

ANTONIO DE DONATIS

AdvancED

# ActionScript Components

Mastering the Flash Component Architecture

Learn how to design component-based applications that work well and look great.

Discover how advanced OOP techniques are applied to the Flash component architecture.

Master the use of every standard component included in both Flash 8 and Flash MX2004.

# AdvancED ActionScript Components

Mastering the Flash Component Architecture

Antonio De Donatis



# AdvancED ActionScript Components: Mastering the Flash Component Architecture

Copyright © 2006 by Antonio De Donatis

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): 978-1-59059-593-0

ISBN-10 (pbk): 1-59059-593-9

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](mailto: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** Chris Mills  
**Assistant Production Director** Kari Brooks-Copony

**Technical Reviewers** Sas Jacobs,  
Paul Barnes-Hoggett  
**Production Editor** Laura Cheu

**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  
**Compositor** Dina Quan  
**Proofreader** Christy Wagner

**Project Manager** Julie M. Smith  
**Indexer** Becky Hornyak  
**Cover Image Designer** Bruce Tang

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

**Copy Editors** Ami Knox, Marilyn Smith  
**Manufacturing Director** Tom Debolski

*To JenFeng*

# CONTENTS AT A GLANCE

---

<b>About the Author</b> .....	<b>xvii</b>
<b>About the Technical Reviewers</b> .....	<b>xviii</b>
<b>About the Cover Image Designer</b> .....	<b>xix</b>
<b>Acknowledgments</b> .....	<b>xx</b>
<b>Introduction</b> .....	<b>xxi</b>
<b>PART ONE INTRODUCING THE ARCHITECTURE</b> .....	<b>1</b>
Chapter 1 <b>OOP in Component Design</b> .....	<b>3</b>
Chapter 2 <b>Core Classes and Component Design</b> .....	<b>33</b>
Chapter 3 <b>Exploring the UI Components</b> .....	<b>59</b>
Chapter 4 <b>Building Component-based Applications</b> .....	<b>91</b>
<b>PART TWO EXPLOITING THE ARCHITECTURE</b> .....	<b>129</b>
Chapter 5 <b>Architecture-based Development</b> .....	<b>131</b>
Chapter 6 <b>XML for Defining User Interfaces</b> .....	<b>149</b>
Chapter 7 <b>Extending the Application Framework</b> .....	<b>171</b>
Chapter 8 <b>Making Your XML Life Easier</b> .....	<b>201</b>
Chapter 9 <b>The Customization Process</b> .....	<b>219</b>

---

<b>PART THREE CUSTOMIZING THE COMPONENTS</b> .....	<b>251</b>
Chapter 10 <b>The Accordion Component</b> .....	<b>253</b>
Chapter 11 <b>The Button Component</b> .....	<b>275</b>
Chapter 12 <b>The CheckBox and RadioButton Components</b> .....	<b>295</b>
Chapter 13 <b>The List, ComboBox, and DataGrid Components</b> .....	<b>309</b>
Chapter 14 <b>The DateChooser and DateField Components</b> .....	<b>349</b>
Chapter 15 <b>The Loader, ScrollPane, and ProgressBar Components</b> .....	<b>371</b>
Chapter 16 <b>The Menu and MenuBar Components</b> .....	<b>395</b>
Chapter 17 <b>The NumericStepper Component</b> .....	<b>427</b>
Chapter 18 <b>The TextArea, TextInput, and Label Components</b> .....	<b>439</b>
Chapter 19 <b>The Tree Component</b> .....	<b>459</b>
Chapter 20 <b>The Window and Alert Components</b> .....	<b>483</b>
Chapter 21 <b>Handling the Scrollbars</b> .....	<b>507</b>
<b>PART FOUR APPENDIXES</b> .....	<b>521</b>
Appendix A <b>Locating the Source Code of the Component Architecture</b> .....	<b>523</b>
Appendix B <b>Transitions and Easing Classes</b> .....	<b>529</b>
<b>Index</b> .....	<b>535</b>

