

4TH EDITION

Free Sampler

Learning Web Design

A BEGINNER'S GUIDE TO HTML, CSS, JAVASCRIPT, AND WEB GRAPHICS

Jennifer Niederst Robbins

Whether you're a beginner or bringing your skills up to date, this book gives you a solid footing in modern web production. I teach each topic visually at a pleasant pace, with frequent exercises to let you try out new skills. Reading it feels like sitting in my classroom! —Jennifer Robbins

O'REILLY

O'Reilly Ebooks—Your bookshelf on your devices!



When you buy an ebook through oreilly.com you get lifetime access to the book, and whenever possible we provide it to you in five, DRM-free file formats—PDF, .epub, Kindle-compatible .mobi, Android .apk, and DAISY—that you can use on the devices of your choice. Our ebook files are fully searchable, and you can cut-and-paste and print them. We also alert you when we've updated the files with corrections and additions.

Learn more at ebooks.oreilly.com

You can also purchase O'Reilly ebooks through the
iBookstore, the [Android Marketplace](#), and [Amazon.com](#).

O'REILLY®

Spreading the knowledge of innovators

oreilly.com

Learning Web Design

Fourth Edition

A Beginner's Guide to HTML, CSS, JavaScript, and Web Graphics

Jennifer Niederst Robbins

O'REILLY®

Beijing • Cambridge • Farnham • Köln • Sebastopol • Tokyo

Learning Web Design, Fourth Edition

by Jennifer Niederst Robbins

Copyright © 2012 Littlechair, Inc. All rights reserved.
Printed in Canada.

Published by O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.

O'Reilly Media books may be purchased for educational, business, or sales promotional use. Online editions are also available for most titles (safari.oreilly.com). For more information, contact our corporate/institutional sales department: 800-998-9938 or corporate@oreilly.com.

Editor: Simon St. Laurent

Production Editor: Melanie Yarbrough

Copy Editor: Genevieve d'Entremont

Technical Reviewer: Aaron Gustafson, Matt Menzer, Joel Marsh

Interior Designer: Ron Bilodeau

Cover Designer: Mark Paglietti

Indexer: Ellen Troutman Zaig

Print History:

February 2001: First edition.

March 2004: Second edition.

June 2007: Third edition.

August 2012: Fourth edition.

The O'Reilly logo is a registered trademark of O'Reilly Media, Inc. This book's trade dress is a trademark of O'Reilly Media, Inc. Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and O'Reilly Media, Inc. was aware of a trademark claim, the designations have been printed in caps or initial caps.

While every precaution has been taken in the preparation of this book, the publisher and author assume no responsibility for errors or omissions, or for damages resulting from the use of the information contained herein.

ISBN: 978-1-449-31927-4
[TI]

CONTENTS

Preface	xi
---------------	----

Part I Getting Started

Chapter 1

Where Do I Start?	3
Where Do I Start?	4
What Does a Web Designer Do?	4
What Languages Do I Need to Learn?	11
What Do I Need to Buy?	14
What You've Learned	19
Test Yourself.....	20

Chapter 2

How the Web Works.....	21
The Internet Versus the Web	21
Serving Up Your Information	21
A Word About Browsers	23
Web Page Addresses (URLs)	24
The Anatomy of a Web Page	26
Putting It All Together.....	30
Test Yourself.....	32

Chapter 3

Some Big Concepts You Need to Know	33
A Dizzying Multitude of Devices	34
Sticking with the Standards	36
Progressive Enhancement	36
Responsive Web Design	38

One Web for All (Accessibility)	41
The Need for Speed (Site Performance)	43
Test Yourself	45

Part II HTML Markup for Structure

Chapter 4

Creating a Simple Page	49
A Web Page, Step by Step	49
Before We Begin, Launch a Text Editor	50
Step 1: Start with Content	53
Step 2: Give the Document Structure	55
Step 3: Identify Text Elements	58
Step 4: Add an Image	61
Step 5: Change the Look with a Style Sheet	64
When Good Pages Go Bad	65
Validating Your Documents	66
Test Yourself	67
Element Review: Document Structure	68

Chapter 5

Marking Up Text	69
Paragraphs	70
Headings	70
Lists	73
More Content Elements	76
Organizing Page Content	79
The Inline Element Roundup	84
Generic Elements (div and span)	95
Some Special Characters	99
Putting It All Together	100
Test Yourself	102
Element Review: Text	104

Chapter 6

Adding Links	105
The href Attribute	106
Linking to Pages on the Web	107
Linking Within Your Own Site	108
Targeting a New Browser Window	118

Mail Links	119
Telephone Links.....	120
Test Yourself.....	121
Element Review: Links	122

Chapter 7

Adding Images.....	123
First, a Word on Image Formats.....	123
The img Element	124
A Window in a Window	130
Test Yourself.....	131
Element Review: Images	132

Chapter 8

Table Markup.....	133
How Tables Are Used	133
Minimal Table Structure	135
Spanning Cells.....	139
Table Accessibility.....	142
Wrapping Up Tables	144
Test Yourself.....	146
Element Review: Tables	146

Chapter 9

Forms.....	147
How Forms Work	147
The form Element	149
Variables and Content	151
The Great Form Control Roundup	152
Form Accessibility Features	171
Form Layout and Design.....	173
Test Yourself.....	175
Element Review: Forms	176

Chapter 10

What's Up, HTML5?	181
A Funny Thing Happened on the Way to XHTML 2	182
In the Markup Department	185
Meet the APIs	189
Video and Audio	192

Canvas	198
Final Word	202
Test Yourself	203

Part III CSS for Presentation

Chapter 11

Cascading Style Sheets Orientation	207
The Benefits of CSS	207
How Style Sheets Work	209
The Big Concepts	214
Moving Forward with CSS	221
Test Yourself	223

Chapter 12

Formatting Text	225
The Font Properties	225
Changing Text Color	243
A Few More Selector Types	244
Text Line Adjustments	249
Underlines and Other "Decorations"	252
Changing Capitalization	252
Spaced Out	253
Text Shadow	254
Changing List Bullets and Numbers	259
Test Yourself	261
CSS Review: Font and Text Properties	263

Chapter 13

Colors and Backgrounds	265
Specifying Color Values	265
Foreground Color	272
Background Color	273
Playing with Opacity	275
Introducing...Pseudo-class Selectors	276
Pseudo-element Selectors	279
Attribute Selectors	281
Background Images	284
The Shorthand background Property	293
Like a Rainbow (Gradients)	296
Finally, External Style Sheets	300

Test Yourself.....	303
CSS Review: Color and Background Properties	304

Chapter 14

Thinking Inside the Box

The Element Box	305
Specifying Box Dimensions	306
Padding	312
Borders	316
Margins	328
Assigning Display Roles	333
Adding Drop Shadows to Boxes	335
Test Yourself.....	336
CSS Review: Basic Box Properties	338

Chapter 15

Floating and Positioning.....

