Использование компонентов ADOBE® ACTIONSCRIPT® 3.0

Adobe

Использование компонентов ActionScript™ 3.0

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

Содержание

Глава 1. Введение

Adobe® Flash® CS4 Professional является стандартным инструментом разработки для эффективной работы в Интернете. Компоненты являются кирпичиками, при помощи которых строятся функциональные интернет-приложения, обеспечивающие эффективность работы. *Компонент* — это фрагмент ролика с параметрами, позволяющими настраивать компонент либо во время разработки во Flash, либо при исполнении при помощи методов, свойств и событий Adobe® ActionScript®. Компоненты позволяют разработчикам многократно использовать коды и обмениваться ими, а также заключают в себе сложные функции, которые дизайнеры могут использовать и настраивать, не прибегая к ActionScript.

Компоненты позволяют легко и быстро построить надежные приложения, характеризующиеся последовательным поведением и единообразным внешним видом. В данном руководстве описывается процедура построения приложений с использованием компонентов Adobe ActionScript 3.0. В документе *Справочник по языку Adobe® ActionScript® 3.0 и компонентам* приводится описание API-интерфейса (интерфейса программирования приложений) каждого компонента.

Можно использовать компоненты, созданные Adobe®, загружать компоненты, созданные другими разработчиками, или создавать собственные.

Целевая аудитория

Данное руководство предназначено для разработчиков, которые разрабатывают приложения Flash и хотят использовать компоненты для ускорения процесса разработки. Вы уже должны быть знакомы с разработкой приложений во Flash и языком ActionScript.

Если у вас мало опыта в использовании ActionScript, то в документ можно добавлять компоненты, настроить их параметры в Инспекторе свойств или компонентов, а для обработки их событий использовать панель "Поведения". Например, можно назначить поведение "Перейти на веб-страницу" компоненту Button, открывающему URL в веб-обозревателе при нажатии кнопки, не используя код ActionScript.

Если же вы программист и хотите создавать более функциональные приложения, можно создавать компоненты в динамическом режиме, использовать ActionScript для задания свойств и вызывать методы во время выполнения, а также использовать модель прослушивателя для обработки событий.

Дополнительные сведения см. в разделе «Работа с компонентами» на странице 18.

Системные требования

Компоненты Flash не имеют системных требований, выходящих за рамки системных требований для Flash.

Для просмотра любого SWF-файла, в котором используются компоненты Flash CS3 или Flash CS4, требуется проигрыватель Adobe® Flash® Player 9.0.28.0 или более поздней версии. Кроме того, этот файл должен быть опубликован для ActionScript 3.0 (это можно задать через меню "Файл" > "Параметры публикации" на вкладке Flash).

Об этом документе

В данном документе подробно описывается использование компонентов для разработки приложений Flash. Предполагается, что вы уже имеете общее представление о Flash и ActionScript 3.0. Конкретная документация по Flash и связанным продуктам доступна отдельно.

Данный документ доступен в формате PDF и в виде интерактивной справки. Для просмотра интерактивной справки запустите Flash и выберите меню "Справка" > "Справка Flash" > "Использование компонентов Adobe ActionScript 3.0".

Информацию о Flash см. в следующих документах:

- *Использование Flash*
- *Программирование на Adobe ActionScript 3.0*
- *Справочник по языку Adobe ActionScript 3.0 и компонентам*

Условные обозначения

В данном руководстве приняты следующие условные обозначения:

- *Курсивный шрифт* указывает на значение, которое следует заменить (например, в пути к папке).
- □□□□□ □□□□ указывает на код ActionScript, включая названия методов и свойств.
- *Курсивный шрифт кода* указывает на элемент кода, который следует заменить (например, параметр ActionScript).
- **Полужирный шрифт** указывает на вводимое вами значение.

Термины, принятые в данном руководстве

В данном руководстве используются следующие термины:

при исполнении При выполнении кода в проигрывателе Flash Player.

во время разработки Во время работы в среде разработки Flash.

Дополнительные ресурсы

Помимо информации, содержащейся в данном руководстве, Adobe предоставляет регулярно обновляемые статьи, идеи по проектированию и примеры в Центре Adobe Developer и в Центре Adobe Design.

Дополнительные образцы компонентов можно найти на сайте www.adobe.com/go/learn_fl_samples_ru.

Центр Adobe Developer
Центр Adobe Developer предоставляет ресурсы с последними сведениями о ActionScript, статьи о реальных разработках приложений, а также информацию о важных возникающих проблемах. Центр разработчиков находится по адресу www.adobe.com/go/flash_devcenter_ru.

Центр Adobe Design

Получите последние сведения из области цифрового дизайна и анимационной графики. Просматривайте работы ведущих дизайнеров, узнавайте о новых тенденциях в дизайне и оттачивайте свое мастерство при помощи учебных руководств, основных рабочих процессов и продвинутых технологий. Дважды в месяц проверяйте наличие новых учебных руководств и статей, а также вдохновляющих образцов в галерее. Центр дизайна находится по адресу www.adobe.com/go/fl_designcenter_ru.

Глава 2. О компонентах ActionScript 3.0

Компоненты Adobe® ActionScript® 3.0 являются фрагментами роликов, которые имеют параметры для изменения их внешнего вида и поведения. Компонент может представлять собой простой элемент управления пользовательского интерфейса, например RadioButton или CheckBox, а может включать в себя содержимое, такое как List или DataGrid.

Эти компоненты позволяют легко и быстро построить надежные приложения ActionScript, характеризующиеся последовательным поведением и единообразным внешним видом. Вместо создания пользовательских кнопок, флажков и списков, вы можете использовать компоненты Flash, в качестве этих элементов управления. Для этого просто перетащите их с панели "Компоненты" в документ приложения. Можно также легко настроить внешний вид и функциональность этих компонентов в соответствии с требованиями вашего приложения.

Хотя все это возможно без высокого уровня понимания ActionScript, при помощи ActionScript 3.0 также можно изменять поведение компонентов или реализовывать новое. Каждый компонент имеет уникальный набор методов, свойств и событий ActionScript, которые составляют его *интерфейс программирования приложений* (API). API-интерфейс позволяет создавать компоненты и манипулировать ими во время выполнения приложения.

API-интерфейс также позволяет создавать собственные, пользовательские компоненты. Вы можете загрузить компоненты, разработанные членами сообщества Flash, с сервера Adobe Exchange по адресу www.adobe.com/go/flash_exchange_ru. Информация о создании компонентов доступна на сайте www.adobe.com/go/learn_fl_creating_components_ru.

Архитектура компонентов ActionScript 3.0 включает в себя классы, на которых основаны все компоненты, обложки и стили, которые позволяют настраивать внешний вид, модель обработки событий, управление фокусом, специальные возможности интерфейса и т. д.

Примечание. *Adobe Flash CS4 включает в себя компоненты ActionScript 2.0, а также компоненты ActionScript 3.0. Сочетание этих двух наборов компонентов невозможно. Для каждого приложения можно использовать либо один, либо другой набор компонентов. Flash CS4 представляет либо компоненты ActionScript 2.0, либо компоненты ActionScript 3.0 в зависимости от того, откроете вы файл ActionScript 2.0 или ActionScript 3.0. При создании нового документа Flash необходимо указать либо "Файл Flash (ActionScript 3.0)", либо "Файл Flash (ActionScript 2.0)". При открытии существующего документа Flash проверяет параметры публикации, чтобы определить, какой набор компонентов использовать. Информацию о компонентах ActionScript 2.0 см. в разделе "Использование компонентов Adobe® ActionScript® 2.0".*

Полный список компонентов Flash ActionScript 3.0 см. в разделе «Типы компонентов» на странице 6.

Преимущества использования компонентов

Компоненты позволяют разграничить процесс разработки приложения и процесс кодирования. Они позволяют разработчикам создавать функциональные возможности, которые дизайнеры могут использовать в приложениях. Разработчики могут заключить в компоненты часто используемые функциональные возможности, а дизайнеры могут настроить размер, местоположение и поведение компонентов путем изменения их параметров. Они также могут изменить внешний вид компонентов, отредактировав их графические элементы, или обложки.

Компоненты имеют такие общие функциональные возможности, как стили, обложки и управление фокусом. При добавлении первого компонента в приложение эти функциональные возможности занимают примерно 20 килобайт. При последующем добавлении компонентов исходно выделенный объем памяти распределяется между добавленными компонентами, сокращая увеличение размера вашего приложения.

В данном разделе описываются некоторые преимущества компонентов ActionScript 3.0.

Возможности ActionScript 3.0 предоставляют мощный, объектно-ориентированный язык программирования, который является важным шагом в развитии возможностей проигрывателя Flash Player. Этот язык предназначен для создания функциональных интернет-приложений на основе кодов многократного использования. ActionScript 3.0 основан на ECMAScript — международном стандартизированном языке сценариев — и соответствует спецификации языка ECMAScript (ECMA-262) Выпуск 3. Подробную информацию о языке ActionScript 3.0 см. в руководстве *Программирование на ActionScript 3.0*. Справочную информацию о языке см. в документе Справочник по языку ActionScript 3 и компонентам.

Компоненты пользовательского интерфейса на базе FLA обеспечивают легкий доступ к обложкам для настройки компонентов во время разработки. Эти компоненты также предоставляют стили, включая стили обложек, которые позволяют настраивать внешний вид компонентов и загружать обложки при исполнении. Дополнительную информацию см. в разделе «Настройка компонентов пользовательского интерфейса» на странице 103, а также в документе Справочник по языку ActionScript 3 и компонентам.

Новый компонент FVLPlayback содержит компонент FLVPlaybackCaptioning наряду с поддержкой полноэкранного режима, улучшенным интерактивным просмотром, обложками, позволяющими добавлять параметры цвета и альфа-каналов, а также улучшенную загрузку FLV и свойства макета.

Инспектор свойств и Инспектор компонентов позволяют изменять параметры компонентов при их разработке во Flash. Дополнительную информацию см. в разделе «Работа с файлами компонентов» на странице 20 и «Настройка параметров и свойств» на странице 23.

Диалоговое окно "Создать коллекцию" для компонентов ComboBox, List и TileList позволяет заполнить их свойство `dataProvider` при помощи пользовательского интерфейса. Дополнительную информацию см. в разделе «Создание объекта DataProvider» на странице 31.

Модель событий ActionScript 3.0 позволяет приложению прослушивать события и вызывать для них обработчики событий. Дополнительную информацию см. в разделах «Модель обработки событий ActionScript 3.0» на странице 10 и «Обработка событий» на странице 25.

Классы Manager позволяют легко управлять фокусом и стилями в приложениях. Дополнительную информацию см. в документе Справочник по языку ActionScript 3 и компонентам.

Базовый класс UIComponent содержит основные методы, свойства и события для компонентов, которые расширяют этот класс. Все компоненты пользовательского интерфейса ActionScript 3.0 наследуют от класса UIComponent. Дополнительную информацию см. в описании класса UIComponent в документе Справочник по языку ActionScript 3 и компонентам.

Использование SWC в компонентах пользовательского интерфейса на основе FLA использовать определения ActionScript в качестве актива во временной шкале компонента для ускорения компиляции.

Легко расширяемая при помощи ActionScript 3.0 иерархия класса позволяет легко создавать уникальные пространства имен, импортировать классы и создавать подклассы для расширения компонентов.

Дополнительную информацию см. в описании свойства Справочник по языку ActionScript 3 и компонентам.

Примечание. *Flash CS4 поддерживает как компоненты на базе FLA, так и компоненты на базе SWC. Дополнительную информацию см. в разделе «Архитектура компонентов» на странице 18.*

Типы компонентов

Компоненты Flash устанавливаются при установке Flash CS4.

Компоненты ActionScript 3.0 включают в себя следующие компоненты пользовательского интерфейса:

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

Помимо компонентов пользовательского интерфейса компоненты Flash ActionScript 3.0 включают в себя следующие компоненты и классы:

- Компонент FLVPlayback (fl.video.FLVPlayback), который является компонентом на базе SWC.

  Компонент FLVPlayback позволяет включить в приложение Flash видеопроигрыватель для воспроизведения последовательно загружаемых потоковых видеофайлов с использованием протокола HTTP, из службы Adobe® Flash® Video Streaming Service (FVSS) или с сервера Adobe Macromedia® Flash® Media Server (FMS). Дополнительную информацию см. в разделе «Использование компонента FLVPlayback» на странице 141.

- Компоненты пользовательского интерфейса для воспроизведения FLV-файлов на базе FLA, которые работают с компонентом FLVPlayback версий ActionScript 2.0 и ActionScript 3.0. Дополнительную информацию см. в разделе «Использование компонента FLVPlayback» на странице 141.

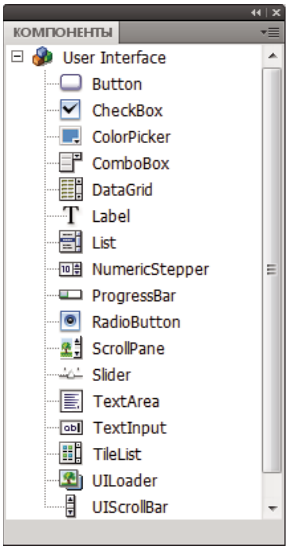
- Компонент FLVPlayback Captioning, который обеспечивает ввод кодированных субтитров между кадрами для компонента FLVPlayback. См. раздел «Использование компонента FLVPlaybackCaptioning» на странице 181.

  Полный список компонентов ActionScript 3.0 и поддерживающих классов см. в документе Справочник по языку ActionScript 3 и компонентам.

Просмотр компонентов Flash:

Вы можете просматривать компоненты Flash ActionScript 3.0 на панели "Компоненты", выполнив следующие шаги.

1. Запустите приложение Flash.
2. Создайте новый файл Flash (ActionScript 3.0) или откройте существующий документ Flash, в параметрах публикации которого указан ActionScript 3.0.
3. Выберите меню "Окно" > "Компоненты", чтобы открыть панель "Компоненты", если она еще не открыта.



Панель "Компоненты" с компонентами пользовательского интерфейса

Можно загрузить дополнительные компоненты с сервера Adobe Exchange по адресу www.adobe.com/go/flash_exchange_ru. Для установки компонентов, загруженных с сервера Exchange, загрузите и установите приложение Adobe® Extension Manager с сайта www.adobe.com/go/exchange_ru. Нажмите ссылку "Главная страница Adobe Exchange" и найдите ссылку на приложение Extension Manager.

На панели "Компоненты" во Flash может появиться любой компонент. Выполните следующие действия, чтобы установить компоненты на компьютер с ОС Windows® или Macintosh®.

Установка компонентов на компьютер с ОС Window или Macintosh:

1. Закройте приложение Flash.
2. Поместите SWC- или FLA-файл, содержащий компонент, в следующую папку на жестком диске:
   - В Windows:

     C:\Program Files\Adobe\Adobe Flash CS4*language*\Configuration\Components
   - В Macintosh:

     Macintosh HD:Applications:Adobe Flash CS4:Configuration:Components
3. Запустите приложение Flash.
4. Выберите меню "Окно" > "Компоненты" для просмотра компонента на панели "Компоненты", если она еще не открыта.

   Дополнительную информацию о файлах компонентов см. в разделе «Работа с файлами компонентов» на странице 20

Вставка в документ и удаление из него

При перетаскивании компонента на базе FLA с панели "Компоненты" в рабочую область Flash импортирует редактируемый фрагмент ролика в библиотеку. При перетаскивании компонента на базе SWC в рабочую область Flash импортирует скомпилированный фрагмент в библиотеку. После импорта компонента в библиотеку можно перетащить его экземпляры в рабочую область либо с панели "Библиотека", либо с панели "Компоненты".

Вставка компонентов во время разработки

Компонент можно вставить в документ, перетащив его с панели "Компоненты". Можно задать свойства для каждого экземпляра компонента в Инспекторе свойств или на вкладке "Параметры" в Инспекторе компонентов.

1. Выберите меню "Окно" > "Компоненты".
2. Либо дважды щелкните компонент на панели "Компоненты", либо перетащите его в рабочую область.
3. Выберите компонент в рабочей области.
4. Если Инспектор свойств не виден, выберите меню "Окно" > "Свойства" > "Свойства".
5. В Инспекторе свойств введите имя экземпляра компонента.
6. Выберите меню "Окно" > "Инспектор компонентов" и перейдите на вкладку "Параметры", чтобы задать параметры экземпляра.

   Дополнительную информацию см. в разделе «Настройка параметров и свойств» на странице 23.
7. Можно изменить размер компонента, изменив значения свойств ширины (W:) и высоты (H:).

   Дополнительную информацию об изменении размера определенных типов компонентов см. в разделе «Настройка компонентов пользовательского интерфейса» на странице 103.
8. Выберите меню "Управление" > "Тестировать ролик" или нажмите Ctrl+Enter для компиляции документа и просмотра результатов настроек параметров.

   Можно также изменить цвет и форматирование текста компонентов, задав свойства стиля, или настроить внешний вид компонентов, отредактировав их обложки. Дополнительную информацию по этой теме см. в разделе «Настройка компонентов пользовательского интерфейса» на странице 103.

   Если перетащить компонент в рабочую область во время разработки, сослаться на компонент можно при помощи имени его экземпляра (например, `myButton`).

Добавление компонентов при исполнении с использованием ActionScript

Для добавления компонента в документ при исполнении с использованием ActionScript его необходимо сначала поместить в библиотеку приложения ("Окно" > "Библиотека") при компиляции SWF-файла. Для добавления компонента в библиотеку перетащите его с панели "Компоненты" на панель "Библиотека". Дополнительную информацию о библиотеке см. в разделе «Библиотека» на странице 24.

Необходимо также импортировать файл класса компонента, чтобы сделать его API-интерфейс доступным для вашего приложения. Файлы классов компонентов устанавливаются в виде *пакетов*, содержащих один или более классов. Чтобы импортировать класс компонента, используйте выражение `import` и укажите имя пакета и класса. Класс Button, например, импортируется при помощи следующей инструкции `import`:

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

Информацию о том, в каком пакете находится компонент, см. в документе Справочник по языку ActionScript 3 и компонентам. Информацию о местоположении исходных файлов компонентов см. в разделе «Работа с файлами компонентов» на странице 20.

Чтобы создать экземпляр компонента, необходимо вызвать метод конструктора ActionScript компонента. Например, следующее выражение создает экземпляр компонента Button с именем `aButton`:

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

Завершающим шагом является вызов метода `addChild()` для добавления экземпляра компонента в рабочую область или контейнер приложения. Например, следующее выражение добавляет экземпляр `aButton`:

```
addChild(aButton);
```

На данном этапе можно использовать API-интерфейс компонента для динамического указания размера и расположения компонента в рабочей области, прослушивания событий и задания свойств для изменения поведения компонента. Дополнительную информацию об API-интерфейсе конкретного компонента см. в документе Справочник по языку ActionScript 3 и компонентам.

Дополнительную информацию о методе `addChild()` см. в разделе «Работа со списком отображения» на странице 27.

Удаление компонента

Чтобы удалить экземпляр компонента из рабочей области во время разработки, просто выделите его и нажмите кнопку "Удалить". При этом будет удален экземпляр из рабочей области, но сам компонент не будет удален из приложения.

Чтобы удалить компонент из документа Flash после его помещения в рабочую область или на панель "Библиотека", необходимо удалить этот компонент и связанные с ним активы из библиотеки. Недостаточно просто удалить компонент из рабочей области. Если не удалить компонент из библиотеки, он будет включен в приложение при его компиляции.

1. На панели "Библиотека" выберите символ компонента.
2. Нажмите кнопку "Удалить" внизу панели "Библиотека" или выберите пункт "Удалить" в меню панели "Библиотека".

   Повторите эти шаги для удаления всех активов, связанных с компонентом.

   Информацию о том, как удалить компонент из его контейнера во время работы приложения, см. в разделе «Удаление компонента из списка отображения» на странице 28.

Поиск версии компонента

Компоненты Flash ActionScript 3.0 имеют свойство version, которое можно отобразить, если нужно предоставить эту информацию Службе технической поддержки Adobe или просто узнать, какую версию компонента вы используете.

Отображение номера версии компонента пользовательского интерфейса:

1. Создайте новый файл Flash (ActionScript 3.0).
2. Перетащите компонент в рабочую область и присвойте ему имя экземпляра. Например, перетащите компонент ComboBox в рабочую область и назовите его **aCb**.

3 Нажмите клавишу **F9** или выберите меню "Окна" > "Действия", чтобы открыть панель "Действия".

4 Щелкните Кадр 1 на основной временной шкале и вставьте следующий код на панель "Действия":

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

Номер версии, подобный тому, что показан на рисунке ниже, должен отобразиться на панели "Вывод".

В отношении компонентов FLVPlayback и FLVPlaybackCaptioning поиск версии необходимо выполнять по имени класса, а не по имени экземпляра, так как номер версии хранится в константе класса.

Отображение номера версии компонентов FLVPlayback и FLVPlaybackCaptioning:

1 Создайте новый файл Flash (ActionScript 3.0).

2 Перетащите компоненты FLVPlayback и FLVPlaybackCaptioning на панель "Библиотека".

3 Нажмите клавишу **F9** или выберите меню "Окна" > "Действия", чтобы открыть панель "Действия".

4 Щелкните Кадр 1 на основной временной шкале и вставьте следующий код на панель "Действия".

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

Номера версий появятся на панели "Вывод".

Модель обработки событий ActionScript 3.0

ActionScript 3.0 представляет единую модель обработки событий, которая заменяет различные механизмы обработки событий, существовавшие в предыдущих версиях ActionScript. Новая модель событий основана на модели Document Object Model (DOM) Level 3 Events Specification.

Для разработчиков с опытом использования метода ActionScript 2.0 `addListener()` полезно выделить различия между моделью прослушивания событий в ActionScript 2.0 и моделью событий в ActionScript 3.0. В следующем списке приведены основные различия двух моделей событий:

- Для добавления прослушивателей событий в ActionScript 2.0 в некоторых случаях используется метод `addListener()`, а в других метод `addEventListener()`, тогда как в ActionScript 3.0 во всех случаях используется метод `addEventListener()`.
- В ActionScript 2.0 отсутствует поток событий, поэтому метод `addListener()` можно вызвать только для объекта, который рассылает событие, тогда как в ActionScript 3.0 метод `addEventListener()` можно вызвать для любого объекта, который является участником потока событий.
- В ActionScript 2.0 прослушиватели событий могут быть функциями, методами или объектами, тогда как в ActionScript 3.0 только функции или методы могут быть прослушивателями событий.
- Синтаксис `on(`*event*`)` не поддерживается в ActionScript 3.0, поэтому к фрагменту ролика нельзя прикрепить код события ActionScript. Добавить прослушиватель событий можно только при помощи метода `addEventListener()`.

  Следующий пример, который прослушивает событие `MouseEvent.CLICK` для компонента Button с именем `aButton`, иллюстрирует основную модель обработки событий в ActionScript 3.0:

```
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);
}
```

Дополнительную информацию об обработке событий в ActionScript 3.0 см. в руководстве *Программирование на ActionScript 3.0*. Дополнительную информацию по обработке событий компонентов в ActionScript 3.0 см. в разделе «Обработка событий» на странице 25.

Простое приложение

В данном разделе описывается процедура создания простого приложения ActionScript 3.0 с использованием компонентов Flash и инструмента разработки Flash. Приведен пример использования FLA-файла с кодом ActionScript на временной шкале, а также использования внешнего файла класса ActionScript с FLA-файлом, содержащим только компоненты из библиотеки. Как правило, возникает необходимость создания более сложных приложений с использованием внешних файлов класса, чтобы можно было обмениваться кодом между классами и приложениями и сделать ваши приложения более легкими в обслуживании. Дополнительную информацию о программировании на ActionScript 3.0 см. в руководстве *Программирование на ActionScript 3.0*.

Проектирование приложения

Первым примером приложения с использованием компонентов ActionScript является вариант стандартного приложения "Hello World", дизайн которого довольно прост:

- Приложение будет называться Greetings.
- Компонент TextArea используется для отображения приветствия, которое исходно звучит как Hello World.
- Компонент ColorPicker позволяет изменять цвет текста.
- Три компонента RadioButton позволяют задавать маленький, средний и большой размер текста.
- Компонент ComboBox позволяет выбирать различные приветствия в раскрывающемся списке.
- В приложении используются компоненты, приведенные на панели "Компоненты", а также создаются другие элементы приложения при помощи кода ActionScript.

  Определившись с компонентами, можно приступать к созданию приложения.

Создание приложения Greetings

Следующие шаги позволяют создать приложение Greetings, используя инструмент разработки Flash для создания FLA-файла, поместить компоненты в рабочую область и вставить код ActionScript во временную шкалу.

Создание приложения Greetings в FLA-файле:

1. Выберите "Файл" > "Создать".
2. В диалоговом окне "Новый документ" выберите "Файл Flash (ActionScript 3.0)" и нажмите кнопку "ОК".

   Откроется новое окно Flash.
3. Выберите "Файл" > "Сохранить", присвойте файлу Flash имя **Greetings.fla** и нажмите кнопку "Сохранить".

4 На панели Flash "Компоненты" выберите компонент TextArea и перетащите его в рабочую область.

5 В окне "Свойства" при выбранном компоненте TextArea в рабочей области введите **aTa** в качестве имени экземпляра и введите следующую информацию:

- Введите **230** для значения W (ширина).
- Введите **44** для значения H (высота).
- Введите **165** для значения X (положение по горизонтали).
- Введите **57** для значения Y (положение по вертикали).
- Введите **Hello World!** в качестве значения параметра text на вкладке "Параметры".

6 Перетащите компонент ColorPicker в рабочую область, поместите его слева от компонента TextArea и присвойте ему имя экземпляра **txtCp.** Введите следующую информацию в Инспекторе свойств:

- Введите **96** для значения X.
- Введите **72** для значения Y.

7 Перетащите по одному три компонента RadioButton в рабочую область и присвойте им имена экземпляров **smallRb**, **largerRb** и **largestRb**. Введите для них следующую информацию в Инспекторе свойств:

- Введите **100** для значения W и **22** для значения H для каждого из этих компонентов.
- Введите **155** для значения X.
- Ведите **120** для значения Y для smallRb, **148** для largerRb и **175** для largestRb.
- Введите **fontRbGrp** для параметра groupName каждого из компонентов.
- На вкладке "Параметры" введите метки компонентов: **Маленький**, **Средний**, **Большой**.

8 Перетащите компонент ComboBox в рабочую область и присвойте ему имя экземпляра **msgCb**. Введите для него следующую информацию в Инспекторе свойств:

- Введите **130** для значения W.
- Введите **265** для значения X.
- Введите **120** для значения Y.
- На вкладке "Параметры" введите **Greetings** в качестве значения параметра prompt.
- Дважды щелкните текстовое поле параметра dataProvider, чтобы открыть диалоговое окно "Значения".
- Щелкните знак "плюс" и замените значение метки на **Hello World!**
- Повторите предыдущий шаг для добавления значений метки **Have a nice day!** и **Top of the Morning!**
- Нажмите кнопку "OK", чтобы закрыть диалоговое окно "Значения".

9 Сохраните файл.

10 Если панель "Действия" еще не открыта, откройте ее, нажав клавишу **F9** или выбрав пункт "Действия" в меню "Окно". Щелкните Кадр 1 на основной временной шкале и вставьте следующий код на панель "Действия":

```
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);
```

Первые три строки импортируют классы событий, используемые приложением. Событие возникает при взаимодействии пользователя с одним из компонентов. Следующие пять строк регистрируют обработчики событий для событий, которые будет прослушивать приложение. Событие `click` возникает для компонента RadioButton, когда пользователь щелкает его мышью. Событие `change` возникает, когда пользователь выбирает другой цвет в компоненте ColorPicker. Событие `change` для компонента ComboBox возникает, когда пользователь выбирает другое приветствие в раскрывающемся списке.

Четвертая строка импортирует класс RadioButtonGroup, чтобы приложение могло назначить прослушивателя событий группе компонентов RadioButton вместо назначения прослушивателя каждой кнопке в отдельности.

11 Вставьте следующий код на панель "Действия" для создания объекта TextFormat `tf`, при помощи которого приложение изменяет свойства стиля `size` и `color` текста компонента TextArea.

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

12 Вставьте следующий код для создания функции обработки событий `rbHandler`. Эта функция обрабатывает событие `click`, когда пользователь щелкает мышью один из компонентов RadioButton.

```
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);
}
```

Функция использует выражение `switch` для проверки свойства `target` объекта `event`, чтобы определить, какой из переключателей вызвал событие. Свойство `currentTarget` содержит имя объекта, вызвавшего событие. В зависимости от того, какой из переключателей нажал пользователь, приложение изменяет размер текста компонента TextArea на 14, 18 или 24 пункта.

13 Введите следующий код для выполнения функции `cpHandler()`, которая обрабатывает изменение значения в компоненте ColorPicker:

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

Данная функция устанавливает свойство `color` объекта TextFormat `tf` на выбранный в компоненте ColorPicker цвет, затем вызывает метод `setStyle()`, чтобы применить цвет к тексту экземпляра `aTa` компонента TextArea.

14 Введите следующий код для выполнения функции `cbHandler()` которая обрабатывает изменение выбора в компоненте ComboBox:

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

Эта функция просто заменяет текст в компоненте TextArea текстом, выбранным в компоненте ComboBox: `event.target.selectedItem.label`.

15 Выберите меню "Управление" > "Тестировать ролик" или нажмите Ctrl+Enter для компиляции кода и тестирования приложения Greetings.

Далее описывается процедура построения того же приложения с использованием внешнего класса ActionScript и FLA-файла, который имеет только требуемые компоненты в библиотеке.

Создание приложения Greetings2 с использованием внешнего файла класса:

1 Выберите "Файл" > "Создать".

2 В диалоговом окне "Новый документ" выберите "Файл Flash (ActionScript 3.0)" и нажмите кнопку "ОК".

Откроется новое окно Flash.

3 Выберите "Файл" > "Сохранить", присвойте файлу Flash имя **Greetings2.fla** и нажмите кнопку "Сохранить".

4 Перетащите каждый из следующих компонентов с панели "Компоненты" в библиотеку:

- ColorPicker
- ComboBox
- RadioButton
- TextArea

В скомпилированном SWF-файле будет использоваться каждый из этих активов, поэтому их необходимо добавить в библиотеку. Перетащите компоненты в нижнюю часть панели "Библиотека". При добавлении этих компонентов в библиотеку другие активы (например, List, TextInput и UIScrollBox) добавляются автоматически.

5 В окне "Свойства" в поле "Класс документа" введите **Greetings2**.

Если Flash отобразит предупреждение о том, что "определение класса документа не найдено", игнорируйте его. Определение класса Greetings2 будет выполнено в последующих шагах. Данный класс определяет основную функциональность приложения.

6 Сохраните файл Greetings2.fla.

7 Выберите "Файл" > "Создать".

8 В диалоговом окне "Новый документ" выберите "Файл ActionScript" и нажмите кнопку "ОК".

Откроется новое окно сценария.

9 Вставьте следующий код в окно сценария:

```
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() {
```

Сценарий определяет класс ActionScript 3.0 под названием Greetings2. Сценарий выполняет следующее:

- Импортирует классы, которые будут использоваться в файле. Обычно данные инструкции `import` добавляются по мере ссылки на различные классы в коде, но для краткости данный пример импортирует их все сразу.
- Объявляет переменные, представляющие различные типы объектов компонентов, которые будут добавлены в код. Другая переменная создает объект TextFormat `tf`.
- Определяет функцию конструктора, `Greetings2()`, для класса. В последующих шагах мы добавим строки к этой функции и добавим другие методы к классу.

10 Выберите "Файл" > "Сохранить", присвойте файлу имя **Greetings2.as** и нажмите кнопку "Сохранить".

11 Вставьте следующие строки кода в функцию `Greeting2()`:

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

Функция теперь должна выглядеть следующим образом:

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

12 Вставьте следующие строки кода после закрывающей скобки метода `Greeting2()`:

```
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);
}
```

Эти строки выполняют следующее:

- Создают экземпляры компонентов, используемых в приложении.
- Задают размер, положение и свойства каждого компонента.

- Добавляют каждый компонент в рабочую область при помощи метода `addChild()`.

13 После закрывающей скобки метода `bldRadioButtons()` добавьте следующий код метода `setUpHandlers()`:

```
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;
}
}
}
```

Эти функции определяют прослушивателей событий для компонентов.

14 Выберите "Файл" > "Сохранить", чтобы сохранить файл.

15 Выберите меню "Управление" > "Тестировать ролик" или нажмите Ctrl+Enter для компиляции кода и тестирования приложения Greetings2.

Разработка и выполнение последующих примеров

Разработав и выполнив приложение Greetings вы получили основные знания, необходимые для выполнения других примеров кода, приведенных в данной книге. Соответствующий код ActionScript 3.0 в каждом примере будет выделен и объяснен, а вы сможете вырезать и вставить каждый пример в этой книге в FLA-файл, скомпилировать и выполнить его.

Глава 3. Работа с компонентами

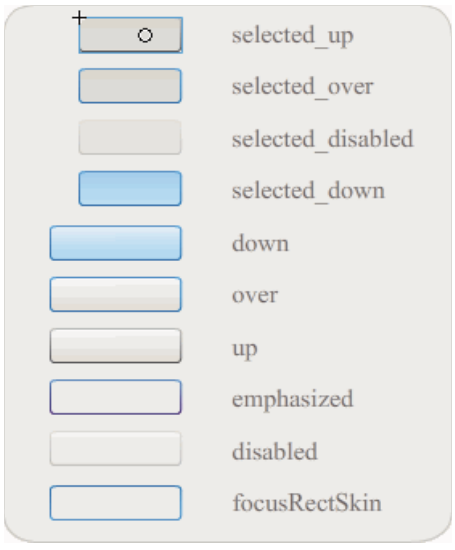
Архитектура компонентов

Компоненты Adobe® ActionScript 3.0® поддерживаются проигрывателем Adobe® Flash Player 9.0.28.0 и более поздними версиями. Эти компоненты несовместимы с компонентами, созданными до появления Flash CS4. Дополнительные сведения об использовании компонентов Adobe® ActionScript® 2.0 см. в документах *Использование компонентов Adobe® ActionScript® 2.0* и *Справочник по компонентам языка Adobe® ActionScript® 2.0*.

Компоненты пользовательского интерфейса Adobe ActionScript 3.0 реализуются как компоненты на базе FLA, но Flash CS4 поддерживает компоненты на базе как SWC, так и FLA. К примеру, компоненты FLVPlayback и FLVPlaybackCaptioning создаются на базе SWC. Компоненты любого типа можно поместить в папку "Компоненты", чтобы они появились на панели "Компоненты". Эти два типа компонентов создаются по-разному, поэтому каждый из них будет описан отдельно.

Компоненты ActionScript 3.0 на базе FLA

Компоненты пользовательского интерфейса ActionScript 3.0 представляют собой файлы на базе FLA (.fla) со встроенными обложками, которые можно открыть для редактирования двойным щелчком по компоненту в рабочей области. Обложки компонента и другие ресурсы помещаются в кадре 2 временной шкалы. При двойном щелчке по компоненту Flash автоматически переходит к кадру 2 и открывает палитру обложек компонента. На следующем рисунке показана палитра обложек, отображаемых для компонента Button.



Обложки для компонента Button

Дополнительные сведения об обложках и настройке компонентов см. в разделах «Настройка компонентов пользовательского интерфейса» на странице 103 и «Настройка компонента FLVPlayback» на странице 161.

Чтобы ускорить компиляцию для приложений и избежать конфликтов с параметрами ActionScript 3.0, компоненты пользовательского интерфейса Flash CS4 на базе FLA также содержат SWC с уже скомпилированным кодом ActionScript компонента. ComponentShim SWC помещается в рабочей области в кадре 2 каждого компонента пользовательского интерфейса, чтобы сделать доступными предварительно скомпилированные определения. Чтобы быть доступным для ActionScript, компонент должен быть либо в рабочей области, либо в библиотеке, а в его свойствах связывания должен быть выбран параметр "Экспортировать в первый кадр". Чтобы создать компонент с помощью ActionScript, необходимо также импортировать класс с помощью оператора `import`. Дополнительные сведения об операторе `import` см. в *Справочнике по языку ActionScript 3.0 и компонентам*.

Компоненты на базе SWC

Компоненты на базе SWC также имеют FLA-файл и файл класса ActionScript, но они скомпилированы и экспортированы как SWC. SWC-файл — это пакет предварительно скомпилированных символов Flash и кода ActionScript, который позволяет избежать повторной компиляции неизменяемых символов и кода.

FLVPlayback и FLVPlaybackCaptioning являются компонентами на базе SWC. Они имеют внешние, а не встроенные обложки. Компонент FLVPlayback имеет обложку по умолчанию, которую можно изменить, выбрав другую из коллекции готовых обложек, настроив элементы управления пользовательского интерфейса на панели "Компоненты" (BackButton, BufferingBar и т. д.) или создав собственную обложку. Дополнительные сведения см. в разделе «Настройка компонента FLVPlayback» на странице 161.

В Flash можно преобразовать фрагмент ролика в скомпилированный фрагмент следующим образом.

Компиляция фрагмента ролика

- Правой кнопкой мыши (Windows) или мышью при нажатой клавише "Control" (Macintosh) щелкните фрагмент ролика на панели "Библиотека" и выберите "Преобразовать в скомпилированный фрагмент".

  Скомпилированный фрагмент ведет себя так же, как и его исходный фрагмент ролика, только скомпилированные фрагменты публикуются намного быстрее, чем обычные фрагменты роликов. Скомпилированные фрагменты нельзя редактировать, но их свойства отображаются в Инспекторе свойств и Инспекторе компонентов.

  Компоненты SWC содержат скомпилированный клик, предварительно скомпилированные определения ActionScript компонента и другие файлы, описывающие компонент. Если вы создаете собственный компонент, его можно экспортировать в виде SWC-файла для распределения.

Экспорт SWC-файла

- Выберите фрагмент ролика на панели "Библиотека", щелкните его правой кнопкой мыши (Windows) или удерживая клавишу "Control" (Macintosh), а затем выберите "Экспортировать SWC-файл".

  Примечание. *Формат SWC-файла версии Flash CS4 или более поздней совместим с форматом Flex SWC, что позволяет использовать SWC-файлы в обоих продуктах, но, возможно, с изменениями.*

Дополнительные сведения о создании компонентов на базе SWC см. на странице www.adobe.com/go/learn_fl_creating_components_ru.

API-интерфейс компонентов ActionScript 3.0

Каждый компонент ActionScript 3.0 строится на базе класса ActionScript 3.0, который находится в папке пакета и имеет имя в формате fl.*имя_пакета.имя_класса*. Например, компонент Button является экземпляром класса Button и имеет имя пакета `fl.controls.Button`. При импорте класса компонентов в приложение необходимо ссылаться на имя пакета. Класс Button импортируется с помощью следующего оператора:

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

Дополнительные сведения о местоположении файлов с классами компонентов см. в разделе «Работа с файлами компонентов» на странице 20.

Класс компонента определяет методы, свойства, события и стили, позволяющие взаимодействовать с ним в приложении. Компоненты пользовательского интерфейса ActionScript 3.0 являются подклассами классов Sprite и UIComponent и наследуют их свойства, методы и события. Класс Sprite является основным элементом для создания списка отображения, он похож на класс MovieClip но не имеет временной шкалы. Класс UIComponent является базовым для всех визуальных компонентов, как интерактивных, так и неинтерактивных. Путь наследования каждого компонента, а также свойства, методы, события и стили, описаны в *справочнике по языку ActionScript 3.0 и компонентам*.

Все компоненты ActionScript 3.0 используют модель обработки событий ActionScript 3.0. Дополнительные сведения об обработке событий см. в документах «Обработка событий» на странице 25 и *Программирование на ActionScript 3.0*.

Работа с файлами компонентов

В этом разделе объясняется, где хранятся файлы компонентов, где можно найти исходные файлы ActionScript и как добавлять и удалять компоненты с панели "Компоненты".

Где хранятся файлы компонентов

Компоненты Flash хранятся в папке "Configuration" в каталоге приложения.

Примечание. *Дополнительные сведения об этих папках см. в разделе "Папки Configuration, установленные вместе с Flash" руководств "Использование Flash".*

Компоненты устанавливаются в следующих местоположениях:

- Windows 2000 или Windows XP: C:\Program Files\Adobe\Adobe Flash CS4*язык*\Configuration\Components
- Mac OS X: Macintosh HD:Applications:Adobe Flash CS4:Configuration:Components

  В папке "Components" компоненты пользовательского интерфейса находятся в файле User Interface.fla, а компоненты FLVPlayback (FLVPlaybackAS3.swc) и FLVPlaybackCaptioning — в папке "Video".

  Кроме того, компоненты можно хранить в следующих местоположениях в каталоге пользователя:

- Windows 2000 или Windows XP: C:\Documents and Settings*имя_пользователя*\Local Settings\Application Data\Adobe\Adobe Flash CS4\ru\Configuration\Components
- Windows Vista: C:\Users*имя_пользователя*\Local Settings\Application Data\Adobe\Adobe Flash CS4\ru\Configuration\Components

  Примечание. *В Windows папка "Application Data" скрыта по умолчанию. Чтобы показать скрытые папки и файлы, выберите "Мой компьютер", чтобы открыть проводник Windows, выберите "Сервис" > "Свойства папки" и перейдите на вкладку "Вид". На это вкладке выберите переключатель "Показывать скрытые файлы и папки".*

- Mac OS X: Macintosh HD:Users:<имя_пользователя>:Library:Application Support:Adobe Flash CS4:Configuration:Components

Где хранятся исходные файлы компонентов

Файлы классов ActionScript (.as) (или *исходные файлы*) для компонентов установлены в следующих папках приложения в ОС Windows 2000 или Windows XP:

компоненты пользовательского интерфейса C:\Program Files\Adobe\Adobe Flash CS4\ru\Configuration\Component Source\ActionScript 3.0\User Interface\fl

FLVPlayback C:\Program Files\Adobe\Adobe Flash CS4\ru\Configuration\Component Source\ActionScript 3.0\FLVPlayback\fl\video

FLVPlaybackCaptioning C:\Program Files\Adobe\Adobe Flash CS4\ru\Configuration\Component Source\ActionScript 3.0\FLVPlaybackCaptioning\fl\video

В ОС Mac OS X исходные папки компонентов находятся в следующих местоположениях:

компоненты пользовательского интерфейса Macintosh HD:Applications:Adobe Flash CS4:Configuration:Component Source:ActionScript 3.0:User Interface:fl

FLVPlayback Macintosh HD:Applications:Adobe Flash CS4:Configuration:Component Source:ActionScript 3.0:FLVPlayback:fl:video

FLVPlaybackCaptioning Macintosh HD:Applications:Adobe Flash CS4:Configuration:Component Source:ActionScript 3.0:FLVPlaybackCaptioning:fl:video

Исходные папки компонентов и путь к классам

Так как код компонентов ActionScript 3.0 предварительно скомпилирован, не требуется указывать местоположение файлов классов ActionScript в переменной пути к классам Classpath. Если же указать их местоположение в Classpath, это увеличит время, затрачиваемое на компиляцию приложений. Однако, если Flash находит файлы классов компонентов в параметре Classpath, файл класса всегда имеет более высокий приоритет, чем скомпилированный код компонента.

Добавлять местоположение исходных файлов компонентов в переменную Classpath следует только для отладки приложения с компонентами. Дополнительные сведения см. разделе «Отладка приложений с компонентами» на странице 22.

Изменение файлов компонентов

При обновлении, добавлении или удалении компонентов на базе SWC или при добавлении новых компонентов на базе FLA в ПО Flash, их необходимо повторно загрузить на панель "Компоненты", чтобы они стали доступными. Чтобы повторно загрузить компоненты, можно перезапустить Flash или выбрать "Перезагрузить" в меню панели "Компоненты". В результате этого Flash сделает доступными все компоненты, добавленные в папку "Components".

Перезагрузка компонентов с панели "Компоненты" в ходе работы Flash:

- Выберите "Перезагрузить" в меню панели "Компоненты".

Удаление компонента с панели "Компоненты":

- Удалите FLA-, SWC- или MXP-файл с панели "Компоненты", а затем перезапустите Flash или выберите "Перезагрузить" в меню панели "Компоненты". MXP-файл — это файл компонента, загруженный из Adobe Exchange.

  Компоненты на базе SWC можно удалять и заменять в процессе работы Flash и перезагружать панель "Компоненты", чтобы отобразить изменения. При удалении компонентов на базе FLA изменения отображаются только после завершения работы и повторного запуска Flash. Однако при добавлении компонентов на базе FLA обновить панель "Компоненты" можно с помощью команды "Перезагрузить".

  Компания Adobe рекомендует сохранять копию файлов компонентов Flash (.fla или .as), прежде чем вносить в них изменения. Это позволит восстановить их в случае необходимости.

Отладка приложений с компонентами

Компоненты ActionScript 3.0 содержат весь свой исходный код, что сокращает время, затрачиваемое на компиляцию приложения. Однако отладчик Flash не может проверять код внутри скомпилированных фрагментов. Поэтому, если требуется выполнить полную отладку приложения с анализом исходного кода компонентов, в настройке пути к классам необходимо указать исходные файлы компонентов.

Местоположение папок пакета компонентов указывается относительно местоположения исходных файлов для типа компонента. Чтобы добавить ссылку на все исходные файлы ActionScript 3.0 для всех компонентов пользовательского интерфейса, укажите следующее местоположение в поле "Путь к классам" для пакетов пользовательского интерфейса:

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

Примечание. *Это переопределит скомпилированный код для всех компонентов пользовательского интерфейса и увеличит время компиляции для приложения. Если по какой-либо причине исходный файл компонента был изменен, его поведение может измениться.*

Чтобы задать путь к классам, выберите "Настройки" в меню "Правка", затем выберите "ActionScript" из списка "Категория" и нажмите кнопку "Параметры ActionScript 3.0". Чтобы добавить новую запись, щелкните знак "+" над окном с текущими параметрами.

