

Project Report

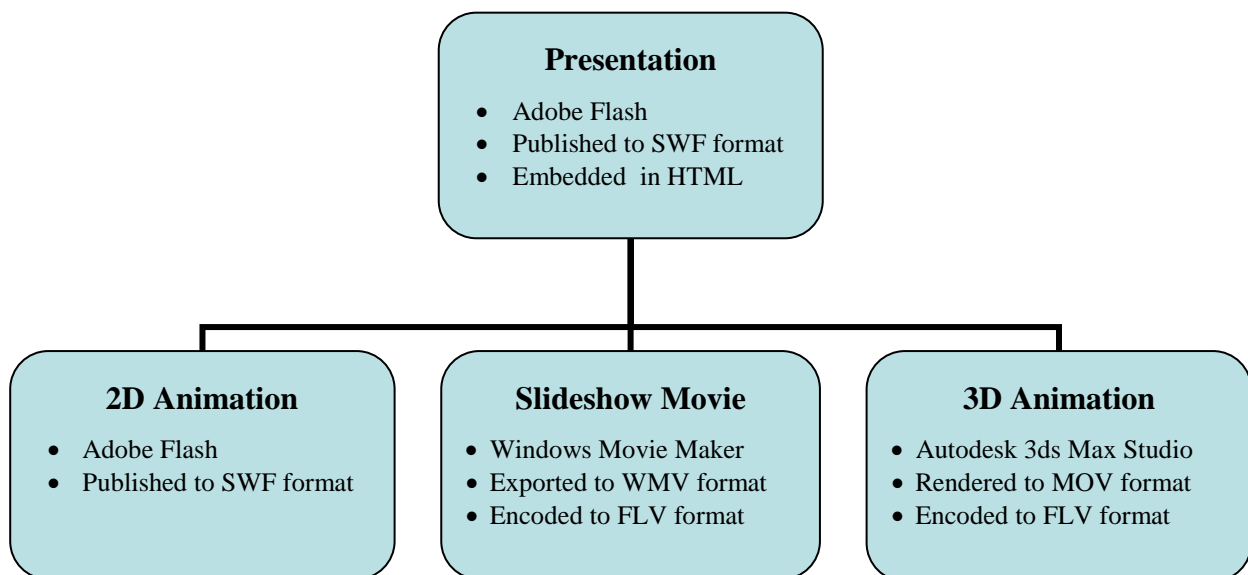
Objectives

This project is to give us a taste of multimedia presentations and hands-on experiences in some of distinct tools and concepts that we've covered throughout the course. Meanwhile, the emphasis would rather be on understanding fundamental composition of different multimedia contents than acquiring masterful skills and techniques for particular multimedia creation tools. In addition, this project will demonstrate an example of how multimedia contents can be delivered through the Web, which is the most common way to deliver multimedia these days.

The “presentation” will literally present some multimedia contents, each of which consists of different multimedia formats (e.g. bitmaps, graphics, text, animation, and etc) and is created by a different tool. All the contents are finally encoded to Flash Movie formats, thereby being put together in the presentation. As a whole, the presentation will have a theme – Appreciation of Multimedia.

Structure of Presentation and Deliverables

The presentation consists of three multimedia contents: 2D animation, Slideshow Movie, and 3D animation. Actually, the presentation file itself is a multimedia content, too.



Each of three contents is created by a different tool and exported to a native file format that the particular software supports. The presentation is also a Flash movie that is embedded in the HTML page and enables users to load each movie as they click on the interactive menus.

2D Animation

The simple 2D animation in the presentation is created by Adobe Flash. All the contents in the animation are primarily vector graphics, which are made of individual scalable objects rather than fixed bitmap images. With the virtue of vector-based objects, graphics can be edited and modified any time later and then recompiled for an updated animation.

