Foundation ActionScript for Flash 8

Kristian Besley
Sham Bhangal
David Powers
with Eric Dolecki

friendsof ED™
DESIGNER TO DESIGNER™
an Apress® company

Foundation ActionScript for Flash 8

ISBN-13 (pbk): 978-1-59059-618-0

ISBN-10 (pbk): 1-59059-618-8

Printed and bound in the United States of America 9 8 7 6 5 4 3 2 1

Distributed to the book trade worldwide by Springer-Verlag New York, Inc., 233 Spring Street, 6th Floor, New York, NY 10013. Phone 1-800-SPRINGER, fax 201-348-4505, e-mail orders-ny@springer-sbm.com, or visit www.springeronline.com.

For information on translations, please contact Apress directly at 2560 Ninth Street, Suite 219, Berkeley, CA 94710. Phone 510-549-5930, fax 510-549-5939, e-mail info@apress.com, or visit www.apress.com.

The source code for this book is freely available to readers at www.friendsofed.com in the Downloads section.

Credits

Lead Editor
Chris Mills

Technical Reviewers
Kristian Besley and David Powers

Editorial Board
Steve Anglin, Dan Appleman, Ewan Buckingham, Gary Cornell, Jason Gilmore, Jonathan Hassell, James Huddleston, Chris Mills, Matthew Moodie, Dominic Shakeshaft, Jim Sumser, Matt Wade

Project Manager
Denise Santoro Lincoln

Copy Edit Manager
Nicole LeClerc

Copy Editors
Ami Knox and Nicole LeClerc

Assistant Production Director
Kari Brooks-Copony

Production Editor
Kelly Winquist

Compositor
Dina Quan

Artist
Kinetic Publishing Services, LLC

Proofreader
Linda Seifert

Indexer
Julie Grady

Cover Image Designer
Corné van Dooren

Interior and Cover Designer
Kurt Krames

Manufacturing Director
Tom Debolski

CONTENTS AT A GLANCE

CONTENTS

ABOUT THE AUTHORS

Image courtesy of Simon James at `www.thefresh.co.uk`

Kristian Besley is a Flash/web developer working in education and specializing in interactivity and dynamically driven content using ASP.NET and PHP. In his spare time, he is also a lecturer in multimedia at the higher education level.

Kristian has written a number of friends of ED books, such as the *Foundation Flash* series (including the recently published *Foundation Flash 8*), *Flash MX Video*, and *Learn Programming with Flash MX*. He was a contributor to the *Flash Math Creativity* books, *Flash MX 2004 Games Most Wanted*, *Flash MX Video Creativity*, and countless others. He also writes for *Computer Arts* magazine and has produced freelance work for numerous clients, including the BBC.

Kristian currently resides in Swansea, Wales, the city of his birth. He is a fluent Welsh speaker and is the creator of the first-ever Welsh translation search plug-in for Firefox and Mozilla (available from `http://mycroft.mozdev.org`).

Sham Bhangal has written on new media for friends of ED since the imprint's inception. In that time, he has been involved in the writing, production, and specification of just under 20 books.

Sham has considerable working experience with Macromedia and Adobe products, with a focus on web design and motion graphics. Creating books that tell other people about his favorite subjects is probably the best job he has had (ignoring the long hours, aggressive deadlines, lost manuscripts, and occasional wiped hard drives). If he was doing something else, he'd probably be losing sleep thinking about writing anyway.

Sham currently lives in the north of England with his longtime partner, Karen.

David Powers is a professional writer who has been involved in electronic media for more than 30 years, first with BBC radio and television, and more recently with the Internet. This is his sixth book for Apress/friends of ED on programming for the Web. Among his previous titles are the highly successful *Foundation PHP 5 for Flash* (friends of ED, ISBN: 1-59059-466-5) and *Foundation PHP for Dreamweaver 8* (friends of ED, ISBN: 1-59059-569-6). David's other main area of expertise is Japan. He was a BBC correspondent in Tokyo during the late 1980s and early 1990s, and later was Editor, BBC Japanese TV. He has also translated several plays from Japanese.

ABOUT THE COVER IMAGE DESIGNER

Corné van Dooren designed the front cover image for this book. Having been given a brief by friends of ED to create a new design for the Foundation series, he was inspired to create this new setup combining technology and organic forms.

With a colorful background as an avid cartoonist, Corné discovered the infinite world of multimedia at the age of 17—a journey of discovery that hasn't stopped since. His mantra has always been "The only limit to multimedia is the imagination," a philosophy that is keeping him moving forward constantly.

After enjoying success after success over the past years—working for many international clients, as well as being featured in multimedia magazines, testing software, and working on many other friends of ED books—Corné decided it was time to take another step in his career by launching his own company, Project 79, in March 2005.

You can see more of Corné's work and contact him through `www.cornevandooren.com` or `www.project79.com`. If you like his work, be sure to check out his chapter in *New Masters of Photoshop: Volume 2* (friends of ED, ISBN: 1-59059-315-4).

INTRODUCTION

Welcome to *Foundation ActionScript for Flash 8*, the fourth edition of this legendary ActionScript book.

ActionScript is, quite simply, the driving force behind Flash applications, allowing you to go beyond simple tweened animations and give your movies intelligence, power, and class! The current version of ActionScript in Flash 8, 2.0, is a fully featured, very powerful programming language.

But ActionScript is that scary code stuff that programmers do, right? Wrong. ActionScript adds power and potential to your design work. It's not going to turn you into a reclusive nerd who speaks in 1s and 0s, and who only comes out after dark. It's going to turn you into someone who finally has the power to achieve his or her web design goals, rather than being hemmed in by frustrating limitations.

