- 賦予實體名稱 **valueLabel**。
- 指定 **0percent** 做為 text 參數的值。

3 將 Slider 組件從「組件」面板拖曳到 value_lbl 下方中間處。

- 賦予實體名稱 **aSlider**。

- 將其「寬度」(W:) 指定為 **200**。
- 將其「高度」(H:) 指定為 **10**。
- 指定 **100** 做為 maximum 參數的值。
- 指定 **10** 做為 snapInterval 和 tickInterval 參數的值。

4 將另一個 Label 實體從「元件庫」面板拖曳到 aSlider 下方中間處。

- 賦予實體名稱 **promptLabel**。
- 將其「寬度」(W:) 指定為 250。
- 將其「高度」(H:) 指定為 22。
- 輸入 **Please indicate your level of satisfaction** 做為 text 參數的值。

5 開啟「動作」面板，選取主要「時間軸」中的「影格 1」，然後輸入下列 ActionScript 程式碼：

```
import fl.controls.Slider;
import fl.events.SliderEvent;
import fl.controls.Label;

aSlider.addEventListener(SliderEvent.CHANGE, changeHandler);

function changeHandler(event:SliderEvent):void {
    valueLabel.text = event.value + "percent";
}
```

6 選取「控制 > 測試影片」。

在此範例中，隨著您將滑動軸的縮圖從某個間隔移動到另一個間隔，SliderEvent.CHANGE 事件的偵聽程式會持續更新 valueLabel 的 text 屬性，以顯示對應到縮圖位置的百分比。

使用 ActionScript 建立具有 Slider 組件的應用程式

下列範例使用 ActionScript 建立 Slider。此範例會下載一朵花的影像，並使用 Slider 藉由將該影像的 alpha 屬性變更為對應到 Slider 的值，讓使用者可使影像淡出或變亮。

1 建立新的 Flash (ActionScript 3.0) 文件。

2 將 Label 組件和 Slider 組件從「組件」面板拖曳到目前文件的「元件庫」面板。

此舉會將這兩個組件加入元件庫中，但不會使其在應用程式中顯示。

3 開啟「動作」面板，選取主要「時間軸」中的「影格 1」，然後輸入下列程式碼以建立並放置組件實體：

```
import fl.controls.Slider;
import fl.events.SliderEvent;
import fl.controls.Label;
import fl.containers.UILoader;

var sliderLabel:Label = new Label();
sliderLabel.width = 120;
sliderLabel.text = "< Fade - Brighten >";
sliderLabel.move(170, 350);

var aSlider:Slider = new Slider();
aSlider.width = 200;
aSlider.snapInterval = 10;
aSlider.tickInterval = 10;
aSlider.maximum = 100;
aSlider.value = 100;
aSlider.move(120, 330);

var aLoader:UILoader = new UILoader();
aLoader.source = "http://www.flash-mx.com/images/image1.jpg";
aLoader.scaleContent = false;

addChild(sliderLabel);
addChild(aSlider);
addChild(aLoader);

aLoader.addEventListener(Event.COMPLETE, completeHandler);

function completeHandler(event:Event) {
    trace("Number of bytes loaded: " + aLoader.bytesLoaded);
}

aSlider.addEventListener(SliderEvent.CHANGE, changeHandler);

function changeHandler(event:SliderEvent):void {
        aLoader.alpha = event.value * .01;
}
```

4 選取「控制 > 測試影片」，執行應用程式。

5 將 Slider 的縮圖移到左邊可讓影像變暗，移到右邊則可讓影像變亮。

使用 **TextArea** 組件

此 TextArea 組件是原生 ActionScript TextField 物件的包裝函式。您可以使用 TextArea 組件來顯示文字，且若 editable 屬性為 true，還能用來編輯與接收文字輸入。此組件可以顯示或接收多行文字，且若 wordWrap 屬性設定為 true，還能將較長的文字行換行。restrict 屬性能讓您限制使用者可以輸入的字元，而 maxChars 則能讓您指定使用者可以輸入的字元數上限。如果文字超出文字區域的水平邊界或垂直邊界，便會自動顯示水平捲軸或垂直捲軸，除非這兩種捲軸的關聯屬性 (horizontalScrollPolicy 或 verticalScrollPolicy) 設定為 off。

如果您需要多行的文字欄位，就可以使用 TextArea 組件。例如，您可以使用 TextArea 組件做為表單中的註解欄位。您可以設定偵聽程式，在使用者以 Tab 鍵移出欄位時，檢查這個欄位是否為空白。偵聽程式可以顯示錯誤訊息，提醒必須在欄位中輸入註解。

如果您需要單行的文字欄位，請使用 TextInput 組件。

您可以使用 setStyle() 方法設定 textFormat 樣式，以變更 TextArea 實體中顯示的文字樣式。您也可以在 ActionScript 中使用 htmlText 屬性，將 TextArea 組件格式化為 HTML 格式，甚至將 displayAsPassword 屬性設定為 true，以星號來遮蔽文字。如果 condenseWhite 屬性設定為 true，Flash 便會移除新輸入文字中因空格或斷行而形成的額外空白字元。原本就已存在控制項中的文字則不受影響。

與 TextArea 組件的使用者互動

應用程式可以啟用或停用 TextArea 組件。處於停用狀態時，便不接受滑鼠或鍵盤輸入。若處於啟用狀態，則其焦點、選取範圍和導覽規則如同 ActionScript TextField 物件。當 TextArea 實體成為焦點時，您可以使用下列按鍵加以控制：

按鍵	說明
方向鍵	如果可以編輯文字，則會將插入點在文字中向上、向下、向左或向右移動。
Page Down	如果可以編輯文字，則會將插入點移到文字的結尾。
Page Up	如果可以編輯文字，則會將插入點移到文字的開頭。
Shift+Tab	將焦點移到定位鍵迴圈中的上一個物件。
Tab 鍵	將焦點移到定位鍵迴圈中的下一個物件。

如需有關控制焦點的詳細資訊，請參閱 Flash Professional 的 ActionScript 3.0 參考中的 「FocusManager 類別」，以及 第 24 頁 「使用 FocusManager」。

TextArea 組件參數

您可以在 「屬性」檢測器或 「組件檢測器」中，為每個 TextArea 組件實體設定下列編寫參數：condenseWhite、editable、hortizontalScrollPolicy、maxChars、restrict、text、verticalScrollPolicy 和 wordwrap。這些參數都具有相對應的 ActionScript 同名屬性。如需有關這些參數可能值的詳細資訊，請參閱 適用於 Adobe Flash Platform 的 ActionScript 3.0 參考中的 TextArea 類別。

進行編寫時，每個 TextArea 實體的即時預覽會反映您在 「屬性」檢測器或 「組件檢測器」面板中對參數的變更。若有需要，捲軸會出現在即時預覽中，但是在即時預覽中沒有作用。即時預覽時無法選取文字，也不能將文字輸入 「舞台」上的組件實體。

您可以撰寫 ActionScript 利用 TextArea 組件的屬性、方法和事件，來控制上列各種選項及其它選項。如需詳細資訊，請參閱 適用於 Adobe Flash Platform 的 ActionScript 3.0 參考中的 TextArea 類別。

建立具有 TextArea 組件的應用程式

下列程序說明如何在編寫時將 TextArea 組件加入應用程式。此範例會對 TextArea 實體設定 focusOut 事件處理常式，以確認使用者將焦點移至介面的其它部分之前，是否已在文字區域中輸入了一些內容。

1. 建立新的 Flash 文件 (ActionScript 3.0)。
2. 將 TextArea 組件從 「組件」面板拖曳到 「舞台」，並賦予實體名稱 **aTa**。所有參數設定均保留預設值。
3. 將第二個 TextArea 組件從 「組件」面板拖曳到 「舞台」，置於第一個 TextArea 組件下方，並賦予實體名稱 **bTa**。所有參數設定均保留預設值。
4. 開啟 「動作」面板，選取主要 「時間軸」中的 「影格 1」，然後輸入下列 ActionScript 程式碼：

```
import flash.events.FocusEvent;

aTa.restrict = "a-z,'\" \"";
aTa.addEventListener(Event.CHANGE,changeHandler);
aTa.addEventListener(FocusEvent.KEY_FOCUS_CHANGE, k_m_fHandler);
aTa.addEventListener(FocusEvent.MOUSE_FOCUS_CHANGE, k_m_fHandler);

function changeHandler(ch_evt:Event):void {
    bTa.text = aTa.text;
}
function k_m_fHandler(kmf_event:FocusEvent):void {
    kmf_event.preventDefault();
}
```

此範例會限制您只能在 aTa 文字區域中輸入小寫字元、逗號、所有格符號，以及空格。此外，也針對 aTa 文字區域的 change、KEY_FOCUS_CHANGE 和 MOUSE_FOCUS_CHANGE 事件，設定事件處理常式。changeHandler() 函數會在每次發生 change 事件時，將 aTa.text 指定給 bTa.text，而使 aTa 文字區域中輸入的文字自動顯示在 bTa 文字區域中。KEY_FOCUS_CHANGE 和 MOUSE_FOCUS_CHANGE 事件的 k_m_fHandler() 函數可以防止您在尚未輸入任何文字的情況下，就按 Tab 鍵移到下一個欄位。這是藉由防止發生預設行為而達到目的。

5 選取 「控制 > 測試影片」。

如果您在尚未輸入任何文字的情況下，就按 Tab 鍵將焦點移到第二個文字區域，則會出現錯誤訊息，且焦點將返回第一個文字區域。當您在第一個文字區域中輸入文字時，第二個文字區域會重製該文字。

使用 ActionScript 建立 TextArea 實體

下列範例使用 ActionScript 建立 TextArea 組件。其中，condenseWhite 屬性會設定為 true 以壓縮空白字元，並且將文字指定給 htmlText 屬性以利用 HTML 文字格式化特質的優點。

1 建立新的 Flash (ActionScript 3.0) 文件。

2 將 TextArea 組件拖曳到 「元件庫」面板。

3 開啟 「動作」面板，選取主要 「時間軸」中的 「影格 1」，然後輸入下列 ActionScript 程式碼：

```
import fl.controls.TextArea;

var aTa:TextArea = new TextArea();

aTa.move(100,100);
aTa.setSize(200, 200);
aTa.condenseWhite = true;
aTa.htmlText = '<b>Lorem ipsum dolor</b> sit amet, consectetuer adipiscing elit. <u>Vivamus quis nisl
vel tortor nonummy vulputate.</u> Quisque sit amet eros sed purus euismod tempor. Morbi tempor. <font
color="#FF0000">Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos
hymenaeos.</font> Curabitur diam. Suspendisse at purus in ipsum volutpat viverra. Nulla pellentesque
libero id libero.';
addChild(aTa);
```

此範例利用 htmlText 屬性，使文字區塊套用 HTML 粗體與底線特質，並將文字區塊顯示在 a_ta 文字區域中。範例中也將 condenseWhite 屬性設定為 true，以壓縮文字區塊中的空白字元。setSize() 方法設定了文字區域的高度與寬度，而 move() 方法則設定其位置。addChild() 方法會將 TextArea 實體加入至 「舞台」。

4 選取 「控制 > 測試影片」。

使用 TextInput 組件

此 TextInput 組件是原生 ActionScript TextField 物件的包裝函式，屬於單行文字組件。如果您需要多行的文字欄位，請使用 TextArea 組件。例如，您可以使用 TextInput 組件做為表單中的密碼欄位。您也可以設定偵聽程式，在使用者以 Tab 鍵移出欄位時，檢查欄位內是否有足夠的字元。偵聽程式可以顯示錯誤訊息，提醒必須至少輸入幾個字元。

您可以使用 setStyle() 方法設定 textFormat 屬性，以變更 TextInput 實體中顯示的文字樣式。TextInput 組件也可以格式化為 HTML 格式，或做為掩飾文字的密碼欄位。

與 TextInput 組件的使用者互動

應用程式可以啟用或停用 TextInput 組件。處於停用狀態時，便不接受滑鼠或鍵盤輸入。若處於啟用狀態，則其焦點、選取範圍和導覽規則如同 ActionScript TextField 物件。當 TextInput 實體成為焦點時，您也可以使用下列按鍵加以控制：

按鍵	說明
方向鍵	將插入點向左或向右移動一個字元。
Shift+Tab	將焦點移到上一個物件。
Tab 鍵	將焦點移到下一個物件。

如需有關控制焦點的詳細資訊，請參閱 Flash Professional 的 ActionScript 3.0 參考中的 「FocusManager 類別」，以及 第 24 頁 「使用 FocusManager」。

進行編寫時，每個 TextInput 實體的即時預覽會反映您在 「屬性」檢測器或 「組件檢測器」中對參數的變更。即時預覽時無法選取文字，也不能將文字輸入 「舞台」上的組件實體。

將 TextInput 組件加入應用程式時，您可以利用 「輔助功能」面板讓螢幕朗讀程式能夠存取此組件。

TextInput 組件參數

您可以在 「屬性」檢測器或 「組件檢測器」中，為每個 TextInput 組件實體設定下列編寫參數：editable、displayAsPassword、maxChars、restrict 和 text。這些參數都具有相對應的 ActionScript 同名屬性。如需有關這些參數可能值的詳細資訊，請參閱 適用於 Adobe Flash Platform 的 ActionScript 3.0 參考中的 TextInput 類別。

您可以撰寫 ActionScript 利用 TextInput 組件的屬性、方法和事件，來控制其各種選項及其它選項。如需詳細資訊，請參閱 適用於 Adobe Flash Platform 的 ActionScript 3.0 參考中的 TextInput 類別。

建立具有 TextInput 組件的應用程式

下列程序說明如何將 TextInput 組件加入應用程式。此範例使用兩個 TextInput 欄位來接收及確認密碼。事件偵聽程式將用於檢查是否已經輸入至少八個字元的密碼，以及兩個欄位中輸入的文字是否相符。

1 建立新的 Flash (ActionScript 3.0) 文件。

2 將 Label 組件從 「組件」面板拖曳到 「舞台」，然後在 「屬性」檢測器中為組件指定下列值：

- 輸入實體名稱 **pwdLabel**。
- 輸入 **100** 做為 W 值。
- 輸入 **50** 做為 X 值。
- 輸入 **150** 做為 Y 值。
- 在 「參數」區段，輸入 **Password:** 做為 text 參數的值。

3 將第二個 Label 組件從 「組件」面板拖曳到 「舞台」，然後為組件指定下列值：

- 輸入實體名稱 **confirmLabel**。
- 輸入 **100** 做為 W 值。
- 輸入 **50** 做為 X 值。
- 輸入 **200** 做為 Y 值。
- 在 「參數」區段，輸入 **ConfirmPassword:** 做為 text 參數的值。

4 將 TextInput 組件從 「組件」面板拖曳到 「舞台」，然後為組件指定下列值：

- 輸入實體名稱 **pwdTi**。
- 輸入 **150** 做為 W 值。
- 輸入 **190** 做為 X 值。
- 輸入 **150** 做為 Y 值。
- 在 「參數」區段，按兩下 displayAsPassword 參數的值，並選取 **true**。這麼做會使得文字欄位中輸入的值以星號遮蔽。

5 將第二個 TextInput 組件從 「組件」面板拖曳到 「舞台」，然後為組件指定下列值：

- 輸入實體名稱 **confirmTi**。
- 輸入 **150** 做為 W 值。
- 輸入 **190** 做為 X 值。
- 輸入 **200** 做為 Y 值。
- 在 「參數」區段，按兩下 displayAsPassword 參數的值，並選取 **true**。這麼做會使得文字欄位中輸入的值以星號遮蔽。

6 開啟 「動作」面板，選取主要 「時間軸」中的 「影格 1」，然後輸入下列 ActionScript 程式碼：

```
function tiListener(evt_obj:Event){
if(confirmTi.text != pwdTi.text || confirmTi.length < 8)
{
        trace("Password is incorrect. Please reenter it.");
}
else {
        trace("Your password is: " + confirmTi.text);
}
}
confirmTi.addEventListener("enter", tiListener);
```

此程式碼會對 TextInput 實體 (名為 confirmTi) 設定 enter 事件處理常式。如果兩組密碼不相符，或是使用者輸入少於八個字元，範例就會顯示 「Password is incorrect.Please reenter it.」訊息。如果兩組密碼相符且至少都是八個字元，範例則在 「輸出」面板中顯示輸入的值。

7 選取 「控制 > 測試影片」。

使用 ActionScript 建立 TextInput 實體

下列範例使用 ActionScript 建立 TextInput 組件。此範例也會建立 Label，以提示使用者輸入自己的名稱。組件的 restrict 屬性將設定為只允許大小寫字母、句號和空格。另外還會建立 TextFormat 物件，用來格式化 Label 和 TextInput 組件中的文字。

1 建立新的 Flash (ActionScript 3.0) 文件。

2 將 TextInput 組件從 「組件」面板拖曳到 「元件庫」面板。

3 將 Label 組件從 「組件」面板拖曳到 「元件庫」面板。

4 開啟 「動作」面板，選取主要 「時間軸」中的 「影格 1」，然後輸入下列 ActionScript 程式碼：

```
import fl.controls.Label;
import fl.controls.TextInput;

var nameLabel:Label = new Label();
var nameTi:TextInput = new TextInput();
var tf:TextFormat = new TextFormat();

addChild(nameLabel);
addChild(nameTi);

nameTi.restrict = "A-Z .a-z";

tf.font = "Georgia";
tf.color = 0x0000CC;
tf.size = 16;

nameLabel.text = "Name: " ;
nameLabel.setSize(50, 25);
nameLabel.move(100,100);
nameLabel.setStyle("textFormat", tf);
nameTi.move(160, 100);
nameTi.setSize(200, 25);
nameTi.setStyle("textFormat", tf);
```

5 選取「控制 > 測試影片」，執行應用程式。

使用 TileList 組件

此 TileList 組件是由欄與列組成的清單，其中包含資料提供者所提供的資料。儲存在 TileList 儲存格中的資料單位稱為一個「項目」。一般而言，項目 (源自資料提供者) 具有 label 屬性和 source 屬性。label 屬性識別儲存格所要顯示的內容，而 source 則是為項目提供值。

您可以建立 Array 實體，或是從伺服器擷取該實體。TileList 組件具有一組可代理資料提供者執行操作的方法，例如 addItem() 和 removeItem() 方法。如果清單並未由外部資料提供者提供資料，這些方法會自動建立一個資料提供者實體 (透過 List.dataProvider 顯露)。

與 TileList 組件的使用者互動

TileList 將會使用實作 ICellRenderer 介面的 Sprite 來呈現每一個儲存格。您可以使用 TileList cellRenderer 屬性，指定這個輸出器。TileList 組件的預設 CellRenderer 是 ImageCell (顯示的影像來自類別、點陣圖、實體或 URL) 和一個選擇性的標籤。該標籤會以單行形式固定對齊儲存格底端。您只能朝著單一方向捲動 TileList。

當 TileList 實體成為焦點時，您也可以使用下列按鍵控制其中的項目：

按鍵	說明
向上鍵和向下鍵	允許您在一欄內向上及向下移動。如果 allowMultipleSelection 屬性為 true，您便可以搭配 Shift 鍵來使用這些按鍵，以選取多個儲存格。
向左鍵和向右鍵	允許您在一列中向左或向右移動。如果 allowMultipleSelection 屬性為 true，您便可以搭配 Shift 鍵來使用這些按鍵，以選取多個儲存格。

按鍵	說明
首頁	選取 TileList 中的第一個儲存格。如果 allowMultipleSelection 屬性為 true，按住 Shift 鍵再按 Home 鍵將會選取介於目前選取範圍到第一個儲存格之間的所有儲存格。
End	選取 TileList 中的最後一個儲存格。如果 allowMultipleSelection 屬性為 true，按住 Shift 鍵再按 End 鍵將會選取介於目前選取範圍到最後一個儲存格之間的所有儲存格。
Ctrl	如果 allowMultipleSelection 屬性為 true，您便可以不依特定順序選取多個儲存格。

將 TileList 組件加入應用程式時，您可以加入下列 ActionScript 程式碼，讓螢幕朗讀程式能夠存取此組件：

```
import fl.accessibility.TileListAccImpl;

TileListAccImpl.enableAccessibility();
```

不論組件有多少個實體，您只需要啟用組件的輔助功能一次。如需詳細資訊，請參閱 「使用 Flash」中的第 18 章 「建立輔助功能內容」。

TileList 組件參數

您可以在 「屬性」檢測器或 「組件檢測器」中，為每個 TileList 組件實體設定下列編寫參數：allowMultipleSelection、columnCount、columnWidth、dataProvider、direction、horizontalScrollLineSize、horizontalScrollPageSize、labels、rowCount、rowHeight、ScrollPolicy、verticalScrollLineSize 和 verticalScrollPageSize。這些參數都具有相對應的 ActionScript 同名屬性。如需有關 dataProvider 參數用法的詳細資訊，請參閱第 25 頁 「使用 dataProvider 參數」。

您可以撰寫 ActionScript，使用 TileList 實體的屬性、方法和事件，以設定 TileList 實體的其它選項。如需詳細資訊，請參閱適用於 Adobe Flash Platform 的 ActionScript 3.0 參考中的 TileList 類別。

建立具有 TileList 組件的應用程式

此範例使用 MovieClip 繪製顏色的陣列，來為 TileList 進行填色。

1 建立新的 Flash (ActionScript 3.0) 文件。

2 將 TileList 組件拖曳到 「舞台」，並且賦予實體名稱 **aTl**。

3 開啟 「動作」面板，選取主要 「時間軸」中的 「影格 1」，然後輸入下列 ActionScript 程式碼：

```
import fl.data.DataProvider;
import flash.display.DisplayObject;

var aBoxes:Array = new Array();
var i:uint = 0;
var colors:Array = new Array(0x00000, 0xFF0000, 0x0000CC, 0x00CC00, 0xFFFF00);
var colorNames:Array = new Array("Midnight", "Cranberry", "Sky", "Forest", "July");
var dp:DataProvider = new DataProvider();
for(i=0; i < colors.length; i++) {
    aBoxes[i] = new MovieClip();
    drawBox(aBoxes[i], colors[i]);// draw box w next color in array
    dp.addItem( {label:colorNames[i], source:aBoxes[i]} );
}
aTl.dataProvider = dp;
aTl.columnWidth = 110;
aTl.rowHeight = 130;
aTl.setSize(280,150);
aTl.move(150, 150);
aTl.setStyle("contentPadding", 5);

function drawBox(box:MovieClip,color:uint):void {
            box.graphics.beginFill(color, 1.0);
            box.graphics.drawRect(0, 0, 100, 100);
            box.graphics.endFill();
}
```

4 選取 「控制 > 測試影片」，測試應用程式。

使用 ActionScript 建立 TileList 組件

此範例會以動態方式建立 TileList 實體，並將 ColorPicker、ComboBox、NumericStepper 以及 CheckBox 組件的實體加入其中。接著建立 Array 以包含所要顯示之各組件的標籤及名稱，並將該 Array (dp) 指定給 TileList 的 dataProvider 屬性。範例中使用 columnWidth 和 rowHeight 屬性以及 setSize() 方法來配置 TileList、使用 move() 方法將 TileList 放置於 「舞台」、使用 contentPadding 樣式在 TileList 實體的邊界與內容之間騰出空間，並且使用 sortItemsOn() 方法依照 TileList 的標籤排序內容。

1 建立新的 Flash (ActionScript 3.0) 文件。

2 將下列組件從 「組件」面板拖曳到 「元件庫」面板：ColorPicker、ComboBox、NumericStepper、CheckBox 和 TileList。

3 開啟 「動作」面板，選取主要 「時間軸」中的 「影格 1」，然後輸入下列 ActionScript 程式碼：

```
import fl.controls.CheckBox;
import fl.controls.ColorPicker;
import fl.controls.ComboBox;
import fl.controls.NumericStepper;
import fl.controls.TileList;
import fl.data.DataProvider;

var aCp:ColorPicker = new ColorPicker();
var aCb:ComboBox = new ComboBox();
var aNs:NumericStepper = new NumericStepper();
var aCh:CheckBox = new CheckBox();
var aTl:TileList = new TileList();

var dp:Array = [
{label:"ColorPicker", source:aCp},
{label:"ComboBox", source:aCb},
{label:"NumericStepper", source:aNs},
{label:"CheckBox", source:aCh},
];
aTl.dataProvider = new DataProvider(dp);
aTl.columnWidth = 110;
aTl.rowHeight = 100;
aTl.setSize(280,130);
aTl.move(150, 150);
aTl.setStyle("contentPadding", 5);
aTl.sortItemsOn("label");
addChild(aTl);
```

4 選取「控制 > 測試影片」，測試應用程式。

使用 UILoader 組件

此 UILoader 組件是一個容器，可以顯示 SWF、JPEG、漸進式 JPEG、PNG 和 GIF 檔。如果您需要從遠端位置擷取內容再放入 Flash 應用程式中，就可以使用 UILoader。例如，您可以使用 UILoader 將公司的標誌 (JPEG 檔) 加入表單中。您也可以在顯示相片的應用程式中使用 UILoader 組件。使用 load() 方法可載入內容、percentLoaded 屬性可判斷已載入多少內容、complete 事件可判斷載入作業何時完成。

您可以縮放 UILoader 的內容，或者調整 UILoader 本身的大小來配合內容的大小。在預設狀況下，內容會進行縮放來配合 UILoader。您也可以在執行階段載入內容，並且監視載入進度 (只不過既然已經快取內容，載入進度便會很快就跳至 100%)。如果您在載入內容至 UILoader 時指定位置，則必須將該位置 (X 和 Y 座標) 指定為 0, 0。

與 UILoader 組件的使用者互動

UILoader 組件不能成為焦點。但是，載入至 UILoader 組件中的內容可以成為焦點，並且可以有自己的焦點互動。如需有關控制焦點的詳細資訊，請參閱 Flash Professional 的 ActionScript 3.0 參考中的「FocusManager 類別」，以及第 24 頁「使用 FocusManager」。

UILoader 組件參數

您可以在 「屬性」檢測器或 「組件檢測器」中，為每個 UILoader 組件實體設定下列編寫參數：autoLoad、maintainAspectRatio、source 和 scaleContent。這些參數都具有相對應的 ActionScript 同名屬性。

進行編寫時，每個 UILoader 實體的即時預覽會反映您在 「屬性」檢測器或 「組件檢測器」中對參數的變更。

您可以撰寫 ActionScript，使用 UILoader 實體的屬性、方法和事件，以設定 UILoader 實體的其它選項。如需詳細資訊，請參閱 適用於 Adobe Flash Platform 的 ActionScript 3.0 參考中的 UILoader 類別。

建立具有 UILoader 組件的應用程式

下列程序說明如何在編寫時將 UILoader 組件加入應用程式。在此範例中，載入器會載入某標誌的 GIF 影像。

1 建立新的 Flash (ActionScript 3.0) 文件。

2 將 UILoader 組件從 「組件」面板拖曳到 「舞台」。

3 在 「屬性」檢測器中，輸入實體名稱 **aUI**。

4 選取 「舞台」上的載入器，然後在 「組件檢測器」中輸入 **http://www.helpexamples.com/images/logo.gif** 做為 source 參數的值。

使用 ActionScript 建立 UILoader 組件實體

此範例使用 ActionScript 建立 UILoader 組件，並載入 JPEG 花朵影像。發生 complete 事件時，「輸出」面板將會顯示已載入的位元組數。

1 建立新的 Flash (ActionScript 3.0) 文件。

2 將 UILoader 組件從 「組件」面板拖曳到 「元件庫」面板。

3 開啟 「動作」面板，選取主要 「時間軸」中的 「影格 1」，然後輸入下列 ActionScript 程式碼：

```
import fl.containers.UILoader;

var aLoader:UILoader = new UILoader();
aLoader.source = "http://www.flash-mx.com/images/image1.jpg";
aLoader.scaleContent = false;
addChild(aLoader);

aLoader.addEventListener(Event.COMPLETE, completeHandler);
function completeHandler(event:Event) {
    trace("Number of bytes loaded: " + aLoader.bytesLoaded);
}
```

4 選取 「控制 > 測試影片」。

使用 UIScrollBar 組件

此 UIScrollBar 元件讓您可以在文字欄位增加捲軸。您可以在編寫期間為文字欄位加上捲軸，或在執行階段使用 ActionScript 進行。若要使用 UIScrollBar 組件，請在 「舞台」上建立文字欄位，並將 UIScrollBar 組件從 「組件」面板拖曳到文字欄位範圍框的任何區域。

如果捲軸長度小於捲動箭頭的合併大小，則無法正確顯示捲軸。其中一個箭頭按鈕會變成隱藏在另一按鈕之後。Flash 不檢查這方面的錯誤。此時最好使用 ActionScript 隱藏捲軸。如果捲軸太小以致放不下捲軸方塊 (縮圖)，Flash 會隱藏捲軸方塊。

UIScrollBar 組件的功能如同其它捲軸。兩端各有一個箭頭按鈕，其間又有捲動軌道和捲軸方塊 (縮圖)。此組件可附加到文字欄位的任一邊，以垂直或水平方向使用。

如需有關 TextField 的詳細資訊，請參閱 適用於 Adobe Flash Platform 的 ActionScript 3.0 參考中的 TextField 類別。

與 UIScrollBar 組件的使用者互動

UIScrollBar 不同於其它多數組件，此組件可接受連續的滑鼠輸入，例如使用者按住滑鼠按鈕而不需要重複點選。

UIScrollBar 組件與鍵盤之間沒有互動。

UIScrollBar 組件參數

您可以在 「屬性」檢測器或 「組件檢測器」中，為每個 UIScrollBar 組件實體設定下列編寫參數：direction 和 scrollTargetName。這些參數都具有相對應的 ActionScript 同名屬性。

您可以撰寫 ActionScript，使用 UIScrollBar 實體的類別方法、屬性和事件，以設定 UIScrollBar 實體的其它選項。如需詳細資訊，請參閱 適用於 Adobe Flash Platform 的 ActionScript 3.0 參考中的 UIScrollBar 類別。

建立具有 UIScrollBar 組件的應用程式

下列程序說明如何在編寫時將 UIScrollBar 組件加入應用程式。

1 建立新的 Flash (ActionScript 3.0) 文件。

2 建立動態文字欄位，其高度必須要能容納一行或兩行文字，然後在 「屬性」檢測器中賦予實體名稱 **myText**。

3 在 「屬性」檢測器中，將文字輸入欄位的 「字行類型」設定為 「多行」；若要想使用水平捲軸，請設定為 「多行不換行」。

4 開啟 「動作」面板，在主要 「時間軸」中選取 「影格 1」，然後輸入下列 ActionScript 程式碼以填入 text 屬性，讓使用者必須捲動才能檢視全部的文字：

```
myText.text="When the moon is in the seventh house and Jupiter aligns with Mars, then peace will guide the planet and love will rule the stars."
```