The animation is achieved by two primary techniques: frame-by-frame and motion tweening techniques. First, frame-by-frame animation is the most intuitive way of creating animation since animation is the illusion of movement. Simply, frame-by-frame animation is created by a series of frames in which static parts of the scene remain unchanged and only the objects that change are redrawn. This technique is used only for a couple of parts of the entire animation. As shown in Figure 1, different scenes are drawn in sequential keyframes, which are indicated by the red rectangle in the figure. The penguin image is the only bitmap image (in PNG format) that is imported to Flash and converted to a “symbol” in order to apply motion tweening to it.

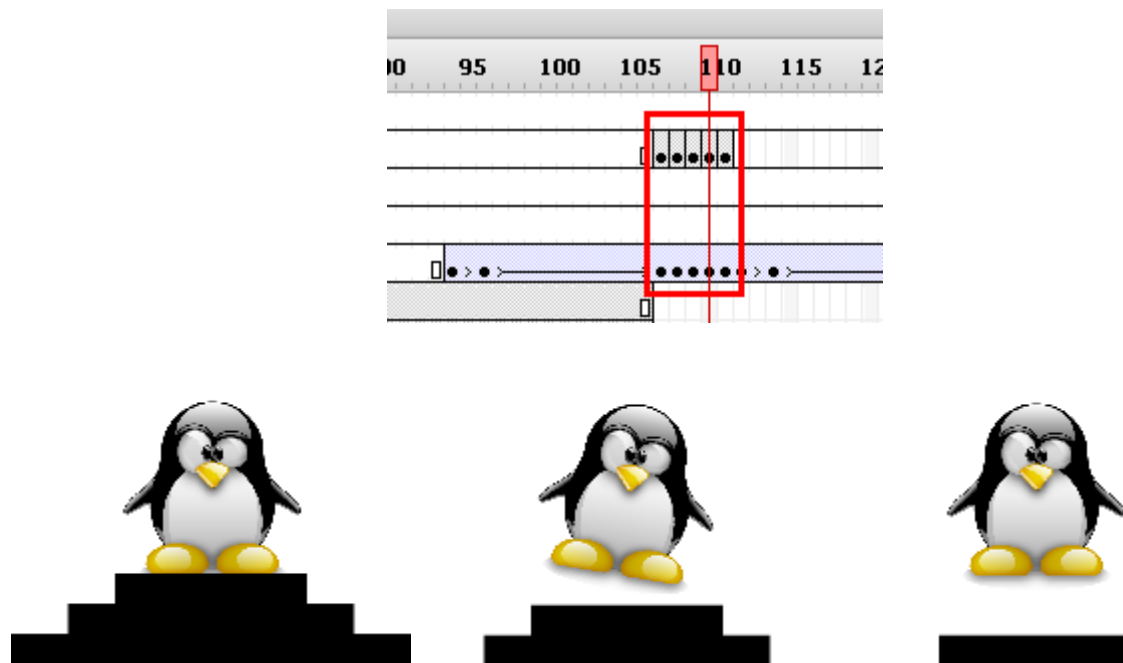


Figure 1

The second technique used for most parts of the animation is motion tweening. By motion tweening, it's not necessary to draw every single frame. Instead, we can set up the position and attributes of graphics in the starting frame and the ending frame. Based on the settings, Flash will mathematically compute the frames in between and gradually transform one frame to the other – so called “morphing” – to create an animation. Hence, motion tweening is more efficient and elegant than the naive frame-by-frame animation. As shown in Figure 2, the animation employs motion tweening along motion guidelines (the blue paths in the figure), which is invisible during animation. Unlike this, the positions of the eye-look-alike ball are manually changed by “free transform”. During tweening, ease-in and ease-out properties are changed in order to add an effect that objects gradually move slower or faster. In addition to the positions, tweening alters the attributes of graphics, such as color, alpha, and scale, based on the settings.