Normal Flow	341
Floating	342
Positioning Basics	356
Relative Positioning	358
Absolute Positioning	359
Fixed Positioning	368
Test Yourself.....	370
CSS Review: Floating and Positioning Properties	371

Chapter 16

Page Layout with CSS

Page Layout Strategies	373
Page Layout Techniques	380
Multicolumn Layouts Using Floats	381
Positioned Layout	392
Top-to-Bottom Column Backgrounds	395
Test Yourself.....	398

Chapter 17

Transitions, Transforms, and Animation

Ease-y Does It (CSS Transitions)	399
CSS Transforms	410
Keyframe Animation	420

Test Yourself.....	423
CSS Review: Transitions, Transforms, and Animation	426

Chapter 18 CSS Techniques 427

A Clean Slate (CSS Reset)	427
Image Replacement Techniques	429
CSS Sprites	430
Styling Forms	434
Styling Tables	441
Basic Responsive Web Design	444
Wrapping Up Style Sheets	454
Test Yourself	454
CSS Review: Table Properties	456

Part IV JavaScript for Behaviors

Chapter 19 Introduction to JavaScript..... 459

What Is JavaScript?	459
Adding JavaScript to a Page	463
The Anatomy of a Script	463
The Browser Object	478
Events	478
Putting It All Together	481
Test Yourself	483

Chapter 20 Using JavaScript 485

Meet the DOM	485
Polyfills	493
JavaScript Libraries	497
Big Finish	501
Test Yourself	502

Part V Creating Web Graphics

Chapter 21

Web Graphics Basics	507
Image Sources	507
Meet the Formats	510
Image Size and Resolution	522
Working with Transparency	526
Introduction to SVG	533
Summing Up Images	538
Test Yourself	539

Chapter 22

Lean and Mean Web Graphics	541
General Image Optimization Strategies	542
Optimizing GIFs	543
Optimizing JPEGs	547
Optimizing PNGs	552
Optimize to File Size	553
Optimization in Review	554
Test Yourself	555

Appendix A

Answers	557
----------------------	------------

Appendix B

CSS3 Selectors	583
-----------------------------	------------

Index.....

.....	587
-------	-----

WHERE DO I START?

The Web has been around for more than 20 years now, experiencing euphoric early expansion, an economic-driven bust, an innovation-driven rebirth, and constant evolution along the way. One thing is certain: the Web as a communication and commercial medium is here to stay. Not only that, it has found its way onto devices such as smartphones, tablets, TVs, and more. There have never been more opportunities to put web design know-how to use.

Through my experience teaching web design courses and workshops, I've had the opportunity to meet people of all backgrounds who are interested in learning how to build web pages. Allow me to introduce you to just a few:

"I've been a print designer for 17 years, and now I am feeling pressure to provide web design services."

"I work as a secretary in a small office. My boss has asked me to put together a small internal website to share company information among employees."

"I've been a programmer for years, but I want to try my hand at design. I feel like the Web is a good opportunity to explore new skills."

"I am an artist and I want to know how to get samples of my paintings and sculpture online."

"I tinkered with web pages in high school and I think it might be something I'd like to do for a living."

Whatever the motivation, the first question is always the same: "Where do I start?" It may seem like there is a mountain of stuff to learn, and it's not easy to know where to jump in. But you have to start somewhere.

This chapter attempts to put the learning curve in perspective by answering the most common questions I get asked by people ready to make the leap. It provides an introduction to the disciplines, technologies, and tools associated with web design.

IN THIS CHAPTER

Where do I start?

What does a web designer do?

What languages do I need to learn?

What software and equipment do I need to buy?

Where Do I Start?

Your particular starting point will no doubt depend on your background and goals. However, a good first step for everyone is to get a basic understanding of how the Web and web pages work. This book will give you that foundation. Once you learn the fundamentals, there are plenty of resources on the Web and in bookstores for you to further your learning in specific areas.

There are many levels of involvement in web design, from building a small site for yourself to making it a full-blown career. You may enjoy being a full-service website developer or just specializing in one skill. There are a lot of ways you can go.

If your involvement in web design is purely at the hobbyist level, or if you have just one or two web projects you'd like to publish, you may find that a combination of personal research (like reading this book), taking advantage of available templates, and perhaps even investing in a visual web design tool such as Adobe Dreamweaver may be all you need to accomplish the task at hand. Many Continuing Education programs offer introductory courses to web design and production.

If you are interested in pursuing web design or production as a career, you'll need to bring your skills up to a professional level. Employers may not require a web design degree, but they will expect to see working sample sites that demonstrate your skills and experience. These sites can be the result of class assignments, personal projects, or a simple site for a small business or organization. What's important is that they look professional and have well-written, clean HTML, style sheets, and possibly scripts behind the scenes. Getting an entry-level job and working as part of a team is a great way to learn how larger sites are constructed and can help you decide which aspects of web design you would like to pursue.

I Just Want a Blog!

You don't necessarily need to become a web designer to start publishing your words and pictures on the Web. You can start your own "blog" or personal journal site using one of the free or inexpensive blog hosting services. These services provide templates that generally spare you the need to learn HTML (although it still doesn't hurt). These are some of the most popular as of this writing:

- WordPress (www.wordpress.com)
- Blogger (www.blogger.com)
- Tumblr (www.tumblr.com)

Another drag-n-drop site design and hosting service that goes beyond the blog is Squarespace (www.squarespace.com).

AT A GLANCE

The term "web design" has come to encompass a number of disciplines, including:

- Visual (graphic) design
- User interface and experience design
- Web document and style sheet production
- Scripting and programming
- Content strategy
- Multimedia

What Does a Web Designer Do?

Over the years, the term "web design" has become a catchall for a process that encompasses a number of different disciplines, from user experience design, to document markup, to serious programming. This section describes some of the most common roles.

