

# RESPONSIVE WEB DESIGN

with **Adobe Photoshop**

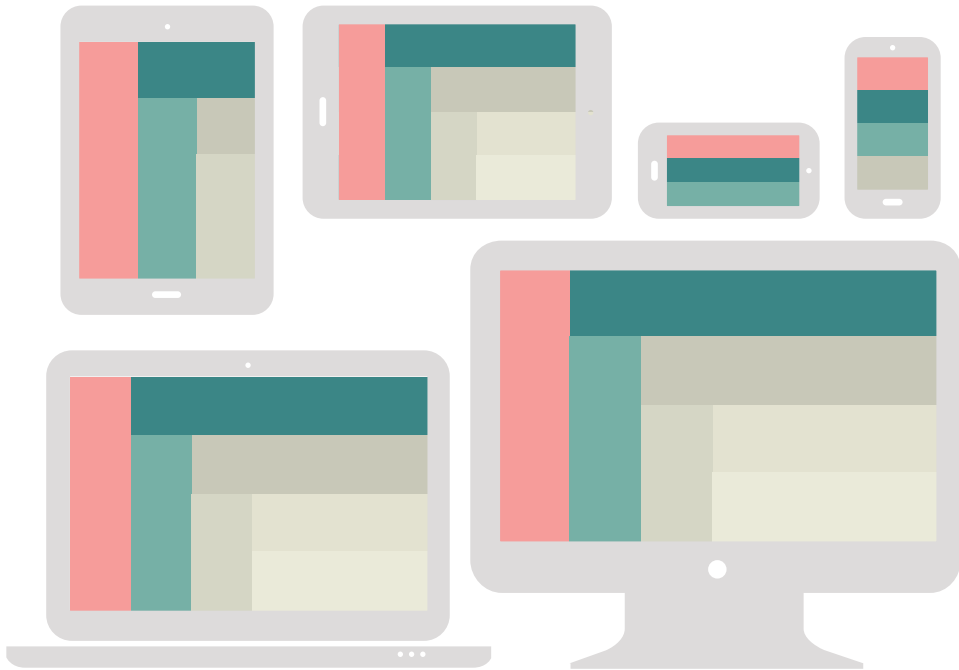


DAN ROSE



# RESPONSIVE WEB DESIGN

with **Adobe Photoshop**



**DAN ROSE**



# Responsive Web Design with Adobe Photoshop

**Dan Rose**

Copyright © 2015 Dan Rose

Adobe Press books are published by Peachpit, a division of Pearson Education.

For the latest on Adobe Press books, go to [www.adobepress.com](http://www.adobepress.com). To report errors, please send a note to [errata@peachpit.com](mailto:errata@peachpit.com).

**Adobe Press Editor:** Victor Gavenda

**Development Editor:** Stephen Nathans-Kelly

**Production Editor:** Maureen Forys, Happenstance Type-O-Rama

**Compositor:** Cody Gates, Happenstance Type-O-Rama

**Technical Editors:** Dennis Kardys and Joel Baer

**Copyeditor:** Kim Wimpsett

**Proofreader:** Kim Wimpsett

**Indexer:** Jack Lewis

**Cover Design:** Aren Straiger

**Interior Design:** Maureen Forys, Happenstance Type-O-Rama

## Notice of Rights

All rights reserved. No part of this book may be reproduced or transmitted in any form by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of the publisher. For information on getting permission for reprints and excerpts, contact [permissions@peachpit.com](mailto:permissions@peachpit.com).

## Notice of Liability

The information in this book is distributed on an “As Is” basis, without warranty. While every precaution has been taken in the preparation of the book, neither the author, Adobe Systems Incorporated, nor the publisher 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 instructions contained in this book or by the computer software and hardware products described in it.

## Trademarks

Adobe, Photoshop, Fireworks, Dreamweaver, InDesign, Illustrator, Creative Cloud, Generator, Typekit, and Flash are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries. All other trademarks are the property of their respective owners.

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and Peachpit was aware of a trademark claim, the designations appear as requested by the owner of the trademark. All other product names and services identified throughout this book are used in editorial fashion only and for the benefit of such companies with no intention of infringement of the trademark. No such use, or the use of any trade name, is intended to convey endorsement or other affiliation with this book.

Printed and bound in the United States of America

ISBN-13: 978-0-134-03563-5

ISBN-10: 0-134-03563-1

9 8 7 6 5 4 3 2 1

# Acknowledgments

This book was guided by the resourcefulness of Victor Gavenda, the eagle eye of Stephen Nathans-Kelly (and the Adobe Press team), and the sage wisdom of Dennis Kardys. My gratitude to Dan Mall for trailblazing the Photoshop-for-RWD path. My love and dedication to Amanda for her support and encouragement, to Holly and Norah for play and snuggle breaks, and to Jesus, with whom all things are indeed possible.

*This page intentionally left blank*

# Contents

<b>1</b>	<b>Photoshop's New Groove</b>	<b>1</b>
	Called Into Question	2
	Stick in the Mud	3
	<i>Fear of the Unknown?</i>	4
	<i>Can I Still Get by Without Knowing Code?</i>	4
	More Process Than Tool	6
	A Battle of Two Short Words	7
	Not on the Menu Tonight	8
	<i>The Core Tenets of Responsive Web Design</i>	8
	<i>Responsive Patterns</i>	8
	<i>Performance</i>	9
	<i>Photoshop Basics</i>	9
	<i>The Minutiae of Version Disparity</i>	9
	<i>The Merits of Comparable Tools</i>	9
	Finding Photoshop's Groove	10
	We Need to Make This Responsive!	11
<b>2</b>	<b>How Did We Get Here?</b>	<b>13</b>
	How We Used to Know Photoshop	14
	The Faults of Traditional Photoshop	15
	<i>On Full-Page Comps</i>	15
	Pain Point du Jour	18
	<i>Fixed-Width Comps</i>	18
	<i>Lack of Interactivity</i>	22
	<i>Some Fonts Are Better Than No Fonts?</i>	23
	<i>The Big Reveal</i>	25
	<i>What Did You Expect?</i>	26
	<i>Presentation Woes</i>	26
	<i>Bound by Approval</i>	28

<i>Not So Stable</i> . . . . .	29
<i>Less-Than-Seamless Exporting</i> . . . . .	29
<i>Empty Your Pockets</i> . . . . .	30
<i>Double the Effort, Double the Pain</i> . . . . .	31
If Not Photoshop, What? . . . . .	31
<b>3 The Case for Designing in the Browser</b> . . . . .	<b>33</b>
You Get a Tool! And You Get a Tool! Everyone Gets a Tool! . . . . .	33
Designing in the Browser 101 . . . . .	34
<i>Text Editor and Live Preview</i> . . . . .	35
<i>Inspect Element</i> . . . . .	37
Fluid by Nature: The Inherent Benefits of the Browser . . . . .	37
<i>Interactivity</i> . . . . .	38
<i>Global Changes</i> . . . . .	39
<i>Free</i> . . . . .	40
<i>1x the Effort</i> . . . . .	40
Web Design's Natural Habitat . . . . .	42
<i>Public Testing</i> . . . . .	43
<i>PSDs for Proofreading, Browser for Evaluating Behavior</i> . . . . .	43
<i>Reaffirming Expectations That Things Look Different in</i> <i>Different Browsers</i> . . . . .	44
<i>Easy to Change on the Fly</i> . . . . .	45
<i>Assessment as a Client Education Tool</i> . . . . .	45
<i>Fold</i> . . . . .	46
Designer/Developer Bonding . . . . .	47
OK to Kill Photoshop Now? . . . . .	48
<b>4 A Plea for Photoshop–Browser Harmony</b> . . . . .	<b>49</b>
Photoshop Is the New Vinyl . . . . .	49
The Power of Manipulation . . . . .	51
Creative Mode vs. Correct Mode . . . . .	52
The Path of Least Resistance . . . . .	52

---

Responsive Design Sameness . . . . .	56
Using Photoshop Only When Necessary . . . . .	58
<i>The Megaman Principle</i> . . . . .	59
<i>Practical Photoshopping: An Overview</i> . . . . .	61
<b>5 Vetting Direction . . . . .</b>	<b>63</b>
The Contrast Conundrum . . . . .	64
<i>The Comp Approach</i> . . . . .	64
<i>Within the Realm of Possibility</i> . . . . .	65
Including Your Stakeholders in the Design Process . . . . .	66
Moodboards . . . . .	67
<i>Methods of Moodboarding</i> . . . . .	68
<i>Finding and Storing Inspiration</i> . . . . .	70
Visual Inventories . . . . .	72
<i>The Pursuit of Efficiency</i> . . . . .	75
<i>Conversations, Not Deliverables</i> . . . . .	77
<i>Experimenting with Style</i> . . . . .	78
<b>6 Establishing Style . . . . .</b>	<b>79</b>
Suitable Mock-up Replacements . . . . .	79
<i>On Sketching</i> . . . . .	80
<i>Style Tiles</i> . . . . .	81
<i>Style Prototypes</i> . . . . .	84
Component Inventory . . . . .	87
Element Collages . . . . .	90
<i>Stripping Out the Abstraction</i> . . . . .	91
<i>Crafting an Element Collage</i> . . . . .	91
<i>Covering a Lot of Ground Quickly</i> . . . . .	94
<i>Do Not Make It Look Like a Website</i> . . . . .	96
<i>Color Comparisons</i> . . . . .	99
<i>Scope Creep</i> . . . . .	100
<i>Asking the Right Questions</i> . . . . .	101
<i>Do Make It Look Like a Website</i> . . . . .	103



