

Дмитрий Альберт
Елена Альберт



ActionScript 2.0

- Детальное описание синтаксиса и возможностей языка
- Динамические эффекты и интерактивная анимация
- Объектно-ориентированное программирование в стиле ActionScript 2.0

**Наиболее
полное
руководство**

+CD



В ПОДЛИННИКЕ®

УДК 681.3.06
ББК 32.973.26-018.1
А56

Альберт Д. И., Альберт Е. Э.

А56 ActionScript 2.0. — СПб.: БХВ-Петербург, 2005. — 1136 с.: ил.
ISBN 5-94157-567-X

Книга является подробным руководством по языку сценариев ActionScript 2.0 для среды Macromedia Flash MX 2004. Описаны основные возможности, синтаксис и семантика языка, его базовые элементы: переменные, типы данных, циклы, условные операторы, массивы, функции. Рассмотрены событийная модель и процедура трансляции событий, встроенные классы и технология объектно-ориентированного программирования в стиле ActionScript 2.0. Изложены основанный на этой технологии подход к трехмерному моделированию и процедура создания пользовательских компонентов.

Книга написана ясным и логичным языком. Материал организован так, чтобы ее можно было использовать и как обучающее руководство, и как справочное пособие. В начале каждой темы приводятся теоретические сведения, которые затем иллюстрируются конкретными практическими примерами. В книгу включено более 100 примеров, что позволит читателю приобрести практические навыки работы с языком. Все исходные файлы примеров помещены на сопровождающий компакт-диск.

Для программистов

УДК 681.3.06
ББК 32.973.26-018.1

Группа подготовки издания:

Главный редактор	<i>Екатерина Кондукова</i>
Зам. главного редактора	<i>Евгений Рыбаков</i>
Зав. редакцией	<i>Григорий Добин</i>
Редактор	<i>Татьяна Лапина</i>
Компьютерная верстка	<i>Натальи Смирновой</i>
Корректор	<i>Наталья Першакова</i>
Дизайн обложки	<i>Игоря Цырульниково</i>
Зав. производством	<i>Николай Тверских</i>

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 17.06.05.

Формат 70×100^{1/16}. Печать офсетная. Усл. печ. л. 91,59.

Тираж 3000 экз. Заказ №

"БХВ-Петербург", 194354, Санкт-Петербург, ул. Есенина, 5Б.

Санитарно-эпидемиологическое заключение на продукцию № 77.99.02.953 Д.006421.11.04 от 11.11.2004 г. выдано Федеральной службой по надзору в сфере защиты прав потребителей и благополучия человека.

Отпечатано с готовых диапозитивов
в ОАО "Техническая книга"
190005, Санкт-Петербург, Измайловский пр., 29.

ISBN 5-94157-567-X

© Альберт Д. И., Альберт Е. Э., 2005
© Оформление, издательство "БХВ-Петербург", 2005

Оглавление

Введение	1
Часть I. Основы ACTIONSCRIPT	3
Глава 1. Представление о Flash	5
Компоненты программы Flash MX 2004 Professional	5
Процесс создания Flash-проекта	6
Системные требования программы Flash MX 2004 Professional	7
Обзор интерфейса	8
Основы работы с векторным редактором Flash MX 2004	15
Понятие о векторной графике	15
Иерархия графических объектов Flash	17
Операции над объектами и трансформация объектов	21
Настройка глобальных параметров фильма	23
Работа с цветом	24
Работа с растровой графикой	26
Структура монтажной линейки	28
Работа со слоями	29
Работа с кадрами	31
Символы и экземпляры	35
Типы символов и операции с ними	36
Экземпляр символа и его свойства	38
Работа с библиотекой	40
Работа с общими библиотеками (Shared Libraries)	41
Разрешение вопросов политики безопасности при импорте внешних данных из других доменов	44
Анимация	45
Покадровая анимация (Frame by frame animation)	47
Анимация формы (Shape tweening)	47
Анимация движения (Motion tweening)	49
Использование звука в фильме	50
Публикация исходного документа	54
Формат Flash (SWF)	56
Формат HTML	59
Формат Windows Projector (exe)	64
Глава 2. Новые возможности ActionScript	65
ActionScript 2.0	65
Зависимость от регистра	65

Строгий контроль типа данных	66
Формальный синтаксис классов	66
Новая архитектура компонентов	67
Массивы	67
Работа с клипами	68
Связывание символа с классом	68
Компилированные клипы	68
Контроль заполнения уровней стека	68
Свойство <i>_lockroot</i>	69
Метод <i>getSWFVersion()</i>	69
Работа с текстом	69
Поддержка CSS (Cascading Style Sheets)	69
Возможность загрузки изображений и внешних SWF-документов в текстовый блок	70
Прокрутка текста с помощью колеса мыши	70
Контроль статического текста	70
Класс <i>MovieClipLoader</i> . Организация процесса загрузки	71
Класс <i>Sound</i> . Поддержка ID3-тегов второй версии	71
Динамическая загрузка видео	72
Управление контекстным меню проигрывателя	72
Объект <i>System.capabilities</i>	73
Печать из фильма	73
Политика безопасности	74
Глава 3. Введение в ActionScript. Среда разработки	75
Представление о сценариях ActionScript	75
Использование редактора ActionScript для создания сценариев	76
Раздел <i>Script pane</i> (Окно сценария)	78
Раздел <i>Actions toolbox</i>	87
Раздел <i>Script navigator</i>	89
Носители сценариев AS	89
Синтаксические нормы и правила	92
Предложения	92
Комментарии	93
Имена	94
Функции	95
Функция <i>trace()</i>	96
Инициирование и последовательность выполнения сценариев.	
Системные и пользовательские события	97
Последовательность выполнения сценариев кадров	98
Сценарии кнопок и клипов	99
Представление об объектно-ориентированном программировании	101
Создание сценария во внешнем файле	103
Применение директивы <i>#include</i>	103
Проекты. Панель <i>Project</i>	104
Работа с проектами	104