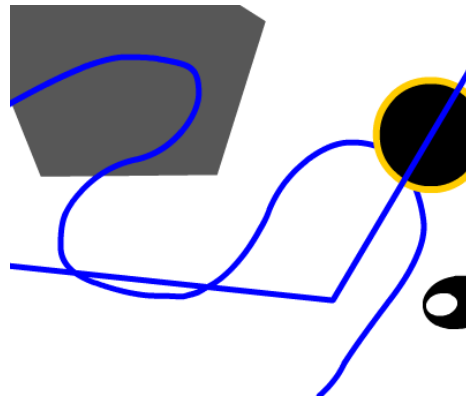
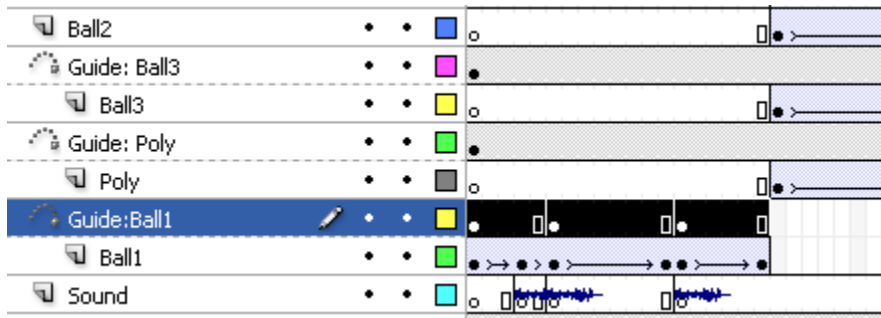


Figure 2

As shown in Figure 2, sounds (in WAV format) are also imported and synchronized to the motions of objects. Another technique used along with motion tweening is the use of “mask layer”. A mask layer can be used to show only a certain area beneath a defined mask. As shown in Figure 3, the ending part of the animation uses a “mask layer” (the blue circle in the figure) and applies motion tweening to it in order to create an effect that the underlying image is vanishing into the shrinking circle.

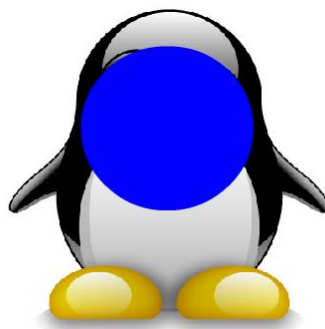
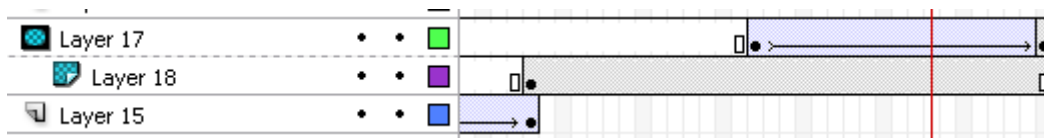


Figure 3

Slideshow Movie

The slideshow movie is created by Windows Movie Maker. Creating a slideshow movie by using Windows Movie Maker is quite simple and doesn't require any high-level skill. All we need to do is just to understand the concept of "timeline" of a movie and how to apply built-in transition effects and other functions. In contrast with the 2D animation based on vector-based graphics, the slideshow movie has a large file size because it works with bitmap images, photographs in JPEG format. Furthermore, as the frame size of the movie changes, its display quality significantly changes since bitmap images are not well scalable. For this reason, the slideshow movie is chosen for the presentation as a good example of bitmap images versus vector graphics and movie versus computer animation.

To create the slideshow movie, JPEG images are imported to Windows Movie Maker and arranged in the timeline as shown in Figure 4. Then, built-in transition effects are inserted between each image, and a MP3 audio track is also added to the timeline.

