Министерство образования и науки Украины

Севастопольский национальный технический университет

РАСШИРЕННЫЕ ВОЗМОЖНОСТИ WEB-ТЕХНОЛОГИЙ

Методические указания
к выполнению лабораторных работ № 6 – 7
по дисциплине «WEB-технологии»
для студентов специальности 7.091401
«Системы управления и автоматики»
дневной и заочной форм обучения

Севастополь
2007

УДК 681.3

Расширенные возможности web-технологий: Методические указания к выполнению лабораторных работ № 6 – 7 по дисциплине «WEB-технологии» для студентов специальности 7.091401 «Системы управления и автоматики» дневной и заочной форм обучения / Разраб. В.В. Альчаков, В.А. Крамарь. – Севастополь: Изд-во СевНТУ, 2007. – 20 с.

Целью методических указаний является оказание помощи студентам при выполнении лабораторных работ по дисциплине «WEB-технологии». В указаниях описываются расширенные возможности web-технологий по созданию интерактивной анимации и трехмерных моделей для Internet.

Методические указания предназначены для студентов специальности 7.091401 «Системы управления и автоматики» дневной и заочной форм обучения.

Методические указания рассмотрены на заседании кафедры технической кибернетики, протокол № 4 от « 23 » февраля 2006г.

Допущено учебно-методическим центром и научно-методическим советом СевНТУ в качестве методических указаний.

Рецензент

Карапетьян В.А. канд. техн. наук, доцент кафедры ТК

СОДЕРЖАНИЕ

1. ЛАБОРАТОРНАЯ РАБОТА № 6. СОЗДАНИЕ ИНТЕРАКТИВНОЙ АНИМАЦИИ ДЛЯ INTERNET

1.1 Цель работы

Изучить возможности пакета Macromedia Flash для подготовки интерактивной web-анимации. Научиться создавать простейшие анимационные ролики и внедрять их в HTML-документ.

1.2 Краткие теоретические сведения

Пакет Macromedia Flash предназначен для создания web-анимации. Он ориентирован на работу с векторной графикой. Flash-ролики могут сопровождаться видео и аудио фрагментами.

Применение Flash для создания HTML-документов. Возможны следующие варианты применения Flash: 1) в документе используются традиционные элементы, однако, имеются Flash вставки; 2) весь документ построен на Flash.

Следует отметить, что Flash не оптимизирован для работы с большими фрагментами текста, поэтому для HTML-документов с значительным текстовым содержанием рекомендуется применение обычной HTML-верстки, т.е. предпочтителен первый вариант. С другой стороны, для HTML-документов с интенсивным использованием медиаданных (графика, звук, анимация) второй вариант подходит больше.

Рассмотрим следующие возможности Flash:

- создание графического объекта, перемещающегося по экрану;
- создание анимированного текста;
- создание анимированной кнопки.

Создание графического объекта, перемещающегося по экрану. Рассмотрим некоторые ключевые понятия Flash. Здесь и далее под *объектом* (символом) будем понимать многократно повторяющееся изображение, анимацию или кнопку. Экземпляр объекта – копия объекта, изменение которой (форма, цвет, и т.д.) не влияет на сам объект, в то же время изменение объекта влечет за собой изменение всех его экземпляров. Под *анимацией* будем понимать изменение содержания последовательно расположенных кадров. Во Flash существуют два способа создания анимации – покадровый (*Frame-by-Frame*) и автоматический (*Tweened*). При покадровой анимации необходимо определить изображения для каждого кадра вручную. В случае автоматического создания ролика достаточно определить начальное и конечное положение и состояние объекта в первом и последнем кадре ролика соответственно. Все промежуточные кадры будут сгенерированы программой автоматически. В свою очередь автоматическая анимация делится на следующие два вида:

1) движение (*Motion-Tweening*) – изменение положения объекта на рабочем поле;

2) превращение (*Shape-Tweening*) – изменение формы и размеров объекта.

Объект можно создать (нарисовать) непосредственно во Flash, импортировать из другого графического редактора, либо воспользоваться библиотекой готовых объектов, входящей в состав пакета.

Создадим произвольное изображение, используя стандартные инструменты рисования Flash. Внешний вид панели рисования приведен на рисунке 1.1. Для тех, кто когда-либо работал с графическими редакторами, не составит труда быстро адаптироваться к рисованию во Flash.

