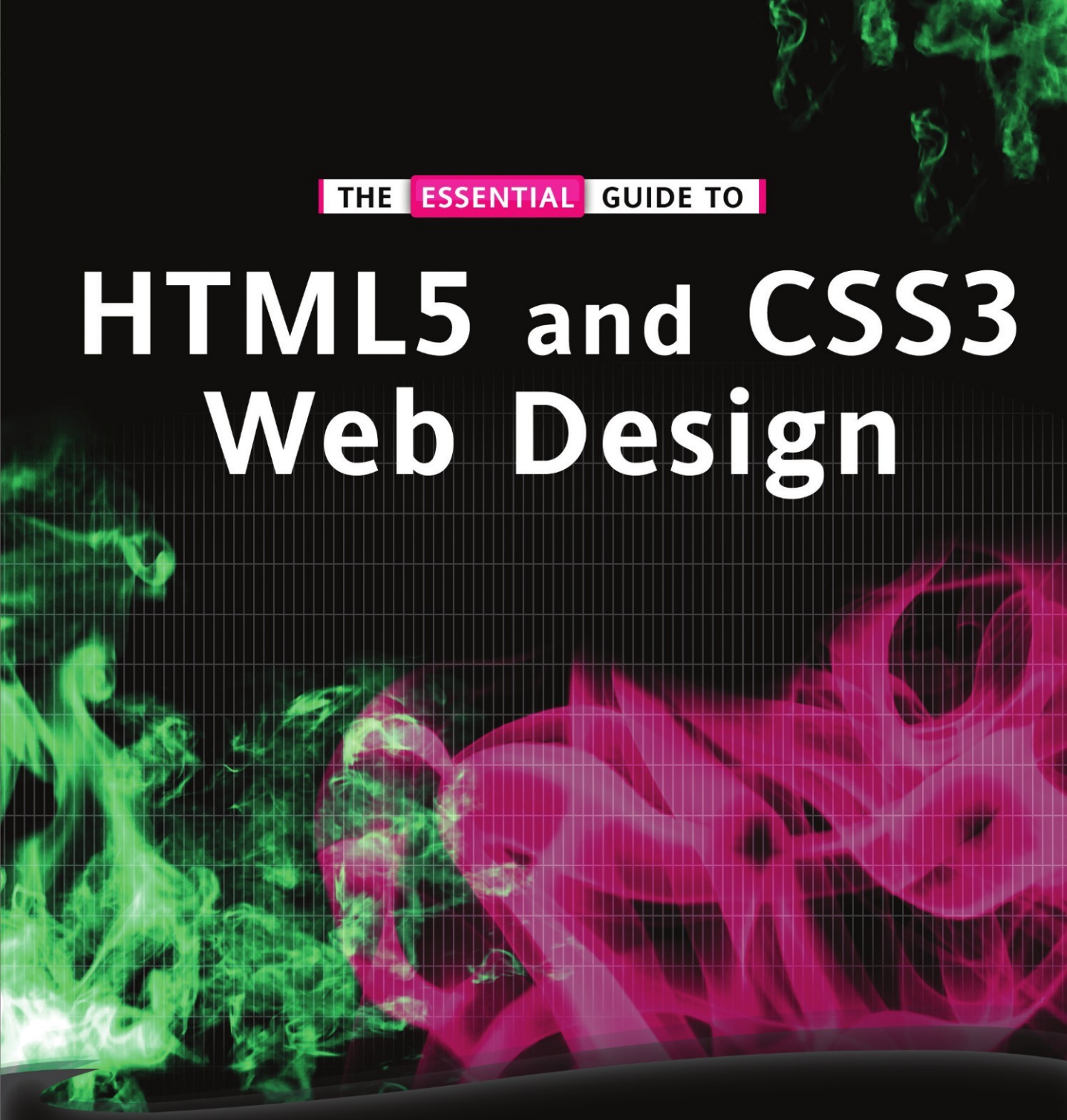
THE ESSENTIAL GUIDE TO

HTML5 and CSS3 Web Design



CRAIG GRANNELL, VICTOR SUMNER
AND DIONYSIOS SYNODINOS

friendsof
an Apress® company

The Essential Guide to HTML5 and CSS3 Web Design

Craig Grannell

Victor Sumner

Dionysios Synodinos

The Essential Guide to HTML5 and CSS3 Web Design

ISBN 978-1-4302-3786-0

ISBN 978-1-4302-3787-7 (eBook)

Distributed to the book trade worldwide by Springer Science+Business Media New York, 233 Spring Street, 6th Floor, New York, NY 10013. Phone 1-800-SPRINGER, fax (201) 348-4505, e-mail orders-ny@springer-sbm.com, or visit www.springeronline.com.

For information on translations, please e-mail rights@apress.com or visit www.apress.com.

Apress and friends of ED books may be purchased in bulk for academic, corporate, or promotional use. eBook versions and licenses are also available for most titles. For more information, reference our Special Bulk Sales–eBook Licensing web page at www.apress.com/bulk-sales.