Глава 4. Переменные	113
Понятие переменной	113
Создание и типизация переменной.....	114
Явное объявление переменной.....	114
Неявное объявление переменной.....	117
Динамическое создание переменной.....	117
Присваивание, извлечение и изменение значения переменной.....	118
Присваивание значения.....	118
Автоматическая типизация переменной.....	120
Извлечение значения.....	121
Область видимости и адресация переменных	121
Вопросы области видимости.....	121
Локальные переменные	123
Глобальные переменные.....	126
Адресация переменных.....	129
Время жизни переменной	135
Удаление переменной.....	136
Глава 5. Элементарные типы данных	138
Элементарные и ссылочные типы данных.....	138
Числовой тип. Работа с числами.....	140
Литералы чисел.....	141
Специальные числовые значения.....	142
Операции над числами	144
Встроенные методы для работы с числами	147
Строковый тип. Работа со строками	148
Литералы строк.....	148
Операции над строками.....	151
Встроенные свойства и методы для работы со строками	153
Булев тип.....	164
Логические операции.....	165
Встроенные методы для работы с булевыми значениями	167
Типы <i>null</i> и <i>undefined</i>	168
Значение <i>null</i>	168
Значение <i>undefined</i>	168
Явное и автоматическое преобразование элементарных типов данных	169
Явное преобразование типов данных.....	169
Автоматическое преобразование типов данных.....	172
Глава 6. Выражения и операторы	176
Приоритет операторов	176
Ассоциативность.....	180
Модификация значения	180
Простое присваивание.....	181
Составное присваивание	182
Инкрементирование/декрементирование	182

Побитовые операции	184
Битовое представление числа	185
Применение двоичных чисел.....	187
Логические побитовые операции	188
Побитовые операции сдвига	193
Глава 7. Управляющие структуры: Операторы ветвления. Циклы	198
Ветвление кода	198
Предложение <i>if</i>	199
Предложение <i>else</i>	202
Предложение <i>else if</i>	206
Предложение <i>switch</i>	210
Многократное выполнение кода	214
Понятие программного цикла	215
Цикл <i>while</i>	216
Цикл <i>do while</i>	218
Цикл <i>for</i>	219
Цикл <i>for...in</i>	222
Контроль выхода из цикла.....	224
Вложенные циклы.....	225
Циклы монтажной линейки.....	226
Глава 8. Пользовательские функции	231
Общие сведения о функциях	231
Создание функции	232
Объявление функции	232
Использование литерала функции	233
Параметры функции	234
Контроль типа параметров.....	235
Класс <i>Arguments</i> . Контроль фактических параметров функции	236
Возвращение значения	240
Контроль типа возвращаемого значения.....	243
Вызов функции.....	245
Вызов функции по имени	245
Вызов функции с помощью ссылки	246
Вызов функции с помощью метода <i>apply()</i>	248
Вызов функции с помощью метода <i>call()</i>	250
Периодический вызов функции. Глобальные функции <i>setInterval()</i> и <i>clearInterval()</i>	250
Вопросы области видимости и времени существования функций.....	255
Область видимости функции	255
Доступ к функции	258
Время жизни функции.....	259
Рекурсивные функции	259

Глава 9. Массивы	262
Понятие массива	262
Создание массива	263
Конструктор массива	263
Литерал массива	264
Глобальная функция <i>Array()</i>	265
Вывод массива в панель <i>Output</i>	266
Обращение к элементам массива	266
Операции над элементами массива	267
Определение количества элементов в массиве	267
Присваивание значения элементу массива	268
Добавление элементов в массив	269
Строковое представление массива	274
Сортировка массива	276
Извлечение части массива	288
Удаление элементов массива	289
Ассоциативные массивы	290
Многомерные массивы	292
Часть II. БАЗОВЫЕ ВСТРОЕННЫЕ КЛАССЫ ACTIONSCRIPT	299
Глава 10. Класс <i>Button</i>. Работа с кнопками	301
Создание кнопки в рабочей среде	302
Кнопка как объект <i>AS</i>	304
Имя экземпляра	305
Режимы кнопки	306
Обработка событий кнопки	307
События кнопки	307
Обработчик событий кнопки	308
Контроль нажатия клавиатурной клавиши	311
Область видимости обработчика событий кнопок	313
Глава 11. Обработка событий клипов. Событийная модель <i>ActionScript</i>	315
Обработчик событий <i>on()</i> . Использование клипов в качестве кнопок	316
Создание клипа-кнопки	316
Область видимости обработчика событий <i>on()</i> клипа	317
События клипов	319
Событие <i>enterFrame</i>	320
Событие <i>load</i>	321
Событие <i>unload</i>	322
Событие <i>data</i>	322
Событие <i>mouseMove</i>	323
Событие <i>mouseDown</i>	323
Событие <i>mouseUp</i>	323
Событие <i>keyDown</i>	324

Событие <i>keyUp</i>	324
Область видимости обработчика <i>onClipEvent()</i>	325
Применение методов обработчиков событий	325
Понятие метода обработчика событий	326
Вопросы области видимости методов обработчиков событий кнопок и клипов	329
Удаление методов обработчиков событий	331
Применение методов обработчиков событий с использованием адресации	332
Трансляция событий	333
Передача и обработка событий	334
Прекращение трансляции событий	337
Создание источников событий. Объект <i>AsBroadcaster</i>	337
Инициализация источника и трансляция событий	338
Применение объекта <i>AsBroadcaster</i>	340
Глава 12. Класс <i>MovieClip</i>. Работа с клипами	344
Встроенные свойства класса <i>MovieClip</i>	344
Координаты. Свойства <i>_x</i> и <i>_y</i>	345
Размеры. Свойства <i>_width</i> , <i>_height</i> , <i>_xscale</i> и <i>_yscale</i>	347
Поворот. Свойство <i>_rotation</i>	352
Прозрачность. Свойство <i>_alpha</i>	352
Имя. Свойство <i>_name</i>	353
Отображение. Свойство <i>_visible</i>	354
Координаты мыши. Свойства <i>_xmouse</i> , <i>_ymouse</i>	355
Кадры. Свойства <i>_totalframes</i> , <i>_currentframe</i> и <i>_framesloaded</i>	358
Свойства клипов-кнопок: <i>useHandCursor</i> , <i>enabled</i> , <i>trackAsMenu</i> , <i>hitArea</i>	361
Создание выпадающего списка	363
Прочие свойства	369
Адресация с помощью предложения <i>with</i>	374
Управление воспроизведением	376
Остановка и запуск воспроизведения	376
Переход между кадрами	378
Управление воспроизведением монтажной линейки	383
Организация навигации внутри фильма	385
Программный реверс анимации	386
Загрузка сетевого ресурса	388
Организация навигации между Web-страницами	391
Динамическое создание клипов	391
Стеки графических объектов Flash	392
Стек документов проигрывателя	394
Присоединение экземпляра. Метод <i>attachMovie()</i>	394
Дублирование экземпляра. Метод <i>duplicateMovieClip()</i>	400
Контроль уровней стека	406
Создание пустого клипа. Метод <i>createEmptyMovieClip()</i>	409
Удаление динамически созданных клипов	409