Рисунок 1.1 – Панель инструментов рисования Flash

Для того чтобы созданное изображение стало объектом, необходимо выполнить следующие действия:

- выделите все изображение (для этого щелкните правой кнопкой мыши на изображении и в появившемся контекстном меню выберете пункт **Select All** или выполните двойной щелчок над областью изображения);

- выберете пункт меню **Insert|Convert to Symbol** или нажмите клавишу **F8**. В появившемся диалоговом окне **Symbol Properties** задайте произвольное имя для объекта в поле **Name**, а для поля **Behavior** выберете тип **Graphic**.

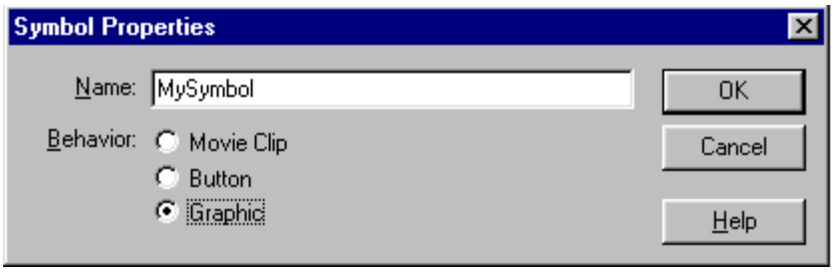

Рисунок 1.2 – Панель задания свойств объекта

После выполнения указанных действий изображение будет восприниматься как единый объект.

Если теперь открыть окно библиотеки **Window|Library**, то в поле **Name** можно увидеть имя только что созданного объекта. Щелкните мышью на имени объекта. В окне библиотеки появиться соответствующее ему изображение. Для того чтобы создать экземпляр объекта, достаточно просто перетащить его изображение из окна библиотеки в рабочую область.

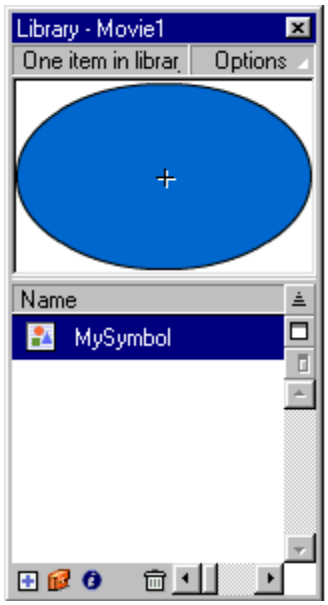

Рисунок 1.3 – Окно библиотеки объектов

Для создания нового Flash ролика используется команда **File|New**. Определить размеры рабочего поля, цвет фона и некоторые другие параметры ролика можно с помощью пункта меню **Modify|Movie**. В открывшемся диалоговом окне **Movie Properties** следует указать соответствующие значения параметров. Активизируйте первый кадр ролика на временной линейке и перетащите на рабочую поверхность экземпляр класса. В результате получится первое ключевое изображение, и вы зададите начальную точку своей анимации.