# CONTENTS

---

<b>About the Author</b> . . . . .	<b>xvii</b>
<b>About the Technical Reviewers</b> . . . . .	<b>xviii</b>
<b>About the Cover Image Designer</b> . . . . .	<b>xix</b>
<b>Acknowledgments</b> . . . . .	<b>xx</b>
<b>Introduction</b> . . . . .	<b>xxi</b>
<b>PART ONE INTRODUCING THE ARCHITECTURE</b> . . . . .	<b>1</b>
<b>Chapter 1 OOP in Component Design</b> . . . . .	<b>3</b>
A very short history of the architecture . . . . .	4
Creating Flash components . . . . .	4
An apparently useless component . . . . .	5
Creating the Vogon component . . . . .	5
Adding a method . . . . .	7
Variables, properties, and metadata tags . . . . .	8
Implementing a property explicitly . . . . .	9
Implementing a property implicitly . . . . .	11
Properties in the authoring environment . . . . .	12
Inheritance . . . . .	14
A little help from Darwin . . . . .	15
Appreciating the benefits of inheritance . . . . .	17
ActionScript limit on multiple inheritance . . . . .	19
Events . . . . .	20
What is an event in component terms? . . . . .	21
Implementing a custom event . . . . .	21
Triggering a custom event . . . . .	21
Listening to a custom event . . . . .	23
Building the example . . . . .	23
Polymorphism . . . . .	26
And God took a rib from a Vogon . . . . .	26
A method's signature . . . . .	29
Appreciating the benefits of polymorphism . . . . .	30
Summary . . . . .	31

---

<b>Chapter 2 Core Classes and Component Design</b> . . . . .	<b>33</b>
The legacy of the UIObject class . . . . .	36
Creating a component instance dynamically . . . . .	37
Overriding the symbolName property . . . . .	38
Overriding the symbolOwner property . . . . .	38
Overriding the className property . . . . .	38
The createClassObject method . . . . .	39
Our components join the architecture . . . . .	40
Inside the process of building a component instance . . . . .	44
Step 1: Initialization . . . . .	44
Step 2: Creating the children . . . . .	45
Step 3: Drawing the component instance . . . . .	45
Refining our sample components . . . . .	46
The component framework . . . . .	51
The UIComponent class . . . . .	51
Accessibility/Keyboard use . . . . .	51
Other features of the UIComponent . . . . .	52
The View class . . . . .	53
The ScrollView class . . . . .	54
An ActionScript template for new components . . . . .	54
Summary . . . . .	57
<b>Chapter 3 Exploring the UI Components</b> . . . . .	<b>59</b>
The Reusability Card . . . . .	60
Frequency (of use) . . . . .	61
Complexity . . . . .	62
Stability . . . . .	62
Maturity . . . . .	63
Popularity . . . . .	64
Multitier applications . . . . .	64
UI components provided with Flash . . . . .	65
Button components . . . . .	65
Button component . . . . .	66
CheckBox component . . . . .	66
RadioButton component . . . . .	67
Text components . . . . .	67
Label component . . . . .	68
TextInput component . . . . .	68
TextArea component . . . . .	69



- Cell-structured components . . . . . 69
  - List component . . . . . 70
  - ComboBox component . . . . . 70
  - DataGrid component . . . . . 71
  - Tree component . . . . . 72
- Container components . . . . . 72
  - ScrollPane component . . . . . 73
  - Loader component . . . . . 74
  - Window component . . . . . 74
  - Accordion component . . . . . 75
- Peculiar components . . . . . 76
  - Alert component . . . . . 76
  - DateChooser component . . . . . 77
  - DateField component . . . . . 78
  - Menu component . . . . . 78
  - MenuBar component . . . . . 79
  - NumericStepper component . . . . . 80
  - ProgressBar component . . . . . 80
  - UIScrollBar component . . . . . 81
- Using the UI components . . . . . 82
  - A first example of interaction . . . . . 82
- Typical structure of a component . . . . . 85
  - The actions layer . . . . . 87
  - The assets layer . . . . . 87
  - The bounding box layer . . . . . 88
- Summary . . . . . 88

**Chapter 4 Building Component-based Applications . . . . . 91**

- Screens . . . . . 92
  - Building an application using screens . . . . . 93
  - Content hierarchy in nested screens . . . . . 95
  - Reviewing the purpose of slides and forms . . . . . 96
    - Forms visibility . . . . . 98
    - Conclusion: should you use slides or forms? . . . . . 99
  - Screen hierarchies with external subtrees . . . . . 99
    - The complete path to an external screen . . . . . 101
- Creating a slide presentation dynamically . . . . . 102
  - Building the example . . . . . 103
    - Importing the Slide class . . . . . 108
    - Creating the screen hierarchy dynamically . . . . . 108
    - Adding navigation in the master screen . . . . . 109
    - Implementing the buttons-based navigation . . . . . 110
    - Using Loader components in the child screens . . . . . 111
    - Introducing the transitions . . . . . 111
    - Importing the transition classes . . . . . 112
    - Screen events and transition sequencing . . . . . 113
    - Working with forms . . . . . 114

