

# Foundation Game Design with Flash

Rex van der Spuy



# Foundation Game Design with Flash

Copyright © 2009 by Rex van der Spuy

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): 978-1-4302-1821-0

ISBN-13 (electronic): 978-1-4302-1822-7

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 2855 Telegraph Avenue, Suite 600, Berkeley, CA 94705. 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).

Apress and friends of ED books may be purchased in bulk for academic, corporate, or promotional use. eBook versions and licenses are also available for most titles. For more information, reference our Special Bulk Sales—eBook Licensing web page at <http://www.apress.com/info/bulksales>.

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**      **Production Editor**  
Ben Renow-Clarke      Laura Esterman

**Technical Reviewer**      **Composer/Artist**  
Josh Freeney      Diana Van Winkle

**Editorial Board**      **Proofreader**  
Clay Andres, Steve Anglin, Mark Beckner,      Linda Seifert  
Ewan Buckingham, Tony Campbell, Gary Cornell,  
Jonathan Gennick, Jonathan Hassell, Michelle Lowman,  
Matthew Moodie, Jeffrey Pepper, Frank Pohlmann,  
Ben Renow-Clarke, Dominic Shakeshaft,  
Matt Wade, Tom Welsh

**Project Manager**  
Beth Christmas

**Copy Editor**  
Nancy Sixsmith

**Associate Production Director**  
Kari Brooks-Copony

**Cover Image Designer**  
Corné van Dooren

**Interior and Cover Designer**  
Kurt Krames

**Manufacturing Director**  
Tom Debolski

# CONTENTS

---

<b>About the Author</b> .....	<b>xv</b>
<b>About the Technical Reviewer</b> .....	<b>xvi</b>
<b>About the Cover Image Designer</b> .....	<b>xvii</b>
<b>Introduction</b> .....	<b>xix</b>
<b>Chapter 1 Programming Foundations: How to Make a Video Game</b> ....	<b>3</b>
Basics you need to have .....	4
Things you need to know .....	4
And the things you don't need to know .....	4
It's all about programming .....	5
Programming? But I'm terrible at math! .....	6
I already know how to program! .....	6
What kind of games can I make? .....	7
Learning some new terms .....	7
Laying the foundation .....	7
Files you'll need .....	8
Setting up the work environment .....	8
Setting up the Flash Developer workspace .....	10
Setting up the ActionScript code format preferences .....	10
Writing your first program .....	11
I'll take that to go! .....	11
Don't skip class! .....	12
Using the constructor method .....	14
Aligning code .....	16
What's your directive? .....	17
Importing and extending the MovieClip class .....	18
Adding comments to your code .....	21
Publishing the SWF file .....	22
It didn't work? .....	24
Project Panel .....	26
A little more about AS3.0 and the Flash Player .....	27
Naming conventions .....	28
Summary .....	29

---

<b>Chapter 2 Making Objects</b> .....	<b>31</b>
Understanding Interactive Objects .....	32
Setting up the work environment .....	32
Creating the first page .....	35
Drawing the first page .....	38
Drawing the background .....	39
Organizing layers and the timeline .....	43
Drawing the foreground objects .....	45
Creating a hill .....	46
Making some water .....	49
Grouping objects .....	54
Adding some clouds .....	55
Creating some flowers .....	57
Learning a few more techniques .....	60
Creating a character .....	62
Adding some more pages .....	65
Using buttons .....	66
Creating a button symbol .....	67
Understanding button states .....	71
Creating the Over state .....	72
Creating the Down state .....	73
Duplicating the button .....	74
Organizing the Library .....	75
Adding the buttons to your scene .....	76
Summary .....	77
<b>Chapter 3 Programming Objects</b> .....	<b>79</b>
But I'm a bit scared of programming! .....	80
Displaying the first page of the storybook .....	80
How did that work? .....	82
Variables .....	83
Variable types .....	84
Creating empty boxes .....	85
Creating an instance .....	86
Displaying the instance on the stage .....	88
Programming buttons .....	89
Using dot notation .....	89
Invoking methods .....	91
Using method calls .....	91
Using function definitions .....	92
Creating method arguments and parameters .....	95
Using multiple arguments and parameters .....	97
Understanding events and event listeners .....	98
Importing an event class .....	99
Adding an event listener .....	99
Using the event handler .....	100
Understanding other events .....	102