If you are designing a small website on your own, you will need to wear many hats. The good news is that you probably won't notice. Consider that the day-to-day upkeep of your household requires you to be part-time chef, housecleaner, accountant, diplomat, gardener, and construction worker—but to you it's just the stuff you do around the house. In the same way, as a solo web designer, you may be a part-time graphic designer, writer, HTML author, and information architect, but to you, it'll just feel like "making web pages." Nothing to worry about.

There are also specialists out there whom you can hire to fill in the skills you don't have. For example, I have been creating websites since 1993 and I still hire programmers and multimedia developers when my clients require interactive features. That allows me to focus on the parts I do well (in my case, it's the content organization, interface, and visual design).

Large-scale websites are almost always created by a team of people, numbering from a handful to hundreds. In this scenario, each member of the team focuses on one facet of the site-building process. If that is the case, you may be able to simply adapt your current set of skills (writing, Photoshop, programming, etc.) and interests to the new medium.

I've divided the myriad roles and responsibilities typically covered under the umbrella term "web design" into four very broad categories: design, development, content strategy, and multimedia.

If you are not interested in becoming a jack-of-all-trades solo web designer, you may choose to specialize and work as part of a team or as a freelance contractor.

Design

Ah, design! It sounds fairly straightforward, but even this simple requirement has been divided into a number of specializations when it comes to creating sites. Here are a few of the new job descriptions related to designing a site, but bear in mind that the disciplines often overlap and that the person calling herself the "Designer" often is responsible for more than one (if not all) of these responsibilities.

User Experience, Interaction, and User Interface design

Often, when we think of design, we think about how something looks. On the Web, the first matter of business is designing how the site *works*. Before picking colors and fonts, it is important to identify the site's goals, how it will be used, and how visitors move through it. These tasks fall under the disciplines of [Interaction Design \(IxD\)](#), [User Interface \(UI\) design](#), and [User Experience \(UX\) design](#). There is a lot of overlap between these responsibilities, and it is not uncommon for one person or team to handle all three.

The goal of the [Interaction Designer](#) is to make the site as easy, efficient, and delightful to use as possible. Closely related to interaction design is [User Interface](#) design, which tends to be more narrowly focused on the functional organization of the page as well as the specific tools (buttons, links, menus, and so on) that users use to navigate content or accomplish tasks.

A more recent job title in the web design realm is the [User Experience Designer](#). The UX designer takes a more holistic view—ensuring the entire experience with the site is favorable. UX design is based on a solid understanding of users and their needs based on observations and interviews. According to Donald Norman (who coined the term), user experience design includes "all aspects of the user's interaction with the product: how it is perceived, learned, and used." For a website or application, that includes

the visual design, the user interface, the quality and message of the content, and even overall site performance. The experience must be in line with the organization's brand and business goals in order to be successful.

Some of the documents an IxD, UI, or UX designer might produce include:

User research and testing reports

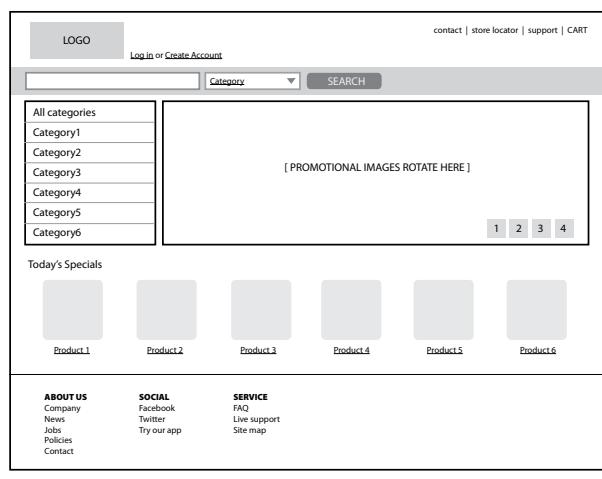
Understanding the needs, desires, and limitations of users is central to the success of the design of the site or web application. This approach of designing around the user's needs is referred to as [User Centered Design \(UCD\)](#), and it is central to contemporary design. Site designs often start with user research, including interviews and observations, in order to gain a better understanding of how the site can solve problems or how it will be used. It is typical for designers to do a round of user testing at each phase of the design process to ensure the usability of their designs. If users are having a hard time figuring out where to find content or how to move to the next step in a process, then it's back to the drawing board.

Wireframe diagrams

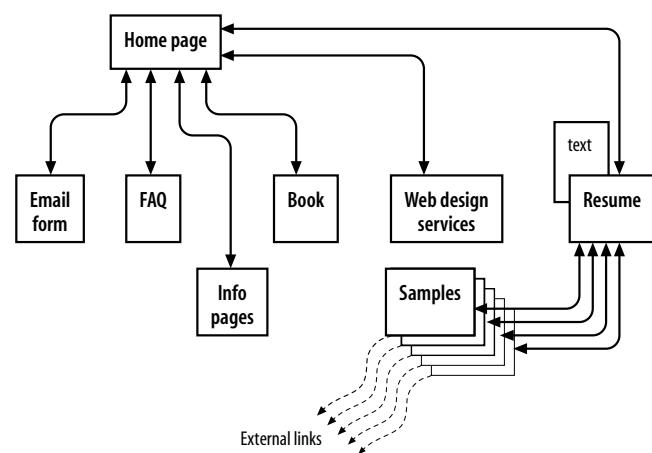
A wireframe diagram shows the structure of a web page using only outlines for each content type and widget ([Figure 1-1](#)). The purpose of a wireframe diagram is to indicate how the screen real estate is divided and indicate where functionality and content such as navigation, search boxes, form elements, and so on, are placed, without any decoration or graphic design. They are usually annotated with instructions for how things should work so the development team knows what to build.

Site diagram

A site diagram indicates the structure of the site as a whole and how individual pages relate to one another. [Figure 1-2](#) shows a very simple site diagram. Some site diagrams fill entire walls!