And Flash 8 has a treasure trove of new features for you to play with. It has amazing new design features such as filters and blend modes, features such as bitmap caching to enhance the speed of your movies, exciting new video capabilities, a great new BitmapData API for manipulating images on the fly, and much more.

If you know nothing or little about ActionScript, this book will provide you with a real foundation of knowledge from which you can build some awe-inspiring structures. You'll learn all the important stuff you'll need to make that giant leap toward becoming an ActionScript guru.

What you need to know

You've picked up this book, so we guess that you have the Flash basics under your belt. You'll probably have built some basic timeline-based movies with tweens and so on, and you may have read an introductory Flash book such as friends of ED's acclaimed *Foundation Flash 8*. If you haven't, we do recommend looking at it; you can find it at `www.friendsofed.com`.

If you don't have a copy of Flash 8 yet, you can download a fully functional 30-day free trial from `www.macromedia.com`. You can use either the Basic or Professional edition of Flash with this book, but we highly recommend going for Professional, as it features even more amazing functionality than the Basic edition!

FLAs for download

There's a wealth of code and support files available for this book. They're organized by chapter at the *Foundation ActionScript for Flash 8* page at `www.friendsofed.com`. Look under the book option on the site's main navigation to find it, and feel free to look around the site in general!

Cut to the Chase!
All Killer, No Filler Content with Flash 8 Essentials

Latest Books

Foundation ASP for Dreamweaver 8

In this book we look specifically at using Dreamweaver's built-in server behaviors to build dynamic web sites using ASP, Microsoft's highly popular classic server-side language, along with Access or MySQL databases.

Flash 8 Essentials

This book cuts to the chase, getting you right up to speed asap with the exciting new features of the latest version of Flash, without dwelling on what you already know.

Foundation ASP.NET for Flash

Whether you're a Flash designer or an ASP.NET developer, this highly readable Foundation book will provide a thorough grounding to integrating

Forthcoming Books

CSS Mastery: Advanced Web Standards Solutions

You'll already know why you should be using CSS, so we don't bore you with pages of theory; instead, this book jumps straight into practical solutions, allowing you to get what you need as quickly as possible.

Foundation Web Design with Dreamweaver 8

We know that you don't always require a full database driven site, so this book focuses on using the latest version of Dreamweaver to design and create usable, standards-compliant websites using XHTML and

Newsletter

To sign up for our newsletter, please enter your email address*:

updatED

Flex 2 Beta
The Flex 2 Beta is now available for download at **Adobe Labs**. Still no Mac version though :(
by ED on 01 Feb 2006

CodeTree
Our **Processing book**, is still a couple of months away, so in the meantime why not visit **CodeTree** for some inspiration.
by ED on 31 Jan 2006

fastr
more flickr **fun**
by ped on 30 Jan 2006

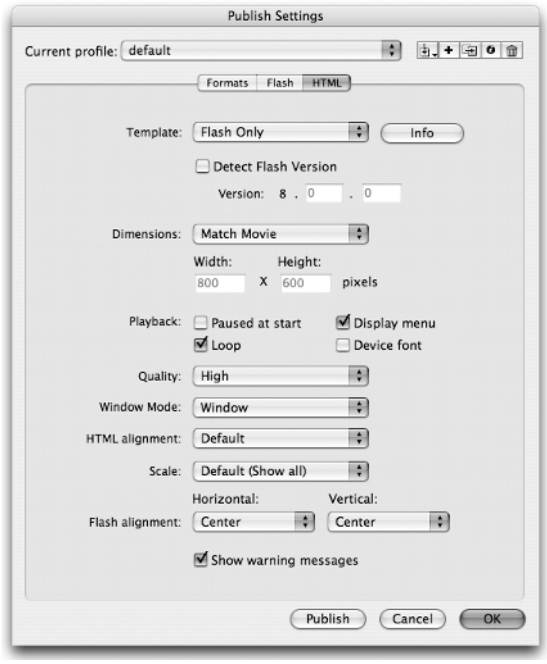
Rustboy
Brian Taylor's wonderful **Rustboy**, is now available as a limited-edition vinyl figure via

The case study: Futuremedia

Throughout the course of this book you'll create a website called Futuremedia from scratch. You can access a fully functioning version of the website you'll be building as you progress through this book at the URL you'll find on the downloads page (or you can go to `www.futuremedia.org.uk` for the latest incarnation).

Centering the Futuremedia site in the browser

When you publish the Futuremedia site, you should use the following settings in the File ➤ Publish Settings ➤ HTML tab:



