Штенников Дмитрий Геннадьевич

MACROMEDIA FLASH. ОСНОВЫ ПРОГРАММИРОВАНИЯ. ПРОСТЕЙШИЕ СКРИПТЫ. УРОК 6

В предыдущей статье началось описание и обсуждение простых приложений, созданных с использованием языка программирования ActionScript 2. В данной статье мы рассмотрим уроки по созданию приложений с использованием событий кадра.

События кадра (фрейма) – одни из самых часто используемых и простых для понимания начинающих пользователей событий. При прикреплении скрипта к тому или иному кадру следует учитывать, что события прикрепляются только к ключевым кадрам. Когда вы прикрепляете ряд действий к ключевому кадру, программа выполняет эти действия, когда ваш ролик уже отыграл тот кадр, в котором находились программные действия. События фрейма используются для того, чтобы вызывать действия, основанные на работе с временной линейкой, или для того, чтобы действия были синхронизированы с элементами, видимыми на экране.

Многие динамические проекты могут иметь множество объектов в момент начала проигрывания. Известный как инициализация данный процесс, как правило, включает данные создания или установление некоторых аспектов функциональных возможностей кино. В связи с этой причиной многие разработчики размещают акции скрипта в первом кадре основного графика времени, чтобы оптимальным образом осуществить процесс инициализации. Эти действия используются, чтобы создать переменные, массивы и определить функции – все элементы, которые могут быть необходимы вскоре после того, как приложение будет запущено.

В следующем упражнении будет создано приложение с использованием несинхронизированного представления, которое будет осуществлять демонстрацию изображений и связанный текст, последовательно использующий события кадра.

В данном случае будет "Мой отдых в Европе".

Для начала создадим заготовку и создадим на сцене дополнительно 5 слоев. Нижний слой назовем **Фон**, следующий слой – **Текстовые поля**, третий – **Кнопки**, четвертый – **Временной индикатор**, затем **Картинки** и **Действия**, соответственно (рисунок 1).

В слое для фона можно расположить какую-либо картинку, в том числе из тех, которые в дальнейшем должны появиться на экране. Также в этом слое можно написать общий заголовок. В данном случае будет: «Мой отдых в Европе». Во втором слое расположим 5 текстовых полей, все поля должны иметь атрибут **Dynamic**, для того, чтобы мы могли отобрать в них текстовые сообщения, но в то же время чтобы не было воз-

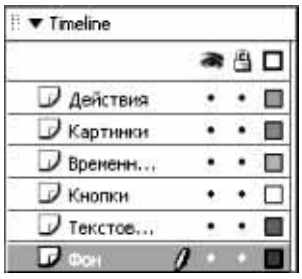


Рисунок 1.

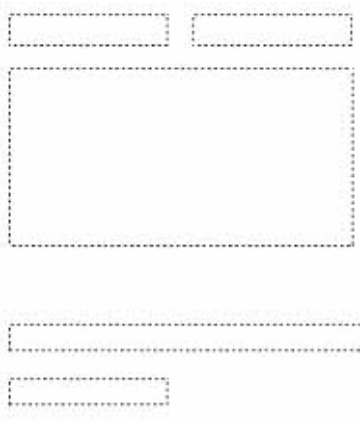


Рисунок 2.

Рисунок 3.

Рисунок 4.

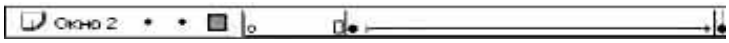


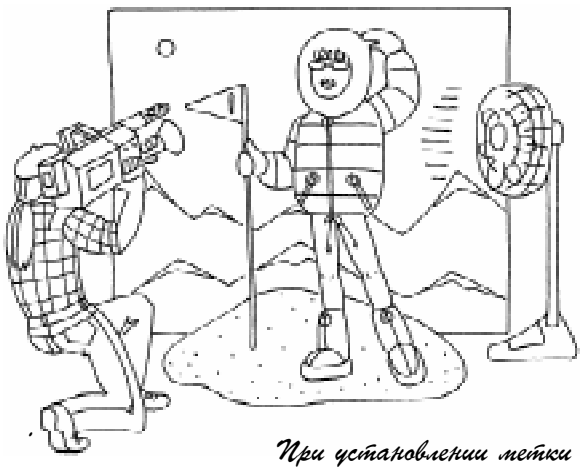
Рисунок 5.

