Jennifer Kyrnin

24 Proven One-Hour Lessons

Sams Teach Yourself

HTML5 Mobile Application Development in 24 Hours

SAMS

FREE SAMPLE CHAPTER

SHARE WITH OTHERS

Jennifer Kyrnin

Sams Teach Yourself HTML5 Mobile Application Development in 24 Hours

SAMS 800 East 96th Street, Indianapolis, Indiana, 46240 USA

Sams Teach Yourself HTML5 Mobile Application Development in 24 Hours
Copyright © 2012 by Pearson Education, Inc.

All rights reserved. No part of this book shall be reproduced, stored in a retrieval system, or transmitted by any means, electronic, mechanical, photocopying, recording, or otherwise, without written permission from the publisher. No patent liability is assumed with respect to the use of the information contained herein. Although every precaution has been taken in the preparation of this book, the publisher and author assume no responsibility for errors or omissions. Nor is any liability assumed for damages resulting from the use of the information contained herein.

ISBN-13: 978-0-672-33440-5
ISBN-10: 0-672-33440-2

Library of Congress Cataloging-in-Publication Data:

Kyrnin, Jennifer.
Sams teach yourself HTML5 mobile application development in 24 hours / Jennifer Kyrnin.
p. cm.
Includes bibliographical references and index.
ISBN 978-0-672-33440-5 (pbk. : alk. paper)
1. HTML (Document markup language) 2. Mobile computing—Programming. 3. Application software—Development. I. Title.
QA76.76.H94K97 2012
006.7'4—dc23
2011036380

Printed in the United States of America

First Printing: November 2011 with corrections April 2013

Trademarks

All terms mentioned in this book that are known to be trademarks or service marks have been appropriately capitalized. Sams Publishing cannot attest to the accuracy of this information. Use of a term in this book should not be regarded as affecting the validity of any trademark or service mark.

Warning and Disclaimer

Every effort has been made to make this book as complete and as accurate as possible, but no warranty or fitness is implied. The information provided is on an "as is" basis. The author and the publisher shall have neither liability nor responsibility to any person or entity with respect to any loss or damages arising from the information contained in this book or programs accompanying it.

Bulk Sales

Sams Publishing offers excellent discounts on this book when ordered in quantity for bulk purchases or special sales. For more information, please contact

U.S. Corporate and Government Sales
1-800-382-3419
corpsales@pearsontechgroup.com

For sales outside of the U.S., please contact

International Sales
international@pearson.com

Editor-in-Chief
Mark Taub

Acquisitions Editor
Trina MacDonald

Development Editor
Michael Thurston

Managing Editor
Kristy Hart

Project Editors
Jovana San Nicolas-Shirley and Jess DeGabriele

Copy Editor
Paula Lowell

Senior Indexer
Cheryl Lenser

Proofreader
Leslie Joseph

Technical Editors
Evan Burchard
Pascal Rettig

Publishing Coordinator
Olivia Basegio

Cover Designer
Gary Adair

Compositor
Nonie Ratcliff

Contents at a Glance

Preface ... xiv

Part I: Building Web Pages and Applications with the Open Web Standard

HOUR 1 Improving Mobile Web Application Development with HTML5 ... 1
2 New HTML5 Tags and Attributes with Mobile Development ... 15
3 Styling Mobile Pages with CSS3 ... 33
4 Detecting Mobile Devices and HTML5 Support ... 49
5 JavaScript and HTML5 Web Applications ... 67
6 Building a Mobile Web Application ... 85
7 Upgrading a Site to HTML5 ... 105
8 Converting Web Apps to Mobile ... 119

Part II: Learning the HTML5 Essentials

HOUR 9 Adding Meaning with HTML5 Sectioning and Semantic Elements ... 141
10 Drawing with the HTML5 Canvas Element ... 163
11 Fonts and Typography in HTML5 ... 187
12 Audio and Video in HTML5 ... 205
13 HTML5 Forms ... 223
14 Editing Content and User Interaction with HTML5 ... 245
15 Microformats and Microdata ... 261
16 Working with HTML5 Drag-and-Drop Functionality ... 275
17 HTML5 Links ... 293

Part III: HTML5 for Mobile and Web Applications

HOUR 18 Web Application APIs and Datasets ... 309
19 WebSockets, Web Workers, and Files ... 323
20 Offline Web Applications ... 341
21 Web Storage in HTML5 ... 355
22 Controlling the Browser History with the History API ... 373

23 Adding Location Detection with Geolocation 387
24 Converting HTML5 Apps to Native Apps 405

Appendixes

A Answers to Quizzes 423
B HTML Elements and Attributes 437
C HTML5 and Mobile Application Resources 447

Index 449

Table of Contents

Preface xiv

Part I: Building Web Pages and Applications with the Open Web Standard

HOUR 1: Improving Mobile Web Application Development with HTML5 **1**

Understanding How We Got to HTML5 1

Learning What's Different with HTML5 3

Defining Web Applications 4

Using the Open Web Standard 4

Using HTML5 with iOS and Android Devices 6

Writing Mobile Websites 7

Summary 11

Q&A 12

Workshop 13

HOUR 2: New HTML5 Tags and Attributes with Mobile Development **15**

The New HTML5 Tags 15

The New HTML5 Attributes 24

Changes to HTML 4 Tags and Attributes 25

Changes to HTML Syntax in HTML5 27

Mobile Support of HTML5 Tags and Attributes 28

Benefits of HTML5 for Mobile Web Development 29

Summary 30

Q&A 30

Workshop 31

HOUR 3: Styling Mobile Pages with CSS3 **33**

Quick Introduction to CSS 33

What CSS3 Adds to the Party 42

Using CSS3 on Mobile Devices 46

Summary ... 46
Q&A ... 46
Workshop ... 47

HOUR 4: Detecting Mobile Devices and HTML5 Support 49
Choosing What HTML5 Elements to Use ... 50
Android and iOS Support for HTML5 ... 50
Detecting HTML5 Functions ... 52
Degrading Gracefully ... 58
Using CSS3 Media Queries to Detect Mobile Browsers ... 60
Testing Your Applications ... 63
Summary ... 64
Q&A ... 64
Workshop ... 65

HOUR 5: JavaScript and HTML5 Web Applications 67
What is JavaScript? ... 67
What is jQuery? ... 72
Using jQuery Mobile ... 79
Summary ... 82
Q&A ... 82
Workshop ... 83

HOUR 6: Building a Mobile Web Application 85
Building a Site that Works on All Devices ... 85
Deciding on What Type of Application You Want ... 86
Building the Application in HTML ... 89
Using CSS to Make the HTML Look Good ... 91
Adding Mobile Meta Tags for More Effective HTML5 Pages ... 96
Optimizing Your Site for Mobile ... 99
Summary ... 102
Q&A ... 102
Workshop ... 103

HOUR 7: Upgrading a Site to HTML5 105
Deciding When and How to Upgrade from HTML 4 105
HTML5 Features that Work Right Now 111
The State of HTML5 Browser Support 112
Adding HTML5 Features as Extras on Your Site 113
HTML5 Features that Turn Your Site into a Killer Mobile Application 115
Summary 116
Q&A 116
Workshop 118

HOUR 8: Converting Web Apps to Mobile 119
Choosing a Web Editor 120
Testing Your Application 121
Evaluating Your Content 123
Changing the Visual Design for Mobile 124
Checking for HTML5 and CSS3 130
Supporting Multiple Devices 132
Evaluating Finished Apps on Other Devices 133
Getting an Application to Work on Older Browsers 134
Summary 138
Q&A 139
Workshop 139

Part II: Learning the HTML5 Essentials

HOUR 9: Adding Meaning with HTML5 Sectioning and Semantic Elements 141
What Are Sectioning Elements? 141
Using the New Sectioning Elements 142
Marking Up HTML Semantically 154
Summary 159
Q&A 159
Workshop 160

HOUR 10: Drawing with the HTML5 Canvas Element 163

Using the Canvas Element 163
Drawing Shapes on the <Canvas> Element 165
Writing Fonts and Text on the Canvas 177
Displaying Images 179
How Is Canvas Different from SVG or Flash 182
Summary 183
Q&A 184
Workshop 185

HOUR 11: Fonts and Typography in HTML5 187

Defining the Elements of Typography 187
Using Proper Typographical Entities 196
Understanding Web Open Font Format (WOFF) 198
Summary 202
Q&A 202
Workshop 203

HOUR 12: Audio and Video in HTML5 205

Why Use HTML5 for Audio and Video vs. Flash 205
Choosing Video Formats for the Best Compatibility 207
Choosing Audio Codecs for the Widest Support 208
The New HTML5 Media Elements 210
Useful Attributes to Extend Your Media 212
Creating Fallback Options for Internet Explorer 216
Creating Custom Controls with API Methods 217
Summary 219
Q&A 220
Workshop 220

HOUR 13: HTML5 Forms 223

New Usability Features in HTML5 Forms 223
HTML5 Input Types 228
Other New Form Elements 235

Form Validation 237
Summary 241
Q&A 242
Workshop 242

HOUR 14: Editing Content and User Interaction with HTML5 **245**
The New `contenteditable` Attribute 245
The `execCommand` Method 247
Adding Spellcheck to Web Pages 251
Hiding Elements 252
Additional UI Components of HTML5 253
Browser Support of UI and Editing Features 255
Summary 257
Q&A 258
Workshop 258

HOUR 15: Microformats and Microdata **261**
Using Microformats 261
Using Microdata 267
Using RDFa 269
Deciding Which Format to Use 270
Mobile and Microformats 271
Summary 272
Q&A 273
Workshop 274

HOUR 16: Working with HTML5 Drag-and-Drop Functionality **275**
Implementing Drag and Drop 275
Drag-and-Drop Events 276
Drag-and-Drop Attributes 279
Helpful CSS Extensions 280
Building a Drag-and-Drop Interface 280
Using Drag and Drop on iOS 287

Summary 290
Q&A 290
Workshop 291

HOUR 17: HTML5 Links 293
How Links Have Changed in HTML5 293
Link Types and Relationships 299
Using the New Link Types 300
Summary 306
Q&A 306
Workshop 307

Part III: HTML5 for Mobile and Web Applications

HOUR 18: Web Application APIs and Datasets 309
Creating Web Applications 309
Datasets and `data-*` Attributes 318
Summary 320
Q&A 320
Workshop 321

HOUR 19: WebSockets, Web Workers, and Files 323
Two-Way Communication with WebSockets 323
Running Scripts in the Background with Web Workers 326
Handling Client-Side Files with the File API 331
Summary 337
Q&A 338
Workshop 339

HOUR 20: Offline Web Applications 341
Building Offline Apps and Converting Apps to Work Offline 341
The Cache Manifest 343
Using DOM Events and Properties for Offline Apps 347
Debugging the Application Cache 351

Summary ... 352
Q&A ... 352
Workshop ... 353

HOUR 21: Web Storage in HTML5 355
What Is Web Storage? ... 356
Web SQL and Indexed DB ... 361
Summary ... 370
Q&A ... 371
Workshop ... 372

HOUR 22: Controlling the Browser History with the History API 373
Why Control the Browser History? ... 373
History API Methods ... 375
Using the History API ... 375
Dangers and Annoyances of the History API ... 383
Summary ... 384
Q&A ... 385
Workshop ... 386

HOUR 23: Adding Location Detection with Geolocation 387
What Is Geolocation? ... 387
Privacy and Geolocation ... 394
Creating a Mobile Geolocation Application ... 396
Summary ... 401
Q&A ... 401
Workshop ... 402

HOUR 24: Converting HTML5 Apps to Native Apps 405
Comparing the Difference Between Native and HTML5 Apps ... 405
Converting to Native Apps ... 408
Creating Application Icons ... 414
Testing Your Applications ... 416
Selling Your App in the App Stores ... 418