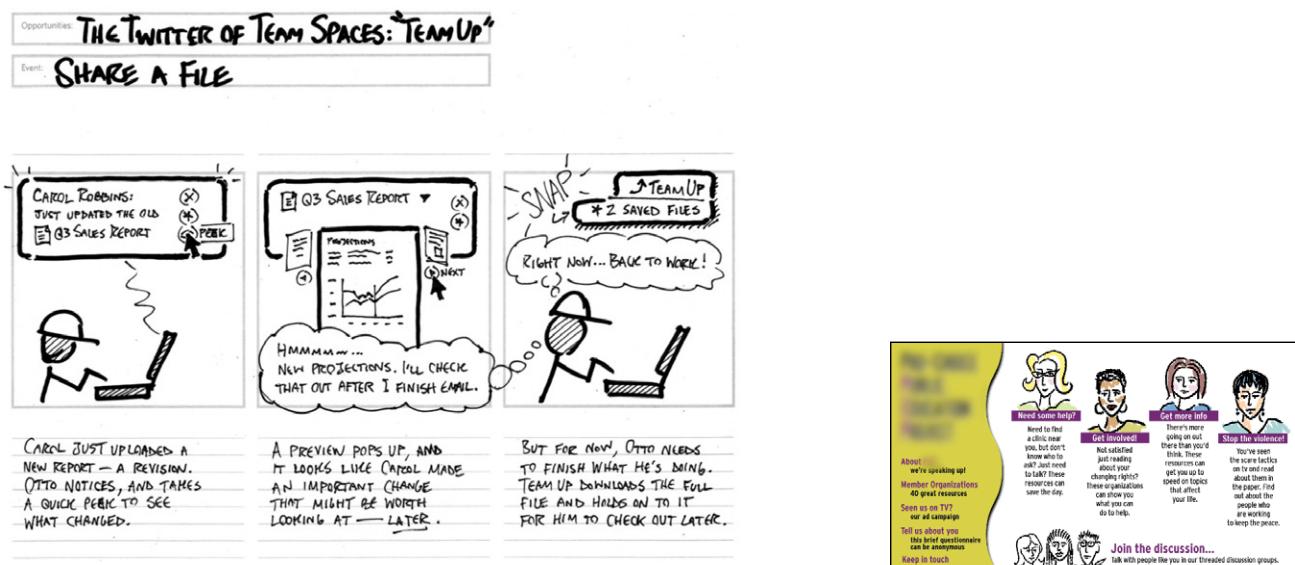
[Figure 1-1.](#) Wireframe diagram.



[Figure 1-2.](#) A simple site diagram.

Storyboards and user flow charts

A storyboard traces the path through a site or application from the point of view of a typical user (a [persona](#) in UX lingo). It usually includes a script and “scenes” consisting of screen views or the user interacting with the screen. The storyboard aims to demonstrate the steps it takes to accomplish tasks, possible options, and also introduces some standard page types. [Figure 1-3](#) shows a simple storyboard. A user flow chart is another method for showing how the parts of a site or application are connected that tends to focus on technical details rather than telling a story. For example, when the user does this, it triggers that function on the server. It is common for designers to create a user flow chart for the steps in a process such as member registration or online payments.



[Figure 1-3](#). A typical storyboard [courtesy of Adaptive Path; drawn by Brandon Schauer].

Visual (graphic) design

Because the Web is a visual medium, web pages require attention to presentation and design. A graphic designer creates the “look and feel” of the site—logos, graphics, type, colors, layout, etc.—to ensure that the site makes a good first impression and is consistent with the brand and message of the organization it represents. Visual designers typically generate sketches of the way the site might look, as shown in [Figure 1-4](#). They may also be responsible for producing the graphic files in a way that is optimized for delivery over the Web (see [Chapter 21, Lean and Mean Web Graphics](#) for image optimization techniques).

If you are interested in doing the visual design of commercial sites professionally, I strongly recommend graphic design training as well as a strong proficiency in Adobe Photoshop (the industry standard) or Adobe Fireworks.



[Figure 1-4](#). Look and feel sketches for a simple site.

Style Tiles

Another approach to capturing the look and feel of a site is to create style tiles, which give examples of color schemes, branding elements, content and UI treatments, and mood boards without applying them to a specific page layout. The idea is to agree upon a consistent visual language for the site. For more on this technique, read the article “Style Tiles and How They Work,” by Samantha Warren (www.alistapart.com/articles/style-tiles-and-how-they-work), and visit her excellent site where you can download a template at styletiles.es.

If you are already a graphic designer, you will be able to adapt your skills to the Web easily, although this will not excuse you from acquiring a solid understanding of HTML, CSS, and other web technologies. Because most sites have at least a few images, even hobbyist web designers will need to know how to create and edit images, at minimum.

Again, I want to note that all of these responsibilities may fall into the hands of one designer who creates both the look and the functionality of a site. But for larger sites with bigger budgets, there is an opportunity to find your own special niche in the design process.

Development

A fair amount of the web design process involves the creation and troubleshooting of the documents, style sheets, scripts, and images that make up a site. At web design firms, the team that handles the creation of the files that make up the website (or templates for pages that get assembled dynamically) is usually called the **development** or **production** department.

Web developers may not design the look or structure of the site themselves, but they do need to communicate well with designers and understand the intended site goals so they may suggest solutions that meet those goals.

The broad disciplines that fall under development are authoring, styling, and scripting/programming.

Authoring/markup

Authoring is the term used for the process of preparing content for delivery on the Web, or more specifically, marking up the content with HTML tags that describe its content and function. If you want a job as a web developer, you need to have an *intricate* knowledge of HTML and how it functions on various browsers and devices. The HTML specification is constantly evolving, which means you’ll need to keep up with the latest best practices and opportunities as well as bugs and limitations. The good news is, it’s not difficult to get started, and from there, you can gradually increase your skills. We’ll be dabbling with HTML in [Chapter 2, How the Web Works](#) and then discussing it in great detail in Part II of this book.

NOTE

Many visual designers translate their designs into HTML and CSS documents themselves. In fact, there is a popular argument that in order to call yourself a “web designer,” you must be able to build your designs yourself, and nearly everyone agrees that your job prospects will be better if you are able to code as well as design.

Styling

In web design, the appearance of the page in the browser is controlled by style rules written in CSS (Cascading Style Sheets). We’ll get deep into CSS in Part III of this book (including what “cascading” means!), but for now just know that in contemporary web design, the appearance of the page is handled separately from the HTML markup of the page. Again, if you are interested in working in web development, knowing your way around CSS and how it is supported (or not supported) by browsers is guaranteed to be part of your job description.