можности редактировать оригинальные данные. Текстовым полям необходимо присвоить следующие **Instance name**: **`date_txt`**, **`country_txt`**, **`caption_txt`**, **`warning_txt`**, **`balloon_txt`** (рисунок 2).

В слое **Кнопки** создадим кнопки, при помощи которых будет осуществляться управление. Такими кнопками традиционно могут являться кнопки начала и остановки проигрывания, кроме них, установим кнопку перемотки в начало (рисунок 3).

Создадим Movie Clip (MC) для отображения последовательности смены картинок с именем **`indicator_mc`**. Этот MC должен состоять из четырех слоев, например, следующих: **Окно 1**, **Окно 2**, **Действия**, **Метки** (рисунок 4).

В слоях **Окно 1**, **Окно 2**, **Метки** создим ключевые кадры на 10 кадрах, а в слоях **Окно 2** и **Действия** потребуется создать ключевые кадры еще на 40 кадрах, продлим действие оставшихся слоев тоже до 40 кадров. В слое **Окно 1** во втором ключевом кадре расположим цветной прямоугольник, который для удобства преобразуем в MC. В слое **Окно 2** во втором и третьем ключевых кадрах разместим тот же MC, в третьем ключевом кадре зададим параметр MC – **`alpha`**, равное 0, и переместим MC таким образом, чтобы MC в слоях **Окно 1** и **Окно 2** совпадали. Для того чтобы движение в слое **Окно 2** было плавным, зададим анимацию **Motion** из второго ключевого кадра в третий (рисунок 5).



При установлении метки в тот или иной ключевой кадр должен появиться красный флажок.

В слое **Действия** необходимо остановить проигрывание клипа, чтобы тот не стал проигрываться с момента загрузки приложения. Для этого с незапамятных времен применения Flash служит команда:

```
stop();
```

После окончания проигрывания необходимо вернуться в начальное положение. Для этого в третий ключевой кадр слоя **Действия** необходимо написать команду, которая бы переводила в первый кадр:

```
gotoAndStop(1);
```

где 1 – это номер кадра, в который необходимо перейти, в данном случае это первый кадр.

В слое метки создадим две метки: в первом ключевом кадре – метку **`Off`**, а во втором – метку **`On`** (рисунок 6).

При установлении метки в тот или иной ключевой кадр должен появиться красный флажок. После задания меток стало возможным изменить скрипт, записанный в третьем ключевом кадре слоя **Действия**, на следующий:

Рисунок 6.

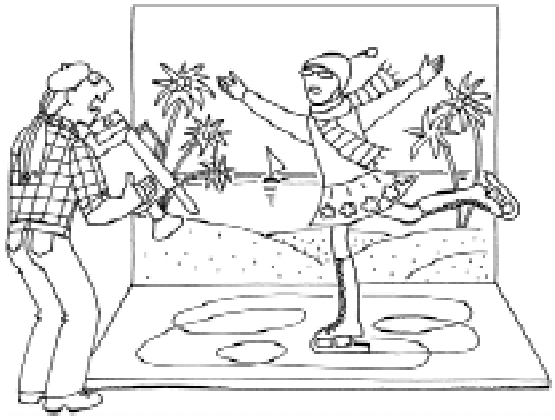
```
gotoAndStop("Off");
```

В данном случае **"Off"** – это название метки, на которую должен перейти ролик. Если вы следовали указаниям, то временная линейка должна выглядеть примерно так, как показано на рисунке 7.

После завершения работы по созданию этого МС вернемся на основную сцену.

В слое **Картинки** создадим МС с именем **pictures_mc**. Внутри данного МС создадим слой. Для удобства использования назовем первый слой – **Картинки** (в него мы будем помещать графические изображения), а второй слой – **Действия** (разумеется, для команд скрипта) (рисунок 8).

