

# Object-Oriented ActionScript for Flash 8

Peter Elst and Todd Yard  
with Sas Jacobs and William Drol



# Object-Oriented ActionScript for Flash 8

Copyright © 2006 by Peter Elst, Todd Yard, Sas Jacobs, and William Drol

All rights reserved. No part of this work may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or by any information storage or retrieval system, without the prior written permission of the copyright owner and the publisher.

ISBN-13 (pbk): 978-1-59059-619-7

ISBN-10 (pbk): 1-59059-619-6

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

Trademarked names may appear in this book. Rather than use a trademark symbol with every occurrence of a trademarked name, we use the names only in an editorial fashion and to the benefit of the trademark owner, with no intention of infringement of the trademark.

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](http://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](mailto:info@apress.com), or visit [www.apress.com](http://www.apress.com).

The information in this book is distributed on an “as is” basis, without warranty. Although every precaution has been taken in the preparation of this work, neither the author(s) nor Apress shall have any liability to any person or entity with respect to any loss or damage caused or alleged to be caused directly or indirectly by the information contained in this work.

The source code for this book is freely available to readers at [www.friendsofed.com](http://www.friendsofed.com) in the Downloads section.

## Credits

**Lead Editor**      **Assistant Production Director**  
Chris Mills      Kari Brooks-Copony

**Technical Reviewers**      **Production Editor**  
Jared Tarbell,      Katie Stence  
Stephen Downs

**Editorial Board**      **Compositor**  
Steve Anglin, Dan Appleman,      Dina Quan  
Ewan Buckingham, Gary Cornell,      **Proofreader**  
Jason Gilmore, Jonathan Hassell,      April Eddy  
James Huddleston, Chris Mills,      **Indexer**  
Matthew Moodie, Dominic Shakeshaft,      Michael Brinkman  
Jim Sumser, Matt Wade

**Project Manager**      **Artist**  
Sofia Marchant      April Milne

**Copy Edit Manager**      **Interior and Cover Designer**  
Nicole LeClerc      Kurt Krames

**Copy Editor**      **Manufacturing Director**  
Ami Knox      Tom Debolski

# CONTENTS

**Foreword** . . . . . **xv**  
**About the Authors** . . . . . **xvi**  
**About the Technical Reviewer** . . . . . **xvii**  
**Acknowledgments** . . . . . **xviii**

**PART ONE: OOP AND ACTIONSCRIPT** . . . . . **xx**

---

**Chapter 1: Introduction to OOP** . . . . . **1**  
    The scoop with OOP? . . . . . 2  
    Understanding the object-oriented approach . . . . . 2  
        Classes and objects . . . . . 3  
        Properties . . . . . 3  
        Encapsulation: Hiding the details . . . . . 4  
        Polymorphism: Exhibiting similar features . . . . . 7  
        Inheritance: Avoid rebuilding the wheel . . . . . 8  
    What's next? . . . . . 9

**Chapter 2: Programming Concepts** . . . . . **11**  
    About programming slang . . . . . 12  
    Building blocks of programming . . . . . 13  
        Variables . . . . . 13  
            About variable data . . . . . 14  
    Arrays . . . . . 14

Functions . . . . .	15
About calling functions . . . . .	15
About function parameters . . . . .	15
Loops . . . . .	16
Conditionals . . . . .	16
OOP concepts . . . . .	16
What's next? . . . . .	17

**Chapter 3: ActionScript 2.0 Programming . . . . . 19**

ActionScript 1.0 vs. ActionScript 2.0 . . . . .	20
Declaring variables . . . . .	20
Classes vs. prototypes . . . . .	21
Public and private scope . . . . .	25
Strong typing and code hints . . . . .	27
ActionScript trouble spots . . . . .	29
Case sensitivity . . . . .	29
Declaring variables . . . . .	30
Use of the this keyword . . . . .	30
What's next? . . . . .	31

**PART TWO: FLASH OOP GUIDELINES . . . . . 32**

---

**Chapter 4: Planning . . . . . 33**

The importance of planning . . . . .	34
Initial phase: Planning reusability! . . . . .	35
Planning encapsulation . . . . .	35
Planning inheritance . . . . .	36

