

DESIGNING WEB & MOBILE GRAPHICS

Christopher Schmitt

FUNDAMENTAL CONCEPTS FOR WEB AND INTERACTIVE PROJECTS



DESIGNING WEB & MOBILE GRAPHICS

FUNDAMENTAL CONCEPTS FOR WEB AND INTERACTIVE PROJECTS

Christopher Schmitt

DESIGNING WEB AND MOBILE GRAPHICS:

Fundamental concepts for web and interactive projects

Christopher Schmitt

New Riders

www.newriders.com

New Riders is an imprint of Peachpit, a division of Pearson Education.

To report errors, please send a note to errata@peachpit.com

Copyright © 2013 by Christopher Schmitt

Senior Acquisitions Editor: Michael J. Nolan

Associate Development Editor: Margaret Anderson/Stellarvisions

Version Wrangler: Rose Weisburd

Production Editor: Becky Chapman Winter

Copyeditor: Gretchen Dykstra

Indexer: James Minkin

Proofreader: Jan Seymour

Cover and Interior Designer: Charlene Charles-Will

Compositor: Kim Scott/Bumpy Design

Illustrator: Richard Sheppard

Page 105, Figure 5.13: Engraving by G.J. Stodart after a photo by Fergus of Greenack, before 1890; Figure 5.15: Autochrome photograph by Louis Lumiere, 1910. Page 128, Figure 6.3: Photo courtesy of Flickr user FaceMePLS, under a Creative Commons - Attribution license.

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.

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 nor Peachpit 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

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.

ISBN 13: 978-0-321-85854-2

ISBN 10: 0-321-85854-9

9 8 7 6 5 4 3 2 1

Printed and bound in the United States of America

*For Nick, Elisabeth, Matt, Mary Rose, Michael,
Ryan, Megan, Meredith, Gianna*

Acknowledgments

A book such as this is blessed to have been touched by many skilled and talented folks.

Thanks to Molly Holzschlag and Estelle Weyl for some last minute edits, and to Christina Ramey for figure and example development.

New Riders is an integral part of the web design community, having published groundbreaking books and continuing to push the industry today. Thanks to Michael Nolan for giving me this opportunity to be part of their legacy.

Special thanks to Rose Weisburd, who remained polite and cool as the whooshing sound of deadlines grew louder and louder. I have profound appreciation for the dedication and resourcefulness of the crew at Peachpit: Becky Winter, Tracey Croom, Margaret Anderson, Gretchen Dykstra, Jan Seymour, Kim Scott, James Minkin, Claudia Smelser, Richard Sheppard. You could not find a finer team anywhere.

From speaking at conferences and helping students to writing books, web education is more than a second job to me. So, a very special thanks to Ari Stiles for understanding how much I love what I do and realizing it's an integral part of who I am. I love her for that.

CONTENTS

Introduction	xi
--------------------	----

CHAPTER 1 Understanding HTML **3**

Why Learn HTML?	4
HTML Made the Internet Popular	4
Knowing HTML Gives You a Better Understanding of Web Design	5
Learning HTML Is Easy	5
HTML Coding Basics	7
The Text Editor	7
Word Processors Aren't Text Editors	7
Coding with HTML	8
Structuring a Web Page	9
Specifying the DOCTYPE	9
Saving and Viewing the Page	12
Structuring Page Content	13
HTML Headings	13
HTML Text Markup	14
Creating a Link to a Website	16
Adding a Title Attribute	17
Linking within a Site	17
In Conclusion	20

CHAPTER 2 Styling with CSS **23**

CSS to the Rescue	24
Getting to Know Style	25
Declarations	25
Selectors	25
More CSS Hooks	26
Adding CSS Formatting	28
Linking to CSS	28
Internal Styles	30
Block and Inline Formatting	31
HTML's Generic Elements	32
Pseudo-classes and Pseudo-elements	33

Delivering CSS Just to IE	36
Normal Flow and Positioning	38
Static Positioning	38
Fixed Positioning	39
Relative Positioning	39
Absolute Positioning	40
In Conclusion	43

CHAPTER 3 Web Typography **45**

Working with Web Type	46
Size Properties and Values	49
So, What Measurement Should You Use?	50
Weights, Styles, Variants, and Decorations	52
Practicing Safe Typography	55
Web-Safe Fonts	55
Mobile-Safe Fonts	56
Making Better Font Stacks	57
“Real Fonts” in Web Pages	58
The @font-face Property	59
Generating Files for Embedding	60
Free Real Fonts	62
Commercial Font Services	66
In Conclusion	69

CHAPTER 4 Challenges in Web Design **71**

The Web Environment	72
Revealing Browser Issues	72
Using a Test Page	74
Color	75
Screens and Pixels	76
Accessibility	77
Determining <i>Your</i> Cross-Browser Goals	79
Analyzing Your Site Traffic	79
Developing Your Site for Different Devices	87
Resetting and Normalizing Browser Styles	87
Vendor Prefixes	89
Validation	90
Testing	92
In Conclusion	94

CHAPTER 5	Color for the Web	97
	Coding Web Color	98
	Hexadecimal Color Notation	98
	Overriding HTML's Default Colors	99
	Image Borders	102
	Transparency with CSS Color	103
	Color Properties	104
	Primary Color Systems	105
	Color Combinations	106
	Building a Color Scheme	108
	Finding the Base Color	108
	Cultures and Color	111
	Browsing for Color Inspiration	112
	Picturing a Color Scheme	116
	Consistent Colors	117
	Calibrating Colors	117
	Picking Up Colors	117
	CSS Color Chart	117
	In Conclusion	125
CHAPTER 6	Images for the Web	127
	A Matter of Bits	128
	Bit Depth	128
	Posterization and Dithering	129
	More Bits Means More Colors	129
	Why Is Bit-Depth Important?	130
	Raster Image Formats	133
	GIF Image Format	133
	JPEG Image Format	134
	PNG Image Format	135
	Image Compression Chart	136
	SVG: Vector Files for the Web	144
	In Conclusion	145
CHAPTER 7	Creating Images for the Web	147
	Working in Illustrator	148
	Setting Up Workspace for Web	148
	Setting Document Sizes	149
	Exporting Raster Images	154
	Exporting Vector Images from Illustrator	156

Working in Photoshop	158
Setting Up a New Document	158
Exporting Raster Images	158
Naming Web Image Files	160
Reducing Image File Size	160
Compressing Raster Images	160
Using HTTP Compression	162
In Conclusion	165

CHAPTER 8 Transparency and Shadow **167**

Creating Transparency with GIFs	168
Matting	170
Transparency with PNGs	172
8-bit PNG	172
24-bit PNG	173
CSS Transparency	174
Rounding Corners	174
Image Masking	174
Element Opacity	176
Background Color Transparency	177
Text Shadow	177
3D Text Shadow	178
Box Shadow	178
In Conclusion	179

CHAPTER 9 Favicons and Mobile Bookmarks **181**

Where Favicons Are Found	182
Image Formats for Favicons	183
Inserting Favicons into a Website	184
Favicons for Your Subsite	184
Spread Your Favicons Around	184
Creating Favicons for Web Pages	185
Building Retina-Ready Favicons	186
Mobile Bookmarks	190
File Format	191
Naming Conventions	191
Automating Graphic Treatment	191
Inserting Icons as Mobile Bookmarks	192
Combining Favicons and Mobile Bookmarks	193
In Conclusion	193

CHAPTER 10	Lists and Icon Fonts	195
	Unordered Lists	196
	CSS List Icons	196
	Inserting Custom Icons	197
	Definition Lists	198
	Ordered Lists	199
	Changing the Order	199
	Setting up a Table of Contents	200
	Another Way to Add List Markers	201
	Effective List Design	203
	Icon Fonts	206
	Selecting an Icon Font	207
	Highlighting a Word or Phrase	208
	Making a Stand-alone Link Icon	209
	In Conclusion	211
CHAPTER 11	Image Maps	213
	Making Image Maps	214
	Basic Handcoding	214
	Using Fireworks	219
	Responsive Image Maps	223
	In Conclusion	224
CHAPTER 12	Laying Out Pages	227
	Float Behavior	228
	Float Property	228
	Multiple Floats	229
	Clear Property	229
	Laying Out Pages with Floats	230
	Page Structure	230
	Two-Column Fluid Layout	232
	Two-Column Fixed Layout	236
	Three-Column Fluid Layout	237
	Three-Column Fixed Layout	239
	Pros and Cons of Layouts with Floats	241
	CSS Frameworks	241
	Grid Systems	242
	Final Look at Frameworks	246

Responsive Layouts	247
Adapting to Media Queries	248
Fluid Layouts	249
Text Reflow	253
Media Queries in Action	254
Responsive Framework	259
In Conclusion	259

CHAPTER 13 Images for Responsive Web Design 261

Scaling Images and Media	262
The Problem with Scaling Images	263
Large Images Required	263
Even Larger Images Required	264
Internet Speed Concerns	264
Adaptive Image Solutions	266
Using Alternatives	266
Compressing Retina Images	267
Many-Image Solution	270
Picture Element	270
Srcset Attribute	272
Implementing the Picture Patch	274
In Conclusion	281

CHAPTER 14 Aligning Images 283

Aligning Images in Relation to the Text	284
Baseline	284
Text-Bottom	285
Text-Top	285
Top and Bottom	286
Middle	286
Centering Images in the Window	287
In the Background with CSS	287
Just the CSS	288
No Need to Hardcode Numbers, Thanks to jQuery	289
Stretching an Image Across a Browser Window	290
In Conclusion	291
Conclusion	292
Index	294

INTRODUCTION

In the beginning, print designers created web designs in Photoshop and exported them as one enormous image, declaring that a web site. It wasn't pretty.

Designers started to change their ways after realizing that text and multiple images can not only make designs with HTML and CSS, but make *great* designs. But a part of that process often meant designs would be at least 960 pixels wide or some fixed width.

With the increased adoption of mobile devices like smartphones and tablets that can present a rich web experience in portrait or landscape mode, the desktop browser window width is no longer the standard for which to design. This new mobile component to web design has led us to re-examine our best practices and adopt new techniques.

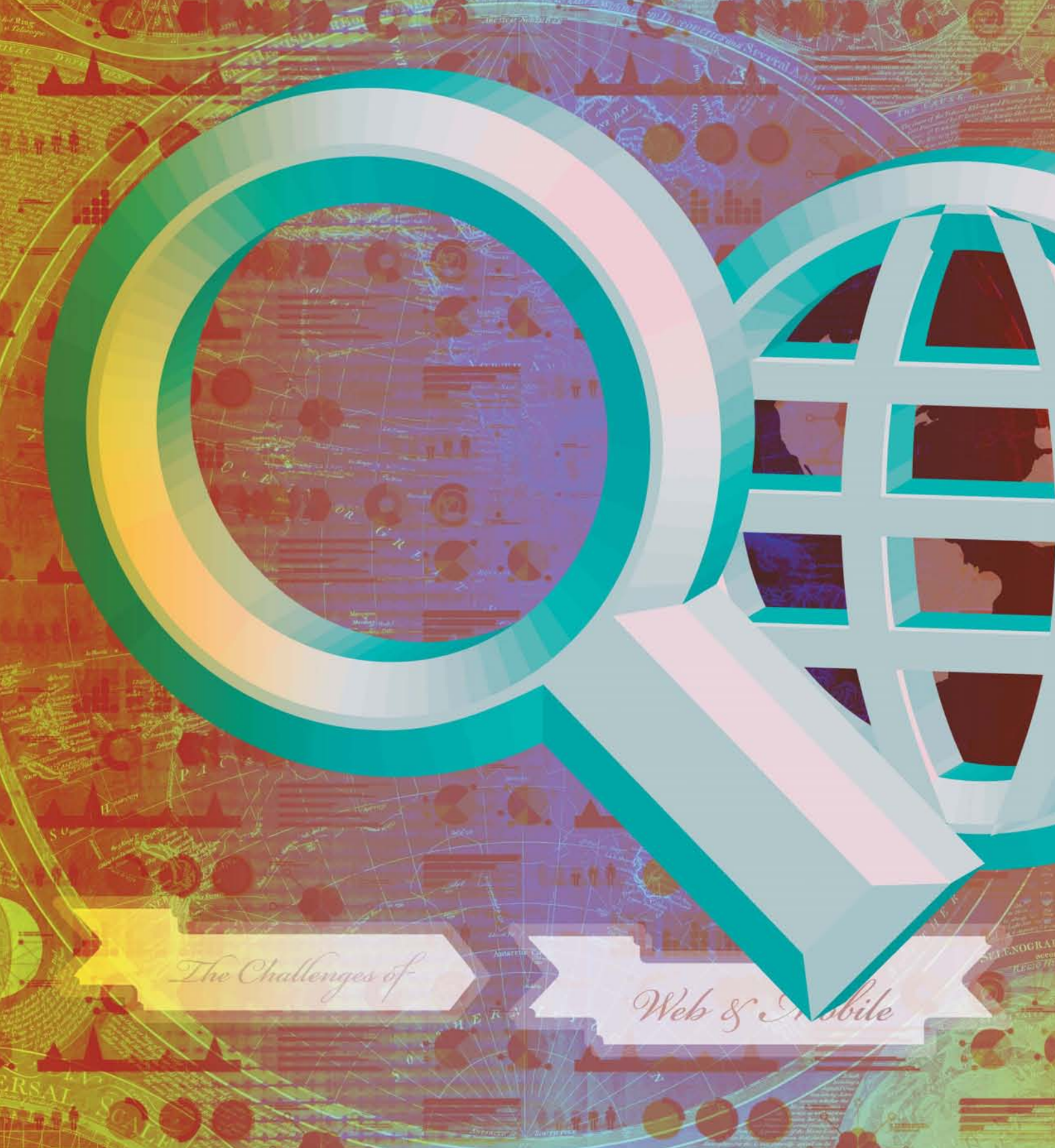
Designing Web and Mobile Graphics is intended to give the beginner or intermediate web designer a look into how to create and build up visuals that meet current web and mobile standards.