Any source code or other supplementary materials referenced by the author in this text is available to readers at www.apress.com. For detailed information about how to locate your book's source code, go to www.apress.com/source-code.

Credits

President and Publisher:
Paul Manning

Lead Editor:
Tom Welsh

Technical Reviewer:
Jeffrey Sambells

Editorial Board:
Steve Anglin, Ewan Buckingham, Gary Cornell, Louise Corrigan, Morgan Ertel, Jonathan Gennick, Jonathan Hassell, Robert Hutchinson, Michelle Lowman, James Markham, Matthew Moodie, Jeff Olson, Jeffrey Pepper, Douglas Pundick, Ben Renow-Clarke, Dominic Shakeshaft, Gwenan Spearing, Matt Wade, Tom Welsh

Coordinating Editors:
Jessica Belanger, Anamika Panchoo

Copy Editor:
Kim Wimpsett

Compositor:
Bytheway Publishing Services

Indexer:
SPi Global

Artist:
SPi Global

Cover Image Artist:
Corné van Dooren

Cover Designer:
Anna Ishchenko

Dedicated to my grandmother, Ellen, whose passion for life has always inspired me to take on any challenge.

—Victor Sumner

I dedicate this book to my wonderful family.

To my loving mother, Aggeliki.

To my beautiful wife, Elisa.

To my beloved daughter, Aggeliki.

To my precious newborn son.

You make me feel like the luckiest person alive.

—Dionysios Synodinos

Contents at a Glance

Contents

About the Authors

Craig Grannell is a writer and designer. Originally trained in the fine arts, the mid-1990s saw Craig immersed in the world of digital media, his creative projects encompassing video, installation-based audio work, and strange live performances-sometimes with the aid of a computer, televisions, videos, and a PA system, and sometimes with a small bag of water above his head. His creative, playful art, which contained a dark, satirical edge, struck a chord with those who saw it, leading to successful appearances at a number of leading European media arts festivals.

Craig soon realized he'd actually have to make a proper living, however. Luckily, the Web caught his attention, initially as a means to promote his art via an online portfolio but then as a creative medium in itself, and he's been working with it ever since. He founded tiny studio Snub Communications (www.snubcommunications.com) and has subsequently worked on design and writing projects for a diverse range of clients.

Along with writing the original version of the book you're holding right now (this version ably updated by Victor Sumner and Dionysios Synodinos), Craig has authored *Web Designer's Reference* (friends of ED, 2005) and various books on Dreamweaver. Elsewhere, he's penned numerous articles for Computer Arts, MacFormat, .net, Digital Arts, TechRadar, Tap!, and many other publications besides.

When not designing websites, Craig can usually be found hard at work in his quest for global superstardom by way of his eclectic audio project, the delights of which you can sample at www.projectnoise.co.uk.

Victor Sumner is a senior software engineer at LookSmart, LTD, helping to build and maintain an online advertising platform. As a self-taught developer, he is always interested in emerging technologies and enjoys working on and solving problems that are outside his comfort zone.

When not at the office, Victor has a number of hobbies, including photography, horseback riding, and gaming. He lives in Ontario, Canada, with his wife, Alicia.

Dionysios Synodinos is the research platform team lead at C4Media and a freelance consultant, focusing on rich Internet applications, web application security, mobile web, and web services. He's the lead editor for HTML5 and JavaScript for InfoQ, where he regularly writes about the JVM platform. He's also the author of *Pro HTML5 and CSS3 Design Patterns*, published by Apress. Going back and forth between server-side programming and UI design for more than a decade, he has been involved in diverse software projects and has contributed to different technical publications.

About the Technical Reviewer

"I've seen the future. It's in my browser!"

Jeffrey Sambells does what he loves. He is a father, designer, developer, author, and entrepreneur, among many other things. He started dabbling in the Web more than a decade ago and has turned it into a passion, pushing the limits of what's possible. With an expertise in creating slick end-to-end user experiences, Jeffrey is always on top of the latest technologies, especially when it comes to mobile devices.

You can probably find him writing something interesting at http://jeffreysambells.com or possibly catch him working on a stealth project via Twitter's @iamamused.

About the Cover Image Artist

Corné van Dooren designed the front cover image for this book. After taking a break from friends of ED to create a new design for the Foundation series, he worked at combining technological and organic forms, with the results now appearing on the cover of this and other books.

Corné spent his childhood drawing on everything at hand and then began exploring the infinite world of multimedia—and his journey of discovery hasn't stopped since. His mantra has always been "the only limit to multimedia is the imagination," a saying that keeps him moving forward constantly.

Corné works for many international clients, writes features for multimedia magazines, reviews and tests software, authors multimedia studies, and works on many other friends of ED books. If you like Corné's work, be sure to check out his chapter in *New Masters of Photoshop: Volume 2* (friends of ED, 2004). You can see more of his work (and contact him) at his website, www.cornevandooren.com.