Publish Settings

Current profile: default

Formats | Flash | HTML

Template: Flash Only
Info

☐ Detect Flash Version
Version: 8 . 0 . 0

Dimensions: Match Movie
Width: 800 X Height: 600 pixels

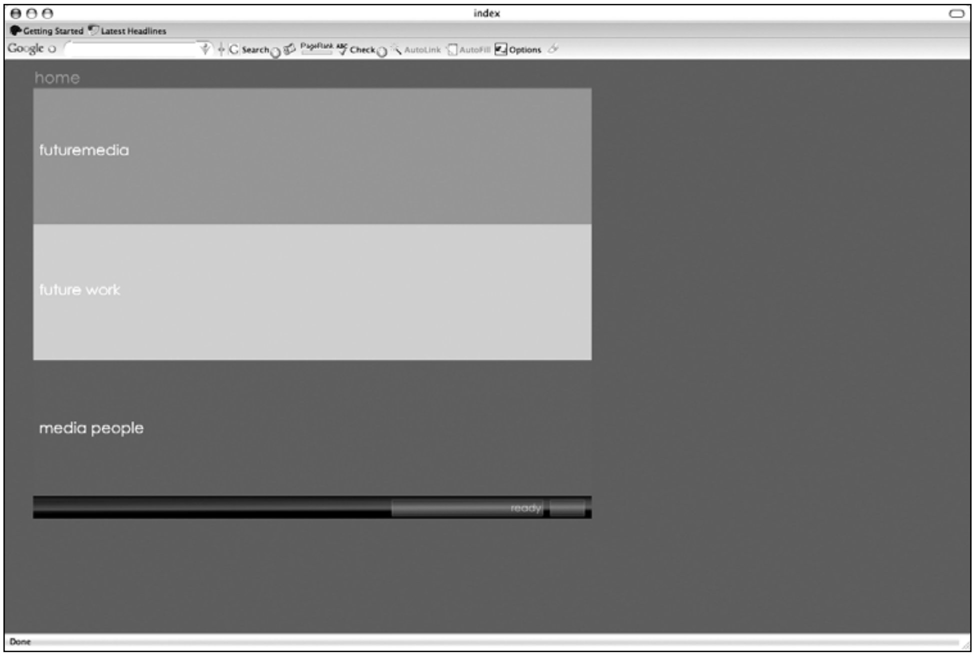
Playback: ☐ Paused at start ☑ Display menu
☑ Loop ☐ Device font

Quality: High
Window Mode: Window
HTML alignment: Default
Scale: Default (Show all)
Flash alignment: Horizontal: Center Vertical: Center

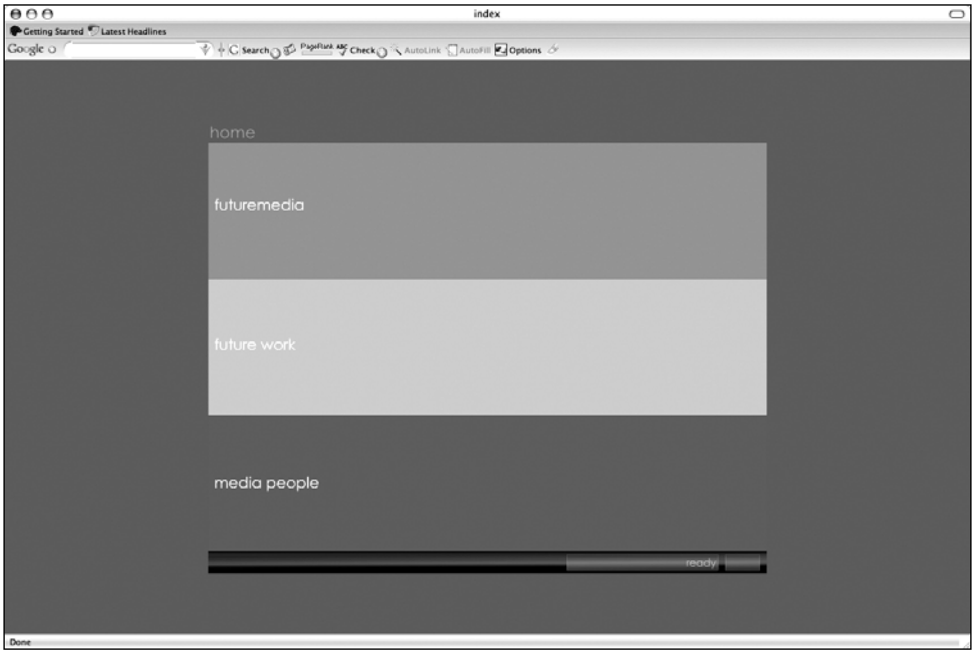
☑ Show warning messages

Publish | Cancel | OK

With these settings, you'll see something like this in the browser (press F12 to publish the site and view it in your browser):



That's fine, but most professional sites center the Flash site in the browser, so it looks something like this instead:



There's no direct way of achieving this in Flash—you have to edit the HTML. To do this, find the HTML file created by Flash (it will be in the same folder as the FLA), and open it in a text editor such as Notepad. You'll see something like this:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" ➥
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns=http://www.w3.org/1999/xhtml xml:lang="en" lang="en">
<head>
<meta http-equiv="Content-Type" content="text/html;➥
  charset=iso-8859-1" />
<title>index</title>
</head>
<body bgcolor="#666666">
<!--urls used in the movie-->
<!--text used in the movie-->
<!--
  futuremedia
  future work
  media people
  loading:
  this is a skip-intro free site
-->
```