Перейдем в слой **Картинки** и создадим в нем еще 4 ключевых кадра, в каждый из них необходимо импортировать картинки (желательно, чтобы их размер не превышал размер сцены). В качестве графических форматов можно использовать наиболее распространенные форматы – jpg, png, gif, bmp. Как видно, среди форматов есть и jpg, то есть формат, в котором сохраняют свои файлы подавляющее большинство цифровых фотокамер, таким образом, существует возможность импортировать фотографии



...может оказаться что порядок фотографий перепутан местами.

во Flash – приложение непосредственно с цифрового фотоаппарата. Для того чтобы импортировать картинку, необходимо выбрать пункт меню **File** и далее **Import → Import to Stage** (файлы будут помещаться непосредственно на сцену) или **Import to Library** (файлы будут помещаться в библиотеку фильма). Возможно, разумнее пользоваться традиционным импортом файлов на сцену, тогда у вас есть возможность увидеть, что вы импортировали (рисунок 9).

После того как картинки импортированы и размещены в необходимых координатах, может оказаться, что порядок фотографий перепутан местами и, например, не соответствует хронологии событий. В таком случае существует возможность использовать команду **Swap Bitmap**, которая осуществляет перемену местами графических изображений. Для этого необходимо зайти в пункт меню **Modify → Bitmap → Swap**

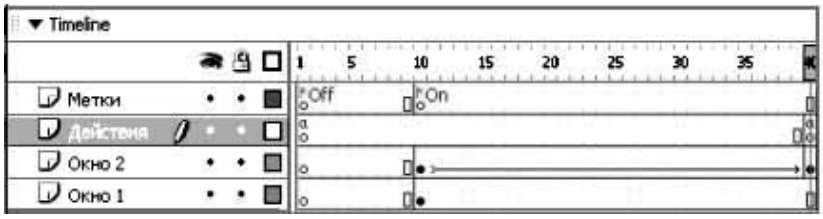


Рисунок 7.

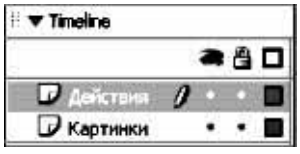


Рисунок 8.

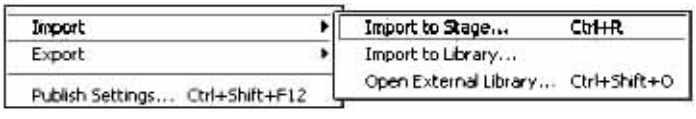


Рисунок 9.

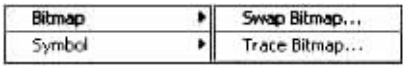


Рисунок 10.

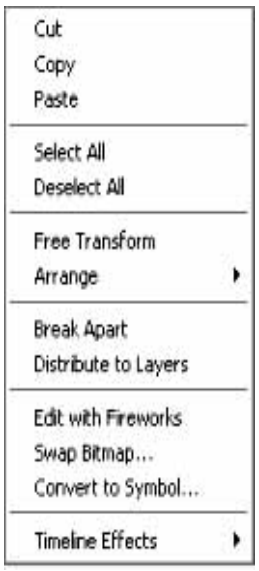


Рисунок 11.

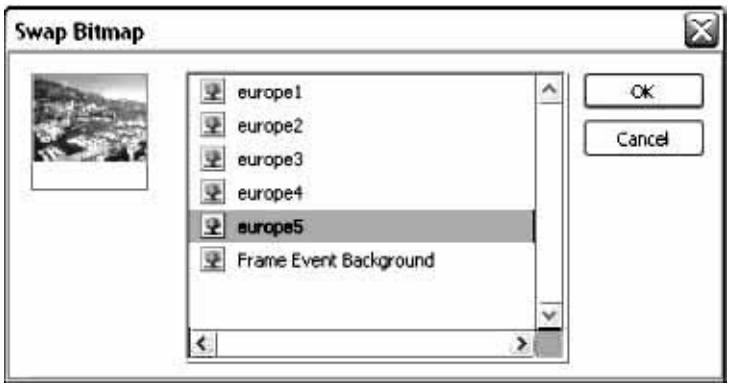


Рисунок 12.