Options Other Than Converting to Native Apps 419
Summary 419
Q&A 420
Workshop 421

Part IV: Appendixes

APPENDIX A: Answers to Quizzes **423**

Hour 1, “Improving Mobile Web Application Development with HTML5” 423
Hour 2, “New HTML5 Tags and Attributes with Mobile Development” 423
Hour 3, “Styling Mobile Pages with CSS3” 424
Hour 4, “Detecting Mobile Devices and HTML5 Support” 424
Hour 5, “JavaScript and HTML5 Web Applications” 425
Hour 6, “Building a Mobile Web Application” 425
Hour 7, “Upgrading a Site to HTML5” 426
Hour 8, “Converting Web Apps to Mobile” 426
Hour 9, “Adding Meaning with HTML5 Sectioning and Semantic Elements” 427
Hour 10, “Drawing with the HTML5 Canvas Element” 427
Hour 11, “Fonts and Typography in HTML5” 428
Hour 12, “Audio and Video in HTML5” 428
Hour 13, “HTML5 Forms” 429
Hour 14, “Editing Content and User Interaction with HTML5” 429
Hour 15, “Microformats and Microdata” 430
Hour 16, “Working with HTML5 Drag-and-Drop Functionality” 430
Hour 17, “HTML5 Links” 430
Hour 18, “Web Application APIs and Datasets” 431
Hour 19, “WebSockets, Web Workers, and Files” 431
Hour 20, “Offline Web Applications” 432
Hour 21, “Web Storage in HTML5” 432

Hour 22, "Controlling the Browser History with the History API" 433
Hour 23, "Adding Location Detection with Geolocation" 434
Hour 24, "Converting HTML5 Apps to Native Apps" 434

APPENDIX B: HTML Elements and Attributes **437**

HTML5 Elements 437
HTML5 Attributes 442

APPENDIX C: HTML5 and Mobile Application Resources **447**

Books 447
Websites 448
This Book's Website 448

Index **449**

Preface

The web is changing very quickly these days. New browser versions are being released every few months rather than every few years, and new devices are entering the marketplace all the time. For a web developer, staying up to date on the latest trends and technology is important, and the trending technology right now is HTML5.

In fact, some news outlets are claiming that HTML5 and mobile applications are two of the fastest-growing areas of job creation. According to Freelancer.com and iTWire, requests for freelancers knowing HTML5 rose by 34% in the first quarter of 2011, while general HTML jobs rose only by 7%. They also noted that Android jobs rose by 20%, and iPhone jobs rose by 9%.[1] HTML5 and mobile applications are where the jobs are, and this book can help you learn all about HTML5 and the application programming interfaces (APIs) that relate to it. After 24 hours, you will be able to build complex web applications and convert them into native mobile applications.

HTML5 Is More Than HTML

This book covers more than HTML tags and attributes. Although these things are the basis of the HTML5 specification, when people talk about HTML5 they often include many other programming interfaces that are not strictly part of the HTML5 specification, like geolocation or the History API. This book covers the basics of HTML5 and how it has changed from previous versions of HTML. It also introduces you to some of the technologies that are lumped in with HTML5, including:

- Drawing with the `canvas` element
- Adding streaming media with the `video` and `audio` elements
- Editing pages online and checking spelling
- Using drag-and-drop functions on web applications

[1] "Freelancer.com job listings show growth in HTML5, Adsense, and Android." iTWire. July 11, 2011. www.itwire.com/it-people-news/recruitment/48392-freelancercom-job-listings-show-growth-in-html5-adsense-and-android. July 25, 2011.

- Building more user-friendly forms
- Creating semantic divisions with new elements, such as `article`, `section`, and `nav`

This book covers several other specifications beyond HTML5, including:

- Web Open Font Format (WOFF) web fonts
- Microformats and Microdata
- WebSockets
- Web Workers
- Files API
- Web Storage
- Offline Web Applications API
- History API
- Geolocation

Web Pages Are for More Than Computers

HTML used to be used primarily in web browsers on computers, but now, with the popularity of smartphones and tablet computers, more people are accessing web pages on mobile devices.

Every hour of this book provides examples of how the lesson's contents apply both to web browsers and mobile devices and shows you techniques for getting your applications to look better on mobile devices.

With this book, you will learn how to create applications that work on the most popular mobile smartphones and tablets out there: Android and iOS (iPhone, iPad, and iPod touch devices). Screenshots from both Android and iOS devices appear throughout as well as tips and warnings about how the different devices perform.

How to Use This Book

This book is divided into 24 lessons. Each lesson covers a specific topic related to HTML5 or an API that is part of the Open Web Standard. Each lesson takes about an hour to complete.

Organization of This Book

This book is divided into three sections:

- Part I, "Building Web Pages and Applications with the Open Web Standard," teaches you the basics of HTML, CSS, and JavaScript, and teaches you how to build a basic web application for mobile and non-mobile devices. After reading this section, you will know how to build a basic website with HTML, CSS, and JavaScript.
- Part II, "Learning the HTML5 Essentials," covers some of the more important new features of HTML5. You will learn more about new HTML5 elements to help you build better applications.
- Part III, "HTML5 for Mobile and Web Applications," describes some of the more useful APIs and tools for mobile application development and goes into detail about how to create mobile applications.

Conventions Used in This Book

Code samples are written in `mono font` within the text of the book, while blocks of code will be called out separately, for example:

```
This is a block
Of code
```

Some code examples that are too long to display as one line in the book use the ➥ symbol to indicate that these lines should be all on one line, like this:

```
<link rel="stylesheet" href="styles-320.css"
➥media="only screen and (max-width:320px)">
```

This book has three types of sidebars:

By the Way

By the Way notes provide additional information about the topics that are discussed in the hour.

Did you Know?

Did you Know? tips share interesting facts or tidbits about the related content.

Watch Out!

Watch Out! warnings alert you of things that can cause problems for your applications.

You can also use the Try It Yourself sections to help you practice what you've learned in the hour.

Try It Yourself ▼

Nearly every hour will have at least one step-by-step tutorial called "Try It Yourself" to help you use what you've learned. ▲

Q&A, Quiz, and Exercises

Every hour ends with a short question-and-answer section to help with follow-up questions that occur as a result of reading the hour. You can also take a short quiz on the hour (Appendix A provides the answers) as well as do some suggested exercises to help you get more out of what you learned and apply this knowledge to your own applications.

Where to Go to Learn More

Appendix C includes more websites and books you can access to learn more about HTML5 and mobile web applications. This book also has a companion website at www.html5in24hours.com/ where you can go to see the examples, view and download the source code for each hour, view and report errata about the book, and continue to learn and ask questions about HTML5 mobile applications.

About the Author

Jennifer Kyrnin has been teaching HTML, XML, and web design online since 1997. She has built and maintained websites of all sizes from small, single-page brochure sites to large, million-page databased sites for international audiences. She lives with her husband, son, and numerous animals on a small farm in Washington state.

Dedication

To Mark and Jaryth, you helped me find time I didn't know I had. I love you.

Acknowledgments

I would like to thank all the people at Sams for the opportunity to write this book and work with you. I would particularly like to thank Trina MacDonald and Olivia Basegio for keeping me moving and the book on track as well as my two fabulous technical editors Pascal Rettig and Evan Burchard for all the great suggestions and corrections. Any technical errors you find in the book are mine alone; they probably tried to stop me.

I would also like to thank my family for putting up with me while I wrote the book, and the members of the Woodinville Writers Group, without whom I would have felt very alone as I hunkered down writing.

We Want to Hear from You!

As the reader of this book, *you* are our most important critic and commentator. We value your opinion and want to know what we're doing right, what we could do better, what areas you'd like to see us publish in, and any other words of wisdom you're willing to pass our way.

You can email or write me directly to let me know what you did or didn't like about this book—as well as what we can do to make our books stronger.

Please note that I cannot help you with technical problems related to the topic of this book, and that due to the high volume of mail I receive, I might not be able to reply to every message.

When you write, please be sure to include this book's title and author as well as your name and phone or email address. I will carefully review your comments and share them with the author and editors who worked on the book.

E-mail:	webdev@samspublishing.com
Mail:	Mark Taub Editor-in-Chief Pearson Education 1330 6th Avenue New York, NY 10019 USA

Reader Services

Visit our website and register this book at informit.com/register for convenient access to any updates, downloads, or errata that might be available for this book.

This page intentionally left blank

HOUR 1

Improving Mobile Web Application Development with HTML5

What You'll Learn in This Hour:

- ▶ How HTML has grown and changed since it was invented
- ▶ Where HTML5 fits in with the other versions of HTML
- ▶ What the Open Web Standard is and how it relates to HTML5
- ▶ How a web application differs from typical web pages
- ▶ How to build a very simple HTML5 web page
- ▶ Why you want to use HTML5 for your mobile applications

HTML5 is the latest version of HTML, and although adoption on desktop browsers such as Internet Explorer has been slow, mobile devices are jumping on the bandwagon in record numbers. Nearly every smartphone and tablet device sold today supports HTML5, and those numbers are growing.

In this hour you will learn how HTML5 came into being and how it has changed the landscape for web designers and developers as well as the customers viewing your pages. You'll learn to build a simple HTML5 document and why HTML5 is the language you should know if you want to design and develop mobile applications.

Understanding How We Got to HTML5

In March 1989, Sir Tim Berners-Lee wrote a proposal that suggested using hypertext to link related documents together over a network. After collaborating with others at CERN, hypertext eventually became HTML or Hypertext Markup Language.

HTML was based on a language already in use for marking up documents—SGML (Standard Generalized Markup Language). In September 1991, a discussion began across the internet about how the web and HTML should evolve.

Up until around 1993, the only browser available was a text-only browser called Lynx. Then Mosaic came out with features such as images, nested lists, and forms. Most designers these days take these things for granted, but back in the early 1990s many people browsed the web in a black-and-white (or green-and-black), text-only environment. Getting a browser to support images was very exciting.

It wasn't until 1994 that the HTML working group was set up by the IETF (Internet Engineering Task Force). In July it released a working draft of HTML 2. Later that year, the W3C, or World Wide Web Consortium, was formed at MIT to act as a standards body for HTML. HTML 3 was released as a draft in 1995, and HTML 3.2 was endorsed as a standard in 1997. HTML 4 was published as a recommendation in 1999.

XML and XHTML

After 1999 things began to change. The W3C no longer felt that HTML should remain as it was. Instead, they wanted to make it more machine-readable, more consistent, and much stricter. So, rather than working on a new version of HTML, they began turning HTML into a strict markup language called XHTML.

XHTML was created as a version of HTML 4.01 that was rewritten in XML (eXtensible Markup Language). It was developed in 1998 as a way to create markup languages that are machine readable. XHTML documents must be well formed and valid. In fact, the W3C wanted all browsers that read XHTML to stop rendering the page if the page's HTML was not valid or well-formed.

XML is still used by many companies. For example, many content management systems (CMSs) use XML on the back end to manage large websites; many books are written in DocBook, which is an XML language for publishing; and ePub books use XML to create ebooks.

By the Way

Well-Formed Versus Valid

A document that is well-formed has the declaration statement at the top—including the specification, all attributes are surrounded by quotation marks, all elements are closed, and there is only one container element. A document that is valid is one that is checked against the specification and has no errors.

HTML5 is Born

XHTML, because it is based on XML, has the same strict requirements as XML, which makes XHTML very difficult to write. Although most web designers recognize the importance of creating HTML that is valid, at the end of the day the most important thing is that the HTML works in readers' browsers. Every beginning web designer who has ever validated a page knows that just because a page isn't valid doesn't mean browsers won't be able to display it. In fact, web browsers have no problem displaying technically invalid HTML.