Acknowledgments

I would like to thank the Apress team for providing invaluable support putting this book together. Also, thanks to my wife and soul mate, Alicia, who continues to be supportive of everything I do and who inspires me to always do better.

—Victor Sumner

I'd like to thank Petros Efstathopoulos for motivating me to buy my first HTML book back in 1996 and for doing our first web programming together.

—Dionysios Synodinos

Introduction

The Web is an ever-changing, evolving entity, and it's easy to get left behind. As designers and writers, we see a lot of books on web design, and although many are well written, few are truly integrated, modular resources that anyone can find useful in their day-to-day work. Most web design books concentrate on a single technology (or, commonly, a piece of software), leaving you to figure out how to put the pieces together.

This book is different

The Essential Guide to HTML5 and CSS3 Web Design provides a modern, integrated approach to web design. Each of the chapters looks at a specific aspect of creating a web page, such as formatting type, working with images, creating navigation, and creating layout blocks. In each case, relevant technologies are explored in context and at the appropriate times, just like in real-world projects; for example, markup is explored along with associated CSS and JavaScript, rather than each technology being placed in separate chapters, and visual design ideas are discussed so you can get a feel for how code affects page layouts. Dozens of practical examples are provided, which you can use to further your understanding of each subject. This highly modular and integrated approach means you can dip in and out of the book as you need, crafting along the way a number of web page elements that you can use on countless sites in the future.

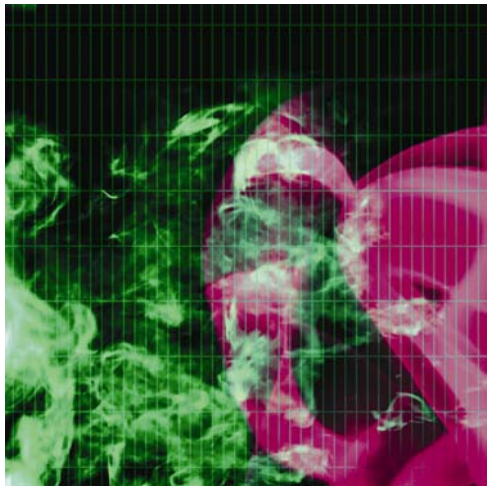
Because the entire skills gamut is covered—from foundation to advanced—this book is ideal for beginners and longtime professionals alike. If you're making your first move into standards-based web design, the "ground floor" is covered, rather than an assumption being made regarding your knowledge. However, contemporary ideas, techniques, and thinking are explored throughout, ensuring that the book is just as essential for the experienced designer wanting to work on CSS layouts or for the graphic designer who wants to discover how to create cutting-edge websites.

This book's advocacy of web standards, usability, and accessibility with a strong eye toward visual design makes it of use to technologists and designers alike, enabling everyone to build better websites. For those moments when a particular tag or property value slips your mind, this book provides a comprehensive reference guide that includes important and relevant HTML5 elements and attributes, HTML5 entities, web colors, and CSS 3 properties and values.

Code Examples

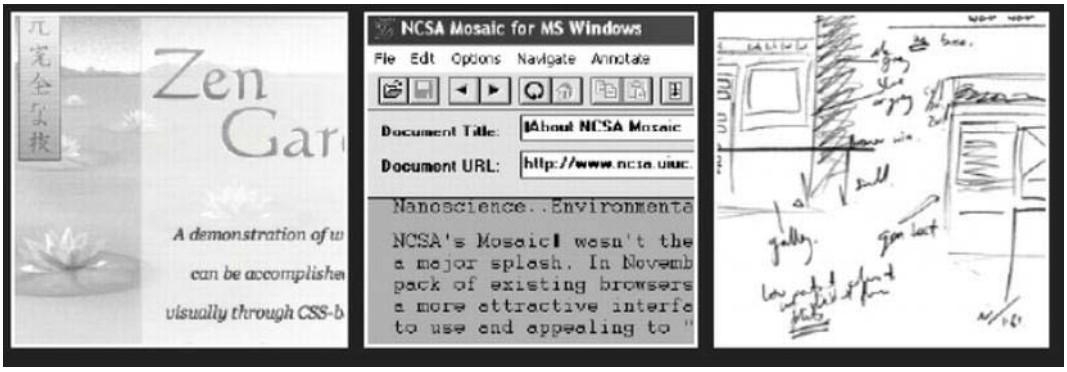
Remember that you can also download files associated with this book from www.apress.com—just find the book and follow its instructions to access downloads and other associated resources.

To make it easier to work through the exercises, each one has an introductory box that lists where you can find any required files and the completed files within the downloadable file archive. A short overview of what you'll learn is also included.