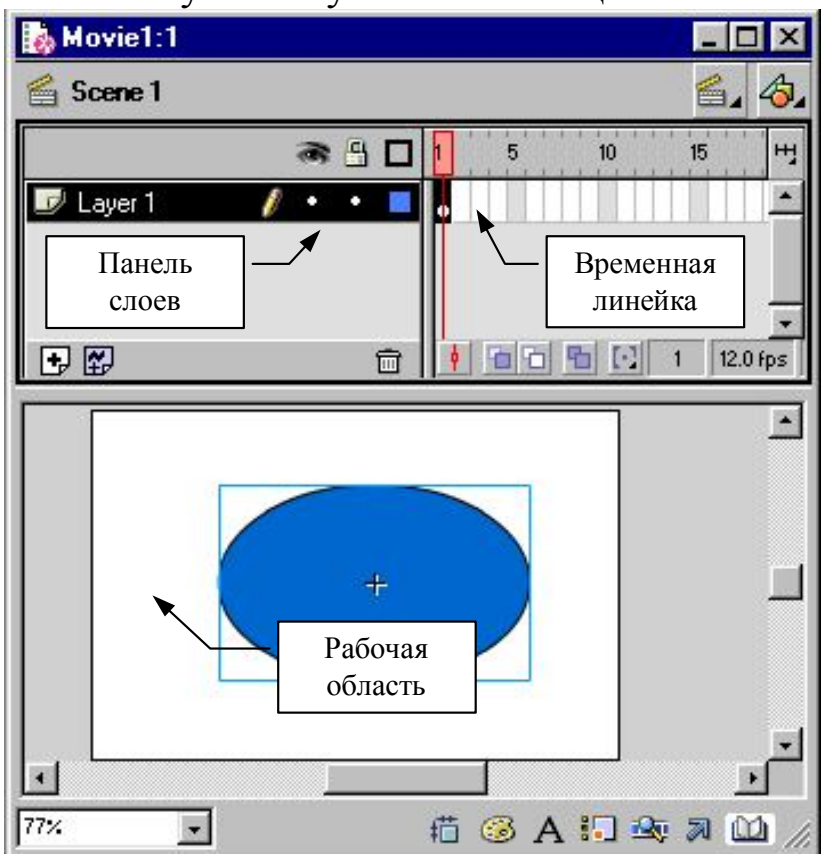

Рисунок 1.4 – Рабочее окно ролика

Допустим, что ролик будет состоять из пятнадцати изображений. Щелкните на пятнадцатом кадре и выполните команду **Insert|Keyframe**. Тем самым вы вставите в ролик второе ключевое изображение. Переместите второе ключевое изображение в любое место рабочего поля. Теперь необходимо, чтобы Flash сгенерировал промежуточные изображения между первым и пятнадцатым кадром. Для этого необходимо вновь активизировать первое изображение (первый кадр) и вызвать окно **Frame** с помощью команд **Window|Panels|Frame**. Откройте закладку **Frame** и под записью **Tweening** в выпадающем списке выберите строку Motion. На временной шкале кадры с первого по пятнадцатый будут выделены фиолетовым цветом, и от первого кадра к пятнадцатому будет проведена стрелка. Это указывает на то, что кадры созданы.

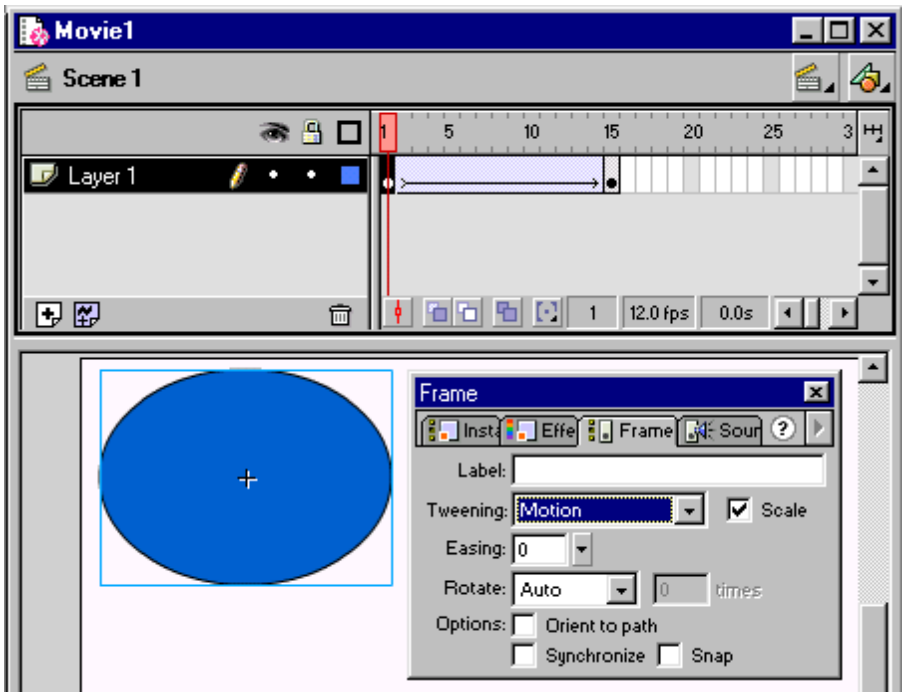

Рисунок 1.5 – Внешний вид окна после автоматической генерации кадров

Просмотреть ролик можно с помощью пункта меню **Control|Test Movie**.