Because of the difficulties writing XHTML, a group of web designers and developers as well as browser makers got together in 2004 and formed the Web Hypertext Application Technology Working Group (WHATWG). They started building the HTML5 specification to address the needs of designers, developers, and browser makers. Finally, in 2008, the W3C decided to scrap XHTML development in favor of reintegrating with the HTML5 community, and added the HTML5 specification into the W3C framework.

Learning What's Different with HTML5

HTML 4 is the last recommendation developed by the W3C alone. Most web pages right now are built in HTML 4 because it is widely supported by web browsers and editors.

XHTML was created by rewriting the HTML 4.01 specification as XML, which means that all tags must be closed, the XHTML tags must be written in all lowercase, all attributes must have quotation marks around them, and tags must be nested without overlapping.

Nesting Tags Correctly

When you nest two HTML tags, you should think of them as a stack of bowls—one inside the other. Always close the nested tag first, and then close the outer tag.

Incorrect:

```
This text is <em>italic, and this is <strong>bold and
italic</em></strong>
```

Correct:

```
This text is <em>italic, and this is <strong>bold and
italic</strong></em>
```

HTML5 goes back to a less restrictive version of HTML. End tags are no longer required for all elements, you can write in upper- or lowercase, and attributes don't need to have quotations around them all the time.

HTML5 also adds a lot of new elements, including a streamlined doctype (or DTD—the first line of your HTML document. It tells the browser that this document is an HTML5 document), sectioning elements, many new form features, and support for drag and drop and other features useful for creating web applications.

Did you Know?

A New HTML5 Doctype

HTML5 has a new streamlined doctype that is very easy to remember—`<!doctype html>`. Nothing else is required. It doesn't even have to be written in all caps.

Defining Web Applications

Applications are software programs that are used on a local computer to do various tasks. The most commonly used applications are web browsers (such as Internet Explorer or Firefox), document editors (such as Word), and email clients (such as Outlook or Thunderbird). These programs are very similar to one another because they all run on the same operating system. They have features such as

- A similar look and feel, such as the menus at the top
- Functionality such as drag-and-drop, saving to the hard drive, and interactivity

Web applications are web pages that are attempting to look and act like desktop applications. They are written to run inside a web browser, rather than directly on the computer. This means that they are limited by the functions that the web browser can and cannot do:

- Web applications rely on the web browser for functionality that would otherwise have to be coded (such as the back button, rendering the page, and so on).
- Web applications are limited the same way a browser is limited. They can't save data to the hard drive, they have only limited scripting functions, and they can't interact directly with the computer operating system.

Web applications, unlike desktop applications, are not limited to one operating system. A web application runs in a browser, and so anywhere a browser will run, the web application will run.

Using the Open Web Standard

HTML5 was written primarily as a way to develop better, more efficient web applications, and it is part of the suite of APIs and specifications developed under the Open Web Standard. The Open Web Standard or Open Web Platform is a collection of royalty-free technologies that enable the web.

Many people think HTML5 includes more than it does. In fact, features such as the History API (discussed in Hour 22, “Controlling the Browser History with the History API”), local storage (Hour 21, “Web Storage in HTML5”), and geolocation (Hour 23, “Adding Location Detection with Geolocation”) are all separate specifications that work with HTML5 to create a suite of tools you can use to build web pages, web applications, mobile applications, and more. These all are part of the Open Web Standard.

Some of the specifications in this standard include:

- HTML5
- CSS3
- Web Fonts
- HTML Canvas
- SVG
- Web storage
- Geolocation

By using standards-based specifications for your web applications, you will know that your pages and applications will work for a wider audience, and that your pages and applications will last longer.

Try It Yourself ▼

Building Your First HTML5 Document

HTML5 is, at its heart, HTML, which is what you use to build web pages. So before you can get started on the applications that you'll develop in later hours, you need to know how to build a web page.

You start by writing some HTML, which is very easy to write. All you need is a text editor.

Did you Know?

Finding Your Computer's Text Editor

If you have access to a computer, you have access to a text editor for writing HTML. On Windows type in **Notepad** in the Search programs and files box in your Start menu. On Macintosh, type in **TextEdit** in the Spotlight. Use either the **vi** or **Emacs** command on a Linux computer.

After you have a text editor up and running, you can begin writing your HTML, which is defined by tags that are written inside of less-than (<) and greater-than (>) signs.

1. Open your text editor and type the following:

```
<!doctype html>
<html>
    <head>
        <title>This is my first HTML5 page</title>
    </head>
    <body>
        <h1>My First HTML Document</h1>
        <p>This is my first HTML5 document.
   </body>
</html>
```

2. Save your file as **`mypage.html`**.

Watch Out!

Check That File Extension

Make sure to check the extension of your HTML file in your file system. Notepad will often convert it to a .txt file if you aren't careful. If it does, simply close Notepad and replace the .txt extension with .html.

3. Now open this page in your favorite web browser (by browsing to it in the File menu) to test that your page displays correctly. If it doesn't display correctly, you'll need to check that you opened the right file and that you wrote the HTML correctly.

As you can see from the missing closing tags, the HTML is not nearly as strict as XHTML, and the first line (the doctype or DTD) is simple to use and easy to remember.

Using HTML5 with iOS and Android Devices

Many designers are reluctant to get started using HTML5 on their web pages because Internet Explorer has relatively little support for it. In fact, only Internet Explorer 9 and 10 have decent HTML5 support. Other computer browsers, such as Firefox, Chrome, Opera, and Safari, all have good support for most HTML5 features.

Watch Out!

Testing Is Critical

If you plan to create pages and applications for iOS and Android devices as well as desktop browsers, always test your documents in Internet Explorer 8. Internet Explorer 8 and 9 still have a large share of the browser market, and if your page or application doesn't work with it, your page or application won't work for most people browsing the web. If you don't have a Windows machine you can use an online tool such as Browsershots (http://browsershots.org/) or IE NetRenderer (http://netrenderer.com/) to test in Internet Explorer and other browsers. You will learn more about testing in Hour 4, "Detecting Mobile Devices and HTML5 Support."

But what about mobile devices running on Android and iOS, such as a Samsung Galaxy Tab or iPad? They all come with HTML5 support out of the box because they each run a browser (Safari on iOS and Chrome on Android) based on WebKit, which has excellent support for HTML5.

The best thing about designing web pages and applications using HTML5 for Android and iOS is that what you are creating will work on future devices. Right now operating systems exist that run on tablets and phones and to some extent televisions. But these operating systems are moving into other devices such as cars, picture frames, and even refrigerators.

Writing Mobile Websites

In some ways, writing websites for mobile devices is a lot easier than it used to be. Although a lot more devices are out there, including smartphones and not-so-smart phones, tablets, internet TV devices, and even some picture frames, the devices are converging in what HTML5 features they support, and even in their sizes and shapes (to some extent).

When you're creating a mobile website, the first thing to remember is that a mobile website is just a website. The best websites are built for every browser and operating system, or as many as possible.

However, you should still consider some basic questions when building a website that is intended for mobile devices:

- What is the screen size and resolution of the mobile device?
- What content do your mobile users need?
- Is your HTML, CSS, and JavaScript valid and compact?
- Should your site have a separate domain for mobile users?
- What testing does your mobile site need?

What is the Screen Size and Resolution of the Mobile Device

When you're working with mobile devices, obviously the screen size is going to be smaller than on a desktop. In general, with smartphones, you have to prepare for a few standard sizes:

- **128 x 160 pixels**—Phones such as the Fujitsu DoCoMo F504i
- **176 x 220 pixels**—Phones such as the HP iPAQ 510
- **240 x 320 pixels**—Smartphones such as Blackberry 8100 or the HTC Elf
- **320 x 480 pixels**—PDAs such as the Garmin-AsusA50 or the Palm Pre

Tablets add to the mix by having not only an increased screen size, but also having a variation in how they can be viewed. For example, most tablets (and some smartphones for that matter) can be viewed in portrait or landscape mode. There are also several sizes for tablets:

- **1024 x 768**—iOS tablets like iPad mini and iPad 2
- **1280 x 800**—Smaller tablets like Kindle fire
- **1920 x 1200**—For tablets like the Transformer Pad Infinity
- **2048 x 1536**—For iPad retina
- **2560 x 1600**—Larger tablets like the Nexus 10

In general, the tablets provide a lot more screen space for you to play with on mobile devices. You can assume you have at least 1024–1280 pixels by 600–800 pixels for tablet devices.

Browsing most websites in their standard format on an iPad is easy because the browser is as clear and easy to use as on a computer monitor. Plus, with the zooming capabilities on both iOs and Android, making small, harder-to-read areas bigger is easy.

What Content Do Your Mobile Users Need?

When you are designing a site for mobile devices, remember that users don't always want to access the same content as someone browsing on a desktop.

For example, mobile customers are often, well, mobile. In other words, they may be in motion or away from their home or office and have a very specific need or desire when they visit your site. For example, when visiting a restaurant website on a mobile phone, a user riding in a car might need to quickly find the location of the restaurant and the phone number. If the mobile site doesn't have the phone number and location front-and-center, the user will quickly give up on the site.

Watch Out!

Don’t Limit the Content

One thing mobile sites often get wrong is that they remove content from the mobile version of the site. Adjusting the content so that information that is most important to mobile users is easily available is essential. But if the content they need isn’t on the mobile site, you must allow the user the opportunity to look for the content on the full site.

Content for mobile sites shouldn’t be limited, however. In fact, the W3C recommends “…making, as far as is reasonable, the same information and services available to users irrespective of the device they are using.”[1]

This doesn’t mean that you can’t change the format or location of your content, but getting to the same content on a mobile device as on a computer should be possible.

Is Your HTML, CSS, and JavaScript Valid and Compact?

You don’t have to worry about writing well-formed XHTML for mobile devices, but sticking to correct, standards-based HTML, CSS, and JavaScript ensures that your pages are visible by the largest number of devices. Plus, by validating your HTML, you will know it is correct.

Did you Know?

The W3C Validator

The W3C has a validator located at http://validator.w3.org/ that you can use to check HTML, XHTML, and other markup languages. But you can also validate CSS and RSS, and even find broken links on your pages from this site. Don’t be afraid to check your site in the validator periodically. You may be surprised at what you find.

Beyond writing valid HTML, you should consider avoiding a few things if you are writing web pages for mobile devices:

- **HTML tables**—Avoiding tables as much as you can in mobile layouts is best because of the small size of the screen. Scrolling horizontally is difficult and makes the tables hard to read.
- **HTML tables for layout**—You shouldn’t use HTML tables for layout of web pages in general, but on mobile devices they can make the pages load slower and look bad, especially if the table doesn’t fit in the browser window. Plus, when you use tables for layout, you almost always use nested tables, which make the pages load slower and are much more difficult for mobile devices to render.

[1] Mobile Web Best Practices. www.w3.org/TR/mobile-bp/#OneWeb.

- **Pop-up windows**—Pop-up windows are often annoying in general, but on mobile devices they can make the site unusable. Some mobile browsers don't support them and others open them in unexpected ways (often by closing the current window to open a new one).
- **Graphics for layout**—Like using tables for layout, adding invisible graphics to add spaces and affect layout typically make many older mobile devices choke or display the page incorrectly. Plus, they add to the download time.
- **Frames and image maps**—Many mobile devices don't support these features in HTML. In fact, HTML5 no longer includes frames (other than the iframe) as a part of the specification because of the usability issues involved.

By the Way

Fewer Limitations for iOS and Android

Although avoiding tables, popup windows, and image maps in mobile pages is best, if you are focusing on mobile pages for iOS or Android, you can rest easy. Both of these handle them without trouble. Frames, however, are not part of HTML5, and you should not rely on their being supported in iOS or Android.

Also remember that mobile users often have to pay a fee for their bandwidth, so your web pages should be as small (in KB) as you can make them. The fewer HTML tags and CSS properties you use and server requests you make, the better browsing will be for mobile users.