Chapter 1

An Introduction to Web Design



In this chapter:

- Introducing the Internet and web design
- Working with web standards
- Working with HTML
- Understanding and creating CSS rules
- Creating web page boilerplates

- Organizing web page content

A brief history of the Internet

Even in the wildest dreams of science-fiction and fantasy writers, few envisioned anything that offers the level of potential that the Internet now provides for sharing information on a worldwide basis. For both businesses and individuals, the Internet is now the medium of choice, largely because it enables you to present your wares to the entire world on a 24/7 basis. But the technology's origins were more ominous than and very different from the ever-growing, sprawling free-for-all that exists today.

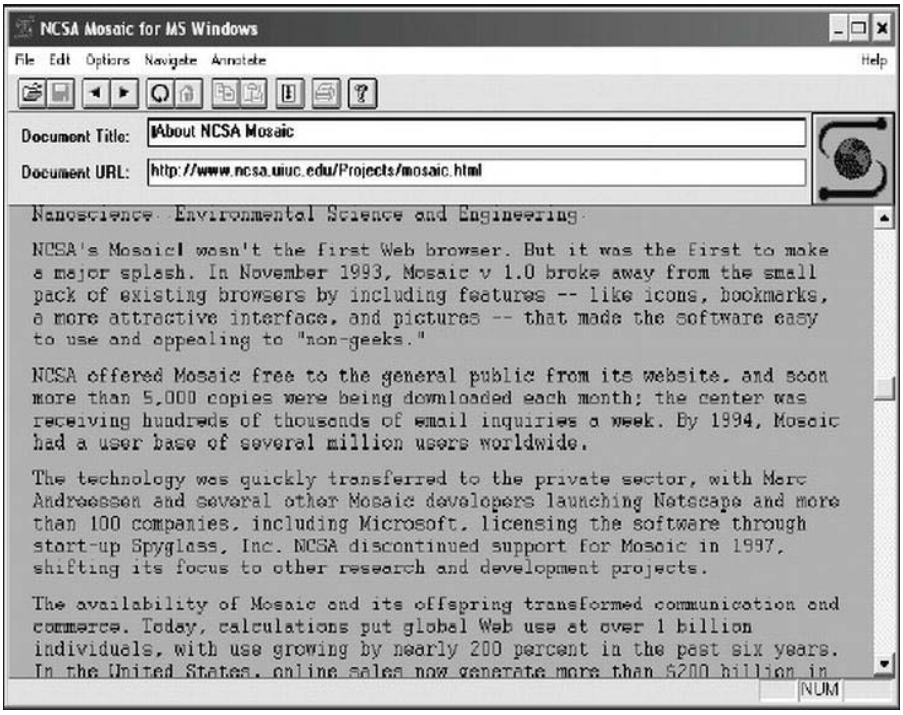
In the 1960s, the American military was experimenting with methods by which the U.S. authorities might be able to communicate in the aftermath of a nuclear attack. The suggested solution was to replace point-to-point communication networks with one that was more akin to a net. This meant information could find its way from place to place even if certain sections of the network were destroyed. Despite the project eventually being shelved by the Pentagon, the concept itself lived on, eventually influencing a network that connected several American universities.

During the following decade, this fledgling network went international and began opening itself up to the general public. The term Internet was coined in the 1980s, which also heralded the invention of Transmission Control Protocol/Internet Protocol (TCP/IP), the networking software that makes possible communication between computers running on different systems. During the 1980s, Tim Berners-Lee was also busy working on HTML, his effort to weld hypertext to a markup language in an attempt to make communication of research between himself and his colleagues simpler.

Despite the technology's healthy level of expansion, the general public remained largely unaware of the Internet until well into the 1990s. By this time, HTML had evolved from a fairly loose set of rules—browsers having to make assumptions regarding coder intent and rendering output—to a somewhat stricter set of specifications and recommendations. This, along with a combination of inexpensive hardware, the advent of highly usable web browsers such as Mosaic (see the following image), and improved communications technology, saw an explosion of growth that continues to this day.

Initially, only the largest brands dipped their toes into these new waters, but soon thousands of companies were on the Web, enabling customers all over the globe to access information and, later, to shop online. Home users soon got in on the act, once it became clear that the basics of web design weren't rocket science and that, in a sense, everyone could do it—all you needed was a text editor, an FTP client, and some web space. Designers soon got in on the act, increasingly catered for by new elements within HTML; Cascading Style Sheets (CSS), which took a while to be adopted by browsers but eventually provided a means of creating highly advanced layouts for the Web; and faster web connections, which made media-rich sites accessible to the general public without forcing them to wait ages for content to download.

Therefore, unlike most media, the Web is truly a tool for everyone, and in many countries, the Internet has become ubiquitous. For those working in a related industry, it's hard to conceive that as recently as the mid-1990s relatively few people were even aware of the Internet's existence!



So, from obscure roots as a concept for military communications, the Internet has evolved into an essential tool for millions of people, enabling them to communicate with each other, research and gather information, telecommute, shop, play games, and become involved in countless other activities on a worldwide basis.