Проверка состояния загрузки	411
Метод <i>getBytesTotal()</i>	411
Метод <i>getBytesLoaded()</i>	412
Создание предварительного загрузчика	412
Загрузка внешних Flash-документов и изображений	415
Загрузка на уровень стека проигрывателя	415
Организация навигации с использованием динамической загрузки внешних документов	417
Особенности загрузки в клип	418
Создание интерактивного слайд-шоу	420
Использование абсолютных ссылок во внешних документах	423
Загрузка переменных из внешнего источника	425
Удаление загруженных документов и содержимого клипов	427
Преобразование координат и контроль границ клипа	428
Преобразование в глобальные координаты	428
Преобразование в локальные координаты	430
Контроль границ клипа	431
Равномерное распределение клипов	431
Перетаскивание объектов	436
Перетаскивание с помощью метода <i>startDrag()</i>	437
Прекращение перетаскивания	438
Контроль наложения клипов	440
Контроль пересечения ограничивающих рамок клипов	440
Контроль попадания точки в заданную область	443
Глава 13. Динамическое рисование	446
Рисование прямолинейных сегментов	447
Задание начальной позиции рисования	447
Определение атрибутов контура	448
Соединение точек прямолинейным отрезком	448
Удаление графики	449
Создание инструмента "карандаш"	450
Применение сплошной заливки	454
Рисование многоугольников и звезд	457
Применение градиентной заливки	463
Определение матрицы трансформации градиента	465
Рисование криволинейных сегментов	479
Динамическое маскирование	483
Глава 14. Применение динамического и пользовательского текста	488
Создание и настройка параметров текстового блока в рабочей среде	489
Настройка параметров текстового блока в рабочей среде	490
Встроенные и машинно-независимые шрифты	495
Использование текстового блока, созданного в рабочей среде	496
Использование шрифтовых символов	497

Динамическое создание и форматирование текста.....	500
Класс <i>TextField</i> . Настройка параметров текстового блока	501
Класс <i>TextFormat</i> . Свойства форматирования текста.....	510
Создание всплывающих подсказок	517
Получение информации о параметрах форматирования текста	522
Получение информации о метриках шрифта.....	523
Проверка наличия у пользователя используемого шрифта	526
Организация прокрутки текста	527
Автоматическая прокрутка текста.....	527
Свойства управления вертикальной прокруткой текста.....	528
Свойства управления горизонтальной прокруткой текста.....	529
Контроль прокрутки содержимого текстового блока	531
Примеры реализации текстовой прокрутки	532
Контроль выделения и передачи фокуса. Класс <i>Selection</i>	540
Передача фокуса	540
Обнаружение носителя фокуса	542
Контроль передачи фокуса.....	543
Определение границ выделения и положения курсора	544
Управление выделением	547
Управление порядком переноса фокуса с помощью клавиши табуляции.....	548
Обработка событий текстового блока	550
Замена содержимого текстового блока	554
Удаление текстового блока.....	557
Форматирование текста с помощью HTML.....	558
Активизация поддержки HTML и вывод текста с применением HTML-форматирования	559
Теги HTML, используемые для форматирования текста.....	560
Поддержка и использование CSS. Класс <i>StyleSheet</i>	572
Создание таблицы стилей.....	573
Определение стилей.....	576
Добавление стилей в таблицу.....	581
Получение информации об используемых стилях	593
Очистка таблицы стилей.....	595
Загрузка текстовых данных из XML-документа	595
Контроль статического текста. Класс <i>TextSnapshot</i>	601
Создание объекта <i>TextSnapshot</i>	602
Чтение и поиск в статическом тексте	603
Выделение статического текста цветом	605
Глава 15. Класс <i>Math</i>. Выполнение математических вычислений	611
Встроенные свойства. Математические константы.....	611
Возведение числа в степень и извлечение корня	612
Вычисление логарифма числа.....	613
Определение наибольшего и наименьшего значений.....	614
Определение модуля числа.....	615
Округление значений.....	616
Математическое округление.....	616

Округление до большего целого	617
Округление до меньшего целого	617
Тригонометрические функции.....	617
Применение генератора случайных чисел.....	624
Эффект падающего снега	625
Глава 16. Класс <i>Date</i>. Контроль времени и даты	629
Всемирное скоординированное время и машинное время	629
Создание экземпляра класса <i>Date</i>	631
Вызов конструктора без параметров	631
Вызов конструктора с передачей ему машинного времени.....	631
Вызов конструктора с передачей ему параметров в формате даты.....	632
Определение и установка параметров даты.....	633
Определение местного времени и даты	634
Определение UTC-времени и даты	636
Установка местного времени и даты.....	636
Установка UTC-времени и даты.....	638
Работа с машинным временем.....	638
Создание календаря	639
Глобальная функция <i>getTimer()</i>	644
Глава 17. Классы <i>Mouse</i> и <i>Key</i>. Контроль мыши и клавиатуры.....	648
Контроль отображения курсора и обработка событий мыши.....	648
Управление отображением курсора.....	649
Обработка событий мыши.....	649
Контроль клавиатуры.....	655
События клавиатуры	655
Определение характеристик и проверка состояния клавиши	656
Глава 18. Класс <i>Color</i>. Работа с цветом	664
Цветовая модель RGB.....	665
Цветовая модель HSB	668
Перевод цвета из модели RGB в модель HSB и наоборот	670
Создание объекта <i>Color</i>	677
Окрашивание клипов. Методы <i>setRGB()</i> и <i>getRGB()</i>	678
Формирование каталога цветов на основе безопасной Web-палитры.	
Сортировка по тону	681
Цветовые трансформации. Методы <i>setTransform()</i> и <i>getTransform()</i>	686
Типы цветовых трансформаций	686
Применение методов <i>setTransform()</i> и <i>getTransform()</i>	687
Создание панели синтеза цвета на основе модели HSB.....	691
Создание интерактивного цветового редактора.....	696
Глава 19. Класс <i>Sound</i>. Управление звуком.....	698
Создание объекта <i>Sound</i>	698
Присоединение звука.....	700