備註：請確定 「舞台」上的文字欄位小到需要捲動才看得到所有文字。否則，捲軸就不會出現，或可能只出現兩行但沒有縮圖底框 (拖曳以捲動內容的部分)。

5 確認已開啟物件貼齊功能 (「檢視 > 貼齊 > 貼齊物件」)。

6 將 UIScrollBar 實體從 「組件」面板拖曳到文字輸入欄位附近要附加捲軸的位置。等到組件與文字欄位重疊時才放開滑鼠，讓組件適當地繫結至欄位。賦予組件實體名稱 **mySb**。

組件的 scrollTargetName 屬性會自動填入，其值等於文字欄位在 「屬性」檢測器和 「組件檢測器」中的實體名稱。如果此名稱未出現在 「參數」索引標籤中，表示 UIScrollBar 實體可能重疊得不夠多。

7 選取 「控制 > 測試影片」。

使用 ActionScript 建立 UIScrollBar 組件實體

您可以使用 ActionScript 建立 UIScrollBar 實體，讓捲軸與文字欄位在執行階段產生關聯。下列範例會建立水平方向的 UIScrollBar 實體，並且將它附加到名為 **myTxt** 的文字欄位實體下方，而該文字欄位實體則是從 URL 載入文字。此範例也設定捲軸的大小，以符合文字欄位的大小：

1 建立新的 Flash (ActionScript 3.0) 文件。

2 將 ScrollBar 組件拖曳到 「元件庫」面板。

3 開啟 「動作」面板，選取主要 「時間軸」中的 「影格 1」，然後輸入下列 ActionScript 程式碼：

```
import flash.net.URLLoader;
import fl.controls.UIScrollBar;
import flash.events.Event;

var myTxt:TextField = new TextField();
myTxt.border = true;
myTxt.width = 200;
myTxt.height = 16;
myTxt.x = 200;
myTxt.y = 150;

var mySb:UIScrollBar = new UIScrollBar();
mySb.direction = "horizontal";
// Size it to match the text field.
mySb.setSize(myTxt.width, myTxt.height);

// Move it immediately below the text field.
mySb.move(myTxt.x, myTxt.height + myTxt.y);

// put them on the Stage
addChild(myTxt);
addChild(mySb);
// load text
var loader:URLLoader = new URLLoader();
var request:URLRequest = new URLRequest("http://www.helpexamples.com/flash/lorem.txt");
loader.load(request);
loader.addEventListener(Event.COMPLETE, loadcomplete);

function loadcomplete(event:Event) {
    // move loaded text to text field
    myTxt.text = loader.data;
    // Set myTxt as target for scroll bar.
    mySb.scrollTarget = myTxt;
}
```

4 選取「控制 > 測試影片」。

第 5 章 自訂使用者介面組件

關於自訂 UI 組件

您可以藉由修改下列一個或兩個元素，自訂應用程式中組件的外觀：

樣式　每一個組件都具有一組樣式，您可以設定這些樣式，指定 Flash 用來呈現組件外觀的值。一般而言，樣式會指定要在組件的不同狀態中，用於該組件的外觀元素和圖示，並且也會指定要使用的文字格式與邊框間距值。

外觀元素　「外觀元素」是由一組元件集合所組成，這些元件則構成指定狀態中的組件圖像外觀。樣式會指定所要使用的外觀元素，而所謂外觀元素就是 Flash 用來繪製組件的圖像元素。「外觀設定」是修改或取代組件的來源圖像，以變更其外觀的程序。

備註：您可以將 ActionScript 3.0 組件的預設外觀視為主題 (Aeon Halo)，但是這些外觀元素是內建在這些組件中。ActionScript 3.0 組件不支援 ActionScript 2.0 組件支援的外部主題檔案。

設定樣式

一般而言，當 Flash 將組件繪製為各種狀態時，該組件的樣式會指定其外觀元素、圖示、文字格式以及邊框間距的值。例如，Flash 會以不同的外觀元素繪製 Button，以顯示它的滑入狀態 (當您以滑鼠按鈕按一下此 Button 時會發生此狀態) 有別於一般及正常狀態。當它處於停用狀態 (將 enabled 屬性設定為 false) 時，也會使用不同的外觀元素。

您可以在文件、類別與實體層級設定組件的樣式。此外，某些樣式屬性也可以繼承自父輩組件。例如，List 組件會藉由繼承自 BaseScrollPane 來繼承 ScrollBar 樣式。

您可以使用下列方式，設定樣式自訂組件：

- 設定組件實體上的樣式。您可以變更單一組件實體的顏色和文字屬性。這種方式在某些狀況下很有效，但是如果必須設定文件中所有組件的各個屬性，則可能很花時間。
- 您可以設定文件中指定類型之所有組件的樣式。如果您要針對指定之樣式的所有組件 (例如，針對文件中的所有 CheckBox 或 Button) 套用一致的外觀，則可以在組件層級設定樣式。

  對容器設定的樣式屬性值，都會由其中含有的組件所繼承。

  當您使用 「即時預覽」功能檢視 「舞台」上的組件時，Flash 不會顯示對於樣式屬性所作的變更。

瞭解樣式設定

這裡列出幾項關於使用樣式的重點：

繼承　依據設計，子組件已設定成要繼承父組件的樣式。您無法在 ActionScript 內設定樣式的繼承。

優先順序　如果組件樣式是以一種以上的方式設定，Flash 就會根據樣式的優先順序，使用它所遇到的第一個樣式。Flash 會以下列順序尋找樣式，直到發現其值為止：

1. Flash 會在組件實體上尋找樣式屬性。
2. 如果樣式是繼承樣式之一，Flash 會查看整個父階層以找出繼承值。
3. Flash 會尋找組件上的樣式。
4. Flash 會尋找 StyleManager 上的全域設定。
5. 如果屬性尚未定義，屬性就會具有 undefined 的值。

存取組件的預設樣式

您可以使用組件類別的靜態 getStyleDefinition() 方法，存取該組件的預設樣式。例如，下列程式碼會擷取 ComboBox 組件的預設樣式，並顯示 buttonWidth 和 downArrowDownSkin 屬性的預設值：

```
import fl.controls.ComboBox;
var styleObj:Object = ComboBox.getStyleDefinition();
trace(styleObj.buttonWidth); // 24
trace(styleObj.downArrowDownSkin); // ScrollArrowDown_downSkin
```

設定並取得組件實體上的樣式

任何 UI 組件實體都可以直接呼叫 setStyle() 和 getStyle() 方法，以設定或擷取樣式。下列語法會設定組件實體的樣式和值：

```
instanceName.setStyle("styleName", value);
```

此語法會擷取組件實體的樣式：

```
var a_style:Object = new Object();
a_style = instanceName.getStyle("styleName");
```

請注意，getStyle() 方法會傳回 Object 類型，因為此方法可以傳回具有不同資料類型的多種樣式。例如，下列程式碼會設定 TextArea 實體 (aTa) 的字體樣式，並使用 getStyle() 方法擷取該樣式。下列範例會將傳回值轉換為 TextFormat 物件，以便將它指定給 TextFormat 變數。如果沒有先進行轉換，編譯器就會發出嘗試強制將 Object 變數轉換為 TextFormat 變數的錯誤。

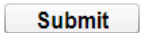
```
import flash.text.TextFormat;

var tf:TextFormat = new TextFormat();
tf.font = "Georgia";
aTa.setStyle("textFormat",tf);
aTa.text = "Hello World!";
var aStyle:TextFormat = aTa.getStyle("textFormat") as TextFormat;
trace(aStyle.font);
```

使用 TextFormat 設定文字屬性

您可以使用 TextFormat 物件，格式化組件實體的文字。TextFormat 物件所擁有的屬性可讓您指定 bold、bullet、color、font、italic、size 等文字特性，以及數種其他特性。您可以在 TextFormat 物件中設定這些屬性，然後再呼叫 setStyle() 方法，將這些屬性套用到組件實體。例如，下列程式碼會設定 TextFormat 物件的 font、size 和 bold 屬性，並將這些屬性套用到 Button 實體：

```
/* Create a new TextFormat object to set text formatting properties. */
var tf:TextFormat = new TextFormat();
tf.font = "Arial";
tf.size = 16;
tf.bold = true;
a_button.setStyle("textFormat", tf);
```

下圖說明這些設定對於具有 「Submit」標籤之按鈕的影響：



透過 setStyle() 對組件實體設定的樣式屬性擁有最高的優先順序，並且會覆寫其它所有樣式設定。但是，在單一組件實體中使用 setStyle() 設定的屬性越多，組件在執行階段呈現的速度就會越慢。

為組件的所有實體設定一種樣式

您可以使用 StyleManager 類別的靜態 setComponentStyle() 方法，為組件類別的所有實體設定一種樣式。例如，首先您可以將 Button 拖曳到「舞台」，再於 「時間軸」之「影格 1」的「動作」面板中，加入下列 ActionScript 程式碼，將所有 Button 的文字顏色設定為紅色：

```
import fl.managers.StyleManager;
import fl.controls.Button;

var tf:TextFormat = new TextFormat();
tf.color = 0xFF0000;
StyleManager.setComponentStyle(Button, "textFormat", tf);
```

您後續加入「舞台」的所有 Button 都會具有紅色的標籤。

為所有組件設定一種樣式

您可以使用 StyleManager 類別的靜態 setStyle() 方法，為所有組件設定一種樣式。

1. 將 List 組件拖曳到「舞台」，並為它指定實體名稱為 **aList**。
2. 將 Button 組件拖曳到「舞台」，並為它指定實體名稱為 **aButton**。
3. 如果「動作」面板還沒有開啟，請按下 **F9** 或是從「視窗」選單選取「動作」加以開啟。接著，再於「時間軸」的「影格 1」中輸入下列程式碼，將所有組件的文字顏色設定為紅色。

```
import fl.managers.StyleManager;

var tf:TextFormat = new TextFormat();
tf.color = 0xFF0000;
StyleManager.setStyle("textFormat", tf);
```

4. 在「動作」面板中加入下列程式碼，將文字填入 List 中。

```
aList.addItem({label:"1956 Chevy (Cherry Red)", data:35000});
aList.addItem({label:"1966 Mustang (Classic)", data:27000});
aList.addItem({label:"1976 Volvo (Xcllnt Cond)", data:17000});
aList.allowMultipleSelection = true;
```

5. 選取「控制 > 測試影片」或按 Control+Enter 鍵，編譯程式碼並測試您的內容。在按鈕標籤與清單中的文字應該都是紅色的。

關於外觀元素

組件的外觀是由如外框、填色顏色、圖示，以及甚至是其它組件這類圖像元素組成。例如，ComboBox 就包含 List 組件，而 List 組件則包含 ScrollBar。這些圖像元素組合在一起便構成 ComboBox 的外觀。不過，組件的外觀會根據它目前的狀態而變更。例如，沒有其標籤的 CheckBox 在應用程式中看起來如下：

處於正常一般狀態的 CheckBox

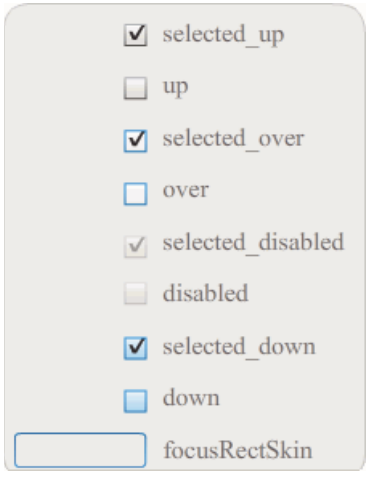
如果您在此 CheckBox 上按一下並按住滑鼠按鈕，它的外觀就會變更如下：

處於按下狀態的 CheckBox

此外，當您放開滑鼠按鈕時，CheckBox 又會回復成原來的外觀，但是會多了核取記號，表示已經選取的狀態。

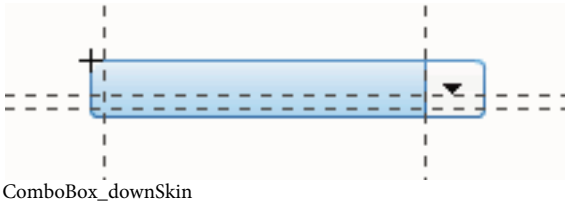

處於選取狀態的 CheckBox

代表組件不同狀態的圖示統稱為該組件的 「外觀元素」。就像處理其它 Flash 元件一樣，您可以在 Flash 中編輯某個組件的外觀元素，藉以變更該組件在任何或所有狀態中的外觀。您可以使用兩種方式存取組件的外觀元素。最簡單的方式就是將該組件拖曳到 「舞台」，然後按兩下該組件。這麼做會開啟該組件之外觀元素的面板，以 CheckBox 為例，它看起來如下：



CheckBox 的外觀元素

您也可以從 「元件庫」面板，個別存取組件的外觀元素。將組件拖曳到 「舞台」時，您也會將該組件連同其資源的資料夾以及組件中所包含的其它組件一併複製到元件庫。例如，如果您將 ComboBox 拖曳到 「舞台」上，「元件庫」面板還會包含 List、ScrollBar 和 TextInput 組件 (都內建在 ComboBox 中)、其中每一個組件之外觀元素的資料夾，以及內含這些組件所共享之元素的 Shared Assets 資料夾。您可以編輯上述組件中任一個組件的外觀元素，只要開啟該組件的外觀元素資料夾 (ComboBoxSkins、ListSkins、ScrollBarSkins 或 TextInputSkins)，然後按兩下您要編輯之外觀元素的圖示即可。例如，按兩下 ComboBox_downSkin 會以元件編輯模式開啟該外觀元素，如下圖所示：


ComboBox_downSkin

建立新的外觀元素

如果您要為文件中的組件建立新的外觀，請編輯該組件的外觀元素來改變這些外觀元素。若要存取組件的外觀元素，只要按兩下 「舞台」上的組件以開啟該組件之外觀元素的面板即可。接著，再按兩下您想要編輯的外觀元素，以元件編輯模式加以開啟。例如，您可以按兩下 「舞台」上的 TextArea 組件，以元件編輯模式開啟該組件的資源。將縮放控制項設定為 400% (如有必要，也可以設為更高)，然後再編輯元件以變更其外觀。編輯完成時，所做的變更會影響文件中該組件的所有實體。另一種方式是，您可以按兩下 「元件庫」面板中的特定外觀元素，以元件編輯模式在 「舞台」上開啟該外觀元素。

您可以使用下列方式修改組件外觀元素：

- 為所有實體建立一個新外觀元素

- 為某些實體建立新的外觀元素

為所有實體建立一個外觀元素

根據預設，當您編輯某個組件的外觀元素時，會變更文件中該組件之所有實體的外觀。如果您要為相同的組件建立不同的外觀，則必須重製所要變更的外觀元素，並且為它們使用不同的名稱、編輯它們，再設定適當的樣式來套用它們。如需詳細資訊，請參閱第 91 頁 「為某些實體建立外觀元素」。

本章說明如何改變每一個 UI 組件的一個或多個外觀元素。如果您依照這些程序之一來變更 UI 組件的一個或多個外觀元素，便會變更文件中該組件的所有實體。

為某些實體建立外觀元素

您可以使用下列一般程序，為組件的某些實體建立一個外觀元素：

- 在 「元件庫」面板中，選取該組件之 Assets 資料夾內的外觀元素。
- 重製該外觀元素，並為它指定唯一的類別名稱。
- 編輯該外觀元素，並為它指定您想要的外觀。
- 為該組件實體呼叫 setStyle() 方法，將新的外觀元素指定給該外觀元素樣式。

下列程序會為兩個 Button 實體的其中一個建立新的 selectedDownSkin。

1. 建立新的 Flash 檔案 (ActionScript 3.0) 文件。
2. 將兩個 Button 從 「組件」面板拖曳到 「舞台」，並為它們指定實體名稱為 **aButton** 與 **bButton**。
3. 開啟 「元件庫」面板，然後再開啟其中的 Component Assets 和 ButtonSkins 資料夾。
4. 按一下以選取 selectedDownSkin 外觀元素。
5. 按一下右鍵開啟快顯選單，然後選取 「重製」。
6. 在 「重製元件」對話方塊中，為新的外觀元素指定唯一名稱 (例如 **Button_mySelectedDownSkin**)，然後按一下 「確定」。
7. 在 「元件庫 > Component Assets > ButtonSkins」資料夾中，選取 Button_mySelectedDownSkin，再按一下右鍵開啟快顯選單。選取 「連結」來開啟 「連結屬性」對話方塊。
8. 按一下 「匯出給 ActionScript 使用」核取方塊。讓 「匯出在第一個影格」核取方塊保持為已選取狀態，並確認類別名稱為唯一。按一下 「確定」，再按一下 「確定」回應警告：警告內容說明找不到類別定義，而且會建立類別定義。
9. 按兩下 「元件庫」面板中的 Button_mySelectedDownSkin 外觀元素，以元件編輯模式加以開啟。
10. 按一下外觀元素中心處的藍色填色，直到該顏色出現在 「屬性」檢測器的 「填色」顏色挑選器中為止。按一下顏色挑選器，然後選取 #00CC00 做為外觀元素的填色顏色。
11. 按一下 「舞台」上方編輯列左側的 「後退」按鈕，回到文件編輯模式。
12. 在 「屬性」檢測器中，按一下每一個按鈕的 「參數」索引標籤，然後將 toggle 參數設定為 true。
13. 在 「時間軸」之 「影格 1」的 「動作」面板中，加入下列程式碼：

```
bButton.setStyle("selectedDownSkin", Button_mySelectedDownSkin);
bButton.setStyle("downSkin", Button_mySelectedDownSkin);
```

14. 選取 「控制 > 測試影片」。
15. 按一下每一個按鈕。請注意，bButton 物件的按下外觀元素 (選取及未選取) 會使用新的外觀元素元件。

自訂 Button 組件

您可以在編寫期間和執行階段，沿水平和垂直方向變形 Button 組件。在編寫期間，請在「舞台」上選取組件，並且使用「自由變形」工具或任何「修改 > 變形」命令。在執行階段，請使用 setSize() 方法或 Button 類別適用的任何屬性 (如 height、width、scaleX 和 scaleY)。

調整按鈕大小不會變更圖示和標籤的大小。Button 的範圍框與此 Button 的邊框相對應，並且也會指定實體的作用區域。如果您增加實體的大小，也會同時增加作用區域的大小。如果範圍框太小無法配合標籤，就會裁剪標籤以便配合。

如果 Button 的圖示比 Button 大，此圖示便會延伸超出 Button 的邊框。

搭配 Button 組件使用樣式

一般而言，將 Button 組件繪製為各種狀態時，其樣式會指定其外觀元素、圖示、文字格式以及邊框間距的值。

下列程序會將兩個 Button 置於「舞台」上，並且在使用者按一下其中一個時，將這兩個 Button 的 emphasized 屬性設定為 true。當使用者按一下第二個 Button 時，它也會將其 emphasizedSkin 樣式設定為 selectedOverSkin 樣式，如此這兩個 Button 在相同的狀態就會顯示不同的外觀元素。

1 建立 Flash 檔案 (ActionScript 3.0)。

2 以一次一個的方式，將兩個 Button 拖曳到「舞台」，並為它們指定實體名稱為 **aBtn** 與 **bBtn**。在「屬性」檢測器的「參數」索引標籤中，將它們的標籤命名為 Button A 與 Button B。

3 在「時間軸」之「影格 1」的「動作」面板中，加入下列程式碼：

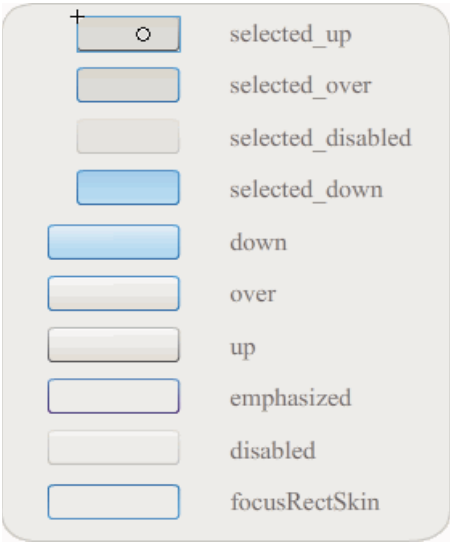
```
bBtn.emphasized = true;
aBtn.emphasized = true;
bBtn.addEventListener(MouseEvent.CLICK, Btn_handler);
function Btn_handler(evt:MouseEvent):void {
    bBtn.setStyle("emphasizedSkin", "Button_selectedOverSkin");
}
```

4 選取「控制 > 測試影片」。

5 按一下每一個按鈕，查看 emphasizedSkin 樣式對每一個按鈕的影響。

搭配 Button 組件使用外觀元素

Button 組件會使用下列外觀元素，這些外觀元素會與它的不同狀態相對應。若要編輯一個或多個外觀元素以變更 Button 的外觀，請按兩下「舞台」上的 Button 實體，以開啟其外觀元素的面板，如下圖所示：

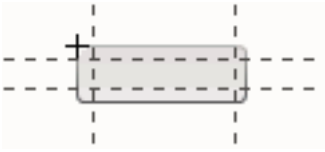


Button 外觀元素

按鈕啟用之後，當指標移動到它上面時，就會顯示它的滑入狀態。按下按鈕時，該按鈕便會成為輸入焦點並且顯示按下狀態。放開滑鼠之後，按鈕又會回復成原來的滑入狀態。如果在按下滑鼠時將指標移開按鈕，該按鈕便會回復成原始狀態。如果將 toggle 參數設定為 true，就會以 selectedDownSkin 顯示按下狀態、以 selectedUpSkin 顯示一般狀態，並且以 selectedOverSkin 顯示滑入狀態。

Button 停用之後，不論使用者的動作為何，都只會顯示其停用狀態。

若要編輯其中一個外觀元素，請按兩下該外觀元素，以元件編輯模式加以開啟，如下圖所示：



元件編輯模式中的 Button

此時您可以使用 Flash 編寫工具，將外觀元素編輯成自己喜歡的樣子。

下列程序會變更 Button 之 selected_over 外觀元素的顏色。

1 建立新的 Flash 檔案 (ActionScript 3.0)。

2 將 Button 從 「組件」面板拖曳到 「舞台」。在 「參數」索引標籤中，將 toggle 參數設定為 true。

3 按兩下 Button，開啟其外觀元素的面板。

4 按兩下 selected_over 外觀元素，以元件編輯模式加以開啟。

5 將縮放控制項設定為 400%，將圖示放大以進行編輯。

6 按兩下背景，直到其顏色出現在 「屬性」檢測器的 「填色」顏色挑選器中為止。

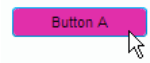
7 在 「填色」顏色挑選器中選取 #CC0099，將該顏色套用到 selected_over 外觀元素的背景。

8 按一下 「舞台」上方編輯列左側的 「後退」按鈕，回到文件編輯模式。

9 選取 「控制 > 測試影片」。

10 按一下該按鈕，使其成為已選取模式。

當您將滑鼠指標移到 Button 上方時，selected_over 狀態看起來應該如下圖所示。



顯示已修改顏色之 selected_over 外觀元素的 Button

自訂 CheckBox 組件

您可以在編寫期間和執行階段，沿水平和垂直方向變形 CheckBox 組件。在編寫期間，請在「舞台」上選取組件，並且使用「自由變形」工具或任何「修改 > 變形」命令。在執行階段，請使用 setSize() 方法或 CheckBox 類別適用的屬性。例如，您可以設定 CheckBox 的 height 和 width 以及 scaleX 和 scaleY 屬性，以變更 CheckBox 的大小。調整 CheckBox 的大小不會變更標籤或核取方塊圖示的大小，只會變更範圍框的大小。

CheckBox 實體的範圍框是隱藏的，而且也指定了實體的作用區域。如果您增加實體的大小，也會同時增加作用區域的大小。如果範圍框太小無法配合標籤，就會裁剪標籤以便配合。

搭配 CheckBox 使用樣式

您可以設定樣式屬性來變更 CheckBox 實體的外觀。例如，下列程序會變更 CheckBox 標籤的大小和顏色。

1 將 CheckBox 組件從「組件」面板拖曳到「舞台」，並為它指定實體名稱為 **myCb**。

2 按一下「屬性」檢測器中的「參數」索引標籤，然後輸入下列值做為 label 參數的值：**Less than $500?**

3 在主要「時間軸」之「影格 1」的「動作」面板中，輸入下列程式碼：

```
var myTf:TextFormat = new TextFormat();
myCb.setSize(150, 22);
myTf.size = 16;
myTf.color = 0xFF0000;
myCb.setStyle("textFormat", myTf);
```

如需詳細資訊，請參閱第 87 頁「設定樣式」。如需設定樣式屬性以變更組件的圖示和外觀元素之詳細資訊，請參閱第 90 頁「建立新的外觀元素」和第 94 頁「搭配 CheckBox 使用外觀元素」。

搭配 CheckBox 使用外觀元素

CheckBox 組件具有下列外觀元素，您可以編輯這些外觀元素來變更該組件的外觀。

CheckBox 外觀元素

此範例會變更該組件在其 up 和 selectedUp 狀態中的外框顏色和背景顏色。您可以遵循類似的步驟，變更其它狀態的外觀元素。

1 建立新的 Flash 檔案 (ActionScript 3.0) 文件。

2 將 CheckBox 組件拖曳到 「舞台」，而此舉也會將該組件連同包含其資源的資料夾一併放置於元件庫中。

3 按兩下 「舞台」上的 CheckBox 組件，開啟其外觀元素圖示的面板。

4 按兩下 selected_up 圖示，以元件編輯模式加以開啟。

5 將縮放控制項設定為 800%，將圖示放大以進行編輯。

6 按一下以選取 CheckBox 的邊框。使用 「屬性」檢測器中的 「填色」顏色挑選器選取 #0033FF，並將該顏色套用到邊框。

7 按兩下以選取 CheckBox 的背景，然後再次使用 「填色」顏色挑選器，將背景顏色設定為 #00CCFF。

8 針對 CheckBox 一般外觀元素重複步驟 4 到步驟 8。

9 選取 「控制 > 測試影片」。

自訂 ColorPicker 組件

您唯一能對 ColorPicker 執行之調整大小的動作是透過它的樣式：swatchWidth、swatchHeight、backgroundPadding、textFieldWidth 和 textFieldHeight。如果您嘗試以 setSize() 方法搭配使用 「變形」工具或 ActionScript，或使用 width、height、scaleX 或 scaleY 屬性來變更 ColorPicker 的大小，則在您建立 SWF 檔時會忽略這些值，而 ColorPicker 會依預設大小顯示。面板背景會調整大小以符合使用 setStyle() 針對 columnCount 樣式所設定的欄數。預設的欄數為 18。您可以設定多達 1024 個自訂顏色，而且面板將會以垂直方式調整大小，以符合色票的數目。

搭配 ColorPicker 組件使用樣式

您可以設定多種樣式來變更 ColorPicker 組件的外觀。例如，下列程序會將 ColorPicker 中的欄數 (columnCount) 變更為 12、變更色票的高度 (swatchHeight) 和寬度 (swatchWidth)，以及變更文字欄位 (textPadding) 和背景 (backgroundPadding) 的邊框間距。

1 建立新的 Flash 檔案 (ActionScript 3.0) 文件。

2 將 ColorPicker 組件拖曳到 「舞台」，並為它指定實體名稱為 **aCp**。

3 開啟 「動作」面板，選取主要 「時間軸」中的 「影格 1」，然後輸入下列程式碼：

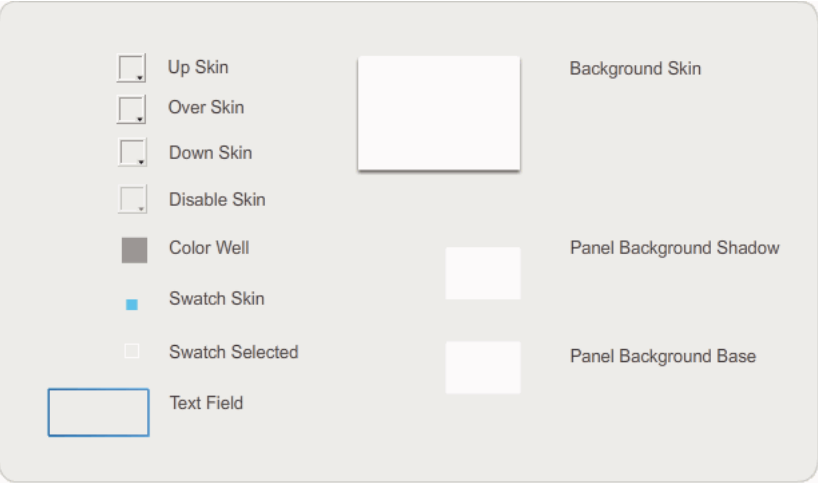
```
aCp.setStyle("columnCount", 12);
aCp.setStyle("swatchWidth", 8);
aCp.setStyle("swatchHeight", 12);
aCp.setStyle("swatchPadding", 2);
aCp.setStyle("backgroundPadding", 3);
aCp.setStyle("textPadding", 7);
```

4 選取 「控制 > 測試影片」。

5 按一下以開啟 ColorPicker，並查看這些設定是以何種方式改變該組件的外觀。

搭配 ColorPicker 組件使用外觀元素

ColorPicker 組件會使用下列外觀元素來代表其視覺狀態。



ColorPicker 外觀元素

您可以變更 「Background Skin」元素的顏色，以變更面板背景顏色。

1 建立新的 Flash 檔案 (ActionScript 3.0) 文件。

2 將 ColorPicker 組件拖曳到 「舞台」。

3 按兩下該組件，開啟其外觀元素的面板。

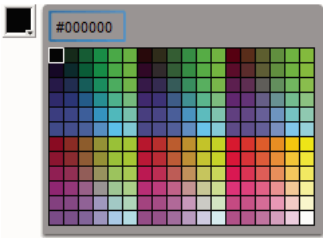
4 按兩下 「Background Skin」，直到選取該外觀元素而且 「填色」顏色挑選器出現在 「屬性」檢測器中為止。