Для того чтобы заставить объект двигаться по заданной траектории необходимо выполнить следующие действия. Щелкните правой кнопкой мыши по слою ролика, в котором расположен объект. В появившемся меню выберите пункт **Add Motion Guide**. Сделайте новый слой активным (щелчок мыши на названии слоя) и затем, с помощью инструмента «карандаш», нарисуйте на рабочей поверхности требуемую траекторию. Чтобы объект перемещался по созданной кривой, необходимо перейти в слой объекта, пометить первый кадр ролика и с помощью инструмента **Arrow Tool** (черная стрелка на панели **Tools**) перетащить объект за его

центр к началу траектории. При этом должна быть включена опция привязки к объектам – **View|Snap to Objects**. Центр объекта свяжется с конечной точкой траектории. Те же самые действия необходимо выполнить для последнего ключевого кадра ролика. В окне **Frame (Window|Panels|Frame)** для поля **Tweening** выберите значение **Motion**.

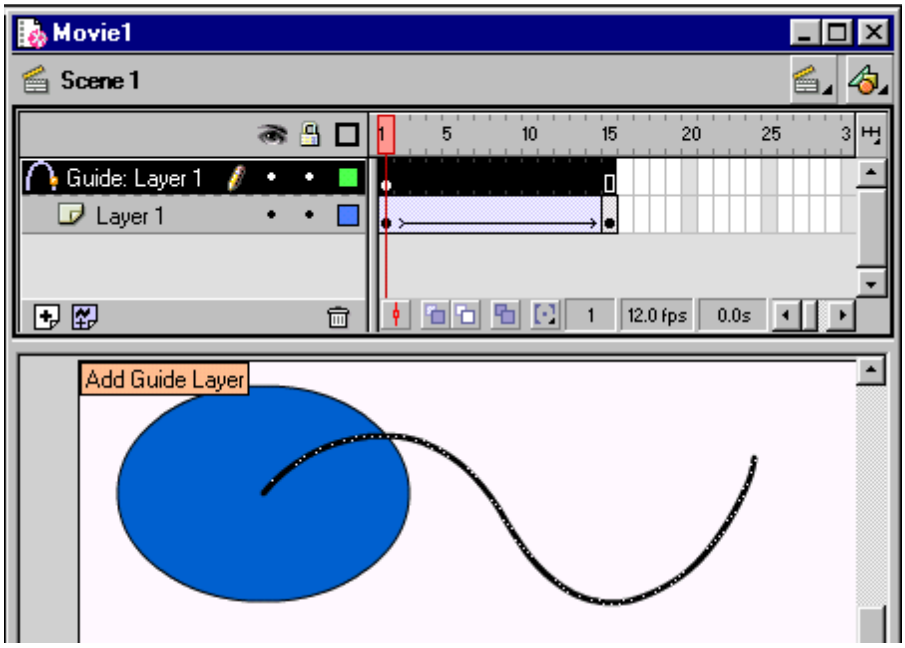

Рисунок 1.6 – Работа со слоем траектории

После этого объект будет двигаться по заданному пути. Для того чтобы объект двигался не прямолинейно необходимо включить опцию **Orient to path** в окне **Frame**.

Создание анимированного текста. Используя Flash можно создать текст, который может динамически изменять свой размер, цвет, положение и т.д. Для работы с Tweening анимацией необходимо, чтобы текст воспринимался как символ.

Создадим произвольный текстовый заголовок. Для этого на панели **Tools** необходимо выбрать инструмент **Text Tool** и в рабочей области выделить участок, в котором будет помещена надпись. После этого введите последовательность символов. Перед тем как создать анимационные эффекты, преобразуем текст в символ. Выделите текст с помощью инструмента **Arrow Tool** и выполните команду **Insert|Convert to Symbol…** В появившемся окне **Symbol Properties** для поля **Behavior** определите значение **Graphic**. Выделите последний кадр анимации и выполните команду **Insert|Keyframe**. Теперь можно определить свойства начального и конечного изображений текста. Выделите первый кадр анимации и выполните команду **Window|Panels|Effects**.