	<i>Point of Reference</i> . . . . .	103
	<i>I Still Can't See It</i> . . . . .	104
	What's Missing . . . . .	105
<b>7</b>	<b>Establishing the System</b> . . . . .	<b>107</b>
	Now It's the Browser's Turn . . . . .	108
	Defining the Style Guide . . . . .	108
	<i>Web-Specific</i> . . . . .	109
	<i>Why the Style Guide Should Live in the Browser</i> . . . . .	114
	Building the Component Library . . . . .	114
	<i>Contents of a Comprehensive Component Library</i> . . . . .	116
	<i>Choosing the Best Environment for Your Components</i> . . . . .	124
	Prototyping . . . . .	126
	<i>Roughing It in Low-Fidelity</i> . . . . .	126
	<i>High-Fidelity and Beyond!</i> . . . . .	129
<b>8</b>	<b>Getting Back into Photoshop with Page Layers</b> . . . . .	<b>131</b>
	Rough Waters Ahead . . . . .	132
	Introducing Page Layers . . . . .	133
	The Struggle to Increase Fidelity . . . . .	134
	<i>Don't Get Too Comfortable in Photoshop</i> . . . . .	135
	<i>Leveraging Linked Smart Objects</i> . . . . .	136
	There's No Easy Way to Suggest Tweaks . . . . .	137
	<i>The Old Screenshot</i> . . . . .	137
	<i>The New Screenshot</i> . . . . .	138
	Our Pages Lack Cohesion . . . . .	140
	<i>Framing Content and the Big Picture</i> . . . . .	140
	<i>Where Skeuomorphism Worked</i> . . . . .	142
	Some Elements Suffer from Responsive Wonkiness . . . . .	143
	<i>Width-Specificity in Page Layers</i> . . . . .	145
	Exit Strategy . . . . .	146

---

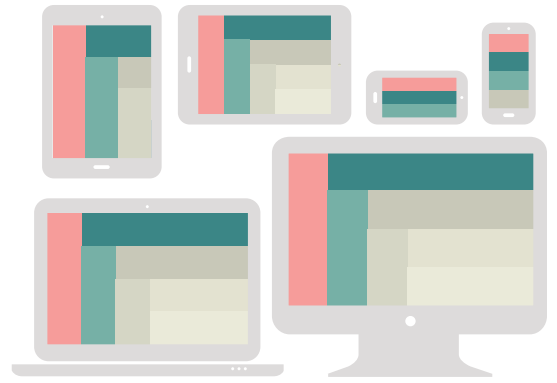
<b>9</b>	<b>Extracting Your Way Out of Photoshop</b>	<b>147</b>
	Asset Extraction Is Like Pulling Teeth	147
	<i>Crop and Save</i>	148
	<i>Copy Merged</i>	150
	<i>Save for Web</i>	151
	Adobe Generator	152
	<i>Auto-magic Generation</i>	153
	<i>Pixel Precision</i>	153
	<i>Speaking Fluent Generator</i>	154
	<i>Layer Naming as a Practice</i>	157
	Extract Assets	157
	<i>Setup</i>	158
	Extract	160
	<i>Setup</i>	161
	<i>Downloading Assets via Libraries</i>	162
	<i>Extracting Values</i>	162
	<i>Generating CSS</i>	163
<b>10</b>	<b>Extending Photoshop</b>	<b>167</b>
	Building the “You” Version of Photoshop	167
	Artwork	169
	<i>Subtle Patterns</i>	169
	<i>Random User Generator</i>	171
	<i>Social Kit</i>	172
	<i>Pictura</i>	172
	<i>Transform Each</i>	173
	<i>DevRocket</i>	174
	<i>Bjango Actions</i>	175
	<i>WebZap</i>	176
	<i>Composer</i>	177
	<i>Layout Wrapper</i>	177
	<i>RotateMe</i>	178

Color . . . . .	179
<i>Oto255</i> . . . . .	179
<i>Adobe Color (formerly Kuler)</i> . . . . .	180
<i>Adobe Color CC for iOS</i> . . . . .	181
<i>Coolorus</i> . . . . .	182
Assets . . . . .	183
<i>iOS Hat</i> . . . . .	183
<i>OtherIcons</i> . . . . .	184
<i>Glifo</i> . . . . .	185
<i>FlatIcon</i> . . . . .	186
<i>TinyPNG</i> . . . . .	186
<i>ImageOptim</i> . . . . .	187
Prototyping . . . . .	188
<i>Framer, Composite, and Stand In</i> . . . . .	188
<i>InVision</i> . . . . .	190
Organization . . . . .	191
<i>GuideGuide</i> . . . . .	191
<i>Renamy</i> . . . . .	192
<i>Ink192</i>	
<i>psdiff</i> . . . . .	193
Miscellaneous Photoshoppery . . . . .	194
<i>ShortcutFoo</i> . . . . .	194
<i>Photoshop Secrets</i> . . . . .	195
<b>11 Remembering Etiquette . . . . .</b>	<b>197</b>
The Problem with Inheriting PSDs . . . . .	197
What Is Photoshop Etiquette? . . . . .	199
<i>Improves Efficiency</i> . . . . .	199
<i>Keeps You Organized</i> . . . . .	199
<i>Creates Conventions</i> . . . . .	200
<i>Increased Importance in an RWD Workflow</i> . . . . .	200
Files . . . . .	201
<i>Name Files Appropriately</i> . . . . .	201

Store Assets Relative to PSD . . . . .	202
File Accessibility . . . . .	202
Layers . . . . .	203
Name Layers and Be Accurate . . . . .	203
Use Groups and Globalize Where Possible. . . . .	204
Delete Unnecessary Layers . . . . .	205
Images. . . . .	205
Be Nondestructive. . . . .	206
Use Blend Modes with Care . . . . .	207
Be Aware of Resolution and Density. . . . .	207
Type. . . . .	207
Standardize Font Access. . . . .	207
Don't Stretch Type. . . . .	208
Control Your Text Boxes and Separate Them . . . . .	209
Effects . . . . .	209
Use Overlays Appropriately. . . . .	210
Nail Tileable Images . . . . .	210
Be Deliberate. . . . .	210
QA . . . . .	211
Proofread. . . . .	211
Account for All Assets. . . . .	212
Be Familiar with Browser Compatibility . . . . .	212
<b>12 Adopting a Completely New Workflow . . . . .</b>	<b>215</b>
Looking Back at Moving Forward . . . . .	215
Full-Page Photoshop Comps Are Disharmonious with RWD. . . . .	215
Designing in the Browser Helps, But Not As Much As We'd Like. . . . .	216
2 Cups Browser, 1 Cup Photoshop . . . . .	216
Vetting Direction Efficiently Is Critical . . . . .	216
Style Can Be Established Through Small Exercises . . . . .	216
Page-Building Is Easier with Component-Based Systems . . . . .	218

<i>Page Layers Makes Going from HTML to Photoshop Simple</i> . . . . .	218
<i>New Extraction Tools Get Us Back to the Browser Quicker.</i> . . . . .	219
<i>We Can Customize Photoshop for RWD with Useful Third-Party Extensions</i> . . . . .	219
<i>A Little Etiquette Goes a Long Way</i> . . . . .	219
<b>On Adoption</b> . . . . .	219
<i>Strategies for Getting Buy-in Internally.</i> . . . . .	220
<i>Strategies for External Getting Buy-In</i> . . . . .	224
<i>What Happens When Things Go Wrong</i> . . . . .	226
<b>Adjusting Your Perspective on Tools.</b> . . . . .	227
<i>Repurposing Tools May Be Better Than Getting New Ones</i> . . . . .	228
<b>Index</b> . . . . .	229

# 4



## A PLEA FOR PHOTOSHOP-BROWSER HARMONY

In Chapter 3, I presented the case for a browser-based approach to web design. In this chapter, I'll take the browser approach down a peg, but I don't want to diminish its importance in a responsive workflow. The browser is still the centerpiece of what we're doing. But we need to work Photoshop back in to the fold—and not just because we're Photoshop fanboys and fangirls. If it didn't make sense to include Photoshop, I wouldn't waste your time advocating its importance in the responsive web design (RWD) workflow.

The good news is that Photoshop, more so than any other tool I can think of, fills an important role missing in a browser-only design process.

### Photoshop Is the New Vinyl

Have you ever had someone send you a link to a YouTube video and, before you know it, you start going down a never-ending hole watching related video after related video? That happened to me recently, and it started with the popular show “How It's Made.” My binge-watching led me to a fascinating piece on vinyl records. (If you just had to Google *vinyl records*, you're probably not the only one.)

The history of vinyl records is arguably more interesting than their manufacturing. Many of us remember their popularity in the 1970s and 1980s. Even when the cassette tape began to dominate music sales in the mid-1980s, records were still holding their own. It wasn't until the compact disc (CD) became the preferred music medium in the 1990s that vinyl records really fell out of favor. In fact, CDs were so superior to records for most music consumers' purposes that they pushed vinyl to the brink of extinction.