Should Your Site Have a Separate Domain for Mobile Users?

Many websites have a separate subdomain for their mobile site. This makes finding the mobile site without having to bother with the regular domain easy for mobile users. These domains are typically something like `m.example.com`.

Having a separate mobile domain offers several advantages:

- It makes your mobile site easier to find.
- You can advertise the mobile URL separately from the normal URL, giving you more reach.
- Having a separate mobile domain enables people on tablets or smartphones to switch to the full site just by switching domains.
- Detecting mobile users and sending them to a separate domain is much easier than scripting changes to your CSS for mobile users.

When trying to decide how to handle your mobile site version, consider how you are going to maintain the site. You can create the mobile domain manually with completely separate pages, or you can use a content management system. Another popular solution for mobile devices is to use responsive design with CSS. Hour 4, "Detecting Mobile Devices and HTML5 Support," covers this topic in more detail.

What Testing Does Your Mobile Site Need?

Be prepared to test your site on as many mobile devices as you possibly can. Although you can use your browser to test or emulate things such as screen size, you won't see some of the horrible things that can go wrong if you don't test on mobile devices directly, such as the following:

- Packet size limitations by mobile carriers preventing your page or images from loading
- Images loading incorrectly or not at all
- Inability to scroll horizontally (which is nearly impossible to do on some phones)
- Device-specific features not working that you were relying on
- File formats not being supported

You likely don't have an unlimited budget for buying mobile phones (and their associated cellphone plans), so what do you do? Here are some suggestions:

- **Use emulators**—Both online and offline emulators are available for many different mobile devices. Most are free to use and they give you at least a start at testing your site for mobile devices.
- **Rent time on devices**—Services exist that will allow you to rent time on multiple phones. You can justify the cost by getting a sense of how the phones handle your applications.
- **Buy some phones**—This might seem like an expensive option, but if you are planning on doing a lot of mobile web development, it could be a reasonable investment.
- **Get help from friends and co-workers**—This is one of the least expensive ways to test your site—simply ask to borrow phones or tablets from people you know. You just need to put your site on a live web server.

Ultimately, if you are going to do mobile development, you should have at least one mobile device you can test your pages on directly. The more devices you can test on, the better your sites will be.

Summary

In this hour, you have learned how HTML started and the reasons for the move from HTML to XHTML to HTML5. You know the basic differences between HTML 4, XHTML 1, and HTML5 as well as what web applications are and how they relate to the Open Web Standard. You learned how to write a basic HTML web page and why HTML5 fits in so well with mobile devices. You also learned some powerful tips for building mobile web pages.

The most important things to remember from this hour are the best practices for building a website for mobile users:

- Focus first on building a website or application for every user, not just mobile ones.
- Determine who your mobile users are and what they are looking for, and present that content to them first, but not only that content.
- Build a site that uses standards-based technology so that your applications are more future proofed.
- Always test your sites and applications in as many web browsers and devices as you can.

Q&A

Q. ***I am not familiar with HTML, and I'm worried that I will have trouble building an HTML5 application. Do I need to know HTML 4 before I learn HTML5?***

A. Although knowing HTML 4 will make moving forward easier for you, learning HTML5 is a fairly straightforward process. Although this book focuses mostly on HTML5, by copying the code samples provided and looking at the source files for the companion website (www.html5in24hours.com/), you should be able to figure it out.

Q. ***I already have a website, and I want to make sure that mobile users can get the most out of it. How do I make sure that I am providing what mobile users need?***

A. The best way to do this is to ask them. Surveys asking your customers how they access your site and what parts are most useful to them are a good indicator of what they want. But you can also look at your web statistics. If you don't have analytics on your website, I recommend installing one such as Google Analytics or Piwik to track what people are looking at on your site. After you know what the popular pages are, you can ensure those pages are easy to access in your mobile version.

You can also use your web analytics to see what browsers (Firefox, IE, Chrome, etc.) are visiting your website and how your customers use the site (pages they click on, where they leave, and so on). With this method, even if you can't get direct customer feedback, you can see what features they are currently using and adjust your site accordingly.

Q. ***You mentioned using a content management system for maintaining a mobile site. Do you have any you can recommend?***

A. I use WordPress with the WordPress Mobile Pack to maintain a lot of sites for mobile and non-mobile users.

Workshop

The workshop contains quiz questions to help you process what you've learned in this chapter. Try to answer all the questions before you read the answers. See Appendix A, "Answers to Quizzes," for answers.

Quiz

1. Who decided to initiate HTML5 and why?
2. What is the first line in an HTML5 document?
3. Why is HTML5 so well suited to mobile application development?

Exercises

1. Create an HTML5 page. Your page should have a title, headline, and at least two or three paragraphs of text in it as well as the HTML5 doctype. Hour 3, "Styling Mobile Pages with CSS3," covers more about how to use CSS for styling the layout and look of your page.
2. Start planning your mobile website. List the content that you have that is particularly well suited to mobile users. List the content you need to create for mobile and non-mobile users. Ask yourself whether you have the resources to maintain a completely separate mobile site or whether you are going to use a content management system to maintain it.

This page intentionally left blank

Index

A

<a> tag, 26, 294
- block-level linking, 295-296
- changes to, 294-295
- link types, 300
- placeholder links, 296-297

<abbr> tag, 27, 155
about attribute, 269
absolute positioning, 38
accept attribute, 333
accessibility validators, 121
<acronym> tag, 27, 155
add() method, 364
addColorStop property, 168
adding
- alerts with jQuery, 73-75
- audio streams, 210
- colors and background images, 93-94
- HTML5 features as extras, 113-115
- JavaScript to web pages, 68-69
- meta tags to mobile web applications, 96-99
- microformats to websites, 266-267
- music files, 212
- shadows to text, 178-179
- video
  - with Flash fallback, 216-217
  - video streams, 210

<address> tag, 26, 152
ADT plug-in, 410
advertising, targeted, 389
Alchemy WebSockets, 325
alerts, adding with jQuery, 73-75
alignment, 191-193
alternate link type, 299-301
Amazon Appstore, 419
Amazon.com website, 108
Android
- emulator for, 134
- HTML5 compatibility, 6-7
- HTML5 support, 28, 50-52
- meta tag support, 98
- video, troubleshooting, 211

Android apps, building
- in PhoneGap, 411-413
- necessary tools, 410

Android devices, putting applications on, 418
Android icons, 416

Android SDK, 410
Android tablets, 126
animated URLs, 384
APIs
  for web applications, 310
  in HTML5, 29
App Store icon, 414
Appcelerator, 409
Apple App store, submitting applications to, 418
Apple icon, 415
Apple iOS developer accounts, 416
apple-mobile-web-app-capable meta tag, 96
apple-mobile-web-app-status-bar-style meta tag, 97
apple-touch-icon meta tag, 98
apple-touch-icon-precomposed meta tag, 98
apple-touch-startup-image meta tag, 97
<applet> tag, 27
application cache
  debugging, 351-352
  lazy cache setup, 346-347
  listening for changes, 348-350
application icons, 414-416
application programming interfaces. *See* APIs
applications
  mobile web applications
    applying CSS, 91-96
    building HTML layout, 89-91
    changing the visual design, 124-130
    checking for HTML5 and CSS3 support, 130-132
    compatibility with older browsers, 134-138
    designing, 85-86
    evaluating content, 123-124
    meta tags, adding, 96-99
    multiple device support, 132-133
    optimizing, 99-102
    planning, 86-88
    testing, 63-64, 121-123, 133-134
    web editors, choosing, 120-121
  offline web applications, 113-115
  putting on Android devices, 418
  putting on iOS devices, 416-417
  selling, 418-419
  submitting to Apple App Store, 418
  testing, 416
  web applications
    popularity of, 30
    versus desktop applications, 4
applications. *See* web applications
Appstore Developer Portal, 419
arc method, 175-177
<area> tag, 294, 297, 300
<article> tag, 16, 144
<aside> tag, 16, 144-145
aspect ratio, checking, 62
aspect-ratio keyword, 62
attaching
  events to scripts, 165
  scripts to events, 312
  style sheets for mobile devices, 61-62
attributes
  <a> tag, 294-295
  <area> tag, 297
  audio and video attributes, 213
  changes from HTML 4, 25-27, 442
  data-* attributes, 318-320
  for drag and drop, 279-280
  event attributes, 24-25
  event-handler attributes, 443
  global attributes
    for links, 297-298
    list of, 25, 442
  list of new, 24-25
  quotation marks and, 114
  RDFa (Resource Description Framework in attributes), 269-270
  source attributes, 214-215
  track attributes, 215
  video attributes, 213-214
audio
  embedding, 21
  audio codecs, browser support, 208-209
    MP3, 209
    Vorbis, 209
    WAV, 209
  audio streams, adding, 210
<audio> tag, 19, 21, 116, 210
  adding music files with, 212
  advantages and disadvantages of, 205-206
  attributes, 213
  fallback options, 216-217
  support for, 113
author link type, 299-301
autocomplete in forms, 227-228
autocomplete attribute, 227-228
autofocus in forms, 226
autofocus attribute, 115, 226
autoplay attribute, 213

B

<b> tag, 26, 155
background colors, rotating, 314-315
background images, adding, 93-94
background property, 94
background threads, 326-332
background-color property, 94, 125
background-image property, 94
background-position property, 94
background-repeat property, 94
base font size, setting, 189-191
<basefont> tag, 27
baselines for non-English text, 177
<bdi> tag, 23
beginPath() method, 171
benefits
  of HTML5 apps, 407-408
  of native apps, 406-407
Berners-Lee, Tim, 1
best practices, optimizing mobile web applications, 99-102
<big> tag, 27
bind() method, 312
Bing Maps API, 396
birthday application example, 365-370
Blackberry, emulator for, 134
blob (binary large object), 332
Blob interface, 332
block-level elements, linking, 295-296
<blockquote> tag, 149
blocks of text, color of, 125
blogs, building with sectioning elements, 152-154
<body> tag, 149
bookmark link type, 299
bookmarking
  without History API, 376-379
  scrolling pages, 376
books for HTML5 information, 447-448
border property, 26, 38
border-bottom-left-radius property, 43
border-bottom-right-radius property, 43
border-color property, 125
border-radius property, 42-43
border-top-left-radius property, 43
border-top-right-radius property, 43
borders
  drawing, 181-182
  for <canvas> tag, 164
box model (CSS), 38-39
box-shadow property, 45
Bräunlich, Philip, 219
browser extensions for placeholder text, 225
browser history
  bookmarking without History API, 376-379
  changing URLs with History API, 379-380
  History API
    animated URLs, 384
    on multiple domains, 383
    phishing schemes with, 383-384
  History API browser support, 375-376
  History API methods, 375
  photo gallery example, 380-383
  reasons for controlling, 373-374
browser prefixes, Indexed Database API and, 364
browser sniffing, 318
browser-specific keywords, 42
BrowserCam, 134
browsers
  <audio> and <video> tag fallback, 216-217
  audio codec support, 208-209
  browser support for data-* attributes, 319
  <canvas> tag support, 164-165
  checking local storage support, 358
  compatibility with older browsers, 134-138
  drag-and-drop support, 276-277
  draggable and dropzone attributes, 279
  editing content in
    contenteditable attribute, 245-247
    execCommand() method commands, 247-251
    spellcheck attribute, 251-252
    support for, 255-257
  evaluating usage, 109-110
  File API support, 333
  geolocation, fallback options for older browsers, 393-394
  Geolocation API support, 388
  growth in mobile web browsing, 110-111
  History API support, 375-376
  history of, 2
  HTML5 support, 50-52, 58-59, 112-113
  offline web application support, 342
  spellcheck support, 252
  support for current standards, 106-108

testing in, 7
video codec support, 206-208
web applications and, 4
Web SQL Database API support, 361
WebSockets API support, 323
WOFF (Web Open Font Format) support, 201-202