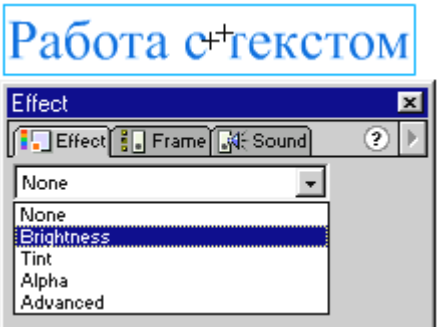

Рисунок 1.7 – Установка свойств объекта

В выпадающем списке выберите одно из возможных свойств: None – нет эффектов, Brightness – яркость, Tint – цвет, Alpha – прозрачность, Advanced – расширенный выбор цвета и прозрачности.

Установкой значений параметров каждого из свойств можно добиться необходимого эффекта для нашего объекта (текста). К примеру, для изменения прозрачности текста установите для объекта в первом кадре ролика значение параметра Alpha равным 10%, а для последнего – 100%. После создания анимации (см. ниже), текст из прозрачного будет постепенно становиться насыщенным.

Для изменения размеров текста следует выделить объект с помощью **Arrow Tool** и щелкнуть правой кнопкой мыши, в появившемся меню выбрать команду **Scale**, после этого произвести необходимые преобразования. Аналогично можно поворачивать объект с помощью команды **Rotate**.

После того, как произведены изменения начального и конечного кадра ролика, необходимо сделать активным первый кадр и во вкладке **Frame** из списка **Tweening** выбрать пункт **Motion**. Выполните команду **Insert|Motion Tween** и посмотрите полученный ролик.

Создание анимированной кнопки. Во Flash кнопки также относятся к категории символов, поэтому для создания новой кнопки необходимо создать новый символ, выполнив команду **Insert|New Symbol**. В появившемся диалоговом окне следует указать имя для создаваемого объекта, например, MyButton, а для поля **Behavior** указать тип **Button**. После этого программа переходит в режим конструирования кнопки.

Рисунок 1.8 – Создание кнопки

Ролик для кнопки состоит из четырех кадров:

- **Up** – отвечает за состояние кнопки «Выключено»;
- **Over** – отвечает за состояние «Указатель мыши над кнопкой»;
- **Down** – отвечает за состояние «Нажатие кнопки»;
- **Hit** – отвечает за состояние кнопки после нажатия.

Нарисуйте с помощью стандартных средств рисования кнопку произвольной формы, при необходимости можно добавить текст. По умолчанию активен кадр, отвечающий за выключенное состояние кнопки. Чтобы определить остальные состояния кнопки, необходимо вставить в соответствующие кадры изображения (**Insert|Keyframe**) и провести необходимые изменения. После того, как работа над кнопкой закончена, необходимо вернуться в режим редактирования ролика, выполнив команду **Edit|Edit Movie**. Откройте окно библиотеки **Window|Library** и перетащите на рабочую поверхность созданную кнопку. Для просмотра эффекта работы кнопки используйте команду **Control|Test Movie**.

Сопоставим нажатию кнопки некоторое событие, например, открытие некоторого HTML-документа в новом окне браузера. Для этого необходимо выполнить следующие действия. Выделите кнопку и откройте окно **Window|Actions**, после чего выберите вкладку **Object Actions**. В левой части окна находятся команды на языке ActionScript. Щелкните на категории **Basic Actions** и при нажатой левой клавише мыши перетащите действие экземпляра **OnMouseEvent** в список действий справа. Пометьте запись **on (release)** в окне справа. В нижней части окна **Object Actions** можно выбрать различные виды событий, связанных с мышью. Снимите галочку напротив пункта **Release** и установите ее напротив пункта **Press**. Далее в категории **Basic Actions** выберите команду **get URL** и перетащите ее в список действий справа. В нижней части диалогового окна **Object Actions** в поле **URL** укажите адрес странички.