Scripting and programming

As the Web has evolved into a platform of applications for getting stuff done, programming has never been more important. JavaScript is the language that makes elements on web pages do things. It adds behaviors and functionality to elements in the page and even to the browser window itself.

There are other web-related programming languages as well, including PHP, Ruby, Python, and ASP.NET, that run on the server and process data and information before it is sent to the user's browser. See the sidebar “[Frontend Versus Backend](#)” for more information on what happens where.

Web scripting and programming definitely requires some traditional computer programming prowess. While many web programmers have degrees in computer science, it is also common for developers to be self-taught. A few developers I know started by copying and adapting existing scripts, then gradually added to their programming skills with each new project. Still, if you have no experience with programming languages, the initial learning curve may be a bit steep.

Teaching web programming is beyond the scope of this book. JavaScript is introduced in [Chapter 19, Introduction to JavaScript](#) (teaching JavaScript could fill a whole book itself). It is possible to turn out content-rich, well-designed sites without the need for programming, so hobbyist web designers should not be discouraged. However, once you get into collecting information via forms or serving information on demand, it is usually necessary to have a programmer on the team. You may also ask your hosting company if they offer the functionality you are looking for in an easy-to-use, canned service.

Frontend Versus Backend

You may hear web designers and developers say that they specialize in either the [frontend](#) or [backend](#) of website creation.

Frontend design

“Frontend” refers to any aspect of the design process that appears in or relates directly to the browser. This book focuses primarily on frontend web design.

The following tasks are commonly considered to be frontend tasks:

- Graphic design and image production
- Interface design
- Information design as it pertains to the user's experience of the site
- HTML document and style sheet development
- JavaScript

Backend development

“Backend” refers to the programs and scripts that work on the server behind the scenes to make web pages dynamic and interactive. In general, backend web development falls in the hands of experienced programmers, but it is good for all web designers to be familiar with backend functionality.

The following tasks take place on the backend:

- Information design as it pertains to how the information is organized on the server
- Forms processing
- Database programming
- Content management systems
- Other server-side web applications using PHP, JSP, Ruby, ASP.NET, Java, and other programming languages

Content strategy and creation

Third on our list, though ideally first in the actual website creation process, is the critical matter of the site's content itself. Anyone who uses the title "web designer" needs to be aware that everything we do supports the process of getting the content, message, or functionality to our users. Furthermore, good writing can help the user interfaces we create be more effective.

Of course, someone needs to create the content and maintain it—don't underestimate the resources required to do this successfully. In addition, I want to call your attention to two content-related specialists on the modern web development team: the Content Strategist and Information Architect (IA).

When the content isn't written right, the site can't be fully effective. A [Content Strategist](#) makes sure that every bit of text on a site, from long explanatory text down to the labels on buttons, supports the brand identity and marketing goals of the company. Content strategy may also extend to data modeling and content management on a large and ongoing scale, such as planning for content reuse and update schedules.

An [Information Architect](#) (also called an [Information Designer](#)) organizes the content logically and for ease of findability. She may be responsible for search functionality, site diagrams, and how the content and data is organized on the server. Information architecture is inevitably entwined with UX and UI design, and it is not uncommon for a single person or team to perform all roles.

Multimedia

One of the cool things about the Web is that you can add multimedia elements to a site, including sound, video, animation, and even interactive games. You may decide to add multimedia skills, such as audio and video editing or Flash development (see the "[A Little More About Flash](#)" sidebar), to your web design tool belt, or you may decide to go all in and become a multimedia specialist. If you are not interested in becoming a multimedia developer, you can always hire one. Web development companies usually look for people who have mastered the standard multimedia tools, and have a good visual sensibility and an instinct for intuitive and creative multimedia design.

A Little More About Flash

Adobe Flash (previously Macromedia Flash, previously FutureSplash) is a multimedia format created especially for the Web. Flash is used for create full-screen animation, interactive graphics, integrated audio and video clips, and even scriptable games and applications, all at remarkably small file sizes. However, recently Flash use has been on the decline due to a number of developments, including:

- Apple's decision not to support Flash on its iPhones and iPads in favor of non-proprietary HTML5 methods.
- Adobe's decision to stop supporting Flash (its own product) for mobile browsers.
- The new programmable **canvas** element in HTML5 that offers some of the same functionality as Flash.
- Criticism that Flash sometimes gets in the way of user goals. For example, who wants to sit through a movie and soundtrack on a restaurant site when all you really want to know is whether they are open on Sunday?
- The fact that a plug-in is required to play Flash media makes some developers squeamish.

In fact, it is not uncommon to hear web professionals cite that "Flash is dead," but despite suddenly becoming the underdog, Flash still has some advantages if used the right way:

- Because it uses vector graphics, Flash files are small and can be resized without loss of detail.
- It is a streaming format, so movies start playing quickly and continue to play as they download.
- You can use ActionScript to add behaviors and advanced interactivity, allowing Flash to be used as the frontend for dynamically generated content or ecommerce functions.
- The Flash plug-in is well-distributed on PCs, so support on desktop browsers is reliable.
- Although HTML5 is promising and rapidly evolving, as of this writing, it cannot match the features and performance of Flash.

Flash is not likely to disappear overnight, but even Adobe is putting its muscle behind HTML5 alternatives.

What Languages Do I Need to Learn?

If you are a visual designer who spends time in Photoshop and Illustrator, you may be put off by needing to learn how to create your designs with text, but I assure you, it's pretty simple to get started. There are also authoring tools that speed up the production process, as we'll discuss later in this chapter.

The following is a list of technologies associated with web development. Which languages and technologies you learn will depend on the role you see yourself in within the web design process. However, I advise *everyone* involved in building websites to know their way around HTML and Cascading Style Sheets, and if you want to do frontend web development for a living, JavaScript know-how is pretty much a job requirement. More technically inclined web professionals may take on server configurations, databases, and site performance, but these are generally not frontend developer tasks (although a basic familiarity with the backend issues never hurts).

AT A GLANCE

Web-related technologies:

- Hypertext Markup Language (HTML)
- Cascading Style Sheets (CSS)
- JavaScript and DOM scripting
- Server-side programming and database management

The World Wide Web Consortium

The World Wide Web Consortium (called the W3C for short) is the organization that oversees the development of web technologies. The group was founded in 1994 by Tim Berners-Lee, the inventor of the Web, at the Massachusetts Institute of Technology (MIT).