In this beautiful web, all these tasks can be done by one person. Each person like yourself, dear reader, has the power to be an independent content producer.

Designing Web & Mobile Graphics provides a foundation in HTML and CSS in the context of development. Building on quick successes from techniques and tips, we move from chapter to chapter to more advanced or unique web design solutions.


GET READY,... GO!

In web design, things are constantly changing. *Designing Web and Mobile Graphics* is the book intended to give you the foundation you need to work with images and much, much more.



The Challenges of

Web & Mobile



“I like the challenge of trying different things
and wondering whether it’s going to work
or whether I’m going to fall flat on my face.”

—Johnny Depp

Chapter 4

CHALLENGES IN WEB DESIGN

Developing a new version of a browser once took a year; now, browser companies update their products every couple of months. The World Wide Web Consortium (W3C), the web standards body, frequently updates its recommendations on how browsers should work. Users are no longer chained to the desktop; they can access the Web on their smartphones, game consoles, kiosks, and so on. The websites we build need to adapt to meet the challenges of these different browsers and environments.

THE WEB ENVIRONMENT

Browsers have come a long way to produce a great base experience for visitors, but you might be inadvertently creating a situation where visitors see a different presentation than the one you think you're giving them.

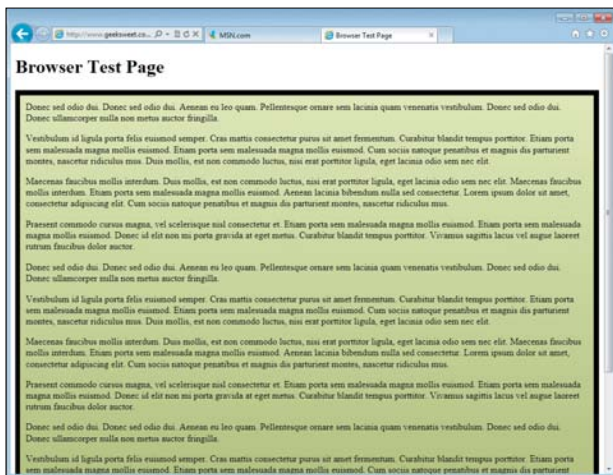


FIGURE 4.1 A test page in Internet Explorer 9.

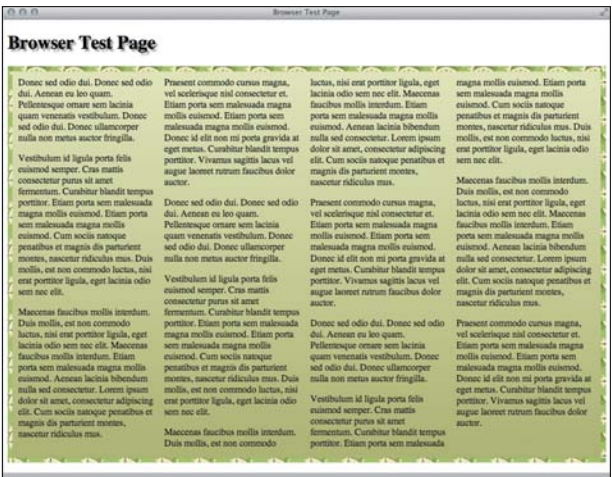


FIGURE 4.2 A test page in Safari 6.

Revealing Browser Issues

The following screenshots show how one page compares in different browsers: Internet Explorer 9 (FIGURE 4.1), Safari (FIGURE 4.2), Chrome (FIGURE 4.3), Firefox (FIGURE 4.4), Nexus 7 Chrome (FIGURE 4.5), Opera (FIGURE 4.6), and Mobile Safari iOS on an iPad (FIGURE 4.7).

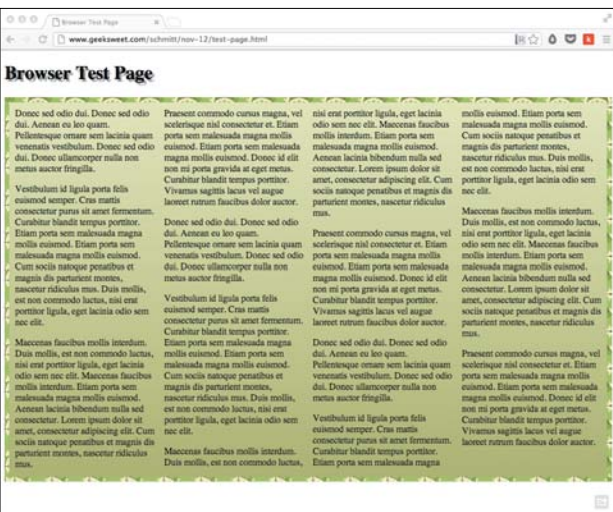


FIGURE 4.3 A test page in Chrome.

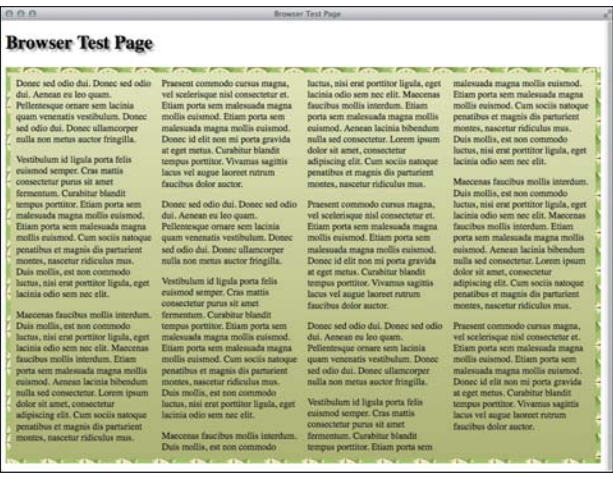


FIGURE 4.4 A test page in Firefox.



FIGURE 4.5 A test page in Nexus 7 Chrome.

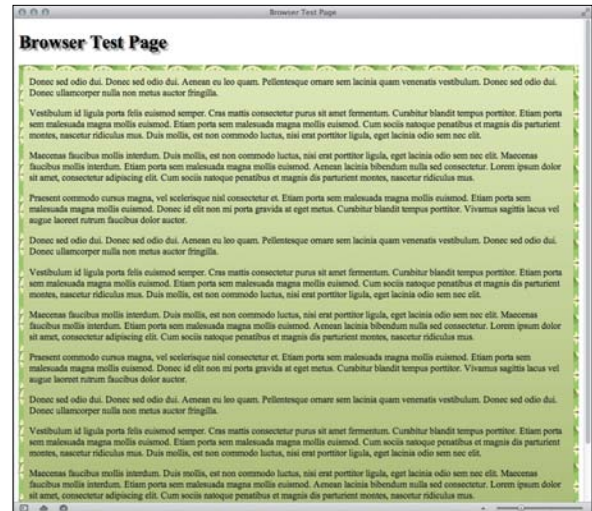


FIGURE 4.6 A test page in Opera.



FIGURE 4.7 A test page in Mobile Safari on an iPad.

TABLE 4.1 showcases which Cascading Style Sheets (CSS) features are supported by various browsers. A “Y” means the browser supports the CSS feature natively. An “N” means the browser does not support it. A “P” means the browser does support the feature, but it needs a custom CSS prefix in order for the feature to display. The custom prefixes will be explained in more detail in Vendor Prefixes later in this chapter.

TABLE 4.1 CSS Feature Support by Browser

Browser	Columns	Border Image	Gradients	Text Shadows
Internet Explorer 9	N	N	N	Y
Safari 6	P	Y	P	Y
Chrome	P	Y	P	Y
Firefox	P	Y	P	Y
Nexus 7 Chrome	P	P	P	Y
Opera	Y	P	P	Y
Mobile Safari iOS on iPad	P	P	P	Y

Using a Test Page

Now let’s break down the test page to show which features are in it and how they are implemented. The test page covers some basic and advanced CSS functionality:

- **CSS3 multi-column** lets you set text in columns. In web design, you don’t see multiple text columns unless the designer manually adjusts the number of words for each column or uses a JavaScript patch.

```
div {
  column-count: 4;
}
```

- **CSS3 border-image** lets you wrap an image around an HTML element. As the HTML stretches as text is added or removed, for example, the image stretches and adapts.

```
div {
  column-count: 4;
  border-image: url(border-img.png) 10px;
}
```

- **CSS3 gradients** sets color transitions in the background of elements.

```
div {
  background: linear-gradient(to bottom,
    rgba(30,87,153,1) 0%,
    rgba(41,137,216,1) 50%,
    rgba(32,124,202,1) 51%,
    rgba(125,185,232,1) 100%);
}
```

- **CSS3 text-shadows** lets you put one or more shadows on text.

```
div h1 {
  text-shadow: 0 1px 1px #bbb,
    0 2px 0 #999,
    0 3px 0 #888,
    0 4px 0 #777,
    0 5px 0 #666,
    0 6px 0 #555,
    0 7px 0 #444,
    0 8px 0 #333,
    0 9px 7px #302314;
}
```

Color

We perceive the color around us thanks to our eyes, which are electromagnetic spectrum detectors. Colors make up only a small portion of this spectrum, which encompasses x-rays, gamma rays, microwaves, radio waves, all the colors we see, and much more.

A computer screen is made of tiny dots, or pixels, arranged in a grid. These pixels change color depending on what the computer instructs the monitor to display

(**FIGURE 4.8**).



FIGURE 4.8 Zooming in closely on a raster image shows the blocks of color arranged in a grid like the grid of pixels that makes up the screen.

! STICK WITH RGB

When you see an option for CMYK in a digital imaging tool like InDesign or Photoshop, it's only there to help give a representation of what the colors look like when *printed*—not how they should appear on screen. Be sure to stick with RGB color mode when creating images or mocking up layouts.

The screens in our desktop PCs and mobile devices don't show colors the same way. No universal calibration system for on-screen color currently exists. Color on computer monitors can vary due to the display brand, the video card brand, the screen's age, the operating system, the amount of ambient light, colors appearing next to each other, and the age and condition of the viewer's eyes. A further translation happens in printing: some colors that are visible on a screen, where they're made with light, can't be printed with any kind of ink or toner (bright, pure blue is the classic example).

No matter how carefully you choose the colors on your screen, they'll never be absolutely accurate since there is no single standard for displaying them.

Screens and Pixels

Units of measurement are expected to be a constant. The problem with constants is that they change.

Take the humble meter, for example. How we determine the starting and ending point of the meter has altered over human history. In the 17th century, a meter was proposed to be part of the distance of the equator to the North Pole. One ten-millionth of the distance to be exact.

In 1875, a meter was then defined as the length of a platinum-iridium bar created by the International Bureau of Weights and Measures near Paris (see <http://museum.nist.gov/object.asp?ObjID=37>). Numerous bars of the same length were made and distributed around the world.

Some two hundred years later, the meter transformed again and is no longer tied to a physical object. A meter is now the distance *light* travels in a vacuum over 1/299,792,458th of a second. What will it be in the future? And don't ask me what happens to the length of a meter if you happen to be near a black hole, where light can't escape.

Defining a Pixel

The pixel has similar identity issues. What defines a pixel is a matter of when you ask the question. If you are a web designer working in the 1990s, the pixel would be about 1/96th of an inch and it would be fairly constant across operating systems and monitors. *Now* when you ask about the size of a pixel, the answer will depend on which kind of pixel you mean.

These days, people aren't "surfing the World Wide Web" with just the desktop anymore. The web can be accessed on screens that are 3.5 inches diagonal held close to the face, and on 60-inch TVs from across the room. To provide roughly the same experience on this wide range of devices, on-screen elements need to end up looking about the same whether they're being seen from 10 inches or 10 feet.