Переменная `$(AppConfig)` ссылается на папку "Flash CS4 Configuration" в каталоге установки Flash CS4. Как правило, путь выглядит следующим образом:

для Windows 2000 или Windows XP

```
C:\Program Files\Adobe\Adobe Flash CS4\language\Configuration\
```

для Mac OS X

```
Macintosh HD:Applications:Adobe Flash CS4:Configuration:
```

Примечание. *Если необходимо внести изменения в исходный файл компонента, компания Adobe настоятельно рекомендует сохранить копию файла в другом местоположении и добавить это местоположение в путь к классам.*

Дополнительные сведения о местоположении исходных файлов компонентов см. в разделе «Где хранятся исходные файлы компонентов» на странице 21.

Настройка параметров и свойств

Каждый компонент имеет параметры, которые можно настраивать, чтобы изменять его вид и поведение. Параметр — это свойство класса компонента, которое отображается в Инспекторах свойств и компонентов. Самые употребительные свойства представлены как параметры разработки, а остальные необходимо настраивать с помощью ActionScript. Все параметры, которые можно настраивать в ходе разработки, можно задавать и с помощью ActionScript. Настройка параметра с использованием ActionScript переопределяет значение, заданное при разработке.

Большинство компонентов пользовательского интерфейса ActionScript 3.0 наследуют свойства и методы от класса UIComponent, а также от базового класса. Например, классы Button и CheckBox наследуют свойства и от класса UIComponent и от класса BaseButton. Для настройки доступны унаследованные свойства компонента, а также его собственные свойства класса. Например, компонент ProgressBar наследует свойство `ProgressBar.enabled` от класса UIComponent, но также имеет собственное свойство `ProgressBar.percentComplete`. Оба эти свойства доступны для взаимодействия с экземпляром компонента ProgressBar. Дополнительные сведения о свойствах компонента см. в описании класса в Справочнике по языку ActionScript 3.0 и компонентам.

Параметры для экземпляра компонента можно настроить в Инспекторе свойств или в Инспекторе компонентов.

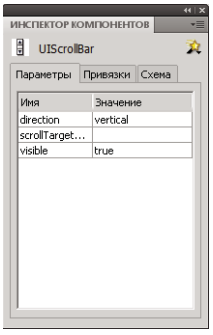
Ввод имени экземпляра для компонента в Инспекторе свойств:

1. Выберите "Окно" > "Свойства" > "Свойства".
2. Выберите экземпляр компонента в рабочей области.
3. Введите имя для экземпляра компонента в поле <Имя экземпляра>, которое находится под раскрывающемся списком "Фрагмент ролика". Перейдите на вкладку "Параметры" и введите имя в поле под заголовком *Компонент*. Введите значения для любых параметров, которые нужно настроить.

   Лучше добавить суффикс к имени экземпляра, чтобы указать тип компонента. Это упростит чтение кода ActionScript. Например, имя экземпляра **licenseSb** указывает на то, что это полоса прокрутки ScrollBar, которая прокручивает лицензионное соглашение в текстовой области **licenseTa** (TextArea).

Ввод параметров для экземпляра компонента с помощью Инспектора компонентов:

1. Выберите меню "Окно" > "Инспектор компонентов".
2. Выберите экземпляр компонента в рабочей области.
3. Перейдите на вкладку "Параметры" и задайте необходимые настройки.



Параметры компонента в Инспекторе компонентов

Настройка свойств компонента в ActionScript

В ActionScript используется оператор "точка" (.) (синтаксис записи через точку) для доступа к свойствам или методам объекта или экземпляра в рабочей области. Выражение синтаксиса с точкой начинается с имени экземпляра, за которым следует точка, а в конце стоит элемент, который нужно указать. Например, следующий код ActionScript задает свойство `width` экземпляра CheckBox с именем `aCh`, чтобы его ширина равнялась 50 пикселам:

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

Следующий оператор `if` проверяет, установил ли пользователь флажок:

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

Библиотека

При добавлении компонента в документ проигрыватель Flash импортирует его в виде фрагмента ролика на панель "Библиотека". Также можно перетащить компонента с панели "Компоненты" прямо на панель "Библиотека", а затем добавить его экземпляр в рабочую область. В любом случае сначала компонент необходимо добавить в библиотеку, и только после этого можно использовать элементы его класса.

Если вы добавляете компонент в библиотеку и создаете экземпляр с помощью ActionScript, сначала необходимо импортировать его класс с помощью оператора `import`. В операторе `import` необходимо указать имя пакета и имя класса для компонента. Например, следующий оператор импортирует класс Button:

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

Когда компонент помещается в библиотеку, Flash импортирует папку его активов, которая содержит обложки для разных состояний. *Обложки* компонента представляют собой коллекцию символов, образующих ее графическое представление в приложении. Одна обложка является графическим представлением, или фрагментом ролика, обозначающим определенное состояние компонента.

Содержимое папки "Активы компонента" позволяет при желании изменять обложки компонента. Дополнительные сведения см. в разделе «Настройка компонентов пользовательского интерфейса» на странице 103.

После того как компонент помещен в библиотеку, его экземпляры можно добавить в документ, перетаскивая его значок в рабочую область с панели "Компоненты" или "Библиотека".

Настройка размера компонентов

Чтобы изменить размер экземпляров компонента, можно использовать инструмент "Свободное преобразование" или метод`setSize()`. Метод `setSize()` можно вызвать из любого экземпляра компонента, размер которого нужно изменить (см. описание метода `UIComponent.setSize()`). Следующий код изменяет размер экземпляра компонента List, задавая ширину 200 пикселов и высоту 300 пикселов.

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

Размер компонента не изменяется в соответствии с меткой автоматически. Если экземпляр компонента, добавленного в документ, недостаточно большой для отображения метки, текст метки обрезается. Чтобы метка отображалась полностью, нужно изменить размер компонента.

Дополнительные сведения об настройке размера компонентов см. в их описании в *Справочнике по языку ActionScript 3.0 и компонентам*.

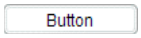
Интерактивный просмотр

Функция интерактивного просмотра включена по умолчанию. Она позволяет просматривать компоненты в рабочей области в том виде, в каком они появятся в опубликованном содержимом Flash. Компоненты отображаются в приблизительном размере.

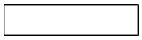
Включение и выключение интерактивного просмотра

- Выберите "Управление" > "Разрешить интерактивный просмотр". Флажок рядом с параметром указывает на то, что он включен.

Интерактивный просмотр отражает разные параметры для каждого компонента. Дополнительные сведения о том, какие параметры компонента отображаются в интерактивном просмотре, см. в описании компонента в *Справочнике по языку ActionScript 3.0 и компонентам*.



Компонент Button при включенном интерактивном просмотре



Компонент Button при выключенном интерактивном просмотре

В интерактивном просмотре компоненты не работают. Чтобы проверить их работу, используйте команду "Управление" > "Тестировать ролик".

Обработка событий

Каждый компонент отправляет события при взаимодействии с пользователем. Например, когда пользователь щелкает компонент Button, он отправляет событие `MouseEvent.CLICK`, а когда пользователь выбирает элемент в списке, компонент List отправляет событие Event.`CHANGE`. Кроме того, событие отправляется, когда происходит значимое для компонента событие. Например, когда завершается загрузка содержимого для экземпляра UILoader, отправляется событие `Event.COMPLETE`. Для обработки события создается код ActionScript, который выполняется при отправке события.

В число событий компонента входят события любого класса, наследуемого компонентом. Это означает, что все компоненты пользовательского интерфейса ActionScript 3.0 наследуют события от класса UIComponent, так как он является их базовым классом ActionScript 3.0. Список событий, передаваемых компонентом, см. в разделе "События" описания класса в *Справочнике по языку ActionScript 3.0 и компонентам*.

Полное описание обработки событий в ActionScript 3.0 см. в руководстве *Программирование на ActionScript 3.0*.

О прослушивателях событий

Следующие ключевые положения относятся к обработке событий для компонентов ActionScript 3.0.

- Все события отправляются экземпляром класса компонента. Экземпляр компонента является *отправителем*.

- *Прослушиватель событий* регистрируется путем вызова метода `addEventListener()` для экземпляра компонента. Например, следующая строка кода добавляет прослушиватель события `MouseEvent.CLICK` для экземпляра Button с именем `aButton`:

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

  Второй параметр метода `addEventListener()` регистрирует имя функции, которая будет вызвана при отправке события (`clickHandler`). Эта функция также называется *функциейобратного вызова*.

- Для одного экземпляра компонента можно зарегистрировать несколько прослушивателей.

  ```
  aButton.addEventListener(MouseEvent.CLICK, clickHandler1);
  aButton.addEventListener(MouseEvent.CLICK, clickHandler2);
  ```

- Можно также зарегистрировать один прослушиватель для нескольких экземпляров компонента.

  ```
  aButton.addEventListener(MouseEvent.CLICK, clickHandler1);
  bButton.addEventListener(MouseEvent.CLICK, clickHandler1);
  ```

- Функция обработчика событий передается объекту события, содержащему информацию о типе события и об отправившем его объекте. Дополнительные сведения см. в разделе «Об объекте события» на странице 26.

- Прослушиватель остается активным, пока приложение не завершит работу или пока он не будет удален методом `removeEventListener()`. Например, следующая строка кода удаляет прослушиватель события `MouseEvent.CLICK` для экземпляра `aButton`:

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

Об объекте события

Объект события является наследником класса объектов Event и имеет свойства, содержащие информацию о событии, включая свойства `target` и `type`, предоставляющие важные сведения о событии.

Свойство	Описание
`type`	Строка, определяющая тип события.
`target`	Ссылка на экземпляр компонента, отправивший событие.

Иногда события имеют дополнительные свойства, которые перечислены в их описаниях в *Справочнике по языку ActionScript 3.0 и компонентам*.

Объект события автоматически создается и передается функции обработчика в момент отправки события.

Объект события можно использовать внутри функции, чтобы получить имя отправленного события или имя экземпляра отправившего его компонента. Через имя экземпляра можно получить другие свойства компонента. Например, следующий код использует свойство `target` объекта события `evtObj`, чтобы получить свойство `label` экземпляра `aButton` и показать его на панели вывода.

```
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");
}
```

Работа со списком отображения

Все компоненты ActionScript 3.0 являются наследниками класса DisplayObject и поэтому имеют доступ к его методам и свойствам для взаимодействия со списком отображения. *Список отображения* представляет собой иерархию отображаемых объектов и графических элементов в приложении. Эта иерархия включает следующие элементы:

- рабочая область, которая является контейнером объектов верхнего уровня;
- экранные объекты, включая фигуры, фрагменты роликов и текстовые поля и др.;
- контейнеры экранных объектов (это особый тип экранных объектов, которые могут содержать дочерние экранные объекты).

Порядок объектов в списке отображения определяет их глубину в родительском контейнере. Глубиной объекта называется его позиция в нисходящем порядке (с переднего к заднему плану рабочей области или контейнера экранного объекта). Порядок глубины можно наглядно проследить, когда объекты пересекаются, но он присутствует, даже если этого не происходит. Каждый объект в списке отображения имеет соответствующую глубину в рабочей области. Чтобы изменить глубину объекта, переместив его вперед или назад по отношению к другим объектам, нужно изменить его позицию в списке отображения. Порядок по умолчанию для объектов в списке отображения соответствует порядку их добавления в рабочую область. Позицию 0 в списке отображения занимает объект, находящийся в начале последовательности глубины.

Добавление компонента в список отображения

Объект можно добавить в объект DisplayObjectContainer, вызвав метод `addChild()` или `addChildAt()` для контейнера. Если контейнером является рабочая область, добавить объект в список отображения можно в ходе разработки: для этого его нужно создать или, если это компонент, перетащить в рабочую область с панели "Компоненты". Чтобы добавить объект в контейнер с помощью ActionScript, сначала создайте его экземпляр, вызвав конструктор с оператором`new`, а после этого вызовите метод `addChild()` или `addChildAt()`, чтобы поместить его в рабочую область и в список отображения. Метод `addChild()` помещает объект в следующую позицию списка отображения, а `addChildAt()` задает позицию, в которую требуется добавить объект. Если указать позицию, которая уже занята другим объектом, этот объект и все находящиеся над ним перемещаются на одну позицию вверх. Свойство `numChildren` объекта DisplayObjectContainer содержит число экранных объектов, содержащихся в контейнере. Чтобы извлечь объект из списка отображения, вызовите метод `getChildAt()` и укажите позицию, а если известно имя объекта, вызовите метод `getChildByName()`.

Примечание. *Добавляя компонент с помощью ActionScript, необходимо задать ему имя с помощью свойства name, чтобы его можно было найти по имени в списке отображения.*

В следующем примере отображаются имена и позиции трех компонентов в списке отображения. Сначала перетащите компоненты NumericStepper, Button и ComboBox в рабочую область так, чтобы они пересекались друг с другом и присвойте их экземплярам имена **aNs**, **aButton** и **aCb**. Затем добавьте следующий код на панель "Действия" в кадре 1 временной шкалы:

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

На панели вывода должны появится следующие строки.

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

Перемещение компонента в списке отображения

Можно изменить позицию объекта в списке отображения и его глубину отображения с помощью метода `addChildAt()`, указав в качестве параметров имя объекта и позицию, в которую его требуется добавить. Например, добавьте следующий код к предыдущему примеру, чтобы первым отображался компонент NumericStepper и повторите цикл, чтобы показать новые позиции компонентов в списке отображения.

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

На панели вывода должны появится следующие строки.

```
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
```

Кроме того, экземпляр NumericStepper должен появится перед остальными компонентами на экране.

Обратите внимание, что свойство `numChildren` представляет количество объектов (от 1 до *n*) в списке отображения, а порядковые номера объектов отсчитываются от 0. Поэтому если в списке отображения содержится три объекта, то третий объект будет занимать в индексе позицию 2. Это означает, что позиция последнего объекта в списке отображения, или верхнего объекта с точки зрения глубины, вычисляется как `numChildren – 1`.

Удаление компонента из списка отображения

Удалить компонент их контейнера экранных объектов и его списка отображения можно с помощью методов `removeChild()` и `removeChildAt()`. В следующем примере в рабочую область добавляется три экземпляра Button, один перед другим, а также по одному прослушивателю событий для каждого из них. При щелчке по каждой кнопке обработчик события удаляет ее из списка отображения и с рабочей области.

1. Создайте новый файл Flash (ActionScript 3.0).
2. Перетащите компонент Button с панели "Компоненты" на панель "Библиотека".
3. Откройте панель "Действия", выберите "Кадр 1" на главной временной шкале и введите следующий код.

```
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);
}
```

Полное описание списка отображения см. в разделе "Программирование отображаемого содержимого" руководства *Программирование на ActionScript 3.0*.

Работа с FocusManager

Когда пользователь нажимает клавишу Tab, для перехода к элементам в приложении Flash или щелкает элементы приложения мышью, класс FocusManager определяет, какой компонент получает фокус для ввода. Не требуется добавлять экземпляр FocusManager в приложение или создавать код для активации FocusManager, если только не нужно создать этот компонент.

Если объект RadioButton получает фокус, FocusManager анализирует этот и все остальные объекты с одинаковым значением `groupName` и переводит фокус на объект, у которого свойство `selected` имеет значение `true`.

Каждый модальный компонент Window содержит экземпляр FocusManager, поэтому элементы управления в этом окне получают собственный порядок табуляции. Это предотвращает непреднамеренный переход пользователя к компонентам в других окнах при нажатии клавиши Tab.

Экземпляр FocusManager использует уровень глубины элементов в контейнере (или порядок по оси *z* в качестве схемы перехода, или *цикла табуляции*, по умолчанию. Пользователь обычно переходит по циклу табуляции с помощью клавиши Tab, причем фокус переходит от первого компонента в фокусе к последнему, а затем снова к первому. Уровни глубины определяются, главным образом, порядком, в котором компоненты перетаскивались в рабочую область, однако для определения окончательного порядка по оси *z* можно использовать команды "Модификация" > "Упорядочить" > "Переместить на передний план"/"Переместить назад". Дополнительные сведения об уровнях глубины см. в разделе «Работа со списком отображения» на странице 27.

Можно вызвать метод `setFocus()`, чтобы переместить фокус на определенный экземпляр компонента в приложении. Следующий пример создает экземпляр FocusManager для текущего контейнера (`this`) и переводит фокус на экземпляр Button с именем `aButton`.

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

Чтобы определить, какой компонент получил фокус, нужно вызвать метод `getFocus()`. А чтобы определить, какой компонент в цикле табуляции получить фокус следующим, нужно вызвать метод `getNextFocusManagerComponent()`. В следующем примере экземпляры CheckBox, RadioButton и Button находятся в рабочей области, каждый из них имеет прослушиватели для событий `MouseEvent.CLICK` и `FocusEvent.MOUSE_FOCUS_CHANGE`. Когда происходит событие `MouseEvent.CLICK` в результате щелчка по компоненту, функция `showFocus()` вызывает метод `getNextFocusManagerComponent()` для определения, какой компонент в цикле табуляции получит фокус следующим. Затем вызывается метод `setFocus()`, чтобы перевести фокус на этот компонент. Когда происходит событие `FocusEvent.MOUSE_FOCUS_CHANGE` функция `fc()` отображает имя компонента, отправившего это событие. Это событие запускается, когда пользователь щелкает компонент, который не является следующим в цикле табуляции.

```
// 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);
}
```

Чтобы создать экземпляр Button, получающий фокус при нажатии клавиши Enter (Windows) или Return (Macintosh), задайте для свойства `FocusManager.defaultButton` экземпляр Button, который должен быть кнопкой по умолчанию, как показано в следующем коде.

```
import fl.managers.FocusManager;

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

Класс FocusManager переопределяет прямоугольник фокуса по умолчанию для Flash Player и рисует пользовательский прямоугольник фокуса со скругленными краями.

Дополнительные сведения о создании схемы фокуса в приложении Flash см. в описании класса FocusManager в *Справочнике по языку ActionScript 3.0 и компонентам*. Чтобы создать пользовательский диспетчер фокуса, необходимо создать класс, реализующий интерфейс *IFocusManager*. Дополнительные сведения см. в разделе IFocusManager в *Справочнике по языку ActionScript 3.0 и компонентам*.

Работа с компонентами на базе класса List

Компоненты List, DataGrid и TileList являются наследниками базового класса SelectableList. Поэтому их считают компонентами, созданными на базе класса List. Компонент ComboBox состоит из текстового поля и списка, поэтому он тоже относится к компонентам на базе класса List.

Компонент List состоит из строк. Компоненты DataGrid и TileList состоят из строк, которые можно разделить на несколько столбцов. Пересечение строки и столбца называется ячейкой. В компоненте List с одним столбцом строк, ячейкой является каждая строка. Ячейка имеет два важных аспекта.

- Данные, содержащиеся в ячейках, называются элементами. *Элемент* — это объект ActionScript, используемый для хранения единиц информации в объекте List. Список можно представить как массив, где каждое проиндексированное пространство является элементом. Элемент списка — это объект, который обычно имеет отображаемое свойство `label` и свойство `data`, которое служит для хранения данных *Поставщик данных* — это модель данных, используемая для элементов списка. Поставщик данных позволяет заполнять компонент на базе класса List путем его назначения свойству `dataProvider` компонента.
- В ячейке могут содержаться разные типы данных от текста до изображений, фрагментов роликов или экземпляров других созданных классов. Поэтому ячейка должна рисоваться или визуализироваться в соответствии с ее содержимым. В следствие этого у компонентов на базе класса List есть *визуализаторы ячеек*. Для компонента DataGrid каждый столбец является объектом DataGridColumn, у которого есть свойство `cellRenderer`, чтобы каждый столбец визуализировался в соответствии с его содержимым.

  Все компоненты на базе класса List имеют свойства `cellRenderer` и `dataProvider`, которые можно настраивать для загрузки и визуализации их ячеек. Дополнительные сведения об использовании этих свойств и работе с компонентами на базе класса List см. в разделах «Работа с объектом DataProvider» на странице 31 и «Работа с объектом CellRenderer» на странице 39.

Работа с объектом DataProvider

Объект DataProvider — это источник данных, с помощью которого можно заполнить данными компоненты ComboBox, DataGrid, List и TileList. Каждый из этих классов компонентов имеет свойство `dataProvider`, которому можно назначить объект DataProvider для заполнения данными ячеек компонента. Как правило, поставщик данных — это коллекция данных, например объект Array или XML.

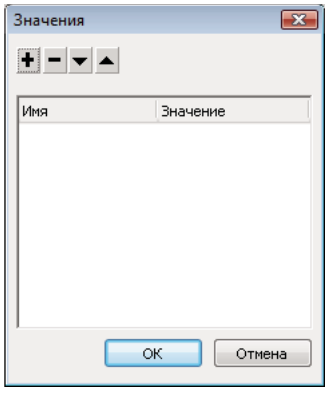
Создание объекта DataProvider

Для компонентов ComboBox, List и TileList можно создать объект DataProvider с помощью параметра `dataProvider` в среде разработки. Компонент DataGrid не имеет параметра dataProvider в Инспекторе свойств, так как он содержит несколько столбцов, в результате чего требуется более сложный поставщик данных. Кроме того, можно использовать ActionScript, чтобы создать DataProvider для этих компонентов, включая DataGrid.

Использование параметра dataProvider

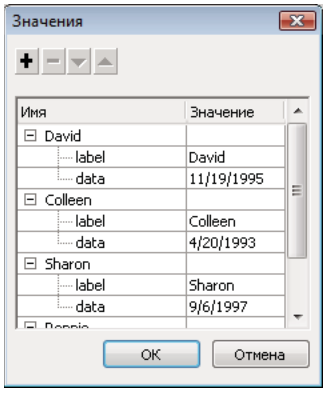
Чтобы создать простого поставщика данных для компонентов ComboBox, List и TileList, щелкните параметр dataProvider на вкладке "Параметры" в Инспекторе свойств или компонентов.

Если дважды щелкнуть ячейку значения, которая сначала показывает пустой массив, откроется диалоговое окно "Значения", в котором можно ввести несколько значений для полей label и data, чтобы создать поставщика данных



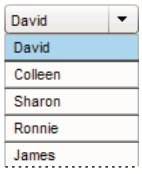
Диалоговое окно "Значение" для dataProvider

Щелкните знак "+", чтобы добавить элемент в свойство dataProvider. Щелкните знак "-", чтобы удалить элемент. Щелкните стрелку вверх, чтобы переместить выделенный элемент вверх по списку, или стрелку вниз, чтобы переместить его вниз. На следующем рисунке показано диалоговое окно "Значения", в котором создается список имен и дней рождения детей.



Диалоговое окно "Значения" с данными

Созданный массив состоит из пар полей меток и значений. Поля меток — это `label` и `data`, а поля значений — это имена и даты рожденья детей. Поле метки определяет содержимое, отображаемое в списке, в данном случае это имя ребенка. В результате компонент ComboBox выглядит так:



Компонент ComboBox, заполненный объектом DataProvider

Завершив ввод данных, нажмите "OK", чтобы закрыть диалоговое окно. Теперь массив в параметре dataProvider заполнен созданными элементами.

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

параметр dataProvider с данными

Чтобы вызвать созданные значения метки и данных, можно получить свойство `dataProvider` компонента с помощью ActionScript.

Создание объекта DataProvider с помощью ActionScript

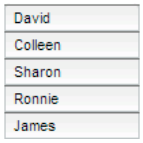
Чтобы создать объект DataProvider, можно создать данные в объекте Array или XML и передать его в качестве параметра `value` конструктору DataProvider.

Примечание. *В ActionScript 3.0 объект Array или XML нельзя назначить напрямую свойству dataProvider, так как свойство определяется в качестве объекта DataProvider и может получать только объекты типа DataProvider.*

В следующем примере заполняется данными компонент List, который представляет собой один столбец строк с именами и датами рождения нескольких детей. В этом примере определяется список в массиве `items`, который затем передается в качестве параметра при создании экземпляра DataProvider (`new DataProvider(items)`) и назначается свойству `dataProvider` компонента List.

```
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);
```

Массив состоит из пар полей "метка" и "данные". Поля меток — это `label` и `data`, а поля значений — это имена и даты рожденья детей. Поле метки определяет содержимое, отображаемое в списке, в данном случае это имя ребенка. В результате компонент List выглядит так:

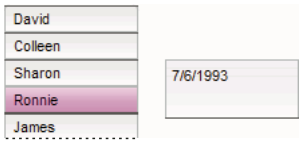


Компонент List, заполненный объектом DataProvider

Значение поля данных отображается, когда пользователь выбирает элемент списка, щелкнув по нему, в результате чего отравляется событие `change`. В следующем примере к предыдущему коду добавляется компонент TextArea (`aTa`) и обработчик события (`changeHandler`), чтобы при выборе имени в списке отображался день рожденья ребенка.

```
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;
};
```

Теперь, когда пользователь выбирает имя ребенка в компоненте List, его день рожденья отображается в компоненте TextArea, как показано на следующем рисунке. Это выполняется с помощью функции `changeHandler()`, которая задает свойству `text` компонента TextArea (`aTa.text`) значение поля данных в выбранном элементе (`event.target.selectedItem.data`). Свойство `event.target` — это объект, отправивший событие, которым в данном случае является компонент List.



Отображение поля данных из объекта DataProvider компонента List

Объект DataProvider может содержать не только текст. В следующем примере в объект DataProvider, заполняющий данными компонент TileList, включены экземпляры MovieClip. Объект DataProvider формируется вызовом метода`addItem()` для добавления каждого созданного ранее элемента MovieClip, цветного окна.

```
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();
```

Для заполнения объекта DataProvider также можно использовать не массив, а данные XML. Например, следующий код сохраняет данные в объекте XML с именем `employeesXML`, а затем передает его в качестве значения конструктору `DataProvider()`.

```
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;
```

Данные можно передавать в виде атрибутов объекта XML, как в предыдущем коде, либо в качестве его свойств, как в следующем коде.

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

Объект DataProvider также имеет набор методов и свойств, с помощью которых им можно пользоваться и манипулировать. Для добавления, удаления, замены, сортировки или объединения элементов в объекте DataProvider можно использовать его API-интерфейс.

Манипулирование объектом DataProvider

Добавлять элементы в объект DataProvider можно с помощью `addItem()` и `addItemAt()`. В следующем примере добавляются элементы, введенные пользователем в текстовое поле редактируемого компонента ComboBox. Для этого необходимо перетащить компонент ComboBox в рабочую область и присвоить ему имя экземпляра `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);
}
```

Кроме того, с помощью объекта DataProvider можно удалять или заменять элементы в компоненте. В следующем примере создается два отдельных компонента List, `listA` и `listB`, а затем добавляется компонент Button с меткой "Синхронизация". Когда пользователь нажимает кнопку (объект Button), вызывается метод `replaceItemAt()` для замены элементов списка `listB`элементами списка `listA`. Если список `listA` длиннее списка`listB`, вызывается метод `addItem()` для добавления дополнительных элементов в `listB`. Если список `listB` длиннее списка`listA`, вызывается метод `removeItemAt()` для удаления лишних элементов из `listB`.

```
// 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++);
        }
    }
}
```

Объект DataProvider можно также объединять с другим и сортировать с помощью методов `merge()`, `sort()` и `sortOn()`. В следующем примере два экземпляра DataGrid (`aDg` и `bDg`) заполняются данными из частичных списков игроков двух футбольных команд. Также добавляется компонент Button с меткой "Объединить", и когда пользователь нажимает на кнопку, обработчик события (`mrgHandler`) объединяет список объекта `bDg` со списком объекта `aDg` и сортирует полученный экземпляр DataGrid по столбцу "Имя".

```
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");
}
```

Дополнительные сведения см. в описании класса DataProvider в Справочнике по языку ActionScript 3.0 и компонентам.

Работа с объектом CellRenderer

CellRenderer — это класс, используемый компонентами на базе класса List (List, DataGrid, TileList и ComboBox) для отображения пользовательского содержимого ячеек в их строках и выполнения манипуляций с ним. Пользовательская ячейка может содержать текст, предварительно созданный компонент, например CheckBox, или любой созданный класс экранных объектов. Чтобы визуализировать данные с помощью CellRenderer, можно либо расширить класс CellRenderer или внедрить интерфейс ICellRenderer, чтобы создать собственный класс CellRenderer.

Классы List, DataGrid, TileList и ComboBox являются подклассами класса SelectableList. Класс SelectableList включает стиль `cellRenderer`. Этот стиль определяет экранный объект, используемый компонентом для визуализации ячеек.

Настроить форматирование используемых стилей можно с помощью CellRenderer, вызвав метод `setRendererStyle()` объекта List (см. раздел «Форматирование ячеек» на странице 39). Также можно определить пользовательский класс для использования в качестве CellRenderer (см. раздел «Определение пользовательского класса CellRenderer» на странице 40).

Форматирование ячеек

Класс CellRenderer включает ряд стилей, позволяющих контролировать формат ячейки.

Следующие стили позволяют определить обложки, используемые для различных состояний ячейки (отключенное, нажатое, наведенное и ненажатое):

- `disabledSkin` и `selectedDisabledSkin`;
- `downSkin` и `selectedDownSkin`;
- `overSkin` и `selectedOverSkin`;

- `upSkin` и `selectedUpSkin`.

  Следующие стили применяются к форматированию текста:

- `disabledTextFormat`;
- `textFormat`;
- `textPadding`.

Чтобы задать эти стили, нужно вызвать метод `setRendererStyle()` объекта List или метод `setStyle()` объекта CellRenderer. Чтобы получить эти стили, нужно вызвать метод `getRendererStyle()` объекта List или метод `getStyle()` объекта CellRenderer. Также можно вызвать объект, определяющий все стили визуализатора (в качестве названных свойств объекта), через свойство `rendererStyles` объекта List или с помощью метода `getStyleDefinition()` объекта CellRenderer.

Вызвав метод `clearRendererStyle()`, можно сбросить стиль и использовать значение по умолчанию.

Чтобы получить или задать высоту строк списка, используйте свойство `rowHeight` объекта List.

Определение пользовательского класса CellRenderer

Создание класса, расширяющего класс CellRenderer, для определения пользовательского CellRenderer

Например, следующий код включает два класса. Класс ListSample создает экземпляр компонента List и использует другой класс, CustomRenderer, чтобы определить визуализатор ячейки для компонента List. Класс CustomRenderer расширяет класс CellRenderer.

1. Выберите "Файл" > "Создать".
2. В открывшемся окне "Создать документ" выберите "Файл Flash (ActionScript 3.0)" и нажмите "ОК".
3. Выберите "Окно" > "Компоненты", чтобы открыть панель "Компоненты".
4. На панели "Компоненты" перетащите компонент "Список" в рабочую область.
5. Если Flash не показывает Инспектора свойств, выберите "Окно" > "Свойства" > "Свойства".
6. Выделив компонент List, задайте свойства в Инспекторе свойств.
   - Имя экземпляра: myList
   - W (ширина): 200
   - H (высота): 300
   - X: 20
   - Y: 20
7. Выберите кадр 1 слоя 1 во временной шкале, а затем "Окно" > "Действия".
8. Введите следующий сценарий на панели "Действия".

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

9. Выберите пункт "Файл" > "Сохранить". Присвойте файлу имя и нажмите кнопку "ОК".
10. Выберите "Файл" > "Создать".
11. В открывшемся окне "Создать документ" выберите "Файл ActionScript" и нажмите "ОК".

12 В окне сценария введите следующий код, чтобы определить класс CustomCellRenderer.

```
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 Выберите пункт "Файл" > "Сохранить". Присвойте файлу имя CustomCellRenderer.as, поместите его в тот же каталог, что и FLA-файл, и нажмите кнопку "ОК".

14 Выберите "Управление" > "Тестировать ролик".

Использование класса, реализующего интерфейс ICellRenderer, для определения пользовательского CellRenderer

Также CellRenderer можно определить с помощью любого класса, наследующего класс DisplayObject и реализующего интерфейс ICellRenderer. Например, следующий код определяет два класса. Класс ListSample2 добавляет объект List в список отображения и определяет его CellRenderer для использования класса CustomRenderer. Класс CustomRenderer расширяет класс CheckBox (который расширяет класс DisplayObject) и реализует интерфейс ICellRenderer. Обратите внимание, что класс CustomRenderer определяет методы get и set для свойств `data` и `listData`, определенных в интерфейсе ICellRenderer. Другие свойства и методы, определенные в интерфейсе ICellRenderer (свойство `selected` и метод `setSize()` уже определены в классе CheckBox.

1 Выберите "Файл" > "Создать".

2 В открывшемся окне "Создать документ" выберите "Файл Flash (ActionScript 3.0)" и нажмите "ОК".

3 Выберите "Окно" > "Компоненты", чтобы открыть панель "Компоненты".

4 На панели "Компоненты" перетащите компонент "Список" в рабочую область.

5 Если Flash не показывает Инспектора свойств, выберите "Окно" > "Свойства" > "Свойства".

6 Выделив компонент List, задайте свойства в Инспекторе свойств.

- Имя экземпляра: myList
- W (ширина): 100
- H (высота): 300
- X: 20
- Y: 20

7 Выберите кадр 1 слоя 1 во временной шкале, затем откройте "Окно" > "Действия".

8 Введите следующий сценарий на панели "Действия".

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

9 Выберите пункт "Файл" > "Сохранить". Присвойте файлу имя и нажмите кнопку "ОК".

10 Выберите "Файл" > "Создать".

11 В открывшемся окне "Создать документ" выберите "Файл ActionScript" и нажмите "ОК".

12 В окне сценария введите следующий код, чтобы определить класс CustomCellRenderer.

```
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 Выберите пункт "Файл" > "Сохранить". Присвойте файлу имя CustomCellRenderer.as, поместите его в тот же каталог, что и FLA-файл, и нажмите кнопку "ОК".

14 Выберите "Управление" > "Тестировать ролик".

Использование символа для определения CellRenderer

Также для определения CellRenderer можно использовать символ в библиотеке. Символ должен быть экспортирован для ActionScript, а имя класса для библиотечного символа должно быть связано с фалом класса, который реализует интерфейс ICellRenderer или расширяет класс CellRenderer или один из его подклассов).

В следующем примере пользовательский CellRenderer определяется с помощью библиотечного символа.

1 Выберите "Файл" > "Создать".

2 В открывшемся окне "Создать документ" выберите "Файл Flash (ActionScript 3.0)" и нажмите "ОК".

3 Выберите "Окно" > "Компоненты", чтобы открыть панель "Компоненты".

4 На панели "Компоненты" перетащите компонент "Список" в рабочую область.

5 Если Flash не показывает Инспектора свойств, выберите "Окно" > "Свойства" > "Свойства".

6 Выделив компонент List, задайте свойства в Инспекторе свойств.

- Имя экземпляра: myList
- W (ширина): 100
- H (высота): 400

- X: 20
- Y: 20

7 Щелкните панель "Параметры", а затем дважды щелкните второй столбец в строке dataProvider.

8 В открывшемся диалоговом окне "Значения" дважды щелкните знак "+", чтобы добавить две строки элементов данных (с метками label0 и label1), затем нажмите кнопку "OK".

9 С помощью инструмента "Текст" нарисуйте в рабочей области текстовое поле.

10 Выделив текстовое поле, задайте свойства в Инспекторе свойств.

- Тип текста: Динамический текст
- Имя экземпляра: textField
- W (ширина): 100
- Размер шрифта: 24
- X: 0
- Y: 0

11 Выделив текстовое поле, выберите "Модификация" > "Преобразовать в символ".

12 В диалоговом окне "Преобразовать в символ" настройте необходимые параметры и нажмите "OK".

- Имя: MyCellRenderer
- Тип: MovieClip
- Экспорт для ActionScript: выбрано
- Экспортировать в первый кадр: выбрано
- Класс: MyCellRenderer
- Базовый класс: flash.display.SimpleButton

  Если Flash отображает предупреждение класса ActionScript, нажмите кнопку "OK" в окне предупреждения.

13 Удалите экземпляр символа нового фрагмента ролика из рабочей области.

14 Выберите кадр 1 слоя 1 во временной шкале, затем откройте "Окно" > "Действия".

15 Введите следующий сценарий на панели "Действия".

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

16 Выберите пункт "Файл" > "Сохранить". Присвойте файлу имя и нажмите кнопку "OK".

17 Выберите "Файл" > "Создать".

18 В открывшемся окне "Создать документ" выберите "Файл ActionScript" и нажмите "OK".

19 В окне сценария введите следующий код, чтобы определить класс MyCellRenderer.

```
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 Выберите пункт "Файл" > "Сохранить". Присвойте файлу имя MyCellRenderer.as, поместите его в тот же каталог, что и FLA-файл, и нажмите кнопку "ОК".

21 Выберите "Управление" > "Тестировать ролик".

Свойства CellRenderer

Свойство `data` представляет собой объект, содержащий все свойства, заданные для CellRenderer. Например, в следующем классе, который определяет пользовательский класс CellRenderer, расширяющий класс CheckBox, функция set для свойства `data` передает значение `data.label` свойству `label`, унаследованному от класса CheckBox.

```
    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;
        }
    }
}
```

Свойство `selected` определяет, выделена ячейка в списке или нет.

Применение объекта CellRenderer к столбцу объекта DataGrid

Объект DataGrid может иметь несколько столбцов, для каждого из которых можно указать разные визуализаторы. Каждый столбец объекта DataGrid представлен объектом DataGridColumn, а класс DataGridColumn включает свойство `cellRenderer`, для которого можно определить объект CellRenderer столбца.

Определение объекта CellRenderer для редактируемой ячейки

Класс DataGridCellEditor определяет визуализатор, используемый для редактируемых ячеек в объекте DataGrid. Он становится визуализатором ячейки, если свойство`editable` объекта DataGrid имеет значение `true` и пользователь щелкает ячейку для редактирования.. Чтобы определить CellRenderer для редактируемой ячейки, задайте свойство `itemEditor` для каждого элемента в массиве `columns` объекта DataGrid.

Использование изображения, SWF-файла или фрагмента ролика в качестве объекта CellRenderer

Класс ImageCell, являющийся подклассом CellRenderer, определяет объект, используемый для визуализации ячеек, в которых основным содержимым является изображение, SWF-файл или фрагмент ролика. Класс ImageCell включает следующие стили для определения внешнего вида ячейки:

- `imagePadding` — заполнитель, отделяющий край ячейки от края изображения, в пикселах;
- `selectedSkin` — обложка, используемая для указания выбранного состояния;
- `textOverlayAlpha` — прозрачность слоя, расположенного за меткой ячейки;

- `textPadding` — заполнитель, отделяющий край ячейки от края текста, в пикселах.

  Класс ImageCell является классом CellRenderer по умолчанию для класса TileList.

Обеспечение расширенного доступа к компонентам

Визуальное содержимое в приложениях Flash можно сделать доступным для пользователей с ослабленным зрением с помощью программы чтения с экрана, которая воспроизводит аудиоописание содержимого экрана. Дополнительные сведения о том, как сделать приложение Flash доступным для программы чтения с экрана, см. в главе 18 "Создание содержимого с расширенной доступностью" в руководстве *Использование Flash*.

Чтобы сделать компонент ActionScript 3.0 доступным для программы чтения с экрана, необходимо также импортировать класс Accessibility и вызвать его метод `enableAccessibility()`. Для программы чтения с экрана можно сделать доступными следующие компоненты ActionScript 3.0.

Компонент	**Класс Accessibility**
Button	`ButtonAccImpl`
CheckBox	`CheckBoxAccImpl`
ComboBox	`ComboBoxAccImpl`
List	`ListAccImpl`
RadioButton	`RadioButtonAccImpl`
TileList	`TileListAccImpl`

Классы расширенного доступа компонентов находятся в пакете `fl.accessibility`. Чтобы сделать класс CheckBox доступным для программ чтения с экрана, необходимо добавить в приложение следующие операторы:

```
import fl.accessibility.CheckBoxAccImpl;

CheckBoxAccImpl.enableAccessibility();
```

Расширенный доступ для компонента включается только один раз, независимо от числа созданных экземпляров.

Примечание. *Включение расширенного доступа незначительно увеличивает размер файла, так как во время компиляции в него включаются необходимые для этого классы.*

Большинство компонентов поддерживают навигацию с помощью клавиатуры. Дополнительные сведения о включении расширенного доступа для компонентов и навигации с помощью клавиатуры см. в разделах, посвященных взаимодействию пользователей в главе «Использование компонентов пользовательского интерфейса» на странице 47, а также описания классов расширенного доступа *Справочнике по языку ActionScript 3.0 и компонентам*.

Глава 4. Использование компонентов пользовательского интерфейса

В этой главе объясняется, как использовать компоненты пользовательского интерфейса ActionScript 3.0, включенные в Flash.

Использование компонента Button

Компонент Button — это прямоугольная кнопка с изменяемым размером, которую пользователь может нажать с помощью мыши или клавиши пробела, чтобы инициировать выполнение действия в приложении. К экземпляру Button можно добавить пользовательский значок. Также можно изменить поведение компонента Button с нажатия на переключение. После щелчка кнопка-переключатель остается в нажатом положении и возвращается в исходное положение при повторном щелчке.

Компонент Button является основным элементом многих форм и веб-приложений. Кнопки можно использовать везде, где пользователь должен инициировать событие. Например, в большинстве форм используется кнопка "Отправить". А в презентацию можно добавить кнопки "Предыдущий" и "Следующий".

Взаимодействие пользователей с компонентом Button

Кнопку в приложении можно включить или отключить. В отключенном состоянии кнопка не реагирует на мышь или клавиатуру. Включенная кнопка получает фокус при щелчке или переходе с помощью клавиши Tab. Когда экземпляр Button получает фокус, им можно управлять с помощью следующих клавиш.

Клавиша	**Описание**
Shift+Tab	Переводит фокус на предыдущий объект.
Пробел	Нажимает или отпускает кнопку и запускает событие `click`.
Tab	Переводит фокус на следующий объект.
Enter/Return	Переводит фокус на следующий объект, если кнопка задана в качестве кнопки по умолчанию для FocusManager.

Дополнительные сведения об управлении фокусом см. в разделах, посвященных интерфейсу IFocusManager и классу FocusManager, в руководствах *Справочник по языку ActionScript 3.0 и компонентам* и «Работа с FocusManager» на странице 29.

Интерактивный просмотр каждого элемента Button отражает изменения, которые были внесены в параметры в Инспекторе свойств или Инспекторе компонентов во время разработки.

Примечание. *Если значок больше кнопки, то он выходит за ее пределы.*

Чтобы назначить кнопку в качестве нажимной кнопки по умолчанию в приложении (то есть, кнопки, получающей событие click, когда пользователь нажимает клавишу Enter), задайте свойство `FocusManager.defaultButton`. Например, в следующем коде в качестве кнопки по умолчанию задается экземпляр Button с именем `submitButton`.

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

Когда в приложение добавляется компонент Button, его можно сделать доступным для программ чтения с экрана путем добавления следующих строк кода ActionScript.

```
import fl.accessibility.ButtonAccImpl;

ButtonAccImpl.enableAccessibility();
```

Расширенный доступ для компонента включается только один раз, независимо от числа созданных экземпляров.

Параметры компонента Button

Следующие параметры разработки можно задать в Инспекторе свойств ("Окно" > "Свойства" > "Свойства") или в Инспекторе компонентов ("Окно" > "Инспектор компонентов") для каждого экземпляра Button : `emphasized`, `label`, `labelPlacement`, `selected` и `toggle`. Каждый из этих параметров имеет соответствующее свойство ActionScript с тем же именем. Когда этим параметрам присваивается значение, задается исходное состояние свойства в приложении. Определение свойства в ActionScript переопределяет значение, заданное параметру. Сведения о возможных значениях для этих параметров см. в описании класса Button в *Справочнике по языку ActionScript 3.0 и компонентам*.

Создание приложения с компонентом Button

Ниже описывается процедура добавления компонента Button в приложение в ходе разработки. В данном примере компонент Button изменяет состояние компонента ColorPicker при щелчке.

1. Создайте новый документ Flash (ActionScript 3.0).
2. Перетащите компонент Button с панели "Компоненты" в рабочую область и введите следующие значения в Инспекторе свойств.
   - Введите имя экземпляра **aButton**.
   - Введите значение **Show** (Показать) для параметра label.
3. Добавьте компонент ColorPicker в рабочую область и присвойте ему имя экземпляра **aCp**.
4. Откройте панель "Действия", выберите "Кадр 1" на главной временной шкале и введите следующий код ActionScript.

```
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;
    }
}
```

Во второй линии кода функция `clickHandler()` регистрируется в качестве обработчика для события MouseEvent.CLICK . Событие происходит, когда пользователь щелкает компонент Button, в результате чего функция `clickHandler()` выполняет одно из следующих действий в зависимости от значения Button:

- "Show" (Показать) делает экземпляр ColorPicker видимым и изменяет метку компонента Button на "Disable" (Выключить).
- "Disable" (Выключить) отключает экземпляр ColorPicker и меняет метку Button на "Enable" (Включить).
- "Enable" (Включить) включает экземпляр ColorPicker и меняет метку Button на "Hide" (Скрыть).
- "Hide" (Скрыть) делает экземпляр ColorPicker невидимым и меняет метку Button на "Show" (Показать).

5 Выберите "Управление" > "Тестировать ролик", чтобы запустить приложение.

Создание приложения с компонентом Button

Ниже приводится процесс создания кнопки-переключателя Button с использованием ActionScript, а также описывается тип события на панели "Вывод" при щелчке экземпляра Button. В этом примере создается экземпляр Button путем вызова конструктора класса и добавляется в рабочую область путем вызова метода addChild() .

1 Создайте новый документ Flash (ActionScript 3.0).

2 Перетащите компонент Button с панели "Компоненты" на панель "Библиотека" текущего документа.

В результате этого компонент добавляется в библиотеку, но остается невидимым в приложении.

3 Откройте панель "Действия", выберите "Кадр 1" в главной временной шкале и введите следующий код, чтобы создать экземпляр Button.

```
import fl.controls.Button;

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

Метод `move()` размещает кнопку в точке рабочей области с координатами 50 (x), 50 (y).

4 Теперь добавьте следующий код ActionScript, чтобы создать прослушиватель событий и функцию обработчика событий.

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

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

5 Выберите "Управление" > "Тестировать ролик".

При щелчке кнопки Flash показывает сообщение"Тип события: click" на панели "Вывод".

Использование компонента CheckBox

Компонент CheckBox — это квадратное окно, в котором можно поставить или снять флажок. При щелчке в окне появляется флажок. К компоненту CheckBox можно добавить текстовую метку и поместить ее слева, справа, над или под окном.