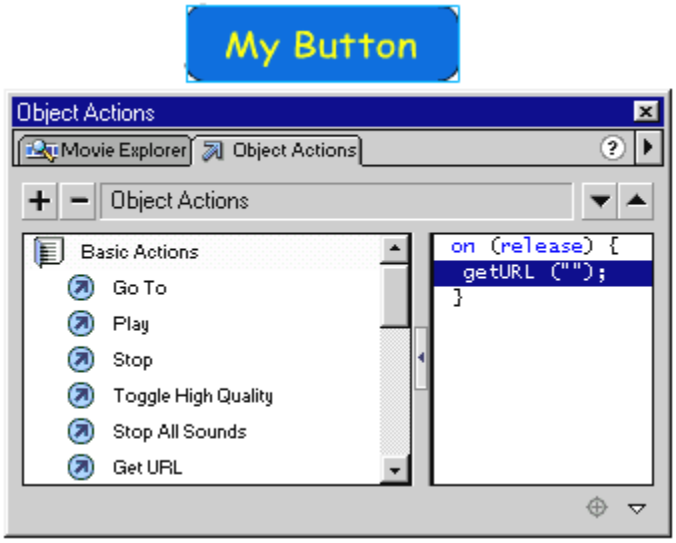

Рисунок 1.9 – Назначение обработчика для кнопки

Для того чтобы страничка открылась в новом окне браузера, в поле **Window** выберите пункт **_blank**. Просмотрите результат с помощью команды **Control|Test Movie**.

Публикация ролика. Публикация ролика подразумевает создание Shockwave-файла (расширение *.swf), предназначенного для размещения в Internet. Для этого необходимо встроить swf-файл в HTML-документ. Делается это следующим образом. Сохраните ролик на жестком диске. Откройте пункт меню **File|Publish Settings…** и в появившемся окне установите галочки напротив пунктов **Flash** и **HTML**. Нажмите кнопку **Publish**. Программа произведет экспорт ролика в выбранные форматы. После этого необходимо лишь вставить в html-код фрагмент, который будет содержать код для открытия Flash-ролика.

1.3 Порядок выполнения работы

Выполнение лабораторной работы состоит из следующих шагов:

1. Изучение основных приемов работы с пакетом Macromedia Flash.
2. Создание Flash-роликов и интеграция их в html-страницы.
3. Оформление отчета.
4. Защита лабораторной работы.

1.4 Задание на лабораторную работу

Подготовить Flash-ролики (tweened-анимацию объекта, перемещающегося по заданной траектории, анимированный текстовый заголовок страницы, анимированные кнопки для навигации по

документам) и встроить их в HTML-документы, разработанные при выполнении лабораторных работ № 2 – 5.

1.5 Содержание отчета

Отчет оформляется в соответствии с требованиями, предъявляемыми к оформлению лабораторных работ в ВУЗе, и должен содержать:

1. Титульный лист.
2. Формулировку цели работы.
3. Постановка задачи в соответствии с заданием.
4. Краткие теоретические сведения и описание этапов выполнения работы.
5. Исходные коды web-страниц.
6. Выводы по работе.

Отчет готовиться каждым студентом индивидуально. При защите работы необходимо наличие электронной версии HTML-документов.

1.6 Контрольные вопросы

1. Что такое Flash-ролик?
2. Какие возможности предоставляет пакет Macromedia Flash?
3. Какие виды анимации вам известны?
4. Как создать анимацию – движение объекта по заданной траектории?
5. Как создать анимированный текст?
6. Как создать анимированную кнопку?
7. Как назначить кнопке обработчик?
8. Как произвести публикацию Flash-ролика в Internet?

2. ЛАБОРАТОРНАЯ РАБОТА № 7. СОЗДАНИЕ ТРЕХМЕРНЫХ МОДЕЛЕЙ С ИСПОЛЬЗОВАНИЕМ ЯЗЫКА VRML

2.1 Цель работы

Изучить основные синтаксические конструкции языка VRML. Научиться создавать трехмерные модели, предназначенные для публикации в Internet.

2.2 Краткие теоретические сведения

Введение. VRML (Virtual Reality Modeling Language) – это специальный язык, предназначенный для описания трехмерного изображения в виде образующих сложную структуру объектов. VRML также включает в себя возможность задать источники света, туман и такие характеристики материала поверхностей, как отражающая способность, степень прозрачности и цвет.

Преимущество использования VRML заключается в том, что для его использования не требуется использование мощных графических станций, с появлением этого языка возникла возможность представлять в сети сложные трехмерные изображения с помощью простых команд. Эти команды описывают специальные эффекты и объекты в виде различных геометрических фигур для придания изображению реалистичности, имитации окружающей обстановки и освещения. Как и HTML VRML не нуждается в высокой пропускной способности каналов связи и не зависит от платформы. Следует обратить внимание на тот факт, что VRML-документ не может быть помещен внутрь HTML-документа, VRML является альтернативой, а не дополнением HTML.