```
<object classid="clsid:d27cdb6e-ae6d-➥
  11cf-96b8-444553540000" codebase=➥
  "http://fpdownload.macromedia.com/pub/➥
  shockwave/cabs/flash/swflash.cab#➥
  version=8,0,0,0" width="800" height=➥
  "600" id="index" align="middle">
<param name="allowScriptAccess" value="sameDomain" />
<param name="movie" value="index.swf" />
<param name="quality" value="high" />
<param name="bgcolor" value="#666666" />
<embed src="index.swf" quality="high"➥
  bgcolor="#666666" width="800" height=➥
  "600" name="index" align="middle"➥
  allowScriptAccess="sameDomain" type=➥
  "application/x-shockwave-flash"➥
  pluginspage="http://www.macromedia.com/➥
  go/getflashplayer" />
</object>
</body>
</html>
```

This is XHTML, so you should really play ball and use CSS and `<div>` and `<span>`, and no HTML tables or table horizontal and vertical centering (not least because vertical centering of a table doesn't work in XHTML!). Add the following lines to create a CSS-based centered-in-browser page:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" ➥
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns=http://www.w3.org/1999/xhtml xml:lang="en" lang="en">
<head>
<meta http-equiv="Content-Type" content="text/html;➥
  charset=iso-8859-1" />
<title>index</title>
<style type="text/css">
<!--
body
{
 margin: 0px;
 background-color:#666666;
}
#centercontent
{
 text-align: center;
 margin-top: -300px;
 margin-left: -400px;
 position: absolute;
 top: 50%;
 left: 50%;
}
```

```
-->
</style>
</head>
<body>
<!--urls used in the movie-->
<!--text used in the movie-->
<!--
  futuremedia
  future work
  media people
  loading:
  this is a skip-intro free site
-->
<div id="centercontent">
<object classid="clsid:d27cdb6e-ae6d-➥
  11cf-96b8-444553540000" codebase=➥
  "http://fpdownload.macromedia.com/pub/➥
  shockwave/cabs/flash/swflash.cab#➥
  version=8,0,0,0" width="800" height=➥
  "600" id="index" align="middle">
<param name="allowScriptAccess" value="sameDomain" />
<param name="movie" value="index.swf" />
<param name="quality" value="high" />
<param name="bgcolor" value="#666666" />
<embed src="index.swf" quality="high"➥
  bgcolor="#666666" width="800" height=➥
  "600" name="index" align="middle"➥
  allowScriptAccess="sameDomain" type=➥
  "application/x-shockwave-flash"➥
  pluginspage="http://www.macromedia.com/➥
  go/getflashplayer" />
</object>
</div>
</body>
</html>
```

Note also that the deprecated `bgcolor` attribute has been removed from the `<body>` element and replaced with a nice shiny new standards-compliant CSS rule.

Layout conventions

To keep this book as clear and easy to follow as possible, the following text conventions are used throughout.

Important words or concepts are normally highlighted on the first appearance in **bold type**.

Code is presented in `fixed-width font`.

New or changed code is normally presented in **`bold fixed-width font`**.

Pseudo-code and variable input are written in *`italic fixed-width font`*.

Menu commands are written in the form Menu ➤ Submenu ➤ Submenu.

Where we want to draw your attention to something, we've highlighted it like this:

Ahem, don't say we didn't warn you.

Sometimes code won't fit on a single line in a book. Where this happens, we use an arrow like this: ➥.

```
This is a very, very long section of code that should be written ➥
all on the same line without a break.
```