Компоненты CheckBox можно использовать для получения набора значений `true` или `false`, не исключающих друг друга. Например, приложение, собирающее информацию о типе автомобиля, который вы хотите купить, может использовать экземпляры CheckBox, чтобы предоставить вам возможность выбрать характеристики.

Взаимодействие пользователей с компонентом CheckBox

Компонент CheckBox в приложении можно включить или отключить. Если компонент CheckBox включен и пользователь щелкает его или его метку, CheckBox получает фокус ввода и отображает нажатый вид. Если пользователь перемещает курсор за пределы области CheckBox или его метки, удерживая нажатой кнопку мыши, вид компонента возвращается к исходному, а фокус ввода сохраняется. Состояние CheckBox не изменяется, пока кнопка мыши не будет отпущена над компонентом. Кроме того, компонент CheckBox имеет два отключенных состояния (выбранное и невыбранное) которые используют `selectedDisabledSkin` и `disabledSkin` соответственно и не допускают взаимодействия с помощью мыши и клавиатуры.

Если компонент CheckBox отключен, он отображает отключенные вид независимо от взаимодействия пользователя. В отключенном состоянии CheckBox не реагирует на мышь или клавиатуру.

Экземпляр CheckBox получает фокус, если пользователь щелкнет по нему или перейдет с помощью клавиши Tab. Когда экземпляр CheckBox получает фокус, им можно управлять с помощью следующих клавиш.

Клавиша	Описание
Shift+Tab	Переводит фокус на предыдущий элемент.
Пробел	Устанавливает или снимает флажок компонента и инициирует событие `change`.
Tab	Переводит фокус на следующий элемент.

Дополнительные сведения об управлении фокусом см. в документе «Работа с FocusManager» на странице 29 и в описании класса FocusManager в *Справочнике по языку ActionScript 3.0 и компонентам*.

Интерактивный просмотр каждого элемента CheckBox отражает изменения, которые были внесены в параметры в Инспекторе свойств или Инспекторе компонентов во время разработки.

Когда в приложение добавляется компонент CheckBox, его можно сделать доступным для программ чтения с экрана путем добавления следующих строк кода ActionScript.

```
import fl.accessibility.CheckBoxAccImpl;

CheckBoxAccImpl.enableAccessibility();
```

Расширенный доступ для компонента включается только один раз, независимо от числа имеющихся экземпляров.

Параметры компонента CheckBox

В Инспекторе свойств или Инспекторе компонентов можно задать следующие параметры разработки для каждого экземпляра компонента CheckBox : `label`, `labelPlacement` и `selected`. Каждый из этих параметров имеет соответствующее свойство ActionScript с тем же именем. Сведения о возможных значениях для этих параметров см. в описании класса CheckBox в *Справочнике по языку ActionScript 3.0 и компонентам*.

Создание приложения с использованием CheckBox

Ниже описывается процедура добавления компонента CheckBox в приложение в процессе разработки с использованием фрагмента формы заявки на предоставление ссуды. В форме задается вопрос, является ли податель заявки владельцем дома, а рядом с ним добавлен компонент CheckBox, чтобы пользователь мог ответить "да". На случай положительного ответа в форме предусмотрены два переключателя, чтобы пользователь мог указать относительную стоимость домовладения.

Создание приложения с использованием компонента CheckBox

1. Создайте новый документ Flash (ActionScript 3.0).
2. Перетащите компонент CheckBox с панели "Компоненты" в рабочую область.
3. В Инспекторе свойств выполните следующие действия.
   - Введите **homeCh** в качестве имени экземпляра.
   - Введите **140** для значения ширины (W).
   - Введите вопрос "**Владеете домом?**" в качестве значения параметра label.
4. Перетащите два компонента RadioButton с панели "Компоненты" в рабочую область и поместите из справа от экземпляра CheckBox. или под ним. Введите для них следующие значения в Инспекторе свойств.
   - Введите имена экземпляров **underRb** и **overRb**.
   - Введите **120** для значения ширины (W) обоих компонентов RadioButton.

- Введите значение **До 500000 долларов США?** для параметра label экземпляра `underRb`.
- Введите значение **Свыше 500000 долларов США?** для параметра label экземпляра `overRb`.
- Введите значение **valueGrp** для параметра groupName обоих экземпляров RadioButton.

5 Откройте панель "Действия", выберите "Кадр 1" на главной временной шкале и введите следующий код ActionScript.

```
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;
}
```

Этот код создает обработчик события `CLICK`, который включает экземпляры RadioButton с именами `underRb` и `overRb`, если выбран экземпляр CheckBox с именем `homeCh`, и отключает их, если `homeCh` не выбран. Дополнительную информацию см. в описании класса MouseEvent в *Справочнике по языку ActionScript 3.0 и компонентам*.

6 Выберите "Управление" > "Тестировать ролик".

В следующем примере создается копия предыдущего приложения, но экземпляры CheckBox и RadioButton создаются с помощью ActionScript.

Создание компонента CheckBox с помощью ActionScript

1 Создайте новый документ Flash (ActionScript 3.0).

2 Перетащите компоненты CheckBox и RadioButton с панели "Компоненты" на панель "Библиотека" текущего документа. Если панель "Библиотека" еще не открыта, нажмите клавиши Ctrl+L или выберите "Окно" > "Библиотека".

В результате этого компоненты станут доступными для приложения, но не появятся в рабочей области.

3 Откройте панель "Действия", выберите Кадр 1 на основной временной шкале и введите следующий код, чтобы создать и разместить экземпляры компонентов:

```
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?";
```

В данном коде используются конструкторы `CheckBox()` и `RadioButton()` для создания компонентов и метод `addChild()` для их добавления в рабочую область. Метод `move()` используется для размещения компонентов в рабочей области.

4 Теперь добавьте следующий код ActionScript, чтобы создать прослушиватель событий и функцию обработчика событий.

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

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

Этот код создает обработчик события `CLICK`, который включает переключатели с именами `underRb` и `overRb`, если выбран экземпляр CheckBox с именем `homeCh`, и отключает их, если `homeCh` не выбран. Дополнительную информацию см. в описании класса MouseEvent в *Справочнике по языку ActionScript 3.0 и компонентам*.

5 Выберите "Управление" > "Тестировать ролик".

Использование компонента ColorPicker

Компонент ColorPicker дает пользователю возможность выбрать цвет из таблицы образцов. По умолчанию экземпляр ColorPicker показывает один цвет в квадратной кнопке. Когда пользователь нажимает кнопку, появляется список доступных цветов в виде панели образцов, а также шестнадцатеричное значение текущего выбранного цвета.

Чтобы настроить цвета, отображаемые в компоненте ColorPicker, необходимо задать свойство `colors`, указав нужные значения цветов.

Взаимодействие пользователей с компонентом ColorPicker

Компонент ColorPicker дает пользователю возможность выбрать цвет и применить его к другому объекту приложения. Например, чтобы разрешить пользователю настраивать элементы приложения, такие как цвет фона или текста, можно добавить компонент ColorPicker и применить цвет, выбранный пользователем.

Пользователь выбирает цвет, щелкая его образец на панели или указывая шестнадцатеричное значение в текстовом поле. После того как пользователь выберет цвет, можно использовать свойство `selectedColor` экземпляра ColorPicker, чтобы применить цвет к тексту или другому объекту в приложении.

Экземпляр ColorPicker получает фокус, если пользователь наведет на него курсор или перейдет к нему с помощью клавиши Tab. Когда открыта панель образцов экземпляра ColorPicker, ею можно управлять с помощью следующих клавиш.

Клавиша	**Описание**
Home	Выделяет первый цвет на панели образцов.
"Стрелка вверх"	Перемещает выделение на одну строку вверх на панели образцов.
"Стрелка вниз"	Перемещает выделение на одну строку вниз на панели образцов.
"Стрелка вправо"	Перемещает выделение на один цвет вправо на панели образцов.
"Стрелка влево"	Перемещает выделение на один цвет влево на панели образцов.
End	Выделяет последний цвет на панели образцов.

Параметры компонента ColorPicker

В Инспекторе свойств или Инспекторе компонентов можно задать следующие параметры разработки для каждого экземпляра компонента ColorPicker: `selectedColor` и `showTextField`. Каждый из этих параметров имеет соответствующее свойство ActionScript с тем же именем. Сведения о возможных значениях для этих параметров см. в описании класса ColorPicker в *Справочнике по языку ActionScript 3.0 и компонентам*.

Создание приложения с компонентом ColorPicker

В следующем примере компонент ColorPicker добавляется в процессе разработки приложения. В данном примере при каждом изменении цвета в экземпляре ColorPicker функция `changeHandler()` вызывает функцию `drawBox()`, чтобы нарисовать новое окно с цветом, выбранным в ColorPicker.

1. Создайте новый документ Flash (ActionScript 3.0).
2. Перетащите ColorPicker с панели "Компоненты" в центр рабочей области и присвойте экземпляру имя **aCp**.
3. Откройте панель "Действия", выберите "Кадр 1" на главной временной шкале и введите следующий код ActionScript.

```
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 Выберите "Управление" > "Тестировать ролик".

5 Щелкните ColorPicker и выберите цвет, в который будет окрашено окно.

Создание компонента ColorPicker с помощью ActionScript

В этом примере используется конструктор`ColorPicker()` и метод `addChild()` для создания экземпляра ColorPicker в рабочей области. Свойству `colors` задаются следующие значения цвета: красный (0xFF0000), зеленый (0x00FF00) и синий (0x0000FF), чтобы указать цвета, отображаемые в ColorPicker. Также создается экземпляр TextArea, и каждый раз при выборе другого цвета в ColorPicker соответствующим образом меняется цвет текста в TextArea.

1 Создайте новый документ Flash (ActionScript 3.0).

2 Перетащите компонент ColorPicker с панели "Компоненты" на панель "Библиотека".

3 Перетащите компонент TextArea с панели "Компоненты" на панель "Библиотека".

4 Откройте панель "Действия", выберите "Кадр 1" на главной временной шкале и введите следующий код ActionScript.

```
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 Выберите "Управление" > "Тестировать ролик".

Использование компонента ComboBox

Компонент ComboBox дает пользователю возможность выбрать один элемент из раскрывающегося списка. Компонент ComboBox может быть статическим или редактируемым. Редактируемый компонент ComboBox позволяет вводить текст непосредственно в текстовое поле в начале списка. Если список, открываясь, достигает конца документа, то он открывается не вниз, а вверх. Компонент ComboBox состоит из трех подкомпонентов: BaseButton, TextInput и List .

В редактируемом компоненте ComboBox областью щелчка является только кнопка, а текстовое поле — нет. В статическом ComboBox область щелчка состоит из кнопки и текстового поля. Эта область реагирует на щелчок, открывая или закрывая раскрывающийся список.

Когда пользователь выбирает элемент в списке с помощью мыши или клавиатуры, метка выделения копируется в текстовое поле в начале ComboBox.

Взаимодействие пользователей с компонентом ComboBox

Компонент ComboBox можно использовать в любых формах или приложениях, где требуется выбрать один элемент из списка. Например, можно добавить раскрывающийся список штатов в форму адреса заказчика. В более сложных сценариях можно использовать редактируемый компонент ComboBox. Например, в приложении, предоставляющем маршрут проезда, можно использовать редактируемый компонент ComboBox для ввода отправного пункта и места назначения. Тогда раскрывающийся список будет содержать адреса, введенные пользователем ранее.

Если компонент ComboBox редактируемый, то есть, свойство `editable` имеет значение `true`, следующие клавиши убирают фокус с поля ввода текста и оставляют предыдущее значение. Исключение составляет клавиша Enter , которая сначала применяет новое значение, если пользователь ввел текст.

Клавиша	**Описание**
Shift+Tab	Переводит фокус на предыдущий элемент. Если выбран новый элемент, отправляется событие `change`.
Tab	Переводит фокус на следующий элемент. Если выбран новый элемент, отправляется событие `change`.
"Стрелка вниз"	Перемещает выделение на один элемент вниз.
End	Перемещает выделение в конец списка.
Escape	Закрывает раскрывающийся список и возвращает фокус в ComboBox.
Enter	Закрывает раскрывающийся список и возвращает фокус в ComboBox. Если компонент ComboBox редактируемый и пользователь вводит текст, клавиша Enter применяет ввод пользователя в качестве нового значения.
Home	Перемещает выделение в начало списка.
Page Up	Перемещает выделение на одну страницу вверх.
Page Down	Перемещает выделение на одну страницу вниз.

Когда в приложение добавляется компонент ComboBox, его можно сделать доступным для программ чтения с экрана путем добавления следующих строк кода ActionScript.

```
import fl.accessibility.ComboBoxAccImpl;

ComboBoxAccImpl.enableAccessibility();
```

Расширенный доступ для компонента включается только один раз, независимо от числа имеющихся экземпляров.

Параметры компонента ComboBox

В Инспекторе свойств и Инспекторе компонентов можно задать следующие параметры для каждого экземпляра ComboBox: `dataProvider`, `editable`, `prompt` и `rowCount`. Каждый из этих параметров имеет соответствующее свойство ActionScript с тем же именем. Сведения о возможных значениях для этих параметров см. в описании класса ComboBox в *Справочнике по языку ActionScript 3.0 и компонентам*. Сведения об использовании параметра dataProvider см. в разделе «Использование параметра dataProvider» на странице 31.

Создание приложения с компонентом ComboBox

Ниже описывается процедура добавления компонента ComboBox в приложение в ходе разработки. Компонент ComboBox — редактируемый, и если ввести **Добавить** в текстовое поле, элемент добавляется в раскрывающийся список.

1 Создайте новый документ Flash (ActionScript 3.0).

2 Перетащите компонент ComboBox в рабочую область и присвойте ему имя экземпляра **aCb**. На вкладке "Параметры" задайте параметру `editable` значение `true`.

3 Откройте панель "Действия", выберите "Кадр 1" на главной временной шкале и введите следующий код.

```
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 Выберите "Управление" > "Тестировать ролик".

Создание компонента ComboBox с помощью ActionScript

В следующем примере создается компонент ComboBox с помощью ActionScript и заполняется списком университетов в области Сан-Франциско, Калифорния. Для свойства `width` экземпляра ComboBox задается достаточное значение, чтобы вместить текст запроса, а свойству `dropdownWidth` задается значение побольше, чтобы вместить самое длинное название университета.

В примере создается список университетов в экземпляре Array с использованием свойства `label` для хранения названий учебных заведений и свойства `data` для хранения URL-адресов их веб-сайтов. Экземпляр Array назначается компоненту ComboBox путем определения его свойства `dataProvider`.

Когда пользователь выбирает университет из списка, запускается событие Event.`CHANGE` и функция `changeHandler()`, которая загружает свойство `data` в URL-запрос, чтобы открыть веб-сайт учебного заведения.

Обратите внимание на то, что в последней строке свойству `selectedIndex` экземпляра ComboBox задается значение -1, чтобы после закрытия списка повторно отображался запрос. В противном случае вместо запроса отображалось бы название выбранного учебного заведения.

1 Создайте новый документ Flash (ActionScript 3.0).

2 Перетащите компонент ComboBox с панели "Компоненты" на панель "Библиотека".

3 Откройте панель "Действия", выберите "Кадр 1" на главной временной шкале и введите следующий код ActionScript.

```
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 Выберите "Управление" > "Тестировать ролик".

Этот пример можно реализовать и выполнить в среде разработки Flash, но при попытке открыть веб-сайты университетов щелчком по элементам в экземпляре ComboBox будут появляться предупреждения. Чтобы открыть полностью функциональный компонент ComboBox в Интернете, введите следующий URL-адрес:

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

Использование компонента DataGrid

Компонент DataGrid позволяет отображать данные в сетке со строками и столбцами, рисовать данные из массива или внешнего XML-файла, который можно проанализировать и добавить в массив для экземпляра DataProvider. Компонент DataGrid включает вертикальную и горизонтальную прокрутку, поддержку событий (включая поддержку редактируемых ячеек) и функции сортировки.

Можно изменять размер и настройки для таких характеристик, как шрифт, цвет и границы столбцов в сетке. Для любого столбца в сетке в качестве визуализатора ячейки можно использовать пользовательский фрагмент ролика. (Визуализатор ячейки отображает ее содержимое.) Можно отключить полосы прокрутки и с помощью методов DataGrid создать отображение стиля вида страницы. Дополнительные сведения о настройке см. в описании класса DataGridColumn в *Справочнике по языку ActionScript 3.0 и компонентам*.

См. также

Создание, заполнение и изменение размера компонента DataGrid

Настройка и сортировка компонента DataGrid

Фильтрация и форматирование данных компонента DataGrid

Взаимодействие пользователей с компонентом DataGrid

С компонентом DataGrid можно взаимодействовать с помощью мыши или клавиатуры.

Если свойство `sortableColumns` и свойство столбца `sortable` имеют значение `true`, то при щелчке по заголовку столбца данные сортируются на базе значений этого столбца. Можно отключить сортировку для отдельного столбца, задав его свойству `sortable` значение `false`.

Если свойство `resizableColumns` имеет значение `true`, можно изменять размер столбцов, перетаскивая разделители столбцов в строке заголовка.

Если щелкнуть в редактируемой ячейке, она получает фокус; при щелчке по нередактируемым ячейкам фокус не меняется. Отдельная ячейка является редактируемой, когда свойства `DataGrid.editable` и `DataGridColumn.editable` оба имеют значение `true`.

Дополнительную информацию см. в описании классов DataGrid и DataGridColumn в *Справочнике по языку ActionScript 3.0 и компонентам.*

Когда экземпляр DataGrid получает фокус в результате щелчка мыши или перехода с помощью клавиши Tab, им можно управлять следующими клавишами.

Клавиша	**Описание**
"Стрелка вниз"	Когда ячейка редактируется, точка вставки перемещается в конец текста ячейки. Если ячейка нередактируемая, клавиша "Стрелка вниз" обрабатывает выделение, как компонент List.
"Стрелка вверх"	Когда ячейка редактируется, точка вставки перемещается в начало текста ячейки. Если ячейка нередактируемая, клавиша "Стрелка вверх" обрабатывает выделение, как компонент List.
Shift+"Стрелка вверх"/"Стрелка вниз"	Если компонент DataGrid нередактируемый и свойство `allowMultipleSelection` имеет значение `true`, выделяются непрерывные строки. При изменении направления с помощью противоположной стрелки отменяется выделение выбранных строк, пока не будет пройдена первая выделенная строка, после чего строки опять начнут выделяться.
Shift+щелчок мыши	Если свойство `allowMultipleSelection` имеет значение `true`, выделяются все строки между выделенной строкой и текущей точкой вставки (выделенной ячейки).
Ctrl+щелчок мыши	Если свойство `allowMultipleSelection` имеет значение `true`, выделяются дополнительные строки, которые могут не примыкать друг к другу.
"Стрелка вправо"	Когда редактируется ячейка, точка вставки сдвигается на один символ вправо. Если ячейка нередактируемая, клавиша "Стрелка вправо" не работает.
"Стрелка влево"	Когда редактируется ячейка, точка вставки сдвигается на один символ влево. Если ячейка нередактируемая, клавиша "Стрелка влево" не работает.
Home	Выделяет первую строку в экземпляре DataGrid.

Клавиша	Описание
End	Выделяет последнюю строку в экземпляре DataGrid.
Page Up	Выделяет первую строку на странице в DataGrid. Страница содержит определенное количество строк, которое DataGrid может отображать без прокручивания.
Page Down	Выделяет последнюю строку на странице в DataGrid. Страница содержит определенное количество строк, которое DataGrid может отображать без прокручивания.
Return/Enter/Shift+Enter	Когда ячейка редактируемая, изменение применяется, а точка вставки перемещается на другую ячейку в том же столбце но на другой строке (выше или ниже в зависимости от переключателя сдвига).
Shift+Tab/Tab	Если экземпляр DataGrid редактируемый, фокус переводится на предыдущий/следующий элемент, пока не будет достигнут конец столбца, а затем на предыдущую/следующую строку, пока не будет достигнута первая или последняя ячейка. Если выделена первая ячейка, комбинация Shift+Tab переводит фокус на предыдущий элемент управления. Если выделена последняя ячейка, клавиша Tab переводит фокус на следующий элемент управления. Если экземпляр DataGrid нередактируемый, фокус переводится на предыдущий/следующий элемент управления.

Компонент DataGrid можно использовать как основу для многочисленных типов приложений на базе данных. Данные можно легко отобразить в виде отформатированной таблицы. Также можно использовать возможности визуализатора ячейки, чтобы создать более сложные и редактируемые элементы пользовательского интерфейса. Ниже приводятся примеры практического применения компонента DataGrid:

- клиенты почтовой веб-службы;
- страницы результатов поиска;
- приложения для работы с электронными таблицами, такие как калькуляторы ссуды или приложения для заполнения налоговых деклараций.

Разрабатывая приложение с использованием компонента DataGrid, лучше разобраться в дизайне компонента List, потому что класс DataGrid расширяет класс SelectableList. Дополнительные сведения о классе SelectableList и компоненте List см. в описании классов SelectableList и List в *Справочнике по языку ActionScript 3.0 и компонентам*.

Когда в приложение добавляется компонент DataGrid, его можно сделать доступным для программ чтения с экрана путем добавления следующих строк кода ActionScript.

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

Расширенный доступ для компонента включается только один раз, независимо от числа его экземпляров. Дополнительные сведения см. в главе 18 "Создание содержимого с расширенной доступностью" в руководстве *Использование Flash*.

Параметры компонента DataGrid

В Инспекторе свойств или Инспекторе компонентов можно задать следующие параметры разработки для каждого экземпляра компонента DataGrid: `allowMultipleSelection`, `editable`, `headerHeight`, `horizontalLineScrollSize`, `horizontalPageScrollSize`, `horizontalScrolllPolicy`, `resizableColumns`, `rowHeight`, `showHeaders`, `verticalLineScrollSize`, `verticalPageScrollSize` и `verticalScrollPolicy`. Каждый из этих параметров имеет соответствующее свойство ActionScript с тем же именем. Сведения о возможных значениях для этих параметров см. в описании класса DataGrid в *Справочнике по языку ActionScript 3.0 и компонентам*.

Создание приложения с компонентом DataGrid

Чтобы создать приложение с компонентом DataGrid, сначала необходимо определиться, откуда будут поступать данные. Как правило, данные поступают из экземпляра Array, который можно добавить в сетку с помощью свойства `dataProvider`. Также для добавления данных в сетку можно использовать методы классов DataGrid и DataGridColumn.

Использование локального поставщика данных с компонентом DataGrid

В этом примере создается экземпляр DataGrid для отображения состава футбольной команды. Список футболистов определяется в объекте Array (`aRoster`) и назначается свойству `dataProvider` экземпляра DataGrid.

1. В Flash выберите "Файл" > "Создать", затем выберите "Файл Flash (ActionScript 3.0)".
2. Перетащите компонент DataGrid с панели "Компоненты" в рабочую область.
3. В Инспекторе свойств введите **aDg** в качестве имени экземпляра.
4. Откройте панель "Действия", выберите "Кадр 1" на главной временной шкале и введите следующий код ActionScript.

```
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);
};
```

Функция `bldRosterGrid()` задает размер экземпляра DataGrid, а также порядок и размеры столбцов.

5. Выберите "Управление" > "Тестировать ролик".

Определение столбцов и добавление сортировки для компонента DataGrid в приложении

Обратите внимание, что для сортировки содержимого столбца DataGrid в нисходящем порядке значений, нужно щелкнуть заголовок нужного столбца.

В следующем примере используется метод `addColumn()` для добавления экземпляров DataGridColumn в компонент DataGrid. В столбцах содержатся имена футболистов и набранные ими очки. В примере также задается свойство `sortOptions`, чтобы определить параметры сортировки для каждого столбца: `Array.CASEINSENSITIVE` для столбца "Имя" и `Array.NUMERIC` для столбца "Очки". Чтобы задать соответствующий размер экземпляру DataGrid, длина определяется по количеству строк, а ширина равна 200.

1. В Flash выберите "Файл" > "Создать", затем выберите "Файл Flash (ActionScript 3.0)".
2. Перетащите компонент DataGrid с панели "Компоненты" в рабочую область.
3. В Инспекторе свойств введите **aDg** в качестве имени экземпляра.
4. Откройте панель "Действия", выберите "Кадр 1" на главной временной шкале и введите следующий код ActionScript.

```
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. Выберите "Управление" > "Тестировать ролик".

Создание экземпляра компонента DataGrid с помощью ActionScript

В этом примере создается экземпляр DataGrid с помощью ActionScript и заполняется объектом Array с именами и очками футболистов.

1. Создайте новый документ Flash (ActionScript 3.0).
2. Перетащите компонент DataGrid с панели "Компоненты" на панель "Библиотека" текущего документа.

   В результате этого компонент добавляется в библиотеку, но остается невидимым в приложении.
3. Откройте панель "Действия", выберите "Кадр 1" на главной временной шкале и введите следующий код ActionScript.

```
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);
```

   Этот код создает экземпляр DataGrid, после чего задает размер и местоположение сетки.

4 Создайте массив, добавьте в него данные и укажите его в качестве поставщика данных для экземпляра DataGrid.

```
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 Выберите "Управление" > "Тестировать ролик".

Заполнение экземпляра DataGrid XML-файлом

В следующем примере используется класс DataGridColumn для создания столбцов DataGrid. Экземпляр DataGrid заполняется путем передачи объекта XML в качестве параметра `value` конструктора `DataProvider()`.

1 Создайте новый документ Flash (ActionScript 3.0).

2 На панели "Компоненты" дважды щелкните DataGrid, чтобы добавить экземпляр в рабочую область.

3 В Инспекторе свойств введите **aDg** в качестве имени экземпляра.

4 Откройте панель "Действия", выберите "Кадр 1" на главной временной шкале и введите следующий код ActionScript.

```
import fl.controls.dataGridClasses.DataGridColumn;
import fl.data.DataProvider;

var teamXML:XML = <team>
        <player name="Player A" avg="0.293" />
        <player name="Player B" avg="0.214" />
        <player name="Player C" avg="0.317" />
</team>;

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;
```

5 Выберите "Управление" > "Тестировать ролик".

Использование компонента Label

Компонент Label отображает одну строку текста, как правило, для определения другого элемента или действия на веб-странице. Можно задать для метки форматирование с использование HTML, чтобы форматировать текст с помощью HTML-тегов. Кроме того, можно управлять выравниванием и размером метки. У компонентов Label нет границ, они не получают фокус и не передают события.

Интерактивный просмотр каждого элемента Label отражает изменения, которые были внесены в параметры в Инспекторе свойств или Инспекторе компонентов во время разработки. Компонент Label не имеет границы, поэтому для интерактивного просмотра необходимо задать его параметр text.

Взаимодействие пользователей с компонентом Label

С помощью компонента Label можно создать текстовую метку для другого компонента в форме, например метку "Имя:" слева от поля TextInput, в которое вводится имя пользователя. Вместо обычного текстового поля лучше использовать компонент Label, так как он позволяет использовать стили, чтобы обеспечить согласованный внешний вид.

Чтобы повернуть компонент Label, необходимо встроить шрифты. В противном случае они не будут отображаться при тестировании ролика.

Параметры компонента Label

В Инспекторе свойств или Инспекторе компонентов можно задать следующие параметры разработки для каждого экземпляра компонента Label : `autoSize`, `condenseWhite`, `selectable`, `text` и `wordWrap`. Каждый из этих параметров имеет соответствующее свойство ActionScript с тем же именем. Сведения о возможных значениях для этих параметров см. в описании класса Label в *Справочнике по языку ActionScript 3.0 и компонентам*.

Создание приложения с компонентом Label

Ниже описывается процедура добавления компонента Label в приложение в ходе разработки. В этом примере метка просто отображает текст "Срок действия".

1 Создайте новый документ Flash (ActionScript 3.0).

2 Перетащите компонент Label с панели "Компоненты" в рабочую область и введите следующие значения в Инспекторе свойств.

- Введите **aLabel** в качестве имени экземпляра.
- Введите значение **80** для ширины (W).
- Введите **100** для значения X.
- Введите **100** для значения Y.
- Введите значение **Срок действия** для параметра `text`.

3 Перетащите компонент TextArea в рабочую область и введите следующие значения в Инспекторе свойств.

- Введите **aTa**в качестве имени экземпляра.
- Введите **22** для значения высоты (H).
- Введите **200** для значения X.
- Введите **100** для значения Y.

4 Откройте панель "Действия", выберите "Кадр 1" на главной временной шкале и введите следующий код ActionScript.

```
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 Выберите "Управление" > "Тестировать ролик".

Создание экземпляра компонента Label с помощью ActionScript

В следующем примере создается параметр Label с помощью ActionScript. Экземпляр Label используется для определения функции компонента ColorPicker, а свойство `htmlText` — для применения форматирования к тексту экземпляра Label.

1 Создайте новый документ Flash (ActionScript 3.0).

2 Перетащите компонент Label с панели "Компоненты" на панель "Библиотека" текущего документа.

3 Перетащите компонент ColorPicker с панели "Компоненты" на панель "Библиотека" текущего документа.

4 Откройте панель "Действия", выберите "Кадр 1" на главной временной шкале и введите следующий код ActionScript.

```
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 Выберите "Управление" > "Тестировать ролик".

Использование компонента List

Компонент List — это прокручиваемый список, в котором можно выбрать один или несколько элементов. В списке могут отображаться графические элементы, включая другие компоненты. Элементы, отображаемые в списке, добавляются в диалоговом окне "Значения", которое открывается при щелчке по меткам или в полях параметров данных. Кроме того, для добавления элементов в список можно использовать методы `List.addItem()` и `List.addItemAt()`.

Компонент List использует индекс от нуля, где элемент с индексом 0 отображается вверху. При добавлении, удалении или замене элементов списка с помощью методов и свойств класса List может потребоваться указать индекс элемента списка.

Взаимодействие пользователей с компонентом List

Список можно настроить таким образом, чтобы пользователи могли выбирать один либо несколько элементов. Например, пользователю, просматривающему веб-сайт электронной торговли, требуется выбрать товар, который он хочет купить. Список, который прокручивает пользователь, состоит из 30 наименований, выбор которых осуществляется нажатием кнопки мыши.

Также экземпляр List можно создать так, чтобы в качестве строк использовались пользовательские фрагменты роликов, чтобы пользователю показывалось больше информации. Например, в почтовом приложении в каждый почтовый ящик можно добавить компонент List, в каждой строке которого содержатся значки, указывающие приоритет и состояние.

Экземпляр List получает фокус при щелчке мышью или переходе с помощью клавиши Tab, после чего им можно управлять с помощью следующих клавиш.

Клавиша	**Описание**
Буквенно-цифровые клавиши	Переходит к следующему элементу, первый символ метки которого является `Key.getAscii()`.
Ctrl	Клавиша переключения, позволяющая выделить несколько непоследовательных элементов или снять с них выделение.
"Стрелка вниз"	Выделение перемещается вниз на один элемент.
Home	Выделение перемещается в начало списка.
Page Down	Выделение перемещается вниз на одну страницу.
Page Up	Выделение перемещается вверх на одну страницу.
Shift	Позволяет выделить несколько последовательных элементов.
"Стрелка вверх"	Выделение перемещается вверх на один элемент.

Примечание. *Обратите внимание, что размеры прокрутки приводятся в пикселах, а не строках.*

Примечание. *Размер страницы, используемый клавишами Page Up и Page Down, на одну единицу меньше чем количество элементов, помещающихся в область отображения. Например, при переходе вниз по страницам в 10-строчном раскрывающемся списке показываются элементы 0-9, 9-18, 18-27 и т.д., а на следующей странице показывается последняя строка предыдущей.*

Дополнительные сведения об управлении фокусом см. в описании интерфейса IFocusManager и класса FocusManager в руководствах *Справочник по языку ActionScript 3.0 и компонентам* и «Работа с FocusManager» на странице 29.

Интерактивный просмотр каждого элемента List в рабочей области отражает изменения, которые были внесены в параметры в Инспекторе свойств или Инспекторе компонентов во время разработки.

Когда в приложение добавляется компонент List, его можно сделать доступным для программ чтения с экрана путем добавления следующих строк кода ActionScript.

```
import fl.accessibility.ListAccImpl;

ListAccImpl.enableAccessibility();
```

Расширенный доступ для компонента включается только один раз, независимо от числа его экземпляров. Дополнительные сведения см. в главе 18 "Создание содержимого с расширенной доступностью" в руководстве *Использование Flash*.

Параметры компонента List

В Инспекторе свойств или Инспекторе компонентов можно задать следующие параметры для каждого экземпляра компонента List : `allowMultipleSelection`, `dataProvider`, `horizontalLineScrollSize`, `horizontalPageScrollSize`, `horizontalScrollPolicy`, `multipleSelection`, `verticalLineScrollSize`, `verticalPageScrollSize` и `verticalScrollPolicy`. Каждый из этих параметров имеет соответствующее свойство ActionScript с тем же именем. Сведения о возможных значениях для этих параметров см. в описании класса List в *Справочнике по языку ActionScript 3.0 и компонентам*. Сведения об использовании параметра dataProvider см. в разделе «Использование параметра dataProvider» на странице 31.

Создание приложения с компонентом List

Ниже описывается процедура добавления компонента List в приложение в ходе разработки.

Добавление простого компонента List в приложение

В этом примере экземпляр List состоит из меток, определяющих модели автомобилей, и полей данных, содержащих цены.

1. Создайте новый документ Flash (ActionScript 3.0).
2. Перетащите компонент List с панели "Компоненты" в рабочую область.
3. В Инспекторе свойств выполните следующие действия.
   - Введите имя экземпляра **aList**.
   - Задайте **200** для значения ширины (W).
4. С помощью инструмента "Текст" создайте текстовое поле под `aList` и присвойте ему имя экземпляра **aTf**.
5. Откройте панель "Действия", выберите "Кадр 1" на главной временной шкале и введите следующий код ActionScript.

```
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;
}
```

В этом коде используется метод `addItem()` для заполнения экземпляра `aList` тремя элементами, каждому из которых присвоено одно значение `label`, отображаемое в списке, и значение `data`. Когда выделяется элемент в экземпляре List, прослушиватель событий вызывает функцию `showData()`, которая отображает значение `data` для выделенного элемента.

6 Выберите "Управление" > "Тестировать ролик", чтобы скомпилировать и запустить это приложение.

Заполнение экземпляра List с помощью поставщика данных

В этом примере создается экземпляр List, в котором перечисляются модели автомобилей и их цены. Однако для заполнения List используется поставщик данных, а не метод `addItem()`.

1 Создайте новый документ Flash (ActionScript 3.0).

2 Перетащите компонент List с панели "Компоненты" в рабочую область.

3 В Инспекторе свойств выполните следующие действия.

- Введите имя экземпляра **aList**.
- Задайте **200** для значения ширины (W).

4 С помощью инструмента "Текст" создайте текстовое поле под `aList` и присвойте ему имя экземпляра **aTf**.

5 Откройте панель "Действия", выберите "Кадр 1" на главной временной шкале и введите следующий код ActionScript.

```
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 Выберите "Управление" > "Тестировать ролик", чтобы просмотреть экземпляр List и его элементы.

Использование компонента List для управления экземпляром MovieClip

В следующем примере создается экземпляр List, в котором перечислены названия цветов. Выбранный цвет применяется к экземпляру MovieClip.

1 Создайте документ Flash (ActionScript 3.0).

2 Перетащите компонент List с панели "Компоненты" в рабочую область и введите следующие значения в Инспекторе свойств.

- Введите **aList** в качестве имени экземпляра.
- Введите **60** для значения высоты (H).
- Введите **100** для значения X.
- Введите **150** для значения Y.

3 Откройте панель "Действия", выберите "Кадр 1" на главной временной шкале и введите следующий код ActionScript.

```
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 Выберите "Управление" > "Тестировать ролик", чтобы запустить приложение.

5 Щелкайте цвета в списке, чтобы они отображались в экземпляре MovieClip.

Создание экземпляра компонента List с помощью ActionScript

В данном примере с использованием ActionScript создается простой список, который заполняется при помощи метода `addItem()`.

1 Создайте новый документ Flash (ActionScript 3.0).

2 Перетащите компонент List с панели "Компоненты" на панель "Библиотека".

3 Откройте панель "Действия", выберите "Кадр 1" на главной временной шкале и введите следующий код ActionScript.

```
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 Выберите "Управление" > "Тестировать ролик", чтобы запустить приложение.

Использование компонента NumericStepper

Компонент NumericStepper дает пользователю возможность выполнять пошаговый выбор в упорядоченной группе чисел. Компонент состоит из числа в текстовом поле, отображаемого рядом с маленькими кнопками "Стрелка вверх" и "Стрелка вниз". Когда пользователь нажимает кнопки, число пошагово увеличивается или уменьшается на единицу, заданную в параметре `stepSize`, пока пользователь не отпустит кнопку или не будет достигнуто максимальное или минимальное значение. Текстовое поле компонента NumericStepper также является редактируемым.

Интерактивный просмотр каждого экземпляра NumericStepper отражает настройку параметра value в Инспекторе свойств или компонентов. Однако в интерактивном просмотре нет взаимодействия мыши или клавиатуры со стрелками экземпляра NumericStepper.

Взаимодействие пользователей с компонентом NumericStepper

Компонент NumericStepper можно использовать везде, где пользователю требуется выбрать числовое значение. Например, компонент NumericStepper можно использовать в форме, чтобы указать месяц, день и год завершения срока действия кредитной карты. Также NumericStepper можно использовать, чтобы предоставить пользователю возможность увеличить или уменьшить размер шрифта.

Компонент NumericStepper обрабатывает только числовые данные. Кроме того, чтобы отображать числа, содержащие более двух знаков (например, 5246 или 1,34), необходимо изменять размер экземпляра NumericStepper в процессе разработки.

Компонент NumericStepper в приложении можно включить или отключить. В отключенном состоянии NumericStepper не реагирует на мышь или клавиатуру. Включенный компонент NumericStepper получает фокус при щелчке мыши или переходе с помощью клавиши Tab, а внутренний фокус находится в текстовом поле. Когда экземпляр NumericStepper получает фокус, им можно управлять с помощью следующих клавиш.

Клавиша	**Описание**
"Стрелка вниз"	Значение изменяется на одну единицу.
"Стрелка влево"	Перемещает точку вставки влево в пределах текстового поля.
"Стрелка вправо"	Перемещает точку вставки вправо в пределах текстового поля.
Shift+Tab	Переводит фокус на предыдущий объект.
Tab	Переводит фокус на следующий объект.
"Стрелка вверх"	Значение изменяется на одну единицу.

Дополнительные сведения об управлении фокусом см. в описании класса FocusManager в руководствах *Справочник по языку ActionScript 3.0 и компонентам* и «Работа с FocusManager» на странице 29.

Параметры компонента NumericStepper

В Инспекторе свойств или Инспекторе компонентов можно задать следующие параметры для каждого экземпляра компонента NumericStepper : `maximum`, `minimum`, `stepSize` и `value`. Каждый из этих параметров имеет соответствующее свойство ActionScript с тем же именем. Сведения о возможных значениях для этих параметров см. в описании класса NumericStepper в *Справочнике по языку ActionScript 3.0 и компонентам*.

Создание приложения с компонентом NumericStepper

Ниже описывается процедура добавления компонента NumericStepper в приложение в ходе разработки. В следующем примере в рабочую область добавляются компоненты NumericStepper и Label, а также создается прослушиватель для события `Event.CHANGE` экземпляра NumericStepper. Когда значение в NumericStepper изменяется, новое значение отображается в свойстве `text` экземпляра Label.

1. Перетащите компонент NumericStepper с панели "Компоненты" в рабочую область.
2. В Инспекторе свойств введите **aNs** в качестве имени экземпляра.
3. Перетащите компонент Label с панели "Компоненты" в рабочую область.
4. В Инспекторе свойств введите **aLabel** в качестве имени экземпляра.
5. Откройте панель "Действия", выберите "Кадр 1" на главной временной шкале и введите следующий код ActionScript.

```
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;
};
```

   В этом примере в качестве значения свойства `text` экземпляра Label задается значение NumericStepper. Функция `changeHandler()` обновляет свойство `text` экземпляра Label при каждом изменении значения NumericStepper.

6. Выберите "Управление" > "Тестировать ролик".

Создание компонента NumericStepper с помощью ActionScript

В этом примере с помощью кода ActionScript создается три экземпляра NumericStepper для ввода месяца, дня и года рождения пользователя. Кроме того, для каждого экземпляра NumericStepper добавляется по одному экземпляру Label.

1. Создайте новый документ Flash (ActionScript 3.0).
2. Перетащите компонент Label на панель "Библиотека".
3. Перетащите компонент NumericStepper на панель "Библиотека".
4. Откройте панель "Действия", выберите "Кадр 1" на главной временной шкале и введите следующий код ActionScript.

```
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 Выберите "Управление" > "Тестировать ролик", чтобы запустить приложение.

Использование компонента ProgressBar

Компонент ProgressBar отображает прогресс загрузки содержимого, чтобы успокоить пользователя, когда загружается большой файл и задерживается выполнение приложения. Компонент ProgressBar полезно использовать для отображения прогресса загрузки изображений или компонентов приложения. Процесс загрузки может быть как определенным, так и неопределенным. *Определенный* индикатор прогресса является линейным представлением хода выполнения задачи, которое используется, когда объем загружаемого содержимого известен. *Неопределенный* индикатор прогресса используется, когда объем загружаемого содержимого неизвестен. Также можно добавить компонент Label, чтобы отображать выполнение загрузки в процентном выражении.

Компонент ProgressBar использует 9-фрагментную шкалу и включает обложку полосы, обложку трека и неопределенную обложку.

Взаимодействие пользователей с компонентом ProgressBar

Существует три режима использования компонента ProgressBar. Чаще всего используются режимы события и опроса. Эти режимы определяют процесс загрузки, отправляющий события `progress` и `complete` (режимы событий и опроса) или сообщающий свойства `bytesLoaded` и `bytesTotal` (режим опроса). Компонентом ProgressBar можно также воспользоваться в ручном режиме, установив вручную свойства `maximum`, `minimum` и `value` и выполнив вызовы метода `ProgressBar.setProgress()`. Можно задать свойство indeterminate, чтобы указать, будет ли экземпляр ProgressBar заполняться полосками и иметь источник с неизвестным размером (`true`), либо он будет заполняться сплошной заливкой и иметь источник с известным размером (`false`).

Чтобы задать режим экземпляра ProgressBar, необходимо определить свойство `mode` либо через параметр `mode` в Инспекторе свойств или Инспекторе компонентов, либо с помощью ActionScript.

Если компонент ProgressBar используется для отображения состояния обработки, например разбора 100000 элементов, и если он находится в одном цикле кадра, то обновления ProgressBar останутся незаметны, так как не будет перерисовываться экран.

Параметры компонента ProgressBar

В Инспекторе свойств или Инспекторе компонентов можно задать следующие параметры для каждого экземпляра компонента ProgressBar : `direction`, `mode` и `source`. Каждый из этих параметров имеет соответствующее свойство ActionScript с тем же именем.

Можно создать код ActionScript для управления этими и дополнительными параметрами компонента ProgressBar с помощью его свойств, методов и событий. Дополнительные сведения см. в описании класса ProgressBar в *Справочнике по языку ActionScript 3.0 и компонентам*.

Создание приложения с компонентом ProgressBar

Ниже описывается процедура добавления компонента NumericStepper в приложение в ходе разработки. В этом примере компонент ProgressBar использует режим событий. В режиме событий загружаемое содержимое отправляет события `progress` и `complete`, которые экземпляр ProgressBar использует для отображения прогресса. Когда происходит событие `progress`, обновляется метка, отражающая загруженного содержимого. Когда происходит событие `complete`, отображается текст "Загрузка завершена" и значение свойства `bytesTotal`, представляющее размер файла.

1. Создайте новый документ Flash (ActionScript 3.0).
2. Перетащите компонент ProgressBar с панели "Компоненты" в рабочую область.
   - В Инспекторе свойств введите **aPb** в качестве имени экземпляра.
   - В разделе "Параметры" введите значение **200** для значения X.
   - Введите **260** для значения Y.
   - Выберите значение `event` для параметра `mode`.
3. Перетащите компонент Button с панели "Компоненты" в рабочую область.
   - В Инспекторе свойств введите **loadButton** в качестве имени экземпляра.
   - Введите **220** для параметра X.
   - Введите **290** для параметра Y.
   - Введите значение **Загрузить звук** для параметра `label`.
4. Перетащите компонент Label в рабочую область и присвойте экземпляру имя **progLabel**.
   - Введите **150** для значения ширины (W).
   - Введите **200** для параметра X.
   - Введите **230** для параметра Y.
   - В разделе "Параметры" очистите значение параметра `text`.
5. Откройте панель "Действия", выберите "Кадр 1" на главной временной шкале и введите следующий код ActionScript, загружающий аудиофайл в формате mp3.

```
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 Выберите "Управление" > "Тестировать ролик".

Создание приложения с компонентом ProgressBar в режиме опроса

В следующем примере компонент ProgressBar создается в режиме опроса. В режиме опроса прогресс определяется путем прослушивания событий `progress` для загружаемого содержимого, а для расчета прогресса используются свойства `bytesLoaded` и `bytesTotal`. В данном примере загружается объект Sound, прослушиваются его события `progress`, и рассчитывается процент загруженного содержимого с помощью его свойств `bytesLoaded` и `bytesTotal`. Процент загруженного содержимого отображается на метке и на панели вывода.

1 Создайте новый документ Flash (ActionScript 3.0).

2 Перетащите компонент ProgressBar с панели "Компоненты" в рабочую область и введите следующие значения в Инспекторе свойств.

- Введите **aPb** в качестве имени экземпляра.
- Введите **185** для значения X.
- Введите **225** для значения Y.

3 Перетащите компонент Label в рабочую область и введите следующие значения в Инспекторе свойств.

- Введите **progLabel** в качестве имени экземпляра.
- Введите **180** для значения X.

- Введите **180** для значения Y.
- В разделе "Параметры" очистите значение параметра text.

4 Откройте панель "Действия", выберите "Кадр 1" на главной временной шкале и введите следующий код ActionScript, который создает объект Sound (`aSound`) и вызывает метод `loadSound()` для загрузки звука в объект Sound.

```
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 Выберите "Управление" > "Тестировать ролик", чтобы запустить приложение.