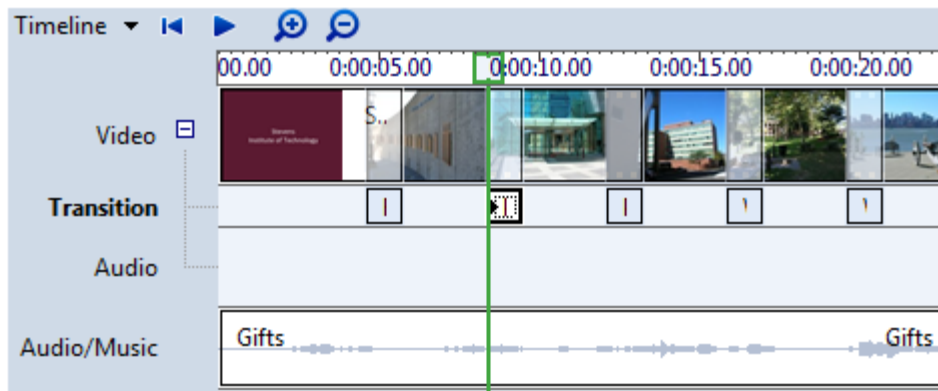


Figure 4

The exported slideshow movie is not directly supported by the presentation file, which is a Flash movie. Therefore, it is encoded to the FLV (Flash Video) format by Adobe Flash Video Encoder so that it can be loaded inside the presentation. The encoding properties are as shown below.

	Intermediate Slideshow Movie	Final Movie Clip
Frame size	640x480 pixels	400x300 pixels
Aspect ratio	4:3	4:3
Bit rate	2.861 Mbps	700 kbps
Frame rate	30 fps	30 fps
Duration	01:49	01:49
File Size	18.4 MB	9.08 MB
File Format	WMV (Windows Media Video)	FLV (Flash Video) / On2 VP6 Codec

3D Animation

The 3D animation in the presentation is created by Autodesk 3ds Max Studio. All the objects are vector-based graphics just like the objects in the 2D animation created by Flash. The main difference is that 3ds Max creates 3d objects whereas Flash creates 2d objects. For our model, three primitive objects are used: box, pyramid, sphere, and cylinder. By inspection, it's pretty straightforward to determine which primitive is developed to which part. The only tricky part is bended parts of two parallel bars, which are created based on cylinders. The cylinders are bended with respect to z-axis by 90 degree, using the Bend modifier, as shown in Figure 5.