5 使用 「填色」顏色挑選器選取 #999999，以便將該顏色套用到 「Background Skin」。

6 按一下 「舞台」上方編輯列左側的 「後退」按鈕，回到文件編輯模式。

7 選取 「控制 > 測試影片」。

當您按一下 ColorPicker 時，面板的背景應該會是灰色，如下圖所示。



具有深灰色 「背景」外觀元素的 ColorPicker

自訂 ComboBox 組件

您可以在編寫期間和執行階段，沿水平和垂直方向變形 ComboBox 組件。在編寫期間，請在 「舞台」上選取組件，並且使用 「自由變形」工具或任何 「修改 > 變形」命令。在執行階段，請使用 setSize() 方法或 ComboBox 類別適用的屬性 (如 height 和 width 以及 scaleX 和 scaleY)。

ComboBox 會調整大小以符合指定的寬度與高度。除非已經設定 dropdownWidth 屬性，否則此清單將會調整大小以符合該組件的寬度。

如果文字太長而無法容納在 ComboBox 中，就會裁剪文字來配合。您必須調整 ComboBox 的大小，並且設定 dropdownWidth 屬性以符合文字。

對 ComboBox 組件使用樣式

您可以設定樣式屬性來變更 ComboBox 組件的外觀。這些樣式會指定該組件的外觀元素、儲存格輸出器、邊框間距與按鈕寬度的值。下列範例會設定 buttonWidth 和 textPadding 樣式。buttonWidth 樣式會設定該按鈕作用區域的寬度；當 ComboBox 可以編輯，而且您只需下該按鈕即可開啟下拉式清單時，此樣式便會生效。textPadding 樣式則指定文字欄位外圍邊框與文字之間的間距大小。如果您將 ComboBox 設定得更高一些，上述作法對於將文字在垂直方向置中對齊會很有用。否則，文字可能就會出現在文字欄位的頂端。

1 建立新的 Flash 檔案 (ActionScript 3.0) 文件。

2 將 ComboBox 組件拖曳到「舞台」，並為它指定實體名稱 **aCb**。

3 開啟「動作」面板，選取主要「時間軸」中的「影格 1」，然後輸入下列程式碼：

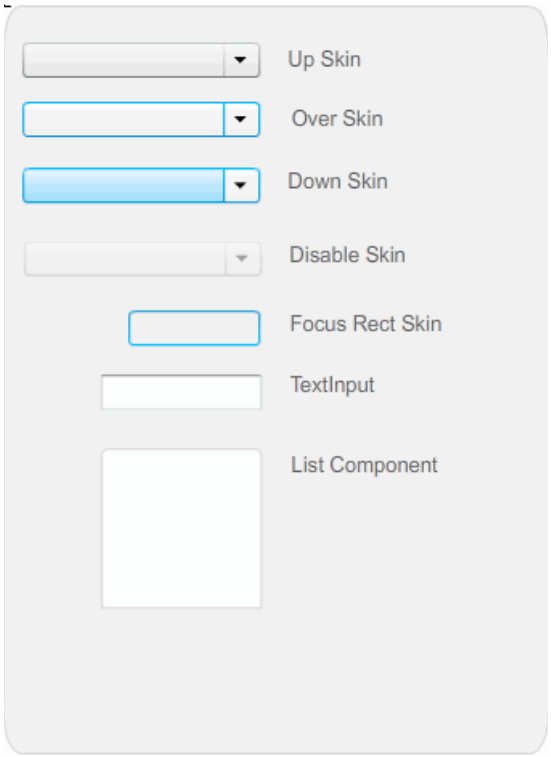
```
import fl.data.DataProvider;

aCb.setSize(150, 35);
aCb.setStyle("textPadding", 10);
aCb.setStyle("buttonWidth", 10);
aCb.editable = true;

var items:Array = [
{label:"San Francisco", data:"601 Townsend St."},
{label:"San Jose", data:"345 Park Ave."},
{label:"San Diego", data:"10590 West Ocean Air Drive, Suite 100"},
{label:"Santa Rosa", data:"2235 Mercury Way, Suite 105"},
{label:"San Luis Obispo", data:"3220 South Higuera Street, Suite 311"}
];
aCb.dataProvider = new DataProvider(items);
```

4 選取「控制 > 測試影片」。

請注意，只有位於右邊的狹窄區域是按一下即可開啟下拉式清單的按鈕區域。此外，還要注意一件事，就是文字欄位中的文字會在垂直方向置中對齊。您可以嘗試在不使用這兩個 setStyle() 陳述式的情況下執行此範例，並得知它們所造成的影響。

搭配 ComboBox 使用外觀元素

ComboBox 會使用下列外觀元素來代表其視覺狀態：



ComboBox 外觀元素

您可以變更 「Up Skin」的顏色，以變更 「舞台」上組件在其非作用狀態中的顏色。

1 建立新的 Flash 檔案 (ActionScript 3.0) 文件。

2 將 ComboBox 組件拖曳到 「舞台」。

3 按兩下該組件，開啟其外觀元素的面板。

4 按兩下 「Up Skin」，直到選取該外觀元素並開啟以供編輯為止。

5 將縮放控制項設定為 400%。

6 按一下外觀元素的中心區域，直到其顏色出現在 「屬性」檢測器的 「填色」顏色挑選器中為止。

7 使用 「填色」顏色挑選器選取 #33FF99，以便將該顏色套用到 「Up Skin」。

8 按一下 「舞台」上方編輯列左側的 「後退」按鈕，回到文件編輯模式。

9 選取 「控制 > 測試影片」。

ComboBox 應該會出現在 「舞台」上，如下圖所示。

背景外觀元素為自訂顏色的 ComboBox

自訂 DataGrid 組件

您可以在編寫期間和執行階段，沿水平和垂直方向變形 DataGrid 組件。在編寫期間，請在 「舞台」上選取組件，並且使用 「自由變形」工具或任何 「修改 > 變形」命令。在執行階段，請使用 setSize() 方法或適用的屬性 (如 width、height、scaleX 和 scaleY)。如果沒有水平捲軸，欄寬會依比例調整。如果欄 (連帶儲存格) 的大小經過調整，則儲存格中的文字可能會被裁掉。

搭配 DataGrid 組件使用樣式

您可以設定樣式屬性來變更 DataGrid 組件的外觀。DataGrid 組件會繼承 List 組件的樣式 (請參閱第 104 頁「使用具有 List 組件的樣式」)。

為個別欄設定樣式

DataGrid 物件可以擁有多個欄，您也可以為每一欄指定不同的儲存格輸出器。DataGrid 的每一欄都是由 DataGridColumn 物件表示，DataGridColumn 類別也具有 cellRenderer 屬性，您可以定義每一欄的 CellRenderer。

1 建立新的 Flash 文件 (ActionScript 3.0)。

2 將 DataGrid 組件拖曳到「元件庫」面板。

3 在時間軸的「影格 1」的「動作」面板中，加入下列程式碼。此程式碼會建立 DataGrid，並且在其第三欄中含有一長串的文字字串。最後，它會將該欄的 cellRenderer 屬性設定為可呈現多行儲存格的儲存格輸出器名稱。

```
/* This is a simple cell renderer example.It invokes
the MultiLineCell cell renderer to display a multiple
line text field in one of a DataGrid's columns. */

import fl.controls.DataGrid;
import fl.controls.dataGridClasses.DataGridColumn;
import fl.data.DataProvider;
import fl.controls.ScrollPolicy;

// Create a new DataGrid component instance.
var aDg:DataGrid = new DataGrid();

var aLongString:String = "An example of a cell renderer class that displays a multiple line TextField"
var myDP:Array = new Array();
myDP = [{firstName:"Winston", lastName:"Elstad", note:aLongString, item:100},
    {firstName:"Ric", lastName:"Dietrich", note:aLongString, item:101},
    {firstName:"Ewing", lastName:"Canepa", note:aLongString, item:102},
    {firstName:"Kevin", lastName:"Wade", note:aLongString, item:103},
    {firstName:"Kimberly", lastName:"Dietrich", note:aLongString, item:104},
    {firstName:"AJ", lastName:"Bilow", note:aLongString, item:105},
    {firstName:"Chuck", lastName:"Yushan", note:aLongString, item:106},
    {firstName:"John", lastName:"Roo", note:aLongString, item:107},
];

// Assign the data provider to the DataGrid to populate it.
// Note: This has to be done before applying the cellRenderers.
aDg.dataProvider = new DataProvider(myDP);

/* Set some basic grid properties.
```

```
Note: The data grid's row height should reflect
the number of lines you expect to show in the multiline cell.
The cell renderer wil size to the row height.
About 40 for 2 lines or 60 for 3 lines.*/

aDg.columns = ["firstName", "lastName", "note", "item"];
aDg.setSize(430,190);
aDg.move(40,40);
aDg.rowHeight = 40;// Allows for 2 lines of text at default text size.
aDg.columns[0].width = 70;
aDg.columns[1].width = 70;
aDg.columns[2].width = 230;
aDg.columns[3].width = 60;
aDg.resizableColumns = true;
aDg.verticalScrollPolicy = ScrollPolicy.AUTO;
addChild(aDg);
// Assign cellRenderers.
var col3:DataGridColumn = new DataGridColumn();
col3 = aDg.getColumnAt(2);
col3.cellRenderer = MultiLineCell;
```

4 將 FLA 檔另存成 MultiLineGrid.fla。

5 建立新的 ActionScript 檔案。

6 將下列 ActionScript 程式碼複製到「Script」視窗中：

```
package {


    import fl.controls.listClasses.CellRenderer;

    public class MultiLineCell extends CellRenderer
    {

        public function MultiLineCell()
        {
            textField.wordWrap = true;
            textField.autoSize = "left";
        }
        override protected function drawLayout():void {
            textField.width = this.width;
            super.drawLayout();
        }
    }
}
```

7 將 ActionScript 檔命名為 MultiLineCell.as，然後儲存到與 MultiLineGrid.fla 檔相同的資料夾中。

8 回到 MultiLineGrid.fla 應用程式，然後選取「控制 > 測試影片」。

該 DataGrid 看起來如下：

firstName	lastName	note	item
Winston	Elstad	An example of a cell renderer class that displays a multiple line TextField	100
Ric	Dietrich	An example of a cell renderer class that displays a multiple line TextField	101
Ewing	Canepa	An example of a cell renderer class that displays a multiple line TextField	102
Kevin	Wade	An example of a cell renderer class that displays a multiple line TextField	103

MultiLineGrid.fla 應用程式的 DataGrid

設定標題樣式

您可以使用 headerTextFormat 樣式，設定標題列的文字樣式。下列範例會使用 TextFormat 物件設定 headerTextFormat 樣式，以便使用大小為 14、斜體及紅色的 Arial 字體。

1 建立新的 Flash 檔案 (ActionScript 3.0) 文件。

2 將 DataGrid 組件拖曳到「舞台」，並為它指定實體名稱 **aDg**。

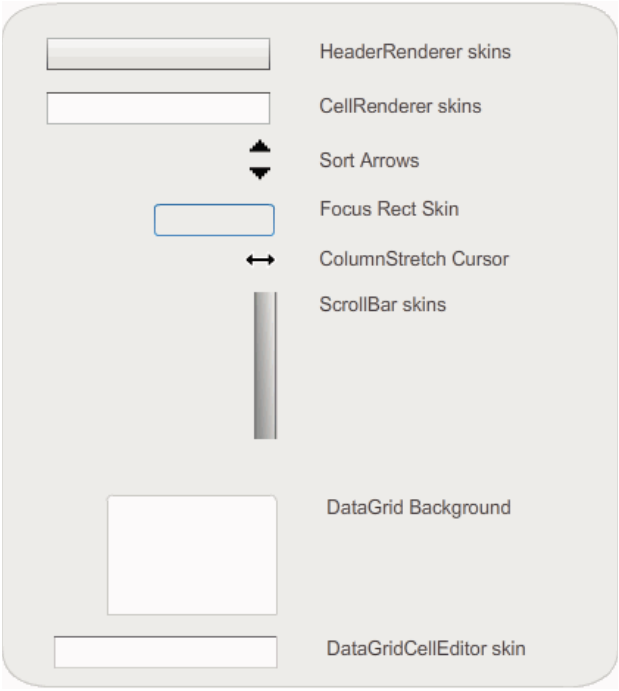
3 開啟「動作」面板，選取主要「時間軸」中的「影格 1」，然後輸入下列程式碼：

```
import fl.data.DataProvider;
import fl.controls.dataGridClasses.DataGridColumn;

var myDP:Array = new Array();
myDP = [{FirstName:"Winston", LastName:"Elstad"},
    {FirstName:"Ric", LastName:"Dietrich"},
    {FirstName:"Ewing", LastName:"Canepa"},
    {FirstName:"Kevin", LastName:"Wade"},
    {FirstName:"Kimberly", LastName:"Dietrich"},
    {FirstName:"AJ", LastName:"Bilow"},
    {FirstName:"Chuck", LastName:"Yushan"},
    {FirstName:"John", LastName:"Roo"},
];

// Assign the data provider to the DataGrid to populate it.
// Note: This has to be done before applying the cellRenderers.
aDg.dataProvider = new DataProvider(myDP);
aDg.setSize(160,190);
aDg.move(40,40);
aDg.columns[0].width = 80;
aDg.columns[1].width = 80;
var tf:TextFormat = new TextFormat();
tf.size = 14;
tf.color = 0xff0000;
tf.italic = true;
tf.font = "Arial"
aDg.setStyle("headerTextFormat", tf);
```

4 選取「控制 > 測試影片」，執行應用程式。

搭配 DataGrid 組件使用外觀元素

DataGrid 組件會使用下列外觀元素來代表其視覺狀態：



DataGrid 外觀元素

CellRenderer 外觀元素是用於 DataGrid 之主體儲存格的外觀元素，而 HeaderRenderer 外觀元素則是用於標題列的外觀元素。下列程序會變更標題列的背景顏色，但是您可以使用相同的程序來變更 DataGrid 的主體儲存格，只要編輯 CellRenderer 外觀元素即可。

1 建立新的 Flash 文件 (ActionScript 3.0)。

2 將 DataGrid 組件拖曳到 「舞台」，並為它指定實體名稱 **aDg**。

3 按兩下該組件，開啟其外觀元素的面板。

4 將縮放控制項設定為 400%，放大圖示以進行編輯。

5 按兩下 HeaderRenderer 外觀元素，開啟 HeaderRenderer 外觀元素的面板。

6 按兩下 Up_Skin，以元件編輯模式加以開啟，然後按一下其背景，直到選取背景，而且 「填色」顏色挑選器出現在 「屬性」檢測器中為止。

7 使用 「填色」顏色挑選器選取 #00CC00，將該顏色套用到 Up_Skin HeaderRenderer 外觀元素的背景。

8 按一下 「舞台」上方編輯列左側的 「後退」按鈕，回到文件編輯模式。

9 在 「時間軸」之 「影格 1」的 「動作」面板中加入下列程式碼，將資料加入 DataGrid：

```
import fl.data.DataProvider;

bldRosterGrid(aDg);
var aRoster:Array = new Array();
aRoster = [
        {Name:"Wilma Carter",Home: "Redlands, CA"},
        {Name:"Sue Pennypacker",Home: "Athens, GA"},
        {Name:"Jill Smithfield",Home: "Spokane, WA"},
        {Name:"Shirley Goth", Home: "Carson, NV"},
        {Name:"Jennifer Dunbar",Home: "Seaside, CA"}
];
aDg.dataProvider = new DataProvider(aRoster);
function bldRosterGrid(dg:DataGrid){
    dg.setSize(400, 130);
    dg.columns = ["Name", "Home"];
    dg.move(50,50);
    dg.columns[0].width = 120;
    dg.columns[1].width = 120;
};
```

10 選取 「控制 > 測試影片」，測試應用程式。

如下圖所示，DataGrid 顯示時，其標題列的背景顏色應該為綠色。

Name	Home
Wilma Carter	Redlands, CA
Sue Pennypacker	Athens, GA
Jill Smithfield	Spokane, WA
Shirley Goth	Carson, NV
Jennifer Dunbar	Seaside, CA

具有自訂標題列背景的 DataGrid

自訂 Label 組件

您可以在編寫期間和執行階段，沿水平和垂直方向變形 Label 組件。在編寫期間，請在 「舞台」上選取組件，並且使用 「自由變形」工具或任何 「修改 > 變形」命令。您也可以設定 autoSize 編寫參數；設定這個參數不會變更即時預覽中的範圍框，但是會調整 Label 大小。Label 會根據 wordwrap 參數來調整大小。如果該參數為 true，Label 就會在垂直方向調整大小以符合文字。如果該參數為 false，Label 會在水平方向調整大小。在執行階段，請使用 setSize() 方法。如需詳細資訊，請參閱 「適用於 Adobe Flash Platform 的 ActionScript 3.0 參考」中的 Label.setSize() 方法和 Label.autoSize 屬性。請參閱第 55 頁 「建立具有 Label 組件的應用程式」。

搭配 Label 組件使用樣式

您可以設定樣式屬性來變更標籤實體的外觀。Label 組件實體中的所有文字必須是相同的樣式。Label 組件具有 textFormat 樣式，這個樣式具有與 TextFormat 物件相同的特質，並且可以讓您為 Label.text 的內容設定與為一般 Flash TextField 所設定的相同屬性。下列範例會將標籤中的文字顏色設定為紅色。

1 將 Label 組件從 「組件」面板拖曳到 「舞台」，並且為它指定實體名稱為 a_label。

2 按一下 「參數」索引標籤，並以下列文字取代 text 屬性的值：

Color me red

3 在主要時間軸中選取 「影格 1」，開啟 「動作」面板，然後輸入下列程式碼：

```
/* Create a new TextFormat object, which allows you to set multiple text properties at a time. */

var tf:TextFormat = new TextFormat();
tf.color = 0xFF0000;
/* Apply this specific text format (red text) to the Label instance. */
a_label.setStyle("textFormat", tf);
```

4 選取「控制 > 測試影片」。

如需有關 Label 樣式的詳細資訊，請參閱適用於 Adobe Flash Platform 的 ActionScript 3.0 參考中的「Label 類別」。

外觀元素和 Label

Label 組件不具有任何外觀元素的視覺元素。

自訂 List 組件

您可以在編寫期間和執行階段，沿水平和垂直方向變形 List 組件。在編寫期間，請在「舞台」上選取組件，並且使用「自由變形」工具或任何「修改 > 變形」命令。在執行階段，請使用 setSize() 方法和 List 類別適用的屬性 (如 height、width、scaleX 和 scaleY)。

調整清單大小時，清單的列會在水平方向縮小，裁剪它們之內的任何文字；在垂直方向，清單將會視需要增加或移除列；捲軸也會視需要自動放置到定位。

使用具有 List 組件的樣式

您可以設定樣式屬性來變更 List 組件的外觀。繪製該組件時，這些樣式會指定該組件外觀元素和邊框間距的值。

這些各式各樣的外觀元素樣式可以讓您指定要用於外觀元素的不同類別。如需有關使用外觀元素樣式的詳細資訊，請參閱第 89 頁「關於外觀元素」。

下列程序會針對 List 組件，設定 contentPadding 樣式的值。請注意，此設定的值會減去 List 的大小，以符合內容周圍的邊框間距。如此一來，您可能就必須增加 List 的大小，以免 List 中的文字遭到裁切。

1 建立新的 Flash 檔案 (ActionScript 3.0) 文件。

2 將 List 組件從「組件」面板拖曳到「舞台」，並為它指定實體名稱 **aList**。

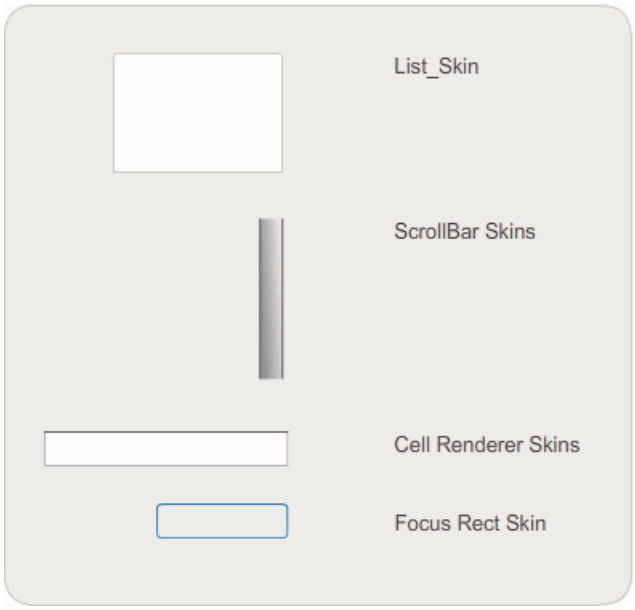
3 在主要「時間軸」中選取「影格 1」，開啟「動作」面板，然後輸入下列程式碼，以設定 contentPadding 樣式並將資料加入 List 中：

```
aList.setStyle("contentPadding", 5);
aList.setSize(145, 200);
aList.addItem({label:"1956 Chevy (Cherry Red)", data:35000});
aList.addItem({label:"1966 Mustang (Classic)", data:27000});
aList.addItem({label:"1976 Volvo (Xcllnt Cond)", data:17000});
aList.rowCount = aList.length;
```

4 選取「控制 > 測試影片」。

搭配 List 組件使用外觀元素

List 組件會使用下列外觀元素來代表其視覺狀態：

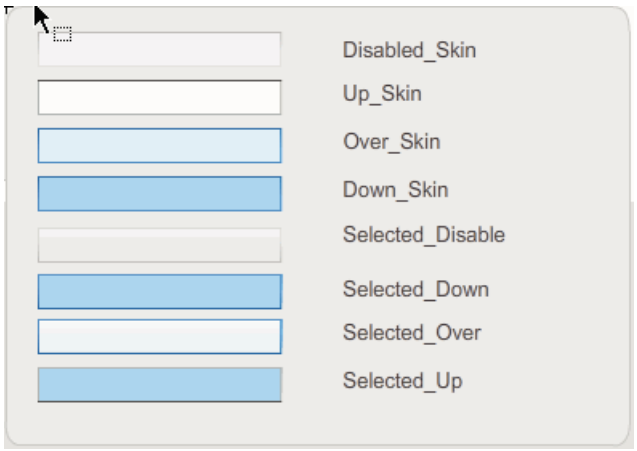


List 外觀元素

如需有關為 ScrollBar 進行外觀設定的詳細資訊，請參閱第 117 頁「自訂 UIScrollBar 組件」。如需有關為「焦點矩形」外觀元素進行外觀設定的資訊，請參閱第 113 頁「自訂 TextArea 組件」。

備註：變更某個組件中的 ScrollBar 外觀元素，也會變更使用該 ScrollBar 之其它所有組件的這個外觀元素。

按兩下「儲存格輸出器外觀」，開啟 List 儲存格不同狀態之外觀元素的第二個面板。



List 的儲存格輸出器外觀元素

您可以編輯這些外觀元素，以變更 List 之儲存格的外觀。下列程序會變更「Up Skin」的顏色，以變更 List 在正常非作用中狀態的外觀。

1 建立新的 Flash 檔案 (ActionScript 3.0) 文件。

2 將 List 組件從「組件」面板拖曳到「舞台」，並為它指定實體名稱 **aList**。

3 按兩下 List，開啟其外觀元素的面板。

4 按兩下「Cell Renderer Skin」，開啟「儲存格輸出器」外觀元素的面板。

5 按兩下以開啟 Up_Skin 外觀元素，準備進行編輯。

6 按一下以選取該外觀元素的填色區域。「填色」顏色挑選器應該會出現在 「屬性」檢測器中，而且具有該外觀元素目前的填色顏色。

7 使用 「填色」顏色挑選器選取 #CC66FF，將該顏色套用到 Up_Skin 外觀元素的填色。

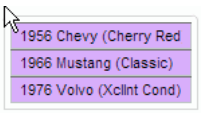
8 按一下 「舞台」上方編輯列左側的 「後退」按鈕，回到文件編輯模式。

9 在 「時間軸」之 「影格 1」的 「動作」面板中加入下列程式碼，將資料加入 List：

```
aList.setStyle("contentPadding", 5);
aList.setSize(145, 200);
aList.addItem({label:"1956 Chevy (Cherry Red)", data:35000});
aList.addItem({label:"1966 Mustang (Classic)", data:27000});
aList.addItem({label:"1976 Volvo (Xcllnt Cond)", data:17000});
aList.rowCount = aList.length;
```

10 選取 「控制 > 測試影片」。

List 看起來應該如下圖：



具有自訂 Up_Skin 顏色的 List 儲存格

外框部分是設定 contentPadding 樣式的結果。

自訂 NumericStepper 組件

您可以在編寫期間和執行階段，沿水平和垂直方向變形 NumericStepper 組件。在編寫期間，請在 「舞台」上選取組件，並且使用 「自由變形」工具或任何 「修改 > 變形」命令。在執行階段，請使用 setSize() 方法或 NumericStepper 類別適用的任何屬性與方法 (如 width、height、scaleX 和 scaleY)。

調整 NumericStepper 組件的大小不會變更向下及向上按鈕的寬度。如果將步進器調整為高於預設高度，預設行為便會將箭頭按鈕固定在組件的頂端和底端。否則，9 分割縮放會決定繪製這些按鈕的方式。方向鍵一定會出現在文字方塊的右邊。

樣式和 NumericStepper 組件

您可以設定 NumericStepper 組件的樣式屬性來變更其外觀。繪製該組件時，這些樣式會指定該組件之外觀元素、邊框間距和文字格式的值。textFormat 樣式可以讓您變更 NumericStepper 之值的大小和外觀。這些各式各樣的外觀元素樣式可以讓您指定要用於外觀元素的不同類別。如需有關使用外觀元素樣式的詳細資訊，請參閱第 89 頁 「關於外觀元素」。

此程序會使用 textFormat 樣式，變更 NumericStepper 所顯示之值的外觀。

1 建立新的 Flash 文件 (ActionScript 3.0)。

2 將 NumericStepper 組件從 「組件」面板拖曳到 「舞台」，並且為它指定實體名稱為 **myNs**。

3 在主要 「時間軸」之 「影格 1」的 「動作」面板中，加入下列程式碼：

```
var tf:TextFormat = new TextFormat();
myNs.setSize(100, 50);
tf.color = 0x0000CC;
tf.size = 24;
tf.font = "Arial";
tf.align = "center";
myNs.setStyle("textFormat", tf);
```

4 選取 「控制 > 測試影片」。

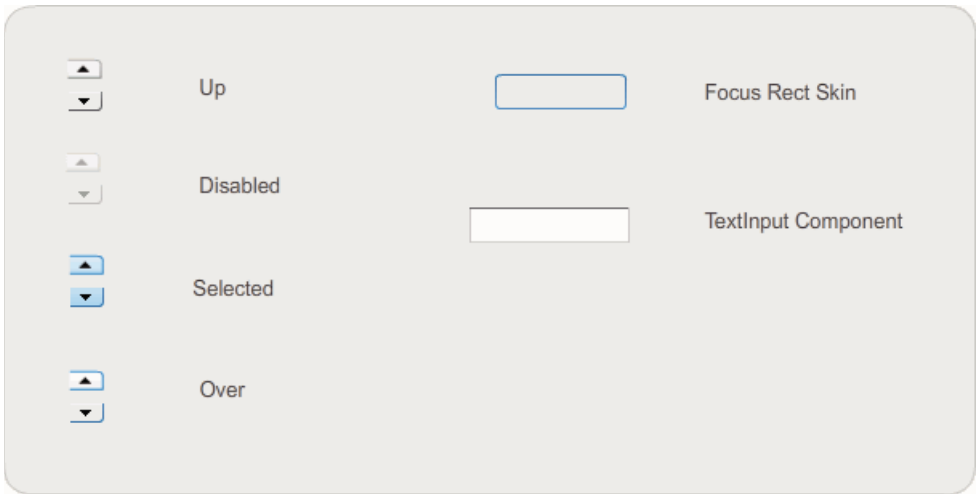
外觀元素和 NumericStepper 組件

NumericStepper 組件具有代表其按鈕之一般、按下、停用與選取狀態的外觀元素。

如果啟用了步進器，當指標移到向下及向上箭頭按鈕上時，這些按鈕就會顯示它們的滑入狀態。按下這些按鈕時，則會顯示它們的按下狀態。當滑鼠放開之後，這些按鈕又會回復它們的滑入狀態。如果在按下滑鼠時將指標移開按鈕，按鈕會回復它們的原始狀態。

Stepper 停用之後，不論使用者的動作為何，都只顯示它的停用狀態。

NumericStepper 組件具有下列外觀元素：



NumericStepper 外觀元素

1 建立新 FLA 檔。

2 將 NumericStepper 組件拖曳到 「舞台」。

3 將縮放控制項設定為 400%，將影像放大以進行編輯。

4 在外觀元素面板上按兩下 TextInput 外觀元素的背景，直到您進入 「群組」層級，而且背景顏色出現在 「屬性」檢測器的 「填色」顏色挑選器中為止。

5 使用 「屬性」檢測器中的 「填色」顏色挑選器，選取 #9999FF 以將此顏色套用到 TextInput 外觀元素的背景。

6 按一下 「舞台」上方編輯列左側的 「後退」按鈕，回到文件編輯模式。

7 再次按兩下 NumericStepper，重新開啟外觀元素面板。

8 按兩下 「一般」群組中向上箭頭按鈕的背景，直到選取該背景，且其顏色出現在 「屬性」檢測器的 「填色」顏色挑選器中為止。

9 選取 #9966FF 並將此顏色套用到向上箭頭按鈕的背景。

10 針對 「一般」群組中的向下箭頭重複步驟 8 和 9。

11 選取 「控制 > 測試影片」。

NumericStepper 實體看起來應該如下圖：

自訂 ProgressBar 組件

您可以在編寫期間和執行階段，沿水平和垂直方向變形 ProgressBar 組件。在編寫期間，請在 「舞台」上選取組件，並且使用 「自由變形」工具或任何 「修改 > 變形」命令。在執行階段，請使用 setSize() 方法或 ProgressBar 類別適用的屬性 (如 height、width、scaleX 和 scaleY)。

ProgressBar 具有三個外觀元素：軌道外觀元素、列外觀元素和非確定式外觀元素。它會使用 9 分割縮放來縮放資源。

樣式和 ProgressBar 組件

您可以設定樣式屬性來變更 ProgressBar 實體的外觀。繪製該組件時，ProgressBar 的樣式會指定其外觀元素和邊框間距的值。下列範例會放大 ProgressBar 實體，並設定其 barPadding 樣式。