Let's say we set a head's font size to 24px. If we then look at that head on a retina display and a regular laptop, there should be no difference in size (**FIGURE 4.9**).

The W3C has recommended a standard visual angle pixel size that hardware and software manufacturers can refer to while developing their products. What this means is that web developers can use the CSS pixel as their unit of measurement, and let the browser and OS take care of mapping it to the device pixel, whatever its physical size may be. (There's just one hitch: so far, a perfect solution for scaling up photographic images hasn't been found, as we'll see in Chapter 13, "Images for Responsive Web Design.") The CSS pixel is an absolute length unit that is anchored to the reference pixel, which is an angular measurement (**TABLE 4.2**).

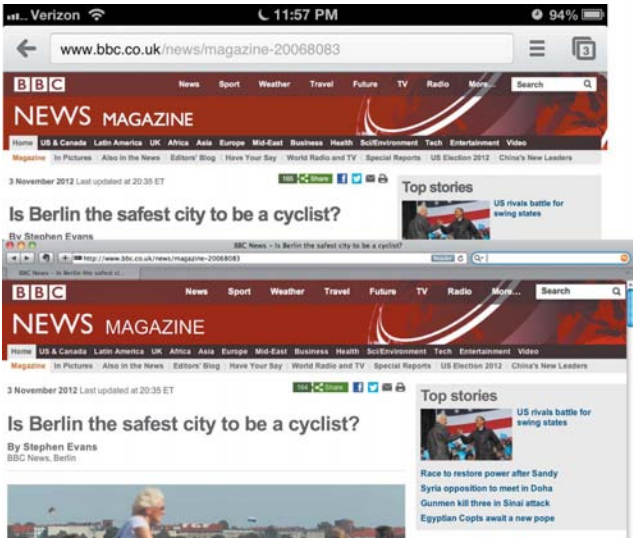


FIGURE 4.9 The same headline seen in Chrome on an iPhone 5 with a 4-inch, 1136 × 640-pixel screen and in Safari on a MacBook Pro with a 13-inch, 1280 × 800-pixel screen.

TABLE 4.2 The CSS Pixel and the Reference Pixel

	Type	Derived from	Used by
reference pixel	visual angle	physics of light	hardware+OS makers
CSS pixel	fixed+anchored	reference pixel	designers/developers

Accessibility

While monitors continue to improve in terms of color clarity, generating millions of colors as faithfully as possible, there are segments of the population who won't be able to see them. Seven percent of men cannot distinguish between red and green colors. Can you see the colors in **FIGURE 4.10**?

These deutan color vision deficiencies, along with others, must be taken into consideration when designing with color.

EYESIGHT STATISTICS

According to the World Health Organization, 285 million people worldwide are visually impaired. That includes 39 million blind and 246 million low-vision people.



FIGURE 4.10 A “normal” image compared to an image seen as a red-green color blind person sees it.

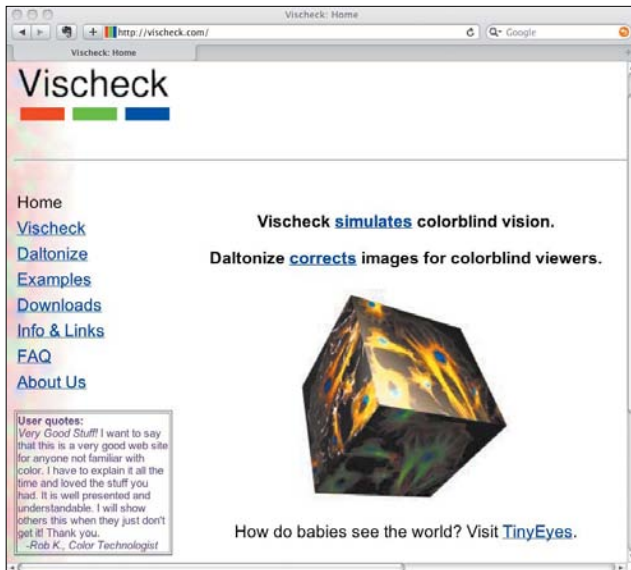


FIGURE 4.11 The Vischeck page.

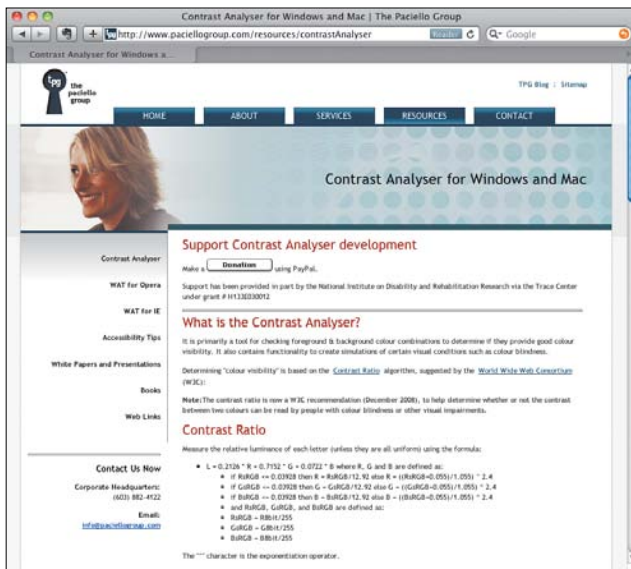


FIGURE 4.12 The Contrast Analyser.

Testing for Color Blindness

To check your design for color blindness issues, there are a couple of online tools.

- **Vischeck** (<http://vischeck.com/>) provides examples of color blindness and converts existing websites to showcase how they appear to people with different types of color blindness (FIGURE 4.11).
- **Contrast Analyser** (<http://www.paciellogroup.com/resources/contrastAnalyser>) uses the W3C's contrast ratio algorithm to determine whether colors have enough visibility or contrast and shows how colors look to people with different types of color blindness (FIGURE 4.12).

Color Vision Is Only One Part of Accessibility

Color blindness is just one accessibility issue. Designing for accessibility in general is another of the challenges of web design:

- People with mobility issues, such as those with carpal tunnel syndrome, may prefer to navigate via keyboard (<http://webaim.org/techniques/keyboard/>).
- People with reduced dexterity appreciate clickable areas that aren't too tiny, and forms that don't time out before they can finish filling them in (<http://otal.umd.edu/UUPractice/mobility/>).
- People with hearing impairments rely on captions or transcripts (<http://webaim.org/articles/auditory/>).
- People who are susceptible to photosensitive epileptic seizures want to avoid blinking elements (<http://www.w3.org/TR/understanding-WCAG20/seizure-does-not-violate.html>).
- People with learning disabilities that affect reading, such as dyslexia, can be helped by good typography, user-selectable type and background colors, and having content available in audio format (<http://www.bdadyslexia.org.uk/about-dyslexia/further-information/dyslexia-style-guide.html>); audio content also helps people with low vision.

- People with cognitive disabilities affecting memory and attention do better with simple navigation and no distracting animation (<http://www.ncdae.org/resources/articles/cognitive/>).
- Best practices include the use of **alt** attributes, unambiguous copy, **title** attributes for links, proper HTML structure, and graceful degradation (<http://webaim.org/techniques/alttext/>).

DETERMINING *YOUR* CROSS-BROWSER GOALS

To determine how to handle different browser issues, you need to understand your visitors. To do that, you need to look at your site statistics.

Analyzing Your Site Traffic

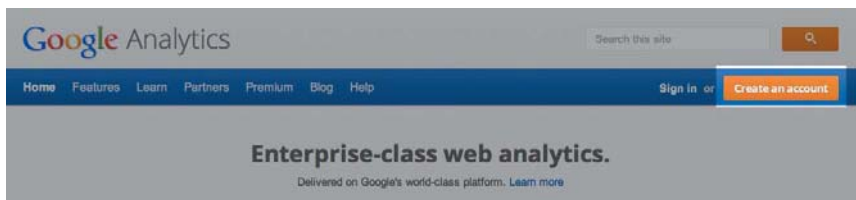
One of the most popular methods for tracking site statistics is Google Analytics (GA), which collects information about site visitors' behavior.

Setting Up Google Analytics

If you already have a Google account (like one for Gmail), use it to set up a Google Analytics account.

TO SET UP GOOGLE ANALYTICS, FOLLOW THESE 6 STEPS:

- 1 If you already have a Google account, go to Google Analytics at <http://www.google.com/analytics/> and click the Create an account button, and then enter your Email and Password and click the Sign In button.



GOOGLE ANALYTICS IS FREE

While there are two available versions, free and premium, most of the time the free version is all you need.

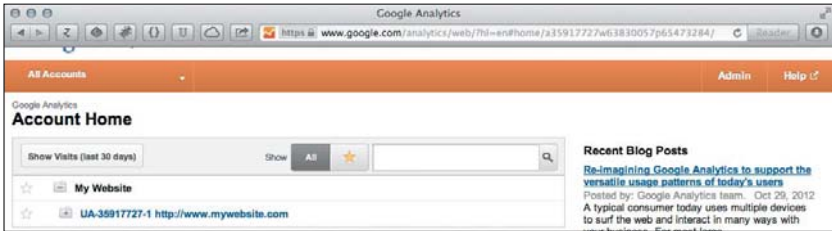
SIGNING UP FOR A GOOGLE ACCOUNT

Don't have a Google account? It's free and opens up a lot of other free resources, like email, web alerts, online word processing and spreadsheets, and more. Sign up at <https://accounts.google.com/NewAccount>.

- 2 Fill out the new account creation form with details about your account name, site URL, industry category, and time zone. Also, review the Terms of Service for the account.

The screenshot shows the Google Analytics account creation form. The browser address bar displays the URL: <https://www.google.com/analytics/web/?hl=en&management/Accounts/a34573408w62134486p63691629/k3facc>. The page title is "Google Analytics". The main heading is "What would you like to track?". There are two buttons: "Web Site" and "App". Below the buttons, there are two columns of text: "Track web sites whose HTML you control" and "Track interactions within Android and iOS apps". The "Web Site" button is selected. The "Setting up your web property" section contains the following fields: "Website Name" (text input with "My Website" entered), "Web Site URL" (text input with "http:// www.mywebsite.com" entered), "Industry Category" (dropdown menu with "Media and Entertainment" selected), and "Reporting Time Zone" (dropdown menu with "United States" selected). The "Reporting Time Zone" dropdown also shows "(GMT-05:00) Eastern Time".

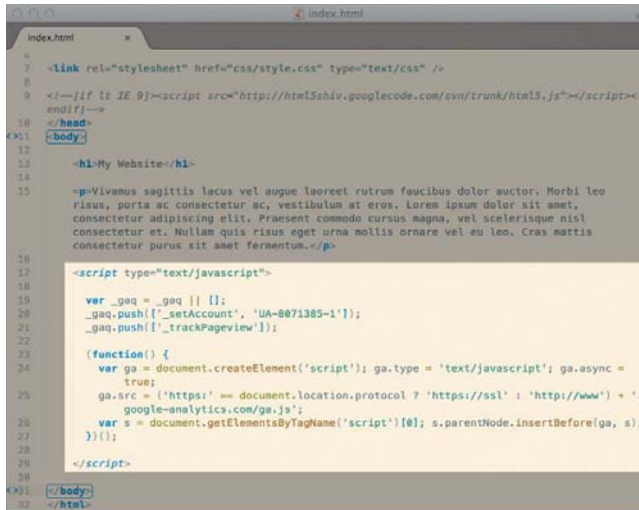
- 3 At this main page of Google Analytics, you see a listing of the site you've entered. Click Tracking Code from the navigation menu.



- 4 Select the domain name you're using. If you're hosting your own domain name, select "A single domain."

The screenshot shows the Google Analytics "Website Tracking" setup page. The browser address bar displays the URL: <https://www.google.com/analytics/web/?hl=en&home/a35917727w63830057p65473284/>. The page title is "Google Analytics". The main heading is "Website Tracking". The "Property Name" is "My Website". The "Website URL" is "http://www.mywebsite.com". The "Tracking Status" is "Tracking Not Installed". The "Standard" tab is selected. The "1. What are you tracking?" section has two radio buttons: "A single domain" (selected) and "One domain with multiple subdomains". The "A single domain" radio button has a tooltip that says "Example: www.mywebsite.com". The "One domain with multiple subdomains" radio button has a tooltip that says "Examples: www.mywebsite.com, apps.mywebsite.com, store.mywebsite.com". There are also checkboxes for "AdWords campaigns" and "DoubleClick data".

- 5 Copy and paste the code from Google's form field in every page of your site above the closing **body** tag.