Browsershots, 7, 64

browsing contexts, 295

business marks, 197

C

cache manifest, 343. *See also* application cache
lazy cache setup, 346-347
usage of, 343-344
writing, 344-346

cached event, 347

calculator example, 327-332

calling scripts at page load, 165

canonical link type, 301

<canvas> tag, 19-21, 116, 127
borders, setting, 164
browser support for, 164-165
clearing shapes, 166
drawing images, 179
creating patterns with, 181-182
scaling and clipping images, 179-181
drawing shapes, 165-166
circles, 175-177
polygons and lines, 171-175
rectangles and squares, 166-171
drawing text, 177-179
how to use, 163-165
support for, 113
SVG and Flash versus, 182-183

Cascading Style Sheets. *See* CSS

case for tags, 28

CDN (content delivery network), 80

cell phones. *See* mobile devices

<center> tag, 27

changeVersion() method, 362

character encoding, declaring, 111

character entities
for advanced punctuation, 198
for business marks, 197
for quotation marks, 197

checked attribute, 24

checking event, 347

choosing
HTML5 tags to use, 50
web editors, 120-121

circles
converting square <div> tags to, 128
drawing, 175-177

<cite> tag, 26

Clarke, Andy, 137

clear() method, 360

clearing shapes, 166

clearInterval() method, 313

clearRect function, 167

clearTimeout() method, 313

clearWatch() method, 389-391

click events in iOS, 313

click-to-call links, 100

click-to-talk links, 100

clicking links, 73-75

client-side file handling, File API, 331-333
browser support, 333
building input field, 333
drag-and-drop file uploader example, 335-337
getting file information, 334
reading file information, 334-335

clients, WebSockets API, 325

clipping images, 179-181

closePath() method, 171

closing windows with JavaScript, 71

cloud scripts, speed of, 80

<code> tag, 155

code validators, 121

codecs
audio codecs, browser support, 208-209
video codecs
browser support, 206-208
checking for support, 55-56
WebM versus H.264, 54

color keyword, 62

color property, 94, 125

color-index keyword, 63

colors
adding, 93-94
background colors, rotating, 314-315
of blocks of text, 125
for <canvas> tag, 168-171
checking, 62
color types, 233-235
in mobile designs, 125-126
transparent colors, creating, 45
web safe colors, 198

ColourLovers web site, 93

<command> tag, 21, 237, 298
commands
execCommand() method commands, 247-251
menu commands, 236-237
comments, conditional, 57
common fonts, 37
communication, WebSockets API, 323
benefits of, 324-325
browser support, 323
establishing, 325-326
security, 324
comparing HTML5 apps with native apps, 405-406
compatibility
of HTML5 with iOS and Android, 6-7
of mobile web applications with older browsers, 134-138
compatibility mode (Internet Explorer), 106
competitors, planning web applications, 87
conditional comments, 57
console icon, 414
contacts, hCard microformat, 261-267
content
editing
browser support, 255-257
contenteditable attribute, 245-247
execCommand() method commands, 247-251
spellcheck attribute, 251-252
evaluating for mobile devices, 123-124
hiding with CSS, 70
what to include, 8-9
content delivery network (CDN), 80
contenteditable attribute, 25, 245-247
browser support, 255-257
support for, 112
context, setting, 166
contextmenu attribute, 25, 297-298
controls, custom video controls, 217-219
controls attribute, 211-213
converting
microformats to microdata, 268-269
to native apps, 408-413
to new layout tags, 16-18
square <div> tags to circles, 128
cookies
disadvantages of, 355
managing with JavaScript, 71-72
web storage versus, 356-357
coords.accuracy property, 390
coords.altitude property, 390
coords.altitudeAccuracy property, 390
coords.heading property, 390
coords.latitude property, 390
coords.longitude property, 390
coords.speed property, 390
copyright symbols, 197
created elements, checking properties of, 53-54
createLinearGradient property, 168
createPattern() method, 181-182
createRadialGradient property, 168
creation callback, 362
cross site scripting (XSS), 361
CSS (Cascading Style Sheets), 33
box model, 38-39
browser extensions for placeholder text, 225
content, hiding, 70
drop zones, highlighting, 282
embedded styles, 34
extensions for drag and drop, 280
external styles, 35-36
font properties, 188-190
font styles, setting, 190-191
@font-face feature, 199-201
hyphenation properties, 195
inline styles, 34
in mobile web applications, 91
colors and background images, adding, 93-94
layout, styling, 94-96
type, changing, 91-93
rollovers, creating, 69-71
style sheets
creating, 34-36
fonts, changing, 36-37
resetting styles, 40
web page layout, 38-41
styling canvas fonts, 178
typography. *See* typography
widows and orphans properties, 196
CSS3, 115
checking for support of, 130-132
media attribute extensions, 60-61
aspect ratio, 62
colors, 62
resolution, 63
screen orientation, 62
tv and grid devices, 63
width and height, 61-62

How can we make this index more useful? Email us at indexes@samspublishing.com

on mobile devices, 46
new features, 42
rounded corners, 42-44
shadows, 45
transparent colors, 45
curly quotes, 196
custom fonts, 199-201
custom video controls, creating, 217-219
customers, planning web applications, 87

D

data storage. *See* web storage
data types, SQLite and, 363
data-* attributes, 318-320
databases. *See* SQL databases
<datalist> tag, 21, 227-228
datalists in forms, 227-228
dataset property, 319
datasets, 112, 318-320
dataTransfer object, 278-279, 283
date and time types, 231-232
datetime attribute, 158-159
debugging application cache, 351-352
declaring character encoding, 111
default attribute, 215
default font size, changing, 92
degrading gracefully, 58-59
degrees, radians versus, 175
<del> tag, 155
deleting cookies with JavaScript, 71-72
design
changing for mobile devices, 124-130
elements of, 125-129
mobile design patterns, 129-130
mobile web applications, 85-86
designmode attribute, 246
desktop applications, web applications versus, 4
desktop emulators, 134
<details> tag, 18, 149
detecting
HTML5 functions, 52-53
with created element properties, 53-54
with element value retention, 56-57
with global element properties, 53
graceful degradation, 58-59
with Internet Explorer previous versions, 57-58
with method values, 54-56
mobile browsers, 60-61
aspect ratio, checking, 62
color, checking, 62
resolution, checking, 63
screen orientation, checking, 62
tv and grid devices, checking, 63
width and height, checking, 61-62
device-height keyword, 61
device-height property, 99
device-width keyword, 61
device-width property, 99
DeviceAnywhere, 134
<dfn> tag, 155
DHTML (Dynamic HTML), 276
<dir> tag, 27
direction in mobile designs, 125
disabling user interface, 247
display property, visibility property versus, 70
<div> tag, 16
converting from square to circle, 128
converting to WYSIWYG editor, 246-247
when to use, 149
<dl> tag, 26
doctype, 4, 28, 114
Document Object Model (DOM), 53
document.queryCommandEnabled() command, 250
document.queryCommandIndeterm() command, 250
document.queryCommandState() command, 250
document.queryCommandSupported() command, 250
document.queryCommandValue() command, 250
Dojo Toolkit, 356
DOM (Document Object Model), 53
custom video controls, 217-219
DOM manipulation, Web Workers API and, 326
events and properties for offline web applications, 347-348
storage. *See* local storage
domains
for mobile device websites, 10
multiple domains, History API and, 383

dotMobi Emulator, 134
downloading event, 347
drag and drop
attributes, 279-280
browser support, 276-277
building, steps for, 280-281
CSS extensions, 280
dataTransfer object, 283
drag events, setting on draggable objects, 281
draggable objects, defining, 281
drop zones, 282-283
effect, defining, 283-284
events, 276-279
explained, 275-276
in iOS, 287
building, 289-290
draggable elements, creating, 287-288
drop zones, creating, 288
puzzle example, 284-287
support for, 112
drag event, 276-278, 281
drag() function, 281
drag-and-drop file uploader, 335-337
dragend event, 277
dragenter event, 276
draggable attribute, 25, 279
draggable objects
creating in iOS, 287-288
defining, 281
setting drag events on, 281
dragleave event, 276
dragover event, 276, 282
dragstart event, 276
drawImage function, 179-181
drawing. ***See also*** **<canvas> tag**
borders, 181-182
images on canvas, 179
creating patterns with, 181-182
scaling and clipping images, 179-181
shapes, 165-166
circles, 175-177
polygons and lines, 171-175
rectangles and squares, 166-171
text on canvas, 177-179
Dreamweaver, 120
drop event, 277
drop zones
creating in iOS, 288
defining, 282
highlighting, 282
setting events on, 282-283
drop() function, 282
dropzone attribute, 25, 279
Dynamic HTML (DHTML), 276

E

e.preventDefault() function, 283
Eclipse Classic IDE, 410
editing content
browser support, 255-257
contenteditable attribute, 245-247
execCommand() method commands, 247-251
spellcheck attribute, 251-252
effects in drag and drop, defining, 283-284
elements. ***See also*** **tags**
of design, 125-129
list of, 437
ellipses, 197
em dashes, 197
<em> tag, 155
email types, 233-235
<embed> tag, 20-21
embedded styles, 34
embedding audio/video, 21
emphasized text in screen readers, 156
ems
as relative font sizes, 192
defined, 92
en dashes, 197
enableHighAccuracy property, 392
error event, 351
evaluating
browser usage, 109-110
web application content for mobile devices, 123-124
event attributes, 24-25
event handlers
for user interface components, 253-255
list of, 310-313
event-handler attributes (HTML5), 443
events
attaching scripts to, 165, 312
click events, 313
DOM events for offline web applications, 347-348
for drag and drop, 276-279
drag events, setting on draggable objects, 281
hCalendar microformat, 262-264
local storage, 361

for mobile devices, 116
offline web applications, 342
on drop zones, setting, 282-283
WebSockets API, 326
execCommand() method
commands supported by, 247-251
disabling user interface, 247
executeSql() method, 362
explicit section of cache manifest, 344
eXtensible Markup Language (XML), 2
external link type, 299
external resources, hyperlinks versus, 294
external styles, 35-36

F

fallback section of cache manifest, 346
fat footers, 94
fault tolerance in JavaScript, 69
<fieldset> tag, 149
<figcaption> tag, 16-18, 114, 149
<figure> tag, 16-18, 114, 149-150
File API, 331-333
browser support, 333
building input field, 333
drag-and-drop file uploader example, 335-337
getting file information, 334
reading file information, 334-335
file extensions of HTML files, 6
File interface, 332
FileError interface, 332
FileException interface, 332
FileList interface, 332-334
FileReader interface, 332
fill() method, 171
fillRect function, 167
fillStyle property, 168, 178
fillText() method, 177
finding text editors, 5
Firefox extensions for border-radius property, 43
Flash
<canvas> tag and SVG versus, 182-183
as video fallback, 216-217
floating positioning, 38
Flowplayer, 216
font property, 37, 177
Font Squirrel @Font-Face Generator, 202
font stacks, 37
<font> tag, 27
font-effect property, 190
font-emphasize property, 190
font-emphasize-position property, 190
font-family property, 36, 188
@font-face feature (CSS), 199-201
font-size property, 36, 188
font-size-adjust property, 189
font-smooth property, 190
font-stretch property, 189
font-style property, 36, 188
font-variant property, 37, 188
font-weight property, 36, 188
fonts. *See also* typography
changing, 36-37, 91-93
CSS font properties, 188-190
default size, changing, 92
drawing on canvas, 177-179
licensing, 200
on mobile devices, 92
relative font sizes, 192
setting base size, 189-191
styles, 190-191
typefaces versus, 188
web safe fonts, 198
WOFF (Web Open Font Format)
advantages and disadvantages of, 198-199
browser support for, 201-202
@font-face feature (CSS), 199-201
Fontspring @Font-Face Syntax, 202
footer elements, 150-152
<footer> tag, 16, 151
form field labels, placeholder text as, 224
form tags, 21-22
format-detection meta tag, 97
formatting. *See* styling
forms
creating, 239-241
HTML5 features, 114
input types, 115
date and time types, 231-232
email, URLs, colors, search box types, 233-235
number types, 228-231
new features, 223
autocomplete and datalists, 227-228
autofocus, 226
key generation form field, 235-236
menu commands, 236-237