1 建立新 FLA 檔。

2 將 ProgressBar 組件從 「組件」面板拖曳到 「舞台」，並且為它指定實體名稱為 **myPb**。

3 在主要 「時間軸」之 「影格 1」的 「動作」面板中，輸入下列程式碼：

```
myPb.width = 300;
myPb.height = 30;

myPb.setStyle("barPadding", 3);
```

4 選取 「控制 > 測試影片」。

如需有關設定外觀元素樣式的資訊，請參閱第 89 頁 「關於外觀元素」。

外觀元素和 ProgressBar 組件

ProgressBar 組件會使用外觀元素來代表進度列軌道、完成列以及非確定式列，如下圖所示。

ProgressBar 外觀元素

此列會置於軌道外觀元素上方，並使用 barPadding 來決定位置。資源則是使用 9 分割縮放來進行縮放。

當 ProgressBar 實體的 indeterminate 屬性設為 true 時，就會使用非確定式列。外觀元素會垂直調整大小以符合 ProgressBar 的大小。

您可以編輯這些外觀元素，以變更 ProgressBar 的外觀。例如，下列範例會變更非確定式列的顏色。

1 建立新 FLA 檔。

2 將 ProgressBar 組件拖曳到舞台，然後按兩下此組件，開啟其外觀元素圖示的面板。

3 按兩下非確定式列外觀元素。

4 將縮放控制項設定為 400%，將圖示放大以進行編輯。

5 按兩下其中一個對角線列，然後按住 Shift 鍵並按一下其它每一個對角線列。目前的顏色便會出現在 「屬性」檢測器的 「填色」顏色挑選器中。

6 按一下以開啟 「屬性」檢測器中的 「填色」顏色挑選器，然後選取 #00CC00 並將此顏色套用到所選取的對角線列。

7 按一下 「舞台」上方編輯列左側的 「後退」按鈕，回到文件編輯模式。

8 選取 「控制 > 測試影片」。

ProgressBar 看起來應該如下圖。

自訂 RadioButton 組件

您可以在編寫期間和執行階段，沿水平和垂直方向變形 RadioButton 組件。在編寫期間，請在 「舞台」上選取組件，並且使用 「自由變形」工具或任何 「修改 > 變形」命令。在執行階段，請使用 setSize() 方法。

RadioButton 組件的範圍框是隱藏的，而且也指定了組件的作用區域。如果您增加組件的大小，也會同時增加作用區域的大小。

如果組件的範圍框太小無法配合組件標籤，就會裁剪標籤來配合。

使用具有 RadioButton 組件的樣式

您可以設定樣式屬性來變更 RadioButton 的外觀。繪製該組件時，RadioButton 的樣式屬性會指定其外觀元素、圖示、文字格式以及邊框間距的值。繪製該組件時，RadioButton 的樣式會指定其外觀元素的值和版面的邊框間距。

下列範例會擷取 CheckBox 組件的 textFormat 樣式，並將此樣式套用到 RadioButton，讓它們的標籤樣式完全相同。

1 建立新的 Flash 文件 (ActionScript 3.0)。

2 將 CheckBox 組件拖曳到 「舞台」，並在 「屬性」檢測器中為它指定實體名稱 **myCh**。

3 將 RadioButton 拖曳到 「舞台」，並在 「屬性」檢測器中為它指定實體名稱 **myRb**。

4 在時間軸的 「影格 1」的 「動作」面板中，加入下列程式碼。

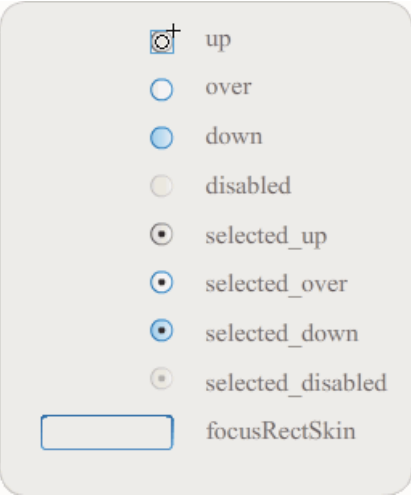
```
var tf:TextFormat = new TextFormat();
tf.color = 0x00FF00;
tf.font = "Georgia";
tf.size = 18;
myCh.setStyle("textFormat", tf);
myRb.setStyle("textFormat", myCh.getStyle("textFormat"));
```

此程式碼會設定 CheckBox 的 textFormat 樣式，然後再針對此 CheckBox 呼叫 getStyle() 方法，將此樣式套用到 RadioButton。

5 選取 「控制 > 測試影片」。

外觀元素和 RadioButton 組件

RadioButton 具有下列外觀元素，您可以編輯這些外觀元素來變更該組件的外觀：



RadioButton 外觀元素

如果已啟用但未選取 RadioButton，當使用者將指標移到它的上方時，它將會顯示其滑入外觀元素。當使用者按一下 RadioButton 時，它會成為輸入焦點並顯示其 selected_down 外觀元素。當使用者放開滑鼠時，該 RadioButton 便會顯示其 selected_up 外觀元素。如果使用者在按滑鼠按鈕時，將指標移出 RadioButton 作用中區域之外，RadioButton 便會重新顯示其 Up 外觀元素。

RadioButton 停用之後，不論使用者的動作為何，都只會顯示其停用狀態。

下列範例會取代表示為已選取狀態的 selected_up 外觀元素。

1 建立新的 Flash 文件 (ActionScript 3.0)。

2 將 RadioButton 組件拖曳到 「舞台」，然後按兩下此組件，開啟其外觀元素的面板。

3 將縮放控制項設定為 800%，將圖示放大以進行編輯。

4 按兩下以選取 selected_up 外觀元素，然後按 Delete 鍵刪除此外觀元素。

5 在 「工具」面板中選取 「矩形」工具。

6 在 「屬性」檢測器中，將線條顏色設定為紅色 (#FF0000)，並將 「填色」顏色設定為黑色 (#000000)。

7 從標記元件之註冊點 (也稱為 「原點」或 「零點」) 的十字準線開始，按一下並拖曳指標以繪製矩形。

8 按一下 「舞台」上方編輯列左側的 「後退」按鈕，回到文件編輯模式。

9 選取 「控制 > 測試影片」。

10 按一下以選取 RadioButton。

在已選取狀態中的 RadioButton 看起來應該與下列其中一張圖類似。

自訂 ScrollPane 組件

您可以在編寫期間和執行階段，沿水平和垂直方向變形 ScrollPane 組件。在編寫期間，請在 「舞台」上選取組件，並且使用 「自由變形」工具或任何 「修改 > 變形」命令。在執行階段，請使用 setSize() 方法或 ScrollPane 類別適用的任何屬性與方法 (如 height、width、scaleX 和 scaleY)。

ScrollPane 組件具有下列圖像特性：

- 其內容的註冊點 (也稱為 「原點」或 「零點」) 位於窗格左上角。
- 當水平捲軸關閉時，垂直捲軸會沿著捲動窗格右邊從上到下顯示。當垂直捲軸關閉時，水平捲軸會沿著捲動窗格底部從左到右顯示。您也可以關閉這兩個捲軸。
- 如果捲動窗格太小，可能會無法正確顯示內容。
- 調整捲動窗格的大小時，捲動軌道和捲動方塊 (縮圖) 則會展開或收縮，並且它們的作用區域也會隨之調整大小；而按鈕則會維持相同的大小。

使用具有 ScrollPane 組件的樣式

繪製 ScrollPane 組件時，其樣式屬性會指定其外觀元素和邊框間距的值做為其版面。透過這些各式各樣的外觀元素樣式，可以讓您指定要用於組件之外觀元素的不同類別。如需有關使用外觀元素樣式的詳細資訊，請參閱第 89 頁 「關於外觀元素」。

1 建立新的 Flash 文件 (ActionScript 3.0)。

2 將 ScrollPane 組件拖曳到 「舞台」，並為它指定實體名稱為 **mySp**。

3 按一下 「屬性」檢測器中的 「參數」索引標籤，並對 source 參數輸入下列值：
http://www.helpexamples.com/flash/images/image1.jpg。

4 在主要 「時間軸」之 「影格 1」的 「動作」面板中加入下列程式碼。

```
mySp.setStyle("contentPadding", 5);
```

請注意，在捲軸以外，位於組件的邊框和其內容之間，會套用邊框間距。

5 選取 「控制 > 測試影片」。

外觀元素和 ScrollPane

ScrollPane 組件會使用邊框和捲軸做為捲動資源。如需有關為捲軸進行外觀設定的資訊，請參閱第 117 頁 「搭配 UIScrollBar 組件使用外觀元素」。

自訂 Slider 組件

您可以在編寫期間和執行階段，沿水平方向變形 Slider 組件。在編寫期間，請在 「舞台」上選取組件，並且使用 「自由變形」工具或任何 「修改 > 變形」命令。在執行階段，請使用 setSize() 方法或 Slider 類別適用的任何屬性 (如 width 和 scaleX 屬性)。

您只能增加滑動軸的長度，而不能增加它的高度。Flash 會忽略 height 屬性與 setSize() 方法的 height 參數。但是，您可以建立垂直滑動軸，並且增加它的垂直方向長度。

樣式和 Slider 組件

UIScrollBar 組件只會指定類別做為其外觀元素和 FocusRectPadding 的值，這會指定要用來當做 (該組件之範圍框和其外圍邊界之間的) 邊框間距的像素數目。如需有關使用外觀元素樣式的詳細資訊，請參閱第 89 頁 「關於外觀元素」。

外觀元素和 Slider 組件

Slider 組件會使用下列外觀元素，您可以編輯這些外觀元素來變更該組件的外觀。

Slider 外觀元素

下列範例會編輯一般軌道，將其顏色變更為藍色。

1. 建立新的 Flash 文件 (ActionScript 3.0)。
2. 將 Slider 組件從 「組件」面板拖曳到 「舞台」。
3. 按兩下 Slider 組件，開啟其外觀元素面板。
4. 按兩下其對齊標記上的一般軌道，以元件編輯模式加以開啟。
5. 將縮放控制項設定為 800%，將圖示放大以進行編輯。請注意，此 Slider 的軌道包含三個列。
6. 按一下以選取位於頂端的那一列。選取該列時，它的顏色會出現在 「屬性」檢測器的 「填色」顏色挑選器中。
7. 使用 「屬性」檢測器中的 「填色」顏色挑選器，選取 #000066 以將此顏色套用到 Slider 軌道中位於頂端的那一列。
8. 按一下以選取 Slider 軌道中位於中間的那一列。選取該列時，它的顏色會出現在 「屬性」檢測器的 「填色」顏色挑選器中。
9. 使用 「屬性」檢測器中的 「填色」顏色挑選器，選取 #0066FF 以將此顏色套用到 Slider 軌道中位於中間的那一列。
10. 按一下以選取 Slider 軌道中位於底端的那一列。選取該列時，它的顏色會出現在 「屬性」檢測器的 「填色」顏色挑選器中。
11. 使用 「屬性」檢測器中的 「填色」顏色挑選器，選取 #00CCFF 以將此顏色套用到 Slider 軌道中位於底端的那一列。
12. 按一下 「舞台」上方編輯列左側的 「後退」按鈕，回到文件編輯模式。
13. 選取 「控制 > 測試影片」。

    Slider 看起來應該如下圖：

自訂 TextArea 組件

您可以在編寫期間和執行階段，沿水平和垂直方向變形 TextArea 組件。在編寫期間，請在 「舞台」上選取組件，並且使用 「自由變形」工具或任何 「修改 > 變形」命令。在執行階段，請使用 setSize() 方法或 TextArea 類別適用的任何屬性，例如 height、width、scaleX 和 scaleY。

調整 TextArea 組件大小時，邊框會調整為新範圍框的大小。如果需要捲軸，捲軸會放在底部和右邊緣。接下來，文字區域會在剩下的區域內重新調整大小；TextArea 組件中沒有固定大小的元素。如果 TextArea 組件的寬度太小而無法顯示文字，則文字將會遭到裁剪。

樣式和 TextArea 組件

繪製 TextArea 組件時，其樣式會指定其外觀元素、邊框間距和文字格式的值。texFormat 和 disabledTextFormat 樣式會管理 TextArea 顯示的文字樣式。如需有關外觀元素樣式屬性的詳細資訊，請參閱第 113 頁 「搭配 TextArea 組件使用外觀元素」。

下列範例會設定 disabledTextFormat 樣式以變更 TextArea 停用時的文字外觀，但是相同的程序會套用到針對已啟用之 TextArea 的 textFormat 樣式的設定作業。

1 建立新的 Flash 檔案。

2 將 TextArea 組件拖曳到 「舞台」，並且為它指定實體名稱為 **myTa**。

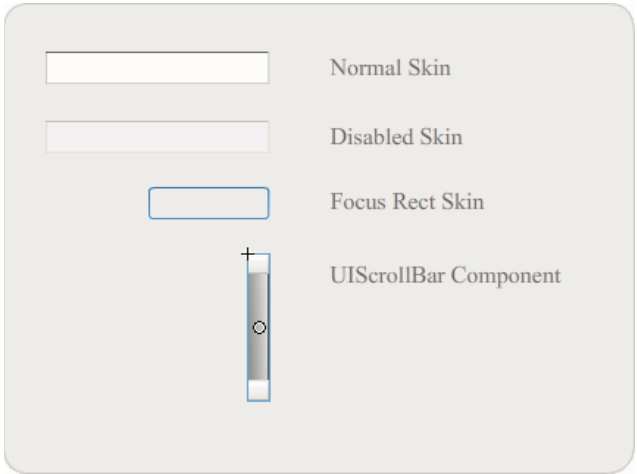
3 在主要 「時間軸」之 「影格 1」的 「動作」面板中，加入下列程式碼。

```
var tf:TextFormat = new TextFormat();
tf.color = 0xCC99FF;
tf.font = "Arial Narrow";
tf.size = 24;
myTa.setStyle("disabledTextFormat", tf);
myTa.text = "Hello World";
myTa.setSize(120, 50);
myTa.move(200, 50);
myTa.enabled = false;
```

4 選取 「控制 > 測試影片」。

搭配 TextArea 組件使用外觀元素

TextArea 組件會使用下列外觀元素，您可以編輯這些外觀元素來變更該組件的外觀。



TextArea 外觀元素

備註：變更某個組件中的 ScrollBar 外觀元素，也會變更使用該 ScrollBar 之其它所有組件的這個外觀元素。

下列程序會變更 「Focus Rect Skin」 (在 TextArea 成為焦點時顯示) 與 「Up」外觀元素的邊框顏色。

1 建立新的 Flash 檔案。

2 將 TextArea 組件拖曳到舞台，然後按兩下此組件，開啟其外觀元素圖示的面板。

3 按兩下 「Focus Rect Skin」。

4 按一下以選取 「Focus Rect Skin」的邊框。選取此邊框時，其顏色便會出現在 「屬性」檢測器的 「填色」顏色挑選器中。

5 按一下以開啟 「屬性」檢測器中的 「填色」顏色挑選器，然後選取 #CC0000，將此顏色套用到該邊框。

6 按一下 「舞台」上方編輯列左側的 「後退」按鈕，回到文件編輯模式。

7 按兩下 TextArea 組件，開啟該組件外觀元素圖示的面板。

8 按兩下 「Normal Skin」。

9 選取 「Normal Skin」之邊框的每一邊 (一次一邊)，然後將其顏色設定為 #990099。

10 按一下 「舞台」上方編輯列左側的 「後退」按鈕，回到文件編輯模式。

11 選取 「控制 > 測試影片」。

當您選取 TextArea 並開始輸入文字時，其邊框看起來應該如下圖所示：

外圍邊框是 「Focus Rect Skin」，而內部邊框則是 「Normal Skin」的邊框。

如需有關編輯 UIScrollBar 外觀元素的資訊，請參閱第 117 頁 「自訂 UIScrollBar 組件」。

自訂 TextInput 組件

您可以在編寫期間和執行階段變更 TextInput 實體的大小。在編寫期間，請在 「舞台」上選取組件，並且使用 「自由變形」工具或任何 「修改 > 變形」命令。在執行階段，請使用 setSize() 方法和 TextInput 類別適用的屬性 (如 height、width、scaleX 和 scaleY)。

調整 TextInput 組件大小時，邊框會調整為新範圍框的大小。TextInput 組件不使用捲軸，但是插入點會隨著使用者與文字的互動而自動捲動。接下來，文字欄位會在剩下的區域中重新調整大小；TextInput 組件中沒有固定大小的元素。如果 TextInput 組件太小，無法顯示文字，文字將會被裁剪。

樣式和 NumericStepper 組件

繪製 TextInput 組件時，其樣式會指定其外觀元素、邊框間距和文字格式的值。texFormat 和 disabledTextFormat 樣式會控制在該組件中顯示的文字樣式。如需有關外觀元素樣式屬性的詳細資訊，請參閱第 115 頁 「外觀元素和 TextInput 組件」。

下列範例會設定 textFormat 樣式，以便設定在 TextInput 組件中顯示的文字字體、大小和顏色。相同的程序會套用到停用該組件時，所套用之 disabledTextFormat 樣式的設定作業。

1 建立新的 Flash 文件 (ActionScript 3.0)。

2 將 TextInput 組件拖曳到 「舞台」，並且為它指定實體名稱為 **myTi**。

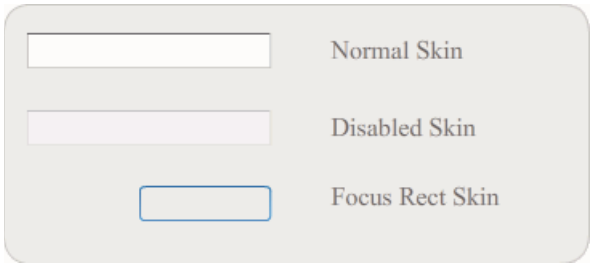
3 在主要 「時間軸」之 「影格 1」的 「動作」面板中，加入下列程式碼。

```
var tf:TextFormat = new TextFormat();
tf.color = 0x0000FF;
tf.font = "Verdana";
tf.size = 30;
tf.align = "center";
tf.italic = true;
myTi.setStyle("textFormat", tf);
myTi.text = "Enter your text here";
myTi.setSize(350, 50);
myTi.move(100, 50);
```

4 選取 「控制 > 測試影片」。

外觀元素和 TextInput 組件

TextInput 組件會使用下列外觀元素，您可以編輯這些外觀元素來變更該組件的外觀。



TextInput 註解

下列程序會變更 TextInput 組件之邊框及背景的顏色：

1 建立新的 Flash 檔案。

2 將 TextInput 組件拖曳到 「舞台」，然後按兩下此組件，開啟其外觀元素的面板。

3 按兩下 「Normal Skin」。

4 將縮放控制項設定為 800%，將圖示放大以進行編輯。

5 選取 「Normal Skin」之邊框的每一邊 (一次一邊)，然後將其顏色設定為 #993399 以套用該顏色。

6 按兩下背景，直到其顏色出現在 「屬性」檢測器的 「填色」顏色挑選器中為止。選取 #99CCCC 並將此顏色套用到背景。

7 按一下 「舞台」上方編輯列左側的 「後退」按鈕，回到文件編輯模式。

8 選取 「控制 > 測試影片」。

TextInput 組件看起來應該如下圖：

自訂 TileList 組件

您可以在編寫期間和執行階段，沿水平和垂直方向變形 TileList 組件。在編寫期間，請在 「舞台」上選取組件，並且使用 「自由變形」工具或任何 「修改 > 變形」命令。在執行階段，請使用 setSize() 方法或適用的屬性 (如 width、height、columnCount、rowCount、scaleX 和 scaleY)。TileList 中所包含的 ScrollBar 會隨清單方塊一起縮放。

樣式和 TileList 組件

繪製 TileList 組件時，其樣式會指定其外觀元素、邊框間距和文字格式的值。texFormat 和 disabledTextFormat 樣式會控制在該組件中顯示的文字樣式。如需有關外觀元素樣式的詳細資訊，請參閱第 116 頁「搭配 TileList 組件使用外觀元素」。

下列範例會使用 textFormat 樣式呼叫 setRendererStyle() 方法，以設定在 TileList 實體中顯示之標籤的字體、大小、顏色，以及文字特質。相同的程序也會套用到當 enabled 屬性設定為 false 時，所套用之 disabledTextFormat 樣式的設定作業。

1 建立新的 Flash 文件 (ActionScript 3.0)。

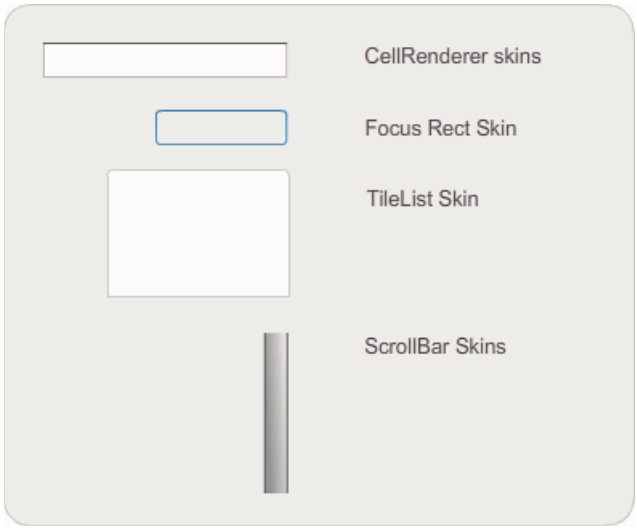
2 將 TileList 組件拖曳到「舞台」，並且為它指定實體名稱 **myTl**。

3 在時間軸的「影格 1」的「動作」面板中，加入下列程式碼。

```
myTl.setSize(100, 100);
myTl.addItem({label:"#1"});
myTl.addItem({label:"#2"});
myTl.addItem({label:"#3"});
myTl.addItem({label:"#4"});
var tf:TextFormat = new TextFormat();
tf.font = "Arial";
tf.color = 0x00FF00;
tf.size = 16;
tf.italic = true;
tf.bold = true;
tf.underline = true;
tf.align = "center";
myTl.setRendererStyle("textFormat", tf);
```

搭配 TileList 組件使用外觀元素

TileList 組件具有 TileList Skin、CellRenderer Skin，以及 ScrollBar Skin。您可以編輯這些外觀元素，以變更 TileList 的外觀：



TileList 外觀元素

備註：變更某個組件中的 ScrollBar 外觀元素，也會變更使用該 ScrollBar 之其它所有組件的這個外觀元素。

下列程序會變更 TileList 之 CellRenderer Selected_Up 外觀元素的顏色。

1 建立 Flash 文件 (ActionScript 3.0)。

2 將 TileList 組件拖曳到「舞台」，然後按兩下此組件，開啟其外觀元素的面板。

3 按兩下「CellRenderer skins」，再按兩下「Selected_Up」外觀元素，然後按一下矩形背景。

4 使用「屬性」檢測器中的「填色」顏色挑選器選取 #99FFFF，將該顏色套用到 Selected_Up 外觀元素。

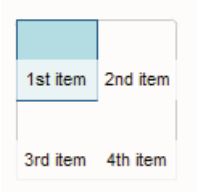
5 按一下「舞台」上方編輯列左側的「後退」按鈕，直到回到文件編輯模式為止。

6 在「屬性」檢測器的「參數」索引標籤中，按兩下 dataProvider 列的第二欄，開啟「值」對話方塊。加入具有下列標籤的項目：1st item、2nd item、3rd item、4th item。

7 選取「控制 > 測試影片」。

8 按一下以選取 TileList 內的其中一個儲存格，然後將滑鼠移開所選取的儲存格。

所選取的儲存格看起來應該如下圖：



已修改 Selected_Up 外觀元素顏色的 TileList 組件

自訂 UILoader 組件

您可以在編寫期間和執行階段，沿水平和垂直方向變形 UILoader 組件。在編寫期間，請在「舞台」上選取組件，並且使用「自由變形」工具或任何「修改 > 變形」命令。在執行階段，請使用 setSize() 方法或適當的屬性，例如 width、height、scaleX 和 scaleY 等屬性。

UILoader 組件的大小調整行為是由 scaleContent 屬性控制。當 scaleContent 為 true，則內容會進行縮放來配合載入器的範圍 (並且會在呼叫 setSize() 時重新縮放)。當 scaleContent 為 false 時，組件的大小將會固定為內容的大小，而且 setSize() 以及調整大小的屬性不會有任何作用。

UILoader 組件具有使用者介面元素，您可以將樣式或外觀元素套用到這些使用者介面元素。

自訂 UIScrollBar 組件

您可以在編寫期間和執行階段，沿水平和垂直方向變形 UIScrollBar 組件。不過，垂直的 UIScrollBar 無法讓您修改寬度，水平的 UIScrollBar 則無法讓您修改高度。在編寫期間，請在「舞台」上選取組件，並且使用「自由變形」工具或任何「修改 > 變形」命令。在執行階段，請使用 setSize() 方法或 UIScrollBar 類別適用的任何屬性 (如 width、height、scaleX 和 scaleY 屬性)。

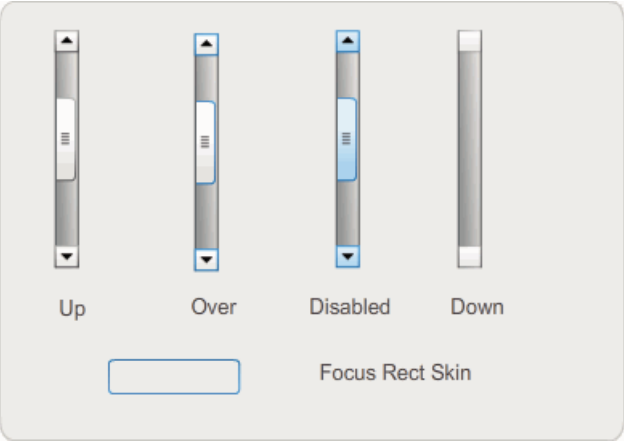
備註：若使用 setSize() 方法，您只能變更水平捲軸的寬度或垂直捲軸的高度。在編寫期間，您可以設定水平捲軸的高度或垂直捲軸的寬度，但是發佈影片時，這些值都會重新設定過。唯有對應長度的捲軸維度可以變更。

搭配 UIScrollBar 組件使用樣式

UIScrollBar 組件只會指定類別做為其外觀元素和 FocusRectPadding 的值，這會指定要用來當做 (該組件之範圍框和其外圍邊界之間的) 邊框間距的像素數目。如需有關使用外觀元素樣式的詳細資訊，請參閱第 89 頁「關於外觀元素」。

搭配 UIScrollBar 組件使用外觀元素

UIScrollBar 組件會使用下列外觀元素。



UIScrollBar 外觀元素

垂直和水平捲軸會使用相同的外觀元素。UIScrollBar 組件在顯示水平捲軸時，會適當的旋轉外觀元素。

備註：變更某個組件中的 ScrollBar 外觀元素，也會變更使用該 ScrollBar 之其它所有組件的這個外觀元素。

下列範例將示範如何變更 UIScrollBar 縮圖和箭頭按鈕的顏色。

1 建立新的 Flash 文件 (ActionScript 3.0)。

2 將 UIScrollBar 組件拖曳到「舞台」，並為它指定實體名稱為 **mySb**。在「參數」索引標籤中，將方向設定為水平方向。

3 按兩下該捲軸，開啟其外觀元素的面板。

4 按一下以選取「Up」外觀元素。

5 將縮放控制項設定為 400%，將圖示放大以進行編輯。

6 按兩下向右箭頭 (若為垂直捲軸則為向上箭頭) 的背景，直到選取該背景，且其顏色出現在「屬性」檢測器的「填色」顏色挑選器中為止。

7 選取 #CC0033 並將此顏色套用到該按鈕的背景。

8 按一下「舞台」上方編輯列左側的「後退」按鈕，直到回到文件編輯模式為止。

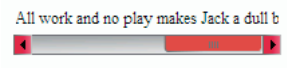
9 針對縮圖及向左箭頭 (若為垂直捲軸則為向下箭頭) 元素重複步驟 6、7 和 8。

10 在「時間軸」之「影格 1」的「動作」面板中，加入下列程式碼，以便將捲軸附加到 TextField。

```
var tf:TextField = new TextField();
addChild(tf);
tf.x = 150;
tf.y = 100;
mySb.width = tf.width = 200;
tf.height = 22;
tf.text = "All work and no play makes Jack a dull boy. All work and no play makes Jack a dull boy. All
. . .";
mySb.y = tf.y + tf.height;
mySb.x = tf.x + tf.width;x
mySb.scrollTarget = tf;
```

11 選取「控制 > 測試影片」。

UIScrollBar 組件看起來應該如下圖。



具有縮圖和紅色左右箭頭的水平 ScrollBar

第 6 章 使用 FLVPlayback 組件

FLVPlayback 組件可讓您輕鬆地在 Adobe Flash CS5 Professional 應用程式中加入視訊播放程式，以便透過 HTTP 播放漸進式下載的視訊檔，或從 Adobe 的 Macromedia Flash Media Server 或 Flash 視訊串流服務 (FVSS) 串流視訊檔。

自 Adobe Flash Player 9 更新 3 發行後 (9.0.115.0 版或更新版本)，在 Flash Player 中播放視訊內容的情況已有重大改進。這項更新包含了對 FLVPlayback 組件的變更，可充分運用使用者的系統視訊硬體以提供更優異的視訊播放效能。FLVPlayback 組件的變更也提升了以全螢幕模式顯示視訊檔的真實度。

此外，Flash Player 9 更新 3 亦新增支援採用業界標準 H.264 編碼的高畫質 MPEG-4 視訊格式，藉此改善 FLVPlayback 組件的功能。這些格式包括 MP4、M4A、MOV、MP4V、3GP 和 3G2。

備註：不支援受保護 (例如自 Apple® iTunes® 下載或經由 FairPlay® 數位加密) 的 MP4 檔案。

簡易好用的 FLVPlayback 組件具備下列特性與優點：

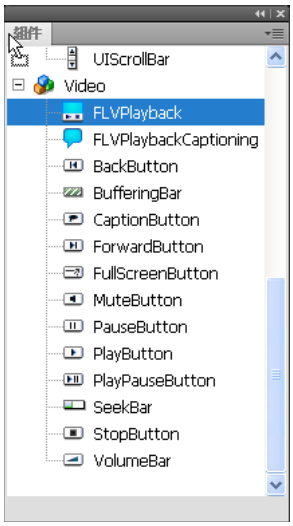
- 能夠拖曳至 「舞台」上，並快速、順利地進行實作
- 支援全螢幕大小
- 提供一組預先設計的 「外觀元素」，讓您可以自訂播放控制項的外觀
- 可以讓您為預先設計的外觀元素選取顏色和 Alpha 值
- 進階使用者可建立個人專屬的外觀元素
- 提供編寫期間的即時預覽
- 提供版面配置屬性，讓視訊檔在調整大小時保持置中
- 在已經下載足夠的漸進式下載視訊檔時，允許檔案開始播放
- 提供提示點，讓您可以將視訊與動畫、文字和圖像同步化
- 讓 SWF 檔的大小維持在合理的範圍內