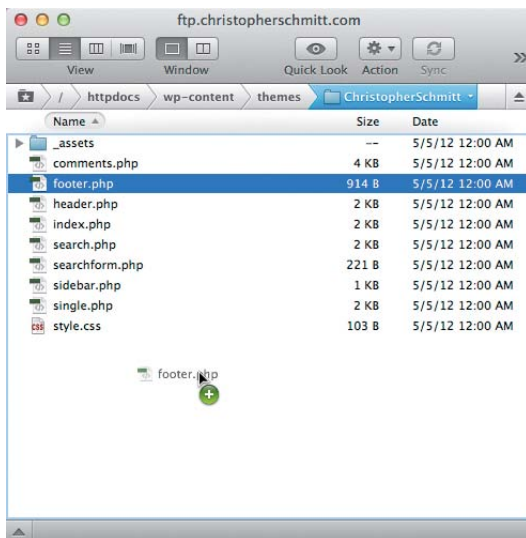


```

1  index.html
2
3  <!--[if lt IE 9]><script src="http://html5shiv.googlecode.com/svn/trunk/html5.js"></script></if>-->
4
5  </head>
6  <body>
7
8      <h1>My Website</h1>
9
10     <p>Vivamus sagittis lacus vel augue laoreet rutrum faucibus dolor auctor. Morbi leo
11     risus, porta ac consectetur ac, vestibulum at eros. Lorem ipsum dolor sit amet,
12     consectetur adipiscing elit. Praesent commodo cursus magna, vel scelerisque nisl
13     consectetur et. Nullam quis risus eget urna mollis ornare vel eu leo. Cras mattis
14     consectetur purus sit amet fermentum.</p>
15
16     <script type="text/javascript">
17
18         var _gaq = _gaq || [];
19         _gaq.push(['_setAccount', 'UA-8071385-1']);
20         _gaq.push(['_trackPageview']);
21
22         (function() {
23             var ga = document.createElement('script'); ga.type = 'text/javascript'; ga.async =
24             true;
25             ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') +
26             '.google-analytics.com/ga.js';
27             var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s);
28         })();
29     </script>
30
31 </body>
32 </html>

```

- 6 Upload the new pages to the server.



Once the code has been added to your site, the next step is to wait. It takes anywhere from four hours to a couple of days for Google to start publishing data to review.

! SERVING USERS, NOT OURSELVES

By placing the JavaScript code at the bottom of the document, the browser gets to it later on and focuses on the other elements of the page instead.

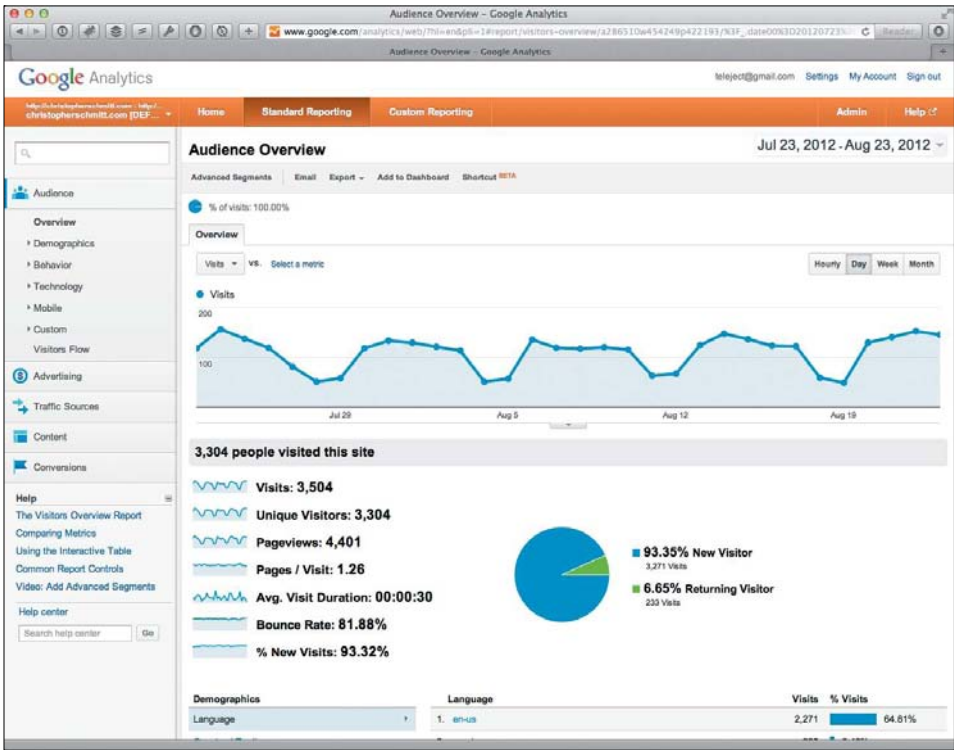
While instructions from Google Analytics state to place the code in the head element, this slows the rendering of a web page ever so slightly. Ensuring that your pages render as quickly as possible is more important than finding out how visitors are using the site. If pages render quickly, then visitors are more inclined to surf more of your site.

Finding Browser Statistics

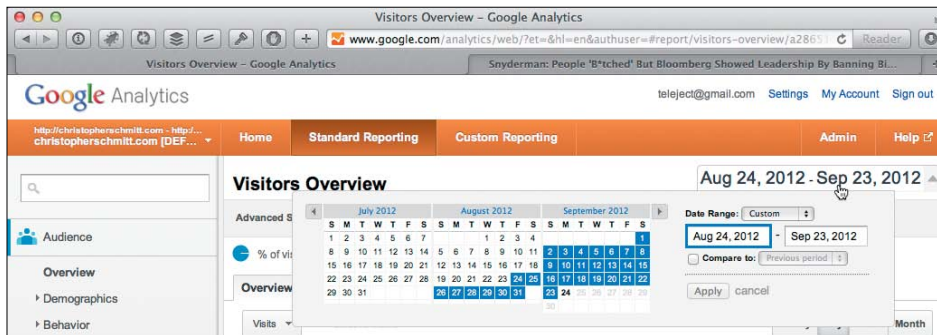
Finding browser statistics about your site's visitors is pretty straightforward. You can find information such as the types of browser your visitors use, and how often they use each browser both in number and percentage. You can even find out these details for specific browser versions.

TO FIND BROWSER INFORMATION ON YOUR SITE'S VISITORS, JUST FOLLOW THESE 5 STEPS:

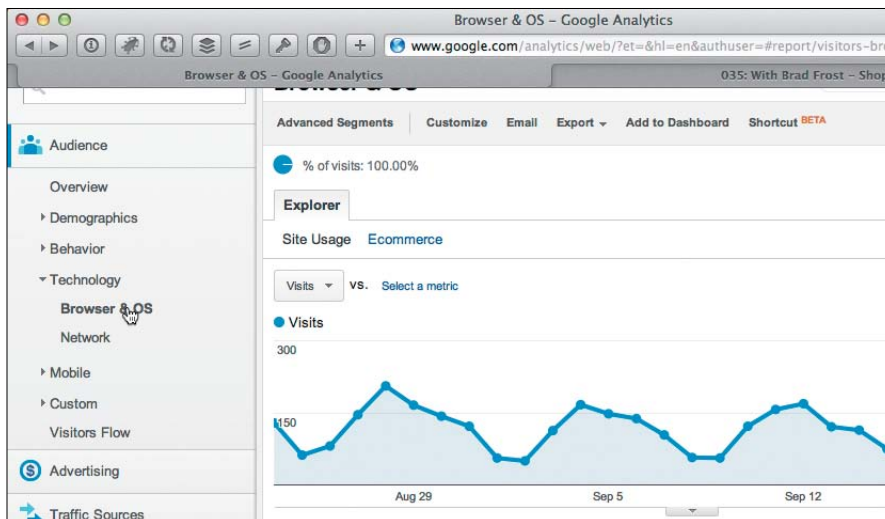
- 1 After logging into Google Analytics, you see the Visitors Overview page.



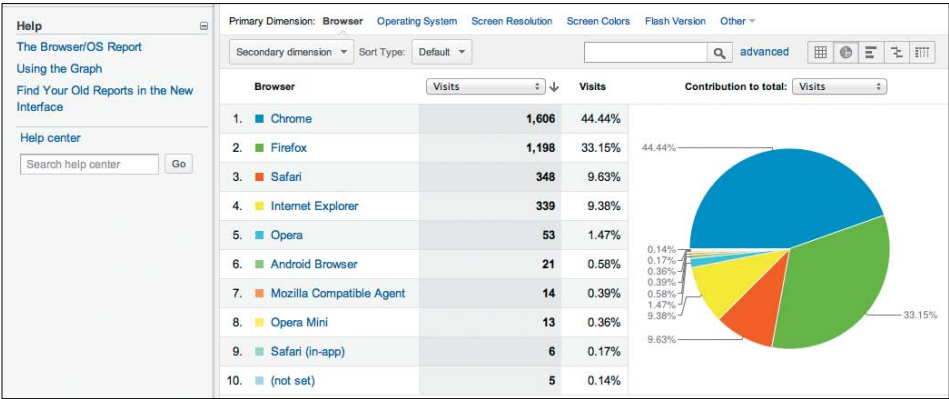
- 2 Click on the large date text in the upper right corner to select the date range you want to review.



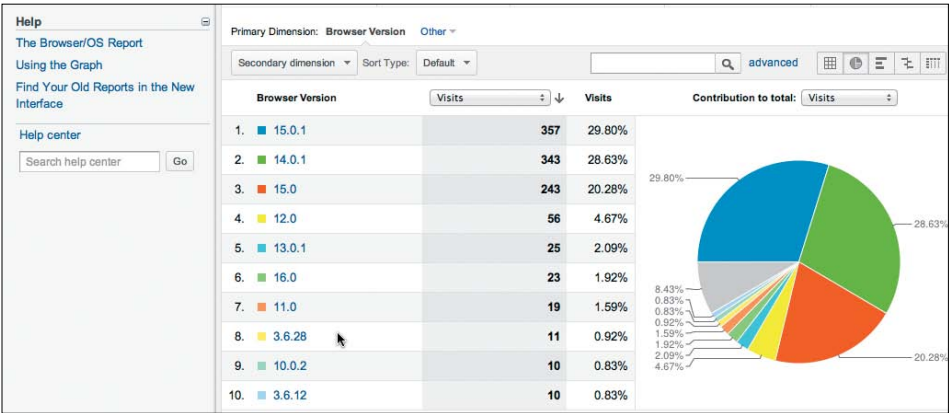
- 3 In the left column, select Technology > Browser & OS.



4 The different browsers are listed on the page and color-coded to a pie chart.



5 For a breakdown on the different versions of a browser visiting a site, click on the browser name. The example shown here is the breakdown of the versions of Firefox visiting the site.

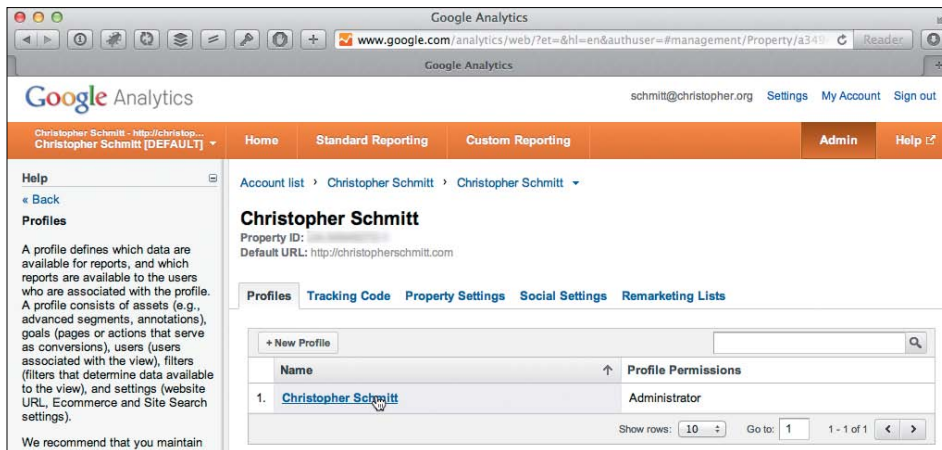


Updating Google Analytics Settings

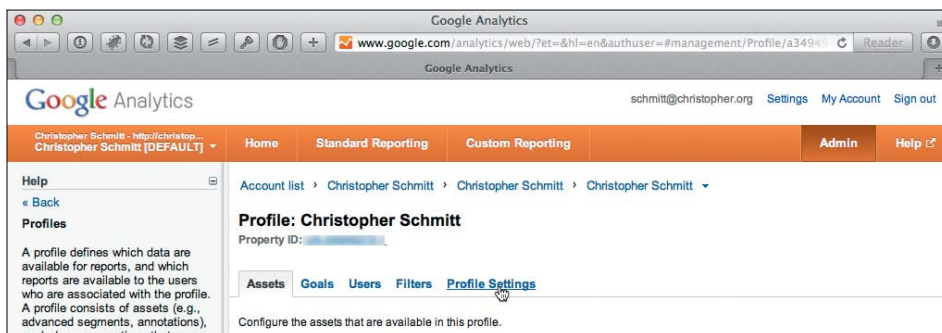
Going back to your profile to keep your site information and goals updated ensures that Google Analytics is gathering the right data to give you an accurate picture of your site usage.