CDs offered a smaller, portable format that could fit significantly more music than records. They were even more durable, depending on how you stored them. It's no shock that CDs were became more popular than vinyl. When the iPod ushered in the digital music revolution of the mid-2000s and began to marginalize disc media, you would assume it made vinyl go away altogether, right?

Wrong. An incredibly strange thing happened: In 2006, people started buying more records. Today, vinyl growth is steady, and while it's by no means equal to digital in sales and mainstream consumer use, it remains relevant. Amazingly, market analysts have assigned no single concrete reason for vinyl's uptick in popularity (see **Figure 4.1**).



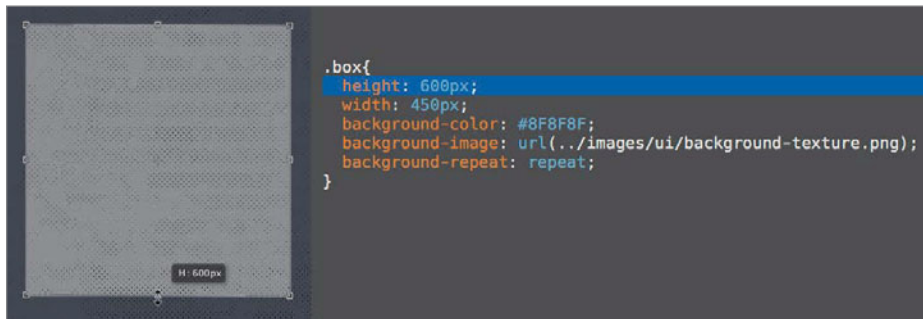
**Figure 4.1** Vinyl has withstood some major innovations in music media, staying relevant even to the current day. Can Photoshop do the same in the face of new apps such as Sketch and workflow alternatives such as designing in the browser?

So, who or what gets the credit for sustaining the life of the vinyl record? Two main audiences: audiophiles, who prefer vinyl's sound quality and treasure their massive record collections, and disc jockeys (DJs). DJs have mastered the art of combining the digital and the analog, so to speak. Digital tracks play off a computer while the DJ spins a record on a turntable for scratching, looping, and adding a host of physical effects.

Curious, I sought after some rationale from a friend of mine who is a popular Boston-area DJ. Why does he prefer to use vinyl in his work? It's the same thing we Photoshop users enjoy but sometimes have trouble putting words to: direct manipulation.

## The Power of Manipulation

At last, we find Photoshop’s first legitimate way back into a respectable design process. It’s unrealistic we’d completely abandon the visual techniques we’ve mastered for years in favor of code-abstracted design. One of the major reasons we tend to feel more “comfortable” in Photoshop is our ability to stretch, squish, shrink, and reposition elements. Sure, it’s possible to do those things with CSS somewhat easily. However, those manipulations sit behind a wall of abstraction, adjusting values and refreshing to see the results (see **Figure 4.2**).



**Figure 4.2** The Free Transform tool, the Move tool, and a host of others offer an ease of use that enables the exploration we desperately need—and won’t readily find in a coding-only design environment.

Designing in the browser can’t compete with the benefits of direct manipulation. Don’t underestimate the value of drawing a shape in Photoshop and seeing every step in its implementation. Your brain is processing all the sizes, colors, and positions, looking for the one that feels right.

“I move shapes around until they make sense.”

—JARED ERONDU (<https://twitter.com/erond/status/465981370930450432>)

Sure, there are some disadvantages to the freedom direct manipulation provides. It engenders the “silo” effect, where we designers go solo and play in Photoshop for extended periods of time, discouraging collaboration. Direct manipulation is neither systematic nor object-oriented, meaning the process by which we transform type and shape often doesn’t adhere to a pattern or set of preset constraints.

But exploring in Photoshop sure does help get the creative juices flowing.



## Creative Mode vs. Correct Mode

I am usually superior creatively in Photoshop but better at being correct in HTML & CSS. I have a propensity to try outlandish ideas on the Photoshop canvas, knowing my actions have few repercussions, whereas when I'm coding, I'm sometimes fearful of not being able to revert to a point when everything was "working." Most designers I know operate this way, although a rare few are just as creative in code as they are in Photoshop.

As responsive web designers, we need to be "correct" in our execution of code, but we also need to be able to vet our ideas adequately. I'm just not sure we can do both in a single environment.

When I first set out to design in the browser, I assumed that because it had similar tools to Photoshop, I'd be able to explore creative ideas in the same way. I could not have been more wrong. Not only were my designs looking less distinctive and considered, but the amount of time I was spending on them increased significantly. I'd sit for hours in CSS just trying one thing, reverting, trying another approach, and never really nailing it.

This is where direct manipulation makes all the difference. Code abstracts design by a layer of syntax. Instead of choosing position, size, and color by dragging or stretching, we use letters and numbers to assign a value. By no means is the latter approach objectively wrong, but the former feels right for many of us. If you come from a print background, you most likely agree that this abstraction hinders your ability to ideate.

The more I've thought about this abstraction, the more I've been able to attribute it to one significant concern: following the path of least resistance.

## The Path of Least Resistance

Even though we have the tools to pull off Photoshop-like effects in CSS, I don't think we always approach using them the same way.

Check out the audience navigation in **Figure 4.3** (on the next page). This low-fidelity prototype was built in HTML & CSS, and now it's ready for styling. My first impulse was to apply the button styles I was already toying with elsewhere and use them accordingly.

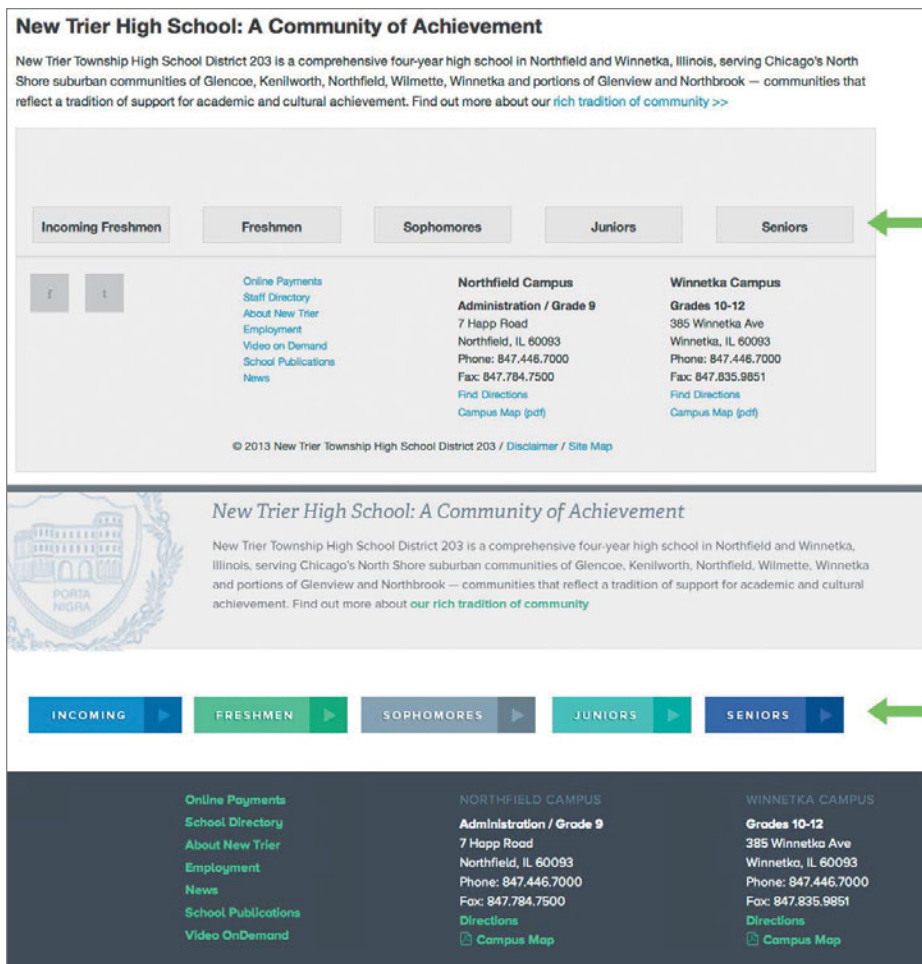


Figure 4.3 A styling-ready, low-fidelity prototype built in HTML & CSS

Objectively, there's nothing wrong with this approach, and certainly it communicates that these are five areas you can visit. When I showed the client, they were quite underwhelmed. The feedback I received was that the section didn't have adequate emphasis relative to the importance of the content.

My reaction was to fill the containing divs with the button color (see **Figure 4.4** on the next page). Bigger = more important, right?