<output> tag, 236
placeholder text, 224-226
validating
with JavaScript, 71
with jQuery, 76-78
validation, 237-239
<frame> tag, 27
frames, lack of support for, 10
<frameset> tag, 27
frameworks for multiple device support, 132-133
fraud detection, geolocation, 389

G

gaming, geolocation, 389
geo microformat, 262
geo.js, 394
geo.js script, 393
geolocation. *See* Geolocation API
Geolocation API, 115, 387-388
browser support, 388
fallback options for older browsers, 393-394
mapping application example, 397-400
methods, 389
positionOptions object, 392-393
privacy and, 394-396
using, 389-391
watchPosition(), 393
GeoTargeting by IP address, 396
getContext() method, 166
getCurrentPosition() method, 389, 391
getItem() method, 360
gift status in online store, saving, 359-360
GitHub, 376
global attributes
for links, 297-298
list of, 25, 442
global elements, checking properties of, 53
goals, SMART, 87
Google
Google Analytics, 110
Google Gears, 393
Google Maps JavaScript API, 397
Google Website Optimizer, 109
new doctype on, 114
GotProject, 287
GPS (global-positioning system), 388, 392
graceful degradation, 58-59, 108
gradients for <canvas> tag, 168-171
grid devices, checking, 63

H

<h#> tags, 150
<h1> tag, 146
H.264 video codec, 207-208
checking for support, 55-56
WebM versus, 54
half circles, drawing, 175-177
handheldFriendly meta tag, 96
hashtags, 272
hCalendar microformat, 262-264
hCard microformat, 261-267
<head> tag, 26
header elements, 150-152
<header> tag, 16, 151
heading elements, 150-152
headlines with sectioning elements, 146
height, checking, 61-62
height keyword, 61
height property, 98, 214
help link type, 299
<hgroup> tag, 16, 150-151
hidden attribute, 25, 252, 297-298
hiding
content with CSS, 70
elements, 252
high attribute, 157
highlighted text, 156-157
highlighting drop zones, 282
History API
animated URLs, 384
bookmarking pages without, 376-379
browser support, 375-376
changing URLs with, 379-380
methods, 375
on multiple domains, 383
phishing schemes with, 383-384
photo gallery example, 380-383
reasons for using, 373-374
support for, 113
history
of browsers, 2
of HTML5, 1-3
history.back() method, 375
history.forward() method, 375
history.go() method, 375
history.js script, 376
home pages, redirection to, 100
<hr> tag, 26, 155
hRecipe microformat, 262-265
href attribute, 270, 296
hreflang attribute, 297
hReview microformat, 262-265

HTML
- deciding when to upgrade, 105-106
  - browser support for standards, 106-108
  - browser usage, 109-110
  - growth in mobile web browsing, 110-111
  - iterative design, 108-109
- entities. *See* character entities
- mobile web applications, building, 89-91
- storing data in, disadvantages of, 355

HTML 4
- changes to tags and attributes from, 25-27
- tags changed to semantic tags, 155-156
- HTML5 versus, 3
- semantic tags, 155
- syntax changes in HTML5, 27-28

HTML validators, 121

HTML5
- APIs in, 29
- attributes, 442
  - event-handler attributes, 443
  - global attributes, 442
- browser support for, 50-52
- changes to HTML 4 tags and attributes, 25-27
- checking for support of, 130-132
- choosing elements to use, 50
- compatibility with iOS and Android, 6-7
- detecting, 52-53
  - with created element properties, 53-54
  - with element value retention, 56-57
  - with global element properties, 53
  - graceful degradation, 58-59
  - with Internet Explorer previous versions, 57-58
  - with method values, 54-56
- drag and drop
  - attributes, 279-280
  - browser support, 276-277
  - building, steps for, 280-281
  - CSS extensions, 280
  - dataTransfer object, 283
  - drag events, setting on draggable objects, 281
  - draggable objects, defining, 281
  - drop zones, 282-283
  - effects, defining, 283-284
  - events, 276-279
  - explained, 275-276
  - in iOS, 287-290
  - puzzle example, 284-287
- tags. *See* tags
- forms. *See* forms
- history of, 1-3
- HTML 4 versus, 3
- links. *See* links
- media elements, 210-212
  - audio and video attributes, 213
  - source attributes, 214-215
  - track attributes, 215
  - video attributes, 213-214
- mobile support for, 28-30
- new features, 318-320
- Open Web Standard specifications, 4-5
- resources for information, 447-448
- semantic elements, 155-159
- syntax changes, 27-28
- upgrading to
  - adding features as extras, 113-115
  - browser support for, 112-113
  - features to start with, 111-112
  - mobile device-specific features, 115-116
- user interface components, 253-255
- web pages, writing, 5-6
- <audio> and <video> tags
  - advantages and disadvantages of, 205-206
  - fallback options, 216-217

HTML5 apps
- benefits of, 407-408
- comparing with native apps, 405-406

HTML5 Doctor website, 448

The HTML5 Gallery website, 448

HTML5 Outliner website, 143

HTML5 Rocks website, 448

html5shiv script, 57-58, 131, 137

HttpOnly flag, 357

hyperlinks, external resources versus, 294

hypertext, 1

hyphenate-character property, 195

hyphenate-limit-chars property, 195

hyphenate-limit-last property, 195

hyphenate-limit-lines property, 195
hyphenate-limit-zone property, 195
hyphenate-resource property, 195
hyphenation, 193-195
hyphens property, 195

I

<i> tag, 26, 155
icon link type, 299
icons, creating, 414-416
id attribute, 18, 27, 295
IE6 Universal CSS, 137
IETF (Internet Engineering Task Force), 2
image maps, 297
image rollovers, creating with jQuery, 75-76
images
- background images, adding, 93-94
- drawing on canvas, 179
  - creating patterns with, 181-182
  - scaling and clipping images, 179-181
- sizes, 415

Indexed Database API, 363-365
initial-scale property, 98
inline styles, 34
input fields for File API, 333
<input> tag, 22
- date and time types, 231-232
- email, URLs, colors, search box types, 233-235
- number types, 228-231

input types on forms, 115
<ins> tag, 155
international support tags, 22-23
Internet Engineering Task Force (IETF), 2
Internet Explorer
- detecting previous versions, 57-58
- HTML5 support, 50
- rich text editors, 256
- support for current standards, 106-108
- testing in, 7

The Internet Explorer 6 Countdown website, 107
iOS
- applications building
  - necessary tools, 409
  - in PhoneGap, 410-411
- click events in, 313
- drag and drop in, 287
  - building, 289-290
  - draggable elements, creating, 287-288
  - drop zones, creating, 288
- emulator for, 134
- file upload, lack of support for, 333
- HTML5 support, 6-7, 28, 50-52
- video, troubleshooting, 210

iOS devices, putting applications on, 416-417
IP addresses, geolocation, 388
iphone-photo-picker, 333
Irish, Paul, 201
itemid attribute, 267
itemprop attribute, 267
itemref attribute, 267
itemscope attribute, 267
itemtype attribute, 267
iterative design, 108-109

J

JavaScript
- adding to web pages, 68-69
- clearing shapes on <canvas> tag, 166
- cookies, managing, 71-72
- datasets and, 320
- defined, 67
- drawing shapes on <canvas> tag, 165-166
  - circles, 175-177
  - polygons and lines, 171-175
  - rectangles and squares, 166-171
- fault tolerance, 69
- forms, validating, 71
- jQuery, 72-75
  - alerts, adding, 73-75
  - cloud scripts, speed of, 80
  - date pickers, 232
  - as fallback for autocomplete and datalists, 228
  - forms, validating, 76-78
  - image rollovers, creating, 75-76
  - web applications, advantages of jQuery, 78-79
- jQuery Mobile, 79-81
- rollovers, creating, 69-71
- windows, opening/closing, 71

Jo framework, 132
joins
- building lines with, 174-175
- types of, 173

jQTouch, 313
jQuery, 72-75
- alerts, adding, 73-75
- bind() method, 312
- cloud scripts, speed of, 80

date pickers, 232
as fallback for autocomplete and datalists, 228
forms, validating, 76-78
image rollovers, creating, 75-76
web applications, advantages of jQuery, 78-79
jQuery Mobile, 79-81, 132
jWebSocket, 325

K

Kaazing HTML5 WebSocket Gateway, 325
<kbd> tag, 155
kerning, 191-192
key generation form field, 235-236
key() method, 360
<keygen> tag, 21, 235-236
keywords, browser-specific, 42
kind attribute, 215
Komodo IDE, 120

L

label attribute, 215
<label> tag, 26
landscape mode, 126
language attribute, 27
language support tags. ***See*** **international support tags**
launch image, 414
launcher icon, 414
layout, styling, 38-41, 94-96
layout tags
converting to, 16-18
list of, 15-16
lazy cache setup, 346-347
leading, 191-193
length attribute, 360
letter-spacing property, 191-192
license link type, 299
licensing fonts, 200
ligatures, 191-193
line height, changing, 92
line-height property, 189, 193
linear gradients, 168
lineCap property, 173
lineJoin property, 173
lines
drawing, 171-175
in mobile designs, 127
lineTo() method, 171
lineWidth property, 173
link relationships, 299-300
alternate link type, 300-301
author link type, 301
prev and next link types, 304
sidebar link type, 305
stylesheet link type, 305
<link> tag, 35, 294, 300
link types
alternate link type, 300-301
author link type, 301
list of, 299-300
prev and next link types, 304
sidebar link type, 305
stylesheet link type, 305
links
adding alerts with jQuery, 73-75
hyperlinks versus external resources, 294
new features, 293-294
alternate link type, 300-301
author link type, 301
block-level element linking, 295-296
global attributes, 297-298
image maps, 297
link relationships, 299-300
placeholder links, 296-297
prev and next link types, 304
sidebar link type, 305
stylesheet link type, 305
<a> tag changes, 294-295
loading scripts, 21
Local Shared Objects in Flash, 356
local storage, 115
checking browser support, 358
cookies versus, 356-357
defining storage object, 358
explained, 356
interface methods, 360-361
plug-ins for, 356
saving gift status in online store example, 359-360
session storage versus, 357-358
local() command, 201
location services, turning off, 396
long-polling, 325
loop attribute, 213
low attribute, 157
Lynx, 2

M

managing cookies with JavaScript, 71-72
manifest attribute, 343
<map> tag, 297
mapping
- geolocation, 389
- mapping application example, 397-400

MapQuest Open API, 397
maps, static, 400
<mark> tag, 18, 114, 156-157
market share for browsers, 50-52
max attribute, 157
maximum-scale property, 98
maximumAge property, 392-393
measure (in typography), 191-193
media attribute, 60-61, 214, 294, 297
- aspect ratio, 62
- colors, 62
- resolution, 63
- screen orientation, 62
- tv and grid devices, 63
- width and height, 61-62

media elements (HTML5), 210-212
- audio and video attributes, 213
- source attributes, 214-215
- track attributes, 215
- video attributes, 213-214

media queries (CSS), 95-96
mediagroup attribute, 213
menu commands, 236-237
<menu> tag, 26, 147, 236-237, 298
<meta charset> tag, 23
meta tags, 96-99
<meter> tag, 18, 157
methods
- checking values of, 54-56
- Geolocation API, 389
- History API, 375
- local storage interface methods, 360-361