使用 FLVPlayback 組件

使用 FLVPlayback 組件包含了幾個動作，例如，將組件放在 「舞台」上，以及指定要播放的視訊檔。此外，您還可以設定各種不同的參數，透過這些參數控制組件的行為及描述視訊檔。

FLVPlayback 組件也包括 ActionScript 應用程式設計介面 (API)，此 API 包括下列類別 (在 適用於 Adobe Flash Platform 的 ActionScript 3.0 參考中有完整的說明)：CuePointType、FLVPlayback、FLVPlaybackCaptioning、NCManager、NCManagerNative、VideoAlign、VideoError、VideoPlayer、VideoState 以及一些事件類別 - AutoLayoutEvent、LayoutEvent、MetadataEvent、SkinErrorEvent、SoundEvent、VideoEvent 以及 VideoProgressEvent。

FLVPlayback 組件包括了 FLV 播放自訂 UI 組件。FLVPlayback 組件是顯示區域的組合 (視訊播放程式)，可供您檢視視訊檔並使用控制項操作播放。FLV 播放自訂 UI 組件提供可用來播放、停止、暫停還有以其它方式操作視訊檔的控制項按鈕和機制。這些控制項包括 BackButton、BufferingBar、CaptionButton (用於 FLVPlaybackCaptioning)、ForwardButton、FullScreenButton、MuteButton、PauseButton、PlayButton、PlayPauseButton、SeekBar、StopButton 和 VolumeBar。FLVPlayback 組件和 FLV 播放自訂 UI 控制項會出現在 「組件」面板上，如下圖所示：



「組件」面板中的 FLVPlayback 組件

將播放控制項加入至 FLVPlayback 組件的程序稱為「外觀設定」。FLVPlayback 組件有一個起始預設外觀元素 SkinOverAll.swf，可提供播放、停止、後退、快轉、搜尋列、靜音、音量、全螢幕和註解功能等控制項。若要變更這個外觀元素，您可以採用下列選擇：

- 從一組預先設計的外觀元素中選取
- 建立自訂的外觀元素，並將它加入一組預先設計的外觀元素中
- 從 FLV 播放自訂 UI 組件中選取並自訂個別控制項

  在編寫期間或執行階段選取預先設計的外觀元素時，您可以分開選擇外觀元素顏色和 Alpha 值。如需詳細資訊，請參閱第 135 頁「選取預先設計的外觀元素」。

  當您選取不同外觀元素之後，選取的外觀元素就會成為新的預設外觀元素。

  如需有關選取或建立 FLVPlayback 組件外觀的詳細資訊，請參閱第 134 頁「自訂 FLVPlayback 組件」。

建立具有 FLVPlayback 組件的應用程式

您可以使用下列方式，在應用程式中加入 FLVPlayback 組件：

- 將 FLVPlayback 組件從「組件」面板拖曳到「舞台」上，並指定 source 參數的值。
- 使用「視訊匯入」精靈，在「舞台」上建立組件，並且選擇外觀元素來自訂這個組件。
- 使用 FLVPlayback() 建構函式動態地在「舞台」上建立 FLVPlayback 實體 (假設該組件位於元件庫中)。

  備註：如果是使用 ActionScript 建立 FLVPlayback 實體，您必須同時使用 ActionScript 設定 skin 屬性，將外觀元素指定給這個實體。若採用這種方式套用外觀元素，就不會自動隨 SWF 檔發佈外觀元素。您必須將應用程式 SWF 檔和外觀元素 SWF 檔一併複製到應用程式伺服器，否則在執行應用程式時將無法使用外觀元素 SWF 檔。

從組件面板拖曳 **FLVPlayback** 組件

1. 在「組件」面板中，按一下加號 (+) 按鈕以開啟視訊項目。
2. 將 FLVPlayback 組件拖曳到「舞台」。
3. 選取「舞台」上的 FLVPlayback 組件，然後在「組件檢測器」的「參數」索引標籤上，找出 source 參數的「值」儲存格，並輸入字串以指定下列其中一項：
   - 視訊檔的本機路徑

- 視訊檔的 URL
- 描述如何播放視訊檔之「同步化多媒體整合語言」(Synchronized Multimedia Integration Language，SMIL) 檔案的 URL

  如需有關如何建立 SMIL 檔以描述一個或多個 FLV 檔的詳細資訊，請參閱第 143 頁「使用 SMIL 檔」。

4 當「舞台」上的 FLVPlayback 組件已選取的情況下，在「組件檢測器」的「參數」索引標籤上，按一下 skin 參數的「值」儲存格。

5 按一下放大鏡圖示以開啟「選取外觀元素」對話方塊。

6 選取下列其中一個選項：

- 從下拉式「外觀元素」清單中，選取其中一個預先設計的外觀元素，將一組播放控制項附加到組件上。
- 如果您建立了自訂的外觀元素，請從彈出式選單中選取「自訂外觀元素 URL」，然後在「URL」方塊中輸入包含該外觀元素之 SWF 檔的 URL。
- 選取「無」，並將個別 FLV 播放自訂 UI 組件拖曳至「舞台」，即可加入播放控制項。

  備註：在前兩種情況下，外觀元素的預覽畫面都會出現在彈出式選單上方的檢視窗格中。您可以使用「顏色挑選器」來變更外觀元素的顏色。

  如果要變更自訂使用者介面控制項的顏色，您必須使用自訂的方式。如需有關使用自訂使用者介面控制項的詳細資訊，請參閱第 136 頁「個別設定 FLV 播放自訂 UI 組件的外觀元素」。

7 按一下「確定」，關閉「選取外觀元素」對話方塊。

8 選取「控制 > 測試影片」，以執行 SWF 檔並啟動視訊。

下列程序會使用「視訊匯入」精靈來加入 FLVPlayback 組件：

使用視訊匯入精靈：

1 選取「檔案 > 匯入 > 匯入視訊」。

2 選取下列其中一個選項以指定視訊檔的位置：

- 在電腦中
- 已經部署至網站伺服器、Flash 視訊串流服務或 Flash Media Server

3 請視您的選擇，輸入指定視訊檔位置的路徑或 URL，然後按「下一步」。

4 如果選取了檔案路徑，接著便會出現「部署」對話方塊，您可以在這個方塊中選取其中一個選項，以指定部署視訊的方式：

- 從網站伺服器漸進式下載
- Flash 視訊串流服務的串流
- Flash Media Server 的串流
- 在 SWF 檔中內嵌視訊並在時間軸中播放

  重要事項：請勿選取「內嵌視訊」選項。FLVPlayback 組件只能播放外部串流視訊。這個選項不會將 FLVPlayback 組件放在「舞台」上。

5 按「下一步」。

6 選取下列其中一個選項：

- 從下拉式「外觀元素」清單中，選取其中一個預先設計的外觀元素，將一組播放控制項附加到組件上。
- 如果您已經為組件建立了自訂的外觀元素，請從彈出式選單中選取「自訂外觀元素 URL」，然後在「URL」方塊中輸入包含該外觀元素之 SWF 檔的 URL。
- 選取「無」，並將個別 FLV 播放自訂 UI 組件拖曳至「舞台」，即可加入播放控制項。

備註：在前兩種情況下，外觀元素的預覽畫面都會出現在彈出式選單上方的檢視窗格中。

7 按一下 「確定」，關閉 「選取外觀元素」對話方塊。

8 請閱讀 「完成視訊匯入」對話方塊以瞭解接下來會發生的事，然後按 「完成」。

9 如果您尚未儲存 FLA 檔，便會出現 「另存新檔」對話方塊。

10 選取 「控制 > 測試影片」，以執行 SWF 並啟動視訊。

下列程序會使用 ActionScript 來加入 FLVPlayback 組件。

使用 **ActionScript** 動態地建立實體：

1 將 FLVPlayback 組件從 「組件」面板拖曳到 「元件庫」面板 (「視窗 > 元件庫」)。

2 在時間軸的 「影格 1」的 「動作」面板中，加入下列程式碼。將 install_drive 變更為 Flash 安裝所在的磁碟機，並修改路徑以反映安裝的 Skins 資料夾位置：

在 Windows 電腦上：

```
import fl.video.*;
var my_FLVPlybk = new FLVPlayback();
my_FLVPlybk.x = 100;
my_FLVPlybk.y = 100;
addChild(my_FLVPlybk);
my_FLVPlybk.skin = "file:///install_drive|/Program Files/Adobe/Adobe Flash
CS5/en/Configuration/FLVPlayback Skins/ActionScript 3.0/SkinOverPlaySeekMute.swf"
my_FLVPlybk.source = "http://www.helpexamples.com/flash/video/water.flv";
```

在 Macintosh 電腦上：

```
import fl.video.*;
var my_FLVPlybk = new FLVPlayback();
my_FLVPlybk.x = 100;
my_FLVPlybk.y = 100;
addChild(my_FLVPlybk);
my_FLVPlybk.skin = "file:///Macintosh HD:Applications:Adobe Flash CS5:Configuration:FLVPlayback
Skins:ActionScript 3.0SkinOverPlaySeekMute.swf"
my_FLVPlybk.source = "http://www.helpexamples.com/flash/video/water.flv";
```

備註：如果尚未設定 source 和 skin 屬性，產生的影片片段將會顯示空白。

3 選取 「控制 > 測試影片」，以執行 SWF 檔並啟動視訊檔。

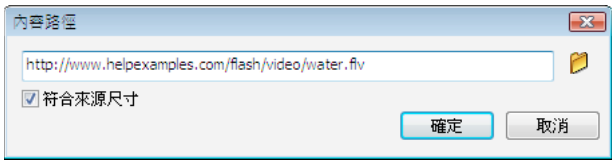
FLVPlayback 組件參數

您可以在 「組件檢測器」或 「屬性」檢測器中，為 FLVPlayback 組件的每個實體設定下列參數：align、autoPlay、cuePoints、preview、scaleMode、skin、skinAutoHide、skinBackgroundAlpha、skinBackgroundColor、source 和 volume。這些參數都具有相對應的 ActionScript 同名屬性。當您指定這些參數的值時，就是設定屬性在應用程式中的起始狀態。在 ActionScript 中設定屬性會覆寫您設定給參數的值。如需有關這些參數可能值的詳細資訊，請參閱 適用於 Adobe Flash Platform 的 ActionScript 3.0 參考中的 FLVPlayback 類別。

指定 FLVPlayback source 參數

source 參數可讓您指定視訊檔的名稱和位置，這兩者會告知 Flash 要以何種方式播放檔案。

在 「組件檢測器」中按兩下 source 參數的 「值」儲存格，以開啟 「內容路徑」對話方塊。



FLVPlayback「內容路徑」對話方塊

「內容路徑」對話方塊提供了「符合來源 FLV 尺寸」核取方塊，可指定「舞台」上的 FLVPlayback 實體是否應該符合來源視訊檔的尺寸。來源視訊檔含有播放時偏好的高度和寬度尺寸。如果選取這個選項，FLVPlayback 實體的尺寸就會調整為符合偏好的尺寸。

來源
請輸入視訊檔或用於描述如何播放視訊檔之 XML 檔的 URL 或本機路徑。如果您不知道視訊檔的確切位置，請按一下資料夾圖示來開啟「瀏覽器」對話方塊，以便尋找正確的位置。瀏覽視訊檔時，若檔案位在目標 SWF 檔的同一層或下方，Flash 便會自動將路徑設成相對於該層位置，讓您能從網站伺服器提供播放。如果不是這個情況，那麼路徑就是絕對的 Windows 或 Macintosh 路徑。若要輸入本機 XML 檔的名稱，請輸入其路徑和名稱。

如果您指定 HTTP URL，視訊檔就會以漸進式下載方式播放。如果您指定的 URL 是 RTMP URL，視訊檔則會從 Flash Media Server 或 FVSS 進行串流處理。XML 檔的 URL 也可以是來自 Flash Media Server 或 FVSS 的串流視訊檔。

重要事項：

您也可以指定 SMIL 檔的位置，使用該檔案來描述如何在多種頻寬下播放多個視訊檔串流。此 SMIL 檔使用「同步化多媒體整合語言」(Synchronized Multimedia Integration Language，SMIL) 來描述 FLV 檔。如需 SMIL 檔的說明，請參閱第 143 頁「使用 SMIL 檔」。

您也可以使用 ActionScript FLVPlayback.source 屬性以及 FLVPlayback.play() 和 FLVPlayback.load() 方法來指定視訊檔的名稱和位置。這三種替代方法優先於「組件檢測器」中的 source 參數。有關詳細資訊，請參閱 適用於 Adobe Flash Platform 的 ActionScript 3.0 參考中 FLVPlayback 類別的 FLVPlayback.source、FLVPlayback.play() 以及 FLVPlayback.load() 項目。

全螢幕支援

FLVPlayback 組件的 ActionScript 3.0 版所支援的全螢幕模式需要使用 Flash Player 9.0.28.0 版或更新版本，同時也必須正確設定用於全螢幕檢視的 HTML。某些預先設計的外觀元素中含有可以切換全螢幕模式開和關的切換按鈕。FullScreenButton 圖示位於下列圖例的右側控制列。


控制列上的全螢幕圖示

只有當 fullScreenTakeOver 屬性設為 true (這是預設值) 時，才支援全螢幕。

無論是否支援硬體加速都能支援全螢幕。如需硬體加速支援的詳細資訊，請參閱第 125 頁「硬體加速」。

實作 **FLVPlayback** 全螢幕支援：

1. 將 FLVPlayback 組件加入應用程式中，並為組件指定視訊檔。
2. 為 FLVPlayback 組件選取一個具有全螢幕按鈕的外觀元素 (例如 SkinUnderPlaySeekFullscreen.swf)，或從「組件」面板的「視訊」區段將 FullScreenButton 使用者介面組件加入到 FLVPlayback 組件。
3. 選取「檔案 > 發佈設定」。
4. 在「發佈設定」對話方塊中，按一下「HTML」索引標籤。
5. 在「HTML」索引標籤上，從「範本」彈出式選單中選取「含有全螢幕支援的 Flash」。

6 同時，在「HTML」索引標籤上選取「偵測 Flash 版本」核取方塊，並依您所使用的 Flash Player 版本，指定 9.0.28 版或更新版本。

7 選取「格式」索引標籤，並確認已選取「Flash (.swf)」和「HTML (.html)」您可以置換預設的檔案名稱。

8 按一下「發佈」，然後按一下「確定」。

您也可以略過步驟 7，直接按「確定」再選取「檔案 > 發佈預覽 > 預設 - (HTML)」，自動使用您的預設瀏覽器開啟匯出的 HTML 檔。若未略過該步驟，則請啟動您的瀏覽器並開啟匯出的 HTML 檔以測試全螢幕選項。

如果要將支援全螢幕的 FLVPlayback 組件加入您的網頁，請開啟匯出的 HTML 檔，並將內嵌 SWF 檔的程式碼複製到網頁的 HTML 檔中。這段程式碼看起來會與下列範例類似：

```
//from the <head> section

<script language="javascript"> AC_FL_RunContent = 0; </script>
<script language="javascript"> DetectFlashVer = 0; </script>
<script src="AC_RunActiveContent.js" language="javascript"></script>
<script language="JavaScript" type="text/javascript">
<!--
// -----------------------------------------------------------------------------
// Globals
// Major version of Flash required
var requiredMajorVersion = 9;
// Minor version of Flash required
var requiredMinorVersion = 0;
// Revision of Flash required
var requiredRevision = 28;
// -----------------------------------------------------------------------------
// -->
</script>

//and from the <body> section

<script language="JavaScript" type="text/javascript">
<!--
if (AC_FL_RunContent == 0 || DetectFlashVer == 0) {
    alert("This page requires AC_RunActiveContent.js.");
} else {
    var hasRightVersion = DetectFlashVer(requiredMajorVersion,
        requiredMinorVersion, requiredRevision);
    if(hasRightVersion) { // if we&apos;ve detected an acceptable version
        // embed the Flash movie
        AC_FL_RunContent(
            &apos;codebase&apos;, &apos;http://download.macromedia.com/pub/
                shockwave/cabs/flash/swflash.cab#version=9,0,28,0&apos;,
            &apos;width&apos;, &apos;550&apos;,
            &apos;height&apos;, &apos;400&apos;,
            &apos;src&apos;, &apos;fullscreen&apos;,
            &apos;quality&apos;, &apos;high&apos;,
            &apos;pluginspage&apos;, &apos;http://www.macromedia.com/go/
                getflashplayer&apos;,
            &apos;align&apos;, &apos;middle&apos;,
            &apos;play&apos;, &apos;true&apos;,
            &apos;loop&apos;, &apos;true&apos;,
            &apos;scale&apos;, &apos;showall&apos;,
            &apos;wmode&apos;, &apos;window&apos;,
            &apos;devicefont&apos;, &apos;false&apos;,
            &apos;id&apos;, &apos;fullscreen&apos;,
            &apos;bgcolor&apos;, &apos;#ffffff&apos;,
            &apos;name&apos;, &apos;fullscreen&apos;,
            &apos;menu&apos;, &apos;true&apos;,
            &apos;allowScriptAccess&apos;,&apos;sameDomain&apos;,
```

```
            &apos;allowFullScreen&apos;,&apos;true&apos;,
            &apos;movie&apos;, &apos;fullscreen&apos;,
            &apos;salign&apos;, &apos;&apos; ); //end AC code
    } else { // Flash is too old or we can&apos;t detect the plug-in.
        var alternateContent = &apos;Alternative HTML content should be placed
                here.&apos;
            + &apos;This content requires Adobe Flash Player.&apos;
            + &apos;<a href=http://www.macromedia.com/go/getflash/>Get Flash</a>
                    &apos;;
        document.write(alternateContent); // Insert non-Flash content.
    }
}
// -->
</script>
<noscript>
    // Provide alternative content for browsers that do not support scripting
    // or for those that have scripting disabled.
    Alternative HTML content should be placed here. This content requires Adobe Flash Player.
    <a href="http://www.macromedia.com/go/getflash/">Get Flash</a>
</noscript>
```

或者，您也可以使用匯出的 HTML 檔當做網頁的範本，再自行加入其它內容。不過，如果您要這麼做，請記得更改 HTML 檔的名稱，以避免日後再次從 Flash 匯出 FLVPlayback HTML 檔時，意外地覆寫了該檔案。

無論採用何種方式，您都必須將與 HTML 檔匯出至相同資料夾的 AC_RunActiveContent.js 檔一併上傳到網站伺服器。

ActionScript 對全螢幕模式的支援包括 fullScreenBackgroundColor、fullScreenSkinDelay 和 fullScreenTakeOver 屬性以及 enterFullScreenDisplayState() 方法。如需有關這些 ActionScript 元素的詳細資訊，請參閱 適用於 Adobe Flash Platform 的 ActionScript 3.0 參考。

使用 enterFullScreenDisplayState()

您也可以呼叫 ActionScript enterFullScreenDisplayState() 方法叫用全螢幕模式，如下列範例所示。

```
function handleClick(e:MouseEvent):void {
    myFLVPlybk.enterFullScreenDisplayState();
}
myButton.addEventListener(MouseEvent.CLICK, handleClick);
```

此範例「並非」透過按一下 FLVPlayback 外觀元素上的全螢幕切換按鈕叫用全螢幕模式，而是按一下網頁建立者所加入用來叫用全螢幕模式的 MyButton 按鈕。按一下該按鈕會觸發 handleClick 事件處理常式，進而呼叫 enterFullScreenDisplayState() 方法。

enterFullScreenDisplayState() 方法會將 Stage.displayState 屬性設定為 StageDisplayState.FULL_SCREEN，因此與 displayState 屬性具有相同的限制。如需有關 enterFullScreenDisplayState() 方法和 Stage.displayState 屬性的詳細資訊，請參閱 適用於 Adobe Flash Platform 的 ActionScript 3.0 參考。

硬體加速

Flash Player 9.0.115.0 及更新版本包含的程式碼可利用現有視訊硬體來改善 FLVPlayback 以全螢幕模式播放 FLV 檔的效能和真實度。只要符合先決條件且 fullScreenTakeOver 屬性是設定為 true，Flash Player 就會使用硬體加速來縮放視訊檔，而非透過軟體進行縮放。如果 FLVPlayback 組件是在較早版本的 Flash Player 中執行，或者未能符合硬體加速的先決條件，Flash Player 則會沿用以往的作法，將視訊檔本身放大。

為了充分運用全螢幕支援下的硬體加速優點，您的電腦必須有相容於 DirectX 7 的視訊卡且 VRAM (視訊記憶體) 為 4 MB 以上。這類硬體支援適用於 Windows 2000 或 Mac OS X 10.2 及其後續版本的作業系統。Direct X® 所提供的 API 構成軟體與視訊硬體之間的介面，可對三維和二維圖像及其它物件進行加速處理。

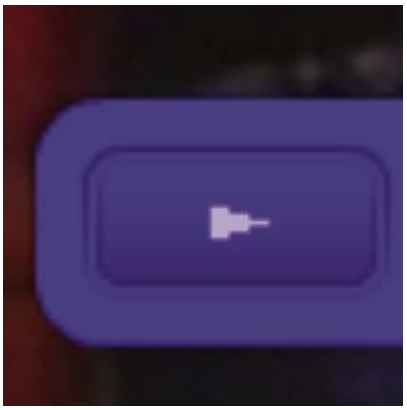
若要利用硬體加速模式，您還必須透過下列任一方法叫用全螢幕模式：

- 使用 FLVPlayback 外觀元素上的全螢幕切換按鈕

- 使用 FullScreenButton 視訊控制項
- 使用 ActionScript enterFullScreenDisplayState() 方法。如需詳細資訊，請參閱第 125 頁 「使用 enterFullScreenDisplayState()」。

如果您透過將 Stage.displayState 屬性設定為 StageDisplayState.FULLSCREEN 以叫用全螢幕模式，FLVPlayback 就不會使用硬體加速，即使視訊硬體和記憶體符合先決條件也一樣。

使用全螢幕支援下的硬體加速時，會導致 FLVPlayback 外觀元素隨著視訊播放程式及視訊檔一起縮放。下圖顯示 FLVPlayback 外觀元素在全螢幕模式下搭配硬體加速的效果，此為佔滿整個螢幕解析度的細部畫面。



在 1600 x 1200 監視器上以全螢幕模式顯示 320x240 像素視訊

這個影像顯示了在 1600 x 1200 的監視器上，使用全螢幕模式播放高度和寬度各為 320 和 240 (即 FLVPlayback 預設尺寸) 的視訊檔結果。如果換成尺寸較小的 FLV 檔或解析度較高的監視器，扭曲情形將更為顯著。反之，若將 FLV 檔的尺寸加大或縮減螢幕解析度，扭曲情形就不會那麼明顯。例如，從 640 x 480 變更為 1600 x 1200 依然會增加外觀元素的大小，卻可使視訊扭曲程度降低。

您可以設定 skinScaleMaximum 屬性來限制 FLVPlayback 外觀元素的縮放。預設值為 4.0，也就是 400%。不過，限制外觀元素的縮放需要結合硬體和軟體縮放處理 FLV，可能會對採用高位元速率編碼的較大尺寸 FLV 效能產生不良的影響。如果是大型尺寸的視訊 (例如，寬度 640 像素以上，高度 480 像素以上)，就不應該將 skinScaleMaximum 設定為較小的值，因為這樣可能導致在大型監視器上出現嚴重的效能問題。skinScaleMaximum 屬性可讓您針對大型外觀元素的效能、品質和外觀做取捨，以求得三者之間的平衡。

結束全螢幕模式

若要結束全螢幕模式，請再按一次全螢幕按鈕或按 Esc 鍵。

設定下列屬性及呼叫下列方法可能造成版面變更，進而導致 FLVPLayback 組件結束全螢幕模式：height、registrationHeight、registrationWidth、registrationX、registrationY、scaleX、scaleY、width、x、y、setScale() 或 setSize()。

一旦您設定 align 或 scaleMode 屬性，FLVPlayback 便會將其分別設定為 center 和 maintainAspectRatio 直到結束全螢幕模式為止。

使用全螢幕時，若將 fullScreenTakeOver 屬性的值從 true 變更為 false，硬體加速模式也會導致 Flash 結束全螢幕模式。

配置播放多個視訊檔的對齊方式

ActionScript 3.0 FLVPlayback 的 align 屬性可以指定當視訊檔調整大小後或放到組件的上方、下方、左側或右側時是否要置中對齊。除了組件的 x、y、width 和 height 屬性以外，ActionScript 3.0 組件也具有 registrationX、registrationY、registrationWidth 和 registrationHeight 屬性。一開始這些屬性會符合 x、y、width 和 height 屬性。載入後續的視訊檔時，自動

重新配置不會變更這些屬性，因此新的視訊檔可以在同一個位置上置中對齊。如果 scaleMode = VideoScaleMode.MAINTAIN_ASPECT_RATIO，後續載入的 FLV 檔就可以符合組件的原始尺寸，不需要改變組件的寬度和高度。

自動播放漸進式下載的視訊檔

載入漸進式下載的視訊檔時，必須等到下載的資料量足夠從頭到尾播放視訊檔後，FLVPlayback 才會開始播放視訊檔。

如果要在尚未下載足夠的資料量之前就開始播放視訊檔，請呼叫 play() 方法 (不使用任何參數)。

如果要回到等待下載足夠視訊檔資料量的狀態，請呼叫 pause() 方法，然後呼叫 playWhenEnoughDownloaded() 方法。

使用提示點

提示點是指在視訊檔播放的同時，視訊播放程式傳送 cuePoint 事件的點。您可以在 FLV 檔中加入提示點，讓網頁上的其它元素在這些時間點發生某個動作。例如，您可能想要顯示文字或圖形或同步化 Flash 動畫，或者透過暫停 FLV 檔、搜尋視訊中的不同時間點或切換至不同的 FLV 檔，以影響 FLV 檔的播放。提示點可讓您經由 ActionScript 程式碼接手控制權，並將 FLV 檔中的時間點與網頁上的其它動作同步化。

提示點共有三種：瀏覽、事件和 ActionScript。瀏覽和事件提示點也稱為 「內嵌」提示點，因為這兩種提示點都內嵌於 FLV 檔串流和 FLV 檔的中繼資料封包。

「瀏覽提示點」會在 FLV 檔內建立最接近您所指定之時間的關鍵影格，讓您可以在 FLV 檔中搜尋特定影格。「關鍵影格」是 FLV 檔串流中影像影格之間出現的資料段落。當您搜尋瀏覽提示點時，組件便會搜尋關鍵影格並啟動 cuePoint 事件。

「事件提示點」讓您能夠將 FLV 檔內的某個時間點與網頁上的外部事件同步化。cuePoint 事件會精確地發生在特定時間。您可以利用 「視訊匯入」精靈或 Flash Video Encoder，將瀏覽和事件提示點內嵌到 FLV 檔中。如需有關 「視訊匯入」精靈和 Flash Video Encoder 的詳細資訊，請參閱 「使用 Flash」中的第 16 章 「使用視訊」。

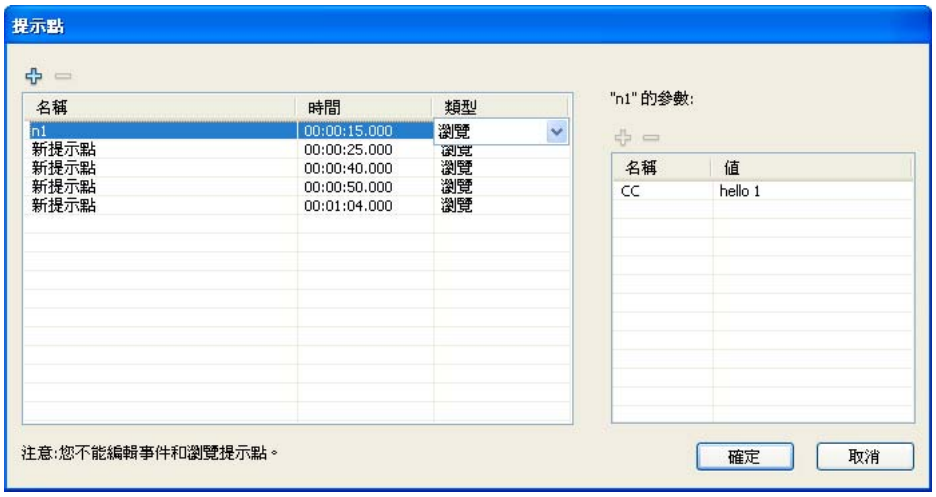
「ActionScript 提示點」是外部提示點，您可以透過組件的 「Flash 視訊提示點」對話方塊或 FLVPlayback.addASCuePoint() 方法加入。組件會將 ActionScript 提示點儲存在 FLV 檔以外的地方，然後再進行追蹤，因此，這些提示點不如內嵌提示點般精確。ActionScript 提示點的精確度為十分之一秒。由於組件會在播放磁頭更新時產生 ActionScript 提示點 cuePoint 事件，因此，您可藉由降低 playheadUpdateInterval 屬性的值，提高 ActionScript 提示點的精確度。如需詳細資訊，請參閱 Adobe® Flash® Professional CS5 的 ActionScript® 3.0 參考中的 FLVPlayback.playheadUpdateInterval 屬性。

在 ActionScript 和 FLV 檔的中繼資料內，提示點會以具有下列屬性的物件來表示：name、time、type 和 parameters。name 屬性是包含所指定之提示點名稱的字串。time 屬性是以小時、分鐘、秒和毫秒 (HH:MM:SS.mmm) 表示提示點發生時間的數字。type 屬性是根據您所建立的提示點類型而定，具有 "navigation"、"event" 或 "actionscript" 值的字串。parameters 屬性是所指定之名稱和值配對的陣列。

當 cuePoint 事件發生時，透過 info 屬性便可從事件物件中取用提示點物件。

使用 Flash 視訊提示點對話方塊

在 「組件檢測器」中按兩下 cuePoints 參數的 「值」儲存格，以開啟 「Flash 視訊提示點」對話方塊。這個對話方塊看起來像下圖：



「Flash 視訊提示點」對話方塊

對話方塊中會顯示內嵌提示點和 ActionScript 提示點。您可以使用這個對話方塊來新增和刪除 ActionScript 提示點以及提示點參數，也可以啟用或停用內嵌提示點。不過，您無法新增、變更或刪除內嵌提示點。