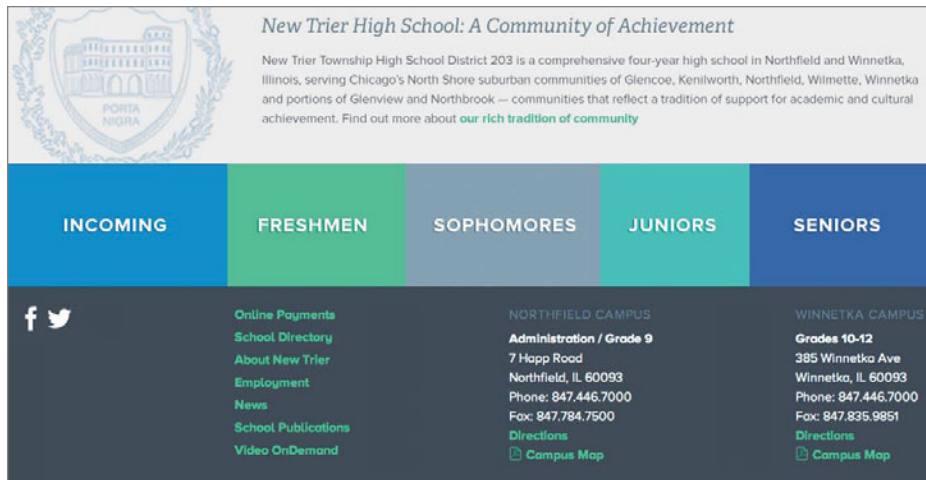


Figure 4.4 Bigger buttons. Don't laugh—you would have done the same thing.

Again, the clients were underwhelmed. They expected to have buttons and appreciated the increased size, but they thought it was a missed opportunity to tell more of the story behind the content. It's a high school, and these portals are all part of the journey.

It wasn't until I toyed around with that section in Photoshop that I started to layer some new elements in, like screened photos and sideways text. The point isn't that those techniques are exclusive to Photoshop (because they're not: adding a simple background image and "transform: rotate" in an HTML & CSS editor is just as easy), but the idea came more naturally in Photoshop.

Sarah Parmenter had a similar assessment of designing in the browser.

“It's a guilty secret I've been harboring for about a year: I cannot design directly into the browser. My creative brain switches at the point when I open my HTML/CSS editor (Espresso) and starts thinking in terms of structure and how to achieve the look of my design using as much native CSS as possible. If I don't have my design to follow, the whole process breaks down for me. I've tried, goodness knows I've tried, but my designs end up suffering, looking boxy, bland, and uninspiring.”

—SARAH PARMENTER ([www.sazzy.co.uk/2012/02/why-i-cant-design-in-the-browser/](http://www.sazzy.co.uk/2012/02/why-i-cant-design-in-the-browser/))

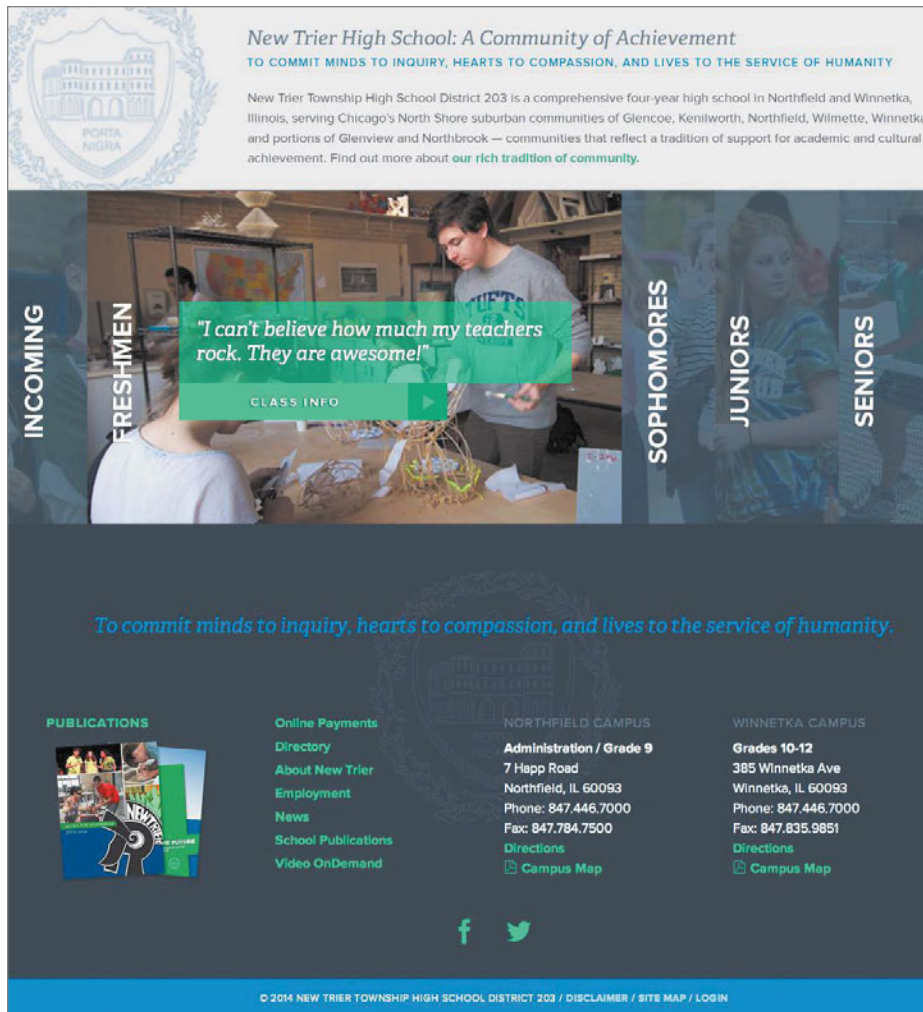


Figure 4.5 A much more robust and considered direction

While I won't advocate for having your entire design predetermined before HTML & CSS, the example shown in **Figure 4.5** is a case where taking one small section or component into Photoshop makes a world of difference.

As unpopular as the idea may be, I think it's easier to take the path of least resistance in the browser than in Photoshop; working in an HTML & CSS editor makes you more inclined to think in terms of boxes, containers, and the styles you usually employ than ones you don't. Heavy layering, offsetting only a single element and not others, and breaking convention usually mean tweaking more than just a few HTML tags and CSS properties. Sometimes, that amount of trial isn't worth the inevitable error. Suffice it

to say, for whatever reason, I'm more inclined to play it safe in the design techniques I use in code than in Photoshop.

Is it possible that lack of courage in our code-exclusive designs is the reason that all RWD sites look so similar?

## Responsive Design Sameness

The idea that responsive sites all look the same isn't a new one. It's a topic that's been broached many times, and you'll find some interesting perspectives on it.

“Have you ever noticed how many websites look the same? How many are just a bunch of rectangles? How many seem to copy from one another? Where's the uniqueness and creativity?”

—STEVEN BRADLEY (*The Web is a Rectangle*, [www.vanseodesign.com/web-design/rectangular-web/](http://www.vanseodesign.com/web-design/rectangular-web/))

“I feel like responsive design has sucked the soul out of web-site design. Everything is boxes and grids. Where has the creativity gone?”

—NOAH STOKES (<http://esbueno.noahstokes.com/post/44088237921/where-has-all-the-soul-gone>)

“I wonder if #RWD looks the way it does because so many projects aren't being run by designers but by front-end dev teams.”

—MARK BOULTON (<https://twitter.com/markboulton/status/445943150247702528>)

The question of “Why do all RWD sites look the same?” is a pertinent one when you consider that diversity in the pre-RWD era was fairly extraordinary. One could even argue that Flash-based sites were the most distinctive of the lot. Today, it's commonplace to pick out the sites designed in the browser from ones designed in Photoshop et al. This notion is where we get such statements like “It looks like a Bootstrap/Foundation site” and “It's like they just slapped their logo on a responsive WordPress theme.”

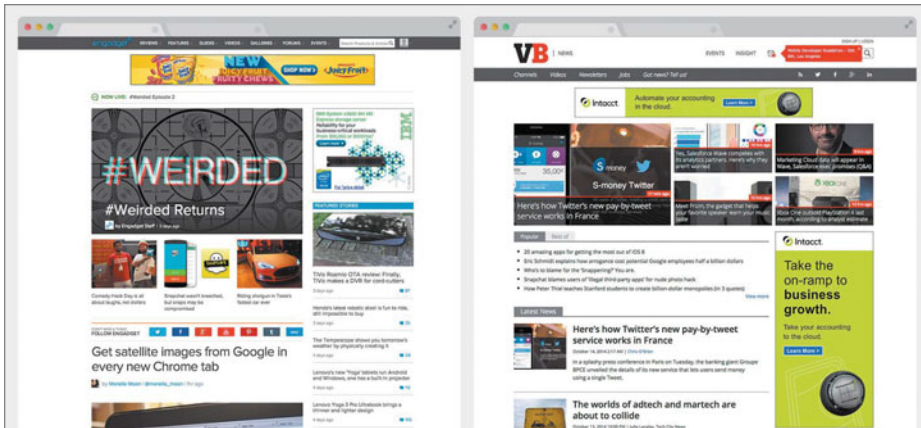


Figure 4.6 Examples of similarities in responsive site design. Common components include “boxy” layouts, white background headers, and hero images with white text overlaid.

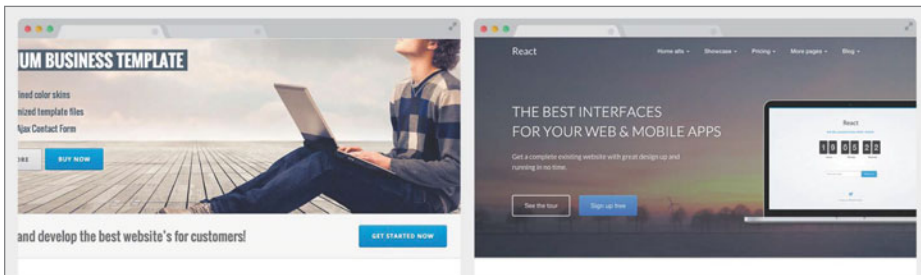


Figure 4.7 Examples of similarities in sites designed on the Bootstrap framework. Unadjusted button styles are a clear giveaway.