Meyer, Eric, 40
microdata
- converting microformats to, 268-269
- explained, 267-268
- microformats and RDFa versus, 270-271
- microformats versus, 267

Microformat website, 271
microformats
- adding to websites, 266-267
- benefits of, 263-264
- converting to microdata, 268-269
- examples of current, 264-265
- explained, 261-263
- microdata and RDFa versus, 270-271
- microdata versus, 267
- mobile devices and, 271-272
- principles for using, 264

MIME types
- for cache manifest, 343
- troubleshooting video support, 207

min attribute, 157
minimum-scale property, 98
miterLimit property, 175
mobile applications, testing, 63-64
mobile browsers, detecting, 60-61
- aspect ratio, checking, 62
- color, checking, 62
- resolution, checking, 63
- screen orientation, checking, 62
- tv and grid devices, checking, 63
- width and height, checking, 61-62

mobile design patterns, 129-130
mobile devices
- CSS3 on, 46
- events for, 116
- GPS hardware on, 392
- HTML5 support, 28-30, 115-116
- installed fonts, 92
- jQuery Mobile, 79-81
- microformats and, 271-272
- modal dialog support, 316
- rollovers, avoiding, 70
- style sheets, attaching, 61-62
- writing websites for, 7
  - content included, 8-9
  - on separate domain, 10
  - screen size/resolution, 8
  - testing, 11
  - validity and size of code, 9-10
  - what to avoid, 9-10

mobile home page, redirection to, 100
mobile validators, 121
mobile web applications
- applying CSS, 91
  - colors and background images, adding, 93-94
  - layout, styling, 94-96
  - type, changing, 91-93
- building HTML layout, 89-91

converting web applications to
changing the visual design, 124-130
checking for HTML5 and CSS3 support, 130-132
compatibility with older browsers, 134-138
evaluating the content, 123-124
multiple device support, 132-133
testing on multiple devices, 133-134
testing the application, 121-123
designing, 85-86
meta tags, adding, 96-99
optimizing, 99-102
planning, 86-88
web editors, choosing, 120-121
mobile web browsing, growth in, 110-111
mobileOptimized meta tag, 96
mobiReady, 121
mockups, planning web applications, 88
modal dialogs, 316
building, 316-317
mobile support for, 316
Modernizr, 130-132
modules (CSS), 42
monchrome keyword, 63
Mosaic, 2
moveTo() method, 171
MP3 audio codec, 209
.mp4 extension, 207-208
multimedia
<audio> and <video> tags
advantages and disadvantages of, 205-206
fallback options, 216-217
audio codecs, 208-209
custom video controls, 217-219
HTML5 media elements, 210-212
audio and video attributes, 213
source attributes, 214-215
track attributes, 215
video attributes, 213-214
video codecs, 207-208
multimedia tags, 19-21
multiple attribute, 333
multiple devices
supporting, 132-133
testing on, 133-134
multiple domains, History API and, 383
multiple sources for audio and video, 210
music files, adding, 212
muted attribute, 214

N

name attribute, 27, 294
nanoformats, 272
native apps
benefits of, 406-407
comparing with HTML5, 405-406
converting to, 408-413
options other than converting to, 419
<nav> tag, 16, 146-147
navigation, 146-147
navigator.geolocation, 389
navigator.onLine property, 348
nesting tags, 3
Nettuts+, 230
new features
in CSS3, 42
in HTML5, 318-320
links, 293-294
<a> tag changes, 294-295
alternate link type, 300-301
author link type, 301
block-level element linking, 295-296
global attributes, 297-298
image maps, 297
link relationships, 299-300
placeholder links, 296-297
prev and next link types, 304
sidebar link type, 305
stylesheet link type, 305
next link type, 299, 304
nofollow link type, 299
<noframes> tag, 27
non-breaking hyphens, 194
non-English text, baselines, 177
noreferrer link type, 299
<noscript> tag, 68-69
notifications, push, 406
noupdate event, 347
novalidate attribute, 237
number types, 228-231

O

object store, exposing with Indexed Database API, 363
<object> tag, 26-27, 216
obsolete tags and attributes, 27

octagons, drawing, 172
offline web applications, 115, 341-342
  browser support, 342
  cache manifest, 343
    lazy cache setup, 346-347
    usage of, 343-344
    writing, 344-346
  debugging application cache, 351-352
  DOM events and properties, 347-348
  events, 342
  listening for cache changes, 348-350
  support for, 113
  web storage and, 342
Ogg Vorbis audio codec, 209
ogg/Theora video codec, 208
onabort attribute, 24
onabort event handler, 253
onbeforeonload attribute, 24
onbeforeonunload attribute, 24
oncanplay event handler, 253
oncanplaythrough event handler, 253
onclose event, 326
oncontextmenu attribute, 24
oncontextmenu event handler, 253
ondrag attribute, 24
ondrag event handler, 253
ondragend attribute, 24
ondragend event handler, 253
ondragenter attribute, 24
ondragenter event handler, 253
ondragleave attribute, 24
ondragleave event handler, 253
ondragover event handler, 253
ondragstart attribute, 24
ondragstart event handler, 253
ondrop attribute, 24
ondrop event handler, 253
ondurationchange event handler, 253
onemptied event handler, 254
onended event handler, 254
onerror attribute, 24
onerror event handler, 254, 326
oninvalid event handler, 254
online emulators, 134
online stores, saving gift status example, 359-360
online whitelist of cache manifest, 345
onloadeddata event handler, 254
onloadedmetadata event handler, 254
onloadstart event handler, 254
onmessage attribute, 24
onmessage event, 326
onopen event, 326
onpause event handler, 254
onplay event handler, 254
onplaying event handler, 254
onprogress event handler, 254
onratechange event handler, 254
onreadystatechange event handler, 254
onresize attribute, 24
onscroll attribute, 24
onseeked event handler, 254
onseeking event handler, 254
onshow event handler, 254
onstalled event handler, 254
onsuspend event handler, 254
ontimeupdate event handler, 254
onunload attribute, 24
onvolumechange event handler, 255
onwaiting event handler, 255
opacity, 45
Open Web Standard, 4-5, 310
openCursor() method, 365
openDatabase() method, 362
opening new windows with JavaScript, 71
Opera Mini, 52
Opera Mini Simulator, 134
Opera Mobile, emulator for, 134
optimizing mobile web applications, 99-102
optimum attribute, 157
orientation keyword, 62
orphans, 196
orphans property, 196
outlines
  creating, 142-144
  testing, 143
<output> tag, 18, 236
OviMaps API, 397

P

page load, calling scripts at, 165
page locations of sectioning elements, 145
pattern attribute, 238
patterns
  creating with images, 181-182
  mobile design patterns, 129-130
pause() event, 218
PERMISSION DENIED (1) code, 390
persistent storage. *See* web storage
phishing schemes with History API, 383-384
PhoneGap, 132, 409
  Build, 410
  converting to native apps, 410-413

How can we make this index more useful? Email us at indexes@samspublishing.com

phones. *See* mobile devices
photo gallery example, 380-383
photo locator, 389
phpwebsocket, 325
picoformats, 272
pingback link type, 299
placeholder attribute, 224-226
placeholder links, 296-297
placeholder text, 114, 224-226
planning mobile web applications, 86-88
play() event, 218
plug-ins for local storage, 356
polling, 324
polygons, drawing, 171-175
popstate listener, 380
portrait mode, 126
POSITION UNAVAILABLE (2) code, 390
positionError object, 390
positionOptions object, 392-393
poster attribute, 210, 213
prefetch link type, 299
preload attribute, 213
preloading rollover images, 75
prev link type, 299, 304
printing as user prompt, 315
privacy, geolocation and, 394-396
progress event, 347
<progress> tag, 18, 157-158
progressive enhancement, 86, 108
properties
- DOM properties for offline web applications, 347-348
- of created elements, checking, 53-54
- of global elements, checking, 53

property attribute, 270
pseudo-classes, placeholder links and, 297
punctuation, 197
push notifications, 406
pushState() method, 375
puzzle example application, building with drag and drop, 284-287
pywebsocket, 325

Q-R

quotation marks
- attributes and, 114
- in typography, 196-197

radial gradients, 169-171
radians, degrees versus, 175
rag (in typography), 195-196
range type, 230
RDFa (Resource Description Framework in attributes), 269-271
reading
- cookies with JavaScript, 71-72
- file information in File API, 334-335

real-time connections. *See* WebSockets API
recipes, hRecipe microformat, 262-265
rectangles, drawing, 166-171
redirection to mobile home page, 100
rel attribute
- link types, 299-300
  - alternate link type, 300-301
  - author link type, 301
  - prev and next link types, 304
  - sidebar link type, 305
  - stylesheet link type, 305
- overview, 269, 297

rel-nofollow microformat, 265
relationships. *See* link relationships
relative font sizes, 192
reloads, session storage and, 358
removeItem() method, 360
Rentacomputer.com, 133
replaceState() method, 375
request-response functionality, 325
required attribute, 114, 238
resetting styles, 40
resolution, checking, 63
resolution keyword, 63
resolution of screen, 8
resource attribute, 270
Resource Description Framework in attributes (RDFa), 269-270
resources for information, 447-448
results attribute, 234
Retina display, 63, 214
rev attribute, 270, 299
reviews, hReview microformat, 262-265
rich text editors, creating, 246-247
risks, planning web applications, 87
rollovers
- creating, 69-71
- image rollovers, creating with jQuery, 75-76

root elements (sections), 149-150
rotating background colors, 314-315
rounded corners, creating, 42-44
<rp> tag, 23

<rt> tag, 23

ruby annotations, support for, 113

<ruby> tag, 23

S

<s> tag, 26, 155

<samp> tag, 155

sans serif fonts, serif fonts versus, 92

saving gift status in online store example, 359-360

Scalable Vector Graphics (SVG), 115

scaling images, 179-181

screen orientation, checking, 62

screen readers, emphasized text, 156

screen size, 8

<script> tag, 26, 68-69

Scripting API, 310-313

scripts. *See also* JavaScript
- attaching events to, 165
- attaching to events, 312
- background color rotation example, 314-315
- calling at page load, 165
- loading, 21
- running in background with Web Workers API, 326-332

scrolling pages, bookmarking, 376

search box types, 233-235

search link type, 299

searches
- displaying results, 234
- field styles, 234-235

sectioning elements
- <article> tag, 144
- <aside> tag, 144-145
- building blog with, 152-154
- heading, header, footer elements, 150-152
- <h1> tag, 146
- list of, 141-142
- <nav> tag, 146-147
- outlines, creating, 142, 144
- page locations of, 145
- root elements, 149-150
- <section> tag, 147-148

<section> tag, 16, 147-148

sectioning tags
- converting to, 16-18
- list of, 15-16

secure flag, 357

secure pages, caching, 345

security
- phishing schemes with History API, 383-384
- web storage, 361
- WebSockets API, 324

Selectivizr, 137

selectors (CSS), 34

selling applications, 418-419

semantic elements, 154
- changed to semantic in HTML5, 25, 155-156
- explained, 16
- function of, 19
- in HTML 4, 155
- in HTML5, 156-159
- layout tags
  - converting to, 16-18
  - list of, 15-16
- list of, 18-19
- support for, 113

semantic markup
- comparing formats for, 270-271
- microdata
  - converting microformats to, 268-269
  - explained, 267-268
  - microformats versus, 267
- microformats
  - adding to websites, 266-267
  - benefits of, 263-264
  - examples of current, 264-265
  - explained, 261-263
  - mobile devices and, 271-272
  - principles for using, 264
- RDFa (Resource Description Framework in attributes), 269-270