Создание приложения с компонентом ProgressBar в ручном режиме

В следующем примере компонент ProgressBar создается в ручном режиме. В ручном режиме прогресс необходимо задать вручную с помощью метода `setProgress()` и вводить текущее и максимальное значения для вычисления прогресса. В ручном режиме не требуется задавать свойство `source`. В примере используется компонент NumericStepper с максимальным значением 250 для увеличения значения ProgressBar. Когда значение в экземпляре NumericStepper изменяется и отправляет событие `CHANGE`, обработчик события (`nsChangeHander`) вызывает метод `setProgress()` для увеличения значения ProgressBar. Также на основе максимального значения вычисляется и отображается процент выполненного процесса.

1 Создайте новый документ Flash (ActionScript 3.0).

2 Перетащите компонент ProgressBar с панели "Компоненты" в рабочую область и введите для него следующие значения в Инспекторе свойств.

- Введите **aPb** в качестве имени экземпляра.
- Введите **180** для значения X.
- Введите **175** для значения Y.

3 Перетащите компонент NumericStepper в рабочую область и введите следующие значения в Инспекторе свойств.

- Введите **aNs** в качестве имени экземпляра.
- Введите **220** для значения X.
- Введите **215** для значения Y.

- В разделе "Параметры" введите **250** для параметра maximum, **0** для параметра minimum, **1** для параметра `stepSize` и **0** для параметра `value`.

4 Перетащите компонент Label в рабочую область и введите следующие значения в Инспекторе свойств.

- Введите **progLabel** в качестве имени экземпляра.
- Введите **150** для значения ширины (W).
- Введите **180** для значения X.
- Введите **120** для значения Y.
- В разделе "Параметры" очистите значение "Метка" параметра text.

5 Откройте панель "Действия", выберите "Кадр 1" на главной временной шкале и введите следующий код.

```
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 Выберите "Управление" > "Тестировать ролик", чтобы запустить приложение.

7 Щелкните стрелку вверх экземпляра NumericStepper, чтобы увеличить значение ProgressBar.

Создание компонента ProgressBar с помощью ActionScript

В этом примере компонент ProgressBar создается с помощью ActionScript. Кроме того, дублируется функция, реализованная в предыдущем примере, которая создает ProgressBar в ручном режиме.

1 Создайте новый документ Flash (ActionScript 3.0).

2 Перетащите компонент ProgressBar на панель "Библиотека".

3 Перетащите компонент NumericStepper на панель "Библиотека".

4 Перетащите компонент Label на панель "Библиотека".

5 Откройте панель "Действия", выберите "Кадр 1" на главной временной шкале и введите следующий код.

```
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 Выберите "Управление" > "Тестировать ролик", чтобы запустить приложение.

7 Щелкните стрелку вверх экземпляра NumericStepper, чтобы увеличить значение ProgressBar.

Использование компонента RadioButton

Компонент RadioButton позволяет поставить пользователя перед выбором единственного варианта из нескольких. Этот компонент должен использоваться в группе, где существует не менее двух экземпляров RadioButton. В любое время можно выбрать только одного члена данной группы. Выбор переключателя в группе снимает выделение с выбранного в данный момент переключателя в этой группе. Параметр `groupName` задается для того, чтобы указать, к какой группе принадлежит переключатель.

Переключатель является основной частью многих приложений с веб-формами. Переключатели можно использовать, когда пользователь должен выбрать один вариант из нескольких. Например, переключатели можно использовать в форме, чтобы узнать у заказчика, какой кредитной картой он хочет воспользоваться.

Взаимодействие пользователей с компонентом RadioButton

Переключатель можно включить или отключить. Отключенный переключатель не реагирует на информацию, поступающую с клавиатуры или мыши. Когда пользователь нажимает на группу компонентов RadioButton или переходит к ней с помощью клавиши Tab, в фокусе оказывается только выделенный переключатель. После этого пользователь может управлять им с помощью следующих клавиш.

Клавиша	Описание
"Стрелка вверх"/"Стрелка влево"	Выделение перемещается на предыдущий переключатель в группе.
"Стрелка вниз"/"Стрелка вправо"	Выделение перемещается на следующий переключатель в группе.
Tab	Фокус перемещается с группы переключателей на следующий компонент.

Дополнительные сведения об управлении фокусом см. в описании класса IFocusManager и классу FocusManager в руководствах *Справочник по языку ActionScript 3.0 и компонентам* и «Работа с FocusManager» на странице 29.

Интерактивный просмотр каждого элемента RadioButton в рабочей области отражает изменения, которые были внесены в параметры в Инспекторе свойств или Инспекторе компонентов во время разработки. Однако в интерактивном просмотре не отображается взаимоисключение выделения. Если задать значение `true` параметрам selected двух переключателей в одной группе, они оба отображаются выбранными, даже если только последний созданный экземпляр выбран во время выполнения. Дополнительные сведения см. в разделе «Параметры компонента RadioButton» на странице 81.

Когда в приложение добавляется компонент RadioButton, его можно сделать доступным для программ чтения с экрана путем добавления следующих строк кода.

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

Расширенный доступ для компонента включается только один раз, независимо от числа имеющихся экземпляров. Дополнительные сведения см. в главе 18 "Создание содержимого с расширенной доступностью" в руководстве "Использование Flash".

Параметры компонента RadioButton

В Инспекторе свойств или Инспекторе компонентов можно задать следующие параметры разработки для каждого экземпляра компонента RadioButton : `groupName`, `label`, `LabelPlacement`, `selected` и `value`. Каждый из этих параметров имеет соответствующее свойство ActionScript с тем же именем. Сведения о возможных значениях для этих параметров см. в описании класса RadioButton в *Справочнике по языку ActionScript 3.0 и компонентам*.

Можно написать код ActionScript, чтобы задать дополнительные параметры для экземпляров RadioButton с помощью методов, свойств и событий класса RadioButton.

Создание приложения с компонентом RadioButton

Ниже описывается процедура добавления компонентов RadioButton в приложение в ходе разработки. В данном примере экземпляры RadioButton используются для выбора ответа "да" или "нет" на заданный вопрос. Данные, полученные от RadioButton, отображаются в экземпляре TextArea.

1 Создайте новый документ Flash (ActionScript 3.0).

2 Перетащите два компонента RadioButton с панели "Компоненты" в рабочую область.

3 Выберите первый переключатель. В Инспекторе свойств присвойте ему имя экземпляра **yesRb** и имя группы **rbGroup.**

4 Выберите второй переключатель. В Инспекторе свойств присвойте ему имя экземпляра **noRb** и имя группы **rbGroup.**

5 Перетащите компонент TextArea с панели "Компоненты" в рабочую область и присвойте ему имя экземпляра **aTa**.

6 Откройте панель "Действия", выберите "Кадр 1" на главной временной шкале и введите следующий код ActionScript.

```
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 Выберите "Управление" > "Тестировать ролик", чтобы запустить приложение.

Создание компонента RadioButton с помощью ActionScript

В этом примере с помощью ActionScript создается три экземпляра RadioButton для красного, синего и зеленого цветов и рисуется серое окно. Свойство `value` каждого экземпляра RadioButton определяет шестнадцатеричное значение цвета, связанное с данным переключателем. Когда пользователь щелкает по одному из экземпляров RadioButton, функция `clickHandler()` вызывает метод `drawBox()` и передает цвет из свойства `value` экземпляра RadioButton для изменения цвета серого окна.

1 Создайте новый документ Flash (ActionScript 3.0).

2 Перетащите компонент RadioButton на панель "Библиотека".

3 Откройте панель "Действия", выберите "Кадр 1" на главной временной шкале и введите следующий код ActionScript.

```
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 Выберите "Управление" > "Тестировать ролик", чтобы запустить приложение.

Дополнительные сведения см. в описании класса RadioButton в *Справочнике по языку ActionScript 3.0 и компонентам*.

Использование компонента ScrollPane

Компонент ScrollPane можно использовать для отображения содержимого, размер которого превышает вмещающую его область. Например, если имеется большое изображение, а в приложении для него есть только небольшое место, можно загрузить его в экземпляр ScrollPane. ScrollPane может вмещать фрагменты роликов, а также файлы в формате JPEG, PNG, GIF и SWF.

Компоненты, такие как ScrollPane и UILoader отправляют события `complete`, что позволяет определять время завершения загрузки. Чтобы задать свойства для содержимого компонента ScrollPane или UILoader, необходимо прослушивать событие `complete` и задать нужное свойство в обработчике событий. Например, следующий код создает прослушиватель для события Event.`COMPLETE` и обработчик события, который задает свойству `alpha` содержимого экземпляра ScrollPane значение 0,5.

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

Если при загрузке содержимого в экземпляр ScrollPane указывается местоположение, необходимо указать точку с координатами (0, 0). Например следующий код правильно загружает экземпляр ScrollPane, так как окно рисуется в точке с координатами (0, 0).

```
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
```

Дополнительные сведения см. в описании класса ScrollPane в *Справочнике по языку ActionScript 3.0 и компонентам*.

Взаимодействие пользователей с компонентом ScrollPane

Компонент ScrollPane можно включить или отключить. Отключенный ScrollPane не реагирует на информацию, поступающую с клавиатуры или мыши. Когда экземпляр ScrollPane получает фокус, им можно управлять с помощью следующих клавиш.

Клавиша	**Описание**
"Стрелка вниз"	Содержимое прокручивается вверх на одну строку по вертикали.
"Стрелка вверх"	Содержимое прокручивается вниз на одну строку по вертикали.
End	Содержимое прокручивается в конец экземпляра ScrollPane.
"Стрелка влево"	Содержимое прокручивается вправо на одну линию по горизонтали.
"Стрелка вправо"	Содержимое прокручивается влево на одну линию по горизонтали.
Home	Содержимое прокручивается в начало экземпляра ScrollPane.
End	Содержимое прокручивается в конец экземпляра ScrollPane.
Page Down	Содержимое прокручивается вверх на одну страницу по вертикали.
Page Up	Содержимое прокручивается вниз на одну страницу по вертикали.

С помощью мыши пользователь может взаимодействовать как с содержимым экземпляра ScrollPane, так и с его полосами прокрутки по вертикали и горизонтали. Пользователь может перетаскивать содержимое с помощью мыши, если свойству `scrollDrag` задано значение `true`. Появление указателя "рука" на содержимом говорит о том, что его можно перетаскивать. В отличие от большинства других элементов управления, действия начинаются в момент нажатия кнопки мыши и продолжаются, пока пользователь ее не отпустит. Если содержимое имеет действительные позиции табуляции, свойству `scrollDrag` необходимо задать значение false. В противном случае все щелчки мыши по содержимому будут вызывать перетаскивание прокрутки.

Параметры компонента ScrollPane

В Инспекторе свойств или Инспекторе компонентов можно задать следующие параметры для каждого экземпляра ScrollPane: `horizontalLineScrollSize`, `horizontalPageScrollSize`, `horizontalScrollPolicy`, `scrollDrag`, `source`, `verticalLineScrollSize`, `verticalPageScrollSize` и `verticalScrollPolicy`. Каждый из этих параметров имеет соответствующее свойство ActionScript с тем же именем. Сведения о возможных значениях для этих параметров см. в описании класса ScrollPane в *Справочнике по языку ActionScript 3.0 и компонентам*.

Можно создать код ActionScript для управления этими и дополнительными параметрами компонента ScrollPane с помощью его свойств, методов и событий.

Создание приложения с компонентом ScrollPane

Ниже описывается процедура добавления компонента ScrollPane в приложение в ходе разработки. В этом примере экземпляр ScrollPane загружает изображение из местоположения, указанного в свойстве `source`.

1. Создайте новый документ Flash (ActionScript 3.0).
2. Перетащите компонент ScrollPane с панели "Компоненты" в рабочую область и присвойте ему имя экземпляра **aSp**.
3. Откройте панель "Действия", выберите "Кадр 1" на главной временной шкале и введите следующий код ActionScript.

```
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. Выберите "Управление" > "Тестировать ролик", чтобы запустить приложение.

Создание компонента ScrollPane с помощью ActionScript

В этом примере создается экземпляр ScrollPane, задается его размер, и загружается изображение с помощью свойства `source` . Кроме того, создаются два прослушивателя событий. Первый прослушивает событие `scroll` и отображает позицию изображения по мере прокручивания по вертикали или горизонтали. Второй прослушивает событие `complete` и отображает на панели вывода сообщение о том, что загрузка изображения завершена.

В этом примере с помощью ActionScript создается экземпляр ScrollPane, в который помещается объект MovieClip (красное окно) шириной 150 пикселов и высотой 300 пикселов.

1. Создайте новый документ Flash (ActionScript 3.0).

2 Перетащите компонент ScrollPane с панели "Компоненты" на панель "Библиотека".

3 Перетащите компонент DataGrid с панели "Компоненты" на панель "Библиотека".

4 Откройте панель "Действия", выберите "Кадр 1" на главной временной шкале и введите следующий код ActionScript.

```
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 Выберите "Управление" > "Тестировать ролик", чтобы запустить приложение.

Использование компонента Slider

Компонент Slider дает пользователю возможность выбрать значение путем перемещения графического *ползунка* между конечными точками линии, соответствующей диапазону значений. Ползунок можно использовать, когда пользователю нужно выбрать определенное значение, например число или процент. Также можно использовать ActionScript, чтобы значение ползунка оказывало влияние на поведение второго объекта. Например, можно связать ползунок с картинкой, чтобы увеличивать или уменьшать ее в зависимости от относительного положения, или значения, ползунка.

Текущее значение экземпляра Slider определяется относительным положением ползунка между конечными точками линии или минимальным и максимальным значениями экземпляра Slider.

Компонент Slider позволяет выбрать значение из непрерывного диапазона между минимальным и максимальным значениями. Однако можно задать параметр `snapInterval`, чтобы указать интервалы между минимальным и максимальным значениями. Экземпляр Slider может показывать на линии деления, независимые от заданных значений ползунка, через указанные интервалы.

По умолчанию ползунок имеет горизонтальную ориентацию, но его можно разместить по вертикали, задав значение vertical для параметра `direction`. Линия ползунка находится между конечными точками, а деления шкалы располагаются слева направо прямо над линией.

Взаимодействие пользователей с компонентом Slider

Когда экземпляр Slider получает фокус, им можно управлять с помощью следующих клавиш.

Клавиша	Описание
"Стрелка вправо"	Увеличивает соответствующее значение для горизонтального ползунка.
"Стрелка вверх"	Увеличивает соответствующее значение для вертикального ползунка.
"Стрелка влево"	Уменьшает соответствующее значение для горизонтального ползунка.
"Стрелка вниз"	Уменьшает соответствующее значение для вертикального ползунка.
Shift+Tab	Переводит фокус на предыдущий объект.
Tab	Переводит фокус на следующий объект.

Дополнительные сведения об управлении фокусом см. в описании класса IFocusManager и классу FocusManager в руководствах *Справочник по языку ActionScript 3.0 и компонентам* и «Работа с FocusManager» на странице 29.

Интерактивный просмотр каждого элемента Slider отражает изменения, которые были внесены в параметры в Инспекторе свойств или Инспекторе компонентов во время разработки.

Параметры компонента Slider

В Инспекторе свойств или Инспекторе компонентов можно задать следующие параметры разработки для каждого экземпляра компонента Slider : `direction`, `liveDragging`, `maximum`, `minimum`, `snapInterval`, `tickInterval` и `value`. Каждый из этих параметров имеет соответствующее свойство ActionScript с тем же именем. Сведения о возможных значениях для этих параметров см. в описании класса Slider в *Справочнике по языку ActionScript 3.0 и компонентам*.

Создание приложения с компонентом Slider

В следующем примере создается экземпляр Slider, чтобы пользователь мог сообщить, насколько ему понравилось предполагаемое событие. Пользователь перемещает экземпляр Slider вправо или влево, чтобы выразить большую или меньшую степень удовлетворения.

1. Создайте новый документ Flash (ActionScript 3.0).
2. Перетащите компонент Label с панели "Компоненты" в центр рабочей области.
   - Присвойте ему имя экземпляра **valueLabel**.
   - Задайте значение **0%** для параметра `text`.
3. Перетащите компонент Slider с панели "Компоненты" и поместите его под экземпляром `value_lbl`.
   - Присвойте ему имя экземпляра **aSlider**.
   - Введите **200** для значения ширины (W).
   - Введите **10** для значения высоты (H).
   - Задайте значение **100** параметру `maximum`.
   - Задайте значение **10** параметрам `snapInterval` и `tickInterval`.
4. Перетащите еще один экземпляр Label с панели "Библиотека" прямо под `aSlider` .
   - Присвойте ему имя экземпляра **promptLabel.**
   - Введите 250 для значения ширины (W).
   - Введите 22 для значения высоты (H).

- Введите значение **Насколько вам понравилось** для параметра `text`.

5 Откройте панель "Действия", выберите "Кадр 1" на главной временной шкале и введите следующий код ActionScript.

```
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 Выберите "Управление" > "Тестировать ролик".

В этом примере при перемещении ползунка от одного интервала к другому прослушиватель события `SliderEvent.CHANGE` обновляет свойство `text` экземпляра`valueLabel`, отображая процент, соответствующий позиции ползунка.

Создание приложения с компонентом Slider помощью ActionScript

В следующем примере компонент Slider создается с помощью ActionScript. Загружается изображение цветка, и с помощью экземпляра Slider пользователю предоставляется возможность сделать его более насыщенным или более прозрачным путем изменения свойства `alpha` в соответствии с позицией ползунка.

1 Создайте новый документ Flash (ActionScript 3.0).

2 Перетащите компоненты Label и Slider с панели "Компоненты" на панель "Библиотека" текущего документа.

В результате этого компоненты добавляются в библиотеку, но остаются невидимым в приложении.

3 Откройте панель "Действия", выберите Кадр 1 на основной временной шкале и введите следующий код, чтобы создать и разместить экземпляры компонентов:

```
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 Выберите "Управление" > "Тестировать ролик", чтобы запустить приложение.

5 Переместите ползунок экземпляра Slider влево, чтобы сделать изображение прозрачнее, или вправо, чтобы оно стало более насыщенным.

Использование компонента TextArea

Компонент TextArea является обложкой для объекта TextField в ActionScript. Компонент TextArea можно использовать для отображения текста, а также для редактирования и получения введенного текста, если свойство `editable` имеет значение `true`. Компонент может отображать или получать многострочный текст и переносить строки, если свойство `wordWrap` имеет значение `true`. С помощью свойства `restrict` можно ограничить символы, которые может ввести пользователь, а свойство `maxChars` позволяет указать максимально допустимое количество символов. Если текст выходит за рамки горизонтальных или вертикальных границ области текста, автоматически появляются полосы прокрутки, если связанным с ними свойствам `horizontalScrollPolicy` и `verticalScrollPolicy` не задано значение `off`.

Компонент TextArea можно использовать, когда требуется добавить многострочное текстовое поле. Например, можно использовать компонент TextArea в качестве поля для комментария в форме. Можно настроить прослушиватель событий, проверяющий наличие текста в поле, когда пользователь покидает его с помощью клавиши Tab. Прослушиватель может показать сообщение об ошибке, указывая, что в поле необходимо ввести комментарий.

Если требуется однострочное текстовое поле, следует использовать компонент TextInput .

Можно задать свойство `textFormat` с помощью метода `setStyle()`, чтобы изменить стиль текста, отображаемого в экземпляре TextArea. Компонент TextArea можно форматировать и с помощью HTML, используя свойство `htmlText` в ActionScript. А чтобы замаскировать текст звездочками, можно задать свойству `displayAsPassword` значение `true`. Если задать свойству `condenseWhite` значение `true`, Flash убирает лишнее пространство между символами в новом тексте, возникшее в результате использования пробелов, разрыва строки и т.д. Это свойство не действует на текст, уже имеющийся в элементе управления.

Взаимодействие пользователей с компонентом TextArea

Компонент TextArea можно включить или выключить в приложении. В отключенном состоянии TextArea не реагирует на мышь или клавиатуру. Включенный компонент следует тем же правилам получения фокуса, выделения и навигации, что и объект TextField в ActionScript. Когда экземпляр TextArea получает фокус, им можно управлять с помощью следующих клавиш.

Клавиша	**Описание**
Клавиши со стрелками	Перемещают точку вставки вверх, вниз, влево или вправо, если текст является редактируемым.
Page Down	Перемещает точку вставки в конец текста, если он является редактируемым.
Page Up	Перемещает точку вставки в начало текста, если он является редактируемым.
Shift+Tab	Перемещает фокус к предыдущему объекту в порядке табуляции.
Tab	Перемещает фокус к следующему объекту в порядке табуляции.

Дополнительные сведения об управлении фокусом см. в описании класса FocusManager в руководствах *Справочник по языку ActionScript 3.0 и компонентам* и «Работа с FocusManager» на странице 29.

Параметры компонента TextArea

В Инспекторе свойств или Инспекторе компонентов можно задать следующие параметры разработки для каждого экземпляра TextArea : `condenseWhite`, `editable`, `hortizontalScrollPolicy`, `maxChars`, `restrict`, `text`, `verticalScrollPolicy` и `wordwrap`. Каждый из этих параметров имеет соответствующее свойство ActionScript с тем же именем. Сведения о возможных значениях для этих параметров см. в описании класса TextArea в *Справочнике по языку ActionScript 3.0 и компонентам*.

Интерактивный просмотр каждого элемента TextArea отражает изменения, которые были внесены в параметры в Инспекторе свойств или Инспекторе компонентов во время разработки. Если требуется полоса прокрутки, она отображается в интерактивном просмотре, но не работает. В интерактивном просмотре текст не выделяется. Кроме того, нельзя вводить текст в экземпляре компонента в рабочей области.

Можно создать код ActionScript для управления этими и дополнительными параметрами компонента TextArea с помощью его свойств, методов и событий. Дополнительные сведения см. в описании класса TextArea в *Справочнике по языку ActionScript 3.0 и компонентам*.

Создание приложения с компонентом TextArea

Ниже описывается процедура добавления компонента TextArea в приложение в ходе разработки. В примере создается обработчик события `focusOut` для экземпляра TextArea, который проверяет, сделал ли пользователь ввод в текстовую область, прежде чем перевести фокус на другой компонент интерфейса.

1 Создайте новый документ Flash (ActionScript 3.0).

2 Перетащите компонент TextArea с панели "Компоненты" в рабочую область и присвойте ему имя экземпляра **aTa**. Оставьте значения параметров, используемые по умолчанию.

3 Перетащите второй компонент TextArea с панели "Компоненты" в рабочую область, поместите его под первым и присвойте ему имя экземпляра **bTa**. Оставьте значения параметров, используемые по умолчанию.

4 Откройте панель "Действия", выберите "Кадр 1" на главной временной шкале и введите следующий код ActionScript.

```
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();
}
```

В этом примере ограничиваются символы, которые можно ввести в текстовую область `aTa`: разрешается использовать только символы нижнего регистра, запятую, апостроф и пробелы. Также создаются обработчики для событий `change`, `KEY_FOCUS_CHANGE` и `MOUSE_FOCUS_CHANGE` для текстовой области `aTa`. Функция `changeHandler()` автоматически отображает текст, введенный в текстовой области `aTa`, в области `bTa`, присваивая свойство `aTa.text` в качестве значения для `bTa.text` при получении каждого события `change`. Функция `k_m_fHandler()` для событий `KEY_FOCUS_CHANGE` и `MOUSE_FOCUS_CHANGE` запрещает перевод фокуса в следующее поле при нажатии клавиши Tab, если не введен текст в текущее поле. Это достигается путем предотвращения поведения по умолчанию.

5 Выберите "Управление" > "Тестировать ролик".

Если нажать клавишу Tab, чтобы перевести фокус на вторую текстовую область, не введя текст в первую, появится сообщение об ошибке, а фокус вернется в первую текстовую область. По мере ввода текста в первую текстовую область, он будет дублироваться во второй области.

Создание компонента TextArea с помощью ActionScript

В следующем примере компонент TextArea создается с помощью ActionScript. Свойству `condenseWhite` задается значение `true`, чтобы сокращать расстояние между символами. Свойству `htmlText` назначается текст, чтобы можно было использовать атрибуты форматирования текста HTML.

1 Создайте новый документ Flash (ActionScript 3.0).

2 Перетащите компонент TextArea на панель "Библиотека".

3 Откройте панель "Действия", выберите "Кадр 1" на главной временной шкале и введите следующий код ActionScript.

```
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);
```

В этом примере используется свойство `htmlText` для применения атрибутов HTML для полужирного и подчеркнутого начертания к блоку текста, который отображается в текстовой области `a_ta`. Кроме того, свойству `condenseWhite` задается значение `true`, чтобы уменьшить расстояние между символами в текстовом блоке. Метод `setSize()` задает высоту и ширину текстовой области, а метод `move()` задает ее местоположение. Метод `addChild()` добавляет экземпляр TextArea в рабочую область.

4 Выберите "Управление" > "Тестировать ролик".

Использование компонента TextInput

Компонент TextInput является однострочной текстовой обложкой для встроенного объекта ActionScript TextField. Если требуется добавить многострочное текстовое поле, используйте компонент TextArea. Например, можно использовать компонент TextInput в качестве поля для пароля в форме. Также можно настроить прослушиватель событий, проверяющий наличие достаточного количества символов в поле, когда пользователь покидает его с помощью клавиши Tab. Прослушиватель может показывать сообщение об ошибке, указывающее на необходимость ввода правильного количества символов.

Можно задать свойство `textFormat` с помощью метода `setStyle()`, чтобы изменить стиль текста, отображаемого в экземпляре TextInput. Компонент TextInput можно форматировать с помощью HTML или как поле пароля, маскирующее текст.

Взаимодействие пользователей с компонентом TextInput

Компонент TextInput можно включить или выключить в приложении. В отключенном состоянии TextInput не реагирует на мышь или клавиатуру. Включенный компонент следует тем же правилам получения фокуса, выделения и навигации, что и объект TextField в ActionScript. Когда экземпляр TextInput получает фокус, им можно управлять также с помощью следующих клавиш.

Клавиша	**Описание**
Клавиши со стрелками	Перемещают точку вставки на один символ влево или вправо.
Shift+Tab	Переводит фокус на предыдущий объект.
Tab	Переводит фокус на следующий объект.

Дополнительные сведения об управлении фокусом см. в описании класса FocusManager в руководствах *Справочник по языку ActionScript 3.0 и компонентам* и «Работа с FocusManager» на странице 29.

Интерактивный просмотр каждого элемента TextInput отражает изменения, которые были внесены в параметры в Инспекторе свойств или Инспекторе компонентов во время разработки. В интерактивном просмотре текст не выделяется. Кроме того, нельзя вводить текст в экземпляре компонента в рабочей области.

При добавлении в приложение компонент TextInput можно сделать доступным для программ чтения с экрана с помощью панели "Расширенный доступ".

Параметры компонента TextInput

В Инспекторе свойств или Инспекторе компонентов можно задать следующие параметры разработки для каждого экземпляра TextInput : `editable`,`displayAsPassword`, `maxChars`, `restrict` и `text`. Каждый из этих параметров имеет соответствующее свойство ActionScript с тем же именем. Сведения о возможных значениях для этих параметров см. в описании класса TextInput в *Справочнике по языку ActionScript 3.0 и компонентам*.

Можно создать код ActionScript для управления этими и дополнительными параметрами компонента TextInput с помощью его свойств, методов и событий. Дополнительные сведения см. в описании класса TextInput в *Справочнике по языку ActionScript 3.0 и компонентам*.

Создание приложения с компонентом TextInput

Ниже описывается процедура добавления компонента TextInput в приложение. В примере используется два поля TextInput для получения и подтверждения пароля. Создается прослушиватель событий, который проверяет, чтобы было введено не меньше восьми символов и чтобы в двух полях был введен одинаковый текст.

1 Создайте новый документ Flash (ActionScript 3.0).

2 Перетащите компонент Label с панели "Компоненты" в рабочую область и введите следующие значения в Инспекторе свойств.

- Введите имя экземпляра **pwdLabel**.
- Введите **100** для значения ширины (W).
- Введите **50** для значения X.
- Введите **150** для значения Y.
- В разделе "Параметры" введите значение**Пароль:** для параметра text.

3 Перетащите второй компонент Label с панели "Компоненты" в рабочую область и введите для него следующие значения.

- Введите имя экземпляра **confirmLabel**.
- Введите **100** для значения ширины (W).
- Введите **50** для значения X.
- Введите **200** для значения Y.
- В разделе "Параметры" введите значение**Подтвердитепароль:** для параметра text.

4 Перетащите компонент TextInput с панели "Компоненты" в рабочую область и введите для него следующие значения.

- Введите имя экземпляра **pwdTi**.
- Введите **150** для значения ширины (W).
- Введите **190** для значения X.

- Введите **150** для значения Y.
- В разделе "Параметры" дважды щелкните значение параметра `displayAsPassword` и выберите **true**. В результате этого значение, введенное в текстовое поле, будет маскироваться звездочками.

5 Перетащите второй компонент TextInput с панели "Компоненты" в рабочую область и введите для него следующие значения:

- Введите имя экземпляра **confirmTi**.
- Введите **150** для значения ширины (W).
- Введите **190** для значения X.
- Введите **200** для значения Y.
- В разделе "Параметры" дважды щелкните значение параметра `displayAsPassword` и выберите **true**. В результате этого значение, введенное в текстовое поле, будет маскироваться звездочками.

6 Откройте панель "Действия", выберите "Кадр 1" на главной временной шкале и введите следующий код ActionScript.

```
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);
```

Этот код создает прослушиватель события `enter` для экземпляра TextInput с именем `confirmTi`. Если два пароля не соответствуют или пользователь вводит меньше 8 символов, выводится сообщение "Пароль неверный. Введите его еще раз". Если пароли содержат по 8 символов или более и соответствуют друг другу, введенное значение отображается на панели вывода.

7 Выберите "Управление" > "Тестировать ролик".

Создание компонента TextInput с помощью ActionScript

В следующем примере компонент TextInput создается с помощью ActionScript. Также создается экземпляр Label, чтобы попросить пользователя ввести свое имя. В примере задается свойство `restrict` компонента, чтобы использовались только заглавные и строчные буквы, точка и пробел. Также создается объект TextFormat для форматирования текста в компонентах Label и TextInput.

1 Создайте новый документ Flash (ActionScript 3.0).

2 Перетащите компонент TextInput с панели "Компоненты" на панель "Библиотека".

3 Перетащите компонент Label с панели "Компоненты" на панель "Библиотека".

4 Откройте панель "Действия", выберите "Кадр 1" на главной временной шкале и введите следующий код ActionScript.

```
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 Выберите "Управление" > "Тестировать ролик", чтобы запустить приложение.

Использование компонента TileList

Компонент TileList содержит список, который состоит из строк и столбцов и заполняется данными, полученными от поставщика данных. *Элементом* называется единица данных, которая хранится в ячейке компонента TileList. Элемент, полученный от поставщика данных, как правило, имеет свойства `label` и `source`. Свойство `label` определяет содержимое для отображения в ячейке, а свойство `source` передает его значение.

Можно создать экземпляр Array самостоятельно или получить его с сервера. Компонент TileList имеет методы для доступа к поставщику данных, например `addItem()` и `removeItem()`. Если у списке нет внешнего поставщика данных, эти методы автоматически создают экземпляр DataProvider через свойство `List.dataProvider`.

Взаимодействие пользователей с компонентом TileList

Экземпляр TileList визуализирует каждую ячейку с использованием объекта Sprite, внедряющего интерфейс ICellRenderer. Указать этот визуализатор можно с помощью свойства `cellRenderer` экземпляра TileList. Для компонента TileList значением свойства CellRenderer по умолчанию является объект ImageCell, который отображает изображение (класс, растровое изображение, экземпляр или URL-адрес) и дополнительную метку. Метка представляет собой одну строку, выровненную по нижнему краю ячейки. Прокручивать экземпляр TileList можно только в одном направлении.

Когда экземпляр TileList получает фокус, для доступа к его элементам можно использовать следующие клавиши.

Клавиша	Описание
"Стрелка вверх" и "Стрелка вниз"	Позволяет перемещаться вверх и вниз по столбцу. Если свойство `allowMultipleSelection` имеет значение `true`, эти клавиши можно использовать в сочетании с клавишей Shift, чтобы выделить несколько ячеек.
"Стрелка влево" и "Стрелка вправо"	Позволяют перемещаться влево или вправо по строке. Если свойство `allowMultipleSelection` имеет значение `true`, эти клавиши можно использовать в сочетании с клавишей Shift, чтобы выделить несколько ячеек.
Home	Выделяет первую ячейку в экземпляре TileList. Если свойство `allowMultipleSelection` имеет значение `true`, то удерживая клавишу Shift при нажатии Home, можно выделить все ячейки с текущей до первой.
End	Выделяет последнюю ячейку в экземпляре TileList. Если свойство `allowMultipleSelection` имеет значение `true`, то удерживая клавишу Shift при нажатии End, можно выделить все ячейки с текущей до последней.
Ctrl	Если свойство `allowMultipleSelection` имеет значение `true`, с помощью этой клавиши можно выбрать несколько непоследовательных ячеек.

Когда в приложение добавляется компонент TileList, его можно сделать доступным для программ чтения с экрана путем добавления следующих строк кода ActionScript.

```
import fl.accessibility.TileListAccImpl;

TileListAccImpl.enableAccessibility();
```

Расширенный доступ для компонента включается только один раз, независимо от числа его экземпляров. Дополнительные сведения см. в главе 18 "Создание содержимого с расширенной доступностью" в руководстве *Использование Flash*.

Параметры компонента TileList

В Инспекторе свойств или Инспекторе компонентов можно задать следующие параметры разработки для каждого экземпляра компонента TileList : `allowMultipleSelection`, `columnCount`, `columnWidth`, `dataProvider`, `direction`, `horizontalScrollLineSize`, `horizontalScrollPageSize`, `labels`, `rowCount`, `rowHeight`, `ScrollPolicy`, `verticalScrollLineSize` `verticalScrollPageSize`. Каждый из этих параметров имеет соответствующее свойство ActionScript с тем же именем. Сведения об использовании параметра dataProvider см. в разделе «Использование параметра dataProvider» на странице 31.

Можно написать код ActionScript, чтобы задать дополнительные параметры для экземпляров TileList с использованием их методов, свойств и событий. Дополнительные сведения см. в описании класса TileList в *Справочнике по языку ActionScript 3.0 и компонентам*.

Создание приложения с компонентом TileList

В этом примере используются объекты MovieClip для заполнения экземпляра TileList массивом цветов краски.

1. Создайте новый документ Flash (ActionScript 3.0).
2. Перетащите компонент TileList в рабочую область и присвойте ему имя экземпляра **aTl**.
3. Откройте панель "Действия", выберите "Кадр 1" на главной временной шкале и введите следующий код ActionScript.

```
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 Выберите "Управление" > "Тестировать ролик", чтобы проверить приложение.

Создание компонента TileList с помощью ActionScript

В этом примере динамически создается экземпляр TileList , и в него добавляются экземпляры компонентов ColorPicker, ComboBox, NumericStepper и CheckBox . Создается экземпляр Array с именем `dp`, содержащий метки и имена компонентов для отображения, и присваивается в качестве значения свойству `dataProvider` экземпляра TileList. Макет экземпляра TileList создается с помощью свойств `columnWidth` и `rowHeight` и метода `setSize()`, затем он размещается в рабочей области с помощью метода `move()`. Стиль `contentPadding` используется для добавления пробела между границами и содержимым экземпляра TileList, а метод `sortItemsOn()` сортирует содержимое по меткам.

1 Создайте новый документ Flash (ActionScript 3.0).

2 Перетащите следующие компоненты с панели "Компоненты" на панель "Библиотека": ColorPicker, ComboBox, NumericStepper, CheckBox и TileList.

3 Откройте панель "Действия", выберите "Кадр 1" на главной временной шкале и введите следующий код ActionScript.

```
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 Выберите "Управление" > "Тестировать ролик", чтобы проверить приложение.

Использование компонента UILoader

Компонент UILoader является контейнером, который может отображать файлы формата SWF, JPEG, прогрессивный JPEG, PNG и GIF. Компонент UILoader можно использовать, когда нужно получить содержимое из удаленного местоположения в приложение Flash. Например, UILoader можно использовать для добавления в форму логотипа компании (JPEG-файл). Компонент UILoader также можно использовать в приложении, отображающем фотографии. Используйте метод `load()` для загрузки содержимого, свойство `percentLoaded` для определения объема загруженного содержимого, и событие `complete` для определения времени завершения загрузки.

Можно масштабировать содержимое компонента UILoader или изменять размер его самого в соответствии с размерами содержимого. По умолчанию содержимое масштабируется в соответствии с размерами компонента UILoader. Также содержимое можно загружать во время выполнения и отслеживать прогресс загрузки (хотя после первой загрузки содержимое кэшируется, и строка прогресса быстро достигает 100 %). Если при загрузке содержимого в экземпляр UILoader указывается местоположение, необходимо указать точку с координатами (0, 0).

Взаимодействие пользователей с компонентом UILoader

Компонент UILoader не может получать фокус. Однако загруженное в него содержимое может получать фокус и реагировать на взаимодействие Дополнительные сведения об управлении фокусом см. в описании класса FocusManager в руководствах *Справочник по языку ActionScript 3.0 и компонентам* и «Работа с FocusManager» на странице 29.

Параметры компонента UILoader

В Инспекторе свойств или Инспекторе компонентов можно задать следующие параметры разработки для каждого экземпляра компонента UILoader : `autoLoad`, `maintainAspectRatio`, `source` и `scaleContent`. Каждый из этих параметров имеет соответствующее свойство ActionScript с тем же именем.

Интерактивный просмотр каждого элемента UILoader отражает изменения, которые были внесены в параметры в Инспекторе свойств или Инспекторе компонентов во время разработки.

Можно написать код ActionScript, чтобы задать дополнительные параметры для экземпляров UILoader с использованием их методов, свойств и событий. Дополнительные сведения см. в описании класса UILoader в *Справочнике по языку ActionScript 3.0 и компонентам*.

Создание приложения с компонентом UILoader

Ниже описывается процедура добавления компонента UILoader в приложение в ходе разработки. В этом примере экземпляр UILoader загружает изображение логотипа в формате GIF.

1. Создайте новый документ Flash (ActionScript 3.0).
2. Перетащите компонент UILoader с панели "Компоненты" в рабочую область.
3. В Инспекторе свойств введите **aUI** в качестве имени экземпляра.
4. Выделите UILoader в рабочей области и в Инспекторе компонентов и введите значение **http://www.helpexamples.com/images/logo.gif** для параметра `source`.

Создание экземпляра компонента UILoader с помощью ActionScript

В этом примере с помощью ActionScript создается компонент UILoader и загружает изображение цветка в формате JPEG. Когда происходит событие `complete`, на панели вывода отображается количество загруженных байт.

1. Создайте новый документ Flash (ActionScript 3.0).
2. Перетащите компонент UILoader с панели "Компоненты" на панель "Библиотека".
3. Откройте панель "Действия", выберите "Кадр 1" на главной временной шкале и введите следующий код ActionScript.

```
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 Выберите "Управление" > "Тестировать ролик".

Использование компонента UIScrollBar

Компонент UIScrollBar позволяет добавлять полосу прокрутки к текстовому полю. Полосу прокрутки можно добавить к текстовому полю в процессе разработки или во время выполнения с помощью ActionScript. Чтобы воспользоваться компонентом UIScrollBar, создайте текстовое поле в рабочей области, перетащите компонент UIScrollBar с панели "Компоненты" в любой квадрант ограничивающего прямоугольника текстового поля.

Если длина полосы прокрутки меньше общего размера стрелок прокрутки, компонент отображается неправильно. Одна кнопка со стрелкой скрывается под другой. Flash не выводит ошибки об этой проблеме. В таком случае лучше скрыть полосу прокрутки с помощью ActionScript. Если размера полосы прокрутки недостаточно для отображения ползунка, Flash скрывает его.

Компонент UIScrollBar выполняет те же функции, что и другие полосы прокрутки. Он содержит кнопки со стрелками по обоим концам, между которыми расположена полоса прокрутки и ползунок. Компонент можно расположить по любому краю текстового поля для прокручивания по вертикали и горизонтали.

Дополнительные сведения о TextField см. в описании класса TextField в *Справочнике по языку ActionScript 3.0 и компонентам*.

Взаимодействие пользователей с компонентом UIScrollBar

В отличие от многих других компонентов, UIScrollBar может получать непрерывный ввод данных мыши, например, когда пользователь удерживает кнопку мыши, не требуя повторных щелчков.

Компонент UIScrollBar не поддерживает взаимодействие с клавиатурой.

Параметры компонента UIScrollBar

В Инспекторе свойств или Инспекторе компонентов можно задать следующие параметры разработки для каждого экземпляра компонента UIScrollBar : `direction` и `scrollTargetName`. Каждый из этих параметров имеет соответствующее свойство ActionScript с тем же именем.

Можно написать код ActionScript, чтобы задать дополнительные параметры для экземпляров UIScrollBar с использованием их методов, свойств и событий. Дополнительные сведения см. в описании класса UIScrollBar в *Справочнике по языку ActionScript 3.0 и компонентам*.

Создание приложения с компонентом UIScrollBar

Ниже описывается процедура добавления компонента UIScrollBar в приложение в ходе разработки.

1 Создайте новый документ Flash (ActionScript 3.0).

2 Создайте динамическое текстовое поле, достаточно высокое для одной или двух строк текста, и присвойте ему имя экземпляра **myText** в Инспекторе свойств.

3 В Инспекторе свойств задайте для свойства LineType поля ввода текста значение Multiline или MultilineNoWrap, если планируется использовать горизонтальную полосу прокрутки.

4 Откройте панель "Действия", выберите "Кадр 1" на главной временной шкале и введите следующий код ActionScript для заполнения свойства `text` таким образом, чтобы пользователю потребовалась прокрутка для его просмотра.

```
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."
```

Примечание. *Убедитесь, что текстовое поле в рабочей области достаточно маленькое, чтобы весь текст нельзя было просмотреть без прокрутки. Если оно будет слишком большое, полоса прокрутки может не появиться или на ней не будет ползунка (для прокручивания содержимого).*

5 Убедитесь, что включена привязка к объектам ("Просмотр" > "Привязка"> "Привязка к объектам").

6 Перетащите экземпляр UIScrollBar с панели "Компоненты" в текстовое поле ввода рядом с тем краем, у которого нужно разместить полосу прокрутки. Чтобы компонент правильно присоединился к текстовому полю, кнопку мыши нужно отпустить, когда он будет перекрывать поле. Присвойте компоненту имя экземпляра **mySb.**

Свойство `scrollTargetName` автоматически заполняется именем экземпляра TextField, которое задано в Инспекторах свойств и компонентов. Если его нет на вкладке "Параметры", значит при размещении компонент UIScrollBar не достаточно перекрывал текстовое поле.

7 Выберите "Управление" > "Тестировать ролик".

Создание экземпляра компонента UIScrollBar с помощью ActionScript

Экземпляр UIScrollBar можно создать с помощью ActionScript и связать его с текстовым полем во время выполнения. В следующем примере создается горизонтальный экземпляр UIScrollBar и присоединяется к нижнему краю текстового поля **myTxt**, в которое загружается текст из URL. Кроме того, размер полосы прокрутки определяется по размеру текстового поля.

1 Создайте новый документ Flash (ActionScript 3.0).

2 Перетащите компонент ScrollBar на панель "Библиотека".

3 Откройте панель "Действия", выберите "Кадр 1" на главной временной шкале и введите следующий код ActionScript.

```
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 Выберите "Управление" > "Тестировать ролик".

Глава 5. Настройка компонентов пользовательского интерфейса

О настройке компонентов пользовательского интерфейса

Можно настроить внешний вид компонентов приложений, изменив один или оба следующих элемента.

Стили Каждый компонент имеет набор стилей, при помощи которого можно указать, какие значения будет использовать Flash при визуализации внешнего вида компонента. Как правило, стили определяют обложки и значки компонента для его различных состояний, а также тип используемого форматирования текста и значения значения для внутренних полей.

Обложки *Обложка* состоит из набора символов, которые составляют графический образ компонента в данном состоянии. Стиль задает обложку, которая является графическим элементом, используемым Flash при рисовании компонента. *Выбор обложки* — это процесс изменения внешнего вида компонента путем изменения или замены его графического образа.

Примечание. *Внешний вид компонентов ActionScript 3.0, заданный по умолчанию, можно рассматривать как тему (Aeon Halo), но эти обложки встроены в компоненты. В отличие от компонентов ActionScript 2.0 компоненты ActionScript 3.0 не поддерживают внешние файлы тем.*

Задание стилей

Как правило, стили компонента задают значения для его обложки, значков, форматирования текста и внутренних полей при рисовании приложением Flash компонента в его различных состояниях. Например, при рисовании во Flash компонента Button для его трех различных состояний используется три разные обложки: нажатого состояния (при нажатии на него кнопкой мыши), отжатого состояния и нормального состояния. Еще один вид обложки используется для неактивного состояния компонента, при котором свойство `enabled` установлено на значение `false`.

Стили компонентов можно задавать на уровне документа, класса и экземпляра. К тому же, некоторые свойства стиля могут наследоваться от родительского компонента. Например, компонент List наследует стили компонента ScrollBar, которые, в свою очередь, наследуются от компонента BaseScrollPane.

Задавать стили для настройки компонента можно следующим образом.

- Применить стиль к экземпляру компонента. Можно изменить свойства цвета и текста отдельного экземпляра компонента. В некоторых случаях это эффективно, но может занять много времени, если нужно задать отдельные свойства для всех компонентов в документе.
- Применить стиль ко всем компонентам определенного типа в документе. Если вы хотите придать единообразный внешний вид всем компонентам определенного типа, например всем компонентам CheckBox или Button в документе, можно применить стиль на уровне компонентов.

  Значения свойств стиля, заданные для контейнеров, наследуются содержащимися в них компонентами.

Flash не отображает изменения свойств стиля при просмотре компонентов в рабочей области с использованием функции "Интерактивный просмотр".

Особенности настройки параметров стиля

При использовании стилей следует учитывать несколько важных моментов.

Наследование Компоненты разработаны так, что дочерние компоненты наследуют стиль от родительских компонентов. В ActionScript нельзя задать наследование стилей.

Очередность Если стиль компонента задан несколькими способами, Flash использует первый найденный стиль согласно порядку очередности. Flash выполняет поиск стилей в следующем порядке, пока не будет найдено значение:

1. Flash выполняет поиск свойства стиля экземпляра компонента.
2. Если это один из наследуемых стилей, Flash выполняет поиск наследуемого значения в иерархии родительских объектов.
3. Flash выполняет поиск стиля компонента.
4. Flash выполняет поиск глобального параметра в StyleManager.
5. Если свойство все еще не определено, оно имеет значение `undefined`.

Доступ к стилям компонента по умолчанию

Доступ к стилям компонента по умолчанию можно получить, вызвав статический метод `getStyleDefinition()` для класса компонентов. Например, следующий код возвращает стили по умолчанию для компонента ComboBox и отображает значения по умолчанию свойств `buttonWidth` и `downArrowDownSkin`:

```
import fl.controls.ComboBox;
var styleObj:Object = ComboBox.getStyleDefinition();
trace(styleObj.buttonWidth); // 24
trace(styleObj.downArrowDownSkin); // ScrollArrowDown_downSkin
```

Задание и определение стилей экземпляра компонента

Любой экземпляр компонента пользовательского интерфейса может вызвать методы `setStyle()` и `getStyle()` напрямую для задания или определения стиля. Следующий синтаксис задает стиль и значение для экземпляра компонента:

```
instanceName.setStyle("styleName", value);
```

Такой синтаксис возвращает стиль экземпляра компонента:

```
var a_style:Object = new Object();
a_style = instanceName.getStyle("styleName");
```

Заметьте, что метод `getStyle()` возвращает значение типа Object, так как он может возвращать несколько стилей, имеющих различные типы данных. Например, следующий код задает начертание шрифта для экземпляра TextArea (`aTa`), а затем возвращает его при помощи метода `getStyle()`. Данный пример приводит возвращенное значение к объекту TextFormat для присвоения этого значения переменной TextFormat. В противном случае компилятор выдаст ошибку о попытке привести переменную Object к переменной TextFormat.