TO UPDATE SITE INFORMATION AS IT CHANGES, FOLLOW THESE 4 STEPS:

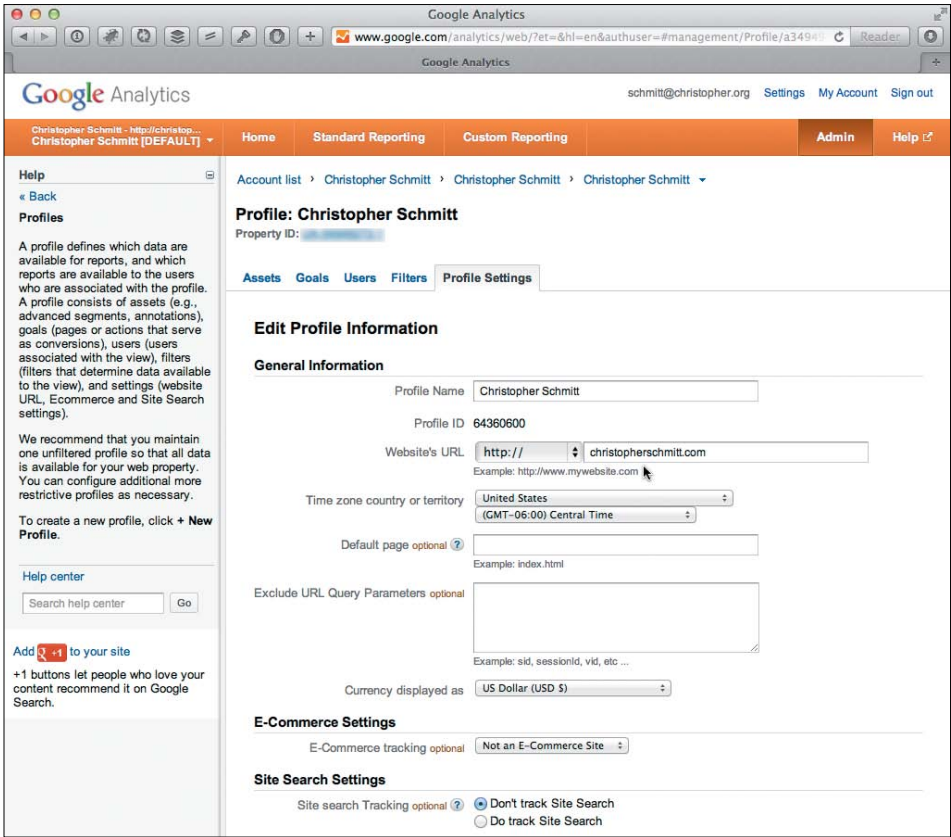
- 1 Click on the Profile name.



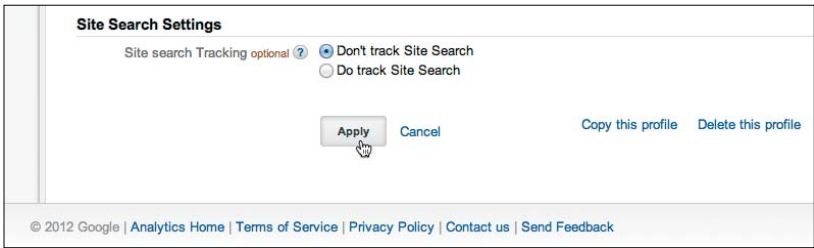
- 2 Select Profile Settings from the navigation submenu.



3 Make updates to the information.



4 Select Apply.



DEVELOPING YOUR SITE FOR DIFFERENT DEVICES

As we have found out already, browsers have varied support for CSS features. Another difference in browsers is in how they display HTML text and white space that doesn't have any CSS customization. The underlying browser styles can influence the CSS presentation layer, so we try to take the browser styles back to a basic and common foundation.

Resetting and Normalizing Browser Styles

Browsers have their own internal style sheets, which designers use to render basic default styles for everything from the distance between lines of text to link colors (FIGURE 4.13).

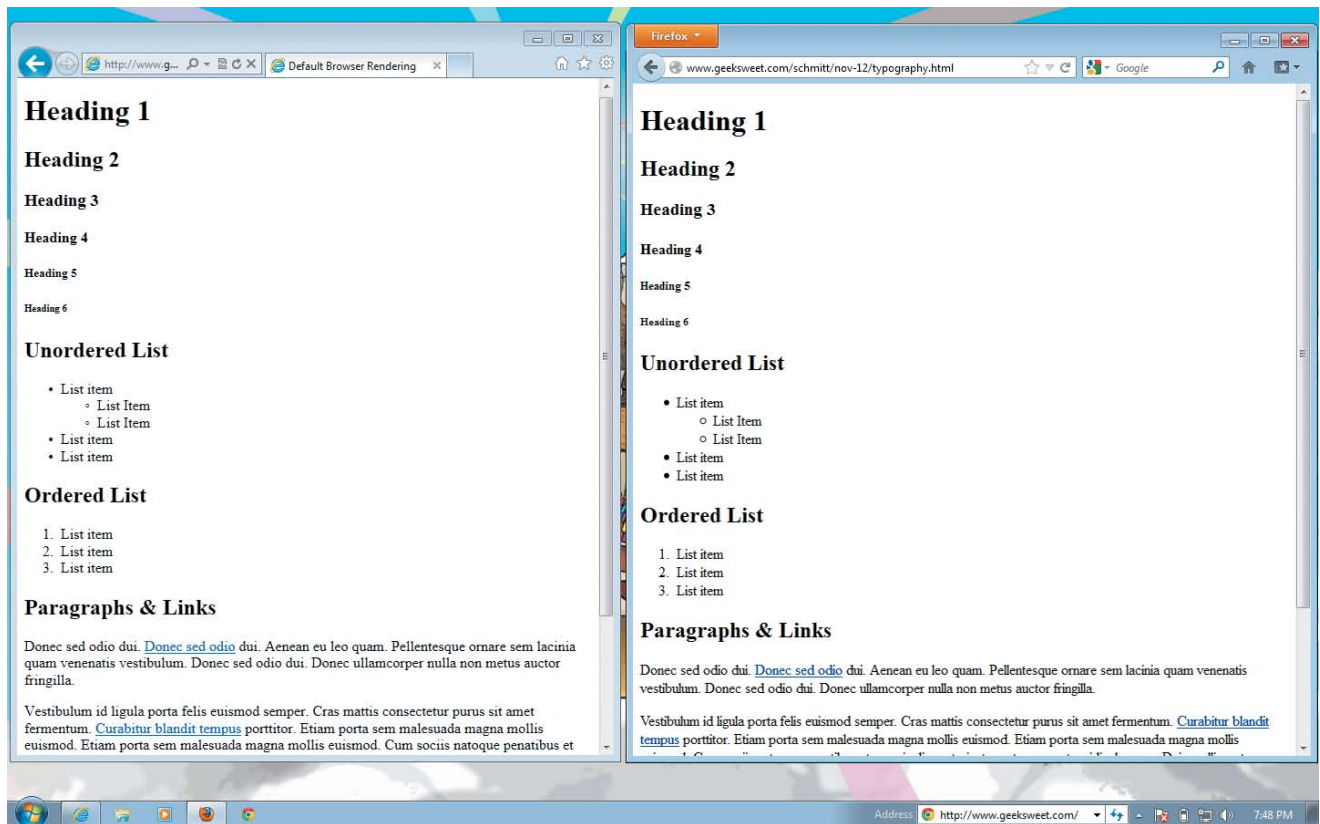


FIGURE 4.13 Differences in Internet Explorer 9 vs. Firefox browser rendering on Windows.

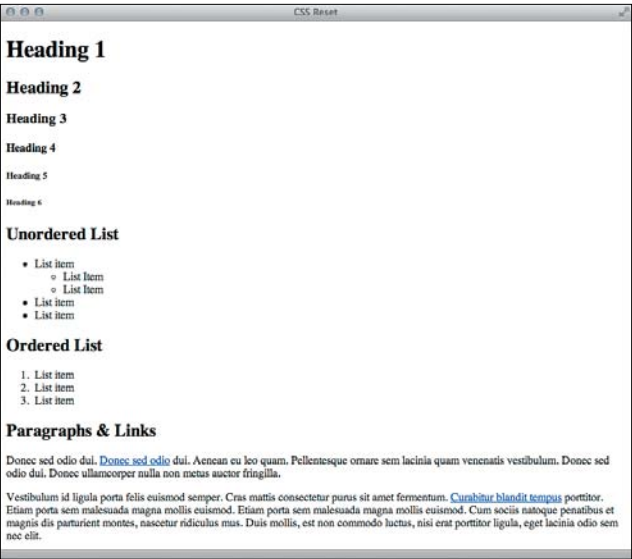


FIGURE 4.14 Before the CSS Reset.

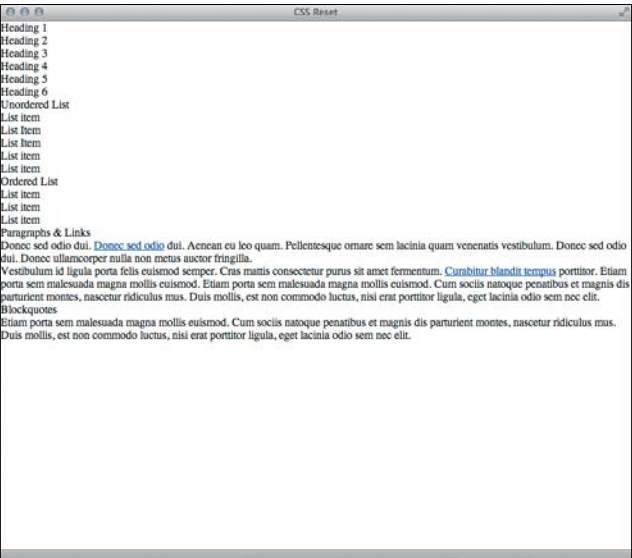


FIGURE 4.15 After the CSS Reset.

Resetting Styles

One way to address browser inconsistencies is to remove or set all CSS properties to zero. This is done through a CSS style sheet known as CSS Reset, like the one premade from Yahoo! through its YUI Library. To set up a reset for a web page (FIGURE 4.14), place a link element at the top of any other reference to the style sheet (FIGURE 4.15):

```
<link rel="stylesheet" type="text/css"
href="http://yui.yahooapis.com/3.7.1/build/
cssreset/cssreset-min.css">
<link rel="stylesheet" type="text/css"
href="style.css">
```

Normalizing Styles

Using the CSS Reset approach means that all default settings are dialed back. What's left is an empty canvas with no hint of design or standards. That's where **normalize.css** steps in. Rather than removing everything, Normalize.css (<http://necolas.github.com/normalize.css/>) creates a cohesive standard for the default rendering of HTML elements (FIGURE 4.16).

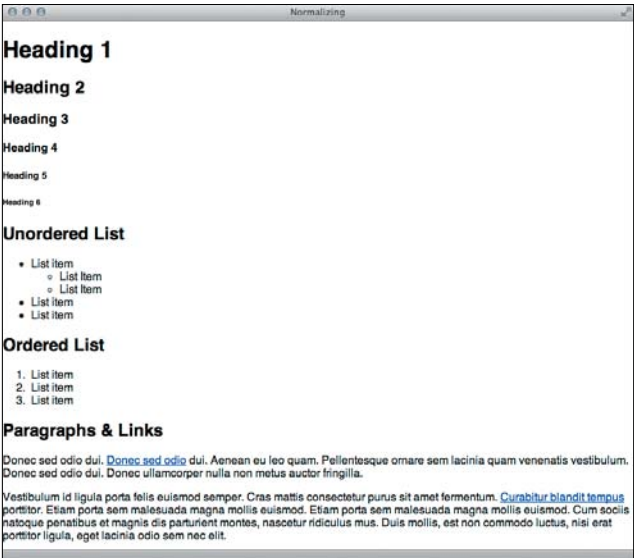


FIGURE 4.16 After normalize.css.

Normalizing style sheets gives you a solid base to build web pages without having to worry about little discrepancies in browser renderings.

Vendor Prefixes

Sometimes a new CSS feature brought into a browser is still in its beginning stages and needs more work. When this happens, the browser vendor usually provides what’s called a **prefix CSS property**.