## CONTENTS

Analyzing a Flash ActionScript project . . . . .	39
Flash files run on the client . . . . .	39
Securing data sent to the server . . . . .	39
Parsing data in Flash . . . . .	40
Introduction to UML modeling . . . . .	40
Why use UML? . . . . .	41
UML offers standardized notation and has a language-neutral syntax . . . . .	41
UML can be used to model anything . . . . .	42
Class diagram . . . . .	42
Association and generalization . . . . .	43
Aggregation and composition . . . . .	44
What's next? . . . . .	46
<b>Chapter 5: Project Workflow . . . . .</b>	<b>49</b>
Introducing version control . . . . .	50
About Concurrent Versions System . . . . .	50
Using TortoiseCVS . . . . .	52
Approaches to programming . . . . .	58
Rapid Application Development . . . . .	59
Extreme Programming . . . . .	60
Usability testing . . . . .	62
What's next? . . . . .	63
<b>Chapter 6: Best Practices . . . . .</b>	<b>65</b>
External ActionScript . . . . .	66
About commenting . . . . .	68
Naming conventions . . . . .	70
Variables . . . . .	70
Constants . . . . .	71
Functions . . . . .	71
Classes . . . . .	71
Methods . . . . .	72
Properties . . . . .	72
Packages . . . . .	72
Programming styles . . . . .	73
Alternative programming styles . . . . .	77
Coding practices: Todd Yard . . . . .	77
Coding practices: Sas Jacobs . . . . .	79
Commenting code . . . . .	79
Naming conventions . . . . .	80
What's next? . . . . .	81

<b>PART THREE: CORE OOP CONCEPTS</b> .....	<b>82</b>
<hr/>	
<b>Chapter 7: Encapsulation</b> .....	<b>83</b>
Setting up encapsulation .....	84
Creating new layers .....	85
Drawing a background .....	87
Aligning and locking the background .....	87
Drawing a ball .....	88
Converting the ball into a Library symbol .....	89
Content summary .....	90
Writing the code .....	91
Creating an event handler .....	91
What about encapsulation? .....	93
Testing the event handler .....	94
Updating the Ball .....	95
Improving the code .....	96
Enhancing behavior with properties .....	96
Narrowing the focus with functions .....	97
Encapsulation summary .....	99
What's next? .....	101
<b>Chapter 8: Classes</b> .....	<b>103</b>
Classes vs. prototypes .....	104
Constructors .....	106
About this .....	109
Methods .....	112
Anonymous functions .....	113
Implementing a class .....	116
The Mover class .....	116
What's next? .....	118
<b>Chapter 9: Inheritance</b> .....	<b>121</b>
About class hierarchy .....	122
A quick inheritance test .....	122
About inheritance syntax .....	125
The Bouncer class .....	126
The Gravity class .....	129
Inheritance summary .....	133
What's next? .....	133

## CONTENTS

<b>Chapter 10: Polymorphism</b> . . . . .	<b>135</b>
Building a polymorphism example . . . . .	136
Implementing polymorphism for application reuse . . . . .	138
Basic concept of polymorphism . . . . .	138
Functional polymorphism at work . . . . .	139
What's next? . . . . .	142
<b>Chapter 11: Interfaces</b> . . . . .	<b>145</b>
Interfaces overview . . . . .	146
Interface use-cases . . . . .	147
What an interface looks like . . . . .	147
Implementing an interface . . . . .	148
What's next? . . . . .	155
<b>Chapter 12: Design Patterns</b> . . . . .	<b>157</b>
Understanding design patterns . . . . .	158
Implementing design patterns . . . . .	160
Observer pattern . . . . .	160
Basic implementation . . . . .	160
Practical implementation . . . . .	167
Extending the practical implementation . . . . .	169
Singleton pattern . . . . .	171
Basic implementation . . . . .	172
Practical implementation . . . . .	177
Building an interface . . . . .	181
Decorator pattern . . . . .	183
Basic implementation . . . . .	183
Practical implementation . . . . .	184
Applying the Decorator pattern . . . . .	186
Model-View-Controller pattern . . . . .	191
Basic implementation . . . . .	192
Practical implementation . . . . .	193
Bringing together the Model, View, and Controller . . . . .	196
Design patterns summary . . . . .	197
What's next? . . . . .	198
<b>Chapter 13: Case Study: An OOP Media Player</b> . . . . .	<b>201</b>
Planning the player . . . . .	202
Picking a pattern . . . . .	202
Guaranteeing methods and datatypes with an interface . . . . .	203
Examining class structure . . . . .	204

Building the media player . . . . .	206
IntervalManager . . . . .	207
Defining the interfaces . . . . .	209
Dispatching events . . . . .	209
Media interfaces . . . . .	214
Controlling media . . . . .	215
Defining properties . . . . .	215
Private methods . . . . .	216
Public methods . . . . .	218
Controlling FLVs . . . . .	222
Building a video view . . . . .	228
Controlling SWFs . . . . .	229
Building a SWF view . . . . .	236
Controlling MP3s . . . . .	238
Summary . . . . .	239
What's next? . . . . .	239

## **PART FOUR: BUILDING AND EXTENDING A DYNAMIC FRAMEWORK . . . . . 240**

---

### **Chapter 14: Framework Overview . . . . . 241**

Introducing the framework . . . . .	242
Understanding the MovieClip class . . . . .	246
Understanding the UIObject class (mx.core.UIObject) . . . . .	253
Understanding the UIComponent class (mx.core.UIComponent) . . . . .	255
Understanding the View class (mx.core.View) . . . . .	256
Framework summary . . . . .	257
What's next? . . . . .	257