新增 **ActionScript** 提示點：

1. 在「組件檢測器」中按兩下 cuePoints 參數的值儲存格，以開啟「Flash 視訊提示點」對話方塊。
2. 在提示點清單上方，按一下左上角的加號 (+)，以新增預設 ActionScript 提示點項目。
3. 按一下「名稱」欄中的「新提示點」文字並加以編輯，為提示點命名。
4. 按一下 00:00:00:000 的「時間」值並加以編輯，指定提示點發生的時間。您可以使用小時、分鐘、秒和毫秒 (HH:MM:SS.mmm) 來指定時間。

   如果有多個提示點，對話方塊就會按照時間優先順序將新的提示點放入清單中。
5. 若要為選取的提示點新增參數，請按一下「參數」區域上方的加號 (+)，並在「名稱」和「值」欄中輸入值。針對每個參數重複執行這個步驟。
6. 若要新增更多 ActionScript 提示點，請針對每個提示點重複執行步驟 2 到 5。
7. 按一下「確定」儲存變更。

刪除 **ActionScript** 提示點：

1. 在「組件檢測器」中按兩下 cuePoints 參數的值儲存格，以開啟「Flash 視訊提示點」對話方塊。
2. 選取您要刪除的提示點。
3. 在提示點清單上方，按一下左上角的減號 (-) 加以刪除。
4. 針對每個要刪除的提示點重複執行步驟 2 和 3。
5. 按一下「確定」儲存變更。

啟用或停用內嵌 **FLV** 檔的提示點：

1. 在「組件檢測器」中按兩下 cuePoints 參數的值儲存格，以開啟「Flash 視訊提示點」對話方塊。
2. 選取您要啟用或停用的提示點。
3. 按一下「類型」欄中的值以觸發彈出式選單，或按一下向下箭頭。
4. 按一下提示點類型名稱 (例如「事件」或「瀏覽」) 加以啟用。按一下「停用」加以停用。

5 按一下「確定」儲存變更。

透過 ActionScript 使用提示點

您可以利用 ActionScript 來新增 ActionScript 提示點、偵聽 cuePoint 事件、尋找任何類型或特定類型的提示點、搜尋瀏覽提示點、啟用或停用提示點、確認提示點是否已啟用以及移除提示點。

本節中的範例使用名為 cuepoints.flv 的 FLV 檔，這個檔案包含下列三個提示點：

名稱	時間	類型
point1	00:00:00.418	瀏覽
point2	00:00:07.748	瀏覽
point3	00:00:16.020	瀏覽

新增 ActionScript 提示點

您可以使用 addASCuePoint() 方法，在 FLV 檔中新增 ActionScript 提示點。下列範例會在已準備要播放的 FLV 檔中新增兩個 ActionScript 提示點。此範例使用提示點物件來新增第一個提示點，並透過物件屬性指定提示點的時間、名稱和類型。第二個呼叫會使用方法的 time 和 name 參數來指定時間和名稱。

```
// Requires an FLVPlayback instance called my_FLVPlybk on Stage
import fl.video.*;
import fl.video.MetadataEvent;
my_FLVPlybk.source = "http://www.helpexamples.com/flash/video/cuepoints.flv"
var cuePt:Object = new Object(); //create cue point object
cuePt.time = 2.02;
cuePt.name = "ASpt1";
cuePt.type = "actionscript";
my_FLVPlybk.addASCuePoint(cuePt);//add AS cue point
// add 2nd AS cue point using time and name parameters
my_FLVPlybk.addASCuePoint(5, "ASpt2");
```

如需詳細資訊，請參閱 適用於 Adobe Flash Platform 的 ActionScript 3.0 參考中的 FLVPlayback.addASCuePoint() 方法。

偵聽 cuePoint 事件

cuePoint 事件可讓您在 cuePoint 事件發生時，經由 ActionScript 程式碼接手控制權。當下列範例中的提示點發生時，cuePoint 偵聽程式會呼叫事件處理常式函數，以顯示 playheadTime 屬性的值以及提示點的名稱和類型。將此範例與上一節「新增 ActionScript 提示點」中的範例搭配使用，即可查看結果。

```
my_FLVPlybk.addEventListener(MetadataEvent.CUE_POINT, cp_listener);
function cp_listener(eventObject:MetadataEvent):void {
        trace("Elapsed time in seconds: " + my_FLVPlybk.playheadTime);
        trace("Cue point name is: " + eventObject.info.name);
        trace("Cue point type is: " + eventObject.info.type);
}
```

如需有關 cuePoint 事件的詳細資訊，請參閱 適用於 Adobe Flash Platform 的 ActionScript 3.0 參考中的 FLVPlayback.cuePoint 事件。

尋找提示點

您可以使用 ActionScript 來尋找任何類型的提示點、最接近某個時間的提示點或具有特定名稱的下一個提示點。

下列範例中的 ready_listener() 事件處理常式會呼叫 findCuePoint() 方法來尋找提示點 ASpt1，接著呼叫 findNearestCuePoint() 方法來尋找最接近提示點 ASpt1 時間的瀏覽提示點：

```
import fl.video.FLVPlayback;
import fl.video.CuePointType;
import fl.video.VideoEvent;
my_FLVPlybk.source = "http://www.helpexamples.com/flash/video/cuepoints.flv"
var rtn_obj:Object; //create cue point object
my_FLVPlybk.addASCuePoint(2.02, "ASpt1");//add AS cue point
function ready_listener(eventObject:VideoEvent):void {
    rtn_obj = my_FLVPlybk.findCuePoint("ASpt1", CuePointType.ACTIONSCRIPT);
    traceit(rtn_obj);
    rtn_obj = my_FLVPlybk.findNearestCuePoint(rtn_obj.time, CuePointType.NAVIGATION);
    traceit(rtn_obj);
}
my_FLVPlybk.addEventListener(VideoEvent.READY, ready_listener);
function traceit(cuePoint:Object):void {
    trace("Cue point name is: " + cuePoint.name);
    trace("Cue point time is: " + cuePoint.time);
    trace("Cue point type is: " + cuePoint.type);
}
```

下列範例中的 ready_listener() 事件處理常式會尋找提示點 ASpt，並呼叫 findNextCuePointWithName() 方法尋找具有相同名稱的下一個提示點：

```
import fl.video.*;
my_FLVPlybk.source = "http://www.helpexamples.com/flash/video/cuepoints.flv"
var rtn_obj:Object; //create cue point object
my_FLVPlybk.addASCuePoint(2.02, "ASpt");//add AS cue point
my_FLVPlybk.addASCuePoint(3.4, "ASpt");//add 2nd Aspt
my_FLVPlybk.addEventListener(VideoEvent.READY, ready_listener);
function ready_listener(eventObject:VideoEvent):void {
    rtn_obj = my_FLVPlybk.findCuePoint("ASpt", CuePointType.ACTIONSCRIPT);
    traceit(rtn_obj);
    rtn_obj = my_FLVPlybk.findNextCuePointWithName(rtn_obj);
    traceit(rtn_obj);
}
function traceit(cuePoint:Object):void {
    trace("Cue point name is: " + cuePoint.name);
    trace("Cue point time is: " + cuePoint.time);
    trace("Cue point type is: " + cuePoint.type);
}
```

如需有關尋找提示點的詳細資訊，請參閱 適用於 Adobe Flash Platform 的 ActionScript 3.0 參考中的 FLVPlayback.findCuePoint()、FLVPlayback.findNearestCuePoint() 以及 FLVPlayback.findNextCuePointWithName() 方法。

搜尋瀏覽提示點

您可以搜尋瀏覽提示點、從特定時間搜尋下一個瀏覽提示點，以及從特定時間搜尋上一個瀏覽提示點。下列範例會播放 FLV 檔 cuepoints.flv，並在 ready 事件發生時，搜尋位於 7.748 的提示點。當 cuePoint 事件發生時，這個範例會呼叫 seekToPrevNavCuePoint() 方法來搜尋第一個提示點。在該 cuePoint 事件發生時，此範例會呼叫 seekToNextNavCuePoint() 方法，透過將 eventObject.info.time (目前提示點的時間) 增加 10 秒的方式，搜尋最後一個提示點。

```
import fl.video.*;

my_FLVPlybk.addEventListener(VideoEvent.READY, ready_listener);
function ready_listener(eventObject:Object):void {
    my_FLVPlybk.seekToNavCuePoint("point2");
}
my_FLVPlybk.addEventListener(MetadataEvent.CUE_POINT, cp_listener);
function cp_listener(eventObject:MetadataEvent):void {
    trace(eventObject.info.time);
    if(eventObject.info.time == 7.748)
        my_FLVPlybk.seekToPrevNavCuePoint(eventObject.info.time - .005);
    else
        my_FLVPlybk.seekToNextNavCuePoint(eventObject.info.time + 10);
}
my_FLVPlybk.source = "http://helpexamples.com/flash/video/cuepoints.flv";
```

如需詳細資訊，請參閱 適用於 Adobe Flash Platform 的 ActionScript 3.0 參考中的 FLVPlayback.seekToNavCuePoint()、FLVPlayback.seekToNextNavCuePoint() 以及 FLVPlayback.seekToPrevNavCuePoint() 方法。

啟用和停用內嵌 FLV 檔的提示點

您可以使用 setFLVCuePointEnabled() 方法，啟用和停用內嵌 FLV 檔的提示點。停用的提示點不會觸發 cuePoint 事件，也不會使用 seekToCuePoint()、seekToNextNavCuePoint() 或 seekToPrevNavCuePoint() 方法。不過，您還是可以利用 findCuePoint()、findNearestCuePoint() 和 findNextCuePointWithName() 方法來尋找停用的提示點。

您可以使用 isFLVCuePointEnabled() 方法來測試內嵌 FLV 檔的提示點是否已啟用。下列範例會在視訊準備好要播放時，停用內嵌提示點 point2 和 point3。不過，當第一個 cuePoint 事件發生時，事件處理常式會進行測試，以查看提示點 point3 是否已停用，若停用了，便會加以啟用。

```
import fl.video.*;
my_FLVPlybk.source = "http://www.helpexamples.com/flash/video/cuepoints.flv";
my_FLVPlybk.addEventListener(VideoEvent.READY, ready_listener);
function ready_listener(eventObject:VideoEvent):void {
    my_FLVPlybk.setFLVCuePointEnabled(false, "point2");
    my_FLVPlybk.setFLVCuePointEnabled(false, "point3");
}
my_FLVPlybk.addEventListener(MetadataEvent.CUE_POINT, cp_listener);
function cp_listener(eventObject:MetadataEvent):void {
        trace("Cue point time is: " + eventObject.info.time);
        trace("Cue point name is: " + eventObject.info.name);
        trace("Cue point type is: " + eventObject.info.type);
        if (my_FLVPlybk.isFLVCuePointEnabled("point2") == false) {
            my_FLVPlybk.setFLVCuePointEnabled(true, "point2");
        }
}
```

如需詳細資訊，請參閱 適用於 Adobe Flash Platform 的 ActionScript 3.0 參考中的 FLVPlayback.isFLVCuePointEnabled() 和 FLVPlayback.setFLVCuePointEnabled() 方法。

移除 ActionScript 提示點

您可以使用 removeASCuePoint() 方法來移除 ActionScript 提示點。下列範例會在提示點 ASpt1 發生時，移除提示點 ASpt2：

```
import fl.video.*;
my_FLVPlybk.source = "http://www.helpexamples.com/flash/video/cuepoints.flv"
my_FLVPlybk.addASCuePoint(2.02, "ASpt1");//add AS cue point
my_FLVPlybk.addASCuePoint(3.4, "ASpt2");//add 2nd Aspt
my_FLVPlybk.addEventListener(MetadataEvent.CUE_POINT, cp_listener);
function cp_listener(eventObject:MetadataEvent):void {
        trace("Cue point name is: " + eventObject.info.name);
        if (eventObject.info.name == "ASpt1") {
            my_FLVPlybk.removeASCuePoint("ASpt2");
            trace("Removed cue point ASpt2");
        }
}
```

如需詳細資訊，請參閱 適用於 Adobe Flash Platform 的 ActionScript 3.0 參考中的 FLVPlayback.removeASCuePoint()。

播放多個視訊檔

如果要由 FLVPlayback 實體連續播放多個視訊檔，只要在上一個視訊檔完成播放時，使用 source 屬性載入新的 URL 即可。例如，下列 ActionScript 程式碼會偵聽在視訊檔完成播放時所發生的 complete 事件。當這個事件發生時，程式碼會為 source 屬性設定新視訊檔的名稱和位置，並且呼叫 play() 方法來播放新的視訊。

```
import fl.video.*;
my_FLVPlybk.source = "http://www.helpexamples.com/flash/video/clouds.flv";
my_FLVPlybk.addEventListener(VideoEvent.COMPLETE, complete_listener);
// listen for complete event; play new FLV
function complete_listener(eventObject:VideoEvent):void {
    if (my_FLVPlybk.source == "http://www.helpexamples.com/flash/video/clouds.flv") {
        my_FLVPlybk.play("http://www.helpexamples.com/flash/video/water.flv");
    }
};
```

使用多個視訊播放程式

您也可以在 FLVPlayback 組件的單一實體內開啟多個視訊播放程式，以播放多個視訊並在播放的同時進行切換。

將 FLVPlayback 組件拖曳到「舞台」時，便會建立初始視訊播放程式。組件會自動指派數字 0 給初始視訊播放程式，讓它成為預設的播放程式。只要將 activeVideoPlayerIndex 屬性設定為新的數字，即可建立其它視訊播放程式。設定 activeVideoPlayerIndex 屬性也會使指定的視訊播放程式成為「作用中」視訊播放程式，而該視訊播放程式將會受 FLVPlayback 類別的屬性和方法影響。不過，設定 activeVideoPlayerIndex 屬性並不會將視訊播放程式顯示出來。若要讓視訊播放程式變成可見的狀態，請將 visibleVideoPlayerIndex 屬性設定為視訊播放程式的編號。如需有關這些屬性如何與 FLVPlayback 類別之方法和屬性互動的詳細資訊，請參閱 適用於 Adobe Flash Platform 的 ActionScript 3.0 參考中的 FLVPlayback.activeVideoPlayerIndex 與 FLVPlayback.visibleVideoPlayerIndex 屬性。

下列 ActionScript 程式碼會載入 source 屬性，以便在預設視訊播放程式中播放視訊檔，並為它新增提示點。當 ready 事件發生時，事件處理常式會將 activeVideoPlayerIndex 屬性設定為數字 1 以開啟第二個視訊播放程式，再為第二個視訊播放程式指定 FLV 檔和提示點，然後讓預設播放程式 (0) 再次成為現用的視訊播放程式。

```
/**
    Requires:
- FLVPlayback component on the Stage with an instance name of my_FLVPlybk
*/
// add a cue point to the default player
import fl.video.*;
my_FLVPlybk.source = "http://www.helpexamples.com/flash/video/clouds.flv";
my_FLVPlybk.addASCuePoint(3, "1st_switch");
my_FLVPlybk.addEventListener(VideoEvent.READY, ready_listener);
function ready_listener(eventObject:VideoEvent):void {
    // add a second video player and create a cue point for it
    my_FLVPlybk.activeVideoPlayerIndex = 1;
    my_FLVPlybk.source = "http://www.helpexamples.com/flash/video/water.flv";
    my_FLVPlybk.addASCuePoint(3, "2nd_switch");
    my_FLVPlybk.activeVideoPlayerIndex = 0;
};
```

若要在播放 FLV 檔時切換至其它 FLV 檔，您必須在 ActionScript 程式碼中進行切換。提示點可讓您利用 cuePoint 事件介入 FLV 檔中特定的時間點。下列程式碼會為 cuePoint 事件建立偵聽程式，並呼叫能夠暫停現用視訊播放程式 (0)、切換至第二個播放程式 (1) 及播放其 FLV 檔的處理常式函數：

```
import fl.video.*;
// add listener for a cuePoint event
my_FLVPlybk.addEventListener(MetadataEvent.CUE_POINT, cp_listener);
// add the handler function for the cuePoint event
function cp_listener(eventObject:MetadataEvent):void {
        // display the no. of the video player causing the event
        trace("Hit cuePoint event for player: " + eventObject.vp);
        // test for the video player and switch FLV files accordingly
        if (eventObject.vp == 0) {
            my_FLVPlybk.pause(); //pause the first FLV file
            my_FLVPlybk.activeVideoPlayerIndex = 1; // make the 2nd player active
            my_FLVPlybk.visibleVideoPlayerIndex = 1; // make the 2nd player visible
            my_FLVPlybk.play(); // begin playing the new player/FLV
        } else if (eventObject.vp == 1) {
            my_FLVPlybk.pause(); // pause the 2nd FLV
            my_FLVPlybk.activeVideoPlayerIndex = 0; // make the 1st player active
            my_FLVPlybk.visibleVideoPlayerIndex = 0; // make the 1st player visible
            my_FLVPlybk.play(); // begin playing the 1st player
        }
}
my_FLVPlybk.addEventListener(VideoEvent.COMPLETE, complete_listener);
function complete_listener(eventObject:VideoEvent):void {
        trace("Hit complete event for player: " + eventObject.vp);
        if (eventObject.vp == 0) {
            my_FLVPlybk.activeVideoPlayerIndex = 1;
            my_FLVPlybk.visibleVideoPlayerIndex = 1;
            my_FLVPlybk.play();
        } else {
            my_FLVPlybk.closeVideoPlayer(1);
        }
};
```

當您建立新的視訊播放程式時，FLVPlayback 實體會將視訊播放程式的屬性設定為預設視訊播放程式的值，但 source、totalTime 和 isLive 屬性除外，因為 FLVPlayback 實體永遠都會將這些屬性設定為預設值：分別為空字串、0 和 false。它會將預設視訊播放程式原本預設為 true 的 autoPlay 屬性設定為 false。cuePoints 屬性不會有任何作用，並且對預設視訊播放程式的後續載入動作也沒有作用。

控制音量、位置、尺寸、可見性和使用者介面控制項的方法和屬性永遠是通用的，而且其行為不會因為 activeVideoPlayerIndex 屬性的設定而受到影響。如需有關這些方法和屬性以及設定 activeVideoPlayerIndex 屬性影響的詳細資訊，請參閱 適用於 Adobe Flash Platform 的 ActionScript 3.0 參考中的 FLVPlayback.activeVideoPlayerIndex 屬性。其餘的屬性和方法會以 activeVideoPlayerIndex 屬性值所識別的視訊播放程式為目標。

不過，控制尺寸的屬性和方法仍將與 visibleVideoPlayerIndex 屬性互動。如需詳細資訊，請參閱 適用於 Adobe Flash Platform 的 ActionScript 3.0 參考中的 FLVPlayback.visibleVideoPlayerIndex 屬性。

從 Flash Media Server 串流處理 FLV 檔

從 Flash Media Server 串流處理 FLV 檔的需求各有不同，需視 Flash 視訊串流服務提供者是否提供本地頻寬偵測功能而定。本地頻寬偵測是指串流伺服器具有內建的頻寬偵測功能以提供更佳的播放效能。請向您的頻寬提供者查詢是否可以使用本地頻寬偵測功能。

若要存取 Flash Media Server 上的 FLV 檔，請使用 rtmp://my_servername/my_application/stream.flv 之類的 URL。

使用 Flash Media Server 播放即時串流時，必須將 FLVPlayback 的 isLive 屬性設定為 true。如需詳細資訊，請參閱 適用於 Adobe Flash Platform 的 ActionScript 3.0 參考中的 FLVPlayback.isLive 屬性。

如需有關管理 Flash Media Server (包括如何設定即時串流) 的詳細資訊，請參閱 Flash Media Server 說明文件，網址為 www.adobe.com/support/documentation/en/flashmediaserver/。

本地頻寬偵測或不使用頻寬偵測

NCManagerNative 類別是 NCManager 的子類別，支援部分 Flash 視訊串流服務提供者可能支援的本地頻寬偵測功能。使用 NCManagerNative 時不需要在 Flash Media Server 上另外安裝特別的檔案。NCManagerNative 也允許不使用 main.asc 檔 (不要求頻寬偵測) 連線至任何版本的 Flash Media Server。

如果要使用 NCManagerNative (而非預設的 NCManager 類別)，請將下列程式碼行加入到 FLA 檔的第一個影格：

```
import fl.video*;
VideoPlayer.iNCManagerClass = fl.video.NCManagerNative;
```

不使用本地頻寬偵測

如果 Flash 視訊串流服務提供者不支援本地頻寬偵測功能，而您需要使用頻寬偵測功能時，就必須將 main.asc 檔加入到 Flash Media Server FLV 應用程式。您可以在線上找到 main.asc 檔案，網址為 www.adobe.com/go/learn_fl_samples_tw。該檔案包含在 Samples.zip 檔中，位於 Samples\ComponentsAS2\FLVPlayback 目錄內。

設定 **Flash Media Server** 以串流處理 **FLV** 檔：

1. 在您的 Flash Media Server 應用程式資料夾中建立資料夾，並加以命名，例如 **my_application**。
2. 將 main.asc 檔複製到 my_application 資料夾。
3. 在 my_application 資料夾中建立名為 **streams** 的資料夾。
4. 在 streams 資料夾中建立名為 **_definst_** 的資料夾。
5. 將您的 FLV 檔放置在 **_definst_** 資料夾中。

自訂 FLVPlayback 組件

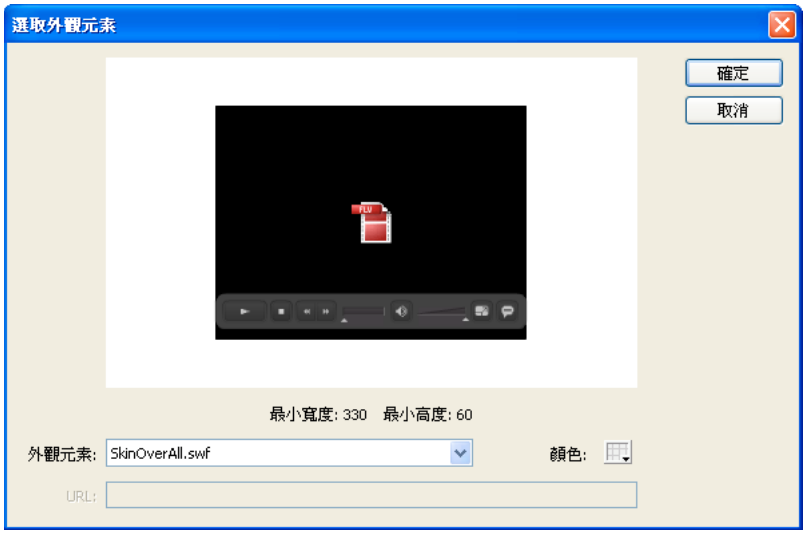
本節說明如何自訂 FLVPlayback 組件。不過，大部分用來自訂其它組件的方法並不適用於 FLVPlayback 組件。若要自訂 FLVPlayback 組件，您只能使用本節中所介紹的技巧。

您可以選用下列方式自訂 FLVPlayback 組件：選取預先設計的外觀元素、個別設定 FLV 播放自訂 UI 組件的外觀元素，或是建立新的外觀元素。您也可以使用 FLVPlayback 屬性來修改外觀元素的行為。

備註：您必須將外觀元素 SWF 檔以及應用程式的 SWF 檔上傳到網站伺服器，外觀元素才能與 FLVPlayback 組件搭配運作。

選取預先設計的外觀元素

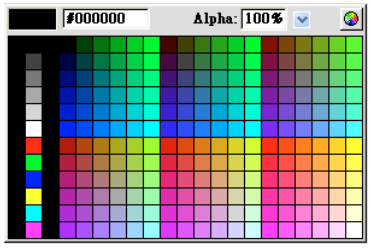
若要選取 FLVPlayback 組件的外觀元素，請在 「組件檢測器」中按一下 skin 參數的 value 儲存格。接著按一下放大鏡圖示來開啟 「選取外觀元素」對話方塊 (如下圖)，然後從中選取外觀元素或輸入 URL，以指定外觀元素 SWF 檔的位置。



FLVPlayback 「選取外觀元素」對話方塊

「外觀元素」彈出式選單中所列的外觀元素是位於 Flash 應用程式的 /Flash Configuration/FLVPlayback Skins/ActionScript 3.0 資料夾內。您可自行建立外觀元素，然後將 SWF 檔放入該資料夾，讓這個對話方塊提供更多新的外觀元素。出現在彈出式選單中的名稱具有 .swf 副檔名。如需有關建立外觀元素集的詳細資訊，請參閱第 140 頁 「建立新的外觀元素」。

只要您是藉由設定 skin 屬性的方式指定外觀元素，無論是在編寫期間設定 skin 參數或在執行階段使用 ActionScript 設定，您都可以指定任何所選外觀元素的顏色和 Alpha (透明度) 值。如果要在編寫期間指定顏色和 Alpha 值，請從 「選取外觀元素」對話方塊開啟 「顏色挑選器」，如下圖所示。



選取外觀元素對話方塊中的顏色挑選器

如果要選擇顏色，請按一下面板中的色票或在文字方塊中輸入數值。如果要選擇 Alpha 值，請使用滑動軸或在 Alpha 文字方塊中輸入一個百分比。

如果要在執行階段指定顏色和 Alpha 值，請設定 skinBackgroundColor 和 skinBackgroundAlpha 屬性。將 skinBackgroundColor 屬性設定為 0xRRGGBB (紅、綠、藍) 值。將 skinBackgroundAlpha 屬性設定為介於 0.0 和 1.0 之間的數字。下列範例是將 skinBackgroundColor 設定為 0xFF0000 (紅色) 並將 skinBackgroundAlpha 設定為 0.5。

```
my_FLVPlybk.skinBackgroundColor = 0xFF0000;
my_FLVPlybk.skinBackgroundAlpha = .5;
```

預設值是使用者上次選擇的值。

如果您想要使用 FLV 播放自訂 UI 組件來設定 FLVPlayback 組件的外觀元素，請從彈出式選單中選取 「無」。

個別設定 FLV 播放自訂 UI 組件的外觀元素

FLV 播放自訂 UI 組件可讓您自訂 FLV 檔中 FLVPlayback 控制項的外觀，並且讓您能夠在預覽網頁時查看結果。不過，這些組件並非設計成可供縮放。您應將影片片段及其內容編輯成特定大小。因此，最好的方式通常是將 「舞台」上的 FLVPlayback 組件調整成您所需要的大小，並將 scaleMode 設定為 exactFit。

如果要開始作業，只需從 「組件」面板拖曳想要的 FLV 播放自訂 UI 組件，並將這些組件放在 「舞台」上所需的位置，然後為每個組件提供實體名稱。

這些組件不需要使用任何 ActionScript 就可以運作。如果將它們放到與 FLVPlayback 組件相同的時間軸和影格，而沒有設定組件的外觀元素，FLVPlayback 組件將會自動與它們連接。如果 「舞台」上有多個 FLVPlayback 組件，或是自訂控制項和 FLVPlayback 實體不在相同的時間軸上，就需要使用 「動作」。

將組件放在 「舞台」之後，您可以依照編輯其它任何元件的方式編輯組件。一旦您開啟組件，將會看到每個組件的設定都稍有不同。

Button 組件

各種按鈕組件具有類似的結構。按鈕包括了 BackButton、ForwardButton、MuteButton、PauseButton、PlayButton、PlayPauseButton 和 StopButton。大部分按鈕的 「影格 1」上都有單一影片片段，影片片段的實體名稱為 placeholder_mc。這通常是按鈕的一般狀態實體，但不一定是如此。在 「影格 2」的 「舞台」上，每個顯示狀態都有四個影片片段：一般、滑入、按下和停用 (在執行階段，組件絕對不會真的進行到 「影格 2」，因為放置在這裡的這些影片片段，其用處是為了能夠更方便地進行編輯，並強迫這些影片片段載入至 SWF 檔，而不需選取 「元件屬性」對話方塊中的 「匯出在第一個影格」核取方塊。然而，您還是必須選取 「匯出給 ActionScript 使用」選項)。

若要設定按鈕的外觀元素，只要編輯這些影片片段即可。您可以變更它們的大小和外觀。

有些 ActionScript 通常會出現在 「影格 1」上。您不必變更這種 Script。其用意只是停止 「影格 1」上的播放磁頭，並指定哪些影片片段要用於何種狀態。

PlayPauseButton、MuteButton、FullScreenButton 和 CaptionButton 按鈕

PlayPauseButton、MuteButton、FullScreenButton 和 CaptionButton 按鈕的設定方式與其它按鈕不同；這些按鈕只有一個影格，而影格中有兩個圖層但沒有 Script。在這個影格上有兩個按鈕，其中一個按鈕位於另一個按鈕之上：PlayPauseButton 是使用 Play 和 Pause 按鈕，MuteButton 是使用 Mute-on 和 Mute-off 按鈕，FullScreenButton 是使用 full-screen-on 和 full-screen-off 按鈕，CaptionButton 是使用 caption-on 和 caption-off 按鈕。若要設定這些按鈕的外觀元素，請依照第 136 頁 「個別設定 FLV 播放自訂 UI 組件的外觀元素」所述，設定這兩個內部按鈕的外觀元素即可，不需執行其它動作。

CaptionButton 是用於 FLVPlaybackCaptioning 組件，因此必須附加至該組件而非 FLVPlayback 組件。

BackButton 和 ForwardButton 按鈕

BackButton 和 ForwardButton 按鈕的設定方式也與其它按鈕不同。這兩個按鈕的 「影格 2」上都有額外的影片片段，您可以在這一或兩個按鈕附近用來當做影格。這些影片片段並非絕對必要且沒有特殊功能，僅為便利讀者而提供。若要使用這些影片片段，只要從 「元件庫」面板將影片片段拖曳到 「舞台」上您想要的位置即可。如果不需要影片片段，可以選擇不要使用或是直接從 「元件庫」面板中刪除。

大部分提供的按鈕都是以共通的影片片段集合為基礎，因此您可以一次變更所有按鈕的外觀。您可以善加利用這個功能，或是置換共通的片段，讓每個按鈕的外觀都不一樣。

BufferingBar 組件

緩衝列組件相當簡單：這是由隨著組件進入緩衝狀態而變得可見的動畫所組成，並且不需要透過任何特殊的 ActionScript 加以設定。根據預設，它是從左向右移動、帶有矩形遮色片的條紋列，可以產生「理髮店招牌桿」(Barber Pole) 的特效，但是這項設定並無任何特殊之處。