Why create a website?

Before putting pen to paper (and mouse to keyboard), it's important to think about the reason behind putting a site online. Millions already exist, so why do you need to create one yourself? Also, if you're working for a company, perhaps you already have plenty of marketing material, so why do you need a website as well?

I should mention here that I'm certainly not trying to put you off—far from it. Instead, I'm trying to reinforce the point that planning is key in any web design project, and although some people swear that "winging it" is the best way to go, most such projects end up gathering virtual dust online. Therefore, before doing anything else, think through why you should build a website and what you're trying to achieve.

Companies and individuals alike have practical and commercial reasons for setting up a website. A website enables you to communicate with like-minded individuals or potential clients on a worldwide basis. If you're a creative talent of some kind, you can use a website to showcase your portfolio, offering online photographs, music tracks for download, or poetry. If you fancy yourself as a journalist, a blog enables you to get your opinion out there. If you own or work for a business, creating a website is often the most

efficient means of marketing your company. And even if you just have a hobby, a website can be a great way of finding others who share your passion—while you may be the only person in town who likes a particular movie or type of memorabilia, chances are there are thousands of people worldwide who think the same, and a website can bring you all together. This is perhaps why the paper fanzine has all but died, only to be reborn online, where development costs are negligible and worldwide distribution is a cinch.

In practical terms, a website exists online all day, every day (barring the odd hiccup with ISPs), which certainly isn't the case with printed media, which is there one minute and in the recycle trash the next. Distribution is less expensive than sending out printed material—a thousand-page website can be hosted for $10 per month or less, but sending a thousand-page document to one person (let alone a thousand or several thousand) may cost more than that. Likewise, development (particularly corrections and updates) is often significantly cheaper, too. For example, if you want to rework a print brochure, you have to redesign it and then reprint it. Reworking a section of a website often means swapping out a few files, which is efficient and affordable. So, for large companies and individuals alike, the ability to have relevant information online in a form that can often be updated in mere minutes, thereby keeping all interested parties up-to-date, is hard to resist!

Audience requirements

This book centers on the design and technology aspects of web design, but close attention must always be paid to your potential audience. It's no good forcing design ideas that result in inappropriate visuals, unusable navigation to all but the most technically minded of people, and huge download times on your site's unsuspecting visitors.

Prior to creating a site, you must ascertain what your audience wants and expects in terms of content, design, and how the site will work (by way of talking to the relevant people, and also, if your budget allows, by using surveys and focus groups). You don't have to take all of your audience's ideas into account (after all, many will be contradictory), but be mindful of common themes and ensure they're not ignored.

Technical considerations must be researched. If you're targeting designers, you can be fairly sure that a large proportion of the audience will be using monitors set to a high resolution and millions of colors, and you can design accordingly. If your site is targeting mobile users, be mindful that it will be displayed on a wide range of devices. From tablets and smartphones with high-resolution Retina or PenTile technology displays to those with low-resolution LCD displays, mobile devices come in all shapes, sizes, and capabilities.

Determining the web browsers your audience members use is another important consideration. Although use of web standards (used throughout this book) is more likely to result in a highly compatible site, browser quirks still cause unforeseen problems; therefore, always check to see what browsers are popular with a site's visitors, and ensure you test in as many as you can. Sometimes you won't have access to such statistics, or you may just be after a "sanity check" regarding what's generally popular. A couple of useful places to research global web browser statistics are `www.w3schools.com/browsers/browsers_stats.asp` and `www.upsdell.com/BrowserNews/`. Note, though, that any statistics you see online are effectively guesswork and are not a definitive representation of the

Web as a whole; still, they do provide a useful, sizeable sample that's often indicative of current browser trends.

Although you might be used to checking browser usage and then, based on the results, designing for specific browsers, we'll be adhering closely to web standards throughout this book. When doing this, an "author once, work anywhere" approach is feasible, as long as you're aware of various browser quirks (many of which are explored in Chapter 9). Of course, you should still always ensure you test sites in as many browsers as possible, just to make sure everything works as intended.

Web design overview

Web design has evolved rapidly over the years. Initially, browsers were basic, and early versions of HTML were fairly limited in what they enabled designers to do. Therefore, many older sites on the Web are plain in appearance. Additionally, the Web was originally largely a technical repository, which is the reason for the boring layouts of many sites in the mid-1990s; after all, statistics, documentation, and papers rarely need to be jazzed up, and the audience didn't demand such things anyway.

As with any medium finding its feet, things soon changed, especially once the general public flocked to the Web. It was no longer enough for websites to be text-based information repositories. Users craved—demanded, even—color! Images! Excitement! Animation! Interaction! Even video and audio managed to get a foothold as compression techniques improved and connection speeds increased.

The danger of eye candy became all too apparent as the turn of the century approached: every site, it seemed, had a Flash intro, and the phrase "skip intro" became so common that it eventually spawned a parody website.