Bitmap (рисунок 10).

Аналогичная команда есть, если щелкнуть правой клавишей мыши на картинке (рисунок 11).

Далее в окне **Swap Bitmap** можно выбрать просто файл с требуемым именем для замены (рисунок 12).

В слое **Действия** создадим 5 ключевых кадров, следующих друг за другом. Таким образом, временная линейка принимает вид, как на рисунке 13.

После того как подготовительные работы и с этим МС закончены, вернемся на сцену.

Продлим действия всех слоев до 200 кадров, а в слое **Действия** создадим ключевые кадры на 140, 160, 180, 200 кадрах. Теперь все работы по созданию заготовки закончены, и можно приступать к основным действиям по созданию приложения (рисунок 14).

Для удобства проигрывания установим скорость проигрывания в панели **Properties**, равную 20 кадрам в секунду (рисунок 15).

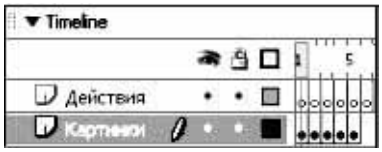
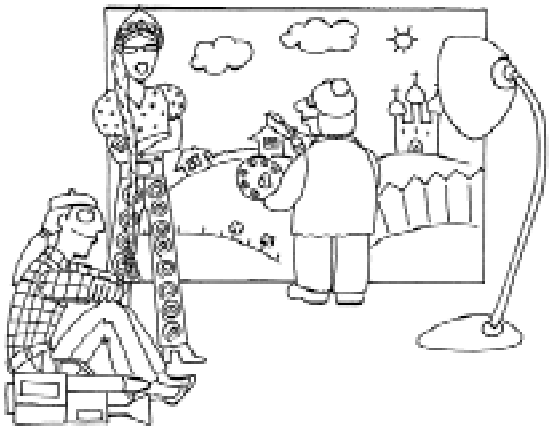


Рисунок 13.

Дважды щелкнем по картинке **pictures_mc**, которая находится примерно посередине сцены, чтобы редактировать ее, дополнив необходимыми программными элементами. Теперь давайте заполним эти пустые ключевые кадры сценариями. В открытой панели **Actions** выберем кадр 1 на уровне **Действия** и добавим следующий порядок команд (сценарий):

```
stop ();
_root.date_txt.text = "15 Июня";
_root.country_txt.text = "Эстония.
                                Нарва";
_root.caption_txt.text =
            "Симпатичное местечко";
```

Первая команда традиционно необхо-



Дважды щелкните по картинке pictures_mc, которая находится примерно посередине сцены, чтобы редактировать его, дополнив необходимыми программными элементами.