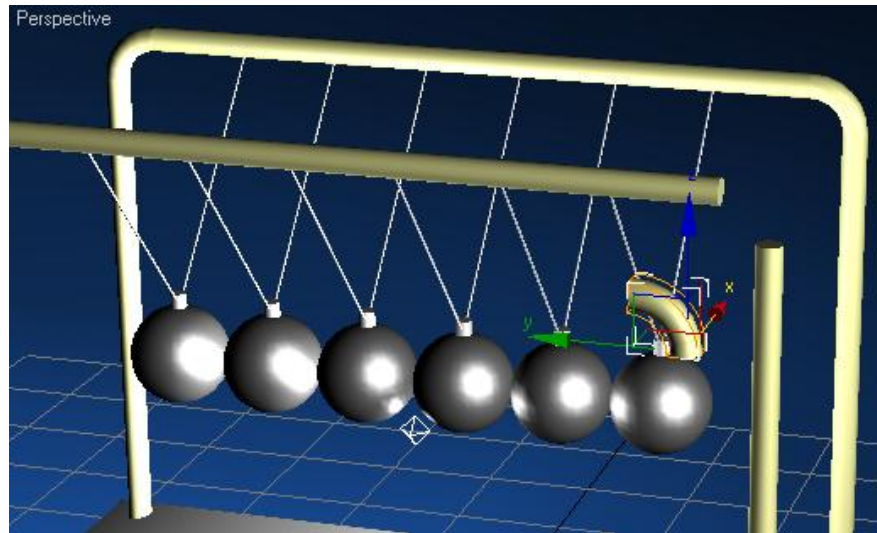


Figure 5

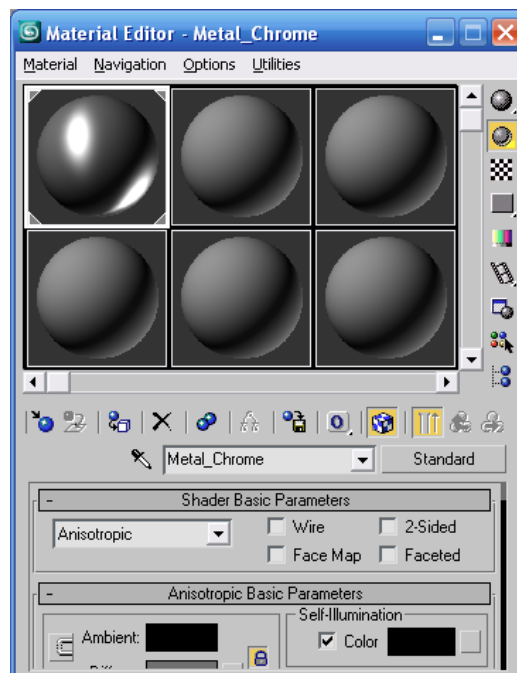


Figure 6

Material is one of the neat features of 3ds Max Studio that creates vector graphics. In essence, vector graphics fall short of realistic views of modeling. This is the main disadvantage of vector graphics against photographs, which are basically bitmap images. However, 3ds Max Studio allows us to overcome this shortcoming by means of Material. As shown in Figure 5 and 6, the metal chrome material, a bitmap image, is applied to the surface of the balls for a more realistic view.

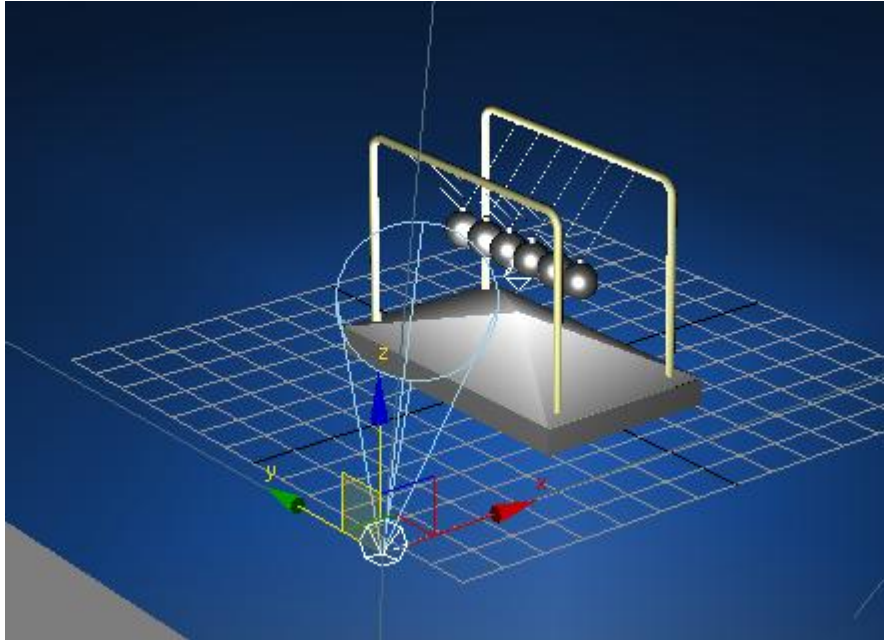


Figure 7

For an even more realistic touch, the Lighting feature, using free spot light, is applied in the scene as shown in Figure 7. As a result, the objects also have shadow and shade effects.