Управление воспроизведением звука	701
Запуск воспроизведения	701
Остановка воспроизведения	702
Обработка окончания звука	704
Определение длительности и текущей позиции воспроизведения звука	704
Загрузка внешнего звукового файла	706
Загрузка звука в потоковом и событийном режимах	706
Контроль состояния загрузки звука	707
Проверка окончания загрузки звука	709
Поддержка ID3-тегов	710
Управление уровнем звука	714
Управление балансом каналов	716
Управление выводом каналов	718
Пример программного управления звуком. Интерактивная перкуссия	720
Глава 20. Управление процессом загрузки внешних документов и изображений.	
Взаимодействие между SWF-документами	727
Класс <i>MovieClipLoader</i>	727
Создание экземпляра класса	728
Методы класса <i>MovieClipLoader</i>	728
Обработка событий	730
Начало загрузки. Событие <i>onLoadStart</i>	731
Поступление данных. Событие <i>onLoadProgress</i>	734
Окончание загрузки. Событие <i>onLoadComplete</i>	734
Инициализация загруженного документа. Событие <i>onLoadInit</i>	735
Ошибка при загрузке. Событие <i>onLoadError</i>	736
Пример управления загрузкой нескольких документов	737
Вопросы политики безопасности, связанные с загрузкой внешних SWF-документов	742
Класс <i>LocalConnection</i> . Взаимодействие между SWF-документами	744
Механизм взаимодействия	744
Инструментарий класса <i>LocalConnection</i>	746
Объявление методов объекта <i>LocalConnection</i> в документе-получателе	746
Открытие соединения. Метод <i>connect()</i>	747
Отправка данных получателю	748
Закрытие соединения	751
Пример организации взаимодействия документов	752
Вопросы политики безопасности, связанные с взаимодействием документов	754
Доменная безопасность	755
Безопасность протоколов	758
Глава 21. Класс <i>SharedObject</i>. Хранение информации на стороне клиента	759
Понятие об объекте <i>SharedObject</i>	759
Ограничения локального хранения информации	760
Вопросы безопасности локального хранения информации	762

Создание объекта <i>Shared Object</i>	763
Запись и чтение данных в объекте <i>Shared Object</i>	766
Сохранение объекта <i>Shared Object</i>	768
Определение объема локального объекта.....	771
Удаление локальных данных.....	771
Пример применения объекта <i>Shared Object</i> для хранения пользовательских настроек интерфейса.....	772
Глава 22. Контроль параметров документа и проигрывателя.	
Информация о системе и оборудовании	779
Класс <i>Stage</i> . Контроль размеров и масштабирования окна фильма.....	779
Управление масштабированием фильма.....	779
Управление выравниванием фильма.....	782
Контроль размеров сцены и окна проигрывателя.....	784
Управление отображением меню проигрывателя.....	789
Управление автономным проигрывателем. Функция <i>fscommand()</i>	790
Класс <i>ContextMenu</i> . Управление контекстным меню проигрывателя.....	792
Механизм управления контекстным меню.....	792
Управление стандартным содержимым меню.....	794
Добавление и настройка пользовательских элементов меню.....	798
Примеры управления контекстным меню.....	801
Информация о системе и оборудовании. Объект <i>System.capabilities</i>	804
Глава 23. Класс <i>PrintJob</i>. Печать из Flash-фильма	810
Механизм создания и передачи задания на печать.....	811
Вызов окна с настройками параметров печати.....	812
Отправка задания в спулер печати.....	814
Вывод содержимого на печать.....	816
Применение класса <i>PrintJob</i>	816
Глава 24. Динамическая загрузка и воспроизведение видео.....	824
Формат <i>Flash Video</i> . Конвертирование видео в формат FLV.....	827
Механизм выполнения загрузки FLV-файлов.....	832
Создание объекта <i>Video</i> . Методы и свойства видеообъекта.....	833
Открытие локального соединения.....	834
Контроль загрузки и воспроизведения FLV-видео. Класс <i>NetStream</i>	835
Инициирование выполнения загрузки.....	835
Контроль процесса загрузки.....	836
Установка размера буфера.....	837
Управление воспроизведением.....	838
Ручной поиск, перемотка.....	839
Контроль звукового сопровождения.....	840
Обработка событий объекта <i>NetStream</i>	842
Прекращение загрузки.....	846
Создание видеопроигрывателя с динамической загрузкой FLV-файла.....	846

Часть III. ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ.**СОЗДАНИЕ КОМПОНЕНТОВ855****Глава 25. Создание пользовательских классов 857**

Файлы классов. Понятие <i>classpath</i>	858
Объявление класса	861
Создание экземпляра класса.....	863
Пакеты.....	864
Объявление свойств	865
Установка слежения за значением свойства	869
Объявление конструктора.....	871
Объявление методов.....	874
Регулирование доступа к членам класса.....	876
Статические члены класса.....	880
Организация считывания и установки значений свойств.	
Определение <i>get/set</i> -методов.....	882
Динамические классы.....	886
Суперклассы и подклассы	888
Представление о внутреннем механизме реализации наследования.....	893
Связывание символа с пользовательским классом.....	895
Связывание символа с классом с помощью окна <i>Linkage Properties</i>	896
Связывание символа с классом с помощью метода <i>Object.registerClass()</i>	898
Директивы <i>#initclip</i> и <i>#endinitclip</i>	899
Интерфейсы	901
Пример создания и применения пользовательских классов	903

Глава 26. Трехмерное моделирование и анимация 918

Понятие о матрицах.....	918
Определение матрицы и ее представление в <i>ActionScript</i>	919
Умножение матриц	920
Класс <i>Calculus</i>	921
Изображение и преобразование трехмерных точек.....	924
Проецирование трехмерных точек на плоскость.....	928
Создание класса <i>Vertex3D</i>	932
Класс <i>Vertex3D</i> . Трехмерные преобразования и визуализация точек	933
Пример реализации: управление центрированным кубом со срезанным углом с помощью клавиатуры	945
Пример реализации: управление трехмерной плоскостью с произвольным числом точек с помощью мыши	952

Глава 27. Создание пользовательских компонентов 959

Представление о компонентах.....	960
Общий процесс создания компонента.....	962
Ключевые этапы создания компонента.....	962
Проектирование компонента.....	964
Создание клипа компонента.....	967

Объявление класса компонента	969
Объявление свойств	970
Применение тегов метаданных	973
Тег <i>Inspectable</i> . Описание параметров внешнего интерфейса компонента	973
Тег <i>InspectableList</i> . Список внешних параметров	978
Тег <i>Event</i> . Описание событий компонента	979
Тег <i>IconFile</i> . Создание пиктограммы компонента	979
Объявление <i>get/set</i> -методов	980
Трансляция и обработка событий компонентов	988
Инициализация источника событий	988
Трансляция события. Событийный объект	990
Обработка событий компонентов	991
Трансляция пользовательского события <i>onUpdate</i>	997
Инициализация компонента	998
Изменение размеров компонента	1000
Динамическое присоединение графических элементов компонента	1001
Компоновка составляющих элементов	1003
Обеспечение функциональности графических элементов компонента	1004
Использование режима предварительного просмотра (Live Preview)	1007
Распространение компонента	1008
Компиляция клипа компонента	1009
Экспортирование SWC-файла	1009
Создание MXP-пакета расширения	1010

Глава 28. Обработка исключительных ситуаций. Отладка сценариев.