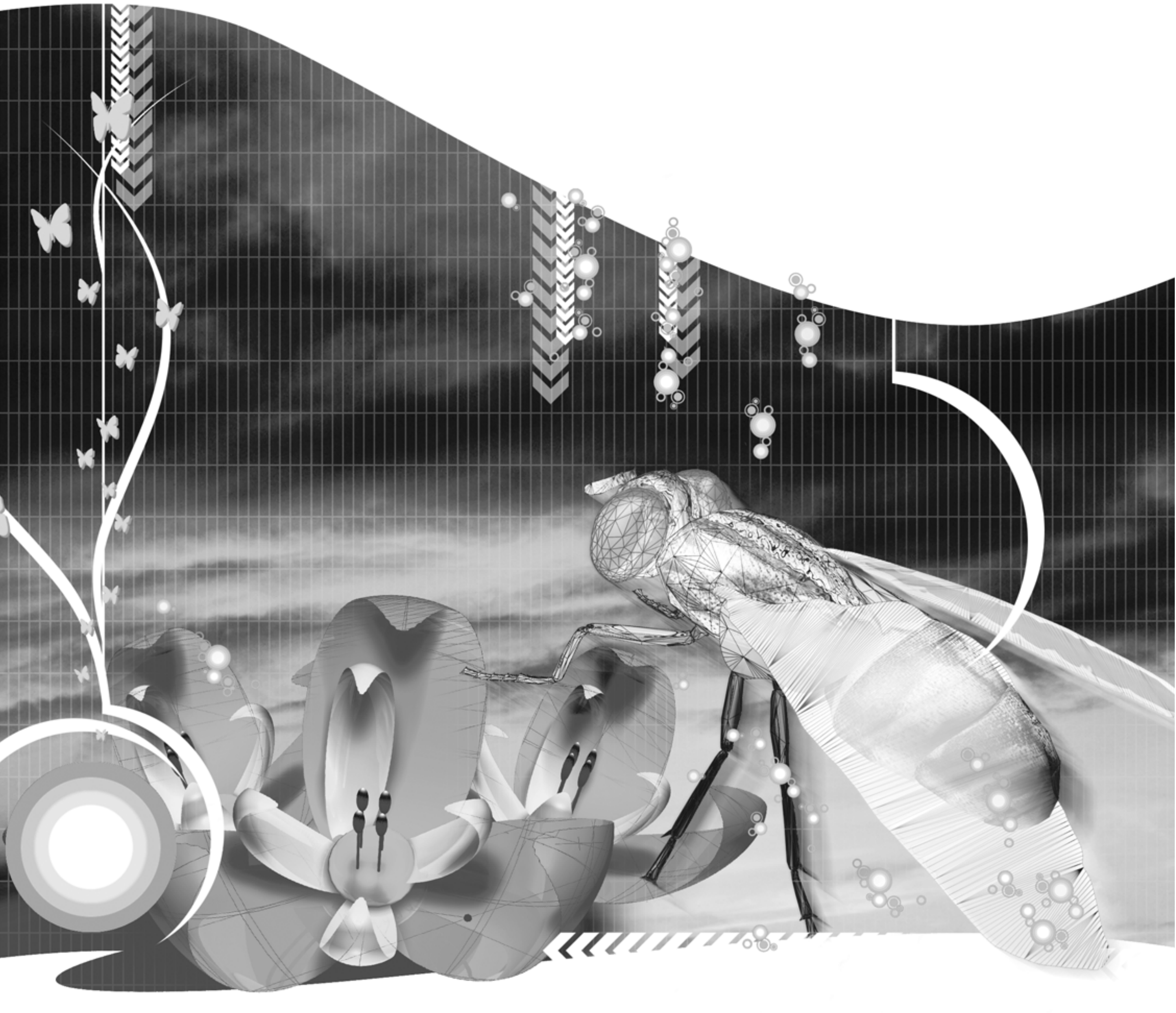
PCs and Macs

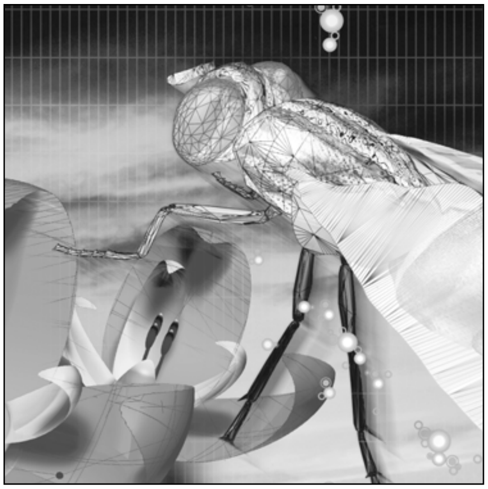
To make sure this book is as useful to you as possible, we've tried to avoid making too many assumptions about whether you're running Flash on a PC or a Mac. However, when it comes to mouse clicks and modifier buttons, we can't generalize. There's no two ways about it: they're different!

When we use the term "click," we mean left-click on the PC or simply click on the Mac. On the other hand, a right-click on the PC corresponds to holding down the Ctrl button and clicking on the Mac. This is abbreviated as Ctrl-click.

Another important key combination on the PC is when you hold down the Ctrl key while you press another key (or click something). This sort of thing is abbreviated as Ctrl-click. The Mac equivalent is to hold down the Cmd key (also known as the "apple" or "flower" key) instead, so you'll normally see this written out in the form Ctrl+C/Cmd+C, or Ctrl+V/Cmd+V, for example.

OK, now that we've taken care of the preliminaries, let's get to work!





Chapter 1

INTERACTIVE FLASH

What we'll cover in this chapter:

- Introducing ActionScript and the Actions panel
- Using actions to give commands to a Flash movie
- Targeting actions at different things in a movie
- Listening to Flash and handling Flash events
- Writing callbacks so that real-time events can trigger actions
- Using variables as containers for storing information

So, what's this ActionScript business all about? You may well have done loads of great stuff with Flash already, but you still keep hearing people say you've barely scratched the surface because you haven't learned to **program** your movies. On the other hand, programming is supposed to be a strange and highly complex business, practiced by myopic young people with pale skin and peculiar taste in music. Programming is for hackers, not artists. That's a path in life many would probably rather avoid.

Fortunately, there's some good news. Programming is a bit like mountaineering: there are dedicated full-timers who might spend every waking minute preparing to tackle the north face of Eiger, even if that means their day-to-day lives come to resemble some kind of hideous training regimen. However, there are plenty more folks who'll just take a few weeks out from "normal life" to join a trek up Mount Kilimanjaro and get back home fit, tanned, and with a whole new perspective on things—not to mention a great big wad of stunning photos!

We're not going to suggest that learning ActionScript is somehow going to get you fit and tanned. But it can help you to race ahead with making rich, powerful, truly fantastic Flash movies that will leave most non-ActionScripting Flash-meisters wondering how to even begin creating such things. So, to get back to the point, what's this ActionScript business all about? Well, as far too many irritating game show hosts have reminded us over the years, "The clue is in the question." There are two words to consider:

Action

Script

We all know what these words mean: an **action** is something you do, and a **script** is something actors read so that they know what to say and do when they go on stage or in front of a camera. Well, that's really all there is to it. ActionScript is like a script that you give to a Flash movie, so that you can tell it what to do and when to do it—that is, you give it **actions** to perform.

The analogy between ActionScript and film scripts is more than just a metaphor. When writing ActionScript, I often like to assume that I'm writing scripts for human actors, and before I write code, I figure out how I would say what I want to do to a real actor before implementing it in ActionScript. This is actually a very good test—if you find you aren't able to find the words needed, it's probably because you don't really understand your problem, and it's time to turn away from the machine and work it out on paper first.

—Sham Bhangal

Giving your movies instructions

Actions are the basic building blocks of ActionScript. It's sometimes helpful to think of them as instructions you give to the movie, or even better, as **commands**. You tell the movie to stop. You tell it to start again. You tell it to go to such and such a frame, and start playing from there. You probably get the idea. Whenever you want to tell your movie to do something that it wouldn't normally do anyway, you can do so by giving it an action.

Since this book is about ActionScript, we'll stick to using the word "actions." If you mention actions to programmers who don't use Flash, though, they may look at you blankly for a moment before saying, "Ah, you mean commands!" In our view, it's well worth keeping the idea that actions are the same thing as commands at the back of your mind.