```
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);
```

Использование TextFormat для задания свойств текста

При помощи объекта TextFormat можно форматировать текст экземпляра компонента. Объект TextFormat имеет свойства, позволяющие задать такие параметры текста, как `bold`, `bullet`, `color`, `font`, `italic`, `size` и другие. Можно задать эти свойства в объекте TextFormat, затем вызвать метод `setStyle()`, чтобы применить их к экземпляру компонента. Например, следующий код задает свойства `font`, `size` и `bold` объекта TextFormat и применяет их к экземпляру Button:

```
/* 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);
```

Следующее изображение иллюстрирует эффект этих параметров кнопки с меткой "Отправить":

Свойства стиля, заданные для экземпляра компонента при помощи метода `setStyle()`, имеют высший приоритет и переопределяют все другие параметры стиля. Однако чем больше свойств вы задаете при помощи метода `setStyle()` для отдельного экземпляра компонента, тем медленнее будет происходить его визуализация при исполнении.

Задание стиля для всех экземпляров компонента

Можно задать стиль для всех экземпляров класса компонента при помощи статического метода `setComponentStyle()` класса StyleManager. Например, можно задать красный цвет текста для всех компонентов Button, сначала перетащив кнопку в рабочую область, затем вставив следующий код ActionScript на панель "Действия" в Кадр 1 временной шкалы:

```
import fl.managers.StyleManager;
import fl.controls.Button;

var tf:TextFormat = new TextFormat();
tf.color = 0xFF0000;
StyleManager.setComponentStyle(Button, "textFormat", tf);
```

Все компоненты Button, добавленные впоследствии в рабочую область, будут иметь красные метки.

Задание стиля для всех компонентов

Задать стиль для всех компонентов можно при помощи статического метода `setStyle()` класса StyleManager.

1 Перетащите компонент List в рабочую область и присвойте ему имя экземпляра **aList**.

2 Перетащите компонент Button в рабочую область и присвойте ему имя экземпляра **aButton**.

3 Нажмите клавишу **F9** или выберите пункт "Действия" в меню "Окно", чтобы открыть панель "Действия", если она еще не открыта, и введите следующий код в Кадр 1 временной шкалы, чтобы задать красный цвет текста для всех компонентов:

```
import fl.managers.StyleManager;

var tf:TextFormat = new TextFormat();
tf.color = 0xFF0000;
StyleManager.setStyle("textFormat", tf);
```

4 Вставьте следующий код на панель "Действия" для заполнения компонента List текстом.

```
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 Выберите меню "Управление" > "Тестировать ролик" или нажмите Ctrl+Enter для компиляции кода и тестирования содержимого. Текст метки кнопки и в списке должен быть красным.

Об обложках

Внешний вид компонента состоит из графических элементов, например контура, цвета заливки, значков и даже других компонентов. ComboBox, например, содержит компонент List, а компонент List содержит компонент ScrollBar. Сочетание графических элементов образует внешний вид компонента ComboBox. Однако внешний вид компонента изменяется в зависимости от его текущего состояния. Например, компонент CheckBox, без метки, выглядит примерно так в вашем приложении:

Компонент CheckBox в состоянии normal up

Если нажать кнопку мыши и удерживать ее нажатой на компоненте CheckBox, внешний вид этого компонента изменится следующим образом:

Компонент CheckBox в состоянии down

Когда вы отпустите кнопку мыши, компонент CheckBox приобретет свой первоначальный вид, но будет иметь метку выбора.

Компонент CheckBox в состоянии selected

В собирательном значении значки, представляющие компонент в его различных состояниях, называются его *обложками*. Можно изменить внешний вид компонента в одном или во всех его состояниях путем редактирования его обложек во Flash так же, как и любой другой символ Flash. Доступ к обложкам компонента можно получить двумя способами. Самое простое — это перетащить компонент в рабочую область и дважды щелкнуть его мышью. При этом откроется палитра обложек компонента, которая в случае с компонентом CheckBox выглядит следующим образом.

Обложки компонента CheckBox

Доступ к отдельным обложкам компонента можно также получить из панели "Библиотека". При перетаскивании компонента в рабочую область он копируется в библиотеку вместе с папкой его активов и всеми компонентами, которые он содержит. Например, при перетаскивании компонента ComboBox в рабочую область панель "Библиотека" также будет содержать компоненты List, ScrollBar и TextInput, встроенные в поле со списком, а также папку с обложками для каждого из этих компонентов и папку "Общие активы", содержащую общие для компонентов элементы. Можно изменить обложку любого из этих компонентов, открыв папку с его обложками (ComboBoxSkins, ListSkins, ScrollBarSkins или TextInputSkins) и дважды щелкнув значок обложки, которую нужно изменить. При двойном щелчке на ComboBox_downSkin, например, открывается обложка в режиме редактирования символов, как показано на рисунке ниже:

ComboBox_downSkin

Создание новой обложки

Для изменения внешнего вида компонента в документе необходимо отредактировать его обложки. Для доступа к обложкам компонента просто дважды щелкните компонент в рабочей области, чтобы открыть палитру его обложек. Затем дважды щелкните обложку, которую нужно отредактировать, чтобы открыть ее в режиме редактирования символов. Например, дважды щелкните компонент TextArea в рабочей области, чтобы открыть его активы в режиме редактирования символов. Установите масштаб на 400 % или выше и отредактируйте символ, чтобы изменить его внешний вид. Внесенные изменения отразятся на всех экземплярах компонента в документе. Вместо этого можно дважды щелкнуть конкретную обложку на панели "Библиотека", чтобы открыть ее в рабочей области в режиме редактирования символов.

Обложки компонента можно изменять следующим образом:

- Создать новую обложку для всех экземпляров.
- Создать новые обложки для отдельных экземпляров.

Создание обложки для всех экземпляров

При редактировании обложки компонента по умолчанию изменяется внешний вид всех экземпляров этого компонента в документе. Если необходимо создать различные обложки для одного и того же компонента, необходимо продублировать обложки, которые нужно изменить, и присвоить им разные имена, затем отредактировать их и задать соответствующие стили для их применения. Дополнительную информацию см. в разделе «Создание обложек для отдельных экземпляров» на странице 108.

В данной главе рассматривается процедура изменения одной или нескольких обложек каждого из компонентов пользовательского интерфейса. Выполнив одну из этих процедур по изменению одной или нескольких обложек компонента пользовательского интерфейса, вы измените обложки для всех экземпляров в документе.

Создание обложек для отдельных экземпляров

При помощи следующей процедуры можно создать обложку для отдельных экземпляров компонента:

- Выберите обложку в папке активов компонента на панели "Библиотека".
- Продублируйте обложку и присвойте ей уникальное имя класса.
- Отредактируйте обложку для придания ей необходимого внешнего вида.
- Вызовите метод `setStyle()` для экземпляра компонента, чтобы назначить новую обложку стилю обложки.

Следующая процедура создает новую обложку selectedDownSkin для одного из двух экземпляров Button.

1. Создайте новый файл Flash (ActionScript 3.0).
2. Перетащите две кнопки с панели "Компоненты" в рабочую область и присвойте им имена экземпляров **aButton** и **bButton**.
3. Откройте папки "Component Assets" и "ButtonSkins" на панели "Библиотека".
4. Щелкните мышью обложку selectedDownSkin, чтобы выделить ее.
5. Щелкните ее правой кнопкой мыши, чтобы открыть контекстное меню, и выберите пункт "Дублировать".
6. В диалоговом окне "Дублировать символ" присвойте новой обложке уникальное имя, например **Button_mySelectedDownSkin**. Затем нажмите кнопку "ОК".
7. В папке "Библиотека" > "Component Assets" > "ButtonSkins" выберите обложку Button_mySelectedDownSkin и щелкните ее правой кнопкой мыши, чтобы открыть контекстное меню. Нажмите "Связывание", чтобы открыть диалоговое окно "Свойства связывания".
8. Установите флажок "Экспорт для ActionScript". Установите флажок "Экспортировать в первый кадр" и убедитесь, что имя класса уникальное. Нажмите кнопку "ОК", затем нажмите "ОК" еще раз в ответ на предупреждение, что определение класса не найдено и будет создано.
9. Дважды щелкните обложку Button_mySelectedDownSkin на панели "Библиотека", чтобы открыть ее в режиме редактирования символов.
10. Щелкните голубой заполнитель в центре обложки, чтобы цвет появился в палитре цветов "Заливка" в Инспекторе свойств. Щелкните палитру цветов и выберите цвет #00CC00 для заливки обложки.
11. Нажмите кнопку "Назад" в левой части панели правки над рабочей областью, чтобы вернуться в режим редактирования документа.

12 В Инспекторе свойств выберите вкладку "Параметры" для каждой кнопки и установите параметр toggle на значение `true`.

13 Вставьте следующий код в Кадр 1 временной шкалы на панели "Действия":

```
bButton.setStyle("selectedDownSkin", Button_mySelectedDownSkin);
bButton.setStyle("downSkin", Button_mySelectedDownSkin);
```

14 Выберите "Управление" > "Тестировать ролик".

15 Щелкните мышью каждую кнопку. Заметьте, что для обложки нажатого состояния (выбранной или невыбранной кнопки) объекта bButton используется новый символ.

Настройка компонента Button

Компонент Button можно изменять по горизонтали и по вертикали как во время разработки, так и при исполнении. При разработке выделите компонент в рабочей области и используйте инструмент "Свободное преобразование" или одну из команд "Модификация" > "Преобразовать". При исполнении используйте метод `setSize()` или любые применимые свойства класса Button, например `height` и `width`, `scaleX` и `scaleY`.

Изменение размера кнопки не влияет на размер значка или метки. Ограничивающий прямоугольник кнопки совпадает с ее рамкой и определяет область попадания экземпляра. При увеличении размера экземпляра увеличивается и размер области попадания. Если метка не помещается в ограничивающий прямоугольник, она обрезается.

Если кнопка имеет значок, который больше самой кнопки, этот значок выходит за границы рамки кнопки.

Использование стилей для компонента Button

Как правило, стили компонента Button задают значения для его обложек, значков, форматирования текста и внутренних полей при рисовании компонента в его различных состояниях.

Следующая процедура помещает два компонента Button в рабочую область и устанавливает свойство `emphasized` обоих компонентов на значение `true` в случае нажатия на один из них пользователем. При этом также стиль `emphasizedSkin` второго компонента Button устанавливается на значение `selectedOverSkin` в случае нажатия на него пользователем. Таким образом, два компонента Button демонстрируют различные обложки для одного и того же состояния.

1 Создайте файл Flash (ActionScript 3.0).

2 Перетащите две кнопки, одну за другой, в рабочую область и присвойте им имена экземпляров **aBtn** и **bBtn**. На вкладке "Параметры" Инспектора свойств присвойте им метки "Кнопка А" и "Кнопка Б".

3 Вставьте следующий код в Кадр 1 временной шкалы на панели "Действия":

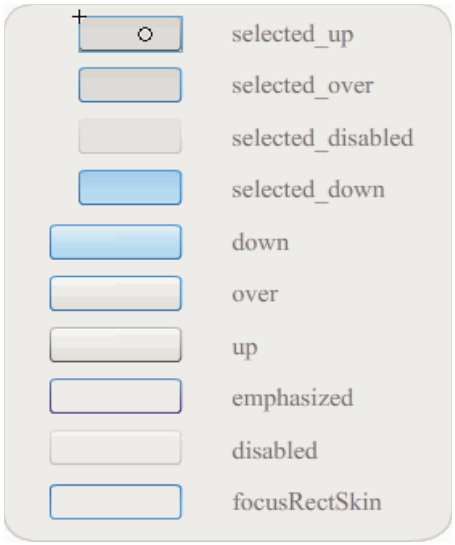
```
bBtn.emphasized = true;
aBtn.emphasized = true;
bBtn.addEventListener(MouseEvent.CLICK, Btn_handler);
function Btn_handler(evt:MouseEvent):void {
    bBtn.setStyle("emphasizedSkin", "Button_selectedOverSkin");
}
```

4 Выберите "Управление" > "Тестировать ролик".

5 Щелкните одну из кнопок, чтобы увидеть эффект применения стиля `emphasizedSkin` к каждой кнопке.

Использование обложек для компонента Button

Компонент Button имеет следующие обложки, соответствующие его различным состояниям. Для редактирования обложек в целях изменения внешнего вида кнопки дважды щелкните экземпляр кнопки в рабочей области, чтобы открыть палитру его обложек, как показано на рисунке ниже:

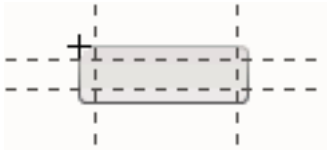


Обложки компонента Button

Если кнопка активна, она отображает состояние over при наведении на нее указателя мыши. Кнопка получает фокус ввода и отображает состояние down при ее нажатии. Кнопка возвращается в состояние over при отпускании кнопки мыши. Если отвести указатель с кнопки при нажатой кнопке мыши, кнопка вернется в исходное состояние. Если параметр toggle установлен на значение `true`, для отображения нажатого состояния используется обложка selectedDownSkin, для отжатого состояния — selectedUpSkin, а для состояния "наведение указателя" — selectedOverSkin.

Если кнопка неактивна, она отображает неактивное состояние disabled вне зависимости от действий пользователя.

Для редактирования одной из обложек дважды щелкните ее, чтобы открыть в режиме редактирования символов, как показано на рисунке ниже:



Компонент Button в режиме редактирования символов

На данном этапе вы можете использовать инструменты разработки Flash для редактирования обложки по своему желанию.

Следующая процедура изменяет цвет обложки selected_over компонента Button.

1. Создайте новый файл Flash (ActionScript 3.0).
2. Перетащите кнопку с панели "Компоненты" в рабочую область. На вкладке "Параметры" установите значение параметра toggle на значение `true`.

3 Дважды щелкните кнопку, чтобы открыть палитру ее обложек.

4 Дважды щелкните обложку selected_over, чтобы открыть ее в режиме редактирования символов.

5 Установите масштаб на 400 %, чтобы увеличить значок для редактирования.

6 Дважды щелкните фон, чтобы его цвет появился в палитре "Заливка" в Инспекторе свойств.

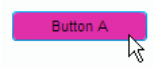
7 В палитре "Заливка" выберите цвет #CC0099 для фона обложки selected_over.

8 Нажмите кнопку "Назад" в левой части панели правки над рабочей областью, чтобы вернуться в режим редактирования документа.

9 Выберите "Управление" > "Тестировать ролик".

10 Нажмите кнопку, чтобы она приняла выбранное состояние.

При наведении указателя мыши на кнопку состояние selected_over должно отображаться так, как показано на рисунке ниже.



Компонент Button, демонстрирующий обложку selected_over с измененным цветом

Настройка компонента CheckBox

Компонент CheckBox можно изменять по горизонтали и по вертикали как во время разработки, так и при исполнении. При разработке выделите компонент в рабочей области и используйте инструмент "Свободное преобразование" или одну из команд "Модификация" > "Преобразовать". При исполнении используйте метод `setSize()` или применимые свойства класса CheckBox. Например, можно изменить размер флажка, задав его свойства `height` и `width`, `scaleX` и `scaleY`. Изменение размера компонента CheckBox не влияет на размер метки или значка; при этом изменяется только размер ограничивающего прямоугольника.

Ограничивающий прямоугольник экземпляра компонента CheckBox невидимый и определяет область попадания экземпляра. При увеличении размера экземпляра увеличивается и размер области попадания. Если метка не помещается в ограничивающий прямоугольник, она обрезается.

Использование стилей для компонента CheckBox

Для изменения внешнего вида экземпляра CheckBox можно задать свойства стиля. Например, следующая процедура изменяет размер и цвет метки флажка.

1 Перетащите компонент CheckBox с панели "Компоненты" в рабочую область и присвойте ему имя экземпляра **myCb.**

2 В Инспекторе свойств выберите вкладку "Параметры" и введите следующее значение для параметра label: **Меньше 500 долларов?**

3 В Кадр 1 основной временной шкалы на панели "Действия" введите следующий код:

```
var myTf:TextFormat = new TextFormat();
myCb.setSize(150, 22);
myTf.size = 16;
myTf.color = 0xFF0000;
myCb.setStyle("textFormat", myTf);
```

Дополнительную информацию см. в разделе «Задание стилей» на странице 103. Информацию о задании свойств стиля в целях изменения значков и обложек компонента см. в разделах «Создание новой обложки» на странице 107 и «Использование обложек для компонента CheckBox» на странице 112.

Использование обложек для компонента CheckBox

Компонент CheckBox имеет следующие обложки, которые можно редактировать в целях изменения внешнего вида компонента.

Обложки компонента CheckBox

Данный пример изменяет цвет контура и фона компонента в состояниях `up` и `selectedUp`. Те же шаги необходимо выполнить и для изменения обложек других состояний.

1. Создайте новый файл Flash (ActionScript 3.0).
2. Перетащите компонент CheckBox в рабочую область. При этом он также появится в библиотеке с папкой его активов.
3. Дважды щелкните компонент CheckBox в рабочей области, чтобы открыть панель его значков обложки.
4. Дважды щелкните значок selected_up, чтобы открыть его в режиме редактирования символов.
5. Установите масштаб на 800 %, чтобы увеличить значок для редактирования.
6. Щелкните рамку флажка, чтобы выделить ее. При помощи палитры "Заливка" в Инспекторе свойств выберите цвет #0033FF и примените его к рамке.
7. Дважды щелкните фон флажка, чтобы выделить его, и снова при помощи палитры "Заливка" задайте цвет фона #00CCFF.
8. Повторите шаги 4-8 для обложки up компонента CheckBox.
9. Выберите "Управление" > "Тестировать ролик".

Настройка компонента ColorPicker

Изменить размер компонента ColorPicker можно только через его стили: `swatchWidth`, `swatchHeight`, `backgroundPadding`, `textFieldWidth` и `textFieldHeight`. При попытке изменить размер палитры при помощи инструмента преобразования или при помощи ActionScript, используя метод `setSize()`, или через свойства `width`, `height`, `scaleX` или `scaleY` эти значения игнорируются при создании SWF-файла, и компонент ColorPicker отображается с размером по умолчанию. Размер фона палитры изменится в соответствии с количеством столбцов, заданным при помощи метода `setStyle()` для стиля `columnCount`. Количество столбцов по умолчанию равно 18. Можно задать 1024 пользовательских цвета, и палитра изменит свой размер по вертикали в соответствии с количеством образцов.

Использование стилей для компонента ColorPicker

Для изменения внешнего вида компонента ColorPicker можно задать несколько стилей. Например, следующая процедура изменяет количество столбцов (`columnCount`) в палитре на 12, изменяет высоту (`swatchHeight`) и ширину (`swatchWidth`) образцов цвета, а также изменяет внутренние поля текстового поля (`textPadding`) и фона (`backgroundPadding`).

1. Создайте новый файл Flash (ActionScript 3.0).
2. Перетащите компонент ColorPicker в рабочую область и присвойте ему имя экземпляра **aCp**.
3. Откройте панель "Действия", выберите Кадр 1 на основной временной шкале и введите следующий код:

```
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. Выберите "Управление" > "Тестировать ролик".
5. Откройте компонент ColorPicker, щелкнув его мышью, чтобы увидеть, как эти параметры изменили его внешний вид.

Использование обложек для компонента ColorPicker

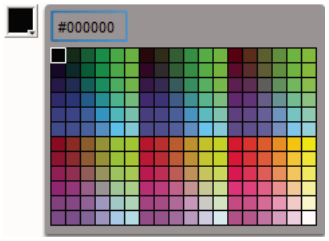
Компонент ColorPicker использует следующие обложки для визуализации своих состояний.

Обложки компонента ColorPicker

Можно изменить цвет обложки Background для изменения цвета фона палитры.

1. Создайте новый файл Flash (ActionScript 3.0).
2. Перетащите компонент ColorPicker в рабочую область.
3. Дважды щелкните компонент, чтобы открыть палитру его обложек.
4. Дважды щелкните обложку Background, чтобы выбрать ее, после чего в Инспекторе свойств появится палитра "Заливка".
5. В палитре "Заливка" выберите цвет #999999, чтобы применить его к обложке Background.
6. Нажмите кнопку "Назад" в левой части панели правки над рабочей областью, чтобы вернуться в режим редактирования документа.
7. Выберите "Управление" > "Тестировать ролик".

   Если щелкнуть палитру, ее фон должен быть серым, как на рисунке ниже.



Компонент ColorPicker с темно-серой обложкой Background

Настройка компонента ComboBox

Компонент ComboBox можно изменять по горизонтали и по вертикали как во время разработки, так и при исполнении. При разработке выделите компонент в рабочей области и используйте инструмент "Свободное преобразование" или одну из команд "Модификация" > "Преобразовать". При исполнении используйте метод `setSize()` или любые применимые свойства класса ComboBox, например `height` и `width`, `scaleX` и `scaleY`.

Размер компонента ComboBox изменится в соответствии с указанной шириной и высотой. Размер списка изменится в соответствии с шириной компонента, если не задано свойство `dropdownWidth`.

Если текст не помещается по длине в поле со списком, он будет обрезан. Необходимо изменить размер поля со списком и задать свойство `dropdownWidth`, чтобы уместился текст.

Использование стилей для компонента ComboBox

Для изменения внешнего вида компонента ComboBox можно задать свойства стиля. Стили задают значения для обложек, средства отображения ячеек, внутренних полей и ширины кнопок компонента. Следующий пример задает стили `buttonWidth` и `textPadding`. Стиль `buttonWidth` задает ширину области попадания кнопки и действует, если компонент ComboBox является редактируемым и вы можете только нажать кнопку, чтобы развернуть раскрывающийся список. Стиль textPadding задает промежуток между внешней границей текстового поля и текстом. Это пригодится при центрировании текста по вертикали в текстовом поле, когда размер компонента ComboBox увеличивается по вертикали. В противном случае текст может отображаться вверху текстового поля.

1 Создайте новый файл Flash (ActionScript 3.0).

2 Перетащите компонент ComboBox в рабочую область и присвойте ему имя экземпляра **aCb**.

3 Откройте панель "Действия", выберите Кадр 1 на основной временной шкале и введите следующий код:

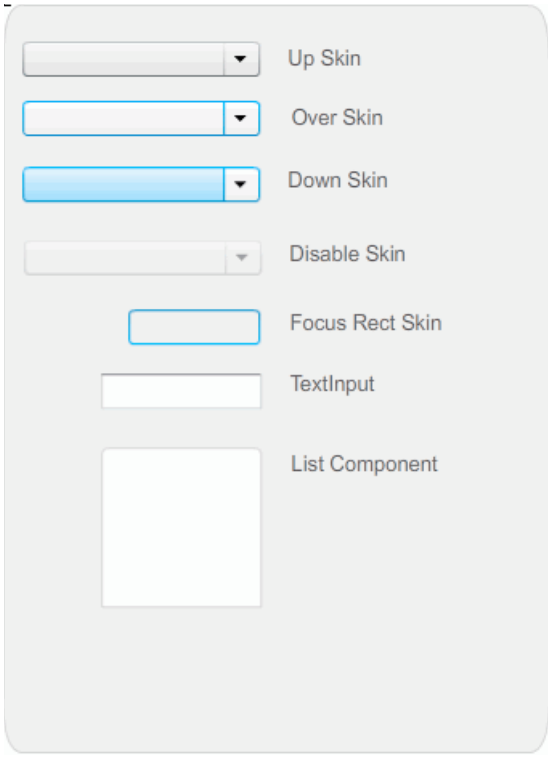
```
import fl.data.DataProvider;

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 Выберите "Управление" > "Тестировать ролик".

Заметьте, что областью кнопки, которую вы можете нажать, чтобы открыть раскрывающийся список, является ограниченная область справа. Также обратите внимание на то, что текст отцентрирован по вертикали в текстовом поле. Можно попробовать выполнить пример без двух выражений `setStyle()`, чтобы увидеть эффект.

Использование обложек для компонента ComboBox

Компонент ComboBox использует следующие обложки для визуализации своих состояний.



Обложки компонента ComboBox

Можно изменить цвет обложки Up в рабочей области для изменения цвета компонента в неактивном состоянии.

1. Создайте новый файл Flash (ActionScript 3.0).
2. Перетащите компонент ComboBox в рабочую область.
3. Дважды щелкните компонент, чтобы открыть палитру его обложек.
4. Дважды щелкните обложку Up, чтобы выбрать ее и открыть для редактирования.
5. Установите масштаб на 400 %.
6. Щелкните мышью в центре обложки, чтобы ее цвет появился в палитре "Заливка" в Инспекторе свойств.
7. В палитре "Заливка" выберите цвет #33FF99 для применения к обложке Up.
8. Нажмите кнопку "Назад" в левой части панели правки над рабочей областью, чтобы вернуться в режим редактирования документа.
9. Выберите "Управление" > "Тестировать ролик".

   Компонент ComboBox должен отображаться в рабочей области так, как показано на рисунке ниже.

Компонент ComboBox с настроенным цветом обложки Background

Настройка компонента DataGrid

Компонент DataGrid можно изменять по горизонтали и по вертикали как во время разработки, так и при исполнении. При разработке выделите компонент в рабочей области и используйте инструмент "Свободное преобразование" или одну из команд "Модификация" > "Преобразовать". При исполнении используйте метод `setSize()` или применимые свойства, например `width`, `height`, `scaleX` и `scaleY`. При отсутствии горизонтальной полосы прокрутки ширина столбцов устанавливается пропорционально. При настройке размера столбцов (а следовательно и ячеек) текст в ячейках может быть обрезан.

Использование стилей для компонента DataGrid

Для изменения внешнего вида компонента DataGrid можно задать свойства стиля. Компонент DataGrid наследует стили от компонента List. (См. раздел «Использование стилей для компонента List» на странице 122.)

Задание стиля для отдельного столбца

Объект DataGrid может иметь несколько столбцов, для каждого из которых можно указать разные визуализаторы. Каждый столбец объекта DataGrid представлен объектом DataGridColumn, а класс DataGridColumn включает свойство `cellRenderer`, для которого можно определить объект CellRenderer столбца.

1. Создайте новый документ Flash (ActionScript 3.0).
2. Перетащите компонент DataGrid на панель "Библиотека".
3. Вставьте следующий код в Кадр 1 временной шкалы на панели "Действия". Данный код создает сетку данных с длинной текстовой строкой в третьем столбце. В конце он устанавливает свойство `cellRenderer` столбца на имя визуализатора, который отображает многострочную ячейку.

```
/* 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 Сохраните FLA-файл как MultiLineGrid.fla.

5 Создайте новый файл ActionScript.

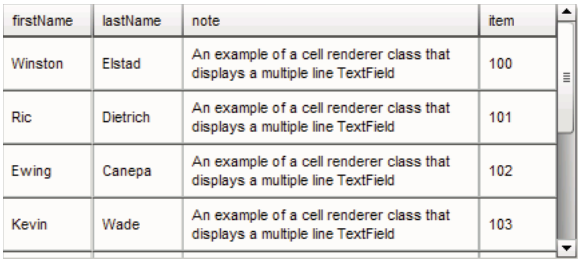
6 Скопируйте следующий код ActionScript в окно "Сценарий":

```
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 Сохраните файл ActionScript как MultiLineCell.as в той же папке, в которой вы сохранили файл MultiLineGrid.fla.

8 Вернитесь в приложение MultiLineGrid.fla и выберите меню "Управление" > "Тестировать ролик".

Компонент DataGrid должен выглядеть следующим образом:



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

DataGrid для приложения MultiLineGrid.fla

Задание стиля заголовка

Можно задать стиль текста строки заголовка, используя стиль `headerTextFormat`. В следующем примере используется объект TextFormat для задания стиля `headerTextFormat` с использованием шрифта Arial, красного цвета, размера шрифта 14 и курсивного начертания.

1 Создайте новый файл Flash (ActionScript 3.0).

2 Перетащите компонент DataGrid в рабочую область и присвойте ему имя экземпляра **aDg**.

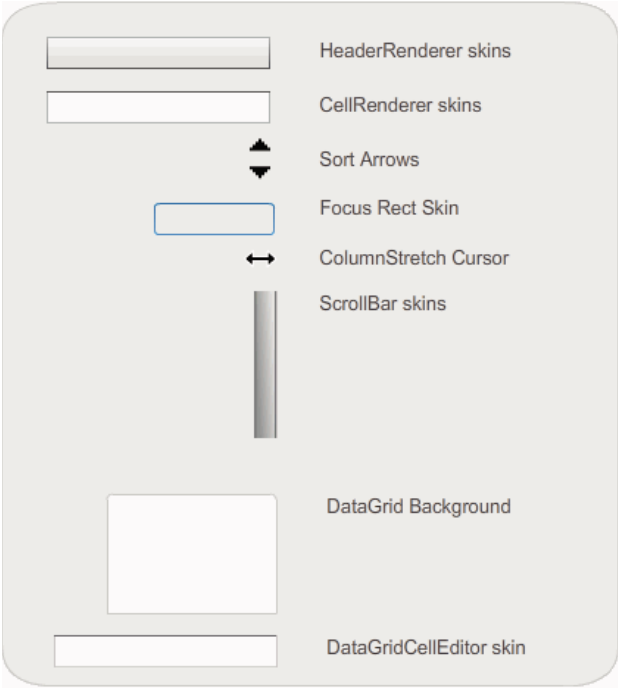
3 Откройте панель "Действия", выберите Кадр 1 на основной временной шкале и введите следующий код:

```
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 Выберите "Управление" > "Тестировать ролик" для запуска приложения.

Использование обложек для компонента DataGrid

Компонент DataGrid использует следующие обложки для визуализации своих состояний.



Обложки компонента DataGrid

Обложка CellRenderer используется для ячеек компонента DataGrid, тогда как обложка HeaderRenderer используется для строки заголовка. Следующая процедура изменяет цвет фона строки заголовка, но она же может использоваться и для изменения цвета фона ячеек сетки данных путем редактирования обложки CellRenderer.

1. Создайте новый документ Flash (ActionScript 3.0).
2. Перетащите компонент DataGrid в рабочую область и присвойте ему имя экземпляра **aDg**.
3. Дважды щелкните компонент, чтобы открыть палитру его обложек.
4. Установите масштаб на 400 %, чтобы увеличить значки для редактирования.
5. Дважды щелкните обложку HeaderRenderer, чтобы открыть палитру обложек HeaderRenderer.
6. Дважды щелкните обложку Up_Skin, чтобы открыть ее в режиме редактирования символов, и щелкните ее фон, чтобы выбрать его. При этом откроется палитра "Заливка" в Инспекторе свойств.
7. В палитре "Заливка" выберите цвет #00CC00 для применения его к фону обложки Up_Skin HeaderRenderer.
8. Нажмите кнопку "Назад" в левой части панели правки над рабочей областью, чтобы вернуться в режим редактирования документа.
9. Вставьте следующий код в Кадр 1 временной шкалы на панели "Действия", чтобы добавить данные в сетку данных.

```
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 Выберите "Управление" > "Тестировать ролик" для тестирования приложения.

Сетка данных должна отобразиться так, как показано на следующем рисунке, с зеленым фоном строки заголовка.

Name	Home
Wilma Carter	Redlands, CA
Sue Pennypacker	Athens, GA
Jill Smithfield	Spokane, WA
Shirley Goth	Carson, NV
Jennifer Dunbar	Seaside, CA

Компонент DataGrid с настроенным фоном строки заголовка

Настройка компонента Label

Компонент Label можно изменять по горизонтали и по вертикали как во время разработки, так и при исполнении. При разработке выделите компонент в рабочей области и используйте инструмент "Свободное преобразование" или одну из команд "Модификация" > "Преобразовать". Можно также задать параметр `autoSize`; задание этого параметра не изменяет ограничивающий прямоугольник в режиме интерактивного просмотра, но размер метки изменяется. Размер компонента Label изменяется в зависимости от значения параметра `wordwrap`. Если параметр установлен на значение `true`, изменяется размер метки по вертикали, чтобы уместился текст. Если параметр установлен на значение `false`, изменяется размер метки по горизонтали. При исполнении используйте метод `setSize()`. Дополнительную информацию см. в описании метода `Label.setSize()` и свойства `Label.autoSize` в документе *Справочник по языку ActionScript 3.0 и компонентам*. Также см. раздел «Создание приложения с компонентом Label» на странице 65.

Использование стилей для компонента Label

Для изменения внешнего вида экземпляра Label можно задать свойства стиля. Стиль должен быть общим для всего текста экземпляра компонента Label. Компонент Label имеет стиль `textFormat`, который имеет те же атрибуты, что и объект TextFormat, и позволяет задать те же свойства для содержимого `Label.text`, как для обычного текстового поля Flash. Следующий пример задает красный цвет текста метки.

1 Перетащите компонент Label с панели "Компоненты" в рабочую область и присвойте ему имя экземпляра `a_label`.

2 Выберите вкладку "Параметры" и замените значение свойства text следующим текстом:

Color me red

3 Выберите Кадр 1 основной временной шкалы, откройте панель "Действия" и введите следующий код:

```
/* 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 Выберите "Управление" > "Тестировать ролик".

Дополнительную информацию о стилях компонента Label см. в описании класса Label в документе *Справочник по языку ActionScript 3.0 и компонентам*.

Обложки и компонент Label

Компонент Label не имеет визуальных элементов, для которых можно выбрать обложку.

Настройка компонента List

Компонент List можно изменять по горизонтали и по вертикали как во время разработки, так и при исполнении. При разработке выделите компонент в рабочей области и используйте инструмент "Свободное преобразование" или одну из команд "Модификация" > "Преобразовать". При исполнении используйте метод `setSize()` и применимые свойства класса List, например `height` и `width`, `scaleX` и `scaleY`.

При изменении размера списка строки списка уменьшаются по горизонтали, обрезая содержащийся в них текст. При изменении по вертикали строки добавляются в список или удаляются из него. Полосы прокрутки размещаются автоматически.

Использование стилей для компонента List

Для изменения внешнего вида компонента List можно задать свойства стиля. Стили задают значения для обложек и внутренних полей компонента при его отрисовке.

Различные стили обложки позволяют задавать различные классы для использования в обложках. Дополнительную информацию об использовании стилей обложки см. в разделе «Об обложках» на странице 106.

Следующая процедура задает значение стиля `contentPadding` для компонента List. Заметьте, что значение данного параметра вычитается из значения размера компонента List для образования внутреннего поля вокруг содержимого, поэтому может потребоваться увеличить размер списка, чтобы предотвратить обрезку текста в нем.

1 Создайте новый файл Flash (ActionScript 3.0).

2 Перетащите компонент List с панели "Компоненты" в рабочую область и присвойте ему имя экземпляра **aList**.

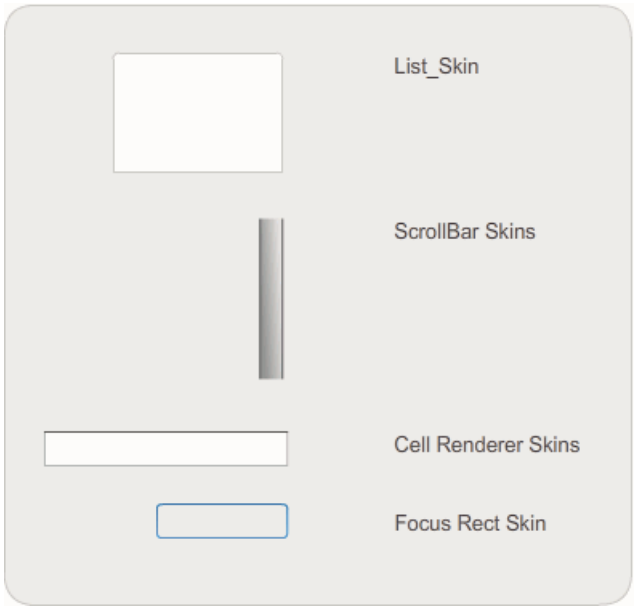
3 Выберите Кадр 1 основной временной шкалы, откройте панель "Действия" и введите следующий код, который задает стиль `contentPadding` и вставляет данные в список:

```
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 Выберите "Управление" > "Тестировать ролик".

Использование обложек для компонента List

Компонент List использует следующие обложки для визуализации своих состояний.

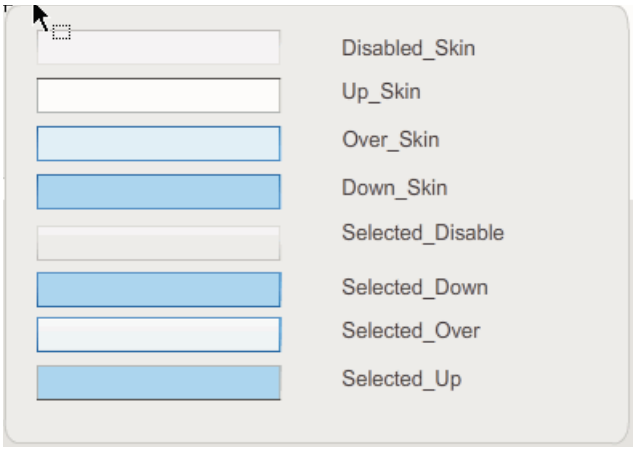


Обложки компонента List

Дополнительную информацию о выборе обложки для компонента ScrollBar см. в разделе «Настройка компонента UIScrollBar» на странице 138. Информацию о настройке обложки Focus Rect см. в разделе «Настройка компонента TextArea» на странице 132

Примечание. *Изменение обложки ScrollBar в одном компоненте приведет к ее изменению во всех других компонентах, использующих компонент ScrollBar.*

Дважды щелкните обложку Cell Renderer, чтобы открыть вторую палитру обложек для различных состояний ячейки компонента List.



Обложки Cell Renderer компонента List

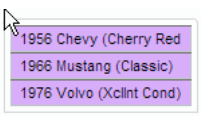
Можно изменить внешний вид ячеек списка, отредактировав эти обложки. Следующая процедура изменяет цвет обложки Up для изменения внешнего вида списка в его нормальном неактивном состоянии.

1. Создайте новый файл Flash (ActionScript 3.0).
2. Перетащите компонент List с панели "Компоненты" в рабочую область и присвойте ему имя экземпляра **aList**.
3. Дважды щелкните компонент List, чтобы открыть палитру его обложек.
4. Дважды щелкните обложку Cell Renderer, чтобы открыть палитру обложек средства отображения ячеек.
5. Дважды щелкните обложку Up_Skin, чтобы открыть ее для редактирования.
6. Щелкните область заливки обложки, чтобы выбрать ее. При этом должна появиться палитра "Заливка" в Инспекторе свойств с текущим цветом заливки обложки.
7. В палитре "Заливка" выберите цвет #CC66FF, чтобы применить его к заливке обложки Up_Skin.
8. Нажмите кнопку "Назад" в левой части панели правки над рабочей областью, чтобы вернуться в режим редактирования документа.
9. Вставьте следующий код в Кадр 1 временной шкалы на панели "Действия", чтобы добавить данные в 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. Выберите "Управление" > "Тестировать ролик".

    Компонент List должен отображаться так, как показано на рисунке ниже:



Ячейки списка с пользовательским цветом обложки Up_Skin

Рамка получилась в результате задания стиля `contentPadding`.

Настройка компонента NumericStepper

Компонент NumericStepper можно изменять по горизонтали и по вертикали как во время разработки, так и при исполнении. При разработке выделите компонент в рабочей области и используйте инструмент "Свободное преобразование" или одну из команд "Модификация" > "Преобразовать". При исполнении используйте метод `setSize()` или любые применимые свойства и методы класса NumericStepper, например `width`, `height`, `scaleX` и `scaleY`.

Изменение размера компонента NumericStepper не влияет на ширину кнопок со стрелками вверх и вниз. Если высота компонента меняется на большую, чем высота по умолчанию, кнопки со стрелками закрепляются сверху и снизу компонента по умолчанию. В противном случае отрисовка кнопок определяется 9-зонным масштабированием. Кнопки со стрелками всегда расположены справа от текстового окна.

Стили и компонент NumericStepper

Для изменения внешнего вида компонента NumericStepper можно задать для него свойства стиля. Стили задают значения для обложек, внутренних полей и формата текста компонента при его отрисовке. Стиль `textFormat` позволяет изменять размер и внешний вид значения, отображаемого компонентом NumericStepper. Различные стили обложки позволяют задавать различные классы для использования в обложках. Дополнительную информацию об использовании стилей обложки см. в разделе «Об обложках» на странице 106.

В данной процедуре используется стиль `textFormat` для изменения внешнего вида значения, отображаемого компонентом NumericStepper.

1. Создайте новый документ Flash (ActionScript 3.0).
2. Перетащите компонент NumericStepper с панели "Компоненты" в рабочую область и присвойте ему имя экземпляра **myNs**.
3. Добавьте следующий код на панель "Действия" в кадре 1 основной временной шкалы:

```
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. Выберите "Управление" > "Тестировать ролик".

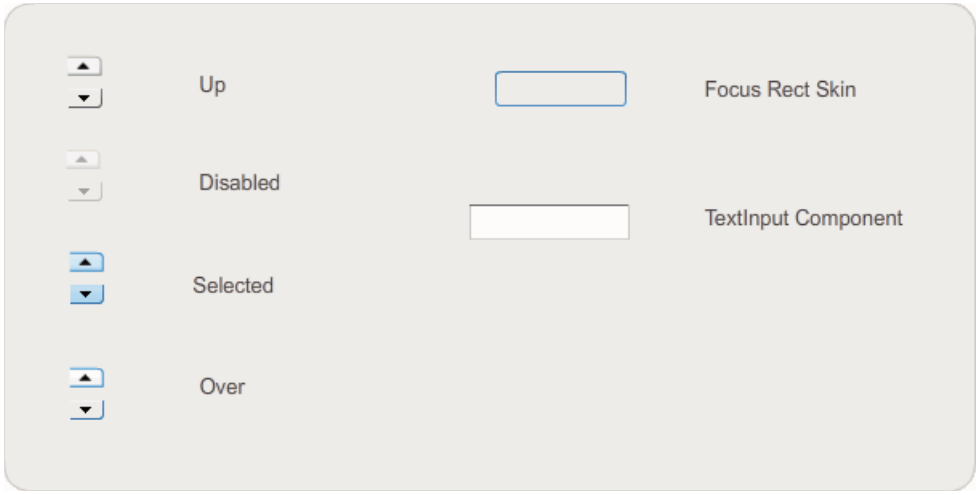
Обложки и компонент NumericStepper

Компонент NumericStepper имеет обложки для отображения нормального активного, нажатого, неактивного и выбранного состояний его кнопок.

Если компонент активен, кнопки для перехода вверх и вниз отображаются в состоянии over. При нажатии кнопки отображаются в состоянии down. При отпускании кнопки мыши кнопки возвращаются в состояние over. Если отвести указатель с кнопок при нажатой кнопке мыши, кнопки вернутся в исходное состояние.

Если компонент неактивен, он отображает состояние disabled независимо от действий пользователя.

Компонент NumericStepper имеет следующие обложки:



Обложки компонента NumericStepper

1. Создайте новый FLA-файл.
2. Перетащите компонент NumericStepper в рабочую область.
3. Установите масштаб на 400 %, чтобы увеличить изображение для редактирования.
4. Дважды щелкните фон обложки компонента TextInput на панели обложек, чтобы перейти вниз по иерархии на уровень "Группа". При этом цвет фона появится в палитре "Заливка" в Инспекторе свойств.
5. В палитре "Заливка" в Инспекторе свойств выберите цвет #9999FF, чтобы применить его к фону обложки компонента TextInput.
6. Нажмите кнопку "Назад" в левой части панели правки над рабочей областью, чтобы вернуться в режим редактирования документа.
7. Снова дважды щелкните компонент NumericStepper, чтобы открыть палитру его обложек.
8. Дважды щелкните фон кнопки со стрелкой вверх в группе Up, чтобы выбрать фон. При этом цвет фона появится в палитре "Заливка" в Инспекторе свойств.
9. Выберите цвет #9966FF, чтобы применить его к фону кнопки со стрелкой вверх.
10. Повторите шаги 8 и 9 для стрелки вниз в группе Up.
11. Выберите "Управление" > "Тестировать ролик".

    Экземпляр NumericStepper должен отображаться так, как показано на рисунке ниже:

Настройка компонента ProgressBar

Компонент ProgressBar можно изменять по горизонтали и по вертикали как во время разработки, так и при исполнении. При разработке выделите компонент в рабочей области и используйте инструмент "Свободное преобразование" или одну из команд "Модификация" > "Преобразовать". При исполнении используйте метод `setSize()` или соответствующие свойства класса ProgressBar, например `height`, `width`, `scaleX` и `scaleY`.

Компонент ProgressBar имеет три обложки: для дорожки, полосы заполнения и неопределенного режима. Для масштабирования активов используется 9-зонное масштабирование.

Стили и компонент ProgressBar

Для изменения внешнего вида экземпляра ProgressBar можно задать свойства стиля. Стили задают значения для обложек и внутренних полей компонента ProgressBar при его отрисовке. Следующий пример увеличивает размер экземпляра компонента ProgressBar и задает его стиль barPadding.

1 Создайте новый FLA-файл.

2 Перетащите компонент ProgressBar с панели "Компоненты" в рабочую область и присвойте ему имя экземпляра **myPb.**

3 В Кадр 1 основной временной шкалы на панели "Действия" введите следующий код:

```
myPb.width = 300;
myPb.height = 30;

myPb.setStyle("barPadding", 3);
```

4 Выберите "Управление" > "Тестировать ролик".

Дополнительную информацию о задании стилей обложки см. в разделе «Об обложках» на странице 106.

Обложки и компонент ProgressBar

Компонент ProgressBar имеет обложки для представления дорожки индикатора, заполненной полосы и полосы неопределенного режима, как показано на рисунке ниже.

Обложки компонента ProgressBar

Полоса заполнения накладывается на дорожку, используя параметр barPadding для определения положения. Активы масштабируются с использованием 9-зонного масштабирования.

Полоса неопределенного режима используется, если свойство `indeterminate` экземпляра ProgressBar установлено на значение `true`. Эта обложка может изменяться по вертикали и горизонтали в соответствии с размером индикатора выполнения.

Можно редактировать эти обложки в целях изменения внешнего вида индикатора выполнения. Следующий пример изменяет цвет полосы неопределенного режима.

1 Создайте новый FLA-файл.

2 Перетащите компонент ProgressBar в рабочую область и дважды щелкните его, чтобы открыть панель значков обложек.

3 Дважды щелкните обложку полосы неопределенного режима.

4 Установите масштаб на 400 %, чтобы увеличить значок для редактирования.

5 Дважды щелкните одну из диагональных полос, затем, удерживая клавишу Shift, щелкните все остальные. Текущий цвет появится в палитре "Заливка" в Инспекторе свойств.

6 Откройте щелчком мыши палитру "Заливка" в Инспекторе свойств и выберите цвет #00CC00, чтобы применить его к выбранным диагональным полосам.

7 Нажмите кнопку "Назад" в левой части панели правки над рабочей областью, чтобы вернуться в режим редактирования документа.

8 Выберите "Управление" > "Тестировать ролик".

Индикатор выполнения должен выглядеть так, как показано на рисунке ниже.

Настройка компонента RadioButton

Компонент RadioButton можно изменять по горизонтали и по вертикали как во время разработки, так и при исполнении. При разработке выделите компонент в рабочей области и используйте инструмент "Свободное преобразование" или одну из команд "Модификация" > "Преобразовать". При исполнении используйте метод `setSize()`.

Ограничивающий прямоугольник компонента RadioButton невидимый и определяет область попадания компонента. При увеличении размера компонента увеличивается и размер области попадания.

Если метка компонента не помещается в ограничивающий прямоугольник, она обрезается.

Использование стилей для компонента RadioButton

Для изменения внешнего вида компонента RadioButton можно задать свойства стиля. Свойства стиля компонента RadioButton задают значения для его обложек, значков, форматирования текста и внутренних полей при рисовании этого компонента. Стили компонента RadioButton задают значения для обложек и внутренних полей макета при рисовании компонента.

Следующий пример возвращает стиль `textFormat` компонента CheckBox и применяет его к компоненту RadioButton, чтобы стиль их меток был одинаковым.

1 Создайте новый документ Flash (ActionScript 3.0).

2 Перетащите компонент CheckBox в рабочую область и присвойте ему имя экземпляра **myCh** в Инспекторе свойств.

3 Перетащите компонент RadioButton в рабочую область и присвойте ему имя экземпляра **myRb** в Инспекторе свойств.

4 Вставьте следующий код в Кадр 1 временной шкалы на панели "Действия".

