

# Pro JavaScript™ Design Patterns



Ross Harmes and Dustin Diaz

## **Pro JavaScript™ Design Patterns**

**Copyright © 2008 by Ross Harmes and Dustin Diaz**

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): 978-1-59059-908-2

ISBN-10 (pbk): 1-59059-908-X

ISBN-13 (electronic): 978-1-4302-0495-4

ISBN-10 (electronic): 1-4302-0495-8

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.

Java and all Java-based marks are trademarks or registered trademarks of Sun Microsystems Inc. in the United States and other countries. Apress Inc. is not affiliated with Sun Microsystems Inc., and this book was written without endorsement from Sun Microsystems Inc.

Lead Editors: Chris Mills, Tom Welsh

Technical Reviewer: Simon Willison

Editorial Board: Steve Anglin, Ewan Buckingham, Tony Campbell, Gary Cornell, Jonathan Gennick, Jason Gilmore, Kevin Goff, Jonathan Hassell, Matthew Moodie, Joseph Ottinger, Jeffrey Pepper, Ben Renow-Clarke, Dominic Shakeshaft, Matt Wade, Tom Welsh

Project Manager: Richard Dal Porto

Copy Editor: Jennifer Whipple

Associate Production Director: Kari Brooks-Copony

Production Editor: Kelly Winkquist

Compositor and Artist: Kinetic Publishing Services, LLC

Proofreader: Dan Shaw

Indexer: Julie Grady

Cover Designer: Kurt Krames

Manufacturing Director: Tom Debolski

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 <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 <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 available to readers at <http://www.apress.com>.

*To Mom, and those who have listened, thanks*

—Dustin Diaz

*To Alec, Dymphi, and Terry*

—Ross Harmes



# Contents at a Glance

About the Authors .....	xv
About the Technical Reviewer .....	xvii
Acknowledgments .....	xix
Introduction .....	xxi

## PART 1 ■ ■ ■ Object-Oriented JavaScript

■ CHAPTER 1	Expressive JavaScript .....	3
■ CHAPTER 2	Interfaces .....	11
■ CHAPTER 3	Encapsulation and Information Hiding .....	25
■ CHAPTER 4	Inheritance .....	41
■ CHAPTER 5	The Singleton Pattern .....	65
■ CHAPTER 6	Chaining .....	83

## PART 2 ■ ■ ■ Design Patterns

■ CHAPTER 7	The Factory Pattern .....	93
■ CHAPTER 8	The Bridge Pattern .....	109
■ CHAPTER 9	The Composite Pattern .....	125
■ CHAPTER 10	The Facade Pattern .....	141
■ CHAPTER 11	The Adapter Pattern .....	149
■ CHAPTER 12	The Decorator Pattern .....	159
■ CHAPTER 13	The Flyweight Pattern .....	179
■ CHAPTER 14	The Proxy Pattern .....	197
■ CHAPTER 15	The Observer Pattern .....	215
■ CHAPTER 16	The Command Pattern .....	225
■ CHAPTER 17	The Chain of Responsibility Pattern .....	245
■ INDEX .....		263



# Contents

About the Authors .....	xv
About the Technical Reviewer .....	xvii
Acknowledgments .....	xix
Introduction .....	xxi

## PART 1 ■ ■ ■ Object-Oriented JavaScript

■ CHAPTER 1	<b>Expressive JavaScript</b> .....	3
	The Flexibility of JavaScript .....	3
	A Loosely Typed Language .....	6
	Functions As First-Class Objects .....	6
	The Mutability of Objects .....	8
	Inheritance .....	9
	Design Patterns in JavaScript .....	9
	Summary .....	10
■ CHAPTER 2	<b>Interfaces</b> .....	11
	What Is an Interface? .....	11
	Benefits of Using Interfaces .....	11
	Drawbacks of Using Interfaces .....	12
	How Other Object-Oriented Languages Handle Interfaces .....	12
	Emulating an Interface in JavaScript .....	14
	Describing Interfaces with Comments .....	14
	Emulating Interfaces with Attribute Checking .....	16
	Emulating Interfaces with Duck Typing .....	17
	The Interface Implementation for This Book .....	18
	The Interface Class .....	19
	When to Use the Interface Class .....	20
	How to Use the Interface Class .....	20
	Example: Using the Interface Class .....	21
	Patterns That Rely on the Interface .....	23
	Summary .....	23