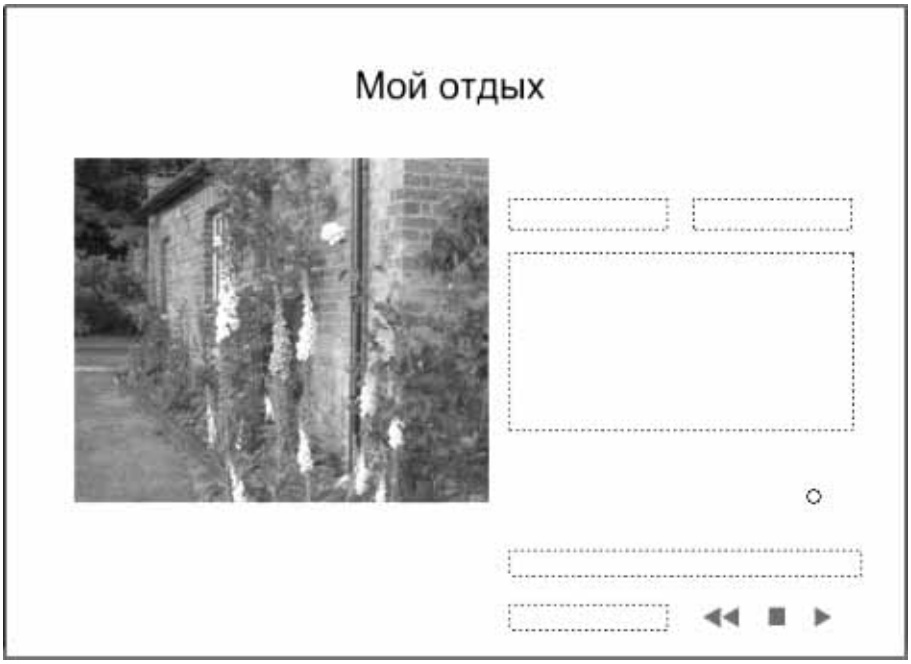


Рисунок 14.

дима, чтобы прекратить проигрывание клипа, как только ролик будет загружен, и, следовательно, пока мы не зададим команды, ролик будет оставаться неподвижным.

Следующие три действия размещают текст в текстовые поля с соответствующими именами на главной сцене (вы еще не забыли, что текстовые поля были созданы чуть раньше?). Эта текстовая информация касается графического символа, который появляется на кадре 1 из уровня **Картинки** на этом графике времени. Другими словами, всякий раз, когда этот график времени находится на этом кадре, видимое изображение и текст, отображенный в текстовых полях, совпадут друг с другом.

С открытой панели **Actions** выберем Кадр 2 на уровне **Действий** и добавим этот сценарий:

```
_root.date_txt.text = "16 Июня";
_root.country_txt.text = "Пярну";
_root.caption_txt.text = "Обед был
                          прекрасен";
```

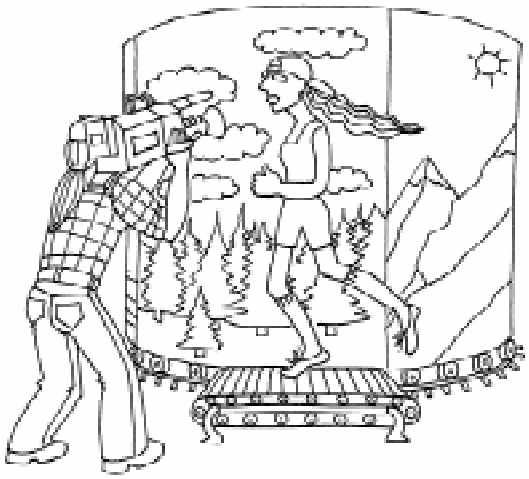
Когда график времени во время проигрывания приложения дойдет до второго ключевого кадра, эти три действия произведут обновление текстового содержимого в соответствующих текстовых полях на сцене в главном графике времени. Эта текстовая информация связана с появляющимся во втором ключевом кадре графическим символом в слое **Картинки** на временной линейке внутри данного МС.

С открытой панелью **Actions** выберем третий, четвертый и пятый ключевые кадры на слое **Действия**, в которые необходимо добавить следующие сценарии, соответственно:

В третьем ключевом кадре:

```
_root.date_txt.text = "17 Июня";
_root.country_txt.text = "Пярну";
_root.caption_txt.text = "Отдых
                          на море прекрасен";
```

Рисунок 15.



...циклы демонстрации проходят через пять ключевых кадров до остановки...

В четвертом ключевом кадре:

```
_root.date_txt.text = "18 Июня";
_root.country_txt.text = "Пирита";
_root.caption_txt.text = "Снова
                прогулка по морю";
```

В пятом ключевом кадре:

```
_root.date_txt.text = "19 Июня";
_root.country_txt.text = "Monaco";
_root.caption_txt.text = "Меня тут
                еще не было";
```

Каждый из этих наборов команд приводит к тому же самому эффекту, что и предыдущие два набора; единственное различие – то, что они будут вызваны, когда график времени этого ролика перейдет к ключевым кадрам 3, 4, 5, соответственно (рисунок 16).