As you know if you've sneaked a look already, there are stacks of different actions available in Flash, which makes it (a) possible to do all sorts of weird and wonderful things to your movies, and (b) pretty hard to know where to start. Don't worry, though, we have a cunning plan! You'll start off by looking at some simple actions that let you start and stop a movie's main timeline, and then you'll move on to look at jumping about from one frame to another. Nothing too tricky to begin with.

Before you start writing actions, though, we need to make an important introduction. The **Actions panel** will be your partner in crime throughout this chapter, the rest of the book, and your future career as an ActionScript hero. Let's check it out.

If you own the Professional version of Flash, you have the ability to open a full-screen ActionScript editor by choosing File ➤ New *and then selecting* ActionScript File *from the* General *tab of the* New Document *window that will appear. This editor is actually very similar to the normal* Actions *panel. The major difference with the full-screen editor is it's just that—there's no stage and no timeline. The full-screen editor is designed for writing a stand-alone ActionScript file, something you'll have the option to use as you build your website. Just because this is a beginner book doesn't mean we're not going to fully cover the Professional version of Flash as well as the Basic edition, so don't worry, we haven't forgotten you!*

Working with the Actions panel

The Actions panel is where you'll be writing all your ActionScript, so it will pay for you to know your way around it before you make a proper start. Before you do, though, let's make sure we're all on the same page. Open a new Flash document and select Window ➤ Workspace Layout ➤ Default, and the interface should go back to the out-of-the-box state.

If you ever get to a point at which you have many panels open and a generally cluttered screen, Window ➤ Workspace Layout ➤ Default *is a good option to select—you don't lose your work in progress by doing it. It's also a good idea to select this option before you start any of the step-by-step tutorials.*

First, you need to open up the Actions panel. As with any panel, there are several ways to do this, but for now we'll go for the Window ➤ Actions menu selection as the cross-platform-friendly option.

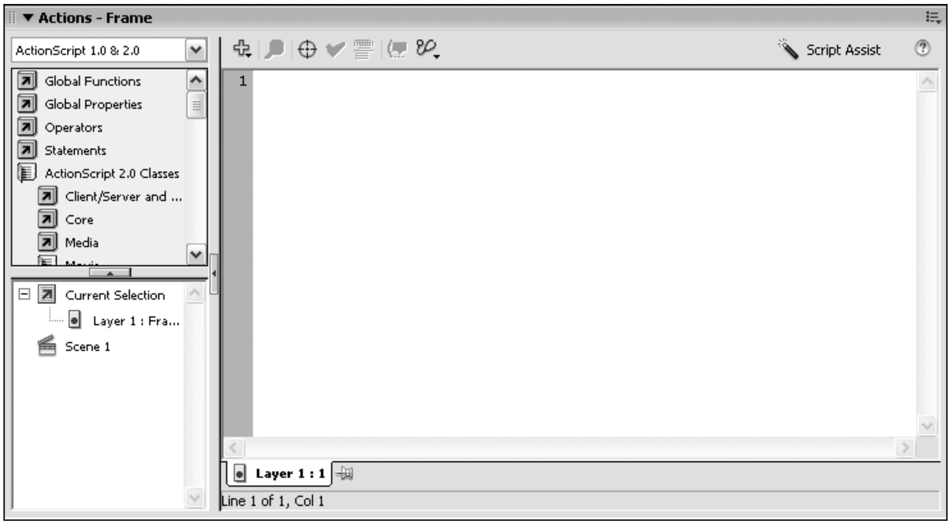
You can also open (and close) the Actions panel with the F9 keyboard shortcut.

Here's a can of worms . . .

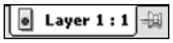
If you're using a Mac with Exposé enabled with the default settings, you'll soon discover that pressing F9 triggers Exposé and doesn't open the Actions *panel in Flash at all! If this is the case and you don't want to mess with Exposé, pressing Option+F9 will open the* Actions *panel.*

However, those who were already hooked on the F9 shortcut before Apple introduced Exposé will have to change the keyboard shortcut for their little zoomy friend Exposé.

However you opened the Actions panel, this is how it should look when you open it for the first time:



Let's take a look at what we have here. At the bottom is a little tab that looks like this:



This tab tells you that any actions you add are going to be **attached to** a frame. You also know the actions will be attached to a layer called Layer 1, and on frame 1 of that layer via the text Layer 1 : 1 . . . um . . . you also know this must be true because that's the only frame you have at the moment!

Actions are always attached to something—usually a keyframe. (You can also attach a script to a movie clip or button using an older style of ActionScript coding, but you will *not* be using that in this book—it's outdated and not recommended anymore.)

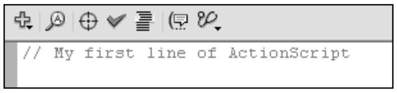
You tell Flash what you're attaching to by selecting it before you start typing in the Actions panel. The tab is very useful because it reminds you what you're attaching to and where it is. Sometimes this may not be obvious (because you have, for example, selected the wrong thing in error), and you should always get into the habit of checking the tab to make sure what Flash thinks it should be attaching to is what you want it to attach the script to. Most scripts will *not work properly* (or at all) if you get this wrong.

Sometimes you'll find that the script you're writing suddenly vanishes. This is one of those "Oh my God, I've lost all my stuff!" moments that every beginner has, but not to worry, it usually only means the keyframe you're attaching to has somehow become unselected, and something on the main stage is now selected instead. Before you start writing it all out again and glare at your cat/dog/co-worker for ruining your concentration and/or day, it usually helps to try reselecting the keyframe first.

