CHAPTER 3

Animation Techniques

IN THIS CHAPTER

• The Basics 40
• DirectX Objects 43
• Techniques 48
• Summary 50
Many of us grew up watching cartoon characters every Saturday morning, but we never really appreciated the work that went into making them. In this section of the book, you’ll be looking at animation techniques that have been around for many years. These techniques gave birth to talking rabbits, wacky ducks, and a mouse named Mickey.

Animation is the art of creating the illusion of movement from inanimate objects. Before we had full-motion moving pictures, there was animation. Flipbooks were small books with pages where small caricatures were drawn. As you flipped the books from front to back, the drawings looked as though they were animated. Some flipbooks came empty, and you could add your own drawings. Flipbooks gave the illusion that the cartoon character in the pages was actually moving before our eyes. This illusion is the cornerstone of animation. With some simple animation techniques, you can add impressive flare or sex appeal to your multimedia applications.

The Basics

In this section, you’ll be looking at the basics of traditional animation—how animators have been able to achieve lifelike expressions of many different objects. This is by no means a thorough explanation of animation, only an introduction to make DirectX easier to understand.

The Model Sheet

If you’re going to create a character to animate, it usually helps to have a model sheet, which is a series of poses of the character to refer to when you’re animating. The sheet could include poses of the character standing still, bouncing a ball, smiling, jumping, and running, for example. The idea is to draw the emotions and actions that would be expected from the character. If your animations are going to be much simpler than an animated person or animal, however, a model sheet might not be necessary.

The Frame

A frame is a snapshot in time. If you were to take your favorite animated movie and pause the playback for a split second, you would see one frame of animation. Imagine a bird in flight. If you could stop this bird while it was flying along its course, you might see the wings folded slightly. Maybe its eyelids are opening or closing, and maybe the beak is slightly open. If you paused the playback a second later, the next frame would look different. Figure 3.1 shows a single frame of animation in which a spinning donut is frozen in time.
FIGURE 3.1
A frame of animation.

The Cycle

Animation moves in cycles, and a cycle is a series of frames that make up an action. For example, the cycle of walking can be made up of about eight frames. This cycle begins with the character picking up his right foot with his right shoulder raised slightly. The cycle continues until frame 4, where the right foot is back on the ground. Then the same process happens with the left foot. Finally, frame 8 ends with the left foot being put back on the ground. Cycles can be repeated, if necessary, thus reducing the need for creating the animated frames from scratch. When frame 8 of the walking cycle is finished, for example, just start over with frame 1. The character can walk on forever, if you want.

A complex animation, such as those seen in cartoons, is made up of many individual cycles. For example, the character moves to a specific spot (cycle 1). The character stands and acts as though he or she is thinking about something (cycle 2). The character suddenly does an about-face (cycle 3). Finally, the character walks back the way he or she came (cycle 4). Figure 3.2 shows four cycles of a character walking in four different directions. Each cycle in the figure is made up of three frames.

FIGURE 3.2
An animation cycle.

Storyboards

I’ve talked about how to go from an individual split second in time into a cycle. Now comes the question “How do I decide what I need and how it’s organized?” This is where the storyboard comes in. Storyboards are a series of small panels showing major scenes of action in the animation. Storyboards were brought into the mainstream of animation with the Walt Disney studios, which perfected their use. Through the use of a storyboard, lead animators could create sample drawings of the scenes as they imagined them and write in the dialogue just
below the storyboard panes. Today developers of multimedia applications use similar processes to organize the action before they begin actual development. Figure 3.3 shows a section of a much larger storyboard used to explain the action of a scene.

**Figure 3.3**

A storyboard.

**Keyframes**

Lead animators would then take these storyboard scenes and create individual frames of the animation at specific pivot points in the action. These points are called *keyframes*. Keyframes are drawn where parts of the character’s body reach their full range of motion. For example, an arm can bend only so far back before it breaks. If the character has reached that point in the animation, then that frame of animation should be a keyframe. The same goes for head movements and any body appendages. These drawings create the templates from which other animators can create the finished frames of the cartoon.

**Backgrounds**

So you’ve created an animation. It looks fantastic, but it’s just bouncing around on a blank screen. Now you need to create a *background*, which is the environment you want to add to your scene that helps set the mood for the animation. Backgrounds were painted in most earlier animated cartoons, but more and more of them have moved to using computer-generated backgrounds. This type of background is very similar to what you’ll be working with in DirectX. You can create backgrounds with many different paint programs because DirectX can use almost any popular graphics format.
DirectX Objects

DirectX makes use of early animation technologies, but has added a few tricks of its own. This section describes DirectX objects and concepts used throughout the computer animation community.

Frame Rates

Frame rates are basically the speed at which frames or cels of animation are shown, expressed in number of frames per second of time. A normal frame rate for animated motion pictures is 24 frames per second (fps). With computer animation, frame rates look much better at about 40fps – 50fps.