С открытой панели **Actions** выберем шестой ключевой кадр на уровне Действия и добавим этот сценарий:

```
gotoAndStop (1);
```

При движении по временной линейке для удобства использования динамической адресации в данном примере не используются метки кадров. В конце, когда MC достигнет последнего ключевого кадра, ему будет дана команда перейти на первый кадр

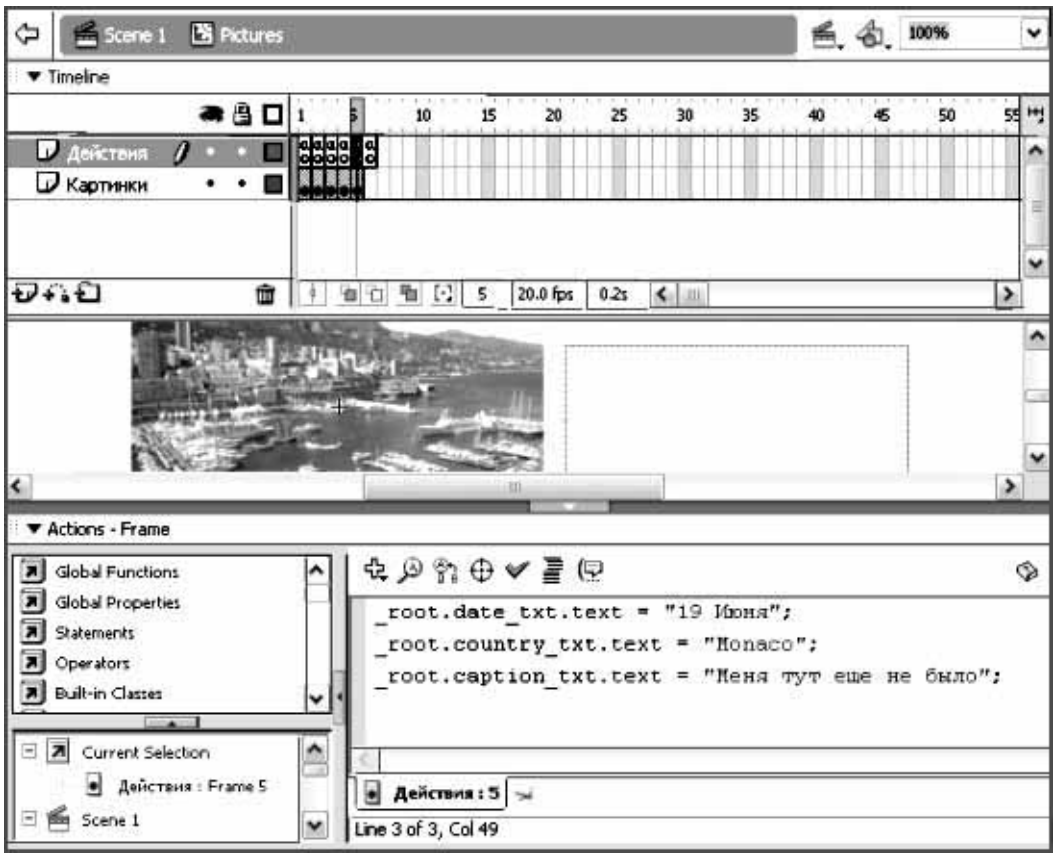


Рисунок 16.

и отобразить первую фотографию с соответствующим текстом, после чего остановиться. Таким образом, наши циклы демонстрации проходят через пять ключевых кадров до остановки или остановка будет возможна по требованию пользователя.

Теперь давайте зададим функциональные возможности, которые и заставят двигаться МС с фотографиями по нашему усмотрению.

Возвращаемся к главному графику времени, перейдя на Сцену 1. В панели **Actions** выберем ключевой кадр на кадре 140 в слое **Действия** и добавим следующий скрипт:

```
warning_txt.text = "Картинка сменится
                    через 3 секунды.";
indicator_mc.gotoAndPlay ("On");
```

В конечном счете ключевой кадр на двухсотом кадре в слое **Действия** будет содержать сценарий, чтобы произвести временной сдвиг в МС (**pictures_mc**), который содержит изображения. Так как наш проект установлен, чтобы проиграть 20 кадров в секунду, помещаем этот сценарий на 140 кадре, что заставит это быть выполненным через три секунды (200 кадров минус 140 кадров равняется 60 кадрам, или 3 секундам) до изменения картинки и текста на экране.