Once you've selected a frame, you add scripts via the Script pane (the big white area to the right of the Actions panel). Simply click inside the Script pane and start typing. Type the following code and then press the Enter key:

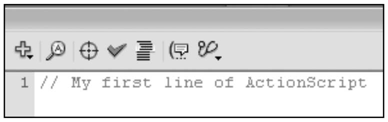
```
// My first line of ActionScript
```

*Flash calls the little windowlike areas within a panel **panes** (as in, "The panel is a window, and the bits inside it are the windowpanes").*



```
// My first line of ActionScript
```

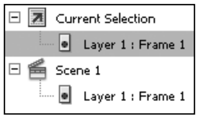
The // tells Flash that this is a **comment** rather than code, and Flash will actually ignore everything after the // until you press Enter. Adding comments to your scripts is a good way to remind yourself what the scripts do six months after you've written them and/or 20 minutes after you've written them, depending on how bad a day you're having. Press Enter on your keyboard to start a new line. Your scripts will consist of many such lines, and it's a good idea to *number* them so you know how far down a script you are (this also makes it easier for us to refer to line numbers later in the book). Select the little pop-up menu icon (at the top-right corner of the Actions panel) and select Line Numbers in the menu that appears.



```
1 // My first line of ActionScript
```

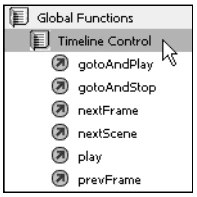
OK, let's look at the other goodies. On the left side are two more panes.

You'll notice that there's a little letter "a" above frame 1 on the timeline. This tells you that the keyframe here has a script attached to it. That's fine for small FLAs with not many timelines, but for bigger Flash productions, you need a centralized place that allows you to search out all your stuff that has ActionScript attached to it. The bottom-left pane, called the **Script navigator**, does just that. It consists of two file trees. The top one shows where the currently displayed script (in the Script pane) is within the Flash hierarchy, and the bottom one shows all scripts in the current FLA. Because you don't know much about the timeline hierarchy and how it affects ActionScript just yet, we'll leave further discussion of this pane until later, except to say that the reason the two trees look the same at the moment is because the one script you're looking at is the only one in the FLA.



The top-left panel (the **Actions toolbox**) is a list of little "book" icons, the first of which is called Global Functions. If you click it, the icon will "open" to reveal more books.

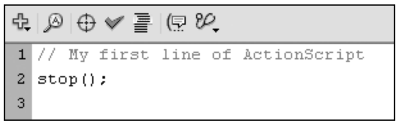
If you also open the Timeline Control book inside the Global Functions book, you'll see some icons:



These icons represent **actions**, and the ones in the Timeline Control book are the most basic ones, so you'll be using them soon. The other books contain operators, functions, constants, and other bits and pieces you can use to tie lots of actions together in the way you want. If you hover the mouse pointer over one of them, a tooltip should appear that gives you a brief definition of the book's contents and where it's used.

Don't worry about the details of all these for now, though—you'll learn about them soon enough. This is the Actions toolbox, and it gives you a neat little way to write code in a no-brainer, "building blocks" fashion.

Look down the list of actions in the Timeline Control book and double-click the second-to-last one, stop. This adds a `stop()` action to your Script pane at line 2:



```
// My first line of ActionScript
stop();

```

You now have an action called `stop()` that is attached to the first frame on the layer called Layer 1. This action will stop the timeline at frame 1. Looking at the rest of the actions in the Timeline Control book, you'll have probably figured out that the `play()` action makes the timeline start playing, and so on.

Direct typing

Well, you've made a start. All the actions you're ever going to need are stashed away inside those icons, so you could continue using this technique to write all the code you're ever likely to need. What's more, working in this way will mean that there won't be any errors in your script (such as mistyping the `stop()` action as `slop()`), so you'll find it very hard to create errors in this way.

However, there's another option to consider: you can bypass the books completely and type your scripts directly into the Script pane.

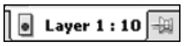
If you have a nervous disposition, this may seem like a worrying step. "It's only the first chapter, but already you're expecting me to type code and get it right first time?" you cry. But honestly, it's not like that at all! There are so many assistive elements when typing directly into the Actions panel that there's really no need to use the Actions toolbox. Although searching out the actions and other stuff in the toolbox seems like a safe option (you can't spell them incorrectly if you do it this way, and you have less to remember), it has one problem: it's darn slow! If you take the time to type everything in directly, except for the odd bit of code you can't remember, not only will you work faster, but also you'll get *much* faster with time.

OK, let's try it.

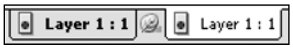
Don't worry about the parentheses and semicolon, `();`, at the end of the `stop` for now. Sure, they make the simple `stop` appear a little otherworldly, and they look suspiciously like they were put there by someone with a serious code-fetish just to scare the nonbelievers off, but they're actually there for some simple reasons that you'll look at later on in the book.

First of all, you want something that will tell you if you've made a mistake. Using the Script pane as you would a normal text editor, try changing `stop();` to `slop();` and you'll see that the first four letters have changed from blue to black. The Script pane will normally show all action names (or **keywords**) in blue, though you may want to use the ActionScript preferences (accessible from Actions ➤ Preferences) to change this color to a more noticeable shade of blue.