In the beginning, the W3C concerned itself mainly with the HTTP protocol and the development of the HTML. Now, the W3C is laying a foundation for the future of the Web by developing dozens of technologies and protocols that must work together in a solid infrastructure.

For the definitive answer on any web technology question, the W3C site is the place to go:

www.w3.org

For more information on the W3C and what they do, see this useful page:

www.w3.org/Consortium/

Hypertext Markup Language (HTML)

You may see HTML and XHTML referred to collectively as (X)HTML.

HTML (HyperText Markup Language) is the language used to create web page documents. There are a few versions of HTML in use today: HTML 4.01 is the most firmly established and the newer, more robust HTML5 is gaining steam and browser support. Both versions have a stricter implementation called XHTML (eXtensible HTML), which is essentially the same language with much stricter syntax rules. We'll get to the particulars of what makes the various versions different in [Chapter 10, What's Up, HTML5?](#).

HTML is not a programming language; it is a markup language, which means it is a system for identifying and describing the various components of a document such as headings, paragraphs, and lists. The markup indicates the document's underlying [structure](#) (you can think of it as a detailed, machine-readable outline). You don't need programming skills—only patience and common sense—to write HTML.

The best way to learn HTML is to write out some pages by hand, as we will be doing in the exercises in this book. If you end up working in web production, you'll live and breathe HTML. But even hobbyists will benefit from knowing what is going on under the hood. The good news is that it's simple to learn the basics.

Cascading Style Sheets (CSS)

While HTML is used to describe the content in a web page, it is Cascading Style Sheets (CSS) that describe how that content should *look*. In the web design biz, the way the page looks is known as its [presentation](#). That means fonts, colors, background images, line spacing, page layout, and so on... all controlled with CSS. With the newest version (CSS3), you can even add special effects and basic animation to your page.

CSS also provides methods for controlling how documents will be presented in contexts other than the traditional desktop browser, such as in print and or on devices with small screen widths. It also has rules for specifying the non-visual presentation of documents, such as how they will sound when read by a screen reader (although those are not well supported).

Style sheets are also a great tool for automating production because you can change the way an element looks across all the pages in your site by editing a single style sheet document. Style sheets are supported to some degree by all modern browsers.

Although it is possible to publish web pages using HTML alone, you'll probably want to take on style sheets so you're not stuck with the browser's default styles. If you're looking into designing websites professionally, proficiency at style sheets is mandatory.

Style sheets are discussed further in [Part III](#).

NOTE

When this book says "style sheets" it is always referring to Cascading Style Sheets, the standard style sheet language for the World Wide Web.

JavaScript/DOM scripting

JavaScript is a scripting language that is used to add interactivity and behaviors to web pages, including these (just to name a few):

- Checking form entries for valid entries
- Swapping out styles for an element or an entire site
- Making the browser remember information about the user for the next time she visits
- Building interface widgets, such as expanding menus

JavaScript is used to manipulate the elements on the web page, the styles applied to them, or even the browser itself. There are other web scripting languages, but JavaScript (also called ECMAScript) is the standard and most ubiquitous.

You may also hear the term [DOM scripting](#) used in relation to JavaScript. DOM stands for [Document Object Model](#), and it refers to the standardized list of web page elements that can be accessed and manipulated using JavaScript (or another scripting language). DOM scripting is an updated term for what used to be referred to as DHTML (Dynamic HTML), now considered an obsolete approach.

Writing JavaScript is a type of programming, so it may be time-consuming to learn if you have no prior programming experience. Many people teach themselves JavaScript by reading books and following and modifying existing examples. Most web-authoring tools come with standard scripts that you can use right out of the box for common functions.

Professional web developers are required to know JavaScript, however, plenty of visual designers rely on developers to add behaviors to their designs. So while JavaScript is useful, learning to write it may not be mandatory for *all* web designers. Teaching JavaScript is outside the scope of this book; I recommend [Learning JavaScript](#) by Shelley Powers (O'Reilly, 2006) as a good starting place if you want to learn more.

The Web Design Layer Cake

Contemporary web design is commonly visualized as being made up of three separate "layers."

The content of the document with its (X)HTML markup makes up the [Structure Layer](#). It forms the foundation upon which the other layers may be applied.

Once the structure of the document is in place, you can add style sheets to control how the content should appear. This is called the [Presentation Layer](#).

Finally, the [Behavior Layer](#) includes the scripts that make the page an interactive experience.

Server-side programming

Some simple websites are collections of static HTML documents and image files, but most commercial sites have more advanced functionality such as forms handling, dynamically generated pages, shopping carts, content management systems, databases, and so on. These functions are handled by web applications running on the server. There are a number of programming languages and frameworks (listed in parentheses) that are used to create web applications, including:

- PHP (CakePHP, CodeIgniter, Drupal)
- Python (Django, TurboGears)

- Ruby (Ruby on Rails, Sinatra)
- JavaScript (Node.js, Rhino, SpiderMonkey)
- Java (Grails, Google Web Toolkit, JavaServer Faces)
- ASP.Net (DotNetNuke, ASP.Net MVC)

Developing web applications is programmer territory and is not expected of all web designers. However, that doesn't mean you can't offer such functionality to your clients. It is possible to get shopping carts, content management systems, mailing lists, and blogs as prepackaged solutions, without the need to program them from scratch.

What Do I Need to Buy?

It should come as no surprise that professional web designers require a fair amount of gear, both hardware and software. One of the most common questions I'm asked by my students is, "What should I get?" I can't tell you specifically what to buy, but I will provide an overview of the typical tools of the trade.

Bear in mind that while I've listed the most popular commercial software tools available, many of them have freeware or shareware equivalents that you can download if you're on a budget (try CNET's [Download.com](#)). With a little extra effort, you can get a full website up and running without big cash.

A Quick Introduction to XML

If you hang around the web design world at all, you're sure to hear the acronym [XML](#) (which stands for [eXtensible Markup Language](#)). XML is not a specific language in itself, but rather a robust set of rules for creating other markup languages.

To use a simplified example, if you were publishing recipes, you might use XML to create a custom markup language that includes the elements `<ingredient>`, `<instructions>`, and `<servings>` that accurately describe the types of information in your recipe documents. Once labeled correctly, that information can be treated as data. In fact, XML has proven to be a powerful tool for sharing data between applications. Despite the fact that XML was developed with the Web in mind, it has actually had a larger impact outside the web environment because of its data-handling capabilities. There are XML files working behind the scenes in an increasing number of software applications, such as Microsoft Office, Adobe Flash, and Apple iTunes.