```
var tf:TextFormat = new TextFormat();
tf.color = 0x00FF00;
tf.font = "Georgia";
tf.size = 18;
myCh.setStyle("textFormat", tf);
myRb.setStyle("textFormat", myCh.getStyle("textFormat"));
```

Данный код задает стиль `textFormat` для флажка, затем применяет его к переключателю путем вызова метода `getStyle()` для флажка.

5 Выберите "Управление" > "Тестировать ролик".

Обложки и компонент RadioButton

Компонент RadioButton имеет следующие обложки, которые можно редактировать в целях изменения внешнего вида компонента.

Обложки компонента RadioButton

Если компонент RadioButton активен и не выбран, он отображает обложку over. Когда пользователь щелкает компонент RadioButton, он получает фокус ввода и отображает обложку selected_down. Когда пользователь отпускает кнопку мыши, компонент RadioButton отображает обложку selected_up. Если пользователь выводит указатель за пределы области попадания компонента RadioButton при нажатой кнопке мыши, компонент RadioButton снова отображается с обложкой up.

Если компонент RadioButton неактивен, он отображает состояние disabled вне зависимости от действий пользователя.

Следующий пример удаляет обложку selected_up, которая указывает на состояние selected.

1 Создайте новый документ Flash (ActionScript 3.0).

2 Перетащите компонент RadioButton в рабочую область и дважды щелкните его, чтобы открыть его палитру обложек.

3 Установите масштаб на 800 %, чтобы увеличить значок для редактирования.

4 Дважды щелкните обложку selected_up, чтобы выделить ее, и нажмите клавишу Delete для ее удаления.

5 Выберите инструмент "Прямоугольник" на панели "Инструменты".

6 В Инспекторе свойств задайте красный цвет линии (#FF0000) и черный цвет заливки (#000000).

7 В месте перекрестия, которое обозначает точку регистрации символа (также известную как *исходная точка* или *нулевая точка*), нажмите кнопку мыши и тащите указатель, чтобы нарисовать прямоугольник.

8 Нажмите кнопку "Назад" в левой части панели правки над рабочей областью, чтобы вернуться в режим редактирования документа.

9 Выберите "Управление" > "Тестировать ролик".

10 Щелкните мышью компонент RadioButton, чтобы выбрать его.

В состоянии selected компонент RadioButton должен выглядеть, как на рисунке ниже.

Настройка компонента ScrollPane

Компонент ScrollPane можно изменять по горизонтали и по вертикали как во время разработки, так и при исполнении. При разработке выделите компонент в рабочей области и используйте инструмент "Свободное преобразование" или одну из команд "Модификация" > "Преобразовать". При исполнении используйте метод `setSize()` или любые применимые свойства и методы класса ScrollPane, например `height`, `width`, `scaleX` и `scaleY`.

Компонент ScrollPane имеет следующие графические параметры:

- Точка регистрации (также известная как *исходная точка* или *нулевая точка*) содержимого находится в верхнем левом углу панели.
- Если горизонтальная полоса прокрутки выключена, вертикальная полоса прокрутки отображается сверху вниз вдоль правого края панели прокрутки. Если вертикальная полоса прокрутки выключена, горизонтальная полоса прокрутки отображается слева направо вдоль нижнего края панели прокрутки. Можно отключить обе полосы прокрутки.
- Если панель прокрутки слишком маленькая, содержимое может отображаться некорректно.
- При изменении размера полосы прокрутки сама дорожка и бегунок также увеличиваются или уменьшаются. При этом изменяется и размер области попадания. Кнопки сохраняют свой размер.

Использование стилей для компонента ScrollPane

Свойства стиля компонента ScrollPane задают значения для обложек и внутренних полей макета при отрисовке компонента. Различные стили обложки позволяют задавать различные классы для использования в обложках компонента. Дополнительную информацию об использовании стилей обложки см. в разделе «Об обложках» на странице 106.

1 Создайте новый документ Flash (ActionScript 3.0).

2 Перетащите компонент ScrollPane в рабочую область и присвойте ему имя экземпляра **mySp**.

3 Выберите вкладку "Параметры" в Инспекторе свойств и введите для параметра `source` следующее значение: **http://www.helpexamples.com/flash/images/image1.jpg**.

4 В Кадр 1 основной временной шкалы на панели "Действия" введите следующий код.

```
mySp.setStyle("contentPadding", 5);
```

Заметьте, что внутренние поля находятся между рамкой компонента и его содержимым, т. е. вне полос прокрутки.

5 Выберите "Управление" > "Тестировать ролик".

Обложки и ScrollPane

Активами компонента ScrollPane являются рамка и полосы прокрутки. Информацию о выборе обложки для полос прокрутки см. в разделе «Использование обложек для компонента UIScrollBar» на странице 138.

Настройка компонента Slider

Компонент Slider можно изменять по горизонтали как во время разработки, так и при исполнении. При разработке выделите компонент в рабочей области и используйте инструмент "Свободное преобразование" или одну из команд "Модификация" > "Преобразовать". При исполнении используйте метод `setSize()` или любые применимые свойства класса Slider, например `width` и `scaleX`.

Можно лишь увеличить длину ползунка. Увеличить его высоту невозможно. Flash игнорирует свойство `height` и параметр height метода `setSize()`. Однако можно создать вертикальный ползунок и увеличить его размер по вертикали.

Стили и компонент Slider

Стили компонента Slider задают только классы для его обложек и значение `FocusRectPadding`, которое указывает количество пикселов для внутренних полей между ограничивающим прямоугольником компонента и его внешней границей. Дополнительную информацию об использовании стилей обложек см. в разделе «Об обложках» на странице 106.

Обложки и компонент Slider

Компонент Slider имеет следующие обложки, которые можно редактировать в целях изменения внешнего вида компонента.

Обложки компонента Slider

Следующий пример изменяет цвет дорожки движения вверх на синий.

1. Создайте новый документ Flash (ActionScript 3.0).
2. Перетащите компонент Slider с панели "Компоненты" в рабочую область.
3. Дважды щелкните компонент Slider, чтобы открыть палитру его обложек.

4 Дважды щелкните точку регистрации дорожки, чтобы открыть ее в режиме редактирования символов.

5 Установите масштаб на 800 %, чтобы увеличить значок для редактирования. Заметьте, что дорожка ползунка состоит из трех полос.

6 Выберите верхнюю полосу, щелкнув ее мышью. При этом ее цвет появится в палитре "Заливка" в Инспекторе свойств.

7 В палитре "Заливка" в Инспекторе свойств выберите цвет #000066, чтобы применить его к верхней полосе дорожки ползунка.

8 Щелкните среднюю полосу дорожки ползунка, чтобы выбрать ее. При этом ее цвет появится в палитре "Заливка" в Инспекторе свойств.

9 В палитре "Заливка" в Инспекторе свойств выберите цвет #0066FF, чтобы применить его к средней полосе дорожки ползунка.

10 Щелкните нижнюю полосу дорожки ползунка, чтобы выбрать ее. При этом ее цвет появится в палитре "Заливка" в Инспекторе свойств.

11 В палитре "Заливка" в Инспекторе свойств выберите цвет #00CCFF, чтобы применить его к нижней полосе дорожки ползунка.

12 Нажмите кнопку "Назад" в левой части панели правки над рабочей областью, чтобы вернуться в режим редактирования документа.

13 Выберите "Управление" > "Тестировать ролик".

Компонент Slider должен отображаться так, как показано на рисунке ниже.

Настройка компонента TextArea

Компонент TextArea можно изменять по горизонтали и по вертикали как во время разработки, так и при исполнении. При разработке выделите компонент в рабочей области и используйте инструмент "Свободное преобразование" или одну из команд "Модификация" > "Преобразовать". При исполнении используйте метод `setSize()` или любые применимые свойства, например `height`, `width`, `scaleX` и `scaleY` класса TextArea.

При изменении размера компонента TextArea размер его рамки изменяется в соответствии с новым ограничивающим прямоугольником. Если требуются полосы прокрутки, они размещаются вдоль нижнего и правого краев. Размер области текста тогда изменяется в пределах оставшегося пространства; компонент TextArea не содержит элементов фиксированного размера. Если ширины компонента TextArea не хватает для отображения всего текста, текст обрезается.

Стили и компонент TextArea

Стили компонента TextArea задают значения для его обложек, внутренних полей и формата текста при отрисовке компонента. Стили `texFormat` и `disabledTextFormat` определяют стиль текста, отображаемого компонентом TextArea. Дополнительную информацию о свойствах стиля обложек см. в разделе «Использование обложек для компонента TextArea» на странице 133.

Следующий пример задает стиль `disabledTextFormat` для изменения внешнего вида текста в неактивном компоненте TextArea, но та же процедура может использоваться для задания стиля `textFormat` для активного компонента TextArea.

1 Создайте новый файл Flash.

2 Перетащите компонент TextArea в рабочую область и присвойте ему имя экземпляра **myTa**.

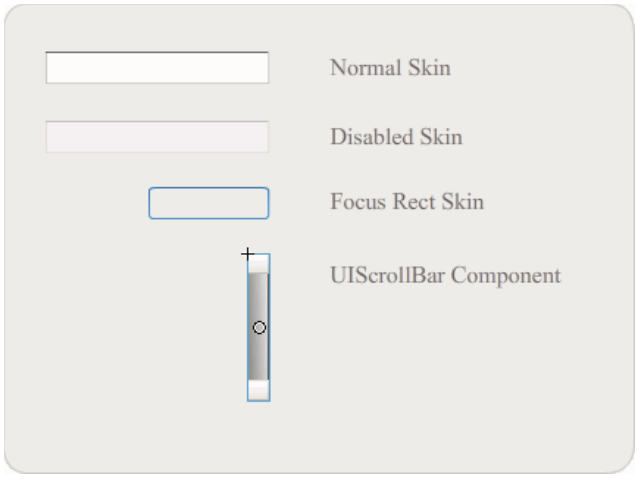
3 Вставьте следующий код в Кадр 1 основной временной шкалы на панели "Действия".

```
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 Выберите "Управление" > "Тестировать ролик".

Использование обложек для компонента TextArea

Компонент TextArea имеет следующие обложки, которые можно редактировать в целях изменения внешнего вида компонента.



Обложки компонента TextArea

Примечание. *Изменение обложки ScrollBar в одном компоненте приведет к ее изменению во всех других компонентах, использующих компонент ScrollBar.*

Следующая процедура изменяет цвет рамки обложки Focus Rect, которая отображается при получении компонентом TextArea фокуса, и цвет рамки обложки Normal.

1 Создайте новый файл Flash.

2 Перетащите компонент TextArea в рабочую область и дважды щелкните его, чтобы открыть панель значков обложек.

3 Дважды щелкните обложку Focus Rect.

4 Щелкните рамку обложки Focus Rect, чтобы выбрать ее. При этом текущий цвет рамки появится в палитре "Заливка" в Инспекторе свойств.

5 Откройте щелчком мыши палитру "Заливка" в Инспекторе свойств и выберите цвет #CC0000, чтобы применить его к рамке.

6 Нажмите кнопку "Назад" в левой части панели правки над рабочей областью, чтобы вернуться в режим редактирования документа.

7 Дважды щелкните компонент TextArea, чтобы открыть панель значков обложки.

8 Дважды щелкните обложку Normal.

9 Выделите один за другим каждый край рамки обложки Normal и задайте для них цвет #990099.

10 Нажмите кнопку "Назад" в левой части панели правки над рабочей областью, чтобы вернуться в режим редактирования документа.

11 Выберите "Управление" > "Тестировать ролик".

При выборе области текста для ввода текста ее рамка должна выглядеть, как на рисунке ниже:

Внешняя рамка — это обложка Focus Rect, а внутренняя рамка — это рамка обложки Normal.

Информацию о редактировании обложки UIScrollBar см. в разделе «Настройка компонента UIScrollBar» на странице 138.

Настройка компонента TextInput

Можно изменить размер экземпляра TextInput как во время разработки, так и при исполнении. При разработке выделите компонент в рабочей области и используйте инструмент "Свободное преобразование" или одну из команд "Модификация" > "Преобразовать". При исполнении используйте метод `setSize()` или применимые свойства класса TextInput, например `height` и `width`, `scaleX` и `scaleY`.

При изменении размера компонента TextInput размер его рамки изменяется в соответствии с новым ограничивающим прямоугольником. Компонент TextInput не использует полос прокрутки, но курсор вставки прокручивается автоматически при взаимодействии пользователя с текстом. Размер текстового поля тогда изменяется в пределах оставшегося пространства; компонент TextInput не содержит элементов фиксированного размера. Если размер компонента TextInput не позволяет отобразить текст полностью, текст обрезается.

Стили и компонент TextInput

Стили компонента TextInput задают значения для его обложек, внутренних полей и форматирования текста при отрисовке компонента. Стили `texFormat` и `disabledTextFormat` определяют стиль текста, отображаемого компонентом. Дополнительную информацию о свойствах стиля обложек см. в разделе «Обложки и компонент TextInput» на странице 135.

Следующий пример задает стиль `textFormat` для шрифта, размера и цвета текста, отображаемого компонентом TextInput. Ту же процедуру можно использовать при задании стиля `disabledTextFormat`, который применяется при неактивном состоянии компонента.

1 Создайте новый документ Flash (ActionScript 3.0).

2 Перетащите компонент TextInput в рабочую область и присвойте ему имя экземпляра **myTi**.

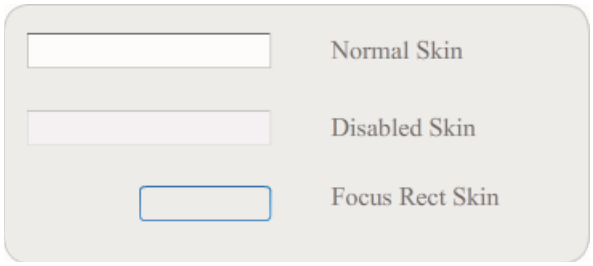
3 Вставьте следующий код в Кадр 1 основной временной шкалы на панели "Действия".

```
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 Выберите "Управление" > "Тестировать ролик".

Обложки и компонент TextInput

Компонент TextInput имеет следующие обложки, которые можно редактировать в целях изменения внешнего вида компонента:



Обложки компонента TextInput

Следующая процедура меняет цвета рамки и фона компонента TextInput:

1 Создайте новый файл Flash.

2 Перетащите компонент TextInput в рабочую область и дважды щелкните его, чтобы открыть панель обложек.

3 Дважды щелкните обложку Normal.

4 Установите масштаб на 800 %, чтобы увеличить значок для редактирования.

5 Выделите один за другим каждый край рамки обложки Normal и задайте для них цвет #993399.

6 Дважды щелкните фон, чтобы его цвет появился в палитре "Заливка" в Инспекторе свойств. Выберите цвет #99CCCC, чтобы применить его к фону.

7 Нажмите кнопку "Назад" в левой части панели правки над рабочей областью, чтобы вернуться в режим редактирования документа.

8 Выберите "Управление" > "Тестировать ролик".

Компонент TextInput должен выглядеть так, как показано на рисунке ниже:

Настройка компонента TileList

Компонент TileList можно изменять по горизонтали и по вертикали как во время разработки, так и при исполнении. При разработке выделите компонент в рабочей области и используйте инструмент "Свободное преобразование" или одну из команд "Модификация" > "Преобразовать". При исполнении используйте метод `setSize()` или соответствующие свойства, например `width`, `height`, `columnCount`, `rowCount`, `scaleX` и `scaleY`. Масштаб компонента ScrollBar, содержащегося в компоненте TileList, изменяется с полем списка.

Стили и компонент TileList

Стили компонента TileList задают значения для его обложек, внутренних полей и форматирования текста при отрисовке компонента. Стили `texFormat` и `disabledTextFormat` определяют стиль текста, отображаемого компонентом. Дополнительную информацию о стилях обложек см. в разделе «Использование обложек для компонента TileList» на странице 136.

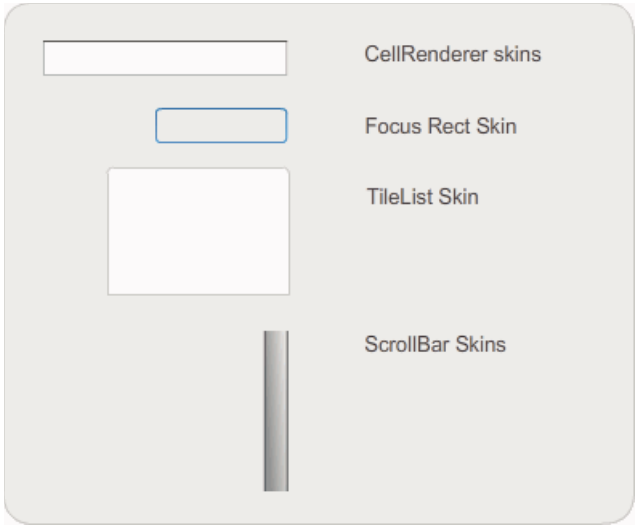
Следующий пример вызывает метод `setRendererStyle()` с использованием стиля `textFormat` для задания шрифта, размера, цвета и атрибутов текста меток, отображаемых в экземпляре компонента TileList. Ту же процедуру можно использовать для задания стиля `disabledTextFormat`, который применяется, если свойство `enabled` установлено на значение `false`.

1. Создайте новый документ Flash (ActionScript 3.0).
2. Перетащите компонент TileList в рабочую область и присвойте ему имя экземпляра **myTl**.
3. Вставьте следующий код в Кадр 1 временной шкалы на панели "Действия".

```
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);
```

Использование обложек для компонента TileList

Компонент TileList имеет обложки TileList, CellRenderer и ScrollBar. Можно редактировать эти обложки в целях изменения внешнего вида компонента TileList:



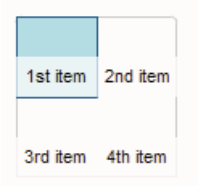
Обложки компонента TileList

Примечание. *Изменение обложки ScrollBar в одном компоненте приведет к ее изменению во всех других компонентах, использующих компонент ScrollBar.*

Следующая процедура изменяет цвет обложки CellRenderer Selected_Up компонента TileList.

1. Создайте документ Flash (ActionScript 3.0).
2. Перетащите компонент TileList в рабочую область и дважды щелкните его, чтобы открыть панель обложек.
3. Дважды щелкните обложку CellRenderer, затем дважды щелкните обложку Selected_Up, а затем щелкните прямоугольник фона.
4. В палитре "Заливка" в Инспекторе свойств выберите цвет #99FFFF, чтобы применить его к обложке Selected_Up.
5. Нажмите кнопку "Назад" в левой части панели правки над рабочей областью, чтобы вернуться в режим редактирования документа.
6. На вкладке Инспектора свойств "Параметры" дважды щелкните второй столбец строки dataProvider, чтобы открыть диалоговое окно "Значения". Вставьте элементы со следующими метками: 1-й элемент, 2-й элемент, 3-й элемент, 4-й элемент.
7. Выберите "Управление" > "Тестировать ролик".
8. Щелкните одну из ячеек компонента TileList, чтобы выделить ее, затем отведите указатель мыши от выделенной ячейки.

   Выделенная ячейка должна выглядеть так, как показано на рисунке ниже:



Компонент TileList с измененным цветом обложки Selected_Up

Настройка компонента UILoader

Компонент UILoader можно изменять по горизонтали и по вертикали как во время разработки, так и при исполнении. При разработке выделите компонент в рабочей области и используйте инструмент "Свободное преобразование" или одну из команд "Модификация" > "Преобразовать". При исполнении используйте метод `setSize()` или соответствующие свойства, например `width`, `height`, `scaleX` и `scaleY`.

Изменением размера компонента UILoader управляет свойство `scaleContent`. Если свойство `scaleContent` имеет значение `true`, содержимое масштабируется в соответствии с границами загрузчика (и повторно масштабируется при вызове метода `setSize()`). Если свойство `scaleContent` имеет значение `false`, размер компонента подгоняется под размер содержимого, а метод `setSize()` и свойства размера не действуют.

Компонент UILoader не имеет элементов пользовательского интерфейса, к которым можно применить обложки или стили.

Настройка компонента UIScrollBar

Компонент UIScrollBar можно изменять по горизонтали и по вертикали как во время разработки, так и при исполнении. Однако невозможно изменить ширину вертикального компонента UIScrollBar и высоту горизонтального компонента UIScrollBar. При разработке выделите компонент в рабочей области и используйте инструмент "Свободное преобразование" или одну из команд "Модификация" > "Преобразовать". При исполнении используйте метод `setSize()` или любые применимые свойства класса UIScrollBar, например `width`, `height`, `scaleX` и `scaleY`.

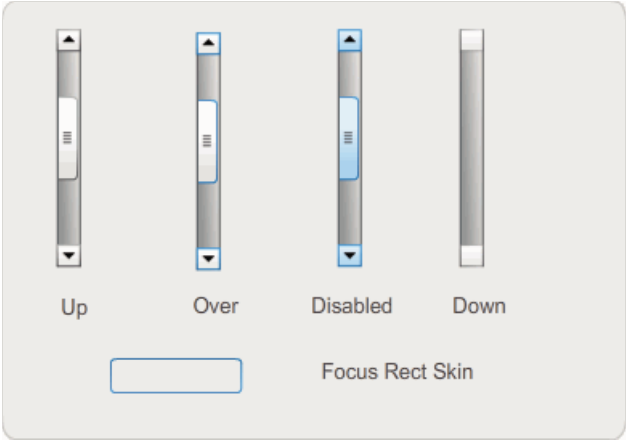
Примечание. *При использовании метода `setSize()` можно изменить только ширину горизонтальной полосы прокрутки или высоту вертикальной полосы прокрутки. Во время разработки можно задать высоту горизонтальной или ширину вертикальной полосы прокрутки, но эти значения будут сброшены при публикации ролика. Можно изменить только тот размер полосы прокрутки, который соответствует ее длине.*

Использование стилей для компонента UIScrollBar

Стили компонента UIScrollBar задают только классы для его обложек и значение FocusRectPadding, которое указывает количество пикселов для внутренних полей между ограничивающим прямоугольником компонента и его внешней границей. Дополнительную информацию об использовании стилей обложек см. в разделе «Об обложках» на странице 106.

Использование обложек для компонента UIScrollBar

Компонент UIScrollBar имеет следующие обложки.



Обложки компонента UIScrollBar

И горизонтальная, и вертикальная полосы прокрутки используют одинаковые обложки; при отображении горизонтальной полосы прокрутки компонент UIScrollBar поворачивает обложку как нужно.

Примечание. *Изменение обложки ScrollBar в одном компоненте приведет к ее изменению во всех других компонентах, использующих компонент ScrollBar.*

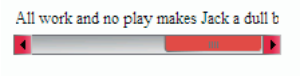
Следующий пример показывает, как изменить цвет бегунка и кнопок со стрелками компонента UIScrollBar.

1. Создайте новый документ Flash (ActionScript 3.0).
2. Перетащите компонент UIScrollBar в рабочую область и присвойте ему имя экземпляра **mySb**. На вкладке "Параметры" задайте горизонтальное расположение.
3. Дважды щелкните полосу прокрутки, чтобы открыть ее панель обложек.
4. Щелкните мышью обложку Up, чтобы выделить ее.
5. Установите масштаб на 400 %, чтобы увеличить значок для редактирования.
6. Дважды щелкните фон правой стрелки (или верхней стрелки вертикальной полосы прокрутки), чтобы его выбрать, и цвет фона появится в палитре "Заливка" в Инспекторе свойств.
7. Выберите цвет #CC0033, чтобы применить его к фону кнопки.
8. Нажмите кнопку "Назад" в левой части панели правки над рабочей областью, чтобы вернуться в режим редактирования документа.
9. Повторите шаги 6, 7 и 8 для бегунка и левой стрелки (или нижней стрелки вертикальной полосы прокрутки).
10. Вставьте следующий код в Кадр 1 временной шкалы на панели "Действия", чтобы добавить полосу прокрутки в текстовое поле.

```
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 Выберите "Управление" > "Тестировать ролик".

Компонент UIScrollBar должен отображаться так, как показано на рисунке ниже.



Горизонтальная полоса прокрутки со стрелками и бегунком красного цвета

Глава 6. Использование компонента FLVPlayback

Компонент FLVPlayback позволяет включить видеопроигрыватель в приложение Adobe Flash CS4 Professional для воспроизведения последовательно загружаемых видеофайлов с использованием протокола HTTP, или для воспроизведения потоковых видеофайлов с сервера Adobe Macromedia Flash Media Server или Flash Video Streaming Service (FVSS).

Выпуск Adobe Flash Player 9 Update 3 (версии 9.0.115.0 или более новой) включает в себя значительные улучшения воспроизведения видеосодержимого в проигрывателе Flash Player. Эти обновления включают в себя изменения компонента FLVPlayback, которые эффективно используют видеоустройства конечного пользователя, обеспечивая лучшее воспроизведение видеосодержимого. Изменения компонента FLVPlayback также повышают надежность видеофайлов, воспроизводимых в полноэкранном режиме.

Более того, обновления Flash Player 9 Update 3 улучшают функциональность компонента FLVPlayback за счет поддержки видеоформатов MPEG-4 высокого разрешения, в которых используется промышленный стандарт кодирования H.264. Эти форматы включают в себя MP4, M4A, MOV, MP4V, 3GP и 3G2.

Примечание. *Защищенные файлы MP4, например те, которые загружаются с сервера Apple® iTunes®, или файлы с цифровым шифрованием FairPlay®, не поддерживаются.*

Легкий в использовании компонент FLVPlayback имеет следующие свойства и преимущества:

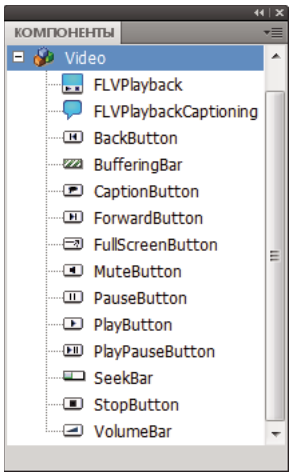
- Его можно перетащить в рабочую область, быстро и успешно реализовать.
- Поддерживает полноэкранный формат.
- Содержит коллекцию предустановленных *обложек*, которые позволяют настраивать внешний вид элементов управления воспроизведением.
- Позволяет выбрать цвет и значения альфа-каналов для предустановленных обложек.
- Позволяет опытным пользователям создавать собственные обложки.
- Позволяет воспользоваться интерактивным просмотром во время разработки.
- Содержит свойства макета, которые позволяют сохранить центрирование видеофайла при изменении размера.
- Позволяет начать воспроизведение при загрузке достаточного объема последовательно загружаемого видеофайла.
- Содержит ключевые точки, которые позволяют синхронизировать видео с текстом, графикой и анимацией.
- Поддерживает SWF-файлы умеренных размеров.

Использование компонента FLVPlayback

Использование компонента FLVPlayback заключается в его размещении в рабочей области и указании видеофайла для воспроизведения при помощи этого компонента. Помимо этого можно задать различные параметры, которые управляют его поведением и характеризуют видеофайл.

Компонент FLVPlayback также включает в себя API-интерфейс ActionScript (интерфейс программирования приложений). API-интерфейс содержит следующие классы, которые подробно описаны в документе *Справочник по языку ActionScript 3.0 и компонентам*: CuePointType, FLVPlayback, FLVPlaybackCaptioning, NCManager, NCManagerNative, VideoAlign, VideoError, VideoPlayer, VideoStateи несколько классов событий — AutoLayoutEvent, LayoutEvent, MetadataEvent, SkinErrorEvent, SoundEvent, VideoEvent и VideoProgressEvent.

Компонент FLVPlayback содержит компоненты пользовательского интерфейса для воспроизведения FLV-файлов. Компонент FLVPlayback сочетает в себе область отображения, или видеопроигрыватель, в котором выполняется просмотр видеофайла, а также элементы управления, которые позволяют управлять проигрывателем. Компоненты пользовательского интерфейса для воспроизведения FLV-файлов имеют кнопки управления и механизмы, при помощи которых можно воспроизводить, останавливать, ставить на паузу видеофайл и другим образом управлять им. Эти элементы управления включают в себя следующие: BackButton, BufferingBar, CaptionButton (для компонента FLVPlaybackCaptioning), ForwardButton, FullScreenButton, MuteButton, PauseButton, PlayButton, PlayPauseButton, SeekBar, StopButton и VolumeBar. Компонент FLVPlayback и элементы управления воспроизведением FLV-файлов отображаются на панели "Компоненты", как показано на рисунке ниже:



Компоненты FLVPlayback на панели "Компоненты"

Процесс добавления элементов управления воспроизведением в компонент FLVPlayback называется *выбор обложки*. Компонент FLVPlayback имеет обложку по умолчанию, SkinOverAll.swf, содержащую такие элементы управления, как воспроизведение, остановка, назад, вперед, поиск, без звука, громкость, во весь экран и субтитры. Для изменения этой обложки можно сделать следующее:

- Выбрать обложку из коллекции предустановленных обложек.
- Создать пользовательскую обложку и добавить ее в коллекцию предустановленных обложек.
- Выбрать отдельные элементы управления из набора компонентов интерфейса воспроизведения FLV-файлов и настроить их.

  При выборе предустановленной обложки можно отдельно выбрать цвет и значения альфа-каналов как во время разработки, так и при исполнении. Дополнительную информацию см. в разделе «Выбор предустановленной обложки» на странице 161.

  При выборе другой обложки она становится новой обложкой по умолчанию.

  Дополнительную информацию о выборе или создании обложки для компонента FLVPlayback см. в разделе «Настройка компонента FLVPlayback» на странице 161.

Создание приложения с компонентом FLVPlayback

Компонент FLVPlayback можно включить в приложение следующим образом:

- Перетащите компонент FLVPlayback с панели "Компоненты" в рабочую область и задайте значение для параметра `source`.
- При помощи мастера импорта видеоданных создайте компонент в рабочей области и настройте его, выбрав обложку.
- При помощи конструктора `FLVPlayback()` динамически создайте экземпляр FLVPlayback в рабочей области при наличии этого компонента в библиотеке.

  Примечание. *При создании экземпляра FLVPlayback с использованием ActionScript необходимо назначить ему обложку, задав свойство `skin` при помощи ActionScript. В случае применения обложки таким способом она не публикуется автоматически с SWF-файлом. Необходимо скопировать и SWF-файл приложения, и SWF-файл обложки на сервер приложений. Иначе SWF-файл обложки будет недоступен при выполнении приложения.*

Перетаскивание компонента FLVPlayback с панели "Компоненты"

1. На панели "Компоненты" нажмите кнопку "плюс" (+), чтобы открыть запись видео.
2. Перетащите компонент FLVPlayback в рабочую область.
3. При выбранном в рабочей области компоненте FLVPlayback перейдите к ячейке "Значение" параметра `source` на вкладке "Параметры" в Инспекторе компонентов и введите строку с указанием на один из следующих вариантов:
   - Локальный путь к видеофайлу.
   - URL-адрес, указывающий на видеофайл.
   - URL-адрес, указывающий на синхронизированный файл Multimedia Integration Language (SMIL) с описанием процесса воспроизведения видеофайла.

     Информацию о том, как создать SMIL-файл для описания одного или нескольких FLV-файлов, см. в разделе «Использование SMIL-файла» на странице 173.
4. На вкладке "Параметры" в Инспекторе компонентов при выбранном в рабочей области компоненте FLVPlayback щелкните ячейку "Значение" для параметра `skin`.
5. Нажмите значок лупы, чтобы открыть диалоговое окно "Выбрать обложку".
6. Выберите один из следующих вариантов:
   - В раскрывающемся списке "Обложка" выберите одну из предустановленных обложек, чтобы прикрепить к компоненту набор элементов управления воспроизведением.
   - Если вы создали пользовательскую обложку, выберите пункт "Нестандартный URL-адрес обложки" во всплывающем меню и введите в соответствующее поле URL-адрес SWF-файла, содержащего обложку.
   - Выберите "Нет" и перетащите отдельные компоненты пользовательского интерфейса для воспроизведения FLV-файлов в рабочую область, чтобы добавить элементы управления воспроизведением.

     Примечание. *В первых двух случаях над всплывающим меню на панели просмотра можно выполнить предварительный просмотр обложки. Цвет обложки можно изменить при помощи палитры цветов.*

Для изменения цвета пользовательского элемента управления его необходимо настроить. Дополнительную информацию об использовании элементов управления пользовательского интерфейса см. в разделе «Выбор обложки для отдельных компонентов пользовательского интерфейса для воспроизведения FLV-файлов» на странице 163.

7 Нажмите кнопку "ОК", чтобы закрыть диалоговое окно "Выбрать обложку".

8 Выберите "Управление" > "Тестировать ролик" для выполнения SWF-файла и запуска видео.

Следующая процедура использует мастер импорта видеоданных для добавления компонента FLVPlayback:

Использование мастера импорта видеоданных:

1 Выберите "Файл" > "Импорт" > "Импорт видеоролика".

2 Укажите местоположение видеофайла выбрав один из следующих вариантов:

- На локальном компьютере
- Уже развернуто на веб-сервере, в службе Flash Video Streaming Service или на сервере Flash Media Server

3 В зависимости от вашего выбора введите либо путь, либо URL-адрес, указывающий на местоположение видеофайла, затем нажмите "Далее".

4 Выбрав путь к файлу, вы увидите диалоговое окно "Развертывание", в котором можно выбрать один из перечисленных вариантов развертывания видео:

- Последовательная загрузка со стандартного веб-сервера
- Потоковая передача с Flash Video Streaming Service
- Потоковая передача с сервера Flash Media Server
- Внедрение видео в SWF-файл и воспроизведение во временной шкале

  Важная информация. *Не выбирайте параметр внедрения видео. Компонент FLVPlayback воспроизводит только внешнее потоковое видео. При выборе данного параметра компонент FLVPlayback не будет помещен в рабочую область.*

5 Нажмите кнопку "Далее".

6 Выберите один из следующих вариантов:

- В раскрывающемся списке "Обложка" выберите одну из предустановленных обложек, чтобы прикрепить к компоненту набор элементов управления воспроизведением.
- Если вы создали пользовательскую обложку для компонента, выберите пункт "Нестандартный URL-адрес обложки" во всплывающем меню и введите в соответствующее поле URL-адрес SWF-файла, содержащего обложку.
- Выберите "Нет" и перетащите отдельные компоненты пользовательского интерфейса для воспроизведения FLV-файлов в рабочую область, чтобы добавить элементы управления воспроизведением.

  Примечание. *В первых двух случаях над всплывающим меню на панели просмотра можно выполнить предварительный просмотр обложки.*

7 Нажмите кнопку "ОК", чтобы закрыть диалоговое окно "Выбрать обложку".

8 Прочтите информацию о том, что произойдет дальше, в диалоговом окне "Завершить импорт видеоролика" и нажмите кнопку "Готово".

9 Если вы не сохранили ваш FLA-файл, появится диалоговое окно "Сохранить как".

10 Выберите "Управление" > "Тестировать ролик" для выполнения SWF-файла и запуска видео.

Следующая процедура добавляет компонент FLVPlayback при помощи ActionScript.

Динамическое создание экземпляра при помощи ActionScript:

1 Перетащите компонент FLVPlayback с панели "Компоненты" на панель "Библиотека" ("Окно" > "Библиотека").

2 Вставьте следующий код в Кадр 1 временной шкалы на панели "Действия". Измените параметр *install_drive* на диск, на котором установлено приложение Flash, и измените путь в соответствии с местоположением папки Skins вашей установки:

На компьютере с ОС Windows:

```
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
CS4/en/Configuration/FLVPlayback Skins/ActionScript 3.0/SkinOverPlaySeekMute.swf"
my_FLVPlybk.source = "http://www.helpexamples.com/flash/video/water.flv";
```

На компьютере с ОС Macintosh:

```
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
CS4:Configuration:FLVPlayback Skins:ActionScript 3.0SkinOverPlaySeekMute.swf"
my_FLVPlybk.source = "http://www.helpexamples.com/flash/video/water.flv";
```

Примечание. *Если не задать свойства* `source` *и* `skin`*, созданный фрагмент ролика будет пуст.*

3 Выберите "Управление" > "Тестировать ролик" для выполнения SWF-файла и запуска видеофайла.

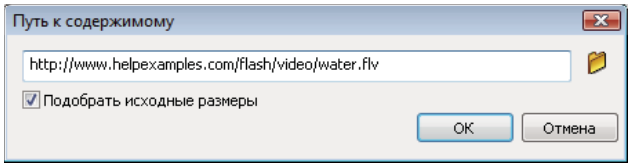
Параметры компонента FLVPlayback

Для каждого экземпляра компонента FLVPlayback можно задать следующие параметры в Инспекторе компонентов или в Инспекторе свойств: `align`, `autoPlay`, `cuePoints`, `preview`, `scaleMode`, `skin`, `skinAutoHide`, `skinBackgroundAlpha`, `skinBackgroundColor`, `source` и `volume`. Каждый из этих параметров имеет соответствующее свойство ActionScript с тем же именем. При присвоении значения этим параметрам вы устанавливаете исходное состояние свойства в приложении. Установка свойства в ActionScript переопределяет значение, заданное параметру. Информацию о возможных значениях этих параметров см. в описании класса FLVPlayback в документе *Справочник по языку ActionScript 3.0 и компонентам*.

Задание параметра source компонента FLVPlayback

Параметр `source` позволяет задать имя и местоположение видеофайла, которые сообщают Flash о том, как воспроизводить этот файл.

Откройте диалоговое окно "Путь к содержимому", дважды щелкнув ячейку "Значение" параметра `source` в Инспекторе компонентов.



Диалоговое окно "Путь к содержимому" компонента FLVPlayback

Диалоговое окно "Путь к содержимому" содержит флажок "Подогнать под размер исходного FLV-файла", указывающий на то, должен ли размер экземпляра FLVPlayback в рабочей области совпадать с размером исходного видеофайла. Исходный видеофайл содержит предпочтительную величину высоты и ширины для воспроизведения. При выборе этого параметра размеры экземпляра FLVPlayback изменяются в соответствии с предпочтительными размерами.

Источник

Введите URL-адрес или локальный путь либо к видеофайлу, либо к XML-файлу, который содержит описание способа воспроизведения видеофайла. Если вы не знаете точное местоположение видеофайла, нажмите значок папки, чтобы открыть диалоговое окно "Обзор" для поиска правильного местоположения. При переходе к FLV-файлу, если он расположен выше или ниже местонахождения целевого SWF-файла, Flash автоматически указывает относительный путь, пригодный для доступа через веб-сервер. В противном случае путь является абсолютным путем Windows или Macintosh. При введении имени локального XML-файла введите путь и имя.

При указании URL-адреса формата HTTP видеофайл воспроизводится как последовательно загружаемое содержимое. При указании URL-адреса формата RTMP выполняется потоковая загрузка видеофайла с сервера Flash Media Server или FVSS. URL-адрес XML-файла также может указывать на потоковый видеофайл, загружаемый с сервера Flash Media Server или FVSS.

Важная информация.

Может быть также задано местоположение SMIL-файла, описывающего способ воспроизведения нескольких потоковых видеофайлов для разной пропускной способности. Этот файл содержит описание FLV-файлов на языке SMIL (Multimedia Integration Language). Описание файла SMIL см. в разделе «Использование SMIL-файла» на странице 173.

Можно также указать имя и местоположение видеофайла при помощи свойства ActionScript `FLVPlayback.source`, а также методов `FLVPlayback.play()` и `FLVPlayback.load()`. Эти три варианта имеют преимущественное значение перед параметром `source` в Инспекторе компонентов. Дополнительную информацию см. в параграфах `FLVPlayback.source`, `FLVPlayback.play()` и `FLVPlayback.load()` описания класса FLVPlayback в документе *Справочник по языку ActionScript 3.0 и компонентам*.

Использование интерактивного просмотра

Параметр FLVPlayback `preview` обеспечивает просмотр кадра исходного видеофайла в компоненте в рабочей области, а также просмотр изменений компонента. При щелчке мышью параметра `preview` открывается следующее диалоговое окно, в котором воспроизводится исходный видеофайл SWF.

Диалоговое окно выбора кадра для интерактивного просмотра

Нажмите кнопку "OK", когда видеофайл достигнет монтажного кадра, захват которого нужно выполнить для предварительного просмотра компонента в рабочей области. Отображение кадра видеофайла в компоненте в рабочей области позволяет увидеть его относительно других элементов приложения.

Можно также экспортировать выбранный кадр, чтобы сохранить его как PNG-файл (portable network graphics) в нужном местоположении.

Поддержка полноэкранного режима

Версия ActionScript 3.0 компонента FLVPlayback поддерживает полноэкранный режим, для которого требуется проигрыватель Flash Player 9.0.28.0 или более поздней версии, а также правильная настройка HTML для полноэкранного просмотра. Некоторые предустановленные обложки включают в себя кнопку включения/выключения полноэкранного режима. Значок FullScreenButton отображается в правой части панели управления, как показано на рисунке ниже.

Значок полноэкранного режима на панели управления

Полноэкранный режим можно включить, только если свойство `fullScreenTakeOver` установлено на значение `true`, которое задается по умолчанию.

Полноэкранный режим поддерживается как при наличии, так и при отсутствии аппаратного ускорения. О поддержке аппаратного ускорения см. в разделе «Аппаратное ускорение» на странице 150.

Реализация поддержки полноэкранного режима для компонента FLVPlayback:

1. Добавьте компонент FLVPlayback в ваше приложение и назначьте ему видеофайл.
2. Выберите обложку для компонента FLVPlayback, которая имеет кнопку полноэкранного режима (например, SkinUnderPlaySeekFullscreen.swf) или добавьте компонент пользовательского интерфейса FullScreenButton в компонент FLVPlayback из раздела "Видео" на панели "Компоненты".
3. Выберите "Файл" > "Параметры публикации".
4. В диалоговом окне "Параметры публикации" выберите вкладку "HTML".

5 На вкладке "HTML" выберите пункт "Flash с поддержкой полноэкранного режима" во всплывающем меню "Шаблон".

6 Также на вкладке "HTML" установите флажок "Определить версию Flash" и укажите версию 9.0.28 или более новую, в зависимости от версии используемого проигрывателя Flash Player.

7 Выберите вкладку "Форматы" и убедитесь, что параметры Flash (.swf) и HTML (.html) выбраны. Можно заменить имена файлов по умолчанию.

