Part One

ACTIONSCRIPTED ANIMATION BASICS

```
targetWidth = shape
shape.targetWidth = shape._width;
shape.targetHeight = shape._height;

shape.onEnterFrame = function() {
                        th - this._width)
                        this._heig
shape.targetH

           Frame = functio

    this._width       Width   s._width
    this._height += (thi     eight   is._he

shape.targetWidth = shape._width;
shape.targetWidth = shape._width;
shape.targetHeight = shape._height;

    n.onEnterFrame = function() {
    speed = 5;
    this._width += (this.targetWidth - this._width)/speed;
    this._height += (this.targetHeight - this._height)/speed;
}shape.targetHeight = shape._heigh
```

Chapter 1

BASIC ANIMATION CONCEPTS

What we'll cover in this chapter:

- What is animation?
- Frames and motion
- Dynamic vs. static animation

Flash, at its core, is an animation machine. From the very earliest versions, Flash has supported animation through tweens—where you create a couple of keyframes that are different and let Flash fill in what is in between. However, this book is not about tweens. This book is about the powerful language built into Flash, called ActionScript. This book covers programming, math, and physics techniques used to make things move with ActionScript. As you'll see, this gives you levels of power, control, and interactivity that you could never hope to match with tweening.

But before we dive into specific techniques and formulas for moving things around with ActionScript, let's take a quick look at exactly what animation is, some of the basic techniques behind it, and some concepts that you can use to make your Flash animations more dynamic and interesting.

Sure, you can skip this chapter if you are just dying to write some code. But I strongly suggest you come back to it at some point. If nothing else, you'll find some interesting insights into the subject.

What is animation?

First, the question of all questions: What is animation? Well, per the *American Heritage Dictionary of the English Language, Fourth Edition* (Houghton Mifflin Company, 2000), it means the following:

1. To give life to; fill with life
2. To impart interest or zest to; enliven
3. To fill with spirit, courage, or resolution; encourage
4. To inspire to action; prompt
5. To impart motion or activity to
6. To make, design, or produce (a cartoon, for example) so as to create the illusion of motion

While I could get philosophical with the first four definitions, what we are really talking about here are the fifth and sixth definitions. Animation means motion. I like to broaden that a bit and say that animation is change over time, specifically some type of visual change. Motion is basically the change in something's position over time. One minute it is over here; the next minute it is over there. Theoretically, it was also in the space between those two points, but I won't get metaphysical about it (not just yet anyway). It moved, and some time elapsed between the time it was at the first point and the time it was at the next one.

But an object doesn't necessarily need to change its location in order to be considered animated. It could just be changing its shape. Remember those photo-morphing programs that were all the rage in the late 1990s? You start with one picture of a girl and one picture of a tiger, and the program creates an animation between them. Or the object could be changing its size or orientation, such as a plant growing or a top spinning. Or it could even simply be changing its color. If you've been around long enough, you might remember some of the earliest animations on home PCs consisted of just cycling colors. You make a picture of a waterfall with a bunch of shapes in various shades of blue. You then cycle the colors of those shapes. If done right, the result gives the impression of falling water even though, technically, nothing is moving at all.

The connection of animation to time is an important one. Without any motion or change, there is no animation, of course, but also there is no sense of time. Sometimes you might see a webcam image of an empty room or a city skyline where nothing seems to be happening. It's impossible to tell if you are looking at a still image or a live video stream. Finally, you might notice some subtle change—a flickering light or a moving shadow. Just that slight flicker has reassured you that time is present, and maybe if you keep watching, something else will change. If you don't see any change after a while, you become convinced that it is a still image. There is no time, and you know nothing else will be happening in the picture.

That brings up another point: Animation keeps us interested in things. While the *Mona Lisa* is a wonderful piece of work and one of the most famous paintings of all time, I bet the average person gets bored after looking at it for 15 minutes tops, and then wanders off to see what else he can brag about having seen. But stick him in front of the latest high-budget Hollywood action film and he won't budge for a good two and a half hours. And if he does need to go to the restroom or to get a snack, he will wait for a "slow" part—one without so much action. That's the power of animation.

Frames and motion

Now, let's go back for a minute to that last definition of animate:

To make, design, or produce (a cartoon, for example) so as to create the *illusion* of motion.

Interesting that the definition writers should choose to throw that word *illusion* in there, yet entirely accurate. It happens that with just about every form of motion media, only an illusion of motion exists. Here's where we get to the concept of frames.

Virtually all visual animation media uses *frames*—a series of still images shown very rapidly to simulate motion or change. Anything you see on a computer, television, or movie screen is based on frames. This goes back to the earliest days of cartoon animation, where the individual pictures were drawn on sheets of cellophane and became known as *cels*, and the earliest motion pictures, where a similar technique was used with multiple photographs.