Первое действие отображает предупреждающее сообщение в текстовом поле с именем **warning_txt**, указывающее, что изображение изменится через три секунды. Следующее действие указывает МС с именем **indicator_mc**, в котором у нас есть наползающие друг на друга окошки, показывающие, как картинки меняются между собой, перейти на ключевой кадр с меткой **"On"**. В этом МС короткая анимация действует как визуальная команда вызова программы, которую изображение собирается изменять (чего на самом деле не происходит, но иллюзия создается).

В ключевом кадре, расположенном на 160 кадре в слое **Действия**, необходимо указать следующий скрипт:

```
warning_txt.text = "Картинка сменится
                    через 2 секунды.";
```

Это действие просто обновляет сообщение в поле текста предупреждения, чтобы указать, что изображение будет изменяться через две секунды.

В панели **Actions** выберем ключевой кадр на кадре 180 в том же слое **Действия** и добавим следующий сценарий:

```
warning_txt.text = "Картинка сменится
                    через 1 секунду.";
```

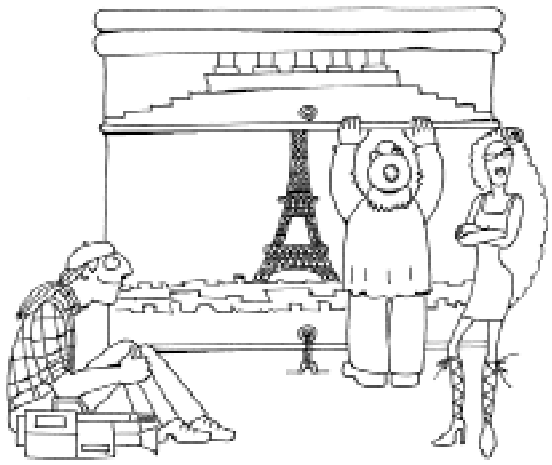
Это действие в последний раз перед появлением картинки обновит текстовое поле с предупреждением, чтобы указать, что изображение будет изменяться через одну секунду.

В панели **Actions** выберем слой **Действия** и ключевой кадр на кадре 200, в который добавим следующий сценарий (рисунок 17):

```
pictures_mc.nextFrame();
warning_txt.text = "";
gotoAndPlay (1);
```

Эти действия представляют основу наших действий по отображению графического и текстового содержимого. Если действия вывода изображений на экран вместе со связанным с ними текстом в течение 10 секунд можно назвать циклом, то эти действия будут выполнены в конце каждого из проходов цикла.

Первое действие вызывает переход внутри МС с фотографиями на следующий ключевой кадр и останавливает это проиг-



...у нас есть наползающие друг на друга окошки, показывающие, как картинки меняются между собой...

рывание, чтобы картинки не сменялись без нашего ведома. Выполнение заставит следующее изображение быть отображенным, кроме этого, будет выполнен тот скрипт, который указан на соответствующем ключевом кадре.

Следующее действие очищает весь текст от поля текста предупреждения, так как стадия предупреждения о появлении нового изображения закончена. Последнее действие посылает главный график времени назад на первый кадр таким образом, чтобы процесс проигрывания приложения не прерывался, а продолжался циклически.