8 Нажмите "Опубликовать", затем кнопку "ОК".

Вместо выполнения шага 7 вы можете нажать кнопку "ОК", затем выбрать "Файл" > "Просмотр публикации" > "По умолчанию — (HTML)", чтобы автоматически открыть экспортированный HTML-файл в обозревателе по умолчанию. Либо откройте экспортированный HTML-файл в обозревателе, чтобы протестировать полноэкранный режим.

Чтобы вставить в веб-страницу компонент FLVPlayback с поддержкой полноэкранного режима, откройте экспортированный HTML-файл и скопируйте код, внедряющий SWF-файл в HTML-файл для вашей веб-страницы. Этот код должен выглядеть, как в следующем примере:

```
//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>
```

В качестве альтернативы можно использовать экспортированный HTML-файл в качестве шаблона веб-страницы и добавлять в него другое содержимое. В этом случае следует изменить имя HTML-файла, чтобы случайно не перезаписать его при последующем экспорте HTML-файла FLVPlayback из Flash.

В любом случае необходимо выгрузить на веб-сервер файл AC_RunActiveContent.js, который экспортируется в ту же папку, что и HTML-файл.

Поддержка полноэкранного режима ActionScript включает в себя свойства `fullScreenBackgroundColor`, `fullScreenSkinDelay` и `fullScreenTakeOver`, а также метод `enterFullScreenDisplayState()`. Информацию об этих элементах ActionScript см. в документе *Справочник по языку ActionScript 3.0 и компонентам*.

Использование метода enterFullScreenDisplayState()

Включить полноэкранный режим можно также путем вызова метода ActionScript `enterFullScreenDisplayState()`, как показано в следующем примере.

```
function handleClick(e:MouseEvent):void {
    myFLVPlybk.enterFullScreenDisplayState();
}
myButton.addEventListener(MouseEvent.CLICK, handleClick);
```

В данном примере полноэкранный режим включается *не* нажатием кнопки включения/выключения полноэкранного режима на обложке компонента FLVPlayback, а нажатием кнопки (MyButton), которую автор веб-страницы включил в страницу для вызова полноэкранного режима. Нажатие этой кнопки инициирует обработчик событий `handleClick`, который вызывает метод `enterFullScreen DisplayState()`.

Метод `enterFullScreenDisplayState()` устанавливает свойство `Stage.displayState` на значение `StageDisplayState.FULL_SCREEN` и, следовательно, имеет те же ограничения, что и свойство `displayState`. Дополнительную информацию о методе `enterFullScreenDisplayState()` и свойстве `Stage.displayState` см. в документе *Справочник по языку ActionScript 3.0 и компонентам*.

Аппаратное ускорение

Проигрыватель Flash Player 9.0.115.0 и более поздних версий включает в себя код, который использует доступные видеоустройства для повышения производительности и надежности FLV-файлов, воспроизводимых компонентом FLVPlayback в полноэкранном режиме. Если необходимые условия выполнены, и свойство `fullScreenTakeOver` установлено на значение `true`, проигрыватель Flash Player использует аппаратное ускорение для масштабирования видеофайла вместо программного масштабирования. Если компонент FLVPlayback выполняется в более ранней версии проигрывателя Flash Player, или если необходимые условия для аппаратного ускорения не существуют, проигрыватель Flash Player выполняет масштабирование видеофайла самостоятельно, как раньше.

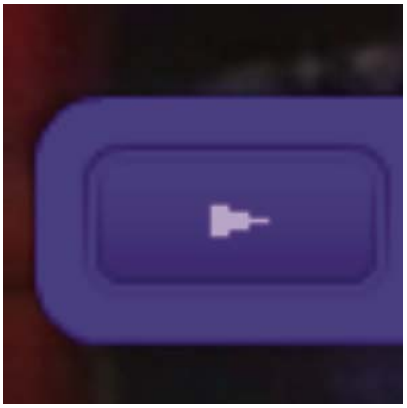
Чтобы использовать аппаратное ускорение для поддержки полноэкранного режима, ваш компьютер должен иметь DirectX 7-совместимую видеокарту с VRAM (видео ОЗУ) 4 МБ или больше. Такая аппаратная поддержка доступна в Windows 2000 или Mac OS X 10.2 и в более поздних версиях этих операционных систем. Direct X® содержит набор API-интерфейсов, которые представляют собой интерфейс взаимодействия программного обеспечения и видеоустройства для ускорения трехмерной и двухмерной графики и реализации других возможностей.

Чтобы воспользоваться преимуществом режима аппаратного ускорения, необходимо вызвать полноэкранный режим одним из следующих способов:

- При помощи кнопки включения/выключения полноэкранного режима на обложке компонента FLVPlayback.
- При помощи элемента управления видео FullScreenButton.
- При помощи метода ActionScript `enterFullScreenDisplayState()`. Дополнительную информацию см. в разделе «Использование метода enterFullScreenDisplayState()» на странице 149.

  При вызове полноэкранного режима путем установки свойства `Stage.displayState` на значение `StageDisplayState.FULLSCREEN` компонент FLVPlayback не использует аппаратное ускорение даже при доступном видеоустройстве и памяти.

  Следствием использования аппаратного ускорения для поддержки полноэкранного режима является масштабирование обложек компонента FLVPlayback вместе с видеопроигрывателем и видеофайлом. Следующий рисунок иллюстрирует влияние полноэкранного режима с использованием аппаратного ускорения на обложку компонента FLVPlayback, фрагмент которой показан на этом рисунке при максимальном разрешении.



Полноэкранный режим на мониторе с разрешением 1600x1200 пикселов и видео размером 320x240 пикселов

На данном рисунке показан результат использования полноэкранного режима на экране с разрешением 1600 x 1200 точек с видеофайлом шириной 320 и высотой 240 пикселов, которые являются размерами компонента FLVPlayback по умолчанию. Искажение обложки более заметно при FLV-файлах с меньшими размерами на экранах с большим разрешением. И наоборот, искажение менее заметно при больших FLV-файлах на экранах с меньшим разрешением. Например, при увеличении размера с 640 x 480 до 1600 x 1200 пикселов размер обложки увеличивается, но искажения менее заметны.

Для ограничения масштабирования обложки компонента FLVPlayback можно задать свойство `skinScaleMaximum`. Значением по умолчанию является 4.0, или 400 %. При ограниченном масштабировании обложки для масштабирования FLV-файла требуется сочетание аппаратных средств и программного обеспечения, что может отрицательно повлиять на исполнение FLV-файлов больших размеров с высокой скоростью потока. Если FLV-файл имеет большие размеры (например, ширина 640 пикселов или больше, высота 480 пикселов или больше), не следует устанавливать для свойства `skinScaleMaximum` низкие значения, так как это может привести к проблемам производительности на больших мониторах. Свойство `skinScaleMaximum` позволяет выбрать оптимальное соотношение между производительностью и качеством и внешним видом большой обложки.

Выход из полноэкранного режима

Для выхода из полноэкранного режима снова нажмите кнопку полноэкранного режима или клавишу Esc.

Задание следующих свойств и вызов следующих методов может повлечь за собой изменения макета, которые могут привести к выходу компонента FLVPLayback из полноэкранного режима: `height`, `registrationHeight`, `registrationWidth`, `registrationX`, `registrationY`, `scaleX`, `scaleY`, `width`, `x`, `y`, `setScale()` или `setSize()`.

При задании свойства `align` или свойства `scaleMode` компонент FLVPlayback устанавливает их на значения `center` и `maintainAspectRatio` соответственно до выхода из полноэкранного режима.

Изменение значения свойства `fullScreenTakeOver` с `true` на `false` при использовании полноэкранного режима с аппаратным ускорением также влечет за собой выход Flash из полноэкранного режима.

Выравнивание макета для воспроизведения нескольких видеофайлов

Компонент ActionScript 3.0 FLVPlayback имеет свойство `align`, которое указывает на то, следует ли центрировать видеофайл при изменении его размера или его расположении по верхнему, нижнему, левому или правому краю компонента. Помимо свойств `x`, `y`, `width` и `height` компонент ActionScript 3.0 также имеет свойства `registrationX`, `registrationY`, `registrationWidth` и `registrationHeight`. Исходно они соответствуют свойствам `x`, `y`, `width` и `height`. При загрузке последовательных видеофайлов автоматическое изменение макета не меняет эти свойства, поэтому новый видеофайл может быть отцентрирован в том же месте. Если `scaleMode = VideoScaleMode.MAINTAIN_ASPECT_RATIO`, размеры последовательных FLV-файлов могут быть подогнаны под исходные размеры компонента, не вызывая изменение его ширины и высоты.

Автоматическое воспроизведение последовательно загружаемых видеофайлов

При загрузке последовательно загружаемого видеофайла компонент FLVPlayback начинает воспроизведение этого файла, только если загружен достаточный объем для воспроизведения файла с начала до конца.

Для преждевременного воспроизведения видеофайла можно вызвать метод `play()` без параметров.

Вернуться в режим ожидания достаточной загрузки файла можно путем вызова метода `pause()` с последующим вызовом метода `playWhenEnoughDownloaded()`.

Использование ключевых точек

Ключевая точка — это точка, в которой видеопроигрыватель отправляет событие `cuePoint` во время воспроизведения видеофайла. Можно добавить ключевые точки в FLV-файл в том месте, когда должно быть выполнено действие другого элемента веб-страницы. Например, можно отобразить текст или графику, или выполнить синхронизацию с Flash-анимацией, или повлиять на воспроизведение FLV-файла, установив паузу, выполнив поиск другой точки в видео или переключившись на другой FLV-файл. Ключевые точки позволяют при помощи кода ActionScript синхронизировать точки в FLV-файле с другими действиями на веб-странице.

Существует три вида ключевых точек: навигация, событие и ActionScript. Ключевые точки навигации и события также известны как *встроенные* ключевые точки, так как они встроены в поток FLV-файла и в пакет метаданных FLV-файла.

Ключевая точка навигации позволяет выполнять поиск конкретного кадра в FLV-файле, так как она создает ключевой кадр в FLV-файле как можно ближе к указанной вами временной точке. *Ключевой кадр* — это фрагмент данных, расположенный между кадрами изображения в потоке FLV-файла. При поиске ключевой точки навигации компонент выполняет поиск ключевого кадра и запускает событие `cuePoint`.

Ключевая точка события позволяет синхронизировать момент времени в FLV-файле с внешним событием веб-страницы. Событие `cuePoint` происходит точно в указанное время. Встроить ключевые точки навигации и события в FLV-файл можно либо при помощи мастера импорта видеоданных, либо при помощи Flash Video Encoder. Дополнительную информацию о мастере импорта видеоданных и о Flash Video encoder см. в Главе 16 "Работа с видео" в руководстве *Использование Flash*.

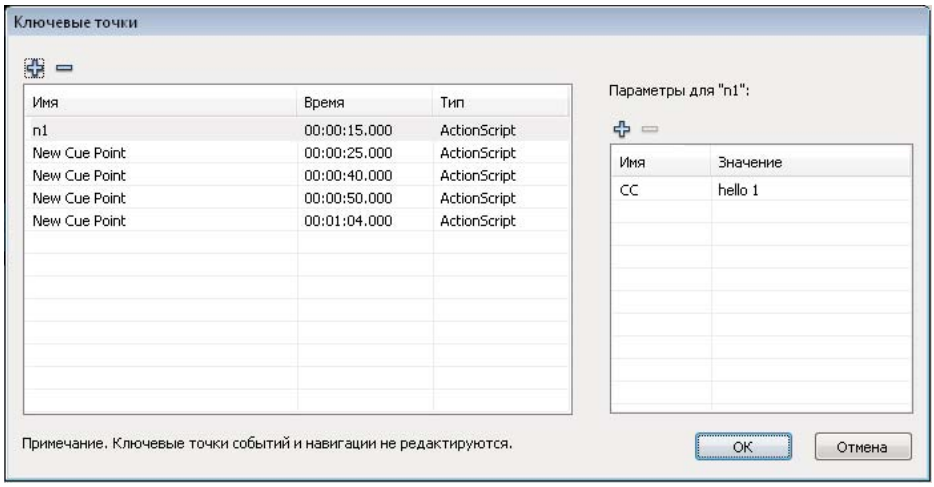
Ключевая точка ActionScript является внешней ключевой точкой, которую можно добавить либо при помощи диалогового окна компонента "Ключевые точки Flash-видео", либо при помощи метода `FLVPlayback.addASCuePoint()`. Компонент хранит и отслеживает ключевые точки ActionScript отдельно от FLV-файла, поэтому они менее точные, чем встроенные ключевые точки. Ключевые точки ActionScript имеют точность до одной десятой секунды. Точность ключевых точек ActionScript можно повысить, снизив значение свойства `playheadUpdateInterval`, так как компонент создает событие `cuePoint` для ключевых точек ActionScript при обновлении указателя воспроизведения. Дополнительную информацию см. в описании свойства FLVPlayback.playheadUpdateInterval в документе *Справочник по языку ActionScript 3.0 и компонентам*.

В ActionScript и в метаданных FLV-файла ключевая точка представлена в виде объекта со следующими свойствами: `name`, `time`, `type` и `parameters`. Свойство `name` является строкой, содержащей присвоенное ключевой точке имя. Свойство `time` является числом, представляющим время появления ключевой точки в часах, минутах, секундах и миллисекундах (ЧЧ:ММ:СС.ммм). Свойство `type` является строкой, значениями которой являются `"navigation"`, `"event"` или `"actionscript"`, в зависимости от типа созданной ключевой точки. Свойство `parameters` представляет собой массив указанных пар имя-значение.

При наступлении события `cuePoint` объект ключевой точки становится доступным в объекте события через свойство `info`.

Использование диалогового окна "Ключевые точки Flash-видео"

Откройте диалоговое окно "Ключевые точки Flash-видео", дважды щелкнув ячейку "Значение" параметра `cuePoints` в Инспекторе компонентов. Диалоговое окно выглядит так, как показано на рисунке ниже:



диалоговое окно "Ключевые точки"

В диалоговом окне отображаются встроенные ключевые точки и ключевые точки ActionScript. При помощи этого диалогового окна можно добавлять и удалять ключевые точки ActionScript, а также параметры ключевых точек. Можно также включать и отключать встроенные ключевые точки. Однако добавление, изменение или удаление встроенных ключевых точек невозможно.

Добавление ключевой точки ActionScript:

1. Дважды щелкните ячейку "Значение" параметра `cuePoints` в Инспекторе компонентов, чтобы открыть диалоговое окно "Ключевые точки Flash-видео".
2. Нажмите знак "плюс" (+) в верхнем левом углу над списком ключевых точек, чтобы добавить запись ключевой точки ActionScript по умолчанию.

3 Щелкните текст "Новая ключевая точка" в столбце "Имя" и присвойте имя ключевой точке.

4 Щелкните значение времени 00:00:00:000, чтобы изменить его, и назначьте время появления ключевой точки. Можно указать время в часах, минутах, секундах и миллисекундах (ЧЧ:ММ:СС.ммм).

Если существует несколько ключевых точек, новая ключевая точка помещается на позицию списка в соответствии с хронологическим порядком.

5 Для добавления параметра выбранной ключевой точки нажмите знак "плюс" (+) вверху раздела "Параметры" и введите значения в столбцы "Имя" и "Значение". Повторите этот шаг для добавления каждого параметра.

6 Для добавления нескольких ключевых точек ActionScript повторите шаги 2-5 для каждой точки.

7 Чтобы сохранить изменения, нажмите кнопку "OK".

Удаление ключевой точки ActionScript:

1 Дважды щелкните ячейку "Значение" параметра `cuePoints` в Инспекторе компонентов, чтобы открыть диалоговое окно "Ключевые точки Flash-видео".

2 Выберите ключевую точку, которую нужно удалить.

3 Нажмите знак "минус" (-) в верхнем левом углу над списком ключевых точек, чтобы удалить точку.

4 Повторите шаги 2-3 для удаления каждой ключевой точки.

5 Чтобы сохранить изменения, нажмите кнопку "OK".

Включение и отключение встроенной ключевой точки FLV-файла:

1 Дважды щелкните ячейку "Значение" параметра `cuePoints` в Инспекторе компонентов, чтобы открыть диалоговое окно "Ключевые точки Flash-видео".

2 Выберите ключевую точку, которую нужно включить или отключить.

3 Щелкните значение в столбце "Тип", чтобы вызвать всплывающее меню, или нажмите стрелку вниз.

4 Щелкните название типа ключевой точки (например, "Событие" или "Навигация"), чтобы включить ее. Нажмите "Отключено", чтобы отключить ее.

5 Чтобы сохранить изменения, нажмите кнопку "OK".

Использование ключевых точек с ActionScript

При помощи ActionScript можно добавлять ключевые точки ActionScript, прослушивать события `cuePoint`, находить какие-либо ключевые точки или ключевые точки определенного типа, искать ключевые точки навигации, включать или отключать ключевые точки, проверять, включена ли ключевая точка, и удалять ключевые точки.

В примерах данного раздела используется FLV-файл с именем cuepoints.flv, который содержит три ключевые точки:

Имя	Время	Тип
point1	00:00:00.418	Навигация
point2	00:00:07.748	Навигация
point3	00:00:16.020	Навигация

Добавление ключевых точек ActionScript

Добавлять ключевые точки ActionScript в FLV-файл можно при помощи метода `addASCuePoint()`. Следующий пример добавляет две ключевые точки ActionScript в FLV-файл, когда он готов к воспроизведению. Первая ключевая точка добавляется с использованием объекта ключевой точки, который указывает имя, время и тип ключевой точки в ее параметрах. При втором вызове указывается время и имя при помощи параметров метода `time` и `name`.

```
// 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");
```

Дополнительную информацию см. в описании метода FLVPlayback.addASCuePoint() в документе *Справочник по языку ActionScript 3.0 и компонентам*.

Прослушивание событий cuePoint

Событие `cuePoint` позволяет получить контроль в коде ActionScript при наступлении события `cuePoint`. При возникновении ключевых точек в следующем примере прослушиватель `cuePoint` вызывает функцию обработчика событий, которая отображает значение свойства `playheadTime`, а также имя и тип ключевой точки. Используйте этот пример в сочетании с примером в предыдущем разделе, "Добавление ключевых точек ActionScript", чтобы увидеть результаты.

```
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);
}
```

Дополнительную информацию о событии `cuePoint` см. в описании события FLVPlayback.cuePoint в документе *Справочник по языку ActionScript 3.0 и компонентам*.

Поиск ключевых точек

При помощи ActionScript можно найти ключевую точку любого типа, найти ключевую точку, ближайшую к моменту времени, или найти следующую ключевую точку с конкретным именем.

Обработчик событий `ready_listener()` в следующем примере вызывает метод `findCuePoint()` для поиска ключевой точки `ASpt1`, а затем вызывает метод `findNearestCuePoint()` для поиска ключевой точки навигации, ближайшей к времени ключевой точки `ASpt1`:

```
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);
}
```

В следующем примере обработчик событий `ready_listener()` находит ключевую точку `ASpt` и вызывает метод `findNextCuePointWithName()` для поиска следующей ключевой точки с тем же именем:

```
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);
}
```

Дополнительную информацию о поиске ключевых точек см. в описании методов FLVPlayback.findCuePoint(), FLVPlayback.findNearestCuePoint() и FLVPlayback.findNextCuePointWithName() в документе *Справочник по языку ActionScript 3.0 и компонентам*.

Поиск ключевых точек навигации

Можно выполнять поиск ключевой точки навигации, ключевой точки навигации, следующей за определенным моментом времени или предшествующей определенному моменту времени. Следующий пример воспроизводит FLV-файл cuepoints.flv и выполняет поиск ключевой точки с временем 7.748 при наступлении события `ready`. При наступлении события `cuePoint` вызывается метод `seekToPrevNavCuePoint()` для поиска первой ключевой точки. При наступлении этого события `cuePoint` вызывается метод `seekToNextNavCuePoint()` для поиска последней ключевой точки путем добавления 10 секунд ко времени `eventObject.info.time`, которое является временем текущей ключевой точки.

```
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";
```

Дополнительную информацию см. в описании методов FLVPlayback.seekToNavCuePoint(), FLVPlayback.seekToNextNavCuePoint()и FLVPlayback.seekToPrevNavCuePoint() в документе *Справочник по языку ActionScript 3.0 и компонентам*.

Включение и отключение встроенных ключевых точек FLV-файла

Встроенные ключевые точки FLV-файла можно включать и отключать при помощи метода `setFLVCuePointEnabled()`. Отключенные ключевые точки не инициализируют события `cuePoint` и не работают с методами `seekToCuePoint()`, `seekToNextNavCuePoint()` и `seekToPrevNavCuePoint()`. Однако отключенные ключевые точки можно найти при помощи методов `findCuePoint()`, `findNearestCuePoint()` и `findNextCuePointWithName()`.

При помощи метода `isFLVCuePointEnabled()` можно проверить, включена ли встроенная ключевая точка FLV-файла. Следующий пример отключает встроенные ключевые точки `point2` и `point3`, когда видео готово к воспроизведению. При наступлении первого события `cuePoint`, однако, обработчик событий проверяет, отключена ли ключевая точка `point3`, и если да, включает ее.

```
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");
        }
}
```

Дополнительную информацию см. в описании свойства FLVPlayback.isFLVCuePointEnabled() и FLVPlayback.setFLVCuePointEnabled()в документе *Справочник по языку ActionScript 3.0 и компонентам*.

Удаление ключевой точки ActionScript

Ключевую точку ActionScript можно удалить при помощи метода `removeASCuePoint()`. Следующий пример удаляет ключевую точку `ASpt2` при встрече ключевой точки `ASpt1`:

```
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");
        }
}
```

Дополнительные сведения см. в разделе FLVPlayback.removeASCuePoint() в документе *Справочник по языку ActionScript 3.0 и компонентам*.

Воспроизведение нескольких видеофайлов

Можно последовательно воспроизводить видеофайлы в экземпляре FLVPlayback, просто задав новый URL-адрес для свойства `source` при окончании воспроизведения предыдущего видеофайла. Например, следующий код ActionScript прослушивает событие `complete`, которое наступает при окончании воспроизведения видеофайла. При наступлении этого события код задает имя и местоположение нового видеофайла в свойстве `source` и вызывает метод `play()` для воспроизведения нового видеофайла.

```
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");
    }
};
```

Использование нескольких видеопроигрывателей

В одном экземпляре компонента FLVPlayback можно открыть несколько видеопроигрывателей для воспроизведения нескольких видеофайлов и перехода между ними во время воспроизведения.

Исходный видеопроигрыватель создается при перетаскивании компонента FLVPlayback в рабочую область. Компонент автоматически присваивает исходному видеопроигрывателю номер 0 и назначает его проигрывателем по умолчанию. Для создания дополнительного видеопроигрывателя просто установите свойство `activeVideoPlayerIndex` на новое число. Задание свойства `activeVideoPlayerIndex` также делает указанный видеопроигрыватель *активным* видеопроигрывателем, т. е. тем, к которому будут применены свойства и методы класса FLVPlayback. Задание свойства `activeVideoPlayerIndex`, однако, не делает видеопроигрыватель видимым. Чтобы сделать его видимым, установите свойство `visibleVideoPlayerIndex` на номер видеопроигрывателя. Дополнительную информацию о том, как эти свойства взаимодействуют со свойствами и методами класса FLVPlayback см. в описании свойств FLVPlayback.activeVideoPlayerIndex и FLVPlayback.visibleVideoPlayerIndex в документе *Справочник по языку ActionScript 3.0 и компонентам*.

Следующий код ActionScript загружает свойство `source` для воспроизведения видеофайла в видеопроигрывателе по умолчанию и добавляет для него ключевую точку. При наступлении события `ready` обработчик событий открывает второй видеопроигрыватель, установив свойство `activeVideoPlayerIndex` на число 1. Он указывает на FLV-файл и ключевую точку для второго видеопроигрывателя, а затем снова назначает проигрыватель по умолчанию (0) активным видеопроигрывателем.

```
/**
    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;
};
```

Для перехода от одного воспроизводимого FLV-файла к другому необходимо создать переключение в коде ActionScript. Ключевые точки позволяют вам вмешаться в определенные моменты в FLV-файле при помощи события `cuePoint`. Следующий код создает прослушиватель для события `cuePoint` и вызывает функцию обработчика, которая ставит активный видеопроигрыватель (0) на паузу, переходит ко второму проигрывателю (1) и воспроизводит его FLV-файл:

```
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);
        }
};
```

При создании нового видеопроигрывателя экземпляр FLVPlayback устанавливает его свойства на значения видеопроигрывателя по умолчанию, кроме свойств `source`, `totalTime`, и `isLive`, которые экземпляр FLVPlayback всегда устанавливает на значения по умолчанию: пустая строка, 0 и `false` соответственно. Он устанавливает свойство `autoPlay`, значением по умолчанию которого является `true` для видеопроигрывателя по умолчанию, на значение `false`. Свойство `cuePoints` не действует и не влияет на последующую загрузку в видеопроигрыватель по умолчанию.

Методы и свойства, управляющие громкостью, расположением, размерами, видимостью и элементами управления пользовательского интерфейса, всегда глобальные, и задание свойства `activeVideoPlayerIndex` не влияет на их поведение. Дополнительную информацию об этих методах и свойствах, а также об эффекте задания свойства `activeVideoPlayerIndex` см. в описании свойства FLVPlayback.activeVideoPlayerIndex в документе *Справочник по языку ActionScript 3.0 и компонентам*. Остальные свойства и методы предназначены для видеопроигрывателя, обозначенного значением свойства `activeVideoPlayerIndex`.

Однако свойства и методы, управляющие размерами, *взаимодействуют* со свойством `visibleVideoPlayerIndex`. Дополнительную информацию см. в описании свойства FLVPlayback.visibleVideoPlayerIndex в документе *Справочник по языку ActionScript 3.0 и компонентам*.

Потоковая передача FLV-файлов с сервера Flash Media Server

Требования к потоковой передаче FLV-файлов с сервера Flash Media Server зависят от того, поддерживает ли ваш поставщик Flash Video Streaming Service встроенную функцию определения полосы пропускания. Встроенная функция определения полосы пропускания означает, что определение полосы пропускания встроено в потоковый сервер и выполняется быстрее. Проверьте, поддерживаются ли у вашего поставщика встроенные функции определения полосы пропускания.

Для доступа к FLV-файлам на сервере Flash Media Server используйте URL-адрес формата rtmp://*my_servername/my_application/stream*.flv.

При воспроизведении интерактивного потока с сервера Flash Media Server необходимо установить свойство компонента FLVPlayback `isLive` на значение `true`. Дополнительную информацию см. в описании свойства FLVPlayback.isLiveв документе *Справочник по языку ActionScript 3.0 и компонентам*.

Дополнительную информацию об администрировании сервера Flash Media Server, включая сведения о том, как создать интерактивный поток, см. в документации по серверу Flash Media Server по адресу www.adobe.com/support/documentation/ru/flashmediaserver/.

В случае наличия встроенной функции определения полосы пропускания или отсутствия определения полосы пропускания

Класс NCManagerNative является подклассом класса NCManager, который поддерживает встроенные функции определения полосы пропускания, которые могу поддерживать некоторые поставщики услуг Flash Video Streaming Service. При использовании NCManagerNative не требуется наличие никаких специальных файлов на сервере Flash Media Server. NCManagerNative также разрешает подключение к любой версии сервера Flash Media Server без файла main.asc, если определение полосы пропускания не требуется.

Для использования NCManagerNative вместо класса по умолчанию NCManager вставьте следующие строки кода в первый кадр FLA-файла:

```
import fl.video*;
VideoPlayer.iNCManagerClass = fl.video.NCManagerNative;
```

В случае отсутствия встроенных функций определения полосы пропускания

Если встроенные функции определения полосы пропускания не поддерживаются вашим поставщиком услуг Flash Video Streaming Service, но определение полосы пропускания необходимо, необходимо добавить файл main.asc в ваше FLV-приложение на сервере Flash Media Server. Файл main.asc доступен в Интернете по адресу www.adobe.com/support/documentation/ru/flash/samples/. Он содержится в файле Samples.zip — в каталоге Samples\ComponentsAS2\FLVPlayback.

Для настройки сервера Flash Media Server на потоковую передачу FLV-файлов:

1. В папке приложения сервера Flash Media Server создайте папку с именем **my_application**.
2. Скопируйте файл main.asc в папку my_application.
3. Создайте папку с именем **streams** в папке my_application.
4. Создайте папку с именем **_definst_** в папке streams.
5. Поместите FLV-файлы в папку **_definst_**.

Настройка компонента FLVPlayback

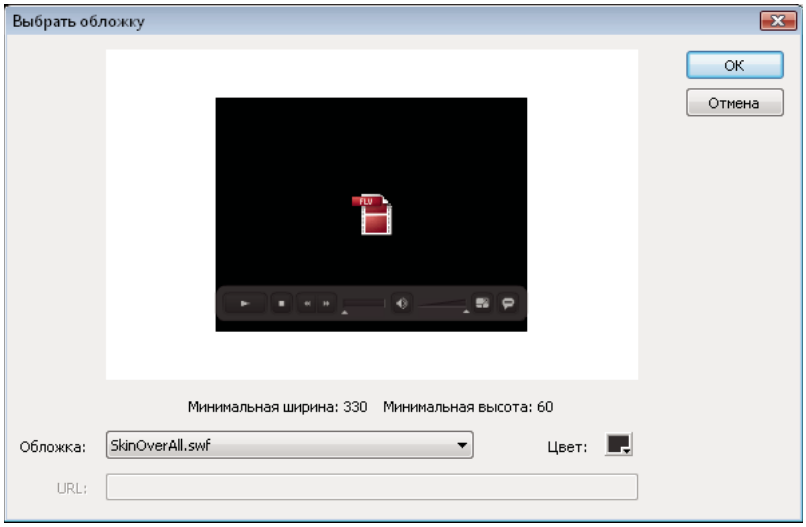
В данном разделе описывается процесс настройки компонента FLVPlayback. Большинство методов, используемых для настройки других компонентов, не работают с компонентом FLVPlayback. Для настройки компонента FLVPlayback используйте только методы, описанные в данном разделе.

Для настройки компонента FLVPlayback можно выполнить следующие действия: выбрать предустановленную обложку, выбрать обложку для отдельных компонентов пользовательского интерфейса для воспроизведения FLV-файлов или создать новую обложку. Для изменения поведения обложки можно также использовать свойства компонента FLVPlayback.

Примечание. *Чтобы обложка работала с вашим компонентом FLVPlayback, необходимо выгрузить SWF-файл обложки на веб-сервер вместе с SWF-файлом приложения.*

Выбор предустановленной обложки

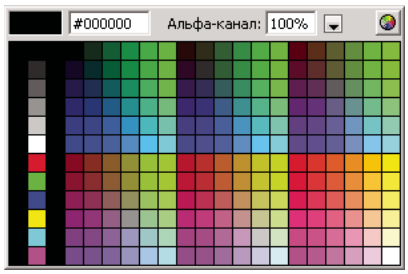
Выбрать обложку для компонента FLVPlayback можно, щелкнув ячейку `value` параметра `skin` в Инспекторе компонентов. Затем нажмите значок лупы, чтобы открыть диалоговое окно "Выбрать обложку", в котором можно выбрать обложку или ввести URL-адрес, указывающий на местоположение SWF-файла обложки.



Диалоговое окно компонента FLVPlayback "Выбрать обложку"

Обложки, перечисленные во всплывающем меню "Обложка", находятся в папке приложения Flash /Flash Configuration/FLVPlayback Skins/ActionScript 3.0. Можно добавить доступные обложки в это диалоговое окно, создав их и поместив SWF-файл в эту папку. Имя обложки появится во всплывающем меню с расширением .swf. Дополнительную информацию о создании набора обложек см. в разделе «Создание новой обложки» на странице 168.

Для обложек, установленных при помощи свойства `skin`, можно задать значения цвета и альфа-каналов (прозрачности) независимо от выбора обложки либо путем задания параметра skin во время разработки, либо при помощи ActionScript при исполнении. Чтобы задать значения цвета и альфа-каналов во время разработки, откройте палитру цветов в диалоговом окне "Выбрать обложку", как показано ниже.



Палитра в диалоговом окне "Выбрать обложку"

Чтобы выбрать цвет, щелкните образец на панели или введите его числовое обозначение в текстовое поле. Чтобы выбрать значение альфа-канала, используйте ползунок или введите величину процента в текстовое поле "Альфа-канал".

Для задания значений цвета и альфа-каналов при исполнении задайте свойства `skinBackgroundColor` и `skinBackgroundAlpha`. Установите свойство `skinBackgroundColor` на значение 0xRRGGBB (красный, зеленый, синий). Установите свойство skinBackgroundAlpha на число между 0,0 и 1,0. Следующий пример устанавливает свойство `skinBackgroundColor` на значение 0xFF0000 (красный), а skinBackgroundAlpha на значение .5.

```
my_FLVPlybk.skinBackgroundColor = 0xFF0000;
my_FLVPlybk.skinBackgroundAlpha = .5;
```

Значениями по умолчанию являются последние выбранные пользователем значения.

Если вы хотите выбрать обложку для компонента FLVPlayback с использованием компонентов пользовательского интерфейса для воспроизведения FLV-файлов, выберите во всплывающем меню пункт "Нет".

Выбор обложки для отдельных компонентов пользовательского интерфейса для воспроизведения FLV-файлов

Компоненты пользовательского интерфейса для воспроизведения FLV-файлов позволяют настроить внешний вид элементов управления компонента FLVPlayback в FLA-файле и увидеть результаты во время предварительного просмотра веб-страницы. Однако эти компоненты не предназначены для масштабирования. Для установки определенного размера необходимо отредактировать фрагмент ролика и его содержимое. По этой причине рекомендуется иметь компонент FLVPlayback в рабочей области нужного размера со свойством `scaleMode` установленным на `exactFit`.

Для начала просто перетащите нужные компоненты пользовательского интерфейса для воспроизведения FLV-файлов с панели "Компоненты" в нужное место в рабочей области и присвойте им имена экземпляров.

Эти компоненты работают без ActionScript. Если вы поместите их на ту же временную шкалу и в тот же кадр, что и компонент FLVPlayback, и для компонента не задана обложка, то компонент FLVPlayback присоединится к ним автоматически. Если в рабочей области расположено несколько компонентов FLVPlayback или если пользовательский элемент управления и экземпляр FLVPlayback находятся не на одной временной шкале, потребуется написать код.

После размещения компонентов в рабочей области их можно отредактировать, как любой другой символ. При открытии компонентов вы увидите, что каждый из них несколько отличается от других.

Компоненты Button

Компоненты Button имеют схожую структуру. Кнопки включают в себя BackButton, ForwardButton, MuteButton, PauseButton, PlayButton, PlayPauseButton и StopButton. Большинство из них имеют один фрагмент ролика в Кадре 1 с именем экземпляра placeholder_mc. Как правило, это экземпляр кнопки в нормальном состоянии, но не обязательно так. В Кадре 2 в рабочей области находятся четыре фрагмента ролика для каждого состояния отображения: нормального, при наведении указателя, нажатого и отключенного. (При исполнении компонент никогда не переходит в Кадр 2; эти компоненты помещены сюда для удобства редактирования и принудительной загрузки в SWF-файл без установки флажка "Экспортировать в первый кадр" в диалоговом окне "Свойства символа". Однако необходимо выбрать параметр "Экспорт для ActionScript".)

Для выбора обложки для кнопки нужно просто отредактировать каждый из этих фрагментов ролика. Можно изменить их размер и внешний вид.

Некий сценарий ActionScript, как правило, отображается в Кадре 1. Этот сценарий изменять не требуется. Он просто останавливает указатель воспроизведения в Кадре 1 и указывает, какие фрагменты роликов использовать для каких состояний.

Кнопки PlayPauseButton, MuteButton, FullScreenButton и CaptionButton

Кнопки PlayPauseButton, MuteButton, FullScreenButton и CaptionButton отличаются от других; они имеют только одну рамку с двумя слоями и не имеют сценария. В этом кадре расположены две кнопки, одна поверх другой. В случае с кнопкой PlayPauseButton — это кнопки воспроизведения и паузы; в случае с MuteButton — кнопки включения и выключения режима без звука; в случае с FullScreenButton — кнопки включения и выключения полноэкранного режима; в случае с CaptionButton — кнопки включения и выключения субтитров. Для выбора обложки этих кнопок необходимо выбрать обложку для каждой из двух внутренних кнопок, как описано в разделе «Выбор обложки для отдельных компонентов пользовательского интерфейса для воспроизведения FLV-файлов» на странице 163; дополнительных действий не требуется.

Кнопка CaptionButton предназначена для компонента FLVPlaybackCaptioning и должна прикрепляться к нему, а не к компоненту FLVPlayback.

Кнопки BackButton и ForwardButton

Кнопки BackButton и ForwardButton также имеют свои особенности. В Кадре 2 они имеют дополнительные фрагменты роликов, которые можно использовать в качестве рамки вокруг одной или обеих кнопок. Эти фрагменты роликов не обязательны и не имеют особых функций; они предоставлены для удобства. Чтобы использовать их, просто перетащите их в рабочую область с панели "Библиотека" и поместите в любое место. Если они вам не нужны, не используйте их или удалите с панели "Библиотека".

Большинство кнопок изначально основано на общем наборе фрагментов роликов, поэтому можно изменить внешний вид всех кнопок сразу. Можете использовать общие наборы или можете заменить эти фрагменты роликов и сделать внешний вид каждой кнопки индивидуальным.

Компонент BufferingBar

Компонент BufferingBar простой: он содержит анимацию, которая проявляется, когда компонент входит в состояние буферизации, и не требует специального сценария ActionScript для настройки. По умолчанию этот компонент представляет собой полосатую строку, двигающуюся слева направо, с прямоугольной маской, но в его конфигурации нет ничего особенного.

Хотя строки буферизации в SWF-файлах обложек используют 9-зонное масштабирование, так как их масштабирование осуществляется при исполнении, компонент пользовательского интерфейса BufferingBar для воспроизведения FLV-файлов не использует и *не может* использовать 9-зонное масштабирование, так как имеет вложенные фрагменты роликов. Если требуется увеличить ширину или высоту компонента BufferingBar, вы можете изменить его содержимое вместо масштабирования.

Компоненты SeekBar и VolumeBar

Компоненты SeekBar и VolumeBar похожи, хотя имеют разные функции. Каждый из компонентов имеет маркеры, использует те же механизмы отслеживания маркера и имеет поддержку вложенных роликов для отслеживания хода выполнения и заполненности.

Во многих случаях код ActionScript в компоненте FLVPlayback предполагает, что точка регистрации (также известная как *исходная точка* или *нулевая точка*) компонента SeekBar или VolumeBar находится в левом верхнем углу содержимого, поэтому очень важно соблюдать это условие. В противном случае могут возникнуть проблемы с маркерами и фрагментами роликов хода выполнения и заполненности.

Хотя строки поиска в SWF-файлах обложек используют 9-зонное масштабирование, так как их масштабирование осуществляется при исполнении, компонент пользовательского интерфейса SeekBar для воспроизведения FLV-файлов не использует и *не может* использовать 9-зонное масштабирование, так как имеет вложенные фрагменты роликов. Если требуется увеличить ширину или высоту компонента SeekBar, вы можете изменить его содержимое вместо масштабирования.

Маркер

Экземпляр фрагмента ролика маркера находится в Кадре 2. Как в случае с компонентами BackButton и ForwardButton, этот компонент никогда не переходит в Кадр 2; эти фрагменты роликов помещены сюда для удобства редактирования и принудительной загрузки в SWF-файл без установки флажка "Экспортировать в первый кадр" в диалоговом окне "Свойства символа". Однако все равно необходимо выбрать параметр "Экспорт для ActionScript".

Как вы заметили, фрагмент ролика маркера имеет фоновый прямоугольник, значение альфа-канала которого установлено на 0. Этот прямоугольник увеличивает размер области попадания маркера, упрощая его захват без изменения внешнего вида, подобно состоянию попадания кнопки. Так как маркер создается динамически при исполнении, он должен быть фрагментом ролика, а не кнопкой. Прямоугольник со значением альфа-канала равным 0 не нужен ни для какой другой цели, и его можно заменить любым изображением. Однако рекомендуется оставить точку регистрации в центре по горизонтали, в середине фрагмента ролика маркера.

Следующий код ActionScript в Кадре 1 компонента SeekBar предназначен для управления маркером:

```
stop();
handleLinkageID = "SeekBarHandle";
handleLeftMargin = 2;
handleRightMargin = 2;
handleY = 11;
```

Вызов функции `stop()` необходим из-за содержимого Кадра 2.

Вторая строка указывает на символ, который нужно использовать в качестве маркера, и его не следует изменять, если вы просто редактируете экземпляр фрагмента ролика маркера в Кадре 2. При исполнении компонент FLVPlayback создает экземпляр указанного фрагмента ролика в рабочей области в качестве компонента того же уровня, что и экземпляр компонента Bar, т. е. они имеют общий родительский фрагмент ролика. Таким образом, если строка находится на корневом уровне, маркер тоже должен быть на корневом уровне.

Переменная `handleLeftMargin` определяет исходное положение маркера (0 %), а переменная `handleRightMargin` определяет его конечное положение (100 %). Эти числа определяют отступ от левого и правого концов элемента управления "строка", где положительные числа задают границы в пределах строки, а отрицательные — за ее пределами. Эти отступы указывают место перемещения маркера на основе его точки регистрации. Если поместить точку регистрации в середине маркера, его дальние левый и правый края выйдут за поля. Для правильной работы фрагмент ролика строки поиска должен иметь точку регистрации в левом верхнем углу содержимого.

Переменная `handleY` определяет положение маркера по оси `y` относительно экземпляра строки. Это положение определяется на основе точек регистрации каждого фрагмента ролика. Точка регистрации образца маркера находится на вершине треугольника, располагая его относительно видимой части, не учитывая невидимый прямоугольник области попадания. Для правильной работы фрагмент ролика строки также должен иметь точку регистрации в левом верхнем углу содержимого.

Например, при таких границах, если для строки заданы границы (100, 100) и ее ширина равна 100 пикселам, границы маркера по горизонтали будут составлять от 102 до 198, а положение по вертикали — 111. Если изменить значения переменных `handleLeftMargin` и `handleRightMargin` на -2, а значение `handleY` на -11, границы маркера по горизонтали будут составлять от 98 до 202, а положение по вертикали — 89.

Фрагменты роликов хода выполнения и заполненности

Компонент SeekBar имеет фрагмент ролика *ход выполнения*, а компонент VolumeBar имеет фрагмент ролика *заполненность*, но на практике любой компонент SeekBar или VolumeBar может иметь любой из этих фрагментов роликов, ни одного из них или оба эти фрагмента роликов. Они имеют одинаковую структуру и схожее поведение, но отслеживают разные значения. Фрагмент ролика хода выполнения заполняется по мере загрузки FLV-файла (что применимо только к загрузке через HTTP, так как эта строка всегда отображается заполненной при потоковой передаче с FMS), а фрагмент ролика заполненности заполняется по мере движения маркера слева направо.

Компонент FLVPlayback находит экземпляры этих фрагментов роликов по конкретным именам экземпляров, поэтому экземпляр фрагмента ролика хода выполнения должен иметь родительский фрагмент ролика строки и имя progress_mc. Экземпляр фрагмента ролика заполненности должен иметь имя fullness_mc.

Фрагменты роликов хода выполнения и заполненности могут иметь или не иметь вложенный экземпляр фрагмента ролика fill_mc. Фрагмент ролика fullness_mc компонента VolumeBar демонстрирует метод *с использованием* фрагмента ролика fill_mc, а фрагмент ролика progress_mc компонента SeekBar демонстрирует метод *без использования* фрагмента ролика fill_mc.

Метод с использованием вложенного фрагмента ролика fill_mc используется в случаях, когда требуется заполнитель, масштабирование которого не обходится без искажения внешнего вида.

Во фрагменте ролика fullness_mc компонента VolumeBar вложенный фрагмент ролика fill_mc имеет маску. Маску можно наложить либо при создании самого фрагмента ролика, либо она будет создана динамически при исполнении. Если маска представляет собой фрагмент ролика, присвойте экземпляру имя **mask_mc** и настройте его так, чтобы fill_mc отображался, как при 100 % заполнении. Если не наложить маску на fill_mc, динамически созданная маска будет прямоугольной и иметь тот же размер, что и fill_mc при 100 % заполнении.

Фрагмент ролика fill_mc проявляется с маской одним из двух способов, в зависимости от того, установлено свойство fill_mc.slideReveal на значение `true` или на значение `false`.

Если значение fill_mc.slideReveal равно `true`, то fill_mc двигается слева направо, проявляясь через маску. При 0 % он находится в крайнем левом положении и совсем не проявляется через маску. По мере роста процентной величины он сдвигается вправо, пока, достигнув 100 %, не окажется в том месте, в котором был создан в рабочей области.

Если значение fill_mc.slideReveal равно `false` или является неопределенным (поведение по умолчанию), размер маски будет изменяться слева направо, все больше проявляя fill_mc. В положении 0 % маска будет масштабирована до 05 по горизонтали, и по мере роста процентной величины значение `scaleX` будет увеличиваться, пока, при 100 %, fill_mc не проявится полностью. Не обязательно `scaleX` = 100, так как экземпляр mask_mc мог быть масштабирован при создании.

Метод без использования fill_mc проще, чем метод с использованием fill_mc, но вызывает искажение заполнителя по горизонтали. Если такое искажение нежелательно, необходимо использовать fill_mc. Фрагмент ролика progress_mc компонента SeekBar иллюстрирует такой метод.

Фрагмент ролика хода выполнения или заполненности масштабируется по горизонтали на основе процентной величины. При 0 % свойство `scaleX` экземпляра установлено на значение 0, делая его невидимым. По мере роста процентной величины значение `scaleX` регулируется, пока, достигнув 100 %, ролик не достигнет того же размера, который имел в рабочей области при создании. Опять же, не обязательно `scaleX` = 100, так как экземпляр ролика мог быть масштабирован при создании.

Соединение компонентов пользовательского интерфейса для воспроизведения FLV-файлов

Если поместить компоненты пользовательского интерфейса на одну временную шкалу и в тот же кадр, что и компонент FLVPlayback, и свойство `skin` не будет задано, FLVPlayback автоматически присоединится к ним без использования ActionScript.