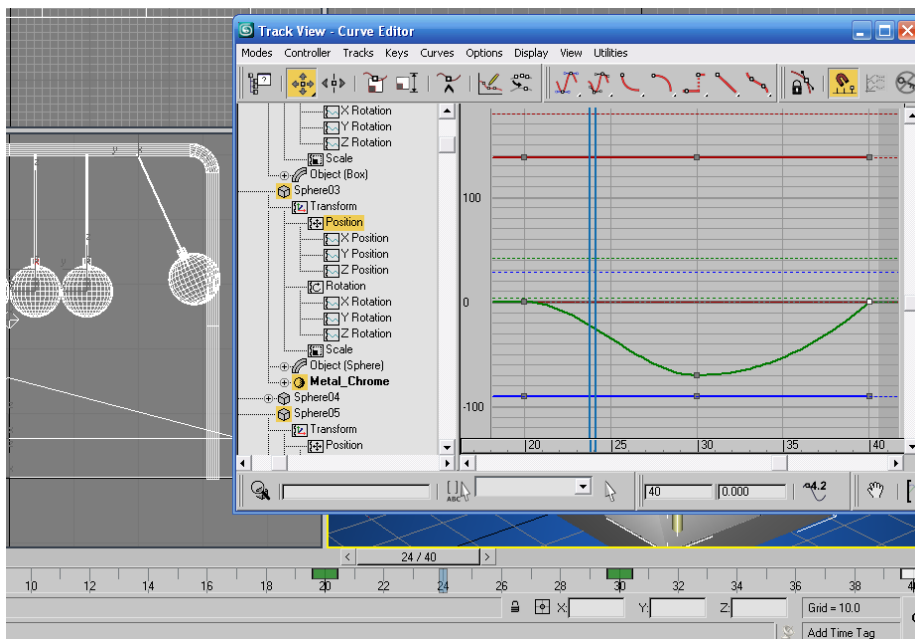


Figure 8

After modeling is done, animation is achieved by specifying the positions of objects in the xyz coordinate systems. In contrast with the 2D animation earlier, the positions are determined in 3d space. Other than that, the fundamental ideas of animation by “transforming” are similar to those of the 2D animation in Flash. As shown in Figure 8, the starting frame and the ending frame are specified in keyframes and the movements of animating objects are specified by the curves, which are essentially mathematical functions (e.g. trigonometric, linear, or quadratic functions).

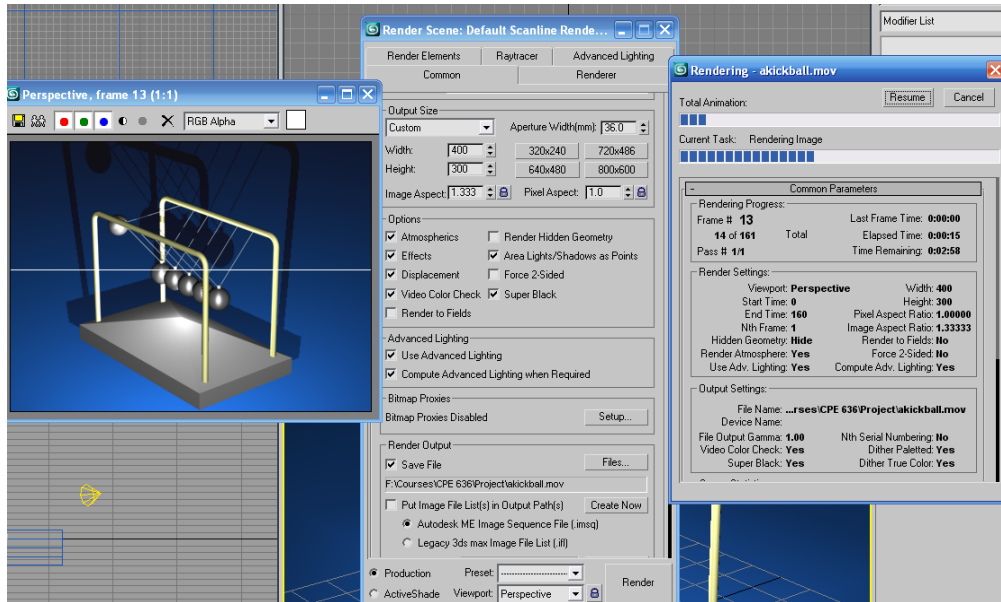


Figure 9

Finally, the 3d animation is rendered to a movie file in the QuickTime video format, MOV, as shown in Figure 9. Even though the 3d model is treated as vector graphics inside software, the rendered video is actually a collection of sequential frames, each of which is rendered to a still bitmap image. Therefore, the final output has a large file size, just like the slideshow movie file, while it’s difficult to preserve the display quality. The rendered movie is not directly supported by the presentation file, which is a Flash movie. Therefore, as mentioned before, it is encoded to FLV (Flash Video) format by Adobe Flash Video Encoder so that it can be loaded inside the presentation. The encoding properties are as shown below.