雖然外觀元素 SWF 檔中的緩衝列由於必須在執行階段縮放而使用了 9 分割縮放，但 BufferingBar FLV 自訂 UI 組件因為具有巢狀影片片段，以致既不會也「無法」使用 9 分割縮放。如果您想要讓 BufferingBar 變得更寬或更高，可能就得變更其內容而非縮放組件。

SeekBar 和 VolumeBar 組件

儘管 SeekBar 和 VolumeBar 組件具有不同的功能，但仍有些類似。這兩種組件都有控制點、都使用相同的控制點追蹤機制，並且都支援巢狀的片段以追蹤進度和完整性。

FLVPlayback 組件中的 ActionScript 程式碼有許多地方都假設 SeekBar 或 VolumeBar 組件的註冊點 (也稱為「原點」或「零點」) 位於內容的左上角，因此，請務必維持這個慣例。否則，您的控制點以及進度和完整性影片片段可能發生問題。

雖然外觀元素 SWF 檔中的搜尋列由於必須在執行階段縮放而使用了 9 分割縮放，但 SeekBar FLV 自訂 UI 組件因為具有巢狀影片片段，以致既不會也「無法」使用 9 分割縮放。如果您想要讓 SeekBar 變得更寬或更高，可能就得變更其內容而非縮放組件。

控制點

控制點影片片段的實體是在「影格 2」上。和 BackButton 及 ForwardButton 組件一樣，組件絕對不會真的進行到「影格 2」；因為放置在這裡的這些影片片段，是為了能夠更方便地進行編輯，並強迫這些影片片段載入至 SWF 檔，而不需選取「元件屬性」對話方塊中的「匯出在第一個影格」核取方塊。然而，您還是必須選取「匯出給 ActionScript 使用」選項。

您可能會注意到，控制點影片片段的背景具有 Alpha 設為 0 的矩形。這個矩形會增加控制點作用區域的大小，以便更輕易地擷取，而不需要將它的外觀變更為類似按鈕的感應狀態。由於控制點是在執行階段動態建立的，所以一定是影片片段，而不是按鈕。這個 Alpha 設為 0 的矩形並非絕對必要，您通常可以使用任何想要的影像來取代控制點的內部。不過，最好應將註冊點置中對齊控制點影片片段的中央。

下列 ActionScript 程式碼位於 SeekBar 組件的「影格 1」，用來管理控制點：

```
stop();
handleLinkageID = "SeekBarHandle";
handleLeftMargin = 2;
handleRightMargin = 2;
handleY = 11;
```

由於「影格 2」的內容之故，必須呼叫 stop() 函數。

第二行指定要將哪個元件當做控制點；若您只是編輯「影格 2」上的控制點影片片段實體，就不需要變更這項設定。在執行階段，FLVPlayback 組件將於「舞台」上建立指定影片片段的實體成為 Bar 組件實體的同級實體，亦即兩者具有相同的父影片片段。因此，如果您的列位於根層級，則控制點一定也會在根層級。

變數 handleLeftMargin 會判斷控制點的原始位置 (0%)，而變數 handleRightMargin 則判斷控制點將在何處結束 (100%)。這些數字代表列控制項從左到右的位移，正數標示列內的周圍限制，而負數標示列的外圍限制。這些位移會根據控制點的註冊點，指定控制點可以到達的位置。如果您將註冊點放在控制點的中央，則控制點的最左邊和最右邊將會超出邊界。搜尋列影片片段的註冊點必須放在內容的左上角，才能正常地運作。

變數 handleY 會判斷控制點相對於列實體的 y 位置。這是根據每個影片片段的註冊點來決定。樣本控制點中的註冊點位於三角形的尖端，放在相對於可見部位的地方，與隱藏的感應狀態矩形無關。此外，列影片片段的註冊點必須保持在內容的左上角，才能正常地運作。

因此，根據這些限制，如果列控制項設定在 (100, 100) 位置，且其寬度為 100 像素，則控制點的範圍介於水平 102 至 198 像素，並垂直保持 111。如果您將 handleLeftMargin 和 handleRightMargin 變更為 -2，且將 handleY 變更為 –11，則控制點的範圍介於水平 98 至 202 像素，並垂直保持 89。

進度和完整性影片片段

SeekBar 組件具有「進度」影片片段，而 VolumeBar 具有「完整性」影片片段，但實際上，任何 SeekBar 或 VolumeBar 都可能具有其中一種影片片段，或是兩種影片片段都有或都沒有。就結構上來看，它們是相同的，且行為也很類似，但不同的是，它們追蹤不同的值。當 FLV 檔下載時，進度影片片段會填滿 (只能用於 HTTP 下載，因為如果從 FMS 進行串流處理，則會永遠是填滿的)，而當控制點從左向右移動時，完整性影片片段就會填滿。

FLVPlayback 組件會藉由尋找特定的實體名稱，以找出這些影片片段實體，因此，您的進度影片片段實體必須具有與父影片片段一樣的列影片片段，並且具有實體名稱 progress_mc。完整性影片片段實體則必須具有實體名稱 fullness_mc。

您可以設定內含或不含巢狀 fill_mc 影片片段實體的進度和完整性影片片段。VolumeBar fullness_mc 影片片段會顯示「具有」 fill_mc 影片片段的方法，而 SeekBar progress_mc 影片片段會顯示「沒有」 fill_mc 影片片段的方法。

當您想要使用不會縮放且不會扭曲外觀的填色時，具有巢狀 fill_mc 影片片段的方法會很有用。

在 VolumeBar fullness_mc 影片片段中，巢狀 fill_mc 影片片段實體會以遮色處理。您可以在建立影片片段時將它遮色，或是在執行階段動態建立遮色片。如果您使用影片片段做為遮色片，請將實體命名為 **mask_mc** 並加以設定，使 fill_mc 能夠如百分比為 100% 般顯示出來。如果您不遮色處理 fill_mc，動態建立的遮色片將會是矩形，且大小與 fill_mc 以 100% 顯示時的大小相同。

fill_mc 影片片段將以兩種遮色片方式之一呈現，視 fill_mc.slideReveal 為 true 或 false 而定。

如果 fill_mc.slideReveal 為 true，則 fill_mc 會從左向右移動，透過遮色片顯露。若百分比為 0%，便會向左移動，如此一來，透過遮色片便沒有東西可顯露。隨著百分比的增加，它會漸漸向右移動，直到百分比為 100%，再返回建立所在的「舞台」位置。

如果 fill_mc.slideReveal 為 false 或未定義 (預設行為)，則遮色片會從左向右重新調整大小，以呈現更多 fill_mc。當百分比為 0% 時，遮色片會水平縮放至 05，而隨著百分比增加，scaleX 會一直增加至 100%，此時便會呈現完整的 fill_mc。您不需要將 scaleX 設定為 100，因為 mask_mc 可能已經在建立時進行縮放。

沒有 fill_mc 的方法比具有 fill_mc 的方法來得簡易，但會水平扭曲填色。如果不希望出現扭曲，您必須使用 fill_mc。SeekBar progress_mc 會說明這個方法。

進度或完整性影片片段會依百分比進行水平縮放。百分比為 0% 時，實體的 scaleX 是設定為 0，讓它隱藏起來。隨著百分比增加，scaleX 會跟著調整到 100%，而片段的大小將與當初在「舞台」上建立時的大小相同。同樣地，您不需要將 scaleX 設定為 100，因為片段實體可能已經在建立時進行縮放。

連接您的 FLV 播放自訂 UI 組件

如果您將自訂 UI 組件放到與 FLVPlayback 組件相同的時間軸和影格，而沒有設定 skin 屬性，FLVPlayback 組件將會自動與它們連接，並不需要使用任何的 ActionScript。

如果「舞台」上有多個 FLVPlayback 組件，或是自訂控制項和 FLVPlayback 不在相同的時間軸上，就必須撰寫 ActionScript 程式碼將自訂 UI 組件連接到 FLVPlayback 組件的實體。首先，您必須為 FLVPlayback 實體指定一個名稱，然後使用 ActionScript 將 FLV 播放自訂 UI 組件實體指派給對應的 FLVPlayback 屬性。在下列範例中，FLVPlayback 實體是 my_FLVPlybk，句點 (.) 後面的部分是 FLVPlayback 屬性名稱，而 FLV 播放自訂 UI 控制項實體是在等號 (=) 的右邊：

```
//FLVPlayback instance = my_FLVPlybk
my_FLVPlybk.playButton = playbtn; // set playButton prop. to playbtn, etc.
my_FLVPlybk.pauseButton = pausebtn;
my_FLVPlybk.playPauseButton = playpausebtn;
my_FLVPlybk.stopButton = stopbtn;
my_FLVPlybk.muteButton = mutebtn;
my_FLVPlybk.backButton = backbtn;
my_FLVPlybk.forwardButton = forbtn;
my_FLVPlybk.volumeBar = volbar;
my_FLVPlybk.seekBar = seekbar;
my_FLVPlybk.bufferingBar = bufbar;
```

以下將會逐步建立自訂的 StopButton、PlayPauseButton、MuteButton 和 SeekBar 控制項：

1 將 FLVPlayback 組件拖曳到 「舞台」上，並賦予實體名稱 **my_FLVPlybk**。

2 透過 「組件檢測器」將 source 參數設定為 **http://www.helpexamples.com/flash/video/cuepoints.flv**。

3 將 Skin 參數設定為 「無」。

4 將 StopButton、PlayPauseButton 和 MuteButton 拖曳到 「舞台」並放在 FLVPlayback 實體之上，從左邊垂直堆放。在 「屬性」檢測器中，指定每個按鈕的實體名稱 (例如 **my_stopbttn**、**my_plypausbttn** 和 **my_mutebttn**)。

5 在 「元件庫」面板中，開啟 FLVPlayback Skins 資料夾，再開啟下方的 SquareButton 資料夾。

6 選取 SquareBgDown 影片片段，並在 「舞台」上按兩下加以開啟。

7 按一下滑鼠右鍵 (Windows) 或 Control+ 按一下 (Macintosh)，再從選單中選取 「全選」，然後刪除元件。

8 選取 「橢圓形」工具，在相同的位置繪製橢圓形，然後將填色設定為藍色 (**#0033FF**)。

9 在「屬性」檢測器中，將「寬度」(W:) 設為 **40**，將「高度」(H:) 設為 **20**。將 x 座標 (X:) 設為 **0.0**，將 y 座標 (Y:) 設為 **0.0**。

10 對 SquareBgNormal 重複執行步驟 6 到 8，但是將填色改為黃色 (**#FFFF00**)。

11 對 SquareBgOver 重複執行步驟 6 到 8，但是將填色改為綠色 (**#006600**)。

12 編輯按鈕內各種元件圖示的影片片段 (PauseIcon、PlayIcon、MuteOnIcon、MuteOffIcon 和 StopIcon)。您可以在 FLV Playback Skins/Label Button/Assets 底下的「元件庫」面板中找到這些影片片段 (其中 Label 是按鈕的名稱，例如 Play、Pause 等等)。為每個影片片段執行下列步驟：

   a 選取 「全選」選項。

   b 將顏色變更為紅色 (**#FF0000**)。

   c 縮放 300%。

   d 將內容的 「X:」位置變更為 **7.0**，以修改每個按鈕狀態中圖示的水平位置。

   備註：藉由這種位置變更方式，可以避免開啟每個按鈕狀態，以及移動圖示影片片段實體。

13 按一下時間軸上方的藍色向後箭頭，以返回 「場景 1」、「影格 1」。

14 將 SeekBar 組件拖曳到 「舞台」放在 FLVPlayback 實體的右下角。

15 在 「元件庫」面板中，按兩下 SeekBar 以在 「舞台」上開啟。

16 將組件縮放至 400%。

17 選取外框，並將顏色設定為紅色 (**#FF0000**)。

18 按兩下 FLVPlayback Skins/Seek Bar 資料夾中的 SeekBarProgress，並將顏色設定為黃色 (**#FFFF00**)。

19 按兩下 FLVPlayback Skins/Seek Bar 資料夾中的 SeekBarHandle，並將顏色設定為紅色 (**#FF0000**)。

20 按一下時間軸上方的藍色向後箭頭，以返回 「場景 1」、「影格 1」。

21 選取 「舞台」上的 SeekBar 實體，並賦予實體名稱 **my_seekbar**。

22 在 「時間軸」的 「影格 1」的 「動作」面板中，新增視訊類別的 import 陳述式，並指定按鈕和搜尋列名稱給對應的 FLVPlayback 屬性，如下列範例所示：

```
import fl.video.*;
my_FLVPlybk.stopButton = my_stopbttn;
my_FLVPlybk.playPauseButton = my_plypausbttn;
my_FLVPlybk.muteButton = my_mutebttn;
my_FLVPlybk.seekBar = my_seekbar;
```

23 按 Control+Enter 測試影片。

建立新的外觀元素

建立外觀元素 SWF 檔的最好方法是，複製 Flash 隨附的其中一個外觀元素檔案，並將它當做起點。您可以在 Flash 應用程式的 Configuration/FLVPlayback Skins/FLA/ActionScript 3.0/ 資料夾內找到這些外觀元素的 FLA 檔。若要讓完成的外觀元素 SWF 檔成為「選取外觀元素」對話方塊中的選項，請將它放在 Flash 應用程式的 Configuration/FLVPlayback Skins/ActionScript 3.0 資料夾或是使用者的本機 Configuration/FLVPlayback Skins/ActionScript 3.0 資料夾中。

由於您可以設定外觀元素的顏色 (與選擇的外觀元素無關)，也就不需要編輯 FLA 檔來修改顏色。如果建立的外觀元素使用特定的顏色，而且您不想讓它成為「選取外觀元素」對話方塊中的可編輯項目，請在外觀元素 FLA 檔 ActionScript 程式碼中設定 this.border_mc.colorMe = false;。如需有關設定外觀元素顏色的詳細資訊，請參閱第 135 頁「選取預先設計的外觀元素」。

在查看已安裝的 Flash 外觀元素 FLA 檔時，您可能會覺得「舞台」上似乎有些東西並不是那麼需要，但那些東西卻放入導引線圖層。一旦使用縮放 9 啟用即時預覽，您就能迅速看出 SWF 檔實際顯示的內容。

下列各節涵蓋更複雜的 SeekBar、BufferingBar 和 VolumeBar 影片片段自訂及變更方式。

使用外觀元素版面配置

當您開啟 Flash 外觀元素 FLA 檔之後，就會在主要「時間軸」上看到外觀元素的影片片段配置。出現在此相同影格中的影片片段和 ActionScript 程式碼，定義了控制項在執行階段的配置方式。

雖然「版面」圖層看起來與外觀元素在執行階段的樣式非常相近，這個圖層的內容並不會在執行階段顯示出來。其用途僅為計算控制項的擺放位置。執行階段將會用到「舞台」上的其它控制項。

「版面」圖層內含一個 FLVPlayback 組件的預留位置，名為 video_mc。所有其它控制項都會相對於 video_mc 進行排列。如果您從某個 Flash FLA 檔開始著手，而後變更了控制項的大小，則或許可藉由搬移這些預留位置片段以修復版面配置。

每個預留位置片段都有特定的實體名稱。預留位置片段的名稱包括 playpause_mc、play_mc、pause_mc、stop_mc、captionToggle_mc、fullScreenToggle_mc、back_mc、bufferingBar_mc、bufferingBarFill_mc、seekBar_mc、seekBarHandle_mc、seekBarProgress_mc、volumeMute_mc、volumeBar_mc 和 volumeBarHandle_mc。當您所選的外觀元素顏色名為 border_mc 時，片段就會重新著色。

控制項使用哪一個片段並不重要。通常，按鈕會使用一般狀態片段。其它控制項則使用該控制項本身的片段，但只是為了方便起見。最重要的是預留位置的 x (水平) 和 y (垂直) 位置，以及高度和寬度。

除了使用標準控制項，您還可以視需要額外加入更多片段。唯一的必要條件是，在「連結」對話方塊中為這類影片片段的元件庫元件選取「匯出給 ActionScript 使用」。「版面」圖層中的自訂影片片段可以擁有任何實體名稱，而非僅限上面列出的實體名稱。只有在設定片段的 ActionScript 以決定版面配置時才需要用到實體名稱。

border_mc 片段的情況比較特殊。如果您將 FlvPlayback.skinAutoHide 屬性設定為 true，則當滑鼠移到 border_mc 片段上方時，外觀元素就會顯示出來。這對於出現在視訊播放程式邊界之外的外觀元素來說，相當重要。如需有關 skinAutoHide 屬性的詳細資訊，請參閱第 143 頁「修改外觀元素行為」。

在 Flash FLA 檔中，border_mc 是當做系統顏色，而且是做為 Forward 和 Back 按鈕的邊框。

border_mc 片段也是屬於外觀元素，可以利用 skinBackgroundAlpha 和 skinBackgroundColor 屬性來變更 Alpha 值和顏色。若要允許可自訂的顏色和 Alpha 值，外觀元素 FLA 檔中的 ActionScript 必須包含下列項目：

```
border_mc.colorMe = true;
```

ActionScript 和外觀元素版面配置

一般而言，下列 ActionScript 程式碼適用於所有控制項。某些控制項具有特定的 ActionScript，定義了各控制項章節中所述的額外行為。

起始 ActionScript 是一個大型區段，指定每個組件各種狀態的類別名稱。您可以在 SkinOverAll.fla 檔案中看到這些類別名稱。例如，Pause 和 Play 按鈕的程式碼看起來會像這樣：

```
this.pauseButtonDisabledState = "fl.video.skin.PauseButtonDisabled";
this.pauseButtonDownState = "fl.video.skin.PauseButtonDown";
this.pauseButtonNormalState = "fl.video.skin.PauseButtonNormal";
this.pauseButtonOverState = "fl.video.skin.PauseButtonOver";
this.playButtonDisabledState = "fl.video.skin.PlayButtonDisabled";
this.playButtonDownState = "fl.video.skin.PlayButtonDown";
this.playButtonNormalState = "fl.video.skin.PlayButtonNormal";
this.playButtonOverState = "fl.video.skin.PlayButtonOver";
```

類別名稱並無實際的外部類別檔案；這些名稱僅由元件庫內所有影片片段的 「連結」對話方塊加以指定。

在 ActionScript 2.0 組件中，「舞台」上有些影片片段是執行階段實際使用的片段。在 ActionScript 3.0 組件中，這些影片片段仍然位於 FLA 檔，但只是為了方便編輯而已。這些片段如今全都位於導引線圖層且不會匯出。元件庫中所有的外觀元素資源都設定成匯出在第一個影格，而且是使用和下列類似的程式碼動態建立：

```
new fl.video.skin.PauseButtonDisabled();
```

此區段後面的 ActionScript 程式碼定義了外觀元素的最小寬度和高度。「選取外觀元素」對話方塊會顯示這些值，而執行階段也會使用這些值，以防止外觀元素縮放成小於其最小尺寸。如果您不想指定最小尺寸，請保留其值為未定義，或是指定為小於或等於零。

```
// minimum width and height of video recommended to use this skin,
// leave as undefined or <= 0 if there is no minimum
this.minWidth = 270;
this.minHeight = 60;
```

各個預留位置可能會套用下列屬性：

屬性	說明
anchorLeft	Boolean。將控制項放在 FLVPlayback 實體的左邊相對位置上。預設值為 true，除非 anchorRight 明確設定為 true 致使其值預設為 false。
anchorRight	Boolean。將控制項放在 FLVPlayback 實體的右邊相對位置上。預設值為 false。
anchorBottom	Boolean。將控制項放在 FLVPlayback 實體的底端相對位置上。預設值為 true，除非 anchorTop 明確設定為 true 致使其值預設為 false。
anchorTop	Boolean。將控制項放在 FLVPlayback 實體的頂端相對位置上。預設值為 false。

如果 anchorLeft 和 anchorRight 屬性都為 true，則控制項會在執行階段進行水平縮放。如果 anchorTop 和 anchorBottom 屬性都為 true，則控制項會在執行階段進行垂直縮放。

若要查看這些屬性的作用，請參閱 Flash 外觀元素的屬性用法。BufferingBar 和 SeekBar 是唯一會進行縮放並能彼此相疊於頂端的控制項，且兩者的 anchorLeft 和 anchorRight 屬性都是設為 true。BufferingBar 和 SeekBar 左邊的所有控制項都將 anchorLeft 設定為 true，而右邊的控制項則將 anchorRight 設定為 true。所有控制項的 anchorBottom 都設為 true。

您可以嘗試編輯 「版面」圖層上的影片片段，讓外觀元素的控制項座落於頂端而非底端。只需要將控制項移到 video_mc 的頂端相對位置上，並將所有控制項的 anchorTop 設定等於 true 即可。

緩衝列

緩衝列具有兩個影片片段：bufferingBar_mc 和 bufferingBarFill_mc。各個片段與 「舞台」上其它片段的相對位置很重要，因為它們會維持這個相對位置。緩衝列使用兩個不同的片段是因為組件會縮放 bufferingBar_mc 而非 bufferingBarFill_mc。

bufferingBar_mc 片段套用了 9 分割縮放，所以在進行縮放時，邊框將不會扭曲。bufferingBarFill_mc 片段非常寬廣，所以寬度永遠足夠而不必進行縮放。執行階段會自動遮色處理此片段，以便只顯示延伸的 bufferingBar_mc 上方的部分。根據預設，遮色片的確切尺寸是以 bufferingBar_mc 和 bufferingBarFill_mc 之間 x (水平) 位置的差異為準計算，使 bufferingBar_mc 內的左邊界和右邊界保持相等。您可以利用 ActionScript 程式碼自訂位置。

如果緩衝列不需要進行縮放，或者不使用 9 分割縮放，您即可將它當成 FLV 播放自訂 UI BufferingBar 組件般加以設定。如需詳細資訊，請參閱第 137 頁 「BufferingBar 組件」。

緩衝列還具有下列另一個屬性：

屬性	說明
fill_mc:MovieClip	指定緩衝列填色的實體名稱。預設為 bufferingBarFill_mc。

搜尋列和音量列

搜尋列也有兩個影片片段：seekBar_mc 和 seekBarProgess_mc。各個片段與 「版面」圖層上其它片段的相對位置很重要，因為它們會維持這個相對位置。雖然這兩個片段都要進行縮放，seekBarProgress_mc 卻不能巢狀放置在 seekBar_mc 內，因為 seekBar_mc 會使用 9 分割縮放，而無法與巢狀的影片片段一起使用。

seekBar_mc 片段套用了 9 分割縮放，所以在進行縮放時，邊框將不會扭曲。seekBarProgress_mc 片段也進行縮放，但會扭曲變形。此片段不使用 9 分割縮放是因僅為填色用途，若發生扭曲也無所謂。

seekBarProgress_mc 片段必須搭配 fill_mc 才能運作，就像 FLV 播放自訂 UI 組件的 progress_mc 片段一樣。換句話說，片段並未以遮色處理，且會進行水平縮放。seekBarProgress_mc 達 100% 時的確切尺寸是由 seekBarProgress_mc 影片片段內的左邊界和右邊界所定義。這些尺寸依預設會保持相等，並且是以 seekBar_mc 和 seekBarProgress_mc 之間 x (水平) 位置的差異為準計算。您可以利用搜尋列影片片段的 ActionScript 自訂尺寸，如下列範例所示：

```
this.seekBar_mc.progressLeftMargin = 2;
this.seekBar_mc.progressRightMargin = 2;
this.seekBar_mc.progressY = 11;
this.seekBar_mc.fullnessLeftMargin = 2;
this.seekBar_mc.fullnessRightMargin = 2;
this.seekBar_mc.fullnessY = 11;
```

您可以將這段程式碼放入 SeekBar 影片片段 「時間軸」內，或是和其它 ActionScript 程式碼一起放到主要 「時間軸」上。如果是使用程式碼自訂尺寸而不修改版面，「舞台」上就不需要包括填色。只要其確切的類別名稱位於元件庫中，並在 「影格 1」設定為 「匯出給 ActionScript 使用」即可。

和 FLV 播放自訂 UI SeekBar 組件一樣，搜尋列也可以建立完整性影片片段。如果搜尋列不需要進行縮放，或者要縮放但不使用 9 分割縮放，您可以使用 FLV 播放自訂 UI 組件採用的任何方法來設定 progress_mc 或 fullness_mc。如需詳細資訊，請參閱。

由於 Flash 外觀元素中的音量列不會進行縮放，其建構方式便與 VolumeBar FLV 播放自訂 UI 組件相同。如需詳細資訊，請參閱第 137 頁 「SeekBar 和 VolumeBar 組件」。唯一的差別在於控制點以不同的方式實作。

Seekbar 和 VolumeBar 控制點

SeekBar 和 VolumeBar 控制點位於列旁邊的 「版面」圖層上。根據預設，控制點的左邊界、右邊界和 y 軸值都是依其相對於列影片片段的位置來設定。左邊界是根據控制點的 x (水平) 位置和列的 x (水平) 位置之間的差異來設定，並且右邊界等於左邊界。您可以透過 SeekBar 或 VolumeBar 影片片段的 ActionScript 自訂這些值。下列範例是 FLV 播放自訂 UI 組件曾經用過的同一段 ActionScript 程式碼：

```
this.seekBar_mc.handleLeftMargin = 2;
this.seekBar_mc.handleRightMargin = 2;
this.seekBar_mc.handleY = 11;
```

您可以將這段程式碼放入 SeekBar 影片片段 「時間軸」內，或是和其它 ActionScript 程式碼一起放到主要 「時間軸」上。如果使用程式碼自訂值而不修改版面，「舞台」上就不需要有控制點。只要其確切的類別名稱位於元件庫中，並在 「影格 1」設定為 「匯出給 ActionScript 使用」即可。

若撇開這些屬性不談，控制點實為簡單的影片片段，設定方式與 FLV 播放自訂 UI 組件做法相同。兩者都具有 alpha 屬性設為 0 的矩形背景。這類背景僅為增加作用區域而顯示，非絕對必要。

背景和前景片段

影片片段 chrome_mc 和 forwardBackBorder_mc 會實作為背景片段。

「舞台」上的 ForwardBackBorder、ForwardBorder 和 BackBorder 影片片段以及預留位置 Forward 和 Back 按鈕，唯一不在導引線圖層上的是 ForwardBackBorder。它只存在於實際使用 Forward 和 Back 按鈕的外觀元素中。

您只需要透過元件庫，在「影格 1」匯出這類片段給 ActionScript 使用。

修改外觀元素行為

bufferingBarHidesAndDisablesOthers 屬性和 skinAutoHide 屬性可讓您自訂 FLVPlayback 外觀元素的行為。

若將 bufferingBarHidesAndDisablesOthers 屬性設為 true，會使 FLVPlayback 組件隱藏 SeekBar 及其控制點，並且在組件進入緩衝狀態時停用 Play 和 Pause 按鈕。透過慢速連線從 FMS 串流 FLV 檔時，如果 bufferTime 屬性設定偏高 (例如 10)，這可能會很有用。在此情況下，沒有耐心的使用者也許會按 Play 和 Pause 按鈕嘗試啟動搜尋，卻反而導致更延後播放檔案。您可以在組件處於緩衝狀態時，將 bufferingBarHidesAndDisablesOthers 設定為 true 並停用 SeekBar 元素和 Pause 與 Play 按鈕，避免這種現象發生。

skinAutoHide 屬性只會影響預先設計的外觀元素 SWF 檔，不會影響到經由 FLV 播放自訂 UI 組件建立的控制項。如果設定為 true，則當滑鼠移到檢視區域之外時，FLVPlayback 組件便會隱藏外觀元素。這個屬性的預設值為 false。

使用 SMIL 檔

為了控制多種頻寬下的多個串流，VideoPlayer 類別會使用支援 SMIL 子集的 helper 類別 (NCManager)。SMIL 是用來識別視訊串流的位置、FLV 檔的版面配置 (寬度和高度)，以及對應至不同頻寬的來源 FLV 檔，也可以用來指定 FLV 檔的位元速率和持續時間。

使用 source 參數或 FLVPlayback.source 屬性 (ActionScript) 來指定 SMIL 檔案的位置。如需詳細資訊，請參閱 適用於 Adobe Flash Platform 的 ActionScript 3.0 參考中的與 FLVPlayback.source 屬性。

下列範例所示的 SMIL 檔會使用 RTMP 從 FMS 串流處理多個頻寬 FLV 檔：

```
<smil>
<head>
<meta base="rtmp://myserver/myapp/" />
<layout>
<root-layout width="240" height="180" />
</layout>
</head>
<body>
            <switch>
                    <ref src="myvideo_cable.flv" dur="3:00.1"/>
                    <video src="myvideo_isdn.flv" system-bitrate="128000" dur="3:00.1"/>
                    <video src="myvideo_mdm.flv" system-bitrate="56000"dur="3:00.1"/>
            </switch>
</body>
</smil>
```

<head> 標籤可能包含 <meta> 和 <layout> 標籤。<meta> 標籤只支援 base 特質，此特質用於指定串流視訊的 URL (RTMP 來自 FMS)。

<layout> 標籤只支援 root-layout 元素，此元素用於設定 height 和 width 特質，進而決定 FLV 檔呈現所在視窗的大小。這些特質只接受像素值，不接受百分比值。

在 SMIL 檔的主體中，您可以加入某個 FLV 來源檔的單一連結，或者若要從 FMS 串流處理多種頻寬的多個檔案 (如上述範例所示)，則可使用 <switch> 標籤列出所有來源檔。

<switch> 標籤內的 video 和 ref 標籤是同義的，兩者皆可使用 src 特質來指定 FLV 檔。再者，每一個標籤都可以使用 region、system-bitrate 和 dur 特質來指定 FLV 檔的區域、所需最低頻寬以及持續時間。

在 <body> 標籤內，只允許 <video>、<src> 或 <switch> 標籤其中之一存在。

下列範例顯示漸進式下載單一 FLV 檔而不使用頻寬偵測：

```
<smil>
        <head>
            <layout>
                <root-layout width="240" height="180" />
            </layout>
        </head>
        <body>
            <video src=""myvideo.flv" />
        </body>
</smil>
```