The Simple CSS Workaround

For example, a simple CSS gradient like the following code will be ignored in Safari 6:

```
div {
  background-image: linear-gradient(#fff, #000);
}
```

To get Safari to work, we have to add a line of code that only Safari will recognize:

```
div {
  background-image: -webkit-linear-gradient(#fff, #000);
  background-image: linear-gradient(#fff, #000);
}
```

The software that powers the rendering or display of a page in Safari is known as WebKit. Therefore, to specify a special feature like a gradient, we need to add “webkit” surrounded by hyphens. To support other browsers that have the same approach to new CSS features, we need to add the special vendor prefix for each of them (**TABLE 4.3**):

```
div {
  background-image: -webkit-linear-gradient(#fff, #000);
  background-image: -moz-linear-gradient(#fff, #000);
  background-image: -ms-linear-gradient(#fff, #000);
  background-image: -o-linear-gradient(#fff, #000);
  background-image: linear-gradient(#fff, #000);
}
```

TABLE 4.3 WebKit Vendor Prefixes

Browser	Vendor Prefix
Safari	-webkit-
Firefox	-moz-
Internet Explorer	-ms-
Opera	-o-

WHY DO BROWSERS HAVE UNTESTED FEATURES?

Sometimes browser vendors want to gain an advantage over their competitors, so they put out a new feature that the others don't yet have. This scenario played out over and over again in what's been called the Browser Wars. Other times, browser vendors want to see if a new feature will be adopted by web developers, or they might feel that a feature is worthwhile to implement without consulting other interested parties.

The group that writes the CSS standards will not approve a CSS property that begins with a hyphen. This enables browser vendors to create vendor-prefixes that start with hyphens for the testing of features. For the specification, check out <http://www.w3.org/TR/CSS21/syndata.html#vendor-keywords>.

Automatic Vendor Prefixing

For one CSS feature in our text shadow example, five additional lines of code were needed to support browsers. If we need to add additional lines of code for each browser for each new CSS feature, the lines of code we generate would quickly get out of control. Web developer Lea Verou realized the vendor prefixes situation and developed a piece of JavaScript called -prefix-free (<http://leaverou.github.com/prefixfree/>), shown in

FIGURE 4.17:



FIGURE 4.17 The -prefix-free homepage.

FIGURE 4.18 Validator reporting errors.

W3C®

Unicorn - W3C's Unified Validator

Improve the quality of the Web

This document has not passed the test: W3C HTML Validator

Info (2)

URI: <http://www.google.com/>

Using experimental feature: HTML5 Conformance Checker

The validator checked your document with an experimental feature: *HTML5 Conformance Checker*. This feature has been made available for your convenience, but be aware that it may be unreliable, or not perfectly up to date with the latest development of some cutting-edge technologies. If you find any issues with this feature, please [report them](#). Thank you.

No Character encoding declared at document level

No character encoding information was found within the document, either in an HTML `meta` element or an XML declaration. It is often recommended to declare the character encoding in the document itself, especially if there is a chance that the document will be read from or saved to disk, CD, etc. See [this tutorial on character encoding](#) for techniques and explanations.

Errors (24)

URI: <http://www.google.com/>

8

2076

...<head><body dir="ltr" bgcolor="#fff"><script>(function(){var src="/images/sr...

The bgcolor attribute on the body element is obsolete. Use CSS instead.

11

99

...area><div id="mngb"><div id=gba><nobr><b class=gb1>Search<a class=gb1 hr...

Element nobr not allowed as child of element div in this context. (Suppressing further errors from this subtree.)

11

176

...ref="http://www.google.com/imghp?hl=en & tab=w">Images<a class=gb1 href="h...

& did not start a character reference. (& probably should have been escaped as &.)

11

245

...1 href="http://video.google.com/vhien & tab=w">Videos<a class=gb1 href="h...

& did not start a character reference. (& probably should have been escaped as &.)

11

317

...ref="http://maps.google.com/maps?hl=en & tab=w">Maps<a class=gb1 href="htt...

& did not start a character reference. (& probably should have been escaped as &.)

11

388

...ef="http://news.google.com/vnwhp?hl=en & tab=w">News<a class=gb1 href="htt...

& did not start a character reference. (& probably should have been escaped as &.)

More than likely your pages won't validate the first time you code them—it's surprisingly easy to forget a tag or a quote. Luckily the validators are specific as to what the error is and where it's located (**FIGURE 4.18**):

At first, validating your code may seem impossible, but it becomes easier as you get used to what clean, syntactical code looks like. The W3C has a markup validation service at <http://validator.w3.org/unicorn/>. Simply upload your document or provide its address, and the validator quickly gives you a report on both HTML and CSS errors. Working through the validator can be a little puzzling at times because of the error messages, but once you clean up any errors you get a clean bill of health (**FIGURE 4.19**).

Another valuable tool is HTML Tidy (<http://infohound.net/tidy/>). This tool actually fixes badly formed markup: it adds closing tags, changes mismatched tags, adds quotes to attribute values, and properly nests tags that are not nested (**FIGURE 4.20**). Stand-alone tidy applications are available for various platforms, and there are online versions as well. HTML Tidy also comes with many HTML and text editors.

BUILT-IN VALIDATORS WYSIWYG editors like Dreamweaver come with their own built-in validators.

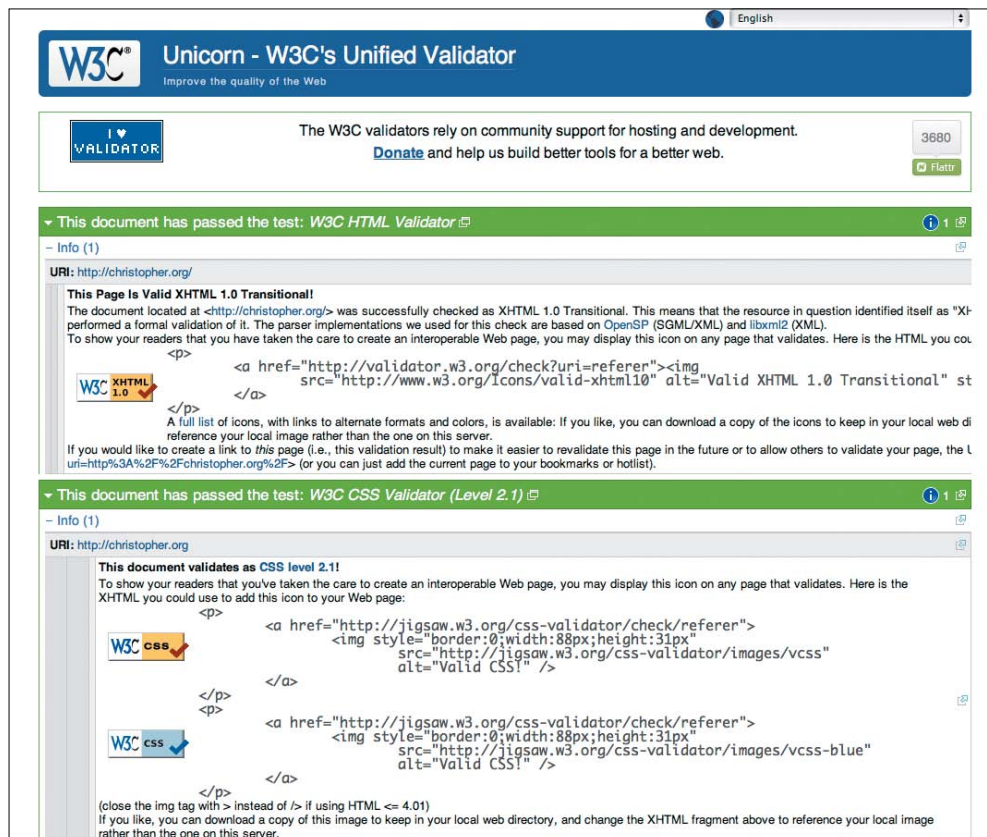


FIGURE 4.19 Gaining the coveted seal of approval from a validator.

FIGURE 4.20 HTML Tidy makes your code cleanly spaced and balanced.

HTML Tidy is a tool for checking and cleaning up HTML source files. It is especially useful for finding and correcting errors in deeply nested HTML, or for making grotesque code legible once more.

This online version enables you use it without installing the client tool on your PC. More information about HTML Tidy is available from the [original W3C page](#), and you can download a local copy of it from the [SourceForge project page](#).

Input options: either enter a URL, paste in some HTML, or upload a HTML file.

URL:

HTML:

Upload: no file selected

Tidy!

Tidy settings: define HTML/XHTML, pretty-print, and encoding settings. [>> Advanced](#)

HTML / XHTML	Pretty printing	Encoding
<input type="checkbox"/> Clean <input type="text" value="auto"/> Doctype <input checked="" type="checkbox"/> Drop empty paras <input checked="" type="checkbox"/> Logical emphasis <input checked="" type="checkbox"/> Output XHTML <input type="checkbox"/> Output XML <input type="checkbox"/> Replace color <input type="checkbox"/> Uppercase attributes <input type="checkbox"/> Uppercase tags <input type="checkbox"/> Word 2000	<input type="text" value="auto"/> Indent <input type="checkbox"/> Indent attributes <input type="text" value="2"/> Indent spaces <input type="text" value="90"/> Wrap <input type="checkbox"/> Wrap attributes	<input type="text" value="ascii"/> Char encoding <input type="text" value="Warnings"/> Output

HTML Tidy Online © 2002 Jonathan Hedley. Feedback appreciated.
 Also by author: [jsoup: Java HTML parser](#) | [Website speed optimization](#) | [Unicode Lookup](#) | [Color Schemer](#) | [AlterSlash](#) | [InfoHound](#) | [World Buddy](#)

Testing

When designing for the web, it's important to test against as many browsers and devices as possible. It can be difficult to maintain a number of desktop browser installations—both new and older—and get your hands on a wide range of devices to create a full mobile testing suite. Here are some suggestions to make cross-device development a little less complex.

Software

Use services like BrowserStack (<http://www.browserstack.com/>) to quickly test different platforms, browsers, resolutions, and connection speeds (**FIGURE 4.21**).

If you own a Mac, buy a software package like Parallels (<http://www.parallels.com/products/desktop/>) or VMware Fusion (<http://www.vmware.com/products/fusion/overview.html>). You can download Oracle VirtualBox (<https://www.virtualbox.org/>) for free (**FIGURE 4.22**). These software packages allow for **virtualization**—running Windows and Windows-based browsers on a Mac. You'll still need to purchase a Windows OS license, but you won't have to buy a separate machine.

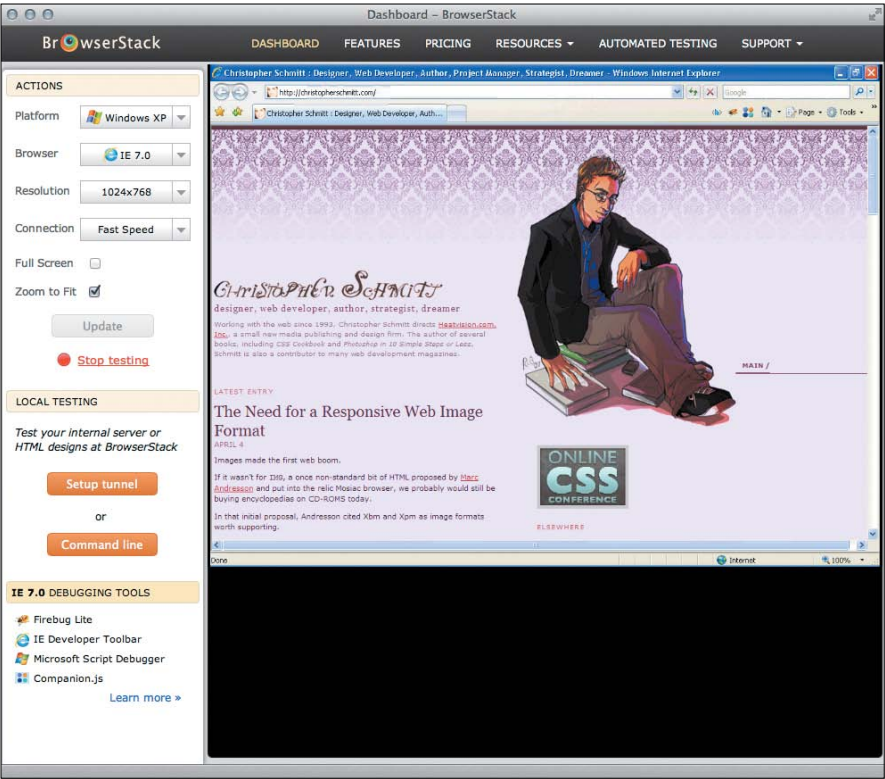


FIGURE 4.21 Using the BrowserStack interface to check out a page design.



FIGURE 4.22 Oracle VirtualBox.

Hardware

After creating a test page, upload it to the web and then take a breather. Go to your local computer store and check out your web page design in the store's display models. Check in on cell phone stores, too, to see how your page looks in their mobile devices.

If you can, buy a smartphone on eBay or ask friends and colleagues for their old phones and devices when they upgrade. You might be surprised how many people have old devices hanging around in a junk drawer. When *you* upgrade to a new device, keep your old one around for testing to create your own mini testing lab. Use services like Adobe Edge Inspect to test web pages on different devices. MobileTh.at is a new service, just being coded at the time of writing, that promises to simplify mobile testing.