VRML-документ – это текстовый файл, который содержит описания трёхмерных фигур и свойств их поверхностей (цвет, текстура материала, освещение и т.п.). VRML-документ запрашивается с web-сервера и поступает пользователю в виде исходного текста, точно так же, как и HTML-документ. Браузер, просматривающий VRML-документ и преобразующий при этом его текст в трёхмерную графику, должен иметь VRML-plugin (http://www.parallelgraphics.com/products/cortona/). Важно отметить, что VRML является интерпретируемым языком, который описывает структуру и оставляет визуализацию объектов за машиной, на которой изображается сцена. Это значит, что при каждом изменении точки зрения пользователя интерпретируется программа VRML,

рассчитывается геометрия нового изображения, вычисляются эффекты поверхностей и освещения и только после этого отображается результат.

Система координат и единицы измерения. Перед тем как перейти к рассмотрению способов задания трехмерных фигур, обратимся к системе координат с которой работает VRML. В VRML используется правосторонняя Декартова координатная система. Для простоты рассмотрим большой, указательный и средний пальцы на правой руке человека. Поставьте локоть правой руки на стол так, чтобы вы видели ладонь и зафиксируйте указанные пальцы перпендикулярно друг другу. Большой палец – это ось x, указательный – ось y, средний – ось z.

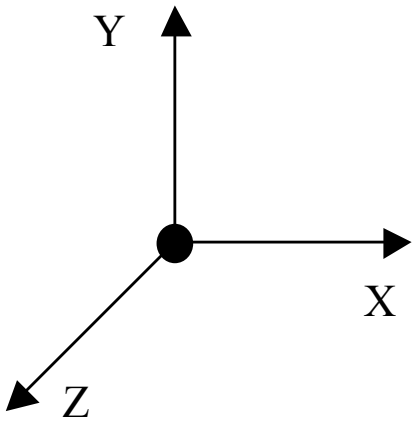

Рисунок 2.1 – Координатная система VRML

Значения по оси x увеличиваются слева направо, по оси y – снизу вверх и по оси z – по направлению к наблюдателю: чем больше значение по оси z, тем ближе точка расположена к наблюдателю. Длина и расстояние в этой системе измеряются в метрах, а углы в радианах.

Заголовок VRML-документа. Как уже говорилось, VRML-документ представляет собой обычный тестовый файл с расширением .wrl. Для того, чтобы VRML-браузер распознал файл с VRML-кодом, в начале файла ставится специальный заголовок – file header:

```
#VRML V2.0 utf8
```

Символ # используется для комментариев, однако для первой строки это правило не действует.

Такой заголовок обязательно должен находиться в первой строке файла, кроме того, перед знаком диеза не должно быть пробелов.

Примитивы VRML. В VRML определены четыре базовые фигуры: куб, сфера, конус и цилиндр. Эти фигуры называются примитивами. Набор примитивов невелик, однако, комбинируя их, можно строить достаточно сложные трехмерные изображения. Описание функций для создания примитивов приведено в таблице 2.1.

Таблица 2.1 – Основные примитивы VRML

Примитив	**Синтаксис**	**Описание**
Параллелепипед	`Box {size width heigh depth }`	Используется для построения прямоугольного параллелепипеда. Поля **width**, **height** и **depth** описывают ширину, высоту и глубину параллелепипеда, измеряемые от его центра. По умолчанию браузер центрирует куб в начале координат (0,0,0) с размерами две единицы по каждому из направлений, т.е. в пределах от -1 до +1.
Сфера	`Sphere { radius 1 }`	Используется для построения сферы. Параметр **radius** определяет размер сферы. По умолчанию сфера располагается в начале координат.
Конус	`Cone { bottomRadius 2 height 1 }`	Используется для построения конуса. Параметр **bottomRadius** отвечает за размер основания конуса. Параметр **height** – высота конуса.
Цилиндр	`Cylinder { radius 1 height 2 }`	Используется для построения цилиндра. Параметр **radius** – радиус основания цилиндра; **height** – высота цилиндра.