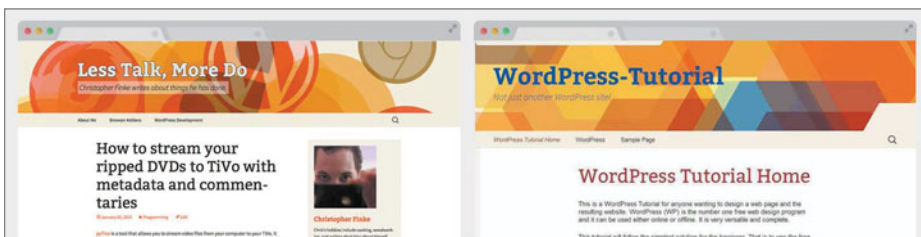


Figure 4.8 Examples of similarities in sites designed as WordPress themes. Similar content layout is a hallmark of blogs.

To some extent, there will undoubtedly be shared styles and components from site to site, no matter how original we think we’re being (Figures 4.6-4.8). That’s OK. I’d much rather have a Web that works than one that succeeds in diversity for diversity’s sake. Remember, the originality we established using Photoshop didn’t always translate to

the browser in a way that was user-friendly, either. Karen McGrane made this witty retort to the complaint of RWD sameness as it applies to print:

“But why do all the magazines have the binding on the left? And columnar layouts, so tired. Where’s the innovation?”

—KAREN MCGRANE (<https://twitter.com/karenmcgrane/status/515162632551411712>)

I’m not sure we can fault responsive design for a lack of uniqueness in sites, but certainly its introduction has added to the number of design considerations we need to make. By extension, new considerations may not always be tested across all screen sizes, so we rely on what “works” and what’s familiar. We’re still getting our feet wet designing responsively; creativity will come.

That doesn’t mean we need to stifle uniqueness or “soul” in the name of being responsively correct. We need to think creatively about being creative.

“Emulation is a part of the evolution of design. And the web, for that matter. But design sameness tends to fade when one forgets all of the existing patterns, all of the Bootstraps, all of the preconceived design solutions. Design sameness fades when designers stop focusing on which solutions for their problem are out there and start focusing on the problem at hand.”

—STEPHEN HAY ([www.the-haystack.com/2014/03/21/responsive-design-sameness/](http://www.the-haystack.com/2014/03/21/responsive-design-sameness/))

I contend that Photoshop can greatly assist in our efforts to infuse our designs with the unconventional and unique. Where designing in the browser may introduce friction in the creative process, Photoshop can help us focus on the best approach regardless of the amount of code needed to execute it.

## Using Photoshop Only When Necessary

Photoshop can ride along with designing in the browser, but we need to determine to what degree. It would be counterproductive to revert to full-page comps, but we still need Photoshop to throw down some high-fidelity ideas on. Likewise, we still need the

browser to keep our designs flexible, while curtailing the amount of time and effort we exhaust trying to come up with truly unique ideas.

If a responsive workflow is your goal, I recommend adopting the following philosophy: *Photoshop can't be the workhorse we've made it in years past*. Instead, Photoshop takes a backseat to in-browser design, but we still use it to ideate when necessary. To what extent do we keep Photoshop involved?

In his article “Responsive Web Design in the Browser Part 1: Kill Photoshop,” Josh Long makes the following statement:

“If you want to be a better designer, I'd start by killing Photoshop”

—JOSH LONG (<http://blog.teamtreehouse.com/responsive-web-design-in-the-browser-part-1-kill-photoshop>)

While some of this sentiment seems in line with what we've explored, I object to the notion that discarding any tool makes you a better designer. Becoming a better designer has more to do with knowing the limits of the tools you use and knowing when to use them for the greatest gain.

The browser can bear the majority of many layout and style decisions, which is great because it's ultimately where our design will live. We'd be shooting ourselves in the foot if we tried to make every decision before writing a line of HTML or CSS. That puts the onus on adopting a workflow that allows the flexibility needed to refine design solutions after being in the browser (I'll talk more about how to sell this kind of process internally and externally in Chapter 12).

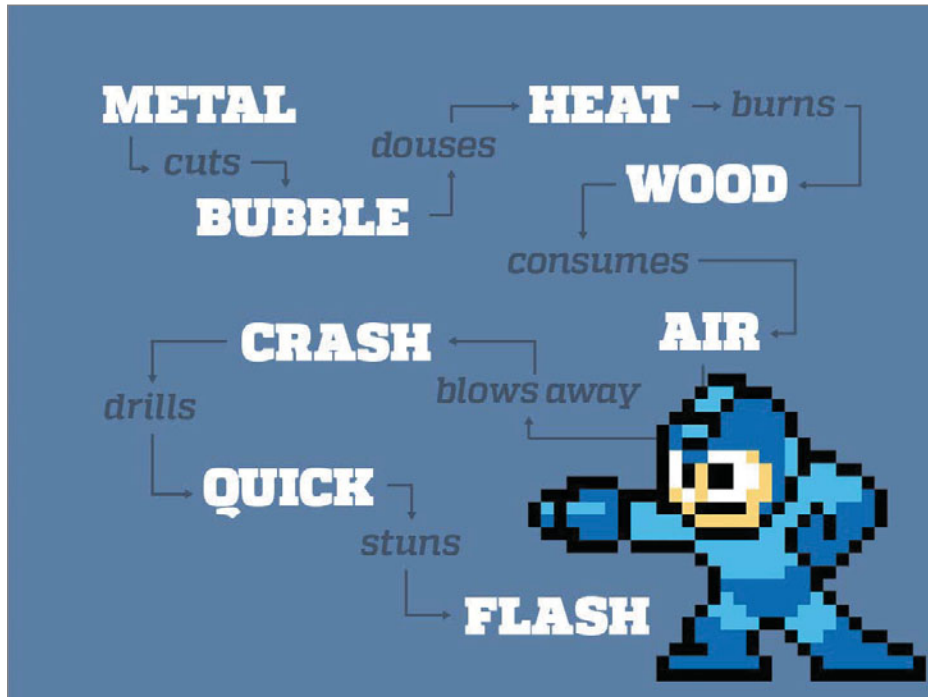
## The Megaman Principle

Raise your hand if you're also a child of the 1980s and grew up playing Nintendo. No, not you, Wii whiz kids. Not you, Atari fans. I'm talking about the 8-bit Nintendo Entertainment System. Specifically, I'm talking about my favorite series of games: Megaman.

For those unfamiliar, Megaman was pretty much the best thing ever. You controlled a blue, android-like robot in a classic quest of good versus evil. Your mission was to take down an army of bad robots, each of which had a unique weapon, be it ice, fire, stones, bubbles, or the even more obscure piles of trash Junk Man used. When you defeated these bad guys, you'd inherit their weapon.



In most instances, you'd be hard-pressed to find a use for your default Mega Buster arm cannon, so the quintessential strategy for beating each boss was to use another's weapon. Ice Man's Ice Slasher was critically effective against Fire Man, for example. Choosing the right weapon at the right time was, essentially, a glorified game of Rock-Paper-Scissors. Mastering when to use each weapon made the game infinitely easier (see **Figure 4.9**).



**Figure 4.9** Just as Megaman leveraged timing to beat his enemies, can we do the same with Photoshop? Specifically, I mean the enemies called “boxy,” “bland,” and “uninspiring.”

Collecting weapons and knowing when to use them is not so different from the problem we face in web design. By using the right tool, at the right time, for the right purpose, we extract more out of said tool than normal. Knowing when to use Photoshop is the only thing that can logically keep it in our workflows. Using it too often, too early, or for the wrong purpose produces frustration, wasted time, and potentially wasted money.

## Practical Photoshopping: An Overview

In the next few chapters, you'll look at when, why, and how to use Photoshop alongside the browser. Because every project is different, implementations will vary, but you'll discover a few inventive ways of getting the most out of Photoshop.

### Inspiration

If our plan is to present multiple design concepts to clients, where does RWD fit in? It seems like a daunting task to show contrasting design directions through Photoshop mock-ups at a few different sizes, so what's a more effective way to move the project forward?

The secret may be getting everyone on track toward a shared ideal far before anything is done in Photoshop. You'll explore a few ways to hedge what would otherwise be discarded work by using inspiration from around the Web to drive your directions. Equally important, you'll discover techniques for extracting more useful feedback from your clients.

### Art Direction

The conundrum of page comps in a responsive world hasn't been ignored. While some of the industry has turned to producing tools to make comps more quickly, a strong contingent of designers have devised better design documents altogether. You'll take a look at this landscape in Chapter 6, including one I find to be the best suited for a responsive process: element collages.

### Building Blocks

You may have created style guides in the past or be familiar with their longstanding print implementation. I'll define what style guides mean for the Web and, more importantly, as part of a responsive process.

Similarly, component (or pattern) libraries extend what's introduced in a style guide into functional collections of elements.

While it behooves us to develop both of these elements in code, there's plenty of opportunity to spice them up in Photoshop, long before we run into any funky page designs.