IN CONCLUSION

Browsers have different levels of support for CSS features, as well as different interpretations of how to render HTML. By using tools like `normalize.css` and validating on multiple devices, we can address many of the challenges of web design. However, there's no substitute for knowing and learning about your audience and working to build them the best site possible. In the next chapter, we'll look at one of the most powerful parts of graphics and design: color.

This page intentionally left blank

INDEX

3D text shadow, 178
8-bit PNGs, 172
16 x 16px graphics, 185
24-bit PNGs, 173
960 Grid System, 245–246
@font-face property, 59
@import rule, 64

A

a element, 16
absolute links, 17, 20
absolute positioning, 40–41, 42, 288
accessibility issues, 77–79
Actions, Photoshop, 160
adaptive design
 images and, 262–281
 responsive design vs., 249
adaptive images, 262–281
additive primary colors, 105
adjacent sibling selectors, 26, 27
Adobe Edge Inspect, 94
Adobe Edge Web Fonts, 65
Adobe Fireworks, 158, 219–222
Adobe Illustrator. *See* Illustrator
Adobe Kuler, 113, 115
Adobe Photoshop. *See* Photoshop
Adobe Typekit, 66–69
Akamai, 163
aligning images, 283–291
 centering in the window, 287–289
 in relation to the text, 284–286
 stretching across the window,
 290–291
 tools in development for, 284
aligning text, 54
alpha channels, 172
analogous color schemes, 107

anchor tag, 16
Andreessen, Marc, 127, 128
Appearance palette, Illustrator, 148
area element, 216, 217
aria-hidden attribute, 208, 210
artboards, setting up, 151–152
Artboards palette, Illustrator, 148
artifacts, 134, 291
Atkins, Tad, 284
attribute selectors, 26, 27
attributes, 16
 class, 25, 36
 title, 17

B

b element, 15
background color
 changing, 99
 transparency of, 177
background-color property, 99
backgrounds
 blurring, 134
 images set as, 262, 287
background-size property, 290
banded color compression, 136–137
Banksy, 227
base color, 108–110
baseline alignment, 284
batch processing, 160
bit depth, 128–130
 explained, 128–129
 file size and, 130
 retro games and, 132
 table of, 130
bloat, code, 246
block elements, 31, 41
blockquote element, 14, 27

Blueprint framework, 242–245
blurring backgrounds, 134
body element, 11, 12, 100
book resources, 6
bookmarks
 favicons in, 183
 mobile, 190–193
border-color property, 100
border-radius property, 174
borders
 color, 100
 image, 74, 102
box shadows, 178–179
brightness, 104
Brody, Neville, 23
browsers. *See* web browsers
BrowserStack service, 92, 93
bullet markers, 196–197
Bulletproof Web Design
 (Cederholm), 6

C

calibrating colors, 117
Cascading Style Sheets. *See* CSS
centering images, 287–289
child selectors, 26, 27
circle bullets, 196
class selectors, 25, 26, 68
clear property, 229, 234, 235
clickable icons, 209–210
CMYK colors, 76, 105
code bloat, 246
Cohen-Or, Daniel, 112
color, 97–125
 base/central, 108–110
 bit depth and, 128–130
 border, 100, 102

- calibrating, 117
 - coding for the web, 98–103
 - combinations of, 106–107
 - CSS chart for, 117–124
 - cultural meanings of, 111
 - deficiencies seeing, 77–79
 - hexadecimal notation for, 98
 - inspiration for, 112–115
 - link, 100–101
 - opacity of, 103
 - overriding HTML, 99–101
 - primary systems of, 105
 - properties of, 104
 - screen display of, 75–76
 - traits associated with, 108
 - transparency of, 103, 177
 - color blindness, 78
 - color harmonies, 112
 - color names, 117
 - color** property, 100
 - color schemes, 97
 - images as basis for, 116
 - techniques for building, 108–116
 - types of, 107
 - color spectrum, 104
 - ColorCombos website, 113, 114
 - ColorMunki calibration tools, 117
 - COLOURlovers website, 113, 115
 - column features, 74
 - comments, conditional, 37–38
 - commercial font services, 66–69
 - complementary color schemes, 107
 - compression
 - artifacts from, 134
 - GIF image, 133
 - HTTP, 162–165
 - JPEG image, 134
 - lossless, 133, 135, 144
 - lossy, 134
 - PNG image, 135
 - raster image, 160–161
 - recommended rate of, 135
 - responsive design and, 267–270
 - SVG image, 144
 - compression charts, 136–143
 - banded color, 136–137
 - illustration, 142–143
 - pattern, 140–141
 - photograph, 138–139
 - computer monitors
 - calibrating colors on, 117
 - document size settings for, 149
 - scaling images for, 263
 - testing websites on, 94
 - conditional comments, 37–38
 - contain** value, 290
 - containing boxes, 40–41
 - content** property, 248
 - contrast, 104
 - Contrast Analyser tool, 78
 - convertico.com website, 185
 - copying code examples, 6
 - counter-increment** property, 201
 - counters, list, 201
 - cover** value, 290
 - Coyier, Chris, 257, 258, 264, 274, 292, 293
 - Creative Commons, 113
 - CSS (Cascading Style Sheets), 23–43
 - absolute positioning, 40–41
 - block elements, 31
 - browser support, 24, 74
 - color chart, 117–124
 - declarations, 25
 - delivering to IE, 36–38
 - fixed positioning, 39
 - formatting pages with, 28–35
 - frameworks based on, 241–246, 259
 - HTML documents and, 24
 - inline elements, 31
 - internal styles, 30
 - linking to, 28–29
 - list design, 196–197, 203–205
 - normalizing, 88–89
 - print-optimized, 248
 - pseudo-classes, 33–35
 - pseudo-elements, 33–35
 - relative positioning, 39
 - resetting, 88
 - selectors, 25–27
 - static positioning, 38–39
 - transparency and, 174–179
 - typography and, 46
 - validating, 90–92
 - CSS frameworks, 241–246
 - 960 Grid System, 245–246
 - advantages/disadvantages of, 246
 - Blueprint framework, 242–245
 - explanation of, 241–242
 - responsive design and, 259
 - CSS pixel, 76, 77
 - CSS-Tricks.com website, 292
 - cultures and color, 111
 - cursive fonts, 48
- ## D
- data-icon** attribute, 208, 209
 - data-media** attribute, 280
 - data-srcset** attribute, 280
 - dd** element, 198
 - declarations
 - CSS, 25
 - DOCTYPE, 9
 - definition lists, 198
 - del** element, 53
 - Depp, Johnny, 71
 - descendant selectors, 26, 27
 - desktop monitors. *See* computer monitors
 - dingbat fonts, 206
 - disc bullets, 196
 - dithering, 129, 144
 - div** element, 32, 41, 231, 241, 246
 - division problems, 251
 - dl** element, 198
 - DOCTYPE declaration, 9
 - drop shadows, 178–179
 - dt** element, 198
 - Ducos du Hauron, Louis, 105
 - dwmgbook.com website, 28, 135, 190

E

- element selectors, 25, 26
- em** element, 15, 31, 52, 53
- em units, 50, 51
- embedding fonts, 58–69
 - @font-face** property for, 59–60
 - commercial font services for, 66–69
 - free font services for, 62–65
 - generating files for, 60–61
- Evernote, 112
- exporting
 - Illustrator files, 154–157
 - image-mapped images, 222
 - Photoshop files, 158–159
 - raster images, 154–156, 158–159
 - vector images, 156–157

F

- fantasy fonts, 48, 49
- favicons, 182–190
 - browser display of, 182–183
 - combining mobile bookmarks and, 193
 - creating for web pages, 185–190
 - image formats for, 183
 - inserting into websites, 184
 - Retina displays and, 186–190
 - subsite or mini-site, 184
- favicon.cc website, 186
- file size
 - bit depth and, 130
 - fonts and, 62, 207
 - reducing for images, 160–165
 - transparency and, 179
- filter** property, 177
- Firefox browser, 182
- Fireworks, Adobe, 158, 219–222
- FitVids.js plugin, 264
- fixed layouts, 236–237, 239–241, 251, 252
- fixed positioning, 39
- flattened images, 168

- flexible image maps, 223–224
- Flickr website, 113
- float** property, 228
- float tolerance, 238
- floats, 228–241
 - behavior of, 228–229
 - clear** property and, 229, 234, 235
 - fixed layouts using, 236–237, 239–241
 - fluid layouts using, 232–235, 237–238
 - page structure and, 230–232
 - pros and cons of using, 241
 - using multiple, 229
- fluid layouts
 - float behavior and, 229
 - responsive design and, 249–253
 - three-column, 237–241
 - two-column, 232–235
- folders, navigating, 18–19
- Font Squirrel, 60
- font stacking, 46
- Fontdeck service, 69
- font-family** property, 46
- fonts
 - embedding, 58–69
 - families of, 46–49
 - file sizes for, 62, 207
 - Google's web fonts, 62–64
 - icon or dingbat, 206–210
 - measurement units for, 49–51
 - mobile-safe, 56
 - size setting, 49–51
 - spaces in names of, 57
 - stacking, 46, 57, 58
 - style setting, 52–53
 - web-safe, 55, 57
 - weight setting, 52
 - See *also* typography
- font-size** property, 49
- font-style** property, 52
- font-weight** property, 52
- foreground color, 100
- frameworks. See CSS frameworks

G

- general sibling selectors, 26, 27
- generic HTML elements, 32
- generic selectors, 25, 26
- GIF images, 133
 - compression scheme for, 133
 - transparency with, 168–171
- Goldilocks approach, 258
- Google accounts, 79
- Google Analytics, 79–86
 - browser statistics, 82–84
 - setting up an account, 79–81
 - site size statistics, 150, 151
 - updating site info in, 85
- Google image search, 113
- Google web fonts, 62–64
- Gore, Al, 147
- gradated color schemes, 107
- gradients, 75
- grid-based frameworks, 242–246
 - 960 Grid System, 245–246
 - Blueprint framework, 242–245
- gzip tool, 162–165

H

- handcoding
 - complete websites, 4
 - image maps, 214–218
- hardware-based testing, 94
- head** element, 10, 29, 30
- headings, HTML, 13–14
- Hedberg, Mitch, 213
- hexadecimal color notation, 98
- Hicks, John, 188
- high-density displays, 264, 265
 - See *also* Retina displays
- Hilton, Paris, 181
- Hische, Jessica, 292
- hotspots
 - creating in Fireworks, 219–221
 - finding coordinates for, 214–215, 216
 - testing in a browser, 222
 - URL links added to, 221

hover effect, 205
hr element, 8
href attribute, 28
 HSL color model, 106
 HSLa color system, 103
 HTML (hypertext markup language), 3–20

- advanced, 20
- coding basics, 7–9
- colors in, 98, 99–101
- CSS used with, 24
- DOCTYPE declaration, 9
- generic elements, 32
- headings, 13–14
- links, 16, 17–20
- list element, 196
- reasons for learning, 4–6
- saving documents in, 12–13
- structuring pages with, 9–12, 13–20
- text markup, 14–15
- title attribute, 17
- validating, 90–92

 HTML5, 9, 31, 230
 HTML5 Boilerplate, 20, 37, 163
HTML & CSS: Design and Build Websites (Duckett), 6
 HTML Dog website, 6
html element, 10, 38, 43
 HTML Goodies website, 6
 HTML Tidy tool, 91–92
 HTTP compression, 162–165
 hue, 104
 Hughes, Kevin, 213
 hyperlinks. *See* links
 hypertext markup language. *See* HTML

I

i element, 15
 ICO files, 183

- creating, 185–186
- storage locker, 186

 icon fonts, 206–210

- clickable icons and, 209–210
- commercial and free, 206

file size warning, 207
 highlighting text with, 208–209
 older IE browsers and, 210
 responsive design and, 266
 Retina displays and, 209
 selecting, 207
Icon Handbook (Hicks), 188
 Icon Slate app, 186
 IconBuilder tool, 187
 icons

- CSS bullet list, 196–197
- inserting as mobile bookmarks, 192
- resource on creating, 188
- tool for building, 187
- web clip sizes for, 190
- See also* favicons

 ID selectors, 25–26
 illustration compression, 142–143
 Illustrator, 148–157

- artboard setup, 151–152
- desktop site design, 149
- mobile browser design, 150
- pixel precision options, 153
- raster image export, 154–156
- Retina display specs, 149, 156
- saving images for the web, 155
- vector image export, 156–157
- workspace setup, 148

 image maps, 213–224

- creating in Fireworks, 219–222
- finding coordinates for, 214–215, 216
- flexible/responsive, 223–224
- handcoding, 214–218