When working with DirectX programming, frame rates are one of the most important factors to keep in mind. If the frame rate of the animation you’re producing is too slow, the user will be bored by your animation. If it’s too fast, it will look unnatural, perhaps making the user think something is wrong with the machine. Therefore, you need to make sure the frame rate is a constant one that’s pleasing to the eye.

Frame rates are implemented in DirectX by how you send data to the video card. You want to make sure you send enough frames of your animation to the card to give you the results you need. However, not all video cards have the same capabilities, so you need to make sure you don’t overload the card. For example, if you know that your frame rate is going to be 50fps, but the average card specification for your users doesn’t support more than 30fps under this load, you could have a problem. A good rule of thumb when working with video cards is to look at trade magazines and online sites to determine what video cards are popular. Then you can choose from the bottom and top of that price range to determine the cards you should test. Also, using programs such as 3D Mark can help you determine the capabilities of various video cards. 3D Mark can be found at http://MadOnion.com. Using timers can help regulate frame rates. Chapter 2, “Multimedia Programming in the Visual Basic Environment,” discusses using timers to control frame rates.

Blits

Blit, an acronym for “bit block transfer,” is the process of moving sections of data from one memory space into another. After you assemble the frame of animation you want to show, you must then move it from the back drawing surface to the front drawing surface, where the user can see it. To do that, you use blits. Blits can transfer a full screen of information from one memory buffer to the other. They can also transfer a small rectangle of information. After the information is blitted to a surface, the surface can be flipped with another surface to change the image the user sees. Figure 3.4 shows how Blits are performed on an offscreen drawing.
surface. The blit process and surfaces are explained in more detail in Chapter 5, “Creating Basic Animations with Surface Objects.”

Sprites

Sprites are rectangles of data that contain an image. These small portions of the overall image are used to create the animation effect.

Sprites can be used in the foreground to give you animation. For example, if you have a kiosk program that shows animated tickets moving across the screen, you would need to create a series of images for each unique frame of the animation. You could then save all the sprite images as one image, which is called a sprite strip. The sprite strip in Figure 3.5 shows a spinning donut in all four possible poses. You would then move the sprite around the screen by changing its coordinates and loading a different image every frame. Chapter 5 discusses sprites and how they are animated in a DirectDraw application.

Figure 3.5
A sprite strip.
Drawing Surfaces

The *drawing surface* is where you assemble your picture for blitting; it can be thought of as the staging area for each frame of animation. The drawing surface can be made up of a background image and one or more sprites. During the process of blitting the image, this surface is transferred from the back buffer to the front buffer. Buffers and surfaces are explained in more detail in Chapter 5. There are three types of drawing surfaces used in animation:

- **Primary:** The primary drawing surface is the one currently displayed on the screen. It’s what the user sees at all times.
- **Secondary:** The secondary drawing surface is where you set up for the next frame of animation. When it’s time for the next animation frame to be displayed, it becomes the primary surface and the old primary surface becomes the new secondary. In essence, you force them to switch identities.
- **Offscreen:** The offscreen drawing surface is where you assemble the content for the primary and secondary surfaces. This part of DirectDraw is always working to keep the primary and secondary surfaces full of information.

**NOTE**

An offscreen surface is used only if your design for flipping uses three surfaces. Three surfaces are used in more complex animations, especially since the memory on the video card can help speed this process up. However, for simple animations you need to use only a primary and secondary surface.

Palettes

*Palettes* are simply a list of colors available for the display, and you choose from this list when rendering the animation. The color depth required for your application determines the number of colors available in a palette. Just as it does for a painter, the palette defines the colors you have to work with in your project. When working with palettes, it’s a good idea to choose a color depth that gives you the best results for your animation work.

It’s possible to change the palette at runtime when producing animations, however. This animation technique is good for full-screen animations that require fading, such as a sunset scene or movie credits.
Rectangles

Rectangles are square areas of the screen that you define with an x1, y1, x2, y2 coordinate system. Many of the methods in DirectX need to know the area of the display that’s being modified. This is accomplished by passing a rectangle object that provides the coordinates of these areas. For example, if you have a small bug on the screen and you want to change its color, it’s easier to just pass in a rectangle object with the bug’s dimensions and location on your screen.

You can tell the DirectDraw object where the bug is by passing in the Rectangle object that contains the location of the rectangle and its dimensions. In Microsoft Visual Basic, the RECT type has the following definition:

```
Type RECT
    Left As Long     ' Top-left. X coordinate.
    Top As Long      ' Top-left. Y coordinate.
    Right As Long    ' Bottom-right. X coordinate.
    Bottom As Long   ' Bottom-right. Y coordinate.
End Type
```

In the preceding example, the Left and Top members are the x- and y-coordinates of a bounding rectangle’s top-left corner. Similarly, the Right and Bottom members make up the coordinates of the bottom-right corner. The following diagram in Figure 3.6 illustrates how you can visualize these values by showing how a rectangle (RECT) is measured on the screen.