Пример 1: Конус с радиусом основания 0,5 и высотой 2

```
#VRML V2.0 utf8
Cone {height 2 bottomRadius 0.5}
```

Пример 2: Прямоугольный параллелепипед

```
#VRML V2.0 utf8
Box {size 1 2 3}
```

Пример 3: Цилиндр с радиусом основания 0,5 и высотой 1

```
#VRML V2.0 utf8
Cylinder {radius 0.5 height 1}
```

Рассмотрим следующий пример:

```
#VRML V2.0 utf8
DEF BOX Shape {
    appearance Appearance {
        material Material {
        }
    }
    geometry Box {
    }
}
```

Зарезервированное слово DEF используется для того, чтобы дать имя узлу (геометрическому объекту) для последующего использования. Теперь будет просто построить аналогичный куб, смещенный в пространстве. Объект Appearance используется для задания свойств материала, текстуры и т.д. Объект Material задает световые характеристики материала.

```
#VRML V2.0 utf8
DEF BOX Shape {
    appearance Appearance {
        material Material {
        }
    }
    geometry Box {
    }
}
Transform {
    translation 1 1 -1
    children [
        USE BOX
    ]
}
```

Другой пример показывает как создать пересекающиеся куб и цилиндр:

```
#VRML V2.0 utf8
DEF BOX Shape {
     appearance Appearance {
          material Material {
          }
     }
     geometry Box {
     }
}

DEF CYL Shape {
     appearance Appearance {
          material Material {
          }
     }
     geometry Cylinder {
     }
}

Transform {
     translation 1 1 -1
     children [
          USE CYL
     ]
}
```

Вставка Transform позволяет задать смещение цилиндра относительно параллелепипеда.

2.3 Порядок выполнения работы

Выполнение лабораторной работы состоит из следующих шагов:

1. Изучение теоретических сведений о VRML.
2. Изучение инструментальных средств для создания VRML-сцен.
3. Создание простейшей трехмерной сцены согласно варианту задания.
4. Оформление отчета.
5. Защита лабораторной работы.

2.4 Задание на лабораторную работу

В соответствии с вариантом задания построить трехмерную сцену, представляющую собой расположенные в пространстве геометрические фигуры.

Таблица 2.2 – Варианты заданий

№ Варианта	**Геометрические фигуры, входящие в сцену**
1	Куб, сфера, конус
2	Сфера, конус, цилиндр
3	Цилиндр, куб, сфера
4	Куб, цилиндр, конус
5	Цилиндр, сфера, конус
6	Цилиндр, куб, цилиндр
7	Куб, сфера, сфера
8	Куб, конус, куб
9	Конус, сфера, конус
10	Конус, куб, конус

Вариант задания определяется по последней цифре номера зачетной книжки студента.

2.5 Содержание отчета

Отчет оформляется в соответствии с требованиями, предъявляемыми к оформлению лабораторных работ в ВУЗе, и должен содержать:

1. Титульный лист.
2. Формулировку цели работы.
3. Постановка задачи в соответствии с заданием.
4. Краткие теоретические сведения и описания этапов выполнения работы.
5. Исходные коды VRML-сцен.
6. Выводы по работе.

Отчет готовиться каждым студентом индивидуально. При защите работы необходимо наличие электронной версии HTML-документов.

2.6 Контрольные вопросы

1. Что такое VRML?
2. Что необходимо для создания и просмотра VRML сцен?
3. Какие примитивы VRML Вы знаете?
4. Как можно именовать объекты в VRML?

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Гурский Д. А. Macromedia Flash MX и ActionScript 2.0 / Д. А. Гурский, Ю.А. Гурский. – М: ООО Новое знание, 2004. – 530с.
2. Панкратова Т. Flash MX 2004: учебный курс / Т. Панкратова. – М.: Питер, 2004. – 175 с.
3. Интернет. Энциклопедия / Под редакцией Л. Мелиховой. – СПб.: ЗАО Питер Бук, 2000. – 528 с.
4. Якушина Е. Изучаем Интернет, создаем Web-страничку / Е. Якушина. – СПб: Питер, 2002. – 256 с.
5. http://www.vrml.org

Заказ № ___ от « ___ »_____200__ Тираж__экз.
Изд-во СевНТУ