Если в рабочей области расположено несколько компонентов FLVPlayback или если пользовательский элемент управления и экземпляр FLVPlayback находятся не на одной временной шкале, необходимо написать код ActionScript, чтобы присоединить компоненты пользовательского интерфейса к экземпляру компонента FLVPlayback. Сначала необходимо присвоить имя экземпляру FLVPlayback, а затем при помощи ActionScript назначить экземпляры компонентов пользовательского интерфейса для воспроизведения FLV-файлов соответствующим свойствам компонента FLVPlayback. В следующем примере экземпляр FLVPlayback имеет имя my_FLVPlybk, имена свойств FLVPlayback написаны после точек (.), а экземпляры компонентов пользовательского интерфейса стоят справа от знака "равно" (=):

```
//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;
```

Следующая процедура создает пользовательские элементы управления StopButton, PlayPauseButton, MuteButton и SeekBar:

1. Перетащите компонент FLVPlayback в рабочую область и присвойте ему имя экземпляра **my_FLVPlybk**.
2. При помощи Инспектора компонентов установите параметр `source` на значение **http://www.helpexamples.com/flash/video/cuepoints.flv**.
3. Установите параметр "Обложка" на значение "Нет".
4. Перетащите StopButton, PlayPauseButton и MuteButton в рабочую область и поместите их поверх экземпляра FLVPlayback, накладывая их вертикально слева. В Инспекторе свойств присвойте каждой кнопке имя экземпляра (например, **my_stopbttn**, **my_plypausbttn** и **my_mutebttn**).
5. На панели "Библиотека" откройте папку FLVPlayback Skins, затем откройте под ней папку SquareButton.
6. Выберите фрагмент ролика SquareBgDown и дважды щелкните его, чтобы открыть в рабочей области.
7. Щелкните его правой кнопкой мыши (в Windows) или щелкните мышью, удерживая клавишу Ctrl (в Macintosh), выберите пункт меню "Выделить все" и удалите символ.
8. Выберите инструмент "Овал", нарисуйте овал в том же месте и задайте синюю заливку (**#0033FF**).
9. В Инспекторе свойств установите ширину (W:) на **40**, а высоту (H:) на **20**. Установите координату "x" (X:) на **0.0**, а координату "y" (Y:) на **0.0**.
10. Повторите шаги 6-8 для SquareBgNormal, но измените заливку на желтую (**#FFFF00**).
11. Повторите шаги 6-8 для SquareBgOver, но измените заливку на зеленую (**#006600**).

12 Отредактируйте фрагменты роликов для различных значков символов внутри кнопок (PauseIcon, PlayIcon, MuteOnIcon, MuteOffIcon и StopIcon). Эти фрагменты роликов можно найти на панели "Библиотека" в разделе FLV Playback Skins/*Метка* Button/Assets, где *Метка* — это имя кнопки, например Play, Pause и т. д. Для каждого фрагмента ролика выполните следующие действия:

   a Выберите параметр "Выделить все".

   b Измените цвет на красный (**#FF0000**).

   c Установите масштаб на 300 %.

   d Измените местоположение X: содержимого на **7.0**, чтобы изменить горизонтальное расположение значка для каждого состояния кнопки.

   Примечание. *Такой способ изменения местоположения позволяет избежать необходимости открывать каждое состояние кнопки и перемещать экземпляр фрагмента ролика значка.*

13 Нажмите синюю стрелку "Назад" над временной шкалой, чтобы вернуться к Монтажному кадру 1, Кадру 1.

14 Перетащите компонент SeekBar в рабочую область и поместите его в правом нижнем углу экземпляра FLVPlayback.

15 На панели "Библиотека" дважды щелкните компонент SeekBar, чтобы открыть его в рабочей области.

16 Установите масштаб на 400 %.

17 Выберите контур и задайте красный цвет (**#FF0000**).

18 Дважды щелкните SeekBarProgress в папке FLVPlayback Skins/Seek Bar и задайте желтый цвет (**#FFFF00**).

19 Дважды щелкните SeekBarHandle в папке FLVPlayback Skins/Seek Bar и задайте красный цвет (**#FF0000**).

20 Нажмите синюю стрелку "Назад" над временной шкалой, чтобы вернуться к Монтажному кадру 1, Кадру 1.

21 Выберите экземпляр SeekBar в рабочей области и присвойте ему имя **my_seekbar**.

22 На панель "Действия" Кадра 1 временной шкалы вставьте инструкцию `import` для классов Video и назначьте имена кнопки и строки поиска соответствующим свойствам компонента FLVPlayback, как показано в следующем примере:

```
import fl.video.*;
my_FLVPlybk.stopButton = my_stopbttn;
my_FLVPlybk.playPauseButton = my_plypausbttn;
my_FLVPlybk.muteButton = my_mutebttn;
my_FLVPlybk.seekBar = my_seekbar;
```

23 Нажмите Control+Enter для тестирования ролика.

Создание новой обложки

Лучшим способом создания SWF-файла обложки является копирование одного из файлов обложки, поставляемых с Flash, и использование его в качестве отправной точки. FLA-файлы для этих обложек можно найти в папке приложения Flash в Configuration/FLVPlayback Skins/FLA/ActionScript 3.0/. Чтобы законченный SWF-файл обложки стал доступным для выбора в диалоговом окне "Выбрать обложку", поместите его в папку Configuration/FLVPlayback Skins/ActionScript 3.0 либо в папке приложения Flash, либо в локальной папке пользователя Configuration/FLVPlayback Skins/ActionScript 3.0.

Так как цвет обложки можно задать отдельно от выбора обложки, нет необходимости редактировать FLA-файл для изменения цвета. Если вы создаете обложку определенного цвета и не хотите, чтобы он был редактируемым в диалоговом окне "Выбрать обложку", установите `this.border_mc.colorMe = false;` в коде ActionScript FLA-файла обложки. Дополнительную информацию о задании цвета обложки см. в разделе «Выбор предустановленной обложки» на странице 161.

Если посмотреть на установленные FLA-файлы обложек во Flash, может показаться, что некоторые элементы в рабочей области не нужны, но многие из них вложены в слои направляющих. При интерактивном просмотре с 9-зонным масштабированием можно увидеть, что на самом деле отобразится в SWF-файле.

В последующих разделах приводятся описания более сложных настроек и изменений фрагментов роликов SeekBar, BufferingBar и VolumeBar.

Использование макета обложки

При открытии FLA-файла обложки Flash вы увидите, что фрагменты роликов обложки расположены на основной временной шкале. Эти ролики и код ActionScript, который находится в том же кадре, определяют расположение элементов управления при исполнении.

Хотя слой макета имеет такой вид, какой будет иметь обложка при исполнении, содержимое этого слоя не видимо при исполнении. Он используется только для расчета размещения элементов управления. Другие элементы управления в рабочей области используются при исполнении.

На слое макета находится местозаполнитель для компонента FLVPlayback с именем video_mc. Все другие элементы управления размещаются относительно video_mc. Если начать с одного из FLA-файлов Flash и изменить размер элементов управления, можно поправить макет, переместив эти ролики местозаполнителей.

Каждый из роликов местозаполнителей имеет свое имя экземпляра. Ролики местозаполнителей имеют имена 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 и volumeBarHandle_mc. Часть, цвет которой изменяется при выборе цвета обложки, называется border_mc.

Неважно, какой из роликов используется для элемента управления. Как правило, для кнопок используется ролик нормального состояния. В отношении других элементов управления, для удобства используется ролик для этого элемента управления. Что действительно важно, так это положение *x* (по горизонтали) и положение *y* (по вертикали), а также ширина и высота местозаполнителя.

Помимо стандартных элементов управления можно иметь сколько угодно дополнительных роликов. Единственным требованием к этим роликам является то, что для их символов в библиотеке должен быть установлен флажок "Экспорт для ActionScript" в диалоговом окне "Связывание". Пользовательские ролики в слое макета могут иметь любое имя экземпляра, кроме зарезервированных имен, перечисленных выше. Имя экземпляра требуется только для сценария ActionScript роликов при определении макета.

Ролик border_mc особенный. Если установить свойство `FlvPlayback.skinAutoHide` на значение `true`, обложка отображается при наведении указателя мыши на ролик border_mc. Это важно для обложек, которые отображаются за границами видеопроигрывателя. Информацию о свойстве `skinAutoHide` см. в разделе «Изменение поведения обложек» на странице 173.

В FLA-файлах Flash border_mc используется для хромирования и рамки кнопок Forward и Back.

Ролик border_mc является также частью обложки, значения альфа-канала и цвета которой были изменены при помощи свойств `skinBackgroundAlpha` и `skinBackgroundColor`. Чтобы значения цвета и альфа-канала были настраиваемыми, ActionScript в FLA-файле обложки должен содержать следующую строку:

```
border_mc.colorMe = true;
```

ActionScript и макет обложки

Следующий код ActionScript, как правило, применим ко всем элементам управления. Некоторые элементы управления имеют особый сценарий ActionScript, который определяет дополнительное поведение, которое описано в разделе, посвященном конкретному элементу управления.

Исходный код ActionScript представляет собой большой раздел, в котором указаны имена классов для каждого состояния каждого компонента. Все эти имена классов приведены в файле SkinOverAll.fla. Например, для кнопок Pause и Play такой код выглядит следующим образом:

```
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";
```

Имена классов не имеют фактических внешних файлов класса; они просто указаны в диалоговом окне "Связывание" для всех фрагментов роликов в библиотеке.

В компоненте версии ActionScript 2.0 в рабочей области находились фрагменты роликов, которые на самом деле использовались при исполнении. В компоненте версии ActionScript 3.0 эти фрагменты роликов также находятся в FLA-файле, но лишь для удобства редактирования. Теперь они все содержатся в слоях направляющих и не экспортируются. Для всех активов обложки в библиотеке задан экспорт в первый кадр, и они создаются динамически при помощи, например, такого кода:

```
new fl.video.skin.PauseButtonDisabled();
```

За этим разделом следует код ActionScript, который определяет минимальную ширину и высоту обложки. Эти значения отображаются в диалоговом окне "Выбрать обложку" и используются при исполнении для предотвращения задания размера обложки меньше минимального. Если вы не хотите указывать минимальный размер, задайте неопределенное значение или значение меньше или равное нулю.

```
// 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;
```

К каждому местозаполнителю можно применить следующие свойства:

Свойство	**Описание**
`anchorLeft`	Логическое. Располагает элемент управления относительно левого края экземпляра FLVPlayback. Значением по умолчанию является `true`, но если свойство `anchorRight` установлено явным образом на значение `true`, то значением по умолчанию будет `false`.
`anchorRight`	Логическое. Располагает элемент управления относительно правого края экземпляра FLVPlayback. Значением по умолчанию является `false`.
`anchorBottom`	Логическое. Располагает элемент управления относительно нижнего края экземпляра FLVPlayback. Значением по умолчанию является `true`, но если свойство `anchorTop` установлено явным образом на значение `true`, то значением по умолчанию будет `false`.
`anchorTop`	Логическое. Располагает элемент управления относительно верхнего края экземпляра FLVPlayback. Значением по умолчанию является `false`.

Если оба свойства — `anchorLeft` и `anchorRight` — имеют значение `true`, элемент управления масштабируется по горизонтали при исполнении. Если оба свойства — `anchorTop` и `anchorBottom` — имеют значение `true`, элемент управления масштабируется по вертикали при исполнении.

Эффект применения этих свойств можно увидеть на примере их использования в обложках Flash. Только элементы управления BufferingBar и SeekBar поддаются масштабированию. Они накладываются поверх друг друга, и оба их свойства — `anchorLeft` и `anchorRight` — имеют значение `true`. Все элементы управления слева от BufferingBar и SeekBar имеют свойство `anchorLeft` установленное на `true`, а все элементы управления, расположенные справа, имеют свойство `anchorRight` установленное на `true`. Свойство `anchorBottom` всех элементов управления установлено на значение `true`.

Можно попытаться отредактировать фрагменты роликов на слое макета так, чтобы создать обложку с элементами управления, расположенными сверху, а не снизу. Для этого требуется просто переместить элементы управления вверх относительно `video_mc` и установить свойство `anchorTop` всех элементов управления на значение `true`.

Строка буферизации

Строка буферизации имеет два фрагмента роликов: bufferingBar_mc и bufferingBarFill_mc. Положение каждого ролика относительно другого ролика в рабочей области имеет значение, так как такое относительное положение сохраняется. Строка буферизации использует два отдельных ролика, так как масштабирование применяется только к bufferingBar_mc, а не к bufferingBarFill_mc.

К ролику bufferingBar_mc применяется 9-зонное масштабирование, поэтому при масштабировании границы не искажаются. Ролик bufferingBarFill_mc очень широкий, поэтому он не нуждается в масштабировании. Для него автоматически создается маска при исполнении для отображения части вытянутого ролика bufferingBar_mc. По умолчанию точные размеры маски будут сохранять равные поля слева и справа в bufferingBar_mc на основе разницы значений положений *x* (по горизонтали) роликов bufferingBar_mc и bufferingBarFill_mc. Расположение можно настроить при помощи кода ActionScript.

Если строка буферизации не нуждается в масштабировании или не использует 9-зонное масштабирование, ее можно настроить как компонент BufferingBar пользовательского интерфейса для воспроизведения FLV-файлов. Дополнительную информацию см. в разделе «Компонент BufferingBar» на странице 164.

Строка буферизации имеет дополнительное свойство:

Свойство	Описание
`fill_mc:MovieClip`	Задает имя экземпляра заполнителя строки буферизации. Значением по умолчанию является bufferingBarFill_mc.

Строка поиска и регулятор громкости

Строка поиска имеет два фрагмента роликов: seekBar_mc и seekBarProgess_mc. Положение каждого ролика относительно другого ролика в слое макета имеет значение, так как такое относительное положение сохраняется. Хотя оба ролика поддаются масштабированию, seekBarProgress_mc нельзя вложить в seekBar_mc, так как seekBar_mc использует 9-зонное масштабирование, которое плохо работает с вложенными фрагментами роликов.

К ролику seekBar_mc применяется 9-зонное масштабирование, поэтому при масштабировании границы не искажаются. Ролик seekBarProgress_mc также поддается масштабированию, но искажение присутствует. К нему не применяется 9-зонное масштабирование, потому что это заполнитель и его искажение не заметно.

Ролик seekBarProgress_mc работает без fill_mc, так же как и ролик progress_mc работает в компонентах пользовательского интерфейса для воспроизведения FLV-файлов. Другими словами, для него не создается маска и он масштабируется по горизонтали. Точные размеры seekBarProgress_mc при 100 % определяются левым и правым полями ролика seekBarProgress_mc. Эти размеры по умолчанию равны и основаны на разнице значений положений *x* (по горизонтали) роликов seekBar_mc и seekBarProgress_mc. Эти размеры можно настроить при помощи кода ActionScript во фрагменте ролика строки поиска, как показано в следующем примере:

```
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;
```

Этот код можно вставить либо на временную шкалу фрагмента ролика SeekBar, либо вместе с другим кодом ActionScript на основную временную шкалу. Если настройка выполняется при помощи кода, а не за счет изменения макета, нет необходимости помещать заполнитель в рабочую область. Достаточно того, чтобы он присутствовал в библиотеке и был настроен на экспорт для ActionScript в Кадр 1 с правильным именем класса.

Как и в случае с компонентом SeekBar пользовательского интерфейса для воспроизведения FLV-файлов, для строки поиска можно создать фрагмент ролика заполненности. Если строка поиска не нуждается в масштабировании или нуждается в нем, но не использует 9-зонное масштабирование, можно установить progress_mc или fullness_mc при помощи любого из методов, используемых для компонентов пользовательского интерфейса для воспроизведения FLV-файлов. Дополнительную информацию см. в разделе .

Так как регулятор громкости в обложках Flash не поддается масштабированию, он построен тем же образом, что и компонент VolumeBar пользовательского интерфейса для воспроизведения FLV-файлов. Дополнительную информацию см. в разделе «Компоненты SeekBar и VolumeBar» на странице 164. Исключением является маркер, который реализован по-другому.

Маркеры SeekBar и VolumeBar

Маркеры SeekBar и VolumeBar расположены в слое макета рядом со строкой. По умолчанию левое поле, правое поле и положение по оси *y* маркера задаются на основе его положения относительно фрагмента ролика строки. Левое поле определяется разницей между положением *x* (по горизонтали) маркера и положением *x* (по горизонтали) строки, а правое поле равно левому полю. Эти значения можно настроить при помощи ActionScript во фрагменте ролика SeekBar или VolumeBar. В следующем примере приведен тот же код ActionScript, который используется с компонентами пользовательского интерфейса для воспроизведения FLV-файлов:

```
this.seekBar_mc.handleLeftMargin = 2;
this.seekBar_mc.handleRightMargin = 2;
this.seekBar_mc.handleY = 11;
```

Этот код можно вставить либо на временную шкалу фрагмента ролика SeekBar, либо вместе с другим кодом ActionScript на основную временную шкалу. Если настройка выполняется при помощи кода, а не за счет изменения макета, нет необходимости помещать маркер в рабочую область. Достаточно того, чтобы он присутствовал в библиотеке и был настроен на экспорт для ActionScript в Кадр 1 с правильным именем класса.

Не взирая на эти свойства маркеры являются простыми фрагментами роликов, устроенными таким же образом, что и маркеры в компонентах пользовательского интерфейса для воспроизведения FLV-файлов. Оба имеют фоновые прямоугольники со свойством `alpha` установленным на 0. Они служат для увеличения области попадания и не являются обязательными.

Ролики фона и основного цвета

Фрагменты роликов chrome_mc и forwardBackBorder_mc реализованы как ролики фона.

Из фрагментов роликов ForwardBackBorder, ForwardBorder и BackBorder в рабочей области и кнопок-местозаполнителей Forward и Back единственным роликом, *не* расположенным в слое направляющих, является ForwardBackBorder. Фактически кнопки Forward и Back используются только в обложках.

Единственным требованием к этим роликам является то, что для них должен быть задан экспорт для ActionScript в Кадр 1 в библиотеке.

Изменение поведения обложек

Свойство `bufferingBarHidesAndDisablesOthers` и свойство `skinAutoHide` позволяют настроить поведение обложки компонента FLVPlayback.

Установка свойства `bufferingBarHidesAndDisablesOthers` на значение `true` заставляет компонент FLVPlayback скрыть SeekBar и его маркер, а также отключить кнопки воспроизведения и паузы, когда компонент входит в состояние буферизации. Это полезно при потоковой передаче FLV-файла с FMS при медленном подключении с высоким значением свойства `bufferTime` (10, например). В такой ситуации нетерпеливый пользователь может начать нажимать на кнопки воспроизведения и паузы, еще больше замедляя воспроизведение файла. Такие действия можно предотвратить, установив `bufferingBarHidesAndDisablesOthers` на значение `true` и отключив элемент SeekBar, а также кнопки воспроизведения и паузы на время пребывания компонента в режиме буферизации.

Свойство `skinAutoHide` влияет только на SWF-файлы предустановленных обложек, а не на элементы управления, созданные из компонентов пользовательского интерфейса для воспроизведения FLV-файлов. Если это свойство имеет значение `true`, компонент FLVPlayback скрывает обложку, когда указатель мыши не находится в пределах области просмотра. Значением по умолчанию этого свойства является `false`.

Использование SMIL-файла

Для управления несколькими потоками для нескольких полос пропускания класс VideoPlayer использует вспомогательный класс (NCManager), который поддерживает подмножество SMIL. SMIL-файл используется для определения местоположения видеопотока, макета (ширины и высоты) FLV-файла и исходных FLV-файлов, соответствующих различным полосам пропускания. Он также может использоваться для указания скорости потока и длительности FLV-файла.

Используйте параметр `source` или свойство FLVPlayback.source (ActionScript) для указания местоположения SMIL-файла. Дополнительную информацию см. в разделе и в описании свойства `FLVPlayback.source` в документе *Справочник по языку ActionScript 3.0 и компонентам*.

Следующий пример иллюстрирует SMIL-файл для потоковой передачи FLV-файлов с разными полосами пропускания с сервера FMS с использованием RTMP:

```
<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>
```

Тег `<head>` может содержать теги `<meta>` и `<layout>`. Тег `<meta>` поддерживает только атрибут `base`, который используется для указания URL-адреса потокового видео (RTMP с FMS).

Тег `<layout>` поддерживает только элемент `root-layout`, который используется для задания атрибутов `height` и `width`, определяя размер окна, в котором отображается FLV-файл. Для этих атрибутов допустимы только значения в пикселах, а не в процентах.

В тело SMIL-файла можно либо включить одну ссылку на исходный FLV-файл, либо, при потоковой передаче нескольких файлов с разными полосами пропускания с FMS (как в предыдущем примере), использовать тег `<switch>` для перечисления исходных файлов.

Теги `video` и `ref` в теге `<switch>` являются синонимами — они оба могут использовать атрибут `src` для указания на FLV-файлы. Более того, каждый из них может использовать атрибуты `region`, `system-bitrate` и `dur` для указания региона, минимальной требуемой пропускной способности и длительности FLV-файла.

В теге `<body>` допустимо только единичное появление одного из следующих тегов: `<video>`, `<src>` или `<switch>`.

Следующий пример иллюстрирует последовательную загрузку одного FLV-файла без определения полосы пропускания:

```
<smil>
        <head>
            <layout>
                <root-layout width="240" height="180" />
            </layout>
        </head>
        <body>
            <video src=""myvideo.flv" />
        </body>
</smil>
```

<smil>

Доступность
Flash Professional 8.

Применение

```
<smil>
...
child tags
...
</smil>
```

Атрибуты

Нет.

Дочерние теги

```
<head>, <body>
```

Родительский тег

Нет.

Описание

Тег верхнего уровня, который идентифицирует SMIL-файл.

Пример

В следующем примере показан SMIL-файл, указывающий три FLV-файла:

```
<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>

Доступность

Flash Professional 8.

Применение

```
<head>
...
child tags
...
</head>
```

Атрибуты
Нет.

Дочерние теги
`<meta>, <layout>`

Родительский тег
`<smil>`

Описание
Поддерживая теги `<meta>` и `<layout>`, указывает местоположение и макет по умолчанию (высоту и ширину) исходных FLV-файлов.

Пример
В следующем примере указан корневой макет размером 240 на 180 пикселов:

```
<head>
    <meta base="rtmp://myserver/myapp/" />
    <layout>
        <root-layout width="240" height="180" />
    </layout>
</head>
```

<meta>

Доступность
Flash Professional 8.

Применение
`<meta/>`

Атрибуты
`base`

Дочерние теги
`<layout>`

Родительский тег
Нет.

Описание
Содержит атрибут `base`, который указывает местоположение (URL-адрес RTMP) исходных FLV-файлов.

Пример
Следующий пример иллюстрирует тег meta для указания местоположения base на `myserver`:

```
<meta base="rtmp://myserver/myapp/" />
```

<layout>

Доступность
Flash Professional 8.

Применение
```
<layout>
...
child tags
...
</layout>
```

Атрибуты
Нет.

Дочерние теги
`<root-layout>`

Родительский тег
`<meta>`

Описание
Указывает ширину и высоту FLV-файла.

Пример
В следующем примере указан макет размером 240 на 180 пикселов:

```
<layout>
    <root-layout width="240" height="180" />
</layout>
```

<root-layout>

Доступность
Flash Professional 8.

Применение
```
<root-layout...attributes.../>
```

Атрибуты
Ширина, высота

Дочерние теги
Нет.

Родительский тег
`<layout>`

Описание
Указывает ширину и высоту FLV-файла.

Пример
В следующем примере указан макет размером 240 на 180 пикселов:

```
<root-layout width="240" height="180" />
```

<body>

Доступность
Flash Professional 8.

Применение

```
<body>
...
child tags
...
</body>
```

Атрибуты
Нет.

Дочерние теги
`<video>`, `<ref>`, `<switch>`

Родительский тег
`<smil>`

Описание
Содержит теги `<video>`, `<ref>` и `<switch>`, указывающие имя исходного FLV-файла, минимальную пропускную способность и длительность FLV-файла. Атрибут `system-bitrate` поддерживается только при использовании тега `<switch>`. В теге `<body>` допускается присутствие только одного экземпляра тега `<switch>`, `<video>` или `<ref>`.

Пример
Следующий пример указывает три FLV-файла, два из которых используют тег `video`, а один — тег `ref`:

```
<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>

Доступность
Flash Professional 8.

Применение
`<video...attributes.../>`

Атрибуты
`src, system-bitrate, dur`

Дочерние теги
Нет.

Родительский тег
`<body>`

Описание
Синоним тега `<ref>`. Поддерживает атрибуты `src` и `dur` , указывающие имя исходного FLV-файла и его длительность. Атрибут `dur` поддерживает полный (00:03:00:01) и сокращенный (03:00:01) форматы времени.

Пример
Следующий пример задает источник и длительность видеофайла:

```
<video src="myvideo_mdm.flv" dur="3:00.1"/>
```

<ref>

Доступность
Flash Professional 8.

Применение
`<ref...attributes.../>`

Атрибуты
`src, system-bitrate, dur`

Дочерние теги
Нет.

Родительский тег
`<body>`

Описание
Синоним тега `<video>`. Поддерживает атрибуты `src` и `dur` , указывающие имя исходного FLV-файла и его длительность. Атрибут `dur` поддерживает полный (00:03:00:01) и сокращенный (03:00:01) форматы времени.

Пример

Следующий пример задает источник и длительность видеофайла:

```
<ref src="myvideo_cable.flv" dur="3:00.1"/>
```

<switch>

Доступность

Flash Professional 8.

Применение

```
<switch>
...
child tags
...
<switch/>
```

Атрибуты

Нет.

Дочерние теги

`<video>, <ref>`

Родительский тег

`<body>`

Описание

Используется с дочерним тегом `<video>` или `<ref>` для вывода списка FLV-файлов для потоковой передачи с несколькими полосами пропускания. Тег `<switch>` поддерживает атрибут `system-bitrate`, который указывает минимальную пропускную способность, а также атрибуты `src` и `dur`.

Пример

Следующий пример указывает три FLV-файла, два из которых используют тег `video`, а один — тег `ref`:

```
<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>
```

Глава 7. Использование компонента FLVPlaybackCaptioning

Компонент FLVPlayback позволяет включать видеопроигрыватель в приложение Adobe Flash CS4 Professional для воспроизведения загруженных и потоковых файлов Adobe Flash Video (FLV). Дополнительные сведения о FLVPlayback, см. в разделе «Использование компонента FLVPlayback» на странице 141.

Компонент FLVPlaybackCaptioning позволяет включить поддержку встроенных субтитров для видеосодержимого. Компонент субтитров поддерживает синхронизированный текст (Timed Text) в формате XML стандарта W3C и включает следующие функции.

Субтитры с встроенными ключевыми точками событий Связывайте встроенные ключевые точки событий в файле FLV с XML, чтобы обеспечить субтитры вместо использования файла Timed Text XML.

Несколько FLVPlaybackCaptioning Создавайте несколько экземпляров FLVPlaybackCaptioning для разных экземпляров FLVPlayback.

Управление с помощью кнопки-переключателя Предоставьте пользователю взаимодействовать с субтитрами в помощью переключателя.

Использование компонента FLVPlaybackCaptioning

Компонент FLVPlaybackCaptioning используется с одним или несколькими компонентами FLVPlayback. Проще всего перетащить компонент FLVPlayback в рабочую область, туда же перетащить компонент FLVPlaybackCaptioning, указать URL-адрес субтитров и запустить их отображение. Кроме того, компонент FLVPlaybackCaptioning можно настроить, задавая различные параметры.

Добавление субтитров в компонент FLVPlayback

Компонент FLVPlaybackCaptioning можно добавить в любой компонент FLVPlayback. Сведения о добавлении компонентов FLVPlayback в приложение см. в разделе «Создание приложения с компонентом FLVPlayback» на странице 143.

Добавление компонента FLVPlaybackCaptioning на панель "Компоненты":

1. На панели "Компоненты" откройте папку "Видео".
2. Перетащите (или дважды щелкните) компонент FLVPlaybackCaptioning и добавьте его в ту же рабочую область, где находится компонент FLVPlayback, в который нужно добавить субтитры.

   Примечание. *Компания Adobe предоставляет два файла для быстрого ознакомления с компонентом FLVPlaybackCaptioning: caption_video.flv (образец FLVPlayback) и caption_video.xml (образец субтитров). Эти файлы можно найти по адресу www.helpexamples.com/flash/video/caption_video.flv и www.helpexamples.com/flash/video/caption_video.xml.*
3. (Необязательно) Перетащите компонент CaptionButton в ту же рабочую область, где находятся компоненты FLVPlayback и FLVPlaybackCaptioning. Компонент CaptionButton дает пользователю возможность включать и выключать субтитры.

Примечание. *Чтобы включить компонент CaptionButton, его нужно перетащить в ту же рабочую область, где находятся компоненты FLVPlayback и FLVPlaybackCaptioning.*

4 Выделите компонент FLVPlaybackCaptioning в рабочей области и на вкладке "Параметры" Инспектора свойств задайте следующие значения.

- Задайте свойству `showCaptions` значение `true`.
- Задайте свойство `source` файла Timed Text XML для загрузки.

Чтобы протестировать субтитры в Flash, необходимо задать свойству `showCaptions` значение `true`. Однако, если добавлен компонент `CaptionButton` для включения или выключения субтитров, свойству `showCaptions` необходимо задать значение `false`.

Имеются и другие параметры для настройки компонента FLVPlaybackCaptioning. Дополнительную информацию см. в разделе «Настройка компонента FLVPlaybackCaptioning» на странице 192 и в *Справочнике по языку ActionScript 3.0 и компонентам*.

5 Выберите "Управление" > "Тестировать ролик", чтобы запустить видео.

Динамическое создание экземпляра с помощью ActionScript:

1 Перетащите компонент FLVPlayback с панели "Компоненты" на панель "Библиотека" ("Окна" > "Библиотека").

2 Перетащите компонент FLVPlaybackCaptioning с панели "Компоненты" на панель "Библиотека".

3 Добавьте следующий код на панель "Действия" в кадре 1 временной шкалы.

```
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
CS4/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 Измените *install_drive*, указав диск, на котором установлено ПО Flash, и укажите путь к папке "Обложки" для установки

Примечание. *Если экземпляр FLVPlayback создается с помощью ActionScript, назначать ему обложку нужно тоже динамически, задав свойство skin с помощью ActionScript. Когда обложка применяется с помощью ActionScript, она не публикуется с SWF-файлом автоматически. Скопируйте SWF-файл обложки и SWF-файл приложения на сервер, иначе SWF-файл обложки не будет доступным при выполнении приложения пользователем.*

Настройка параметров компонента FLVPlaybackCaptioning

В Инспекторе свойств и Инспекторе компонентов для каждого экземпляра компонента FLVPlaybackCaptioning можно дополнительно настроить следующие параметры. В следующем списке приводятся параметры и их краткое описание.

`autoLayout` Определяет, управляет ли компонент FLVPlaybackCaptioning размером области субтитров. Значение по умолчанию — `true`.

`captionTargetName` Определяет имя экземпляра TextField или MovieClip, содержащего субтитры. Значение по умолчанию — auto.

`flvPlaybackName` Определяет имя экземпляра FLVPlayback, к которому нужно добавить субтитры. Значение по умолчанию — auto.

`simpleFormatting` Ограничивает инструкции форматирования из файла Timed Text XML, если установлено значение true. Значение по умолчанию равно `false`.

`showCaptions` Определяет, отображаются ли субтитры. Значение по умолчанию — `true`.

`source` Определяет местоположение файла Timed Text XML.

Дополнительные сведения по всем параметрам компонента FLVPlaybackCaptioning см. в *Справочнике по языку ActionScript 3.0 и компонентам*.

Указание параметра source

Используйте параметр `source`, чтобы указать имя и местоположение файла Timed Text XML, содержащего субтитры для ролика. Введите URL-адрес прямо в ячейке source в Инспекторе компонентов.

Показ субтитров

Чтобы показать субтитры, задайте свойству `showCaptions` значение `true`.

Дополнительные сведения по всем параметрам компонента FLVPlaybackCaptioning см. в *Справочнике по языку ActionScript 3.0 и компонентам*.

В предыдущих примерах описывались процедуры создания и включения компонента FLVPlaybackCaptioning для показа субтитров. Для субтитров можно использовать два источника: (1) файл Timed Text XML с субтитрами или (2) файл XML с текстом субтитров, который нужно связать с встроенными ключевыми точками событий.

Использование субтитров в формате Timed Text

Компонент FLVPlaybackCaptioning включает субтитры для связанного компонента FLVPlayback путем загрузки файла Timed Text (TT) XML. Дополнительную информацию о формате синхронизированного текста Timed Text см. в разделе AudioVideo Timed Text (Синхронизированный текст для аудио и видео) на сайте www.w3.org.

В этом разделе приводится общая информация о поддерживаемых тегах синхронизированного текста, об обязательных тегов файла с субтитрами и пример файла Timed Text XML. Подробные сведения обо всех поддерживаемых тегах см. в разделе «Теги синхронизированного текста» на странице 185.

Компонент FLVPlaybackCaptioning поддерживает следующие теги синхронизированного текста.

Категория	**Задача**
Поддержка форматирования абзаца	Выравнивайте абзацы по центру, по правому или левому краю.
Поддержка форматирования текста	• Задавайте размер текста в виде абсолютных или относительных пиксельных размеров (например, +2, -4). • Задавайте цвет и шрифт текста. • Сделайте текст полужирным или курсивом. • Задавайте выравнивание.
Поддержка других возможностей форматирования	• Задавайте цвет фона объекта TextField для субтитров. • Задавайте прозрачный цвет фона (альфа-значение 0) объекта TextField для субтитров. • Включайте или выключайте перенос по словам для объекта TextField.

Компонент FLVPlaybackCaptioning согласуется с временным кодом файла FLV. Каждый субтитр должен иметь атрибут `begin`, который определяет время появления субтитра. Если у субтитра нет атрибута `dur` или `end`, он исчезает при появлении следующего субтитра или после окончания файла FLV.

Ниже приводится пример файла Timed Text XML. В этом файле (caption_video.xml) содержатся субтитры для файла caption_video.flv. Эти файлы можно найти по адресу www.helpexamples.com/flash/video/.

```
<?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>
```

Теги синхронизированного текста

Компонент FLVPlaybackCaptioning поддерживает теги синхронизированного текста для файлов с субтитрами в формате XML. Дополнительную информацию о тегах синхронизированного текста для аудио и видео см. на странице www.w3.org. В следующей таблице перечислены поддерживаемые и неподдерживаемые теги.

Функция	Тег/Значение	Применение/Описание	Пример
Игнорируемые теги	metadata	Игнорируется / допускается на любом уровне документа.	
	set	Игнорируется / допускается на любом уровне документа.	
	xml:lang	Игнорируется.	
	xml:space	Игнорируется / поведение переопределяется на: xml:space="default".	

Функция	**Тег/Значение**	**Применение/Описание**	**Пример**
	layout	Игнорируется / включая все теги области в разделе тега layout.	
	тег br	Игнорируются все атрибуты и содержимое.	
Синхронизация субтитров с мультимедийным содержимым	Атрибуты begin	Допускаются только в тегах p. Обязательные для развертывания субтитров синхронно с мультимедийным содержимым.	<p begin="3s">
	Атрибуты dur	Допускаются только в тегах p. Рекомендуемые. Если их нет, субтитр заканчивается вместе с файлом FLV или в момент начала другого субтитра.	
	Атрибуты end	Допускаются только в тегах p. Рекомендуемые. Если их нет, субтитр заканчивается вместе с файлом FLV или в момент начала другого субтитра.	
Синхронизация субтитров по времени	00:03:00.1	Полный формат времени	
	03:00.1	Неполный формат времени	
	10	Смещение по времени без единиц. Смещение выражено в секундах.	
	00:03:00:05 00:03:00:05.1 30f 30t	Не поддерживается. Форматы времени, включающие кадры или такты не поддерживаются.	
Тег тела	body	Обязательный / Поддерживается только один тег body.	<body> ... </body>
Тег содержимого	Тег div	Допускаются значения равные или больше нуля. Используется первый тег.	
	Тег p	Допускаются значения равные или больше нуля..	
	Тег span	Логический контейнер для последовательности единиц текстового содержимого. Вложенные диапазоны не поддерживаются. Поддерживаются теги стиля атрибутов.	
	Тег br	Обозначает выраженный разрыв строки.	
Теги стиля (Все теги стиля используются внутри тега p.)	style	Ссылается на один или несколько элементов стиля. Может использоваться в качестве тега или атрибута. В качестве тега ему необходим атрибут ID (стиль может повторно использоваться в документе). В теге style поддерживается один и более тегов style.	

Функция	Тег/Значение	Применение/Описание	Пример
	tts:background Color	Задает свойство стиля, определяющее цвет фона для области. Альфа-канал игнорируется, если ему не задано значение 0 (alpha 0) для получения прозрачного фона. Используется формат цвета #RRGGBBAA.	
	tts:color	Задает свойство стиля, определяющее цвет переднего плана. Альфа-канал для цветов не поддерживается. Значение `transparent` преобразуется в черный цвет.	<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	Задает свойство стиля, определяющее семейство шрифтов.	"default" = `_serif` "monospace" = `_typewriter` "sansSerif" = `_sans` "serif" = `_serif` "monospaceSansSerif" =`_typewriter` "monospaceSerif" =`_typewriter` "proportionalSansSerif" = `_sans`
	tts:fontSize	Задает свойство стиля, определяющее размер шрифта. Если заданы два значения, используется только первое (вертикальный размер). Проценты и единицы игнорируются. Поддерживаются абсолютные пиксельные (например, 12) и относительные (например, +2) размеры.	

Функция	Тег/Значение	Применение/Описание	Пример
	tts:fontStyle	Задает свойство стиля, определяющее стиль шрифта.	"normal" "italic" "inherit"* * Поведение по умолчанию; наследует стиль содержащего тега.
	tts:fontWeight	Задает свойство стиля, определяющее толщину шрифта.	"normal" "bold" "inherit"* * Поведение по умолчанию; наследует стиль содержащего тега.

Функция	Тег/Значение	Применение/Описание	Пример
	tts:textAlign	Задает свойство стиля, определяющее выравнивание встроенных областей внутри содержащего их блока.	"left" "right" "center" "start" (="left") "end" (="right") "inherit"* *Наследует стиль содержащего тега. Если тег textAlign не задан, по умолчанию используется "left".
	tts:wrapOption	Задает свойство стиля, определяющее, применяется ли автоматический перенос строки (разрыв) в контексте затронутого элемента. Этот параметр применяется ко всем абзацам в элементе субтитров.	"wrap" "noWrap" "inherit"* *Наследует стиль содержащего тега. Если тег wrapOption не задан, по умолчанию используется "wrap".
Неподдерживаемые атрибуты	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		

Использование ключевых точек с субтитрами

Ключевые точки позволяют взаимодействовать с видео. Например, можно воздействовать на воспроизведение FLV-файла или отображать текст в определенные моменты воспроизведения видео. Если для FLV-файла нет связанного файла Timed Text XML, в него можно встроить ключевые точки событий и связать их с текстом. В этом разделе описаны стандарты ключевых точек компонента FLVPlaybackCaptioning и процедура их связывания с текстом субтитров. Дополнительные сведения о встраивании ключевых точек событий с помощью мастера импорта видео или кодировщика Flash Video см. в главе 16, "Работа с видео", руководства *Использование Flash*.

Сведения о стандартах ключевых точек FLVPlaybackCaptioning

В метаданных FLV-файла ключевая точка представлена в виде объекта со следующими свойствами: `name`, `time`, `type` и `parameters`. Ключевые точки FLVPlaybackCaptioning ActionScript имеют следующие атрибуты.

`name` Свойство `name` представляет собой строку, содержащую имя, присвоенное ключевой точке. Свойство `name` должно начинаться с префикса *fl.video.caption.2.0.*, за которым следует строка имени. Строка — это последовательность положительных целых чисел, увеличивающихся для каждой следующей точки, чтобы обеспечить уникальность имен. Префикс включает номер версии, соответствующий номеру версии FLVPlayback. Для Adobe Flash CS4 необходимо задать номер версии `2.0`.

`time` Свойство `time` представляет время, в которое должен появиться субтитр.

`type` Свойство `type` представляет собой строку со значением `"event"`.

`parameters` Свойство `parameters` представляет собой массив, поддерживающий следующие пары "имя+значение".

- **`text:String`** Текст субтитра с HTML-форматированием. Этот текст передается непосредственно свойству `TextField.htmlText`. Компонент FLVPlaybackCaptioning поддерживает дополнительное свойство `text:`*n*, которое обеспечивает возможность использования нескольких языковых треков. Дополнительные сведения см. в разделе «Поддержка нескольких языковых треков для встроенных ключевых точек» на странице 192.

- **`endTime:Number`** Время, когда субтитр должен исчезнуть. Если не задать это свойство, компонент FLVPlaybackCaptioning предположит, что это не числовое значение (NaN), и субтитр будет отображаться до окончания воспроизведения FLV-файла (когда экземпляр FLVPlayback отправляет событие `VideoEvent.COMPLETE)`. Свойство `endTime:Number` задается в секундах.

- **`backgroundColor:uint`** Этот параметр задает свойство `TextField.backgroundColor`. Он является необязательным.

- **`backgroundColorAlpha:Boolean`** Если backgroundColor имеет альфа-значение 0 %, то параметр задает свойство `TextField.background = !backgroundColor`. Он является необязательным.

- **`wrapOption:Boolean`** Этот параметр задает свойство TextField.wordWrap. Он является необязательным.

Сведения о субтитрах для встроенных ключевых точках событий

Если для FLV-файла отсутствует связанный файл Timed Text XML с субтитрами, можно связать XML-файл, содержащий субтитры, со встроенными ключевыми точками событий. Образец XML предполагает, что в видео уже встроены ключевые точки событий. Для этого необходимо выполнить следующие действия.

- Добавьте ключевые точки событий (в соответствии со стандартами FLVPlaybackCaptioning) и закодируйте видео.
- В ПО Flash перетащите компоненты FLVPlayback и FLVPlaybackCaptioning в рабочую область.
- Задайте свойство source для компонентов FLVPlayback и FLVPlaybackCaptioning (укажите местоположение FLV-файла и XML-файла).
- Опубликуйте файлы.

  Следующий код импортирует XML в кодировщик:

```
<?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>
```

Компонент FLVPlaybackCaptioning также поддерживает использование нескольких языковых треков для встроенной точки событий. Дополнительные сведения см. в разделе «Поддержка нескольких языковых треков для встроенных ключевых точек» на странице 192.

Поддержка нескольких языковых треков для встроенных ключевых точек

Свойство `track` компонента FLVPlaybackCaptioning поддерживает несколько языковых треков для встроенных ключевых точек, если файл Timed Text соответствует стандартам ключевых точек FLVPlaybackCaptioning. (Дополнительные сведения см. в разделе «Сведения о стандартах ключевых точек FLVPlaybackCaptioning» на странице 190.) Однако компонент FLVPlaybackCaptioning не поддерживает использование нескольких языковых треков в отдельных XML-файлах. Чтобы использовать свойство `track`, задайте для него значение, не равное 0. Например, если задать свойству track значение 1(`track == 1`), то компонент FLVPlaybackCaptioning будет искать параметры ключевых точек. Если соответствия не найдено, используется свойство text в параметрах ключевой точки. Дополнительные сведения см. в описании свойства `track` в *Справочнике по языку ActionScript 3.0 и компонентам*.

Воспроизведение нескольких FLV-файлов с субтитрами

Можно открыть несколько видеопроигрывателей в одном экземпляре компонента FLVPlayback, чтобы открыть несколько видеофайлов и переключаться между ними в процессе воспроизведения. Также можно связать субтитры с каждым видеопроигрывателем в компоненте FLVPlayback. Дополнительные сведения о том, как открыть несколько видеопроигрывателей, см в разделе «Использование нескольких видеопроигрывателей» на странице 158. Чтобы использовать субтитры в нескольких видеопроигрывателях, создайте один экземпляр компонента FLVPlaybackCaptioning для каждого экземпляра `VideoPlayer` и задайте свойству `videoPlayerIndex` FLVPlaybackCaptioning соответствующий индекс. Компонент `VideoPlayer` по умолчанию имеет индекс 0, когда открыт только один экземпляр `VideoPlayer`.

Следующий код назначает уникальные субтитры уникальным видеофайлам. В процессе выполнения вымышленные URL в примере необходимо заменить на существующие адреса.

```
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");
```

Настройка компонента FLVPlaybackCaptioning

Чтобы быстро начать работу с компонентом FLVPlaybackCaptioning, можно использовать параметры FLVPlaybackCaptioning по умолчанию, при которых субтитры накладываются прямо поверх компонента FLVPlayback. Однако можно настроить компонент FLVPlaybackCaptioning так, чтобы субтитры не закрывали видео.

Следующий код динамически создает объект FLVPlayback с кнопкой переключения субтитров:

1. Поместите компонент FLVPlayback на рабочий стол в положение 0,0 и задайте экземпляру имя **player**.
2. Поместите компонент FLVPlaybackCaptioning на рабочий стол в позицию 0,0 и задайте экземпляру имя **captioning**.

3 Поместите в рабочую область компонент CaptionButton.

4 В следующем примере задайте FLV-файл в качестве значения переменной `testVideoPath:String` (указав абсолютный или относительный путь).

Примечание. *В этом примере переменной `testVideoPath` присваивается образец видеофайла Flash, `caption_video.flv`. Измените значение переменной, указав путь к компоненту субтитров, для которого добавляется компонент CaptionButton.*

5 В следующем примере задайте нужный файл Timed Text XML в качестве значения переменной `testCaptioningPath:String` (указав абсолютный или относительный путь).

Примечание. *В этом примере переменной `testCaptioningPath` присваивается образец файла Timed Text XML, `caption_video.xml`. Измените значение переменной, указав путь к файлу Timed Text XML, содержащему субтитры для видео.*

6 Сохраните следующий код под именем FLVPlaybackCaptioningExample.as в том же каталоге, где и FLA-файл.

7 Присвойте классу Document в FLA-файле значение 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;
        }
    }
}
```

Дополнительные сведения по всем параметрам компонента FLVPlaybackCaptioning см. в *Справочнике по языку ActionScript 3.0 и компонентам*.

Указатель

И

К

M

Я