 ImageOptim tool, 160, 161
 images, 127–145, 261–281

- adaptive, 262–281
- aligning, 283–291
- artifacts in, 134
- background, 262, 287
- batch processing of, 160
- bit depth of, 128–130
- blurring background of, 134
- centering, 287–289
- color schemes from, 116

compression charts for, 136–143
 creating for the web, 147–165
 dithering applied to, 129, 144
 exporting, 154–157, 158–159
 GIF format for, 133
 inline vs. background, 262
 JPEG format for, 134
 masking, 174–176
 naming files for, 160
 PNG format for, 135, 144
 positioned within text, 284–286
 posterization of, 129
 raster formats for, 133–135
 reducing file size of, 160–165
 responsive design and, 261–281
 Retina displays and, 149, 156, 264, 265, 267
 rounding corners of, 174
 saving for the web, 154–157, 158–159
 scaling, 262–265
 solutions for adaptive, 266–270
 stretching across a window, 290–291
 SVG format for, 144–145
 transparency for, 167–179
img element, 31, 127, 128, 283
 indexed PNG, 135, 144
 Info palette, Photoshop, 214–215, 217
 inline elements, 31, 41
 inline images, 262
 inline-block elements, 31
 inspiration, color, 112–115
 internal styles, 30
 Internet

- HTML and, 4
- speed issues, 264

 Internet Explorer (IE)

- delivering CSS to, 36–38
- ICO file format and, 182, 183
- icon fonts used in, 210
- older version issues, 177, 210, 262, 268, 281
- transparency problems, 177

Introducing HTML5 (Lawson & Sharp), 230

iOS fonts, 56
iStockphoto, 113
italic fonts, 52
Ives, David, 195

J

JavaScript
 code for fluid images, 262
 flexible image maps and, 223–224
 jQuery and, 223–224, 264, 268, 289
 media queries and, 259
 picturefill code and, 276
 placement of, 81
 popularity of, 67
 -prefix-free file, 90
Jehl, Scott, 274
Johansson, Roger, 201
JPEG image format, 134
jQuery
 center-aligned images and, 289
 responsive image maps plug-in, 223–224
 SVG support for IE plug-in, 268
 YouTube videos plug-in, 264

K

Kapor, Mitchell, 3
Koblentz, Thierry, 264
Kuler, Adobe, 113, 115

L

Lawson, Bruce, 230
layers, Photoshop, 168, 175
Layers palette
 Illustrator, 148
 Photoshop, 168
layouts, 227–259
 adaptive, 249
 clear property in, 229
 CSS frameworks for, 241–246
 floats used in, 228–241
 fluid vs. fixed, 232–241

 page structure for, 230–232
 responsive, 247–259
 three-column, 237–241
 two-column, 232–237
Learning Web Design (Robbins), 6
letter spacing, 54
Lewis, Emily, 32
line height, 54
line length, 253
link element, 28, 29
links, 6
 absolute, 17, 20
 colors for, 100–101
 CSS, 28–29
 icon, 209–210
 relative, 17–18, 20
 root relative, 19–20
 title attribute for, 17
 website, 16
lists, 194, 195–205
 bullet icons for, 196–197
 counters for, 201
 CSS used for, 203–205
 definition, 198
 effective design of, 203–205
 icon fonts used in, 206
 ordered, 199–202
 unordered, 196–197
local web servers, 20
lossless compression, 133, 135, 144
lossy compression, 134

M

map element, 216
Marcotte, Ethan, 247, 262
Markdown files, 20
markers, bullet, 196–197
markup vs. programming, 6
masking images, 174–176
matchmedia.js code, 276
matting issues, 170–171
Maxwell, James Clerk, 105
max-width property, 223, 257, 262
media queries, 248–249, 254–258
 creating, 256–257

 example of using, 254–255
 reference list of, 257–258
 website showcasing, 250
media scaling, 262
media types, 248
microformats, 32
middle alignment, 286
mobile bookmarks, 190–193
 combining favicons and, 193
 file formats for, 191
 icon sizes for, 190
 inserting icons as, 192
 iOS treatment of, 191
 naming conventions for, 191
mobile devices
 document size settings for, 149
 list of media queries for, 257–258
 responsive design for, 247, 255
 testing websites on, 94, 131
Mobile Safari, 190
mobile-safe fonts, 56
Modernizr tool, 268
monitors. *See* computer monitors
monochromatic color schemes, 107
monospace fonts, 48, 49
Mother Teresa, 167
MP3 audio format, 273
multi-step resizing technique, 190

N

naming/renaming
 image files, 160
 web clips, 191
navigating folders, 18–19
normal flow, 39
normal fonts, 52
normalizing styles, 88–89
noscript tags, 276

O

oblique fonts, 52
O’Keefe, Georgia, 97
ol list tag, 199

- opacity
 - color, 103
 - transparency, 176
- opening/closing tags, 8
- Oracle VirtualBox, 92, 93
- ordered lists, 199–202
 - adding markers to, 201–202
 - changing the order of, 199
 - table of contents for, 200–201

P

- p** element, 8, 13
- parent element, 40, 42–43
- patch, picture, 274–281
- patterns, compression for, 140–141
- photographs
 - color schemes from, 116
 - compression chart for, 138–139
- Photoshop, 158–159
 - Actions feature, 160
 - background pattern, 169
 - favicon creation, 187–190
 - finding x and y coordinates in, 214–215
 - GIF transparency creation, 168–169
 - image mask creation, 175–176
 - layers used in, 168, 175
 - multi-step resizing in, 190
 - new document setup, 158
 - raster image export, 158–159
- PHP files, 161
- picture** element, 270–272
- picturefill code, 274–281
- pixels, 76–77
 - display density, 264
 - precision options, 153
 - problem with using, 50
- PNG images, 135
 - 8- and 24-bit, 172–173
 - dithering of, 144
 - transparency with, 172–173
- PNGGauntlet tool, 160, 161
- polyfill code, 274
- position** property, 38

- positioning
 - absolute, 40–41, 42, 288
 - example of, 42–43
 - fixed, 39
 - relative, 39, 42–43
 - static, 38–39
- posterization, 129
- “precomposed” suffix, 191
- prefix CSS properties, 89
- prefix-free file, 90
- primary color systems, 105
- print-optimized CSS, 248
- programming vs. markup, 6
- pseudo-classes, 33–35
- pseudo-elements, 33–35

R

- raster images
 - compressing, 160–161
 - exporting, 154–156, 158–159
 - GIF format for, 133
 - JPEG format for, 134
 - PNG format for, 135, 144
- RatioSTRONG calculator, 251
- reference pixel, 76, 77
- rel** attribute, 28
- relative links, 17–18, 20
- relative positioning, 39, 42–43
- rem units, 50, 51
- resetting styles, 88
- responsive design, 247–259
 - adaptive design vs., 249
 - compressing images for, 267–270
 - CSS framework for, 259
 - fluid layouts and, 249–253
 - icon fonts and, 266
 - images used in, 261–281
 - media queries and, 248–249, 250, 254–258
- picture** element for, 270–272
- picturefill code for, 274–281
- scaling media for, 262–265
- srcset** attribute for, 272–274
- SVG images and, 266
- text reflow and, 253

- responsive image format, 273
- responsive image maps, 223–224
- Responsive Images Community Group, 270
- Retina displays
 - favicons for, 186–190
 - icon fonts and, 209
 - image size and, 149, 156, 264, 265, 267
- RGB color mode, 76
- RGBA color feature, 103
- rollovers, list item, 205
- root folder, 17, 18, 184, 192
- root relative links, 19–20
- rounded corners, 174
- rules, CSS, 25
- Rupert, Dave, 264

S

- sans serif fonts, 48
- saturation, 104
- saving
 - HTML documents, 12–13
 - Illustrator files, 154–157
 - Photoshop files, 158–159
- scaling images, 262–265
 - basics of, 262
 - problems with, 263–265
- security, favicon, 182
- selectors, 25–27
- self-closing elements, 8
- serif fonts, 48
- shadows
 - 3D, 178
 - box, 178–179
 - text, 177–178
- sharing your knowledge, 6
- Sharp, Remy, 230
- sizing/resizing technique, 190
- smartphones. *See* mobile devices
- Smashing Magazine, 6
- software-based testing, 92–93
- source code, viewing in browsers, 5
- span** attribute, 209, 210
- span** element, 32

- square bullets, 197
- srcset** attribute, 272–274
- stacking fonts, 46, 57, 58
- static positioning, 38–39
- storage locker, 186
- strong** element, 15, 52
- style** element, 30
- stylesheet** value, 28
- subsites, favicons for, 184
- subtractive primary colors, 105
- Superman font, 206
- SVG images, 144–145, 266, 268, 291
- SVGZ files, 144
- Swatches palette, Illustrator, 148
- Symbols palette, Illustrator, 148
- syntax for selectors, 26

T

- table of contents, 200–201
- tablets. *See* mobile devices
- tags, HTML, 8
- testing, 92–94
 - hardware-based, 94
 - software-based, 92–93
- text
 - 3D effects for, 178
 - alignment of, 54
 - effects added to, 53
 - enhancing with icons, 208–209
 - HTML markup of, 14–15
 - image alignment related to, 284–286
 - optimum line length for, 253
 - responsive reflow of, 253
 - shadows added to, 177–178
 - spacing of, 54
 - See also* typography
- text editors, 7
- text shadows, 75
- text-align** property, 54
- text-bottom** value, 285
- text-decoration** property, 53
- text-top** value, 285
- three-column layouts, 236–241
 - fixed layout, 236–237
 - fluid layout, 237–241

- title** attribute, 17
- title** element, 11, 17, 29
- tooltips, 17
- top/bottom alignment, 286
- trailing slash, 9
- Transform palette, Illustrator, 148
- transparency, 167–179
 - color, 103, 177
 - CSS-based, 174–179
 - GIF image, 168–171
 - matting and, 170–171
 - older IE browsers and, 177
 - opacity and, 176
 - PNG image, 172–173
 - shadows and, 177–179
- Twain, Mark, 8, 45
- two-column layouts, 232–237
 - fixed layout, 236–237
 - fluid layout, 232–235

- type** attribute, 28, 30
- Typekit fonts, 66–69
- typography, 45–69
 - alignment settings, 54
 - embedded fonts, 58–69
 - font families, 46–49
 - icon fonts, 210
 - letter spacing, 54
 - line height, 54
 - measurement units, 49–51
 - mobile-safe fonts, 56
 - size settings, 49–51
 - stacking fonts, 46, 57, 58
 - style settings, 52–53
 - text effects, 53
 - web-safe fonts, 55, 57
 - weight settings, 52
 - See also* fonts; text

U

- ul** element, 196
- Unicode standard, 7
- universal selectors, 27
- unordered lists, 196–197
- URL palette, Fireworks, 221
- URLs (uniform resource locators), 6
- usemap** property, 214

V

- validators, 90–92
- values
 - attribute, 16
 - color, 104
- vector images
 - exporting from Illustrator, 156–157
 - favicon creation using, 187
 - responsive design and, 266
 - SVG format for, 144–145
- vendor prefixes, 89–90
- Verou, Lea, 90
- vertical-align** property, 284–286
- videos, scaling, 262
- Vine, Tim, 261
- virtualization, 92
- Vischeck tool, 78

W

- web browsers
 - centering images in, 287–289
 - CSS support by, 24, 74
 - differences between, 72–74
 - favicon support by, 182
 - finding statistics about, 82–84
 - historical importance of, 4
 - normalizing styles for, 88–89
 - resetting styles for, 88
 - stretching images in, 290–291
 - test pages for, 72–73, 74–75
 - validating code for, 90–92
 - vendor prefixing for, 89–90
 - viewing source code with, 5
- web clips
 - icon sizes for, 190
 - naming conventions for, 191
 - See also* mobile bookmarks
- web environment, 72–79
 - accessibility issues and, 77–79
 - browser differences and, 72–75
 - color display differences and, 75–76
 - pixel definitions and, 76–77

- web pages
 - favicon creation for, 185–190
 - HTML structuring of, 9–12, 13–20
 - saving and viewing, 12–13
 - two-second standard for, 163
- web resources
 - book’s website, 28, 135, 190
 - color inspiration, 112–115
 - font stack galleries, 58
 - media queries showcase, 250
 - web design/development, 6
- web servers, 20
- Web workspace, 148
- WebINK service, 69
- webkit-mask-box-image** property, 176
- web-safe fonts, 55, 57
- websites
 - analyzing traffic to, 79–86
 - favicons added to, 184
 - hand-coding of, 4
 - HTML links to, 16
 - speed check for, 131
 - testing, 92–94
 - validating, 90–92
- width** property, 229
- Wilhite, Steve, 133
- word processing programs, 7
- World Wide Web Consortium (W3C),
71, 91

X

- XMB and XPM image formats, 133
- XML and XHTML markup
 - languages, 9

Y

- YouTube videos, 264

Z

- Zapf Dingbats font, 206
- zoom** filter, 177