Manager classes . . . . .	114
Managing depth . . . . .	115
MovieClip methods for handling depth . . . . .	115
A more flexible way of stacking objects . . . . .	116
Testing the DepthManager behavior . . . . .	118
Managing the keyboard focus . . . . .	120
Defining a focus schema . . . . .	121
Setting a default button . . . . .	123
Tab order in a browser . . . . .	123
Disabling the focus rect . . . . .	124
Managing windows . . . . .	124
A simple window-based system . . . . .	124
Creating a window instance . . . . .	126
Experimenting with modal windows . . . . .	127
Summary . . . . .	128
<b>PART TWO EXPLOITING THE ARCHITECTURE . . . . .</b>	<b>129</b>
<b>Chapter 5 Architecture-based Development . . . . .</b>	<b>131</b>
Exploiting the architecture . . . . .	132
Key benefits of a component architecture . . . . .	132
What is your job, really? . . . . .	133
Raising the bar . . . . .	134
Extend, expand, and alter . . . . .	134
Extending the architecture . . . . .	134
Expanding the architecture . . . . .	136
Altering the architecture . . . . .	137
From abstract ideas to a concrete example . . . . .	137
What is an XML layout engine? . . . . .	138
Benefits of an XML layout engine . . . . .	138
Further benefits in the Flash context . . . . .	140
XLEFF . . . . .	140
XLEFF main features . . . . .	144
Beyond generating user interfaces . . . . .	144
XLEFF internal architecture . . . . .	145
Summary . . . . .	146
<b>Chapter 6 XML for Defining User Interfaces . . . . .</b>	<b>149</b>
Basics of the XML data structure . . . . .	150
The Color Names section . . . . .	151
The Styles section . . . . .	151
Class styles . . . . .	152
Predefined styles . . . . .	153
Nested styles . . . . .	153
Custom styles . . . . .	154
The Stage section . . . . .	155

---

XLEFF sampler . . . . .	158
How to use it . . . . .	159
Playing with the sampler . . . . .	161
Examining a more complex user interface . . . . .	164
User interface patterns . . . . .	165
A first look into the substructures . . . . .	166
Using custom classes . . . . .	167
Events to be handled . . . . .	169
Summary . . . . .	169
<b>Chapter 7 Extending the Application Framework . . . . .</b>	<b>171</b>
Defining an FLA template . . . . .	172
Using scenes . . . . .	173
The Preloader scene . . . . .	174
The Dynamic Assets scene . . . . .	176
The Main scene . . . . .	178
Licensing issue . . . . .	179
Including the standard components source code . . . . .	179
Progressive update of the template . . . . .	181
Analyzing the size report . . . . .	181
Moving the symbols after the first frame . . . . .	182
Moving the classes after the first frame . . . . .	183
Defining a folder structure . . . . .	183
The role of classpath . . . . .	185
Facilitating event-driven programming . . . . .	186
The Main class . . . . .	187
A concrete example . . . . .	188
Skeleton of the Main class . . . . .	189
Handling the user interface events . . . . .	190
Event handler naming convention . . . . .	194
Managing content . . . . .	194
Pushing the separation paradigm further . . . . .	196
The role of CDATA . . . . .	197
Summary . . . . .	198
<b>Chapter 8 Making Your XML Life Easier . . . . .</b>	<b>201</b>
Parsing XML in ActionScript . . . . .	202
Object models and trees . . . . .	204
The typical job of an XML developer . . . . .	207
Simplifying the parsing process . . . . .	209
Parsing an XML document . . . . .	210
Document root and other nodes . . . . .	211
Identifying a node name . . . . .	211
Identifying a node type . . . . .	212
Accessing the attributes of a node . . . . .	214
Browsing the structure of an XML document . . . . .	215
A few notes on the use of XModel . . . . .	217
Summary . . . . .	217