The concept is simple: You show a bunch of images that vary slightly from one to another, and the mind blurs them together as a single, moving image. But why do we insist on calling it an *illusion of motion*? If you see a man walk across the room on a movie screen, is that not motion? Of course it's only an image of a man, not the real thing, but that's not why we don't consider it to be real motion.

Remember when I talked about an object being over here and then later over there, and I said it moved through the intervening space? Well, that is real motion. Objects move through space smoothly, not in several jumps. (You quantum physicists in the audience, just be quiet.) But any and all frame-based motion does just that. It doesn't move from spot to spot; it disappears and reappears in another location in the next frame. The faster it's moving, the bigger jumps it takes.

If I showed you a picture of a man on the left side of a room and then a few seconds later another picture of the same man on the right side of the room, you'd say I showed you two pictures, not an animation. If I showed you half a dozen pictures of him in the process of crossing the room, you'd still say it was a series of individual photos. (See Figure 1-1 for an example of a series of still photographs.) If I presented enough photos fast enough, that wouldn't change the fact that they are still just a whole bunch of still photos, but you would no longer see it that way. Your mind would take it in as a man moving across the room. It is no more real motion than the original two photos were, but at *some point*, the mind gives up and buys into the illusion. As a matter of fact, that point has been well researched by the film industry.

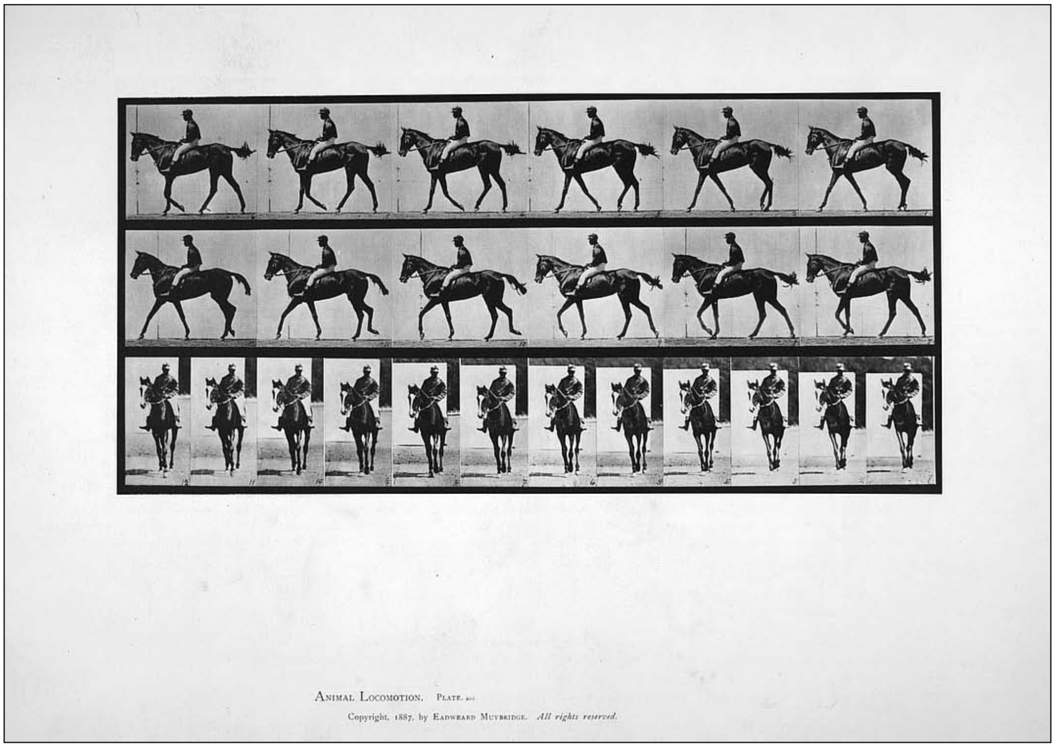
Animal Locomotion. Plate
Copyright, 1887, by Eadweard Muybridge. All rights reserved.

Figure 1-1. A series of still photographs by Eadweard Muybridge

Researchers have found that at a rate of 24 frames per second, people are perfectly happy to accept those frames as a single moving image. Go too much slower than that, and the jumpiness gets annoying and starts to break the illusion. And it seems that the human eye can't distinguish frame rates very much higher than that, so theoretically, going 100 frames per second isn't going to make your movie seem any more realistic (although higher frame rates in programmed animation can result in more responsiveness in interaction, and will seem smoother, especially for fast-moving objects).