	Rendered 3D Animation	Final Movie Clip
Frame size	400x300 pixels	400x300 pixels
Aspect ratio	4:3	4:3
Bit rate	4.39 Mbps	700 kbps
Frame rate	30 fps	30 fps
# of Frames/Duration	161 frames	00:05
File Size	23.5 MB	310 KB
File Format	MOV (QuickTime Video)	FLV (Flash Video) / On2 VP6 Codec

Presentation

The presentation file is another Flash movie, which puts all the contents together and let users to load each movie by clicking on the interactive menu buttons. Like the 2D animation, the presentation also uses motion tweening for animated texts as shown in Figure 10. In addition, by clicking on the main menu, users are redirected to the corresponding frames, each of which contains a “loader” at the position indicated with the red rectangle in Figure 10. This is achieved by a simple ActionScript as follows:

```
on (release) {  
    gotoAndStop(50);  
}
```

An interactive component, Loader, is used to load a JPG or a SWF. Since our 2D animation is a flash movie clip (SWF), the Loader component is used as a “loader” and the path to the movie clip file is set to the parameter, `contentPath`.

For the slideshow movie and 3D animation, the FLVPlayback component is used as a “loader” since they are Flash Video clips (FLV). To load a specific video clip, the following ActionScript is used.

```
FLVPlayer.contentPath = "slideshow.flv";
```

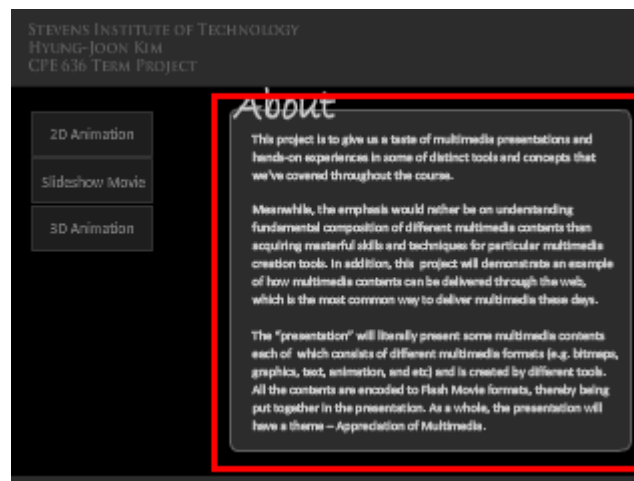
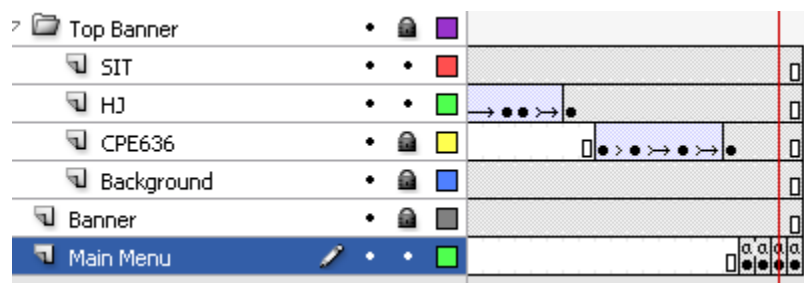


Figure 10

In fact, the presentation movie is made of two scenes. The main page explained above is actually Scene 2. The other scene, Scene 1, is loaded first when users connect to the web page and shows the progress status of downloading the entire movie clip. Often, Flash movie clips have large file sizes, so this is a common way to let users know the downloading status, preventing them from leaving the page. Therefore, the following ActionScript computes the percentage of bytes loaded and starts displaying Scene 2 if the entire clip is loaded.

```
downloaded = int(100*getBytesLoaded()/getBytesTotal())+" %";
ifFrameLoaded ("Scene 2", _root._totalframes) {
    gotoAndPlay("Scene 2", 1);
}
```

Finally, the presentation Flash movie clip is embedded in a HTML page, which is then uploaded to a web server along with other movie clip files so that they can be delivered through the Web.

URL of the presentation movie: <http://personal.stevens.edu/~hkim/CPE636/>