These days, site design has tended toward being more restrained, as designers have become more comfortable with using specific types of technologies for relevant and appropriate purposes. Therefore, you'll find beautifully designed HTML- and CSS-based sites sitting alongside highly animated Flash efforts. Also, with the increasing popularity of JavaScript and the introduction of CSS Transitions and HTML5 Canvas, Flash appears to be on the way out because Adobe has recently discontinued support for Flash on mobile devices.

Of late, special emphasis is being placed on usability and accessibility, and in the majority of cases, designers have cottoned to the fact that content must take precedence. However, just because web standards, usability, and accessibility are key, that doesn't mean design should be thrown out the window. As we'll see in later chapters, web standards do not have to come at the expense of good design—far from it. In fact, a strong understanding of web standards helps improve websites, making it easier for you to create cutting-edge layouts that work across platforms and are easy to update. It also provides you with a method of catering for obsolete devices.

Note: If you're relatively new to web design, you may be wondering about the best platform and software for creating websites. Ultimately, it matters little which platform you choose, as long as you have access to the most popular browsers for testing purposes (a list that I'd now include Apple's Safari in, alongside Chrome, Internet Explorer, Firefox, and Opera). Regarding software, there's an overview in Appendix E, but this isn't an exhaustive guide, so do your own research and find software to your liking.

Why WYSIWYG tools aren't used in this book

With lots of software available and this book being design-oriented, you might wonder why I'm not using WYSIWYG web design tools. This isn't because I shun such tools—it's more that in order to best learn how to do something, you need to start from scratch, with the foundations. Many web design applications make it tempting to "hide" the underlying code from you, and most users end up relying on the graphical interface. This is fine until something goes wrong and you don't know how to fix it.

Removing software from the equation also means we concentrate on the underlying technology that drives web pages, without the distraction of working out which button does what. It also ensures that the book will be relevant to you, regardless of what software you use or your current skill level. Therefore, I suggest you install a quality text editor to work through the exercises or set your web design application to use its code view. Once you're familiar with the concepts outlined in this book, you can apply them to your work, whatever your chosen application for web design. This level of flexibility is important, because you never know when you might have to switch applications—something that's relatively painless if you know how to design for the Web and understand technologies like CSS and HTML.

Introducing HTML5

The foundation of the majority of web pages is HyperText Markup Language, commonly known by its initials, HTML. A curious facet of the language is that it's easy to pick up the basics—anyone who's computer literate should be able to piece together a basic page after learning some tags—but it has enough flexibility and scope to keep designers interested and experimenting, especially when HTML is combined with Cascading Style Sheets (CSS), which we'll discuss later in this chapter.

The HTML5 syntax is designed to be simpler, more flexible, developer-friendly, and backward-compatible than HTML4 and XHTML. HTML5 introduces new features such as animation, offline capabilities, audio, advanced graphics, typography, transitions, and more, which yields a new class of web standards and replaces the need for proprietary technologies, like Flash and native mobile platforms.

Introducing the concept of HTML tags and elements

HTML documents are text files that contain tags, which are used to mark up HTML elements. These documents are usually saved with the `.html` file extension, although other extensions like `.htm can be used.`

The aforementioned tags are what web browsers use to display pages, and assuming the browser is well behaved (most modern ones are), the display should conform to standards as laid out by the World Wide Web Consortium (W3C), the organization that develops guidelines and specifications for many web technologies.

> *Note: The W3C website is found at `www.w3.org`. The site offers numerous useful tools, including validation services against which you can check your web pages.*

HTML tags are surrounded by angle brackets—for instance, <p> is a paragraph start tag. It's good practice to close tags once the element content or intended display effect concludes, and this is done with an end tag. End tags are identical to the opening start tags but with an added forward slash: /. A complete HTML element looks like this:

```
<p>Here is a paragraph.</p>
```

This element consists of the following:

- Start tag: `<p>`
- Content: `Here is a paragraph.`
- End tag: `</p>`

> *Note: HTML doesn't have a hard-and-fast rule regarding the case of tags. If you look at the source code of HTML pages on the Web, you may see lowercase tags, uppercase tags, or, in the case of pages put together over a period of time, a mixture of the two. That said, it's still good practice with any markup language to be consistent, regardless of whether the rules are more flexible.*

Nesting tags

There are many occasions when tags must be placed inside each other; this process is called *nesting*. One reason for nesting is to apply basic styles to text-based elements. Earlier, you saw the code for a paragraph element. We can now make the text bold by surrounding the element content with a `strong` element:

```
<p><strong>Here is a paragraph.</strong></p>
```

> *You might be used to using the bold element to make text bold, but it is a physical element that only amends the look of text rather than also conveying semantic meaning. Logical elements, such as strong, convey meaning and add styling to text and are therefore preferred. These will be covered in Chapter 3.*