### Repair

It's incredibly optimistic—naïve, even—to expect all the disparate elements of a responsive site design to come together seamlessly. When it's time to build pages from

components, we usually find ourselves scrambling to make everything look like the coherent page it should be.

Let's determine to expect the wonkiness of putting components together and combat the seamless spots with Photoshop. You'll take a look at some common examples of where page design can fall short and feel disjointed in the absence of a system and pivot swiftly into an environment to try some stitch-work.

All of this preparation will lead to the union of Photoshop and the browser and to a better responsive workflow.

# Index

## Numbers

- 0to255 tool, for color variation, 179–180
- 320px width comps, for smartphones, 18–19
- 768px width comps, for tablets, 18–20
- 960px width comps, for desktops, 18–20

## A

- abstraction, balanced with detail, 91
- accessibility of files, 202
- accordions, 122
- accuracy, of in-browser design, 42–43
- active states, in Photoshop comps, 22
- Adobe Color, 180
- Adobe Color CC, 181–182
- Adobe Creative Cloud. *see* CC (Creative Cloud)
- Adobe Edge Reflow, for in-browser design, 33–34
- Adobe Generator
  - asset extraction overview, 152–153, 219.  
*see also* asset extraction
  - background asset generation, 153
  - exporting to folders/subfolders, 156–157
  - file formats produced by, 154–155
  - layer naming for, 157
  - Photoshop CC addition, 30
  - pixel precision with, 153–154
  - sizing images/group folders with, 156
  - UI layer, Extract Assets, 157–160
- Adobe Photoshop. *see* Photoshop
- Adobe Typekit, 24, 162, 207–208
- adopting new workflow
  - client indecision on direction, 226
  - designer-developer discord, 227
  - external acceptance, 224–226
  - internal acceptance, 220–224
  - promotion difficulties, 219–220
  - unsuccessful element collage, 226–227
- Aizlewood, Jon, 96
- alerts, in component libraries, 123
- Allsopp, John, 15
- Aptana Studio, 40
- archiving files, 201
- art direction, for Photoshop-browser harmony, 61
- artwork plug-ins/tools
  - Bjango Actions, 175–176
  - Composer, 177
  - DevRocket, 174–175
  - LayoutWrapper, 177–178
  - Pictura, 172–173
  - Random User Generator, 171
  - RotateMe, 178
  - Social Kit, 172
  - Subtle Patterns, 169
  - Transform Each, 173–174
  - WebZap, 176–177
- asset compression, with ImageMagick, 154
- asset extraction
  - Adobe Generator for, 152–157, 219
  - Copy Merged for, 150–151
  - crop and save for, 148–150
  - Extract Assets for, 157–160, 219
  - Extract online service, 160–165
  - Save for Web in, 151–152
- asset plug-ins/tools
  - FlatIcon, 186
  - Glifo, 185
  - ImageOptim, 187
  - iOS Hat, 183–184
  - OtherIcons, 184
  - TinyPNG, 186–187
- Asset Resolution presets, 159
- assets
  - accounting for, 212
  - storing relative to PSD, 202
- Assets tab, 162
- Atomic Design* (Frost), 222

**B**

backgrounds  
 fixed positioning of, 23  
 gradient, 35  
 Bearded design shop, 100  
 the big inclusion, 67  
 the big reveal, 25–26, 67  
 bird’s-eye view, for cohesive style, 142  
 Bjango Actions, 175–176  
 blend modes  
 careful use of, 207  
 with CSS, 35  
 bloat, 128  
 Bootstrap framework sites, 56–58  
 Bowles, Cennyd, 145  
 Brackets, 40  
 breadcrumb navigation, in component  
 libraries, 121  
 breakpoint(s)  
 difficulty of three, 18–22  
 focus on single, 4  
 browser(s)  
 compatibility, awareness of, 212–213  
 design variations in, 44  
 Photoshop as complement to, 135–136  
 resizing, 37–38  
 style guide created in, 114  
 style prototypes for, 84–87  
 web design in. *see* in-browser design  
 buttons  
 calls-to-action, 119  
 for CC Libraries, 125  
 in style guides, 111  
 Buxton, Bill, 65

**C**

Callahan, Ben, 222  
 calls-to-action buttons, 119  
 canvas size, for element collages, 91–92  
 Cascading Style Sheets. *see* CSS (Cascading  
 Style Sheets)  
 CC (Creative Cloud)  
 accessing character styles in, 88  
 Adobe Color CC, 181–182  
 benefits of, 9  
 for Extract online service, 160–165  
 reducing costs with, 30

CC Libraries  
 creating buttons for, 125  
 features/functions of, 124–126  
 CC Market, as art resource, 170  
 CCM (Christian Care Ministry), 92–93, 95  
 Chapman, Shaun, 180  
 character styles, in Photoshop, 88  
 Chrometaphore’s Ink plug-in, 192–193  
*Clandestine Photoshop* (Rose), 222  
 Clarke, Andy, 150  
 Clearleft, 96, 97–98  
 client presentations  
 the big reveal, 25–26  
 design process involvement, 66–67  
 educating on design, 45–46  
 element collages in, 101–102  
 on in-browser design, 43  
 offering contrasting options, 64–66, 74  
 pursuing efficiency in, 75–77  
 selling new workflow ideas, 224–226  
 with traditional Photoshop, 26–27  
 visual inventories for, 74  
 Cloud.typography, 24  
 Coda, 40  
 Code School, 5  
 Codecademy, 5  
 coding skills, 4–6  
 cohesive page design  
 big picture perspective for, 140–142  
 bird’s-eye view in, 142  
 lack of, 140  
 skeuomorphism and, 142–143  
 collages, element. *see* element collages  
 collaging format, for moodboards, 68, 69  
 color plug-ins/tools  
 0to255 tool, 179–180  
 Adobe Color, 180  
 Adobe Color CC, 181–182  
 Colorus, 182  
 color(s)  
 cohesive design and, 142–143  
 comparisons, in element collages, 99–100  
 extraction of, 162–163  
 manipulating with SVG mages, 155  
 in style guide, 109–110  
 comments, in component libraries, 123  
 communication  
 client, 17–18, 225. *see also* client  
 presentations

- designer-developer, 47–48, 227
- internal, 17, 221–222, 223
- multiple channels of, 70–71
- comp approach, to design options, 64–65. *see also* full-page comps
- component inventories, 87–89
- component libraries
  - best environment choice for, 124–126
  - breadcrumbs/pagination in, 121
  - calls-to-action/sign-up modules in, 119
  - comments/alerts/latest news/recent feeds in, 123
  - easier page-building and, 218
  - expanding panels in, 122
  - forms/input types in, 116
  - hero blocks in, 117–118
  - image grids/media blocks in, 117
  - image with caption in, 118
  - navigation in, 119–120
  - overview of, 114–115
  - for Photoshop-browser harmony, 61
  - sidebar callouts in, 120–121
  - tabbed panels in, 121–122
- Composer plug-in, 177
- Composite app, for interactive prototypes, 188–189
- comps
  - fixed-width, static, 18–22
  - full-page. *see* full-page comps
- conferences, design, 223–224
- contrast
  - defining, 74
  - in design options, 64–66
- conventions, creating, 200
- conversations, in design process, 77
- Coolorus plug-in, 182
- Copy All option, 163–164
- Copy CSS function, 164–165
- Copy Merged, for asset extraction, 150–151
- Creative Cloud. *see* CC (Creative Cloud)
- creativity, in Photoshop, 52
- cropping
  - Adobe Generator precision in, 154–155
  - in asset extraction, 148–150
- CSS (Cascading Style Sheets)
  - building a style guide in. *see* style guides
  - coding skills with, 4–5
  - component libraries and, 124
  - controlling hover state, 155

- correct mode in, 52
- Extract generating, 163–164
- Photoshop effects with, 35
- Photoshop generating, 164–165
- setting up colors in, 110

## D

- A Dao of Web Design* (Allsopp), 15
- deliverables
  - conversations vs., 77
  - style guides as, 114
- design conferences, 223–224
- design gallery sites, 71
- design process. *see also* web design
  - including stakeholders in, 66–67
  - moodboards in, 67–72
  - offering contrasting options, 64–66, 74
  - pursuing efficiency in, 75–77
  - through conversations, 77
  - visual inventories for, 72–75
- designer-developer communication, 47–48, 227
- DevRocket plug-in, 174–175
- documentation, Ink plug-in for, 192–193
- downloading assets, 162
- Dribbble design gallery site, 71
- drop shadows, with CSS, 35
- Dziak, Jason, 67

## E

- Ebert, Ralf, 133
- Edwards, Marc, 175
- effects etiquette
  - deliberate layer effects, 210–211
  - Overlay use/tileable images, 210
- efficiency, Photoshop etiquette and, 199
- element collages
  - abstraction and detail in, 91
  - color comparisons in, 99–100
  - crafting, 91–93
  - description/function of, 90–91
  - dissimilar to websites, 96–99
  - full-page comps and, 104–105
  - iterative design with, 94–96
  - limiting scope of, 100–101
  - as point of reference, 103–104
  - style exploration with, 216–217
  - useful client feedback for, 101–102