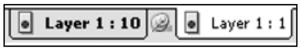
Second, you'll want to fix that "disappearing script" problem mentioned earlier. Just select frame 10 on the main timeline, add a keyframe (press F6), and select the keyframe to see what we mean—look, no more script! Of course, that's not quite true. Your script is still attached to frame 1, but you're looking at the script attached to frame 10, and there currently isn't one. You can see this if you remember to look at the tab at the bottom of the Script pane, though. It has changed:



So what if you want to work on the graphics on the stage at frame 10 while you're looking at the script in frame 1? Easy! Just click frame 1 again, and then click the pushpin button that's just to the right of the tab. This is called **pinning**. The tab won't disappear, even if you select something else. So now you're thinking, "Yeah, I see that, but now I have *two* tabs that show the same thing!"



The one on the right (the light one) is the one that's pinned, and the one on the left shows the frame or object currently selected. The confusing thing at the moment is that they're both showing the same thing. All becomes clear when you click back on frame 10. The tab on the left will change, and the pinned one to the right will stay as it is.



You can pin as many scripts as you need. Pinned scripts will go to the right, and the script for the currently selected item will appear to the left. Some applications with a similar tabbed system work the other way around, which may be confusing to some.

Pinning is a very useful feature to know about, but only so long as you remember to turn it off when you start trying to add actions elsewhere. Click the pushpin button again to unpin the script, and click back on the first frame. Now you're ready to continue.

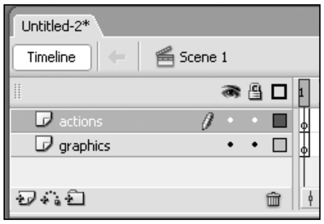
One more thing that's well worth knowing at the start about Flash is that it's **case sensitive**. When typing code, you generally have to get both the spelling *and* the capitalization right; otherwise, the code may not work.

The Actions panel will lie at the center of all the ActionScript that you'll ever write. You'll explore a few more of its mysteries as you work through the book, but you now have enough information to start and create your first ActionScript-enabled movie.

Several more helpful features are built into the Actions *panel—you've probably already noticed all the buttons running along the top of the* Script *pane. You won't examine them right away, though; it will be easier to demonstrate what they do once you have a little more scripting experience under your belt.*

Let's start with a simple example that shows how easily you can bypass the usual flow of a movie with just a tiny bit of ActionScript. You can close the current FLA if you wish (you don't have to save it).

1. Create a new Flash Document movie and add a new layer to the root timeline. Rename the original layer as graphics and add a new layer called actions. This is a good way to start all your ActionScripted movies.



Rather than use the File ➤ New *menu option, you can right-click/Cmd-click the tabs above the timeline to open/close/save files—this is much quicker!*

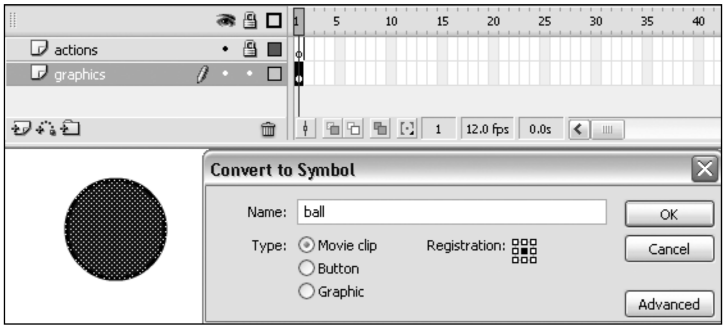
Scripts should normally always be attached to keyframes on the topmost layer (if you put them on the bottommost layer, the scripts may not work on the Web because then they'll be loaded before the graphics, and this tends to confuse Flash). It's also a good idea to have only scripts in the topmost layer, which is why you'll call this layer actions—to remind you that that's what this layer contains, and nothing else. As you progress through the book, you'll quickly become familiar with the idea of having an actions layer residing at the top of the timeline. Naming a script layer actions is a very common convention among Flashers.

A good trick while you're working is to always keep the actions *layer locked. This stops you from inadvertently placing graphics in this layer, but it doesn't stop you from attaching scripts to it or modifying the frames and keyframes in the* Timeline *panel. Locking a layer simply prevents you from editing the contents of the stage.*

2. Lock the actions layer (for the reason just noted).

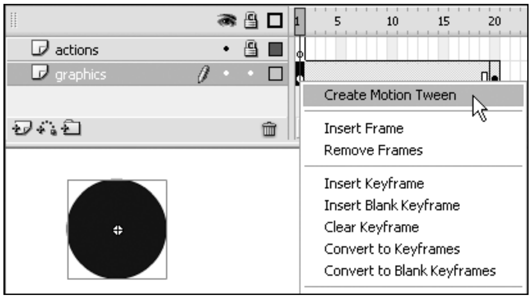
Let's add those graphics now. You don't need any terribly fancy graphics to demonstrate what's going on with the timeline. For simplicity's sake, let's just create a simple motion tween between two points.

3. Select frame 1 on the graphics layer, and use the Oval tool to draw a circle on the left side of the stage. With the Selection tool, select the oval (double-click it on the fill to select both the fill and the outline stroke) and press F8 to make it into a symbol. The Convert to Symbol dialog box will appear. Select the Movie clip option and call it ball.



Name: ball
Type: (•) Movie clip ( ) Button ( ) Graphic
Registration:

4. Now select frame 20 of the graphics layer and press F6 to insert a keyframe. Right-click (or Ctrl-click on the Mac) frame 1, and select Create Motion Tween from the menu that appears.


actions
graphics
5
10
15
20
Create Motion Tween
Insert Frame
Remove Frames
Insert Keyframe
Insert Blank Keyframe
Clear Keyframe
Convert to Keyframes
Convert to Blank Keyframes