<b>Chapter 9 The Customization Process</b> . . . . .	<b>219</b>
Working with styles . . . . .	220
Parameters controlled by styles . . . . .	221
The style lookup process . . . . .	221
Styles as properties of a component instance . . . . .	222
The styleName property . . . . .	223
Class-level styles . . . . .	225
Inheriting styles from a container . . . . .	226
Global styles . . . . .	227
Analyzing skins . . . . .	228
What is a skin? . . . . .	228
Handcrafted skins . . . . .	228
Mixed skins . . . . .	231
Purely coded skins . . . . .	234
Working with themes . . . . .	236
Changing skins and the mirage of code separation . . . . .	236
Changing skins at authoring time . . . . .	237
Changing skins programmatically . . . . .	240
Skins that reflect styles . . . . .	243
In search of a unified approach: subclassing . . . . .	245
An alternative to subclassing . . . . .	249
Summary . . . . .	249
<b>PART THREE CUSTOMIZING THE COMPONENTS</b> . . . . .	<b>251</b>
<b>Chapter 10 The Accordion Component</b> . . . . .	<b>253</b>
A minimal example . . . . .	254
Code-based version . . . . .	255
Codeless version . . . . .	256
XLEFF version . . . . .	257
The component structure . . . . .	257
Segment header . . . . .	258
Segment content area . . . . .	258
A richer example . . . . .	259
Codeless version . . . . .	259
Code-based version . . . . .	261
Supported styles . . . . .	262
Common styles . . . . .	263
Specific styles . . . . .	265
Skinnability . . . . .	265
The border . . . . .	265
The headers . . . . .	266
Solved mysteries . . . . .	268
Inheriting styles . . . . .	268
Creating header styles on a per-instance basis . . . . .	271
Reasons for subclassing . . . . .	273

---

<b>Chapter 11 The Button Component</b> . . . . .	<b>275</b>
Minimal example of the Button component . . . . .	276
A richer example . . . . .	277
Supported styles . . . . .	279
Common styles . . . . .	280
Specific styles . . . . .	281
Halo theme case . . . . .	281
Sample theme case . . . . .	282
Skinnability . . . . .	284
Replacing the purely coded skin . . . . .	285
The 32 skins of a button . . . . .	289
Implementing a toggle button . . . . .	289
Emphasizing a button instance . . . . .	289
Iconic buttons . . . . .	289
Solved mysteries . . . . .	290
A purely coded classic: the pill button . . . . .	290
Reasons for subclassing a Button component . . . . .	293
<b>Chapter 12 The CheckBox and RadioButton Components</b> . . . . .	<b>295</b>
Minimal example of the CheckBox and the RadioButton components . . . . .	296
XLEFF version . . . . .	297
Comparing the authoring parameters . . . . .	297
Supported styles . . . . .	298
Common styles . . . . .	298
Specific styles . . . . .	300
Skinnability . . . . .	302
Solved mysteries . . . . .	305
Where to find the RadioButtonGroup instance . . . . .	305
Reasons for subclassing the CheckBox and the RadioButton components . . . . .	307
<b>Chapter 13 The List, ComboBox, and DataGrid Components</b> . . . . .	<b>309</b>
Minimal example including the List, ComboBox, and DataGrid components . . . . .	310
XLEFF version . . . . .	313
Richer examples . . . . .	314
Itemization . . . . .	315
Custom labels . . . . .	317
Scrolling . . . . .	320
Sorting . . . . .	323
Selection management . . . . .	327
Making it editable . . . . .	331
Supported styles . . . . .	333
Common styles . . . . .	336
DataGrid-specific styles . . . . .	336
List-specific styles . . . . .	337
ComboBox-specific styles . . . . .	337

Skinnability . . . . .	338
Solved mysteries . . . . .	339
Cell rendering . . . . .	339
Building a custom cell renderer . . . . .	343
DataGrid column headers . . . . .	345
The undefined item bug . . . . .	346
Reasons for subclassing the List, ComboBox, and DataGrid components . . . . .	347
<b>Chapter 14 The DateChooser and DateField Components . . . . .</b>	<b>349</b>
Minimal example of the DateChooser and DateField components . . . . .	350
XLEFF version . . . . .	351
A richer example . . . . .	351
Code version . . . . .	353
How to retrieve and set a date . . . . .	353
Ranges definition . . . . .	355
The scroll event . . . . .	358
Supported styles . . . . .	360
Common styles . . . . .	360
Specific styles . . . . .	360
Skinnability . . . . .	363
Skinning the arrow buttons . . . . .	363
Skinning the DateField icon . . . . .	365
Solved mysteries . . . . .	366
Displaying the date in custom format . . . . .	367
A DateField bug . . . . .	367
Reasons for subclassing the DateChooser and DateField components . . . . .	369
<b>Chapter 15 The Loader, ScrollPane, and ProgressBar Components . . . . .</b>	<b>371</b>
Minimal examples . . . . .	372
A minimal example of the Loader component . . . . .	372
A minimal example of the ScrollPane component . . . . .	375
A minimal example of the ProgressBar component . . . . .	377
The ProgressBar's animated behavior . . . . .	377
The indeterminate ProgressBar . . . . .	379
XLEFF versions . . . . .	381
Combined examples . . . . .	381
The ProgressBar communication modes . . . . .	382
Codeless interaction . . . . .	382
ProgressBar and Loader interaction . . . . .	382
ProgressBar and ScrollPane interaction . . . . .	384
Mediated interaction . . . . .	385
Supported styles . . . . .	387
Skinnability . . . . .	389
Solved mysteries . . . . .	392
Reasons for subclassing . . . . .	393