Programming storybook buttons . . . . .	104
Looking at the onHillButtonClick event handler . . . . .	107
Using the onPondButtonClick event handler . . . . .	108
Adding back buttons . . . . .	108
Knowing when to use this model . . . . .	111
Summary . . . . .	112
<b>Chapter 4 Controlling Movie Clip Objects . . . . .</b>	<b>115</b>
Movie Clip properties . . . . .	116
Setting up the project files . . . . .	118
Going up and down . . . . .	118
Understanding x and y positions of objects . . . . .	123
Moving incrementally . . . . .	124
Tracing the output . . . . .	125
Using increment and decrement operators . . . . .	126
Limiting movement . . . . .	127
It's not a bug, it's a feature! . . . . .	129
Making it bigger and smaller . . . . .	133
Vanishing! . . . . .	138
Having a look . . . . .	142
More properties? . . . . .	148
Controlling Movie Clip timelines . . . . .	148
Using the timeline as a state machine . . . . .	161
Taking it further . . . . .	162
Summary . . . . .	163
<b>Chapter 5 Decision Making . . . . .</b>	<b>165</b>
Setting up the project files . . . . .	166
Designing a GUI . . . . .	168
Inputting and outputting . . . . .	169
Adding some text fields . . . . .	170
Creating dynamic text . . . . .	170
Adding input text . . . . .	172
A little more about fonts and text fields . . . . .	174
Adding a button . . . . .	176
Building a simple guessing game . . . . .	176
Setting up the Main.as file . . . . .	176
Learning more about variables . . . . .	178
Making it more obvious . . . . .	181
Making decisions . . . . .	184
Displaying the game status . . . . .	188
Using postfix operators to change variable values by 1 . . . . .	191
Tying up strings . . . . .	192
Hey, why use the gameStatus variable, anyway? . . . . .	195
Using uint vs. int variables . . . . .	196
Winning and losing . . . . .	197
Modular programming with methods . . . . .	202

---

Polishing up .....	203
Tackling random numbers .....	204
Disabling the Guess button .....	208
Playing again? .....	209
Seeing the final code .....	212
Taking it farther .....	215
Tracking guesses .....	215
Adding a visual display .....	215
Entering numbers with the Enter key .....	216
Turning the tables .....	218
Summary .....	219
<b>Chapter 6 Controlling a Player Character .....</b>	<b>221</b>
Setting up the project files .....	221
Controlling a player character with the keyboard .....	222
Controlling with the keyboard—the wrong way! .....	222
Creating a player character .....	223
Adding keyboard control code .....	226
Controlling the keyboard—the right way! .....	230
Moving with velocity .....	233
Using the new onKeyDown event handler .....	235
Using the onKeyUp event handler .....	235
Using the onEnterFrame event handler .....	237
Setting screen boundaries .....	238
Blocking movement at the stage edges .....	239
Building a better pigpen .....	241
Screen wrapping .....	245
Scrolling .....	246
Creating an environment .....	247
Fine-tuning the player character .....	249
Adding a drop shadow .....	249
Scrolling basics .....	251
Better scrolling .....	252
Even better scrolling .....	259
Taking it further .....	262
Parallax scrolling .....	262
Summary .....	263
<b>Chapter 7 Bumping into Things .....</b>	<b>265</b>
Setting up the project files .....	265
Ouch! .....	268
Using hitTestObject .....	268
Changing a dynamic text field .....	270
Triggering a change of state .....	271

Reducing a health meter . . . . .	273
Using scaleX to scale the meter based on a percentage . . . . .	278
Updating a score . . . . .	279
Picking up and dropping objects . . . . .	285
Learning the bad news about hitTestObject . . . . .	291
Detecting collisions with the bounding box . . . . .	292
Learning to live with it . . . . .	294
Creating subobjects . . . . .	295
Using hitTestPoint . . . . .	297
Using hitTestPoint to create an environmental boundary . . . . .	301
Creating objects that block movement . . . . .	305
Working with axis-based collision detection . . . . .	308
Programming with the Collision class . . . . .	309
Using static methods . . . . .	309
Using the method parameters . . . . .	310
Using the Collision.block method . . . . .	312
Pushing objects . . . . .	313
Taking a closer look at the Collision.block method . . . . .	313
Detecting bitmap collisions . . . . .	321
Summary . . . . .	321
<b>Chapter 8 Object-Oriented Game Design . . . . .</b>	<b>323</b>
Introducing object-oriented programming . . . . .	324
Binding classes to symbols . . . . .	324
Using properties and methods . . . . .	325
Private properties and methods . . . . .	326
Using an underscore character to highlight private properties . . . . .	327
Communicating between classes using getters and setters . . . . .	327
Using getters . . . . .	327
Using setters . . . . .	331
Getting started with the object-oriented approach . . . . .	332
Case study: Dungeon Maze Adventure . . . . .	333
Setting up the game . . . . .	334
Gathering project files and objects . . . . .	335
Entering the dungeon! . . . . .	336
Laying out the level . . . . .	337
The objects in the game . . . . .	338
Animating with the timeline . . . . .	340
Creating the enemies . . . . .	340
Animating the object . . . . .	342
EnemyTwo . . . . .	344
Controlling timeline animations with code . . . . .	345
Adding and removing objects from the stage . . . . .	347
ADDED_TO_STAGE event . . . . .	347
REMOVED_FROM_STAGE event . . . . .	349
How Dungeon Maze Adventure works . . . . .	351