Тестирование проекта	1014
Обработка исключительных ситуаций	1014
Тестирование проекта	1028
Управление отображением и воспроизведением фильма	1028
Эмуляция загрузки	1029
Отчет о размерах элементов фильма	1034
Средства отладки сценариев	1035
Панель <i>Output</i>	1035
Применение отладчика (панель <i>Debugger</i>)	1039

ПРИЛОЖЕНИЯ

1051

Приложение 1. Справочник: методы, свойства и обработчики событий основных



встроенных классов ActionScript	1053
Класс <i>Array</i> . Создание и обработка массивов	1053
Класс <i>Button</i> . Работа с кнопками	1054
Класс <i>Color</i> . Работа с цветом	1056
Класс <i>ContextMenu</i> . Управление контекстным меню проигрывателя	1057
Класс <i>ContextMenuItem</i> . Пользовательские элементы контекстного меню	1057
Класс <i>Date</i> . Контроль времени и даты	1058





Класс <i>Error</i> . Исключительные ситуации.....	1060
Класс <i>Key</i> . Контроль клавиатуры.....	1060
Класс <i>LocalConnection</i> . Взаимодействие между фильмами.....	1060
Класс <i>Math</i> . Выполнение математических вычислений.....	1061
Класс <i>Mouse</i> . Контроль мыши	1063
Класс <i>MovieClip</i> . Работа с клипами	1063
Класс <i>MovieClipLoader</i> . Управление процессом загрузки внешних документов и изображений	1069
Класс <i>NetConnection</i> . Установка локального соединения для динамической загрузки FLV-файлов.....	1070
Класс <i>NetStream</i> . Динамическая загрузка и воспроизведение FLV-файлов.....	1070
Класс <i>PrintJob</i> . Печать из Flash-фильма.....	1071
Класс <i>Selection</i> . Контроль выделения и передачи фокуса.....	1071
Класс <i>SharedObject</i> . Хранение информации на стороне клиента.....	1072
Класс <i>Sound</i> . Управление звуком.....	1072
Класс <i>Stage</i> . Контроль параметров документа.....	1074
Класс <i>StyleSheet</i> . Поддержка и использование CSS.....	1074
Класс <i>System</i> . Информация о системе и оборудовании	1075
Класс <i>TextField</i> . Контроль состояния и параметры текстового блока	1075
Класс <i>TextFormat</i> . Динамическое форматирование текста.....	1079
Класс <i>TextSnapshot</i> . Контроль статического текста.....	1080
Класс <i>Video</i> . Работа с видеообъектами.....	1081
Приложение 2. Описание компакт-диска.....	1082
Приложение 3. Виртуальные коды клавиш	1088
Приложение 4. Горячие клавиши инструкций ActionScript	1090
Приложение 5. Интернет-ресурсы, посвященные Flash.....	1092
Предметный указатель	1094

Работа с библиотекой

Каждый FLA-документ содержит свою библиотеку (library), в которой хранятся импортированные объекты (растровые изображения, видеоролики, звуки) и символы. С помощью команд библиотеки можно задавать различные параметры содержащихся в ней объектов и упорядочивать их размещение.



Далее представлен перечень операций, которые можно производить над объектами библиотеки, и способы их выполнения.

- **Выделение.** Для того чтобы выделить объект библиотеки, нужно щелкнуть в перечне объектов по любому месту соответствующей строки. Для выделения нескольких объектов, следующих друг за другом, нужно выделить первый объект диапазона, а затем щелкнуть на последнем, удерживая клавишу <Shift>. С помощью клавиши <Ctrl> можно выделить объекты вразбивку.
- **Создание новых объектов.** Для того чтобы создать новый символ, можно воспользоваться пиктограммой **New Symbol...**  в нижней части окна библиотеки или одноименной командой контекстного меню библиотеки. Для создания нового шрифтового символа (Font Symbol) или видеообъекта необходимо воспользоваться командами контекстного меню библиотеки **New Font...** и **New Video...** соответственно.
- **Переименование.** В качестве имени объекта библиотеки может быть использована любая последовательность символов (буквенные, числовые или знаки пунктуации), при этом допустимо использование кириллицы. Имя одного объекта библиотеки не может быть использовано в качестве имени другого ее объекта. Переименовать объект можно одним из следующих способов.
 - Дважды щелкнуть на существующем имени и задать другое имя.
 - Выделить объект и, нажав клавишу <F2>, задать другое имя.
 - Щелкнуть правой кнопкой мыши по соответствующей строке и из контекстного меню объекта выбрать команду **Rename**.
 - Выделить объект и, воспользовавшись контекстным меню библиотеки, выполнить команду **Rename**.
- **Создание копии.** Чтобы скопировать объект, можно выполнить одно из следующих действий.
 - Щелкнуть правой кнопкой мыши на соответствующей строке и из контекстного меню объекта выбрать команду **Duplicate**.
 - Выделить объект и, воспользовавшись контекстным меню библиотеки, выполнить команду **Duplicate**.
- **Сортировка.** Изменить порядок размещения объектов в библиотеке можно с помощью пиктограммы **Toggle Sorting Order** , расположенной над вертикальной полосой прокрутки библиотеки. Для того чтобы отсортировать объекты библиотеки по порядку на основе любого из свойств, указанных в заголовках столбцов библиотеки (Name, Type и т. д.), необходимо щелкнуть на соответствующем заголовке и затем воспользоваться пиктограммой **Toggle Sorting Order**.