<b>Chapter 16 The Menu and MenuBar Components</b> . . . . .	<b>395</b>
Minimal examples . . . . .	396
Minimal example of the Menu component . . . . .	396
Minimal example of the MenuBar component . . . . .	398
Richer examples . . . . .	399
Generating richer menus by coding . . . . .	399
Generating richer menus using XML . . . . .	405
XLEFF version . . . . .	408
Supported styles . . . . .	409
Stylizing the MenuBar (and its Menu instances) . . . . .	409
Common styles . . . . .	409
Specific styles . . . . .	410
Exploring the styles . . . . .	410
Skinnability . . . . .	414
Solved mysteries . . . . .	417
Further customization of a MenuBar skin . . . . .	417
Creating persistent Menu instances . . . . .	420
Reasons for subclassing the Menu and the MenuBar components . . . . .	424
<b>Chapter 17 The NumericStepper Component</b> . . . . .	<b>427</b>
Minimal example of the NumericStepper component . . . . .	428
XLEFF version . . . . .	429
Retrieving the value . . . . .	429
Minor bug for Flash MX 2004 users . . . . .	431
Styles supported by the NumericStepper component . . . . .	431
Skinning the NumericStepper component . . . . .	433
Solved mysteries . . . . .	435
Reasons for subclassing the NumericStepper component . . . . .	437
<b>Chapter 18 The TextArea, TextInput, and Label Components</b> . . . . .	<b>439</b>
Minimal example . . . . .	440
XLEFF version of the minimal example . . . . .	442
How the Label component resizes automatically . . . . .	442
The text field inside . . . . .	445
Richer example of the TextInput and TextArea components . . . . .	446
Handling the input process . . . . .	446
Supported styles . . . . .	448
Note on the skins . . . . .	450
Solved mysteries . . . . .	450
Hiding the background . . . . .	451
Handling the combination linefeed/CR . . . . .	453
Reasons for subclassing the Label, TextInput, and TextArea components . . . . .	456

<b>Chapter 19 The Tree Component</b> . . . . .	<b>459</b>
Minimal example of the Tree component . . . . .	460
XLEFF version of the minimal example . . . . .	464
Supported styles . . . . .	465
Color styles . . . . .	466
Text styles . . . . .	466
Animation styles . . . . .	466
Icon styles . . . . .	467
Other component-specific styles . . . . .	467
A note on skins . . . . .	467
Stylizing the minimal example . . . . .	468
Solved mysteries . . . . .	470
Taking full control . . . . .	471
Implementing isBranch and other XML attributes . . . . .	477
Reasons for subclassing the Tree component . . . . .	480
<b>Chapter 20 The Window and Alert Components</b> . . . . .	<b>483</b>
Minimal example of the Window and Alert components . . . . .	484
Dynamically creating windows . . . . .	487
Dynamically creating alerts . . . . .	489
Managing the content of a Window instance . . . . .	490
Supported styles . . . . .	495
Skinning the Window and Alert components . . . . .	499
Skin properties of the Window component . . . . .	499
Skin properties of the Alert component . . . . .	500
Adding skins to our previous stylized example . . . . .	500
Reasons for subclassing the Window and Alert components . . . . .	505
<b>Chapter 21 Handling the Scrollbars</b> . . . . .	<b>507</b>
Minimal example of the UIScrollBar component . . . . .	508
Customizing the scrollbars inside a component . . . . .	510
Step 1: Building a stylized version of the DataGrid component . . . . .	510
Step 2: Skinning the scrollbars . . . . .	512
Conclusion . . . . .	519
<b>PART FOUR APPENDIXES</b> . . . . .	<b>521</b>
<b>Appendix A Locating the Source Code of the Component Architecture</b> . . . . .	<b>523</b>
If you are a Windows user . . . . .	524
If you are a Mac user . . . . .	525
FLA source files . . . . .	525
Link them . . . . .	526