Still, there are a number of XML languages that are used on the Web. The most prevalent is XHTML, which is HTML rewritten according to the stricter rules of XML (we'll talk more about XHTML in [Chapter 10, What's Up, HTML5?](#)). There is also RSS (Really Simple Syndication or RDF Site Summary), which allows your content to be shared as data and read with RSS feed readers; SVG (Scalable Vector Graphics), which uses tags to describe geometric shapes; and MathML, which is used to describe mathematical notation.

As a web designer, your direct experience with XML is likely to be limited to authoring documents in XHTML or perhaps adding an RSS feed or SVG images to a website. Developing new XML languages would be the responsibility of programmers or XML specialists.

Equipment

For a comfortable web development environment, I recommend the following equipment:

A solid, up-to-date computer. Macintosh, Windows, or Linux, is fine. Creative departments in professional web development companies tend to be Mac-based. Although it is nice to have a super-fast machine, the files that make up web pages are very small and tend not to be too taxing on computers. Unless you're getting into sound and video editing, don't worry if your current setup is not the very latest and greatest.

Extra memory. Because you'll tend to bounce between a number of applications, it's a good idea to have enough RAM installed on your computer that allows you to leave several memory-intensive programs running at the same time.

A large monitor. Although not a requirement, a large monitor makes life easier, particularly for a visual designer. (I've seen code-based developers get by just fine on an 11" MacBook Air.) The more monitor real estate you have, the more windows and control panels you can have open at the same time. You can also see more of your page to make design decisions.

If you're using large monitor, just make sure you design for users with smaller monitors and devices in mind.

A scanner and/or digital camera. If you anticipate making your own images and textures, you'll need some tools for creating them. I know a designer who has two scanners: one is the "good" scanner, and the other he uses to scan things like dead fish and rusty pans.

A second computer. Many web designers find it useful to have a test computer running a different platform than the computer they use for development (i.e., if you design on a Mac, test on a PC). Because browsers work differently on Macs than on Windows machines, it's critical to test your pages in as many environments as possible, and particularly on the current Windows operating system. If you are a hobbyist web designer working at home, check your pages on a friend's machine. Mac users should check out the "Run Windows on Your Mac" sidebar.

Mobile devices. The Web has gone mobile! That means it is absolutely critical that you test the appearance and performance of your site on a mobile browser on a smartphone or tablet device. You may already have a smartphone yourself. If you don't have a budget for devices with multiple platforms, ask your friends if you can spend a few minutes looking at your site on theirs. I have one web developer friend who checks out his designs on the phones at his local mobile carrier store (although you might quickly wear out your welcome).

Run Windows on Your Mac

If you have a Macintosh computer with an Intel chip running OS X (Leopard or later), you don't need a separate computer to test in a Windows environment. It is now possible to run Windows right on your Mac using the free Boot Camp application, which allows you to switch to Windows on reboot.

There are several other VM (Virtual Machine) products for Mac OS that allow you to toggle between Mac and Windows, including:

- VMFusion (www.vmware.com/fusion) is a commercial product with a free trial you can download.
- Parallels Desktop for Mac (www.parallels.com) is also a commercial product with a free trial.
- Oracle VirtualBox (virtualbox.org) is a free program that allows you to run a number of guest operating systems, including Windows and several flavors of Unix.

All VM products require that you purchase a copy of Microsoft Windows, but it sure beats buying a whole machine.

Software

There's no shortage of software available for creating web pages. In the early days, we just made do with tools originally designed for print. Today, there are wonderful tools created specifically with web design in mind that make the process more efficient. Although I can't list every available software release, I'd like to introduce you to the most common and proven web design tools. Note that you can download trial versions of many of these programs from the company websites, as listed in the “[Popular Web Design Software Links](#)” sidebar later in this chapter.

Web page authoring

Web-authoring tools are similar to desktop publishing tools, but the end product is a web page (an HTML file and its supporting files). These tools provide a visual “WYSIWYG” (What You See Is What You Get, pronounced “whizzy-wig”) interface and shortcuts that save you from typing repetitive HTML and CSS. These tools won’t excuse you from learning HTML. Even the most sophisticated tools won’t generate HTML as clean or well-considered as a professional writing by hand, but they can speed up the process once you know what you’re doing.

NOTE

To do the exercises in this book, all you'll need is the text editor that came with your operating system. No special programs are required.

The following are some popular web-authoring programs:

Adobe Dreamweaver. This is the hands-down industry standard due to its relatively clean code and advanced features.

Microsoft Expression Web (Windows only). Part of Microsoft’s suite of professional design tools, MS Expression Web boasts standards-compliant code and CSS-based layouts.

Nvu (Linux, Windows, and Mac OS X). Don’t want to pay for a WYSIWYG editor? Nvu (pronounced N-view, for “new view”) is an open source tool that matches many of the features in Dreamweaver, and you can download it for free at nvu.com.

HTML editors

HTML editors (as opposed to WYSIWYG authoring tools) are designed to speed up the process of writing HTML by hand. They do not allow you edit the page visually, so you need to check your work in a browser. Many professional web designers actually prefer to author HTML documents by hand, and they tend to recommend the following:

TextPad (Windows only). TextPad is a simple and inexpensive plain-text code editor for Windows.

Sublime Text (Window, Mac, Linux). This inexpensive and up-and-coming text editor looks stripped down but has a lot of functionality (like color coding and full code overviews) that developers love.

Coda by Panic (Macintosh only). Coda users like its visual workflow, file management tools, and built-in terminal access.

TextMate by MacroMates (Macintosh only). This advanced text editor features project management tools and an interface that is integrated with the Mac operating system. It is growing in popularity because it is customizable, feature-rich, and inexpensive.

BBEdit by Bare Bones Software (Macintosh only). Lots of great shortcut features have made this the leading editor for Mac-based web developers.

Image editing and drawing software

You'll probably want to add images to your pages, so you will need an image-editing program. We'll look at some of the more popular programs in greater detail in Part IV. In the meantime, you may want to look into the following popular web-graphics-creation tools:

Adobe Photoshop. Photoshop is undeniably the industry standard for image creation in both the print and web worlds.