### **Chapter 15: Manager Classes . . . . . 259**

Planning the framework . . . . .	260
What to manage . . . . .	260
Diagramming the classes . . . . .	261
Building managers . . . . .	263
StyleFormat . . . . .	263
StyleManager . . . . .	266
Adding style . . . . .	269
SoundManager . . . . .	272
Sounding off . . . . .	275
Summary . . . . .	277
What's next? . . . . .	277

## CONTENTS

### **Chapter 16: UI Widgets . . . . . 279**

Diagramming the classes . . . . .	280
UIObject . . . . .	280
Block . . . . .	283
SimpleButton . . . . .	284
Making the foundation . . . . .	285
Basic building block . . . . .	292
Building a component . . . . .	294
Skinning a widget . . . . .	303
Changing state . . . . .	307
Adding some style . . . . .	308
More ways to skin a cat . . . . .	313
Attaching from scratch . . . . .	316
Tying in events . . . . .	320
Pulling it all together . . . . .	325
Summary . . . . .	329
What's next? . . . . .	330

### **Chapter 17: OOP Animation and Effects . . . . . 333**

Preparing for animation . . . . .	334
Animator . . . . .	335
Tweening properties and values . . . . .	336
Tween . . . . .	336
Easer . . . . .	341
Testing the Tween . . . . .	344
Enhancing Tween . . . . .	347
Mover . . . . .	354
Motion blur . . . . .	357
Transitioning views . . . . .	360
Transition . . . . .	360
FadeTransition . . . . .	363
Testing transitions . . . . .	364
ColorTransition . . . . .	369
BlurTransition . . . . .	371
NoiseTransition . . . . .	374
DissolveTransition and WaterTransition . . . . .	376
Summary . . . . .	381
What's next? . . . . .	381

**PART FIVE: DATA INTEGRATION . . . . . 382****Chapter 18: Interrelationships and Interactions  
Between Components . . . . . 383**

Data binding . . . . .	384
The mx.data.binding package . . . . .	385
Creating a simple binding . . . . .	386
Creating EndPoints . . . . .	386
Specifying a location . . . . .	387
Creating the binding . . . . .	388
Using the execute method . . . . .	388
Working through a simple binding example . . . . .	389
Using formatters . . . . .	395
Using built-in formatters . . . . .	396
Using the Boolean formatter . . . . .	396
Using the Compose String formatter . . . . .	396
Using the Date formatter . . . . .	397
Using the Rearrange Fields formatter . . . . .	397
Using the Number formatter . . . . .	397
Working through a simple formatting example . . . . .	398
Understanding custom formatters . . . . .	404
Including validators . . . . .	406
Working with built-in validators . . . . .	407
Working with a custom validator . . . . .	413
Summary . . . . .	415
What's next? . . . . .	415

**Chapter 19: Communication Between Flash and  
the Browser . . . . . 417**

Communication with Flash Player 7 and below . . . . .	419
Sending variables into Flash . . . . .	419
Calling JavaScript from Flash . . . . .	419
Using the Flash/JS Integration Kit . . . . .	421
Understanding the ExternalInterface class . . . . .	423
Understanding Flash Player 8 security . . . . .	424
Using the call method . . . . .	424
Using the addCallback method . . . . .	429
ActionScript communication with other languages . . . . .	434
Calling a non-JavaScript method . . . . .	434
Calling an ActionScript method from an application . . . . .	435
Summary . . . . .	435
What's next? . . . . .	436

## CONTENTS

<b>Chapter 20: Server Communication (XML and Web Services)</b> . . . . .	<b>439</b>
Understanding XML . . . . .	440
XML declarations . . . . .	442
Using XML in Flash . . . . .	443
XMLConnector component . . . . .	443
XML class . . . . .	447
What are web services? . . . . .	454
Understanding SOAP . . . . .	454
Talking to web services . . . . .	456
WebServiceConnector component . . . . .	456
WebService class . . . . .	464
Flash Player security sandbox . . . . .	478
System.security.allowDomain() . . . . .	478
Cross-domain policy files . . . . .	478
Using a server-side proxy script . . . . .	480
Summary . . . . .	480
What's next? . . . . .	481
<b>Chapter 21: Case Study: Time Sheet Application</b> . . . . .	<b>483</b>
Planning the application . . . . .	484
Structuring the application . . . . .	485
Writing stub code . . . . .	487
Model-View-Controller classes . . . . .	487
TimeSheetModel class (Model) . . . . .	487
TimeSheetView class (View) . . . . .	488
TimeSheetController class (Controller) . . . . .	490
Project and Task classes . . . . .	491
Project class . . . . .	491
Task class . . . . .	492
Bringing it all together . . . . .	493
Initializing the layout . . . . .	493
Adding a project . . . . .	495
Displaying projects . . . . .	498
Adding a task . . . . .	501
Project and task details . . . . .	506
Running a task timer . . . . .	510
Persisting time sheet data . . . . .	512
Summary . . . . .	517
Conclusion . . . . .	518
<b>Index</b> . . . . .	<b>521</b>