<b>CHAPTER 3</b>	<b>Encapsulation and Information Hiding</b>	25
	The Information Hiding Principle	25
	Encapsulation vs. Information Hiding	26
	The Role of the Interface	26
	Basic Patterns	26
	Fully Exposed Object	27
	Private Methods Using a Naming Convention	30
	Scope, Nested Functions, and Closures	32
	Private Members Through Closures	33
	More Advanced Patterns	35
	Static Methods and Attributes	35
	Constants	37
	Singletons and Object Factories	38
	Benefits of Using Encapsulation	39
	Drawbacks to Using Encapsulation	39
	Summary	40
<b>CHAPTER 4</b>	<b>Inheritance</b>	41
	Why Do You Need Inheritance?	41
	Classical Inheritance	42
	The Prototype Chain	42
	The extend Function	43
	Prototypal Inheritance	45
	Asymmetrical Reading and Writing of Inherited Members	46
	The clone Function	48
	Comparing Classical and Prototypal Inheritance	49
	Inheritance and Encapsulation	49
	Mixin Classes	50
	Example: Edit-in-Place	52
	Using Classical Inheritance	52
	Using Prototypal Inheritance	55
	Using Mixin Classes	59
	When Should Inheritance Be Used?	62
	Summary	63
<b>CHAPTER 5</b>	<b>The Singleton Pattern</b>	65
	The Basic Structure of the Singleton	65
	Namespacing	66



A Singleton As a Wrapper for Page-Specific Code.....	68
A Singleton with Private Members.....	70
Using the Underscore Notation.....	70
Using Closures.....	71
Comparing the Two Techniques.....	74
Lazy Instantiation.....	75
Branching.....	78
Example: Creating XHR Objects with Branching.....	79
When Should the Singleton Pattern Be Used?.....	81
Benefits of the Singleton Pattern.....	81
Drawbacks of the Singleton Pattern.....	82
Summary.....	82

■ <b>CHAPTER 6</b>	<b>Chaining.....</b>	<b>83</b>
	The Structure of a Chain.....	84
	Building a Chainable JavaScript Library.....	86
	Using Callbacks to Retrieve Data from Chained Methods.....	89
	Summary.....	90

## PART 2 ■ ■ ■ Design Patterns

■ <b>CHAPTER 7</b>	<b>The Factory Pattern.....</b>	<b>93</b>
	The Simple Factory.....	93
	The Factory Pattern.....	96
	When Should the Factory Pattern Be Used?.....	99
	Dynamic Implementations.....	99
	Combining Setup Costs.....	99
	Abstracting Many Small Objects into One Large Object.....	99
	Example: XHR Factory.....	99
	Specialized Connection Objects.....	101
	Choosing Connection Objects at Run-Time.....	103
	Example: RSS Reader.....	104
	Benefits of the Factory Pattern.....	107
	Drawbacks of the Factory Pattern.....	108
	Summary.....	108

<b>CHAPTER 8</b>	<b>The Bridge Pattern</b>	109
	Example: Event Listeners	109
	Other Examples of Bridges	110
	Bridging Multiple Classes Together	111
	Example: Building an XHR Connection Queue	111
	Including the Core Utilities	112
	Including an Observer System	114
	Developing the Queue Skeleton	114
	Implementing the Queue	116
	Where Have Bridges Been Used?	122
	When Should the Bridge Pattern Be Used?	122
	Benefits of the Bridge Pattern	123
	Drawbacks of the Bridge Pattern	123
	Summary	123
<b>CHAPTER 9</b>	<b>The Composite Pattern</b>	125
	The Structure of the Composite	126
	Using the Composite Pattern	126
	Example: Form Validation	127
	Putting It All Together	133
	Adding Operations to FormItem	133
	Adding Classes to the Hierarchy	133
	Adding More Operations	136
	Example: Image Gallery	136
	Benefits of the Composite Pattern	139
	Drawbacks of the Composite Pattern	139
	Summary	140
<b>CHAPTER 10</b>	<b>The Facade Pattern</b>	141
	Some Facade Functions You Probably Already Know About	141
	JavaScript Libraries As Facades	142
	Facades As Convenient Methods	143
	Example: Setting Styles on HTML Elements	144
	Example: Creating an Event Utility	146
	General Steps for Implementing the Facade Pattern	147
	When Should the Facade Pattern Be Used?	148
	Benefits of the Facade Pattern	148
	Drawbacks of the Facade Pattern	148
	Summary	148