- visual metaphors in, 92–93
- visualizing website from, 103
- Ember cataloging tool, 71–72
- Emmet toolkit, 36
- Enns, Blair, 77
- Erondu, Jared, 51
- establishing style. *see* style, establishing
- etiquette, Photoshop
  - effects and, 209–211
  - files/folders and, 201–202
  - images and, 205–207
  - inheriting PSDs and, 197–198
  - layers and, 203–205
  - quality assurance, 211–213
  - reasons for, 199–200, 219
  - type and, 207–209
  - website, 198
- Evernote cataloging tool, 71
- Evolving the Digital Style Guide* (Pratt), 222
- expanding panels, in component libraries, 122
- expectations
  - in adopting new workflow, 221–222
  - resetting, with clients, 224–225
- extending Photoshop
  - artwork tools for. *see* artwork plug-ins/tools
  - assets for, 183–187
  - CC Market for, 170
  - color tools for, 179–182
  - organizational tools for, 191–194
  - overview of, 167–168, 219
  - Photoshop proficiency tools, 194–196
  - prototyping tools for, 188–190
  - website, 168
- Extract Assets
  - image etiquette and, 207
  - overview of, 157–158, 219
  - setup for, 158–160
- Extract online service
  - asset extraction with, 162
  - extracting values with, 162–163
  - generating CSS with, 163–165
  - overview of, 160
  - setup for, 161
- extracting assets. *see* asset extraction
- extracting values, 162–163

## F

- faults, Photoshop. *see* Photoshop faults
- featured articles, in component libraries, 123
- fidelity, Page Layers and, 135
- Fight phase, of ShortcutFoo app, 195
- file share, for accessibility, 202
- file size
  - importance of, 149–150
  - reducing, 187
- filename clarity, 201
- filters, nondestructive technique for, 206
- fixed positioning, of background, 23
- fixed-width comps
  - multiple, 20–22
  - three-breakpoint trap and, 18–20
- FlatIcon, free icons with, 186
- flexibility, of Illustrator, 173–174
- flexible grids, 8
- Flickr integration, with Pictura, 172–173
- fluid images, 8
- focus states, in Photoshop comps, 22
- the fold, 46–47
- folders
  - Adobe Generator exporting to, 156–157
  - asset storage in, 202
- fonts
  - with CSS, 35
  - extraction of, 162–163
  - in Photoshop comps, 23–24
  - rendering difficulties of, 24–25
  - standardizing access to, 207–208
  - true rendering of, 39–40
- Foral, Jason, 177
- forms, in component libraries, 116
- Framer app, for interactive prototypes, 188
- Fried, Jason, 26, 41
- Frost, Brad, 9, 222
- full-page comps
  - assessment limitations, 28–29
  - code aversion/waterfall method and, 16–17
  - definition of, 15–16
  - element collages and, 104–105
  - as exercises in proofreading, 43–44
  - faulty communication and, 17–18
  - inaccurate representation, 26
  - inefficiency of, 25–26
  - noninteractivity of, 22–23, 215–216
  - past function/goals of, 16

- presentation difficulties, 26–27
- replacements for, 80–87
- static fixed-width, 18–22
- web fonts and, 23–24

Future of Web Design conference, 219

## G

Gaussian Blur filter, 206

Generate function, in iOS Hat, 184

Generator. *see* Adobe Generator

*Get Your Visual Style On* (Perez-Cruz, Callahan & Mall), 222

GIF images, Generator exporting, 154

Gimp, 30

GitHub

- file accessibility with, 202
- psdiff tool previewing in, 193–194

Glifo, packaging icons with, 185

global changes, in-browser design and, 39

gradient backgrounds, with CSS, 35

gray-box mock-ups

- creating, 126–128
- exporting to Photoshop, 135

grids, flexible, 8, 117

group folders, Generator exporting, 156

GuideGuide plug-in, 191

## H

Hay, Stephen, 22

hero blocks, in component libraries, 117–118

high-fidelity prototypes, 129–130

hover states

- CSS controlling, 155
- in Photoshop comps, 22

HTML (Hypertext Markup Language)

- building a style guide in. *see* style guides
- coding skills with, 4–5
- component libraries done in, 124
- correct mode in, 52
- design choice of, 3
- designing in browser with. *see* in-browser design
- exporting components from, 133–134

## I

icons

- packaging with Glifo, 185
- in style guides, 112–113

Illustrator flexibility, 173–174

image

- with caption, 118
- dimensions, 112

image assets

- Generator exporting. *see* Adobe Generator
- sizing, 156, 159

image blocks, in component libraries, 117

image borders, with CSS, 35

image etiquette

- blend modes/resolution/density in, 206
- use of nondestructive techniques, 205–207

image grids, in component libraries, 117

ImageMagick, 154

ImageOptim, reducing file size with, 187

in-browser design

- accuracy of, 42–43
- component library for. *see* component libraries
- with CSS/HTML, 34–35
- designer-developer communication, 47–48
- editing on the fly, 45
- educating clients and, 45–46
- evaluating design behavior in, 43–44, 216
- fluid width with, 37–38
- the fold and, 46–47
- global changes and, 39
- in-progress viewing, 43
- Inspect Element option, 37
- interactivity in, 38
- less effort/more reality in, 40–42
- limitations of, 216
- minimizing time in Photoshop, 135–136
- no additional cost in, 40
- options for, 33–34
- prototypes for. *see* prototypes
- realistic expectations with, 44
- style guide for. *see* style guides
- text editor for, 35–36
- true font rendering and, 39–40

InDesign character styles, 88

Ink plug-in, for documentation, 192–193

inner shadows, with CSS, 35

input types, in component libraries, 116

Inspect Element option, 37

inspiration

- approach to moodboards, 68–70
- finding/storing, 70–72
- for Photoshop-browser harmony, 61



interactivity, lack of, 22–23  
 InVision app, for interactive prototypes, 190  
 iOS devices, DevRocket for, 174–175  
 iOS Hat, 183–184  
 iStock Photo watermark, 212  
 iterative design, 94–96

## J

Jehl, Scott, 9  
 JPEG images, Generator exporting, 154, 155

## K

Kardys, Dennis, 65  
 Khadeyev, Kamil, 174  
 Kipt cataloging tool, 71

## L

latest news, in component libraries, 123  
 layer naming
 

- accuracy in, 203–204
- Adobe Generator improving, 157
- as-you-go method, 204
- Extract Assets and, 159

 layers
 

- converting to Smart Objects, 206
- deleting unnecessary, 205
- effects for, 210–211
- groups/globalizing, 204–205

 Layout Wrapper, framing work in Safari browser, 177–178  
 Learn phase, of ShortcutFoo app, 195  
 Libraries. *see* CC Libraries  
 link styles, in style guides, 111  
 Linked Smart Objects, leveraging of, 136–137  
 live previewing, 36  
 LiveShare plug-in, for interactive prototypes, 190  
 Loggins, Kenny, 136  
 Long, Josh, 59  
 low-fidelity prototypes
 

- creating, 126–128
- exporting to Photoshop, 135
- frameworks for, 128

 Loyd, Jeremy, 85

## M

Macaw design app, 33–34, 180  
 Mall, Dan, 17, 72, 90, 92, 98, 222

Marcotte, Ethan, 8  
 McEfee, Cameron, 191  
 McGrane, Karen, 58  
 Measure function, in iOS Hat, 184  
 media blocks, in component libraries, 117  
 Media Queries design gallery site, 71  
 Megaman principle, 59–60  
 Messina, Jim, 136  
 Mo, Atle, 169  
 mock-ups
 

- full-age comps for, 16–18
- gray-box, 126–128, 135
- replacements for, 80–87

 Monteiro, Mike, 102  
 moodboards
 

- finding/storing inspiration for, 70–72
- organic inspiration approach to, 68–70
- purpose of, 67–68
- traditional collaging format, 68, 69

 movement, in Photoshop comps, 23  
 multiple designers, Linked Smart Objects for, 136–137  
 multiple formats, Generator exporting, 156

## N

naming layers. *see* layer naming  
 navigation, in component libraries, 119–120  
 New Guides, 139  
 news, in component libraries, 123  
 Nice design gallery site, 71  
 Notepad, 40

## O

one-direction approach to design options, 65  
 organic inspiration approach to moodboards, 68–70  
 organization, Photoshop etiquette and, 199–200  
 organizational plug-ins
 