<smil>

適用版本
Flash Professional 8。

用法

```
<smil>
...
child tags
...
</smil>
```

特質
無。

子標籤

```
<head>, <body>
```

父標籤
無。

說明
識別 SMIL 檔的最上層標籤。

範例
下列範例顯示指定三種 FLV 檔的 SMIL 檔：

```
<smil>
<head>
<meta base="rtmp://myserver/myapp/" />
<layout>
<root-layout width="240" height="180" />
</layout>
</head>
<body>
<switch>
                <ref src="myvideo_cable.flv" dur="3:00.1"/>
                <video src="myvideo_isdn.flv" system-bitrate="128000" dur="3:00.1"/>
                <video src="myvideo_mdm.flv" system-bitrate="56000"dur="3:00.1"/>
            </switch>
</body>
</smil>
```

<head>

適用版本

Flash Professional 8。

用法

```
<head>
...
child tags
...
</head>
```

特質

無。

子標籤

```
<meta>, <layout>
```

父標籤

```
<smil>
```

說明

支援 <meta> 和 <layout> 標籤，指定來源 FLV 檔的位置和預設版面配置 (高度和寬度)。

範例

下列範例會將根目錄版面配置設定為 240 像素 x 180 像素：

```
<head>
    <meta base="rtmp://myserver/myapp/" />
    <layout>
        <root-layout width="240" height="180" />
    </layout>
</head>
```

<meta>

適用版本
Flash Professional 8。

用法
```
<meta/>
```

特質
```
base
```

子標籤
```
<layout>
```

父標籤
無。

說明
包含 base 特質，指定來源 FLV 檔的位置 (RTMP URL)。

範例
下列範例顯示 myserver 上基礎位置的 Meta 標籤：

```
<meta base="rtmp://myserver/myapp/" />
```

<layout>

適用版本
Flash Professional 8。

用法
```
<layout>
...
child tags
...
</layout>
```

特質
無。

子標籤
```
<root-layout>
```

父標籤
```
<meta>
```

說明
指定 FLV 檔的寬度和高度。

範例
下列範例將指定 240 像素 x 180 像素的版面配置：

```
<layout>
    <root-layout width="240" height="180" />
</layout>
```

<root-layout>

適用版本
Flash Professional 8。

用法
```
<root-layout...attributes.../>
```

特質
Width、height

子標籤
無。

父標籤
```
<layout>
```

說明
指定 FLV 檔的寬度和高度。

範例
下列範例將指定 240 像素 x 180 像素的版面配置：

```
<root-layout width="240" height="180" />
```

<body>

適用版本
Flash Professional 8。

用法
```
<body>
...
child tags
...
</body>
```

特質
無。

子標籤
```
<video>, <ref>, <switch>
```

父標籤
<smil>

說明
包含 <video>、<ref> 和 <switch> 標籤，指定來源 FLV 檔的名稱、最低頻寬和 FLV 檔的持續時間。system-bitrate 特質只有在使用 <switch> 標籤時才受到支援。在 <body> 標籤內，只允許 <switch>、<video> 或 <ref> 標籤其中之一存在。

範例
下列範例將指定三個 FLV 檔，其中兩個使用 video 標籤，另一個則使用 ref 標籤：

```
<body>
    <switch>
        <ref src="myvideo_cable.flv" dur="3:00.1"/>
        <video src="myvideo_isdn.flv" system-bitrate="128000" dur="3:00.1"/>
        <video src="myvideo_mdm.flv" system-bitrate="56000"dur="3:00.1"/>
    </switch>
</body>
```

<video>

適用版本
Flash Professional 8。

用法
<video...attributes.../>

特質
src, system-bitrate, dur

子標籤
無。

父標籤
<body>

說明
與 <ref> 標籤同義。支援 src 和 dur 特質，指定來源 FLV 的名稱和持續時間。dur 特質支援完整的 (00:03:00:01) 和部分 (03:00:01) 時間格式。

範例
下列範例設定視訊的來源和持續時間：

```
<video src="myvideo_mdm.flv" dur="3:00.1"/>
```

<ref>

適用版本
Flash Professional 8。

用法
```
<ref...attributes.../>
```

特質
```
src, system-bitrate, dur
```

子標籤
無。

父標籤
```
<body>
```

說明
與 <video> 標籤同義。支援 src 和 dur 特質，指定來源 FLV 的名稱和持續時間。dur 特質支援完整的 (00:03:00:01) 和部分 (03:00:01) 時間格式。

範例
下列範例設定視訊的來源和持續時間：
```
<ref src="myvideo_cable.flv" dur="3:00.1"/>
```

<switch>

適用版本
Flash Professional 8。

用法
```
<switch>
...
child tags
...
<switch/>
```

特質
無。

子標籤
```
<video>, <ref>
```

父標籤
```
<body>
```

說明
與 <video> 或 <ref> 子標籤一起使用，以列出多種頻寬視訊串流的 FLV 檔。<switch> 標籤支援 system-bitrate 特質，此特質指定最低頻寬以及 src 和 dur 特質。

範例

下列範例將指定三個 FLV 檔，其中兩個使用 video 標籤，另一個則使用 ref 標籤：

```
<switch>
    <ref src="myvideo_cable.flv" dur="3:00.1"/>
    <video src="myvideo_isdn.flv" system-bitrate="128000" dur="3:00.1"/>
    <video src="myvideo_mdm.flv" system-bitrate="56000"dur="3:00.1" />
</switch>
```

第 7 章 使用 FLVPlayback 註解功能組件

FLVPlayback 組件可以讓您在 Adobe Flash CS5 Professional 應用程式中加入視訊播放程式，播放已下載的 Adobe Flash Video (FLV 或 F4V) 檔和串流 FLV 或 F4V 檔。如需有關 FLVPlayback 的詳細資訊，請參閱第 119 頁「使用 FLVPlayback 組件」。

FLVPlaybackCaptioning 組件可以讓您在視訊中加入封閉式註解功能的支援。註解功能組件支援 W3C 標準 XML 格式 Timed Text，並包括下列功能：

使用內嵌事件提示點加上註解 讓 FLV 檔中的內嵌事件提示點與 XML 產生關聯，以提供註解功能，而不需要使用 Timed Text XML 檔。

多重 FLVPlayback 註解功能 為多個 FLVPlayback 實體建立多個 FLVPlayback 註解。

切換按鈕控制項 透過註解功能切換按鈕讓使用者與註解功能進行互動。

使用 FLVPlaybackCaptioning 組件

您可以在一個或多個 FLVPlayback 組件上使用 FLVPlaybackCaptioning 組件。在這個最簡單的案例中，您會拖曳舞台上的 FLVPlayback 組件、拖曳同一個舞台上的 FLVPlaybackCaptioning 組件、識別註解的 URL，並設定要顯示的註解。此外，您也可以設定參數來自訂 FLVPlayback 註解功能。

將註解功能加入到 FLVPlayback 組件

您可以將 FLVPlaybackCaptioning 組件加入到任何的 FLVPlayback 組件。如需有關將 FLVPlayback 組件加入應用程式中的詳細資訊，請參閱第 120 頁 「建立具有 FLVPlayback 組件的應用程式」。

從組件面板加入 **FLVPlaybackCaptioning** 組件：

1 在 「組件」面板中，開啟 「視訊」資料夾。

2 拖曳 (或按兩下) FLVPlaybackCaptioning 組件，然後將它加入到舞台 (FLVPlayback 組件所在的同一個舞台) 上您要加入註解功能的位置。

備註：Adobe 提供兩個檔案幫助您快速瞭解 FLVPlaybackCaptioning 組件：caption_video.flv (FLVPlayback 樣本) 和 caption_video.xml (註解樣本)。您可以在 www.helpexamples.com/flash/video/caption_video.flv 和 www.helpexamples.com/flash/video/caption_video.xml 找到這些檔案。

3 (選擇性) 將 CaptionButton 組件拖曳到與 FLVPlayback 和 FLVPlaybackCaptioning 組件所在的同一個舞台上。CaptionButton 組件可以讓使用者啟用或停用註解功能。

備註：如果要啟用 CaptionButton 組件，您必須將它拖曳到與 FLVPlayback 和 FLVPlaybackCaptioning 組件所在的同一個舞台上。

4 選取 「舞台」上的 FLVPlaybackCaptioning 組件，然後在 「屬性」檢測器的 「參數」索引標籤中指定下列必要的資訊：

- 將 showCaptions 設定為 true。
- 指定要下載之 Timed Text XML 檔的 source。

在 Flash 中測試註解功能時，您應該將 showCaptions 屬性設定為 true。然而，如果要加入可以讓使用者啟用或停用註解功能的 CaptionButton 組件，就必須將 showCaptions 屬性設定為 false。

其它參數也可以用來協助您自訂 FLVPlaybackCaptioning 組件。如需詳細資訊，請參閱第 160 頁 「自訂 FLVPlayback-Captioning 組件」以及 「適用於 Adobe Flash Platform 的 ActionScript 3.0 參考」。

5 選取 「控制 > 測試影片」，啟動視訊。

使用 **ActionScript** 動態地建立實體：

1 將 FLVPlayback 組件從 「組件」面板拖曳到 「元件庫」面板 (「視窗 > 元件庫」)。

2 將 FLVPlaybackCaptioning 組件從 「組件」面板拖曳到 「元件庫」面板。

3 在時間軸的 「影格 1」的 「動作」面板中，加入下列程式碼。

```
import fl.video.*;
var my_FLVPlybk = new FLVPlayback();
my_FLVPlybk.x = 100;
my_FLVPlybk.y = 100;
addChild(my_FLVPlybk);
my_FLVPlybk.skin = "install_drive:/Program Files/Adobe/Adobe Flash CS5/en/Configuration/FLVPlayback
Skins/ActionScript 3.0/SkinUnderPlaySeekCaption.swf";
my_FLVPlybk.source = "http://www.helpexamples.com/flash/video/caption_video.flv";
var my_FLVPlybkcap = new FLVPlaybackCaptioning();
addChild (my_FLVPlybkcap);
my_FLVPlybkcap.source = "http://www.helpexamples.com/flash/video/caption_video.xml";
my_FLVPlybkcap.showCaptions = true;
```

4 將 install_drive 變更為 Flash 安裝所在的磁碟機，並修改路徑以反映安裝的 Skins 資料夾位置：

備註：如果是使用 ActionScript 建立 FLVPlayback 實體，您必須同時使用 ActionScript 設定 skin 屬性，將外觀元素動態指定給這個實體。如果是使用 ActionScript 套用外觀元素，就不會自動使用 SWF 檔將它發佈。請將外觀元素 SWF 檔和應用程式 SWF 檔一併複製到伺服器，否則使用者在執行應用程式時將無法使用外觀元素 SWF 檔。

設定 FLVPlaybackCaptioning 組件參數

對於每個 FLVPlaybackCaptioning 組件的實體，您都可以在 「屬性」檢測器或 「組件檢測器」中設定下列參數來進一步自訂組件。下列清單將列出並提供這些屬性的簡短說明：

autoLayout 決定 FLVPlaybackCaptioning 組件是否可以控制註解功能區域的大小。預設值為 true。

captionTargetName 確認 TextField 或 MovieClip 實體名稱是否含有註解。預設值為 auto。

flvPlaybackName 確認您要加上註解的 FLVPlayback 實體名稱。預設值為 auto。

simpleFormatting 當設定為 true 時，限制 Timed Text XML 檔的格式指示。預設值為 false。

showCaptions 決定是否要顯示註解。預設值為 true。

source 確認 Timed Text XML 檔的位置。

如需有關所有 FLVPlaybackCaptioning 參數的詳細資訊，請參閱 「適用於 Adobe Flash Platform 的 ActionScript 3.0 參考」。

指定 source 參數

source 參數可以用來指定包含影片註解之 Timed Text XML 檔的名稱和位置。請直接在 「組件檢測器」中的來源儲存格輸入 URL 路徑。

顯示註解

如果要檢視註解功能，請將 showCaptions 參數設定為 true。

如需有關所有 FLVPlaybackCaptioning 組件參數的詳細資訊，請參閱「適用於 Adobe Flash Platform 的 ActionScript 3.0 參考」。

在上一個範例中，您已經學會如何建立和啟用 FLVPlaybackCaptioning 組件來顯示註解。您可以使用兩種註解來源：(1) 內含註解的 Timed Text XML 檔，或 (2) 內含與內嵌事件提示點關聯之註解文字的 XML 檔。

使用 Timed Text 註解

FLVPlaybackCaptioning 組件會下載 Timed Text (TT) XML 檔，為相關的 FLVPlayback 組件啟用註解功能。如需有關 Timed Text 格式的詳細資訊，請參閱 www.w3.org 上的音效視訊 Timed Text 資訊。

本節概要說明支援的 Timed Text 標籤 (必要的註解功能檔標籤) 並提供一個 Timed Text XML 檔的範例。如需有關所有支援之 Timed Text 標籤的詳細資訊，請參閱第 154 頁 「Timed Text 標籤」。

FLVPlaybackCaptioning 組件支援下列 Timed Text 標籤：

類別	工作
段落格式支援	將段落靠右、靠左或置中對齊
文字格式支援	• 以絕對像素大小或 delta 樣式 (例如：+2、-4) 設定文字的大小 • 設定文字顏色和字體 • 設定文字為粗體和斜體 • 設定文字齊行
其它格式支援	• 為註解設定 TextField 的背景顏色 • 為註解將 TextField 的背景顏色設定為透明 (Alpha 0) • 為註解設定 TextField 的文字換行功能 (啟用或停用)

FLVPlaybackCaptioning 組件符合 FLV 檔的時間碼。每個註解都必須具有一個 begin 特質，用來決定註解應該出現的時間。如果註解沒有 dur 或 end 特質，當下一個註解出現或 FLV 檔結束時，註解就會消失。

下列為 Timed Text XML 檔的範例。這個檔案 (caption_video.xml) 將為 caption_video.flv 檔案提供註解功能。您可以在 www.helpexamples.com/flash/video/caption_video.flv 和 www.helpexamples.com/flash/video/caption_video.xml 找到這些檔案。

```
<?xml version="1.0" encoding="UTF-8"?>
    <tt xml:lang="en"
xmlns="http://www.w3.org/2006/04/ttaf1"xmlns:tts="http://www.w3.org/2006/04/ttaf1#styling">
<head>
    <styling>
<style id="1" tts:textAlign="right"/>
<style id="2" tts:color="transparent"/>
<style id="3" style="2" tts:backgroundColor="white"/>
<style id="4" style="2 3" tts:fontSize="20"/>
    </styling>
</head>
<body>
    <div xml:lang="en">
<p begin="00:00:00.00" dur="00:00:03.07">I had just joined <span
tts:fontFamily="monospaceSansSerif,proportionalSerif,TheOther"tts:fontSize="+2">Macromedia</span> in
1996,</p>
<p begin="00:00:03.07" dur="00:00:03.35">and we were trying to figure out what to do about the internet.</p>
<p begin="00:00:06.42" dur="00:00:03.15">And the company was in dire straights at the time.</p>
<p begin="00:00:09.57" dur="00:00:01.45">We were a CD-ROM authoring company,</p>
<p begin="00:00:11.42" dur="00:00:02.00">and the CD-ROM business was going away.</p>
<p begin="00:00:13.57" dur="00:00:02.50">One of the technologies I remember seeing was Flash.</p>
<p begin="00:00:16.47" dur="00:00:02.00">At the time, it was called <span tts:fontWeight="bold"
tts:color="#ccc333">FutureSplash</span>.</p>
<p begin="00:00:18.50" dur="00:00:01.20">So this is where Flash got its start.</p>
<p begin="00:00:20.10" dur="00:00:03.00">This is smart sketch running on the <span
tts:fontStyle="italic">EU-pin computer</span>,</p>
<p begin="00:00:23.52" dur="00:00:02.00">which was the first product that FutureWave did.</p>
<p begin="00:00:25.52" dur="00:00:02.00">So our vision for this product was to</p>
<p begin="00:00:27.52" dur="00:00:01.10">make drawing on the computer</p>
<p begin="00:00:29.02" dur="00:00:01.30" style="1">as <span tts:color="#ccc333">easy</span> as drawing on
paper.</p>
</div>
</body>
</tt>
```

Timed Text 標籤

FLVPlaybackCaptioning 組件支援註解功能 XML 檔的 Timed Text 標籤。如需有關音效視訊 Timed Text 標籤的詳細資訊，請參閱 www.w3.org 上的資訊。下表列出支援和不支援的標籤。

函數	標籤 / 值	用途 / 說明	範例
忽略的標籤	metadata	忽略 / 允許於任何文件層級	
	set	忽略 / 允許於任何文件層級	
	xml:lang	忽略	
	xml:space	忽略 / 行為指令會覆寫至： xml:space="default"	
	layout	忽略 / 包括配置標籤區段中的任何區域標籤	
	br 標籤	所有的特質和內容都會被忽略。	
註解的媒體時間	begin 特質	只在 p 標籤中允許。註解媒體時間部署之必要項目。	<p begin="3s">
	dur 特質	只在 p 標籤中允許。建議使用。如果未包括，註解會隨 FLV 檔結束或在另一個註解開始時結束。	

函數	標籤 / 值	用途 / 說明	範例
	end 特質	只在 p 標籤中允許。建議使用。如果未包括，註解會隨 FLV 檔結束或在另一個註解開始時結束。	
註解的時鐘時間	00:03:00.1	完整時鐘格式	
	03:00.1	部分時鐘格式	
	10	不使用單位的偏移時間。偏移表示秒數。	
	00:03:00:05 00:03:00:05.1 30f 30t	不支援。不支援包括影格或刻度的時間格式。	
主體標籤	body	必要項目 / 只支援一個 body 標籤。	<body> ... </body>
內容標籤	div 標籤	允許零個以上。使用第一個標籤。	
	p 標籤	允許零個以上。	
	span 標籤	一連串文字內容單位的邏輯容器。不支援巢狀範圍。支援特質樣式標籤。	
	br 標籤	表示明確的斷行符號。	
樣式標籤 (所有樣式標籤都用於 p 標籤內)	style	參考一個或多個樣式元素。可以做為標籤和特質使用。做為標籤時，ID 特質為必要項目 (樣式可以重複使用於文件)。支援在樣式標籤內部使用一個或多個樣式標籤。	
	tts:background Color	指定定義區域背景顏色的樣式屬性。除設定為零 (Alpha 0) 使背景透明外，一律忽略 Alpha。顏色格式為 #RRGGBBAA。	

函數	標籤 / 值	用途 / 說明	範例
	tts:color	指定定義前景顏色的樣式屬性。任何顏色都不支援 Alpha。transparent 值將轉譯為黑色。	<style id="3" style="2" tts:backgroundColor="white"/> "transparent" = #00000000 "black"=#000000FF "silver"=#C0C0C0FF "grey"=#808080FF "white"=#FFFFFFFF "maroon"=#800000FF "red"=#FF0000FF "purple"=#800080FF "fuchsia"("magenta")= #FF00FFFF "green"=#008000FF "lime"=#00FF00FF "olive"=#808000FF "yellow"=#FFFF00FF "navy"=#000080FF "blue"=#0000FFFF "teal"=#008080FF "aqua"("cyan")=#00FFFFFF
	tts:fontFamily	指定定義字體系列的樣式屬性。	"default" = _serif "monospace" = _typewriter "sansSerif" = _sans "serif" = _serif "monospaceSansSerif" =_typewriter "monospaceSerif" =_typewriter "proportionalSansSerif" = _sans
	tts:fontSize	指定定義字體大小的樣式屬性。如果提供兩個值，則只會使用第一個 (垂直) 值。忽略百分比值和單位。支援絕對像素 (例如 12) 和相對樣式 (例如 +2) 大小。	
	tts:fontStyle	指定定義字體樣式的樣式屬性。	"normal" "italic" "inherit"* * 預設行為：繼承封閉標籤的樣式。

函數	標籤 / 值	用途 / 說明	範例
	tts:fontWeight	指定定義字體寬度的樣式屬性。	"normal" "bold" "inherit"* * 預設行為；繼承封閉標籤的樣式。
	tts:textAlign	指定定義容器區塊區域內行內區域對齊方式的樣式屬性。	"left" "right" "center" "start" (="left") "end" (="right") "inherit"* * 繼承封閉標籤的樣式。如果沒有設定 textAlign 標籤，預設為使用 "left"。
	tts:wrapOption	指定定義受影響元素之內容內是否套用自動換行 (斷行) 的樣式屬性。這項設定會影響註解元素中的所有段落。	"wrap" "noWrap" "inherit"* * 繼承封閉標籤的樣式。如果沒有設定 wrapOption 標籤，預設為使用 "wrap"。
不支援的特質	tts:direction tts:display tts:displayAlign tts:dynamicFlow tts:extent tts:lineHeight tts:opacity tts:origin tts:overflow tts:padding tts:showBackground tts:textOutline tts:unicodeBidi tts:visibility tts:writingMode tts:zIndex		

搭配使用提示點和註解功能

提示點可以讓您與視訊進行互動；例如，您可以控制 FLV 檔的播放選項或顯示視訊中特定時點的文字。如果不想在 FLV 檔上使用 Timed Text XML 檔，您可以在 FLV 檔中內嵌事件提示點，然後將這些提示點與文字建立關聯。本節將提供 FLVPlaybackCaptioning 組件提示點標準的資訊，並概要說明如何將這些提示點與註解功能文字建立關聯。如需有關如何使用「視訊匯入」精靈或 Flash Video Encoder 內嵌事件提示點的詳細資訊，請參閱「使用 Flash」中的第 16 章「使用視訊」。

瞭解 FLVPlaybackCaptioning 提示點標準

在 FLV 檔的中繼資料內，提示點會以具有下列屬性的物件來表示：name、time、type 和 parameters。FLVPlaybackCaptioning ActionScript 提示點具有下列特質：

name name 屬性是包含所指定之提示點名稱的字串。name 屬性必須以 fl.video.caption.2.0. 前置詞做為開頭，並在此前置詞後面加上字串。字串是一連串的正整數，它會逐漸遞增以保持每個名稱的唯一性。前置詞包括版本號碼，與 FLVPlayback 版本號碼相符。用於 Adobe Flash CS4 以及更新版本時，必須將版本號碼設定為 2.0。

time time 屬性是註解的預定顯示時間。

type type 屬性是一個字串，其值為 "event"。

parameters parameters 屬性是一個支援下列名稱和值配對的陣列：

- text:String 註解的 HTML 格式文字。這個文字會直接傳遞給 TextField.htmlText 屬性。FLVPlaybackCaptioning 組件支援選擇性的 text:n 屬性，它可以支援使用多國語言音軌。如需詳細資訊，請參閱第 159 頁「使用內嵌提示點支援多國語言音軌」。
- endTime:Number 註解應該要消失的時間。如果您並未指定這個「屬性」，FLVPlaybackCaptioning 組件會假設它不是數字 (NaN)，而且註解將一直顯示到 FLV 檔完成 (FLVPlayback 實體傳送 VideoEvent.COMPLETE 事件) 為止。請以秒數為單位，指定 endTime:Number 屬性。
- backgroundColor:uint 此參數會設定 TextField.backgroundColor。這個屬性是選擇性的。
- backgroundColorAlpha:Boolean 如果 backgroundColor 的 Alpha 為 0%，則參數將設定 TextField.background = !backgroundColor。這個屬性是選擇性的。
- wrapOption:Boolean 此參數會設定 TextField.wordWrap。這個屬性是選擇性的。

瞭解事件內嵌提示點的註解功能

如果您沒有包含 FLV 檔註解的 Timed Text XML 檔，可以讓包含註解功能的 XML 檔與事件內嵌提示點產生關聯，以建立註解功能。XML 樣本假設您已經執行過下列步驟而在視訊中建立了事件內嵌提示點：

- 加入事件提示點 (遵照 FLVPlaybackCaptioning 標準)，並將視訊編碼。
- 在 Flash 中，將 FLVPlayback 組件和 FLVPlaybackCaptioning 組件拖曳到「舞台」上。
- 設定 FLVPlayback 和 FLVPlaybackCaptioning 組件的來源屬性 (FLV 檔的位置和 XML 檔的位置)。
- 發佈。

下列樣本會將 XML 匯入編碼器：

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<FLVCoreCuePoints>

    <CuePoint>
        <Time>9136</Time>
        <Type>event</Type>
        <Name>fl.video.caption.2.0.index1</Name>
        <Parameters>
            <Parameter>
                <Name>text</Name>
                <Value><![CDATA[Captioning text for the first cue point]]></Value>
            </Parameter>
        </Parameters>
    </CuePoint>

    <CuePoint>
        <Time>19327</Time>
        <Type>event</Type>
        <Name>fl.video.caption.2.0.index2</Name>
        <Parameters>
            <Parameter>
                <Name>text</Name>
                <Value><![CDATA[Captioning text for the second cue point]]></Value>
            </Parameter>
        </Parameters>
    </CuePoint>

    <CuePoint>
        <Time>24247</Time>
        <Type>event</Type>
        <Name>fl.video.caption.2.0.index3</Name>
        <Parameters>
            <Parameter>
                <Name>text</Name>
                <Value><![CDATA[Captioning text for the third cue point]]></Value>
            </Parameter>
        </Parameters>
    </CuePoint>

    <CuePoint>
        <Time>36546</Time>
        <Type>event</Type>
        <Name>fl.video.caption.2.0.index4</Name>
        <Parameters>
            <Parameter>
                <Name>text</Name>
                <Value><![CDATA[Captioning text for the fourth cue point]]></Value>
            </Parameter>
        </Parameters>
    </CuePoint>

</FLVCoreCuePoints>
```

FLVPlaybackCaptioning 組件也可以使用內嵌提示點來支援多國語言音軌。如需詳細資訊，請參閱第 159 頁「使用內嵌提示點支援多國語言音軌」。

使用內嵌提示點支援多國語言音軌

只要 Timed Text XML 檔遵循 FLVPlaybackCaptioning 提示點標準，FLVPlaybackCaptioning track 屬性就可以使用內嵌提示點來支援多國語言音軌 (如需詳細資訊，請參閱第 158 頁「瞭解 FLVPlaybackCaptioning 提示點標準」)。但是 FLVPlaybackCaptioning 組件在獨立的 XML 檔中不支援多國語言音軌。如果要使用 track 屬性，請將屬性設定為一個不等於

0 的值。例如，設定 track 屬性為 1 (track == 1) 時，FLVPlaybackCaptioning 組件將會搜尋提示點參數。如果找不到相符的部分，則使用提示點參數中的 text 屬性。如需詳細資訊，請參閱「適用於 Adobe Flash Platform 的 ActionScript 3.0 參考」中的 track 屬性。

使用註解功能播放多個 FLV 檔

您可以在 FLVPlayback 組件的單一實體內開啟多個視訊播放程式，以播放多個視訊並在播放的同時進行切換。您也可以將註解功能與 FLVPlayback 組件內的每個視訊播放程式建立關聯。如需有關如何開啟多個視訊播放程式的詳細資訊，請參閱第 132 頁 「使用多個視訊播放程式」。如果要在多個視訊播放程式中使用註解功能，請為每個 VideoPlayer 建立一個 FLVPlaybackCaptioning 組件實體，並將 FLVPlaybackCaptioning videoPlayerIndex 設定為對應的索引。當只有一個 VideoPlayer 存在時，VideoPlayer 索引預設為 0。

下列程式碼範例會將唯一的註解功能指定給唯一的視訊。若要執行此範例，必須將範例中的虛構 URL 置換成實際的 URL。

```
captioner0.videoPlayerIndex = 0;
captioner0.source = "http://www.[yourDomain].com/mytimedtext0.xml";
flvPlayback.play("http://www.[yourDomain].com/myvideo0.flv");
captioner1.videoPlayerIndex = 1;
captioner1.source = "http://www.[yourDomain].com/mytimedtext1.xml";
flvPlayback.activeVideoIndex = 1;
flvPlayback.play ("http://www.[yourDomain].com/myvideo1.flv");
```

自訂 FLVPlaybackCaptioning 組件

如果想儘快使用 FLVPlaybackCaptioning 組件，您可以選擇使用 FLVPlaybackCaptioning 的預設值，直接將註解功能放入 FLVPlayback 組件。您也許想要自訂 FLVPlaybackCaptioning 組件來移除視訊的註解功能。

下列程式碼將示範如何使用切換註解功能按鈕動態建立 FLVPlayback 物件：

1. 將 FLVPlayback 組件放置在舞臺上 0,0 的位置，並提供實體名稱 **player**。
2. 將 FLVPlaybackCaptioning 組件放置在舞臺上 0,0 的位置，並提供實體名稱 **captioning**。
3. 將 CaptionButton 組件放到舞台上。
4. 在下列程式碼範例中，將 testVideoPath:String 變數設定成 FLV 檔 (使用絕對或相對路徑)。

   備註：此程式碼範例會將 testVideoPath 變數設成 Flash 視訊樣本 caption_video.flv。請將此變數改成您要加入註解按鈕組件之註解視訊組件的路徑。
5. 在下列程式碼範例中，將 testCaptioningPath:String 變數設定成適當的 Timed Text XML 檔 (使用絕對或相對路徑)。

   備註：此程式碼範例會將 testCaptioningPath 變數設成 Timed Text XML 檔 caption_video.xml。請將此變數改成內含視訊註解之 Timed Text XML 檔的路徑。
6. 將下列程式碼另存為 FLVPlaybackCaptioningExample.as，與 FLA 檔放在同一個目錄中。
7. 將 FLA 檔中的 DocumentClass 設定為 FLVPlaybackCaptioningExample。

```
package
{
    import flash.display.Sprite;
    import flash.text.TextField;
    import fl.video.FLVPlayback;
    import fl.video.FLVPlaybackCaptioning;

    public class FLVPlaybackCaptioningExample extends Sprite {

        private var testVideoPath:String = "http://www.helpexamples.com/flash/video/caption_video.flv";
        private var testCaptioningPath:String =
"http://www.helpexamples.com/flash/video/caption_video.xml";

        public function FLVPlaybackCaptioningExample() {
            player.source = testVideoPath;
            player.skin = "SkinOverAllNoCaption.swf";
            player.skinBackgroundColor = 0x666666;
            player.skinBackgroundAlpha = 0.5;

            captioning.flvPlayback = player;
            captioning.source = testCaptioningPath;
            captioning.autoLayout = false;
            captioning.addEventListener("captionChange",onCaptionChange);
        }
        private function onCaptionChange(e:*):void {
            var tf:* = e.target.captionTarget;
            var player:FLVPlayback = e.target.flvPlayback;

            // move the caption below the video
            tf.y = 210;
        }
    }
}
```

如需有關所有 FLVPlaybackCaptioning 參數的詳細資訊，請參閱「適用於 Adobe Flash Platform 的 ActionScript 3.0 參考」。