<b>CHAPTER 11</b>	<b>The Adapter Pattern</b>	149
	Characteristics of an Adapter	149
	Adapting Existing Implementations	150
	Example: Adapting One Library to Another	150
	Example: Adapting an Email API	152
	Wrapping the Webmail API in an Adapter	157
	Migrating from fooMail to dedMail	157
	When Should the Adapter Pattern Be Used?	158
	Benefits of the Adapter Pattern	158
	Drawbacks of the Adapter Pattern	158
	Summary	158
<b>CHAPTER 12</b>	<b>The Decorator Pattern</b>	159
	The Structure of the Decorator	159
	The Role of the Interface in the Decorator Pattern	163
	The Decorator Pattern vs. the Composite Pattern	163
	In What Ways Can a Decorator Modify Its Component?	164
	Adding Behavior After a Method	164
	Adding Behavior Before a Method	165
	Replacing a Method	166
	Adding New Methods	167
	The Role of the Factory	169
	Function Decorators	172
	When Should the Decorator Pattern Be Used?	173
	Example: Method Profiler	173
	Benefits of the Decorator Pattern	176
	Drawbacks of the Decorator Pattern	176
	Summary	177
<b>CHAPTER 13</b>	<b>The Flyweight Pattern</b>	179
	The Structure of the Flyweight	179
	Example: Car Registrations	179
	Intrinsic and Extrinsic State	180
	Instantiation Using a Factory	181
	Extrinsic State Encapsulated in a Manager	182
	Managing Extrinsic State	183
	Example: Web Calendar	183
	Converting the Day Objects to Flyweights	185
	Where Do You Store the Extrinsic Data?	186

Example: Tooltip Objects .....	186
The Unoptimized Tooltip Class .....	187
Tooltip As a Flyweight .....	188
Storing Instances for Later Reuse .....	190
When Should the Flyweight Pattern Be Used? .....	192
General Steps for Implementing the Flyweight Pattern .....	193
Benefits of the Flyweight Pattern .....	193
Drawbacks of the Flyweight Pattern .....	194
Summary .....	194
 <b>CHAPTER 14 The Proxy Pattern .....</b>	<b>197</b>
The Structure of the Proxy .....	197
How Does the Proxy Control Access to Its Real Subject? .....	197
Virtual Proxy, Remote Proxy, and Protection Proxy .....	200
The Proxy Pattern vs. the Decorator Pattern .....	201
When Should the Proxy Be Used? .....	201
Example: Page Statistics .....	201
General Pattern for Wrapping a Web Service .....	205
Example: Directory Lookup .....	206
General Pattern for Creating a Virtual Proxy .....	210
Benefits of the Proxy Pattern .....	213
Drawbacks of the Proxy Pattern .....	213
Summary .....	214
 <b>CHAPTER 15 The Observer Pattern .....</b>	<b>215</b>
Example: Newspaper Delivery .....	215
Push vs. Pull .....	216
Pattern in Practice .....	216
Building an Observer API .....	218
Delivery Method .....	219
Subscribe .....	219
Unsubscribe .....	220
Observers in Real Life .....	220
Example: Animation .....	221
Event Listeners Are Also Observers .....	222
When Should the Observer Pattern Be Used? .....	223
Benefits of the Observer Pattern .....	223
Drawbacks of the Observer Pattern .....	223
Summary .....	223

<b>■ CHAPTER 16</b>	<b>The Command Pattern</b>	225
	The Structure of the Command	225
	Creating Commands with Closures	227
	The Client, the Invoker, and the Receiver	227
	Using Interfaces with the Command Pattern	228
	Types of Command Objects	228
	Example: Menu Items	230
	The Menu Composites	231
	The Command Class	233
	Putting It All Together	234
	Adding More Menu Items Later On	235
	Example: Undo and Logging	235
	Implementing Undo with Nonreversible Actions By Logging	
	Commands	239
	Logging Commands for Crash Recovery	242
	When to Use the Command Pattern	242
	Benefits of the Command Pattern	243
	Drawbacks of the Command Pattern	243
	Summary	244
<b>■ CHAPTER 17</b>	<b>The Chain of Responsibility Pattern</b>	245
	The Structure of the Chain of Responsibility	245
	Passing on Requests	251
	Implementing a Chain of Responsibility in an Existing Hierarchy	254
	Event Delegation	255
	When Should the Chain of Responsibility Pattern Be Used?	255
	Example: Image Gallery Revisited	256
	Using the Chain of Responsibility to Make Composites	
	More Efficient	257
	Adding Tags to Photos	258
	Benefits of the Chain of Responsibility Pattern	261
	Drawbacks of the Chain of Responsibility Pattern	262
	Summary	262
<b>■ INDEX</b>		263