<b>Appendix B Transitions and Easing Classes</b> . . . . .	<b>529</b>
Parameters common to all of the transition types . . . . .	530
Easing classes . . . . .	531
Transition-specific parameters . . . . .	532
The Blinds transition . . . . .	532
The Fly transition . . . . .	532
The Iris transition . . . . .	532
The PixelDissolve transition . . . . .	532
The Rotate transition . . . . .	532
The Squeeze transition . . . . .	533
The Wipe transition . . . . .	533
Example of a transition parameters object . . . . .	533
<b>Index</b> . . . . .	<b>535</b>

## ABOUT THE AUTHOR

---



**Antonio De Donatis**, who graduated in Computer Science from Pisa University, has been designing and implementing object-oriented software since 1989, seems like a lifetime, when the first OOP developing tools appeared on the market. So far Antonio has managed, designed, and in several cases single-handedly implemented numerous projects for a variety of industries, ranging from media/communications to pharmaceuticals.

Antonio has worked for both large corporations and leading new media agencies, and currently trades under the name of Managed Source Limited, based in Surrey, England, where he has lived since 1998 after moving from Italy, his native country.

His commercial experience with Flash goes back to the fourth version of the software, released in 1999. In recent years, the object-oriented evolution of ActionScript has allowed him to reuse knowledge and techniques that he mastered when utilizing older programming languages such as C++ and Java.

Antonio is a specialist in the design of component-based architectures for the implementation of knowledge and content management systems and is now working on several projects, including the open source XML layout engine for Flash mentioned in this book, the latest version of which can be downloaded from [www.xleff.org](http://www.xleff.org).

Antonio considers programming a form of art and also enjoys chess, painting, and photography.

## ABOUT THE TECHNICAL REVIEWERS

---

**Sas Jacobs** is a web developer who loves working with Flash. She set up her business, Anything Is Possible, in 1994, working in the areas of web development, IT training, and technical writing. The business works with large and small clients building web applications with ASP.NET, Flash, XML, and databases. Sas has spoken at such conferences as Flash Forward, MXDU, and FlashKit on topics relating to XML and dynamic content in Flash. She is the author of the book *Foundation XML for Flash* (friends of ED, 2005) and contributed two chapters to *Object-Oriented ActionScript for Flash 8* (friends of ED, 2006).

After studying architecture for seven years, **Paul Barnes-Hoggett** changed his mind and decided to spend his time designing the “intergoogle.” He spent time as a lead developer at boxnewmedia, where he built award-winning sites for clients such as Select Model Management. (In his own words, he admits, “It’s a tough job looking at pictures of beautiful people all day, but someone has to do it.”) He set up Eyefodder in 2003, which specializes in building rich Internet applications for the media industry, and has built solutions for clients including FHM, Adidas, Air Miles, and ITV. When not pushing pixels, Paul likes to eat, drink, and be merry. To get in contact with him, visit [www.eyefodder.com](http://www.eyefodder.com).

## ABOUT THE COVER IMAGE DESIGNER

---

**Bruce Tang** is a freelance web designer, visual programmer, and author from Hong Kong. His main creative interest is generating stunning visual effects using Flash or Processing.

Bruce has been an avid Flash user since Flash 4, when he began using Flash to create games, websites, and other multimedia content. After several years of ActionScripting, he found himself increasingly drawn toward visual programming and computational art. He likes to integrate math and physics into his work, simulating 3D and other real-life experiences onscreen. His first Flash book was published in October 2005. Bruce's folio, featuring Flash and Processing pieces, can be found at [www.betaruce.com](http://www.betaruce.com), and his blog at [www.betaruce.com/blog](http://www.betaruce.com/blog).

The cover image uses a high-resolution Henon phase diagram generated by Bruce with Processing, which he feels is an ideal tool for such experiments. Henon is a strange attractor created by iterating through some equations to calculate the coordinates of millions of points. The points are then plotted with an assigned color.

$$x_{n+1} = x_n \cos(a) - (y_n - x_n^p) \sin(a)$$

$$y_{n+1} = x_n \sin(a) + (y_n - x_n^p) \cos(a)$$