Frames as records

The whole concept of frames makes three things possible: storage, transmission, and display. You can't really store, transmit, and display a man walking across a room. But you can store a picture, or many. And you can transmit them and display them. Thus you can show that animation almost anywhere, at any time, as long as you have or can receive the stored images and have a way to display them.

Now, let's get a little more general definition of what a frame is. So far, I've been referring to a frame as a still image or a drawing. Let's call it a record of a system at a specific point in time. That "system" could be your two-year-old daughter caught mid-grin, and the record would be that image. On the other hand, that system could be a collection of virtual objects, and the record could be their shapes, sizes, colors, positions, and so on at that particular moment in time. Thus, your movie would become not a series of still images, but rather a series of descriptions of images. Instead of just displaying the image, the computer would take that description, create the image from it, and then display it. You can even go a step further by using programmed frames.

Programmed frames

Since you have a computer that can calculate things on the fly, you don't really need a long list of descriptions for your frames. You can cut it down to a description of the first frame and some rules on how to build the subsequent frames. So, now the computer is not merely creating an image from a description. It's creating the description first, then creating the image based on the description, and finally displaying the image.

Consider how much file space you could save using this approach. Images take up a lot of hard disk space and bandwidth. And 24 images per second add up fast. If you can boil that down to one description and a set of rules, you've possibly reduced the file size by a factor of hundreds. Even a very complex set of rules for how the objects should move and react takes up less space than a single medium-sized image. Indeed, one of the first things people notice about scripted animation is just how small it winds up being.

Naturally, there is a trade-off. As your system gets larger and your rules get more complex, the computer must work furiously to calculate the next scene description, and then work overtime to render it. If you're trying to maintain a particular frame rate, that gives the CPU a limited amount of time (milliseconds) to think about it. If it can't calculate the scene in time, your frame rate will suffer. On the other hand, image-based animation doesn't care so much about what's in the scene or how complex it is. It just shows the next picture, generally right on time.

I've used prerendered animation to my advantage at least once. I was putting together a presentation of a number of complex Flash ActionScripted animations. File size was not a problem, since the animations were going to be played from a local machine. But timing was critical, and I had no idea how smoothly the ActionScript would render the images on this unknown, untested computer. So I brought the Flash movies into Director and exported the whole thing as a giant QuickTime movie. Since the movie was now just a series of prerendered images, it didn't really matter anymore how complex they were. As long as the computer was capable of displaying a QuickTime movie, I knew it would do so smoothly. The presentation went off without a hitch.

Dynamic vs. static animation

Another advantage to using coded animation goes far beyond simple file size. It's the fact that a coded animation becomes dynamic. Have you ever watched the movie *Titanic*? I hope I'm not giving away too much, but the boat sinks—every time. It sank in the theaters, it sinks on VHS, and it even sinks on DVD. Short of pressing the Stop or Pause button, you can't stop it from sinking. That's because a movie is a series of still images. The images near the end of this particular series show the boat sinking, and that's that.

Now let's move from the *Titanic* movie to a Flash website. Remember the late 1990s, when Flash was originally taking off? Everyone had to have a Flash website intro. Some shapes would slide in and grow or fade out. A cheap audio loop would play. Some trendy buzzwords would fade or slide in or out. Maybe a beam of light or some shadows would appear. Wow!

OK, I won't be too harsh. At least two or three I remember really were "wow" material—real works of art. The intros for the Gabocorp and Ray of Light sites, shown in Figures 1-2 and 1-3, were legendary. But when I think back on it, I recall actually sitting through them only a couple of times. They were a minute or two long, and I watched them two or three times. Was that because they weren't good? No, it was because after you saw them a couple of times, there wasn't much more to see. Just like the *Titanic* movie, the website intros did the same thing each time. I call that *static animation* (my own personal oxymoron!), because the animation never changes. Each frame, from start to finish, is predefined.

Now, a coded animation isn't necessarily dynamic. I could take an object and, using code, put it in a certain position and have it move across the screen. Each time you play the movie, the same code runs and causes the same movement. That's hardly dynamic.

But what if I take an object, and again using code, determine a random point to place it and a random direction and speed to move it? Now each time you play the movie, something different will happen.