![Figure 3.6](chpt_03.qxd_11/9/01_3:18_PM_Page_46)

**Figure 3.6**

A rectangle.

Clippers

Clipping is the process of defining areas of the display that will show the animation. By marking off these areas, you are designating where it is okay for DirectDraw to write(or blit) on the
destination surface. An example of this technique can be seen in a shooting gallery. The duck targets move from one side of the screen to the other; however, you don’t want them to appear on the left and right sides of the shooting area because that would ruin the effect of your shooting gallery. You would need to define a clipper rectangle that encompasses the “shooting area” of the display. This technique gives the illusion that the ducks are going behind the curtain.

Figure 3.7 shows a screen made up of clipper rectangles. Each of the four rectangles is part of a clipper list. The clipper list is associated with a clipper object, which in turn defines the area that can be used for animation. Any blitting that occurs outside this area will not show up on the screen. If a sprite crosses this boundary, only the part of the sprite that is within the clipper area is displayed. In the previous example of the shooting gallery, you would want the ducks to appear only in the “shooting area,” not in front of the curtains.

**Figure 3.7**
*A screen with clipper rectangles.

**Billboards**

*Billboards* are objects used to create the illusion of 3D objects without having to spend the resources on creating complex objects. This effect is accomplished by creating a plane object that’s moved in relation to the person viewing it, which produces the illusion that the object has many sides, even though it actually has only one side. The one side keeps moving in relation to the person viewing it.

For example, if you were creating a driving game that had many trees along the sides of the course, you don’t want to create complex trees made up of thousands of polygons. The memory needed for that operation would be enormous, and your program’s performance would suffer. So you simply create a 2D image of a tree, attach it to a billboard, and move the billboard in relation to the user. The user doesn’t know the difference, and you save on performance. Figure 3.8 shows how billboards work.
Techniques

To understand how to use the objects you’ve been reading about, you need to look at the techniques used in DirectX to create animations. The techniques include blitting, page flipping, back buffering, and palette cycling.

Blitting

Blitting is the process of taking rectangles from offscreen drawing surfaces and copying them onto primary or secondary drawing surfaces. Blitting is much like taking pieces of a jigsaw puzzle and assembling the finished product on a table.

Page Flipping and Back Buffering

Page flipping is the process of cycling through the drawing surfaces, or pages. In back buffering, you populate these surfaces with new frames of information. With these two techniques, you can create true animation by creating a frame, displaying the frame, creating the secondary surface, and then displaying it. Figure 3.9 shows how page flipping occurs.
Before the flip occurs, users see a single vehicle going east. What they are seeing at that point is the primary surface. While this is going on, the secondary surface is being redrawn with an additional vehicle, a little bit larger, going west. When the flip occurs, the users’ view is changed to the secondary surface. In essence, the secondary surface in the previous frame is now the primary surface, and vice versa. Users now see both vehicles. Because the car going east will have traveled off the screen in the next frame, you remove it from the secondary surface. The secondary surface is now redrawn to get ready for the next flip.

Palette Cycling

Palette cycling can best be visualized by imagining a painter’s palette. Painters use a palette of colors to create their paintings. Just imagine what would happen if they changed a few of the colors on their palettes to new colors, and that change resulted in the colors in their painting automatically changing, too. You would have the same painting, but with a few colors changed. If you continued to do that, some of the images in the painting would look animated, simply because you had changed their colors over and over. This effect is called palette animation, and this technique can be used to create psychedelic effects such as those popular in the ‘70s. You can also use it to create fading effects on objects and text to give the impression that they are appearing or disappearing.
Summary

In this chapter, you have traveled through a brief history of animation, examining the tools of the trade: model sheets, frames, cycles, storyboards, keyframes, and backgrounds. You have also seen how blits and sprites are used to transfer and store images, learned how drawing surfaces are used to produce real-time animation through primary, secondary, and offscreen surfaces. You also saw how backgrounds and palettes can be useful for creating full-screen scenery and fading effects.

Rectangles, you learned, are the foundation of measurements in DirectX, and are used to define the position of images and maintain their animations. Clippers are useful in restricting areas of the screen so that animations can appear to have depth to them. With billboards, you can create the illusion of large complex items, such as trees, without actually having to use the enormous processing power needed for high-end graphics.

Finally, you got an overview of the three main techniques in DirectX animation: blitting, the process of placing sprites or images on the surface you’re assembling for display; page flipping (with the help of back buffers), used to move surfaces from front to back in rapid succession to create fast animation effects; and palette cycling, used to animate backgrounds for fading and psychedelic effects.