- GuideGuide, 191
- Ink, 192–193
- psdiff, 193–194
- Renamy, 192

 OtherIcons plug-in, 184

## P

Page Layers
 

- benefits of working with, 138–140, 218

- exporting components to Photoshop with, 133–134
- improving cohesiveness with, 140–143
- increasing fidelity with, 135
- New Guides/Smart Guides for, 139
- old screenshot methods vs., 137–138
- width-specificity in, 145–146
- pagination, in component libraries, 121
- Parmenter, Sarah, 54
- Pattern Tap design gallery site, 71
- Perez-Cruz, Yesenia, 222
- performance of RWDs, 9
- Photoshop
  - basics/version disparities of, 9
  - character styles in, 88
  - as complement to browser, 135–136
  - component libraries done in, 124
  - Copy CSS function in, 164–165
  - direct manipulation with, 51, 216
  - effects, utilizing CSS, 35
  - etiquette. *see* etiquette, Photoshop
  - extracting assets from. *see* asset extraction
  - falling out of favor, 2–3, 11
  - getting from browser back into. *see* Page Layers
  - modifying screenshots in, 137–140
  - optimizing. *see* extending Photoshop
  - personal histories using, 14
  - as preferred web design tool, 9
  - process with, 6–7
  - role in RWD. *see* Photoshop-browser harmony
  - single breakpoint focus with, 4
  - superior creativity in, 52–56
  - toolbars evolution, 13
  - traditional, faults of. *see* Photoshop faults
- Photoshop-browser harmony
  - creative mode/correct mode, 52
  - importance of, 216
  - Megaman principle for, 59–60
  - minimal use of Photoshop, 58–59
  - path of least resistance in, 52–56
  - Photoshop's manipulation in, 51
  - preparation for, 61–62
- Photoshop CC
  - Adobe Generator in, 152–157. *see also* Adobe Generator
  - Adobe Typekit in, 24
  - Character/Paragraph Styles in, 39
  - Linked Smart Objects in, 136–137
- Photoshop Color Shift option, 29
- Photoshop Etiquette Manifesto for Web Designers (website), 198
- Photoshop faults
  - assessment limitations, 28–29
  - code aversion/waterfall method and, 16–17
  - cost factor, 30, 40
  - exporting difficulties, 29–30
  - faulty communication and, 17–18
  - inaccurate representation, 26
  - increased workload, 31
  - inefficiency of, 25–26
  - instability of, 29
  - noninteractive, 22–23
  - presentation difficulties, 26–27
  - static fixed-width, 18–22
  - web fonts and, 23–24
- Photoshop proficiency tools
  - Photoshop Secrets, 195–196
  - ShortcutFoo app, 194–195
- Pictura plug-in, 172–173
- Pinterest tool, 68, 71
- pixel density, image etiquette and, 207
- pixel precision, of Adobe Generator, 154–155
- Pixelmator, 30
- plug-ins
  - artwork. *see* artwork plug-ins/tools
  - asset integration/generation, 183–187
  - color, 179–182
  - LiveShare for interactive prototyping, 45, 190
  - organizational, 191–194
- PNG images, Generator exporting, 154, 155
- The Post-PSD Era* (Mall), 17
- Pratt, Andy, 222
- presentations
  - promoting new workflow ideas, 220–221
  - on style, 222
- proofreading, 211–212
- prototypes
  - high-fidelity, 129–130
  - low-fidelity, 126–128
- Prototyping Style* (Callahan), 222
- prototyping tools
  - Framer/Composite/Stand In, 188–189
  - InVision, 190

PSD, asset storage relative to, 202  
 psdiff tool, for PSD previewing in GitHub,  
 193–194

## R

Raindrop cataloging tool, 71  
 Random User Generator, 171  
 raster graphics, with Page Layers, 135  
 recent feeds, in component libraries, 123  
 RedPen tool, 27  
 Renamy plug-in, 192  
 repair work, 61–62  
*Responsible Responsive Design* (Jehl), 9  
 responsive design sameness, 140  
*Responsive Design Workflow* (Hay), 22  
 responsive patterns, 8–9  
 responsive web design. *see* RWD (responsive web design)  
*Responsive Web Design in the Browser Part 1: Kill Photoshop* (Treehouse blog), 2  
*Responsive Web Design* (Marcotte), 8  
 responsive wonkiness  
   fixing, 145–146  
   identifying, 143–145  
 reverse-engineering style tiles, 83  
 RotateMe script, 178  
 rounded rectangles, with CSS, 35  
 RWD (responsive web design)  
   in browser. *see* in-browser design  
   contrasting options for, 64–66, 74  
   core tenets of, 8  
   including stakeholders in, 66–67  
   lack of cohesive style, 140–143  
   Linked Smart Objects for, 136–137  
   new workflow for. *see* adopting new workflow  
   Page Layers for, 133–135  
   performance of, 9  
   with Photoshop, 7  
   Photoshop etiquette in, 200  
   Photoshop vs., 2–3  
   potential difficulties in, 132–133  
   repurposing tools for, 228  
   responsive patterns in, 8–9  
   responsive wonkiness and, 143–146

## S

Sanchez, Edward, 195–196

Save for Web option, for asset extraction,  
 29–30, 151–152  
 Save function, in asset extraction, 149–150  
 Scalable Vector Graphic (SVG) images, 155  
 scope creep, 100–101  
 screen resolution, image etiquette and, 207  
 screenshots  
   New Guides/Smart Guides for, 139  
   old methods for modifying, 137–138  
   Page Layers for modifying, 138–140  
 search filters, 70  
 selling workflow ideas internally  
   attend conferences, 223–224  
   establish supporters first, 222–223  
   give presentations, 220–221, 222  
   set realistic expectations, 221–222, 223  
 selling workflow ideas to clients  
   affirm direction, 225–226  
   increase communication/involvement, 225  
   reset expectations, 224–225  
 Sheeran, Norm, 176  
 ShortcutFoo app, 194–195  
 sidebar callouts, 120–121  
 sign-up modules, 119  
 Site Inspire design gallery site, 71  
 sizing images  
   with Adobe Generator, 156  
   previewing size for, 159  
 Sketch tool, 3, 30  
 sketching, 80  
*Sketching User Experiences* (Buxton), 65  
*Sketching User Experiences: The Workbook*  
   (Greenberg et al.), 80  
 skeuomorphism, cohesive design and,  
 142–143  
 slicing, 30  
 Smart Guides, 139  
 Smart Objects  
   converting layers to, 206  
   editing, 136  
 Social Kit plug-in, 172  
 Sparkbox, 84  
 spelling errors, proofreading for, 211–212  
 stakeholders, in design process, 66–67  
 Stand In app, for interactive prototypes,  
 188–189  
 state changes, in Photoshop comps, 22  
 style, establishing  
   component inventories for, 87–89

- element collages for. *see* element collages sketching, 80
- style prototypes for, 84–87
- style tiles for, 81–83
- style guides
  - color in, 109–110
  - created in browser, 114
  - defining, 108–109
  - icons in, 112–113
  - link styles/buttons in, 111
  - for Photoshop-browser harmony, 61
  - tables/image dimensions in, 112
  - typography in, 110–111
- style prototypes
  - advantages of, 216
  - design elements of, 85
  - flexibility of, 86–87
  - function of, 84
- style tiles
  - description/advantages of, 82
  - exploration through, 216
  - as mock-up alternative, 81
  - reverse-engineering, 83
- Styles tab, 163
- Sublime Text, 40
- Subtle Patterns plug-in, 169
- SVG (Scalable Vector Graphic) images, 155
- swatches, of design color options, 110

## T

- tabbed panels, in component libraries, 121–122
- tables, in style guides, 112
- Terrett, Ben, 4
- Test phase, of ShortcutFoo app, 195
- text blocks, in component libraries, 117
- text boxes, controlling/separating, 209
- text editor, for in-browser design, 35–36
- TextEdit, 40
- The Win Without Pitching Manifesto* (Enns), 77
- This Is Responsive* (Frost), 9
- three-breakpoint trap, 18–20
- three-direction approach, to design options, 64–65
- “throw it over the wall” workflow, 16, 17
- thumbnails, use of Overlays and, 210
- time savings, with Page Layers, 139–140

- TinyPNG plug-in, for compressing PNGs, 186–187
- toolbars, Photoshop, 13
- tools
  - artwork. *see* artwork plug-ins/tools
  - asset integration/generation, 183–187
  - color, 179–182
  - for in-browser design, 33–34
  - merits of comparable, 9–10
  - organizational, 191–194
  - prototyping, 188–190
  - repurposing existing, 228
- Transform Each script, 173–174
- transparency
  - with CSS, 35
  - in design process, 66–67
- Treehouse, 5
- Twitter Bootstrap site, 128
- type etiquette
  - control/separate text boxes, 209
  - no stretching, 208
  - proofreading, 211–212
  - standardize font access, 207–208
- typography, in style guide, 110–111

## U

- UI Parade online catalog, 176
- UI (user interface), Extract Assets as, 157–160
- Unmatched Style design gallery site, 71

## V

- Verizon Wireless, 150
- versatility, of style tiles, 82
- vetting direction
  - importance of, 216
  - including stakeholders in, 66–67
  - moodboards in, 67–72
  - offering contrasting options, 64–66, 74
  - pursuing efficiency in, 75–77
  - through conversations, 77
  - visual inventories for, 72–75
- video, in responsive web design, 8
- vinyl records, 49–50
- visual inventories
  - for design process, 72–75
  - efficiency of, 75–77
- visual metaphors, in element collages, 92–93

**W**

- Warby Parker “Home Try-On” program, 40–41
- Warren, Samantha, 81
- waterfall method, 17
- web design. *see also* design process
  - coding skills for, 4–6
  - Photoshop as tool of preference for, 9
  - Photoshop history in, 14
  - responsive workflows for, 6–7
  - switching away from Photoshop, 2–3
  - weekly updates on new ideas, 70
- web fonts
  - with CSS, 35
  - extraction of, 162–163
  - in Photoshop comps, 23–24
  - rendering difficulties of, 24–25

- standardizing access to, 207–208
  - true rendering of, 39–40
- Webflow app for in-browser design, 33–34
- websites
  - element collage dissimilarity, 96–99
  - element collages leading to, 103
- WebZap plug-in, 176–177
- weekly updates, on design ideas, 70
- wonkiness
  - fixing, 145–146
  - identifying, 143–145
- Woodworth, Ken, 68
- WordPress sites, 56–57

**Z**

- Zeldman, Jeffrey, 36
- Zurb Foundation site, 128