DungeonOne_Manager class	352
Moving the player	356
Picking up the key	360
Opening the first door	361
Adding sound to the game	364
Creating the Sound and SoundChannel objects	365
Colliding with the enemies	367
Losing the game	368
Picking up the star weapon	370
Firing bullets	372
Using bullet objects	373
Bullets vs. enemy collisions	375
Player vs. wall collisions	378
Synchronizing ENTER_FRAME events	380
Winning the game	381
Modifying the game	382
Adding a new level	382
The problem of dependency	383
Creating a game manager	385
Firing bullets in four directions	386
Accessing the stage outside of the document class	391
Removing objects from the game	392
Summary	395

**Chapter 9 Platform Game: Physics and Data Management .....397**

Natural motion using physics	398
Setting up the project files	398
Acceleration	399
Player_Acceleration class	401
Friction	408
Bouncing	410
Gravity	411
Jumping	414
Stage boundaries and subobjects	417
Case study: Bug Catcher	419
Setting up the project files	420
Using the Player_Platform class	421
Constants	426
Player friction	426
Bounce variables	427
Player collision area	427
Adding Platforms	428
Beveling	429
Tinting	429
Detecting platform collisions	430
Using for loops	434
Looping through platforms	437

Finding the global x and y position of a subobject	439
Rotating toward an object	443
Rotating the frog's eyes toward the player object	446
Changing the stacking order	447
Adding some bugs to the code—literally!	450
Dynamic instance variables	452
Multiple objects sharing one event handler	453
Making the bugs move	454
Artificial intelligence	455
Using arrays	457
Pushing elements into an array	460
Looping arrays	461
Searching arrays	462
Collecting bugs	464
Winning and losing conditions	468
Complete Main_BugCatcher class	470
New Collision.playerAndPlatform utility	474
playerAndPlatform method	474
Platform bounce and friction	479
Detecting the top of the platform	481
Summary	482
<b>Chapter 10 Advanced Object and Character Control</b>	<b>485</b>
Dragging and dropping objects	486
Dragging and dropping the procedural way	486
Using target or currentTarget properties	490
Using dropTarget	491
Snapping the object to the target	492
Centering the drag object to the mouse	493
Confining the drag area	493
Dragging and dropping the object-oriented way	494
DragableObject class	495
Releasing the mouse outside the stage area	497
Main_DragAndDrop2 class	498
Easing	499
An alternative to inheritance: Composition	499
Moving objects with the mouse	502
Fixing an object to the mouse's position	503
Adding a dynamic filter	505
Moving an object with easing	508
Following the mouse with a bit of delay	511
Easing—advanced	513
Properties and methods of the Tween class	515
Easing package classes and methods	518
Tween events	520
Easing to random positions and calculating velocity	522
Calculating velocity	525
Implementing a chase feature	527

- Case study: Complex mouse-driven player control . . . . . 528
  - Player.as . . . . . 528
  - Moving the player . . . . . 531
  - Rotating the wand . . . . . 533
  - Firing bullets in 360 degrees . . . . . 533
  - Bullet.as . . . . . 536
    - Using a bevel filter . . . . . 540
    - Bullet factory: Using switch . . . . . 541
- Object factories . . . . . 543
  - Product classes . . . . . 544
  - Factory class . . . . . 545
  - Client class . . . . . 547
- Enemy AI systems . . . . . 548
  - Following the player . . . . . 548
    - The root property . . . . . 551
    - Moving the object . . . . . 551
  - Running away from the player . . . . . 552
  - Rotating and shooting toward the player . . . . . 553
    - Timer and TimerEvent classes . . . . . 555
    - Using a timer to fire bullets . . . . . 558
    - Shooting at random intervals . . . . . 558
- Using other player control systems . . . . . 559
- Dispatching events . . . . . 560
  - Event “bubbling” . . . . . 561
- Case study: Space Shooter . . . . . 563
  - Game structure . . . . . 564
  - Creating bullets . . . . . 565
  - Checking for bullet collisions with objects . . . . . 566
  - Removing bullets at the stage boundaries . . . . . 569
  - Classes and events . . . . . 570
- Summary . . . . . 570

**Index . . . . . 573**