Sencha Touch, 132

serif fonts, sans serif fonts versus, 92

server cache, 351

server setup, WebSockets API, 325

session storage, local storage versus, 357-358

setInterval() method, 313

setItem() method, 360

setTimeout() method, 313

setting cookies with JavaScript, 71-72

SGML (Standard Generalized Markup Language), 1

shadowBlur property, 178

shadowColor property, 178

shadowOffsetX property, 178

shadowOffsetY property, 178

shadows
- adding to text, 178-179
- creating, 45

shapes
- clearing, 166
- drawing, 165-166
  - circles, 175-177
  - polygons and lines, 171-175
  - rectangles and squares, 166-171
- in mobile designs, 127

sidebars
- <aside> tag, 144-145
- sidebar link type, 299, 305

simple prompts, 315

site statistics, viewing, 110

SiteSpinner Pro, 120

size
- of Android icons, 416
- of code for mobile device websites, 9-10

<small> tag, 26, 114, 156

SMART goals, 87

smart quotes, 196

Socket.IO, 326

soft hyphens, 193-194

<source> tag, 20-21, 210
- attributes, 214-215
- multiple sources, 210

spacing (in typography), 191-193

<span> tags, 192, 262

specifications in Open Web Standard, 4-5

speech attribute, 115

spellcheck attribute, 25, 251-252

split testing, 109

spotlight search icon, 414

SQL databases
- Indexed Database API, 363-365
- Web SQL Database API, 361-370

SQLite, data types and, 363

square <div> tags, converting to circles, 128

squares, drawing, 166-171

src attribute, 213-215, 269

srclang attribute, 215

Standard Generalized Markup Language (SGML), 1

state objects, 379-380

static maps, 400

statistics (site), viewing, 110

stop colors for gradients, 168

storage event, 361

storage object, defining, 358

StorageEvent object, 361

storing data. *See* web storage

straight quotes, 196

<strike> tag, 27

strings, 357

stroke() method, 171

strokeRect function, 167

strokeStyle property, 168

<strong> tag, 26, 155

style sheets
- creating, 34-36
- embedded, 34
- external, 35-36
- fonts, changing, 36-37
- inline, 34
- resetting styles, 40
- rounded corners, 42-44
- shadows, 45
- transparent colors, 45
- web page layout, 38-41

<style> tag, 26

styles
- for mobile devices, 61-62
- for <canvas> tag, 168-171
- of fonts, 190-191
- search field styles, 234-235

stylesheet link type, 299, 305

styling layout, 94-96

submitting applications to Apple App store, 418

summary attribute, 27

<summary> tag, 18, 149

SVG (Scalable Vector Graphics), 115, 182-183

syntax changes in HTML5, 27-28

System information API, 310, 317-318

T

tabindex attribute, 255

tablets. *See* mobile devices

tag link type, 299

tags
- block-level elements, linking, 295-296
- changes from HTML 4, 25-27
- checking value retention of, 56-57
- form tags, list of, 21-22
- hiding, 252
- international support tags, list of, 22-23
- layout tags
  - converting to, 16-18
  - list of, 15-16
- list of HTML5 elements, 437
- media elements, 210-212
  - audio and video attributes, 213
  - list of, 19-21

source attributes, 214-215
track attributes, 215
video attributes, 213-214
nesting, 3
sectioning elements
<article> tag, 144
<aside> tag, 144-145
building blog with, 152-154
<h1> tag, 146
heading, header, footer elements, 150-152
list of, 141-142
<nav> tag, 146-147
outlines, creating, 142-144
page locations of, 145
root elements, 149-150
<section> tag, 147-148
semantic elements, 154
changed to semantic in HTML5, 155-156
function of, 19
in HTML 4, 155
in HTML5, 156-159
list of, 18-19
target attribute, 294-295
targeted advertising, 389
<td> tag, 149
tel type, 230
testing
applications, 416
mobile applications, 63-64
web applications for mobile devices, 121-123
in Internet Explorer, 7
mobile device websites, 11
on multiple devices, 133-134
outlines, 143
split testing, 109
text
blocks of text, color of, 125
drawing on canvas, 177-179
emphasized text, 156
placeholder text, 224-226
text editors, finding, 5
text-align property, 193
text-shadow property, 45
textAlign attribute, 177
textBaseline attribute, 177
texture in mobile designs, 129
threads, running in background, 326-332
time and date types, 231-232
<time> tag, 19, 114, 158-159
timeline, planning web applications, 88
TIMEOUT (3) code, 390
timeout property, 392
Timers API, 310, 313
timestamp property, 390
TinyMCE, 256
title attribute, 157
titles in state objects, 380
TopStyle, 120
touchmove event, 116
touchstart event, 116
<trace> tag, 210
<track> tag, 20-21, 210, 215
tracking, 191-193
trademark symbols, 197
transaction() method, 362
transparent colors, creating, 45
triangles, drawing, 172
triangulation, 388, 392
troubleshooting
application cache debugging, 351-352
video
in Android, 211
checking MIME types, 207
in iOS, 210
Web Workers API, 327
<tt> tag, 27
turning off location services, 396
tv devices, checking, 63
type, changing, 91-93
type attribute, 26, 211, 214-215
typefaces
CSS font properties, 188-190
fonts versus, 188
setting font styles, 190-191
typography
business marks, 197
defined, 187
hyphenation, 193-195
punctuation, 197
quotation marks, 196-197
rag, 195-196
spacing, 191-193
typefaces
CSS font properties, 188-190
fonts versus, 188
setting font styles, 190-191
widows and orphans, 196

U

<u> tag, 27, 156
<ul> tag, 27
UNKNOWN ERROR (0) code, 390

updateready event, 347
upgrading
- deciding when to upgrade, 105-106
  - browser support for standards, 106-108
  - browser usage, 109-110
  - growth in mobile web browsing, 110-111
  - iterative design, 108-109
- to HTML5
  - adding features as extras, 113-115
  - browser support for, 112-113
  - features to start with, 111-112
  - mobile device-specific features, 115-116

uploading files with File API, 331-333
- browser support, 333
- building input field, 333
- drag-and-drop file uploader example, 335-337
- getting file information, 334
- reading file information, 334-335

URI scheme interface, 332
URLs
- animated URLs with History API, 384
- changing with History API, 379-380
- URL types, 233-235

user interface
- browser support for HTML5 features, 255-257
- disabling, 247
- HTML5 components, 253-255

user prompts, 316-317
User prompts API, 310, 315-316
user-scalable property, 98
user-select attribute, 280
userData (local storage plug-in), 356
@username nanoformat, 272
uTest, 133

V

valid documents, well-formed documents versus, 2
validating
- forms, 237-239
  - with JavaScript, 71
  - with jQuery, 76-78
- importance of, 238
- input types, 230

validators, 121
validity of code for mobile device websites, 9-10
value attribute, 157
values
- of elements, 56-57
- of methods, 54-56

<var> tag, 155
video
- adding with Flash fallback, 216-217
- custom video controls, creating, 217-219
- embedding, 21
- troubleshooting
  - in Android, 211
  - in iOS, 210
- video codecs
  - browser support, 206-208
  - checking for support, 55-56
  - WebM versus H.264, 54
- video streams, adding, 210

<video> tag, 20-21, 116, 210
- advantages and disadvantages of, 205-206
- attributes, 213-214
- fallback options, 216-217
- specifying video types, 214-215
- support for, 113

viewing site statistics, 110
viewport meta tag, 96-99
VirtualBox, 64
visibility property, display property versus, 70
visual design, changing for mobile devices, 124-130
voice recognition, 115
Vorbis audio codec, 209

W

W3C (World Wide Web Consortium), 2
W3C mobileOK Checker, 121
W3C validator, 9
W3C website, 448
W3Counter.com, 109
Warpspire Experiments, 376
watchPosition() method, 389-393
WAV audio codec, 209
<wbr> tag, 19
web applications, 309
- APIs, 310
- background color rotation example, 314-315
- click events in iOS, 313

converting to mobile applications
changing the visual design, 124-130
checking for HTML5 and CSS3 support, 130-132
compatibility with older browsers, 134-138
evaluating the content, 123-124
multiple device support, 132-133
testing on multiple devices, 133-134
testing the application, 121-123
desktop applications versus, 4
File API, 331-333
browser support, 333
building input field, 333
drag-and-drop file uploader example, 335-337
getting file information, 334
reading file information, 334-335
jQuery, 78-79
offline web applications, 341-342
browser support, 342
cache manifest, 343-347
debugging application cache, 351-352
DOM events and properties, 347-348
events, 342
listening for cache changes, 348-350
web storage and, 342
popularity of, 30
Scripting API, 310-313
System Information API, 317-318
Timers API, 313
user prompts, 316-317
User Prompts API, 315-316
Web Workers API, 326-332
WebSockets API, 323
benefits of, 324-325
browser support, 323
establishing, 325-326
security, 324
web browsers. *See* browsers
Web Design and HTML at About.com website, 448
Web Developer extension, 64
web editors, choosing, 120-121
Web Hypertext Application Technology Working Group (WHATWG), 3
Web Open Font Format. *See* WOFF (Web Open Font Format)
web pages
CSS for layout, 38-41
editing content
browser support, 255-257
contenteditable attribute, 245-247
execCommand() method commands, 247-251
spellcheck attribute, 251-252
JavaScript, adding, 68-69
writing, 5-6
web safe colors, 198
web safe fonts, 198
web server cache, 351
web sites, deciding to upgrade, 105-106
browser support for standards, 106-108
browser usage, 109-110
growth in mobile web browsing, 110-111
iterative design, 108-109
Web SQL Database API, 361-370
web storage
Indexed Database API, 363-365
local storage
checking browser support, 358
cookies versus, 356-357
defining storage object, 358
explained, 356
interface methods, 360-361
plug-ins for, 356
saving gift status in online store example, 359-360
session storage versus, 357-358
offline web applications and, 342
prior to HTML5, 355
Web SQL Database API, 361-370
Web Workers API, 326-332
WebKit browsers, 234-235
WebKit extensions for border-radius property, 43
webkit_draggable() method, 287
webkit_drop.add() method, 288
-webkit-min-device-pixel-ratio keyword, 63
-webkit-user-drag attribute, 280
WebM video codec, 54, 208
WebOS, emulator for, 134
websites
adding HTML5 features as extras, 113-115
adding microformats to, 266-267

for HTML5 information, 448
writing for mobile devices, 7
content included, 8-9
on separate domain, 10
screen size/resolution, 8
testing, 11
validity and size of code, 9-10
what to avoid, 9-10
WebSockets API, 323
benefits of, 324-325
browser support, 323
establishing, 325-326
security, 324
well-formed documents, valid documents versus, 2
WHATWG (Web Hypertext Application Technology Working Group), 3
WHATWG Wiki, 215
whitelists (for cache manifest), 345
widows, 196
widows property, 196
width, checking, 61-62
width keyword, 61
width property, 27, 98, 193, 214
window.applicationCache.swapCache() method, 351
window.close() function, 71
window.navigator.appName method, 317
window.navigator.appVersion method, 317
window.navigator.platform method, 317
window.navigator.userAgent method, 317
window.open() function, 71
windows, opening/closing, 71
Windows Phone, emulator for, 134
WOFF (Web Open Font Format)
advantages and disadvantages of, 198-199
browser support for, 201-202
@font-face feature (CSS), 199-201
word-spacing property, 193
World Wide Web Consortium (W3C), 2
writing
cache manifest, 344-346
web pages, 5-6
websites for mobile devices, 7
content included, 8-9
on separate domain, 10
screen size/resolution, 8
testing, 11
validity and size of code, 9-10
what to avoid, 9-10
WYSIWYG editors, creating, 246-247

X-Y-Z

Xcode, 409
XFN (XHTML Friends Network), 265
XHTML, 2
XHTML Friends Network (XFN), 265
XML (eXtensible Markup Language), 2
XSS (cross site scripting), 361
YUI Rich Text Editor, 256