Чтобы позволить пользователю управлять воспроизведением представления, с открытой панелью **Actions** выберем треугольную кнопку на панели управления и добавим следующий скрипт:

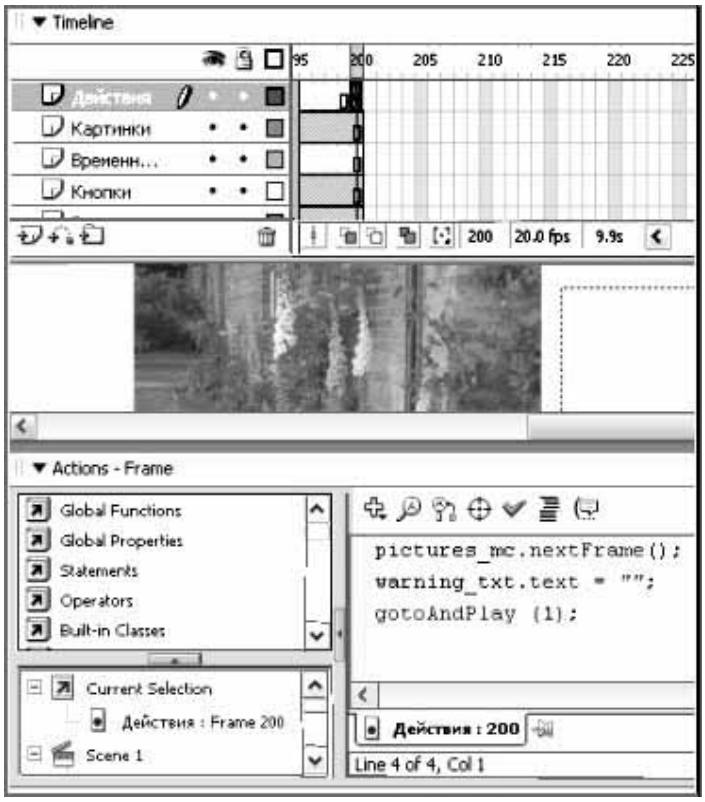


Рисунок 17.

```
on (release) {
     play ();
}

on (rollOver) {
     balloon_txt.text = "Play";
}

on (rollOut) {
     balloon_txt.text = "";
}
```

Эта кнопка среагирует на три события: когда она нажата и отпущена, главный график времени будет проигрывать свое содержимое и пройдет по всем выше описанным ключевым кадрам; когда курсор мышки просто наведен на кнопку, то в текстовом поле появится текст о предназначении данной кнопки, а когда курсор мышки будет убран с кнопки, то поле сообщения будет очищено.

С открытой панелью **Actions** выберем квадратную кнопку на панели управления и добавим следующий сценарий:

```
on (release) {
     stop ();
}

on (rollOver) {
     balloon_txt.text = "Stop";
}

on (rollOut) {
     balloon_txt.text = "";
}
```

Эта кнопка также установлена, чтобы ответить на три события от нажатия мыши: когда кнопка нажата и отпущена, главный график времени остановится, когда курсор мышки наведен на нее, в поле текста подсказки будет отображено слово **Stop**, и, когда пользователь уводит курсор мышки от кнопки, поле текста подсказки будет очищено.

В панели **Actions** выберем двойную треугольную кнопку на панели управления и добавим следующий скрипт:

```
on (release) {
     gotoAndPlay (1);
     pictures_mc.gotoAndStop (1);
     warning_txt.text = "";
}
on (rollOver) {
     balloon_txt.text = "Rewind";
}
on (rollOut) {
     balloon_txt.text = "";
}
```

Когда эта кнопка нажата и отпущена, главный график времени и МС, содержащий картинки, перейдут назад, к кадру с номером 1. Это сбрасывает все переходы между картинками и переводит приложение в первоначальное состояние, независимо от того, как далеко зашел процесс проигрывания картинок с текстом. Следующее действие отобразит слово **Rewind** (перемотка) в поле текста подсказки, когда курсор мышки будет наведен на кнопку. Если убрать курсор мышки, поле текста подсказки будет очищено.

В меню выбираем **Control → Test Movie**, чтобы видеть кино в действии.

Рассмотрите представление от начала до конца, чтобы получить удовлетворение от того, как это работает. Используйте кнопки управления, чтобы управлять воспроизведением. Вы теперь завершили это упражнение.

Рассмотрите представление от начала до конца

Материалы, представленные в статье, используются автором в учебном процессе Санкт-Петербургского регионального центра Федерации Интернет Образования.

Штенников Дмитрий Геннадьевич,
доцент кафедры физики
СПбГУ ИТМО.