- **Организация работы с помощью папок.** Папки выполняют в библиотеке примерно такую же роль, как и каталоги в файловой системе. Они служат для более компактной и упорядоченной организации элементов. Чтобы создать папку можно либо щелкнуть на пиктограмме **New Folder**  в нижней части окна библиотеки, либо выполнить одноименную команду ее контекстного меню. Вновь созданная папка не содержит объектов и отображается пиктограммой . Для того чтобы поместить туда объекты, их нужно выделить и перетащить, "бросив" в требуемую папку. Допускается использование вложенных папок. Папка, наполненная содержимым, отображается пиктограммой . Чтобы раскрыть папку для доступа к ее содержимому, нужно дважды щелкнуть на ее пиктограмме или выполнить команду контекстного меню **Expand Folder**. Раскрытая папка отображается пиктограммой . Объекты, находящиеся внутри раскрытой папки, отображаются с отступом. Для закрытия папки нужно дважды щелкнуть на ее пиктограмме или выполнить команду контекстного меню **Collapse Folder**. Чтобы вытащить объект (или несколько) из папки, необходимо выделить его и переместить за пределы папки.

Для того чтобы поместить выделенные объекты в автоматически созданную папку, можно выполнить команду **Move to New Folder** контекстного меню объектов или библиотеки. В появившемся диалоговом окне **New Folder** нужно будет задать имя новой папки.

При работе с большим количеством символов (или импортированных объектов) имеет смысл размещать тематически близкие объекты библиотеки в отдельных папках, присваивая им лаконичные осмысленные имена.

- **Размещение в фильме.** Для того чтобы поместить экземпляр объекта библиотеки в фильм, необходимо перетащить его курсором из списка объектов или окна просмотра библиотеки на сцену или в среду редактирования символа.
- **Изменение свойств объекта.** Для того чтобы изменить свойства объекта библиотеки (изменить тип символа, задать параметры компрессии растровых изображений и звуков и т. д.), можно щелкнуть на пиктограмме **Properties...**  либо выполнить одноименную команду контекстного меню библиотеки или объекта.
- **Удаление.** Для того чтобы удалить один или несколько объектов библиотеки, необходимо выделить их и либо щелкнуть на пиктограмме **Delete** , либо выполнить одноименную команду контекстного меню объекта или библиотеки.

Работа с общими библиотеками (Shared Libraries)

При работе с проектом, включающим в себя несколько файлов SWF, содержащих различные разделы данного проекта, зачастую возникает необходимость использовать одинаковые элементы, включая их в каждый раздел.

Общая библиотека содержит ресурсы (символы, растровые изображения, звуки), которые могут быть использованы любыми файлами проекта. При необходимости использования объекта вместо его встраивания в файл устанавливается связь (linkage) с соответствующим элементом общей библиотеки. При этом файл, содер-

жащий общую библиотеку, должен быть загружен один раз, после чего его содержимое может многократно использоваться другими файлами, причем применение элементов общей библиотеки практически не увеличивает объем файла-получателя.

Использование общих библиотек предполагает выполнение следующих этапов.

- Определение элементов общей библиотеки.
- Включение элементов общей библиотеки в другие файлы проекта.
- Размещение файла SWF, содержащего общую библиотеку, и остальных файлов проекта на сервере.

Для создания общей библиотеки необходимо выполнить следующие действия:

1. Создать новый FLA-документ или открыть имеющийся.
2. Поместить в библиотеку этого документа элементы, используемые в качестве общих ресурсов: символы (графический, муви-клип или кнопка), растровые изображения, звуки, шрифтовые символы (см. главу 14).
3. Для каждого элемента задать параметры связи. Для этого необходимо выделить соответствующий элемент библиотеки документа и, щелкнув на нем правой кнопкой мыши, выбрать из контекстного меню команду **Linkage...** Далее в появившемся диалоговом окне **Linkage Properties** задать следующие параметры связи:
 - **Identifier** — уникальное имя, используемое для идентификации связанного элемента общей библиотеки. Данное имя должно содержать только символы латинского алфавита и не может включать пробелы;
 - **Export for runtime sharing** — установка этого флажка указывает на то, что данный объект будет экспортироваться в качестве элемента общей библиотеки;
 - **URL** — в это поле необходимо ввести абсолютный или относительный путь к файлу, содержащему общую библиотеку. Относительный путь указывается относительно местоположения файла-получателя. Если файл-получатель размещается на HTML-странице, то в поле **URL** должен быть указан путь к файлу, содержащему общую библиотеку, относительно местоположения HTML-страницы файла-получателя.

Абсолютный адрес общей библиотеки или путь к файлу, содержащему общую библиотеку, относительно документа-получателя может также быть задан с помощью команды контекстного меню библиотеки **Shared Library Properties**. В результате выполнения этой команды появляется одноименное диалоговое окно, в котором можно задать требуемый путь. Путь, указанный в окне **Shared Library Properties**, автоматически прописывается в поле **URL** окна **Linkage Properties**. Верно и обратное — при изменении значения поля **URL** в окне **Linkage Properties** изменяется путь, заданный в окне **Shared Library Properties**.

Примечание

Флажок **Export in first frame**, устанавливаемый по умолчанию, приводит к тому, что при загрузке файла, содержащего объекты, экспортируемые в качестве объектов общей библиотеки, они будут загружены перед загрузкой содержимого первого кадра фильма. Если документ, содержащий элементы общей библиотеки, используется не только как источник общих ресурсов, но и как самостоятельный раздел проекта, предназначенный для

просмотра, то установка данного флажка приведет к задержке начала воспроизведения документа и, в частности, к задержке выполнения сценариев первого кадра, в котором зачастую содержится предзагрузчик. Если флажок **Export in first frame** снят, то объекты, экспортируемые в качестве общих ресурсов, будут загружены при их первом вхождении в фильм, содержащий общую библиотеку. В этом случае их экземпляры должны быть непосредственно помещены на монтажную линейку документа, содержащего общую библиотеку. Если этого не сделать, элементы общей библиотеки не будут экспортированы в конечный файл SWF. Таким образом, если документ используется исключительно как общая библиотека и не должен демонстрироваться зрителю, флажок **Export in first frame** необходимо установить. Если же документ, содержащий элементы общей библиотеки, выполняет еще и роль определенного раздела проекта и содержит какие-то объекты на своей монтажной линейке, нужно исходить из конкретной ситуации и либо установить флажок (это может привести к задержке воспроизведения), либо снять его (в этом случае элементы общей библиотеки должны быть помещены на монтажную линейку данного документа).

4. Сохранить документ (**File>Save As...**) и опубликовать его (**File>Publish**), присвоив имя в соответствии с именем, указанным в поле **URL** окна **Linkage Properties**.

Чтобы включить элемент общей библиотеки в другой документ Flash, необходимо выполнить следующие действия.

1. Открыть документ, в который должен быть помещен элемент общей библиотеки.
2. Открыть библиотеку документа, содержащего элементы общей библиотеки (**File>Import>Open External Library...**).
3. Перетащить требуемые элементы из внешней библиотеки на сцену или в библиотеку текущего документа. В результате этих действий импортированным объектам в окне **Linkage Properties** автоматически будут заданы следующие параметры связи:
 - **Identifier** — уникальный идентификатор импортированного элемента общей библиотеки. Данное имя должно совпадать с именем, заданным при определении этого объекта в качестве элемента общей библиотеки;
 - **Import for Runtime Sharing** — установка этого флажка указывает на то, что данный объект будет импортирован из общей библиотеки в текущий документ;
 - **URL** — адрес SWF-файла, содержащего общую библиотеку, указанный относительно местоположения текущего документа.
4. Сохранить документ (**File>Save As...**) и опубликовать его (**File>Publish**), разместив конечный файл таким образом, чтобы его местоположение относительно файла, содержащего общую библиотеку, соответствовало пути, указанному в поле **URL** окна **Linkage Properties**.

При размещении конечных файлов на сервер необходимо отправить и все SWF-файлы проекта и файл SWF, содержащий общую библиотеку. При этом важно разместить их друг по отношению к другу таким образом, чтобы все ссылки на общую библиотеку разрешались корректно (взаимное размещение файла-получателя и файла с общей библиотекой должно соответствовать пути, указанному в поле **URL** окна **Linkage Properties**).

Разрешение вопросов политики безопасности при импорте внешних данных из других доменов

С выходом новой версии Flash Player 7 изменились правила политики безопасности, действующие при загрузке внешних данных (общих библиотек, документов XML, текстовых документов и т. д.). В частности, импорт ресурсов общих библиотек может быть непосредственно осуществлен только при условии, что фильм-получатель и фильм, содержащий общую библиотеку, расположены в одном и том же домене. Таким образом, фильм, расположенный по адресу **www.whenever.com**, не сможет загрузить ресурсы общей библиотеки с сайта **www.anywhere.com**.

Для того чтобы получить возможность осуществлять загрузку ресурсов общей библиотеки, расположенной в другом домене, необходимо, чтобы в домене, содержащем общую библиотеку, находился файл с описанием междоменной политики безопасности (cross-domain security). Этот файл представляет собой XML-документ, в котором указано, фильмы с какого домена имеют право осуществлять загрузку данных, хранящихся в настоящем домене. Когда Flash-фильм, расположенный в одном домене, пытается осуществить загрузку данных из другого домена, Flash Player автоматически проверяет наличие файла с политиками в корневом каталоге этого домена. Если домен, с которого осуществляется попытка получить доступ к данным, включен в файл с политиками, данные будут доступны.

Файл с политиками должен быть назван **crossdomain.xml** и помещен в корневой каталог сервера, предоставляющего данные. Файлы с политиками действуют только на серверах, использующих протоколы HTTP, HTTPS или FTP, и только в пределах конкретного порта и протокола.

Документ **crossdomain.xml** должен содержать следующие теги:

- `<allow-access-from>` — тег, содержащий атрибут `domain`, значением которого является точное имя домена, фильмам с которого разрешается получать доступ к ресурсам, хранящимся в домене, содержащем **crossdomain.xml**;
- `<cross-domain-policy>` — внешний тег, содержащий необходимое количество вложенных тегов `<allow-access-from>`.

Ниже приведен пример файла с описанием политики, позволяющей осуществить доступ к данным SWF-файлам, расположенным в доменах **www.anywhere.com** и **www.whenever.com**.

```
<cross-domain-policy>
<allow-access-from domain="*www.anywhere.com"/>
<cross-domain-policy/>
```

Значение атрибута `domain` должно представлять собой точный домен, точный IP-адрес, или может быть задано с применением шаблона "звездочка" (*). Звездочка позволяет осуществлять загрузку ресурсов с сервера фильмами со всех поддоменов домена, перед именем которого указана звездочка.

Ниже приведен пример файла с описанием политики, позволяющей осуществить доступ к данным SWF-файлам, расположенным в любом поддомене домена **anywhere.com**.

```
<cross-domain-policy>
<allow-access-from domain="*.anywhere.com"/>
</cross-domain-policy>
```

Для того чтобы организовать доступ к данным, хранящимся в домене **www.whenever.com** из любого домена, можно в качестве значения атрибута `domain` использовать звездочку. Следующий код позволяет получать доступ к ресурсам сервера фильмам, расположенным в любом домене.

```
<cross-domain-policy>
<allow-access-from domain="*" />
</cross-domain-policy>
```

Если файл с политиками не содержит тега `<allow-access-from>`, это равносильно отсутствию данного файла.

Анимация

Flash предоставляет различные способы создания анимации. К их числу можно отнести:

- покадровую анимацию;
- автоматическую анимацию;
- программную анимацию.

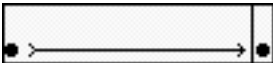
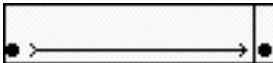
Покадровая анимация является наиболее трудоемкой, но в то же время позволяет добиться точной передачи нюансов движения и, как правило, используется для создания художественных детально проработанных эффектов (например, анимация пламени).

Автоматическая анимация (Tweened animation) создается в рабочей среде на монтажной линейке основного фильма или символа (графического типа или типа муви-клип). Автоматическая анимация, в отличие от покадровой, не требует ручной прорисовки фаз. Достаточно задать начальное и конечное состояние анимируемого объекта, а все промежуточные состояния будут рассчитаны автоматически. Диапазон применения автоматической анимации очень широк. С ее помощью можно создавать эффекты, связанные с изменением формы, цвета и прозрачности, перемещением, вращением и масштабированием объектов. Процесс создания автоматической анимации значительно менее трудоемкий, чем покадровой, и во многих случаях позволяет добиться превосходных результатов. Существует два типа автоматической анимации: *анимация формы* (shape tweening) и *анимация движения* (motion tweening). Сравнительные характеристики типов автоматической анимации представлены в табл. 1.14.

Программная анимация создается средствами ActionScript, обладает широкими возможностями и применяется для разработки анимационных эффектов, обладающих сложным характером движения.

Зачастую в фильме используется сочетание всех трех способов, что позволяет решать широкий спектр задач, поставленных перед аниматором.

Таблица 1.14. Сравнительные характеристики типов автоматической анимации

Сравнительные характеристики	Тип автоматической анимации	
	Анимация формы Shape Tweening	Анимация движения Motion Tweening
Способ создания	Панель Properties , список Tween , тип Shape	1. Панель Properties список Tween тип Motion . 2. Команда Insert>Timeline>CreateMotionTween . 3. Команда контекстного меню кадра Create Motion Tween
Вид на монтажной линейке	 зеленый фон	 голубой фон
Уровень иерархии графических объектов	Рабочий уровень	Наложный уровень
Типы графических объектов	Векторные формы, контуры, разбитые текст и растровая графика	Группы, экземпляры растровых изображений, текстовые блоки, экземпляры символов, внедренные видеоролики
Количество одновременно анимируемых объектов в одном слое	Не ограничено	Только один объект
Свойства объектов, изменяемые при анимации	Форма, цвет, прозрачность, размер, координаты	Координаты, размер, поворот, скос, цветовые эффекты экземпляров символов Color Effect (яркость, цвет и прозрачность)
Возможности	Автоматическое ускорение/замедление анимации, характер перехода (Blend)	Автоматическое ускорение/замедление анимации, автоматическое масштабирование и вращение объекта
Особенности	Использование меток под-сказки Shape Hints	Возможность анимации по маршруту

Покадровая анимация (Frame by frame animation)

Покадровая, или *ручная* анимация предполагает использование ключевых кадров для размещения в них отдельных фаз движения. Если та или иная фаза должна экспонироваться в течение некоторого промежутка времени, более длительного, чем время просмотра ключевого кадра, то после данного ключевого кадра размещается необходимое число простых кадров. Таким образом, покадровая анимация представляет собой последовательность ключевых и простых кадров. Ключевые кадры содержат основные фазы движения, а простые служат для продления времени их просмотра.

При создании покадровой анимации необходимо придерживаться следующих правил.

- **Разделение статики и динамики** — статичные объекты, не подвергающиеся анимации, должны размещаться на отдельных слоях. Так, например, фоновое изображение, остающееся неподвижным в течение всего фильма, должно быть помещено на отдельный слой, в противном случае его придется перерисовывать из кадра в кадр, что в итоге приведет к запутанности структуры фильма и увеличению объема конечного файла.
- **Разделение одновременных анимационных действий** — если в фильме одновременно используется несколько анимационных процессов, то каждый из них должен быть размещен отдельно на своем собственном слое. Это правило позволит правильно организовать структуру фильма и гибко манипулировать объектами в процессе работы. Так, например, если на сцене одновременно движутся два персонажа, то анимировать каждого из них нужно в отдельном слое, в противном случае можно столкнуться с серьезными трудностями при редактировании анимации, изменении порядка следования планов фильма или одновременном выполнении преобразований содержимого нескольких кадров.
- **Разделение планов фильма** — объекты, принадлежащие разным планам (передний, средний, задний и т. д.), должны быть размещены в разных слоях. Такая организация позволит успешно реализовать впечатление пространственной глубины и легко управлять структурой фильма.


Анимация формы (Shape tweening)

Анимация формы может быть применена исключительно к объектам *рабочего* уровня. С ее помощью можно выполнить следующие виды трансформаций:

- изменение формы;
- изменение цвета и прозрачности;
- перемещение;
- изменение масштаба, скос (наклон).

Для того чтобы создать автоматическую анимацию формы, необходимо выполнить следующие действия.

1. Создать новый слой (**Insert>Timeline>Layer**).
2. В начальном ключевом кадре создать объект *рабочего* уровня (векторную форму и/или контур), например, красный квадрат.

3. Исходя из скорости воспроизведения монтажной линейки, рассчитать необходимое для выполнения анимации число кадров и создать конечный ключевой кадр (**Insert Keyframe** или **Insert Blank Keyframe**). Если анимация должна выполняться за 2 секунды, то при скорости воспроизведения 12 кадров в секунду новый ключевой кадр должен иметь номер 24.
4. Поместить новый объект *рабочего* уровня в пустой ключевой кадр или изменить скопированный из начального кадра объект. В 24-м кадре можно в качестве примера создать синюю звезду. Если координаты объектов в начальном и конечном ключевых кадрах различны, то в ходе анимации будет выполняться перемещение.
5. Задать параметры автоматической анимации. Для этого необходимо выделить *первый* ключевой кадр диапазона и в Инспекторе свойств установить следующие настройки.
 - В выпадающем списке **Tween** выбрать тип анимации **Shape**. При этом начальный и конечный ключевые кадры будут соединены прямой неразрывной линией со стрелкой. Если режим **Tinted Frames** контекстного меню монтажной линейки активен, то диапазон кадров автоматической анимации формы будет выделен зеленоватым цветом. Если в анимации допущена ошибка, то на монтажной линейке появляется пунктирная линия, а при выделении соответствующего кадра в Инспекторе свойств появляется кнопка с пиктограммой восклицательного знака , свидетельствующая об ошибке в анимации.
 - Поле **Easing Value** используется для задания относительного замедления анимации в начале или в конце диапазона. Значение, принимаемое параметром **Ease**, изменяется в диапазоне от -100 до $+100$ и может быть непосредственно введено вручную либо задано с помощью вертикального слайдера. Отрицательные значения приводят к замедлению анимации в начале (**Easing In**) и, соответственно, к ускорению в конце. Положительные значения приводят к замедлению анимации в конце (**Easing Out**) и к ускорению в начале. При задании отрицательного значения рядом с полем **Easing Value** появляется предлог **In**, а при задании положительного значения — предлог **Out**. Замедление анимации достигается не за счет замедления скорости воспроизведения монтажной линейки, а за счет более плотного размещения фаз анимируемого объекта в начале или в конце диапазона кадров трансформации.
 - Параметр **Blend** позволяет задать тип перехода между начальной и конечной формами. Значение **Angular** приводит к тому, что при переходе у промежуточных форм сохраняются углы и прямолинейные сегменты. Значение **Distributive**, наоборот, сглаживает промежуточные формы. Переход типа **Angular** может быть применен только к формам, содержащим углы и прямолинейные сегменты; в противном случае переход выполняется с использованием значения **Distributive**.
6. Просмотреть результат в рабочей среде (**Control>Play**) или в среде тестирования (**Control>Test Movie**).