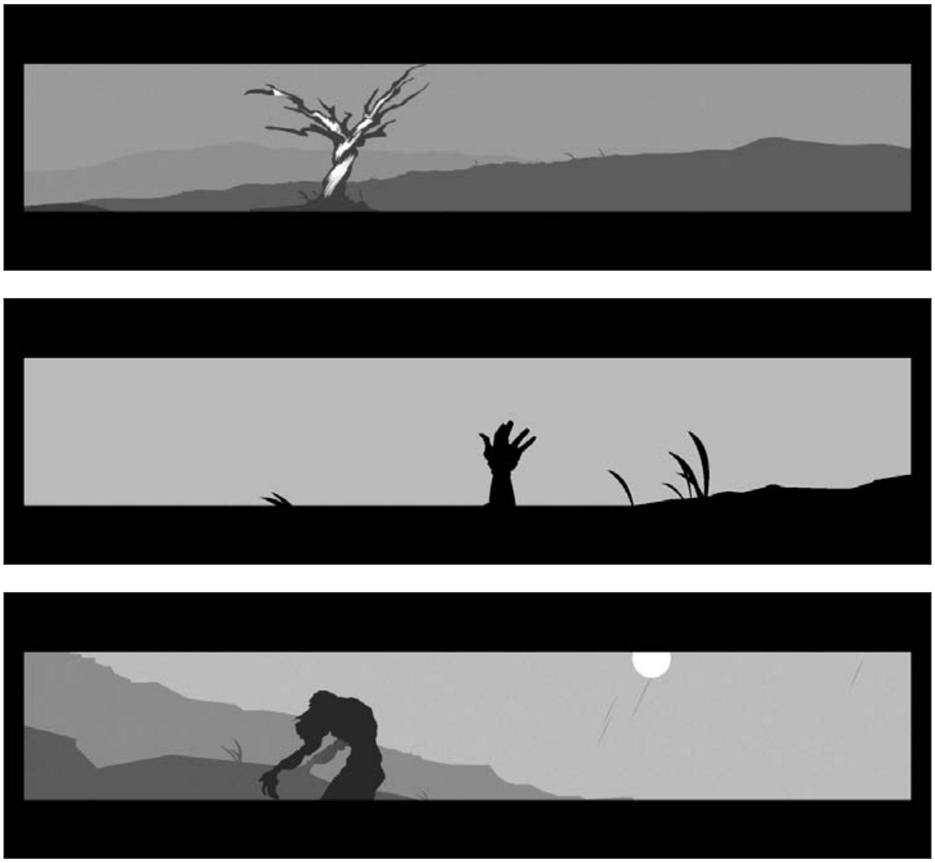

Figure 1-2. The legendary gabocorp.com intro

THE RAY OF LIGHT
A ray of hope, a ray of beauty ... as the clouds part the Darkness gives way to a single Ray from the sun. Find out our philosophy and goals . . .
GUIDE ME
MAIN MENU

Figure 1-3. Intro for rayoflight.com

Or what if, when the movie starts, I find out the time of day and month of the year, and use that data to create a scene—say a winter morning, a summer afternoon, or an evening in April?

Or how about if I have some factors in my movie that can be changed while the movie is running, via the keyboard and mouse? That would allow the user to interact with the objects on the screen. That's about as far from static as you can get. You could even save the *Titanic*!

Perhaps the most interesting aspect of dynamic animation, and what this book is mainly about, is the application of real-world mathematics and physics principles to the objects in the movie. You don't merely have some object move off in some random direction; you also give it some gravity, so that as it moves, it starts to fall down. When it hits the "ground," it bounces, but not as high as it started out. Then it eventually settles down and just sits there. Now you add some user interaction to that, allowing the user to "pick it up" with the mouse or move it around with the keyboard. As the user throws the object around, she starts getting the feeling that she is really handling a physical object.

With this type of animation, the user is no longer just sitting there watching some frames play out. She has entered into the environment you have created. How long is she going to stay there? She will remain as long as the environment keeps her interested. The more she can interact with the environment, the longer she will be interested. Make it interesting enough, and she will stay there a lot longer than she would sit through your intro (and sadly, probably longer than she would sit in front of the *Mona Lisa*). I have many e-mail messages from people telling me they spent their entire morning or afternoon playing with the experiments on `www.bit-101.com`. And not only will people stay longer, but they will also come back for more.

Summary

So, where is all this leading? In this opening chapter, I've gone over some of the basics of animation. But what do you actually do with this? Well, that's up to you.

In the following chapters, I'm going to put some tools in your hands and give you a quick lesson in how to use each one. What you build with these tools is entirely your decision. The most obvious use for much of what's in this book would be for game creation. Games are essentially interactive animations with some goals for players to achieve. But I really want to avoid this becoming simply a games book. I have used almost all of the techniques here in some kind of professional work other than games—from horrendous 3D menus and other not-so-bad navigation systems, to advertisements and educational applications.

A word of warning: Pick up any web design book, and you'll find a chapter telling you all about how too much animation is bad. I won't disagree, but I'm not going to say another word about it. If you want to hang yourself with animation, I'm going to spend the next few hundred pages giving you all the rope you need!