Note that the strong tags are nested within the paragraph tags ( ), not the other way around. That's because the paragraph is the parent element to which formatting is being applied. The paragraph could be made bold and italic by adding another element, emphasis (), as follows:

```
<p><strong><em>Here is a paragraph.</em></strong></p>
```

In this case, the `strong` and `em` tags could be in the opposite order, because they're at the same level in the hierarchy. However, you must always close nested tags in the reverse order to that in which they're opened, as shown in the previous code block; otherwise, some browsers may not display your work as intended. For instance, the following should be avoided:

```
<p><strong><em>Here is a paragraph.</strong></em></p>
```

As previously mentioned, it's good practice to close tags in HTML—even though it's not a requirement for all elements, being sloppy in this area can lead to errors. Take a look at the following:

```
<p><strong><em>Here is a paragraph.</strong></p>
```

Here, the emphasis element isn't closed, meaning subsequent text-based content on the page is likely to be displayed in italics—so take care to close all your tags.

Web standards and HTML

HTML5 is an updated version of the HTML specification that has been around since 1997 and many of its features are already supported in today's browsers. The changes in HTML5 include a focus on semantic markup like the addition of the <header>, <footer>, <section>, and <article> elements and also the addition of the <canvas> element for displaying advanced interactive graphics and the <video> element for displaying video. Websites like html5please.com, caniuse.com, and of coarse the WC3 working draft (http://dev.w3.org/html5/html4-differences/) are great resources for finding out what has changed, what is new, or what is supported in each browser.

HTML5 markup can be defined in whatever way you want it to be. Uppercase, lowercase, quoted, unquoted, self-closing or not—it's your choice. The ultimate goal is semantic markup, ensuring the elements you choose and the style of your markup define the meaning of your content as closely as possible.

Evolution is another aspect that we have to deal with. Just as the survival of the fittest removes some species from nature, so too are tags (and attributes) unceremoniously dumped from the W3C specifications. Such tags and attributes are referred to as *deprecated*, meaning they are marked for removal from the standard and may not be supported in future browsers. In the case of HTML5 obsolete tags and attributes are still supported because of HTML5's backward-compatibility, it is still recommended

that you do not use such tags and attributes because new implementations of browsers may choose not to support them.

Semantic markup

In the previous few subsections, you may have noticed specific elements being used for specific things. This is referred to as *semantic* markup and is a very important aspect of modern web design. Plenty of HTML elements exist, and each one has a clearly defined purpose (although some have more than one use). Because of the flexibility of markup languages, it's often possible to "wrongly" use elements, bashing your page into shape by using elements for design tasks they're not strictly suited for and certainly weren't originally designed for.

During the course of this book, we'll talk about semantics a fair amount. Ultimately, good semantic design enables you to simplify your markup and also provides the greatest scope for being able to style it with CSS (see the following section). By thinking a little before you code and defining your content with the correct markup, you'll end up with cleaner code and make it much easier for yourself in the long run when it comes to adding presentation to your content.

Introducing CSS

CSS is the W3C standard for defining the visual presentation for web pages. HTML was designed as a structural markup language, but the demands of users and designers encouraged browser manufacturers to support and develop presentation-oriented tags. These tags "polluted" HTML, pushing the language toward one of decorative style rather than logical structure. Its increasing complexity made life hard for web designers, and source code began to balloon for even basic presentation-oriented tasks. Along with creating needlessly large HTML files, things like font tags created web pages that weren't consistent across browsers and platforms, and styles had to be applied to individual elements—a time-consuming process.

The concept behind CSS was simple yet revolutionary: remove the presentation and separate design from content. Let HTML deal with structure, and use CSS for the application of visual presentation.

The idea caught on albeit slowly. The initial problem was browser support. At first, most browsers supported only a small amount of the CSS standard—and badly at that. But Internet Explorer 5 for Mac made great strides with regard to CSS support, and it was soon joined by other browsers fighting for the crown of standards king. These days, every up-to-date browser supports the majority of commonly used CSS properties and values, and more besides.

Another problem has been educating designers and encouraging them to switch from old to new methods. Benefits constantly need to be outlined and proven, and the new methods taught. Most designers these days style text with CSS, but many still don't use CSS for entire web page layouts, despite the inherent advantages in doing so. This, of course, is one of the reasons for this book: to show you, the designer, how CSS can be beneficial to you—saving you (and your clients) time and money—and to provide examples for various areas of web page design and development that you can use in your sites.

In this section, we'll look at separating content from design, CSS rules, CSS selectors and how to use them, and how to add styles to a web page.

Separating content from design

Do you ever do any of the following?

- Use tables for website layout
- Hack Photoshop documents to bits and stitch them back together in a web page to create navigation elements and more
- Get frustrated when any combination of the previous leads to unwieldy web pages that are a pain to edit

If so, the idea of separating content from design should appeal to you. On one hand, you have your HTML documents, which house content marked up in a logical and semantic manner. On the other hand, you have your CSS documents, giving you sitewide control of the presentation of your web page elements from a single source. Instead of messing around with stretching transparent GIFs and combining and splitting table cells, you can edit CSS rules to amend the look of your site, which is great for not only those times when things just need subtle tweaking but also when you decide everything needs a visual overhaul. After all, if presentation is taken care of externally, you can often just replace the CSS to provide your site with a totally new design.