Adobe Photoshop Elements. This lighter version of Photoshop is designed for photo editing and management, but some hobbyists may find that it has all the tools necessary for putting images on web pages.

Adobe Illustrator. Because designers need to create logos, icons, and illustrations at a variety of sizes and resolutions, many start with a vector image in Illustrator for maximum flexibility. You can output web graphics directly from Illustrator, or bring them into Photoshop for additional fine-tuning.

Adobe Fireworks. This web graphics program combines an image editor with tools for creating vector-based illustrations. It also features advanced tools for outputting web graphics.

Corel Paint Shop Pro Photo (Windows only). This full-featured image editor is popular with the Windows crowd, primarily due to its low price.

GIMP, “GNU Image Manipulation Program” (Unix, Windows, Mac). This free image-editing program is similar to Photoshop.

Internet tools

Because you will be dealing with the Internet, you need to have some tools specifically for viewing and moving files over the network:

A variety of browsers. Because browsers render pages differently, you'll want to test your pages on as many browsers as possible, both on the desktop and on mobile devices. The following lists the desktop browsers most commonly used on Windows and Macintosh operating systems:

Windows:	Macintosh OS X:
Internet Explorer (the current version and at least two prior versions)	Safari
Chrome	Chrome
Firefox	Firefox
Safari	Opera
Opera	

And don't ignore the mobile browsers! The following list is an overview of the most commonly used mobile web browsers as of this writing (although who knows what mobile browsers will be important by the time you read this):

- Mobile Safari (iOS)
- Android Browser (Android)
- BlackBerry Browser (RIM)
- Nokia Series 40 and Nokia Browser for Symbian
- Opera Mobile and Mini (installed on any device)
- Internet Explorer Mobile (Windows Phone)
- Silk (Kindle Fire)

A file-transfer program (FTP). An FTP program enables you to upload and download files between your computer and the computer that will serve your pages to the web. The web authoring tools listed earlier all have FTP programs built right in. There are also dedicated FTP programs, as listed here:

Windows	Macintosh OS X:
WS_FTP	Transmit
CuteFTP	Cyberduck
AceFTP	Fetch
Filezilla	

Terminal application. If you know your way around the Unix operating system, you may find it useful to have a terminal (command-line) application that allows you to type Unix commands on the server. This may be useful for setting file permissions, moving or copying files and directories, or managing the server software.

Windows users can install a Linux emulator called Cygwin for command-line access. There is also PuTTY, a free Telnet/SSH client. Mac OS X includes an application called Terminal that is a full-fledged terminal application, giving you access to the underlying Unix system and the ability to use SSH to access other command-line systems over the Internet.

AT A GLANCE

Popular Web Design Software Links

Web page authoring

Adobe Dreamweaver www.adobe.com

Microsoft Expression Web www.microsoft.com/products/expression

Nvu (open source web page editor) www.nvu.com

HTML editing

TextMate by MacroMates for Mac OS www.macromates.com

Sublime Text www.sublimetext.com

TextPad for Windows www.txtpad.com

Coda by Panic Software www.panic.com/coda/

BBEdit by Bare Bones Software www.barebones.com

Image editing and drawing

Adobe Photoshop www.adobe.com

Adobe Photoshop Elements www.adobe.com

Adobe Illustrator www.adobe.com

Adobe Fireworks www.adobe.com

Corel Paint Shop Pro Photo www.corel.com/paintshoppro

GIMP gimp.org

Browsers

Microsoft Internet Explorer (Windows only) www.microsoft.com/windows/internet-explorer/

Firefox www.firefox.com

Google Chrome www.google.com/chrome

Opera www.opera.com

Safari www.apple.com/safari

Networking

WS_FTP, CuteFTP, AceFTP, and others for Windows available at: www.download.com

Transmit (for Macintosh OSX) www.panic.com/transmit

Cyberduck (for Macintosh OSX) cyberduck.ch

Fetch (for Macintosh OSX) fetchsoftworks.com

Cygwin (Linux emulator for Windows) www.cygwin.com

PuTTY (telnet/SSH terminal emulator) www.chiark.greenend.org.uk/~sgtatham/putty/

What You've Learned

The lesson to take away from this chapter is: “You don’t have to learn everything.” And even if you want to learn everything eventually, you don’t need to learn it all at once. So relax, and don’t worry. The other good news is that, while many professional tools exist, it is possible to create a basic website and get it up and running without spending much money by using freely available or inexpensive tools and your existing computer setup.

As you’ll soon see, it’s easy to get started making web pages—you will be able to create simple pages by the time you’re done reading this book. From there, you can continue adding to your bag of tricks and find your particular niche in web design.

exercise 1-1 | Taking stock

Now that you're taking that first step in learning web design, it might be a good time to take stock of your assets and goals. Using the lists in this chapter as a general guide, try jotting down answers to the following questions:

- What are your web design goals? To become a professional web designer? To make personal websites only?
- Which aspects of web design interest you the most?
- What current skills do you have that will be useful in creating web pages?
- Which skills will you need to brush up on?
- Which hardware and software tools do you already have for web design?
- Which tools do you need to buy? Which tools would you like to buy eventually?

Test Yourself

Each chapter in this book ends with a few questions that you can answer to see if you picked up the important bits of information. Answers appear in [Appendix A](#).

1. Match these web professionals with the final product they might be responsible for producing.

A. Graphic designer	_____ HTML and CSS documents
B. Production department	_____ PHP scripts
C. User experience designer	_____ Photoshop page sketch
D. Web programmer	_____ Storyboards
2. What does the W3C do?
3. Match the web technology with its appropriate task:

A. HTML	_____ Checks a form field for a valid entry
B. CSS	_____ Creates a custom server-side web application
C. JavaScript	_____ Identifies text as a second-level heading
D. PHP	_____ Defines a new markup language for sharing financial information
E. XML	_____ Makes all second-level headings blue
4. What is the difference between [frontend](#) and [backend](#) web development?
5. What is the difference between a web-authoring program and an HTML-editing tool?

Want to read more?

You can [buy this book](#) at [oreilly.com](#)
in print and ebook format.

Buy 2 books, get the 3rd FREE!

Use discount code: OPC10

All orders over \$29.95 qualify for **free shipping** within the US.

It's also available at your favorite book retailer,
including the iBookstore, the [Android Marketplace](#),
and [Amazon.com](#).



O'REILLY®

Spreading the knowledge of innovators

[oreilly.com](#)