Designers (and clients paying for their time) aren't the only ones to benefit from CSS. Visitors will, too, in terms of faster download times but also with regard to accessibility. For instance, people with poor vision often use screen readers to surf the Web. If a site's layout is composed of complex nested tables, it might visually make sense; however, the underlying structure may not be logical. View the source of a document, and look at the order of the content. A screen reader reads from the top to the bottom of the code and doesn't care what the page looks like in a visual web browser. Therefore, if the code compromises the logical order of the content (as complex tables often do), the site is compromised for all those using screen readers.

Accessibility is now very important in the field of web design. Legislation is regularly passed to strongly encourage designers to make sites accessible for web users with disabilities. It's likely that this trend will continue, encompassing just about everything except personal web pages. (However, even personal websites shouldn't be inaccessible.)

The rules of CSS

Style sheets consist of a number of rules that define how various web page elements should be displayed. Although sometimes bewildering to newcomers, CSS rules are simple to break down. Each rule consists of a selector and a declaration. The selector begins a CSS rule and specifies which part of the HTML document the rule will be applied to. The declaration consists of a number of property/value pairs that set specific properties and determine how the relevant element will look. In the following example, `p` is the selector, and everything thereafter is the declaration:

```
p {
  color: blue;
}
```

As you probably know, p is the HTML tag for a paragraph. Therefore, if we attach this rule to a web page (see the section "Adding styles to a web page" later in this chapter for how to do so), the declaration will be applied to any HTML marked up as a paragraph, thereby setting the color of said paragraphs to blue.

> *Note: CSS property names are not case sensitive, but it's good to be consistent in web design—it's highly recommended to always use lowercase.*

When you write CSS rules, you place the declaration within curly brackets: {}. Properties and values are separated by a colon (:), and property/value pairs are terminated by a semicolon (;). Technically, you don't have to include the final semicolon in a CSS rule, but most designers consider it good practice to do so. This makes sense—you may add property/value pairs to a rule at a later date, and if the semicolon is already there, you don't have to remember to add it.

If we want to amend our paragraph declaration and define paragraphs as bold, we can do so like this:

```
p {
  color: blue;
  font-weight:bold;
}
```

> *Note: You don't have to lay out CSS rules as done in this section; rather, you can add rules as one long string. However, the formatting shown here is more readable in print. Note that in the files available for download, the formatting is changed slightly again: the property/value pairs and closing curly bracket are both tabbed inward, enabling rapid vertical scanning of a CSS document's selectors.*

Types of CSS selectors

In the previous example, the most basic style of selector was used: an element selector. This defines the visual appearance of the relevant HTML tag. In the sections that follow, we'll examine some other regularly used (and well-supported) CSS selectors: class, ID, grouped, and contextual.

Class selectors

In some cases, you may want to modify an element or a group of elements. For instance, you may want your general website text to be blue, as in the examples so far, but some portions of it to be red. The simplest way of doing this is by using a class selector.

In CSS, a class selector's name is prefixed by a period (.), like this:

```
.warningText {
  color: red;
```

```
}
```

This style is applied to HTML elements in any web page the style sheet is attached to using the `class` attribute, as follows:

```
<h2 class="warningText">This heading is red.</h2>
<p class="warningText">This text is red.</p>
<p>This is a paragraph, <span class="warningText">and this text is
 red</span>.</p>
```

If you want a make a class specific to a certain element, place the relevant HTML tag before the period in the CSS rule:

```
p.warningText {
  color: red;
}
```

If you used this CSS rule with the HTML elements shown previously, the paragraph's text would remain red, but not the heading or span, because of the `warningText` class now being exclusively tied to the paragraph selector only.

Usefully, it's possible to style an element by using multiple class values. This is done by listing multiple values in the `class` attribute, separated by spaces:

```
<p class="warningText hugeText">
```

The previous example's content would be styled as per the rules `.warningText` and `.hugeText`.

ID selectors

ID selectors can be used only once on each web page. In HTML, you apply a unique identifier to an HTML element with the `id` attribute:

```
<p id="footer">&copy; 200X The Company. All rights reserved.</p>
```

To style this element in CSS, precede the ID name with a hash mark (#):

```
p#footer {
  padding: 20px;
}
```

In this case, the footer `div` would have 20 pixels of padding on all sides.

Essentially, then, classes can be used multiple times on a web page, but IDs cannot. Typically, IDs are used to define one-off page elements, such as structural divisions, whereas classes are used to define the style for multiple items.

Grouped selectors

Should you want to set a property value for a number of different selectors, you can use grouped selectors, which take the form of a comma-separated list: