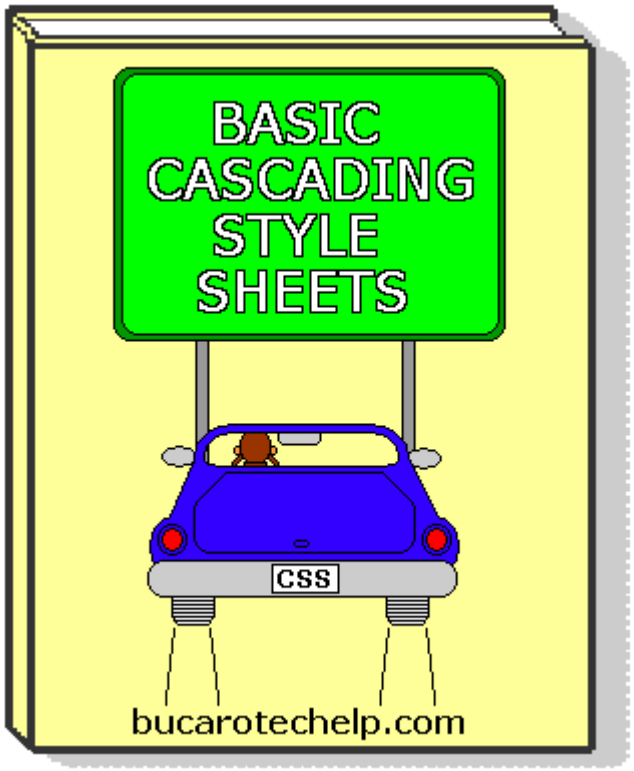


Contents

Warranty, Disclaimer and Copyright Policy

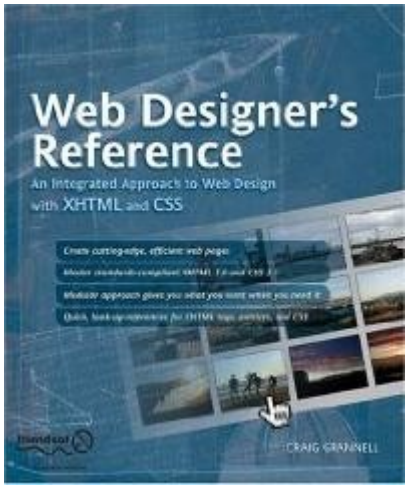
This material is provided on an "as-is" basis, and Bucaro TecHelp makes no warranty or representation, express or implied, with respect to its quality performance or fitness for a particular purpose. In no event shall Bucaro TecHelp be liable for direct, indirect, special, incidental, or consequential damages arising out of the use of this material.

No patent liability is assumed with respect to the use of the information contained herein. Although every precaution has been taken in the preparation of this manual, Bucaro TecHelp assumes no responsibility for errors or omissions. Neither is any liability assumed for damages resulting from the use of the information contained herein. This information is provided with the understanding that Bucaro TecHelp is not engaged in rendering medical, legal, accounting or other professional service. If legal advice or other expert assistance is required, the services of a competent professional person should be sought.

By using this material, the user assumes complete responsibility for any and all damages resulting from that use. Use of this program and materials requires agreement to the terms of this warranty and disclaimer. If you do not agree to the terms of this warranty, do not use this material.

Contents

Web Designer's Reference



Most web design books concentrate on a single technology or piece of software, leaving the designer to figure out how to put all the pieces together. This book is different. Web Designer's Reference provides a truly integrated approach to web design. Each of the dozen chapters covers a specific aspect of creating a web page, such as working with

typography, adding images, creating navigation, and crafting CSS layouts. In each case, relevant XHTML elements are explored along with associated CSS, and visual design ideas are discussed.

Several practical examples are provided, which you can use to further your understanding of each subject. This highly modular and integrated approach means that you learn about technologies in context, at the appropriate time and, upon working through each chapter, you craft a number of web page elements that you can use on countless sites in the future.

This book is ideal for those making their first moves into standards-based web design, experienced designers who want to learn about modern design techniques and move toward creating CSS layouts, graphic designers who want to discover how to lay out their designs, and veteran web designers who want a concise reference guide.

The book's advocacy of web standards, usability, and accessibility with a strong eye toward visual design means itís of use to technologists and designers alike, enabling everyone to build better websites. And for those moments when a particular tag or property value slips your mind, the book provides a comprehensive reference guide that includes important and relevant XHTML elements and attributes, XHTML entities, web colors, and CSS 2.1 properties and values.

With this book:

• Use XHTML elements effectively and efficiently to create lean, standardscompliant, highly compatible web pages.
• Style web pages with CSS.
• Create effective page layouts, highly flexible navigation areas, great online typography, and more.
• Learn how to balance quality contemporary design with future-proof web standards.
• Use the essential reference guides to remind yourself about XHTML elements, CSS properties, and their associated attributes and values

Reader John A. Suda from Rochester, New York says, "It seems as if nearly everyone and his brother is writing books supporting standards-compliant web design with XHTML and CSS. I have read and reviewed a half dozen this year alone. People are obviously trying to tell us something - plain HTML has to go! "Web Designers' Reference: An Integrated Approach to Web Design with XHTML and CSS" by Craig Grannell is the latest of these pronouncements.

The reasons are clear and compelling. The World Wide Web Consortium which promulgates web design standards has decreed HTML as obsolete. Newer, more compliant browsers, will in time not support the older tags and code; the new standards facilitate much better use by the disabled of screen readers and non-graphic browsers. Not least, the newer code makes writing and revising code easier and more efficient, as well as more capable.

These are certainly good reasons for web designers to move to the new code. Nevertheless, surveys show that most web pages are not

compliant and that thousands of designers continue to use deprecated code. I confess that I am one of them. After a number of years learning and getting used to HTML, the need to learn new and more code is onerous. The inertia of habit is a factor I'm sure.

For those web designers like me, Mr. Grannell's book is a welcome addition to the literature because it systematically deals with the topics under discussion. In its coverage of XHTML, CSS, Javascript, and complementary coding like php, it provides a nice framework guiding "old dogs" like me into standards-compliant code. Not only does it provide some historical perspectives on these codes, it compares the old with the new in regard to all of the important elements of web design.

The author is an experienced web designer and operates a design and writing agency. He also writes articles for a number of computer magazines.

Grannell's goals are to teach cutting-edge, efficient coding, and how to master standards-compliant XHTML 1.0 and CSS 2.1. There are a dozen chapters. He breaks down the elements of web design into modular components so that one can focus on each element separately, like page structure, content structure, layout, navigation, text control, user feedback, and multimedia. Relevant technologies are explained in context of producing a typical website.

If one finally decides to move forward, as many suggest, this is a very good volume by which to get your start. It will facilitate a fresh start for the "old dogs". For new designers, this is a nice primer to learn what is expected, in an overall sense, of good, advanced web design.

This is a well-produced book with clear writing, comprehensive approach, dozens of practical examples, and downloadable files with the code examples used in the book. The author writes in a logical sequence much like an engineer would. It is a heavy text-book-like read, only lightly sprinkled with style and personality. It should appeal primarily to novice designers, but has enough advanced information to satisfy an experienced designer who is looking for that fresh start.

The structure of the book facilitates the "fresh-start" idea. It starts with a web design overview giving an experienced user's tips on what software to use to write code, what browsers to design for, how to build pages from the very top to the bottom. (XHTML, unlike HTML, requires a preliminary document-type definition (DTD) to validate. Only after the ntroductory section does the first HTML tag appear.)

Like others writing in this area, he firmly advocates design for standards compliance, usability, accessibility, and last and least, visual design. Marketing Department people may want to choke on that priority list but there is no inherent conflict between function and aesthetics. Grannell does not spend a lot of time on the aesthetics aspect.

The middle chapters concentrate on modular construction of pages -

the XHTML introduction, the structural elements like text blocks and images, the logical structure of the links and navigation flow, and finally, the stylizing with CSS. Comparisons of pages styled with HTML vs. CSS compellingly demonstrate the benefits and advantages of CSS. There will be no going back once you've decided to upgrade your technical approach.

Basic CSS concepts are explained and illustrated with code samples and screenshots. Grannell describes how to use CSS for text control, navigation, and layouts. There is a broad section on frames and another on forms and interactive components.

The last chapter covers testing and tweaking including how to create a 7 item browser test suite. Much time is used throughout the book in discussing overcoming browser quirks. There is detailed technical information, especially in regard to the XHTML introductory section of the page, which I have not seen elsewhere.

There are three welcome reference appendices at the end covering XHTML tags and attributes, web color coding, and a very comprehensive entities chart noting currencies, European characters, math symbols and more.

Much of this material is covered elsewhere in the growing set of publications about standards-compliant code. This book has the virtue of having a useful overall perspective on web design and acts as a framework for new designers and converting designers to renew and upgrade their technical approaches."

Summary of Contents:

- Introduction to Web design with XHTML and CSS
- Web Page Defaults
- Working with Text
- Working with Images
- Creating Navigation
- Introduction to Layout
- Tables for Tabular Data and Layout
- Layouts with CSS
- Working with Frames
- Getting User Input
- Adding Multimedia
- Testing and Uploading Web Pages
- XHTML and CSS Reference Sections

Reader John Woods of San Diego, California says,"Although this book has XHTML in the title, don't let it scare you off, it simply means that the advice author gives is XHTML standard compliant. On the other hand, if you are looking for a comprehensive XHTML guide, you may want to look elsewhere, even though there is a reference section specifically about this standard, which is an XML reformulation of HTML, in one of the appendixes.

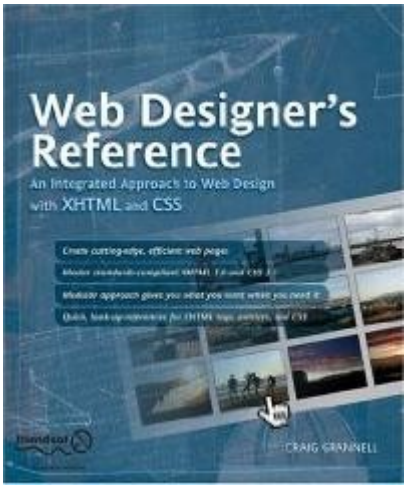
Another acronym in the title - CSS - cascading style sheets is the preferred mechanism for implementation of design on the web. CSS is growing in popularity as it allows separation of content and

presentation on great scale at the same time matching very well to the wide array of http serving technologies from simplest static web pages to dynamic ASP.NET, PHP, and other forms of web sites.

Web design with CSS is in the core of this book, author does excellent job introducing readers to the methodology, and teaches to use deferent aspects: text, images, navigation, layout. The advice is not limited to CSS, on the contrarily, you will see plenty of tips on graphical image preparation for the web, various techniques, JavaScript code, and even some FormMail and PHP suggestions. The later two strictly not belong to the realm of the web design, but this is what makes this book different from many other books, - the out of the box paradigm.

This book was written by an actual artist who is not afraid to let his opinion known on many subjects discussed. You don't have to agree with author on everything, but when you are looking for advise it will be right there in your face, you won't have to dig for it or second-guess it.

This is an excellent well-structured book filled with useful examples. I highly recommend it."



Click here for more information...

Contents

Introduction

Back in 1989 when Tim Berners-Lee first introduced HTML, he designed it to visually communicate technical data in the form of simple lists and tables. As the potential for the Web was recognized, webpage designers wanted the layout capabilities of modern word processing tools. They wanted to create magazine style pages. Cascading Style Sheets (CSS) was developed for that purpose.

CSS provides the page design capability for the Web that print publishing has been enjoying for years. Style sheets give you

accurate control over page layout and positioning, advanced font control, and color control. With style sheets, you can specify text sizes and spacing between lines (leading) in points.

If you work with a word processor, You can alter the appearance of a document by changing the formatting and styles in a template. CSS works similar to word processor templates. A style sheet is a template that controls the formatting and appearance of a Web page.

CSS separates the typographics and page layout from the content of Web pages, making it much easier to revise your content or change your page design. With CSS, you can change the formatting of individual Web pages or your entire site without editing every single HTML tag in every single file.

CSS can reduce the clutter of tags on your HTML pages because you can apply many style attributes to a single html tag, or you can apply the same set of style attributes to a group of related html tags.

In this eBook, you'll learn the basic syntax of styles, the three ways to use styles and how style rules "cascade", and how to use grouping and classes to simplify style settings. You'll get hands-on examples for typographics, layout, and border and background control.

To use this material, you'll need a basic understanding of html. You should be familiar with the basic html tags that make a wepage, the tags for html paragraphs, spans, and divisions, the basic font attributes, and how to specify web color. If you are not familiar with html, don't worry, this ebook gives you the html. And, after learning CSS, you won't need to use much html.

Contents

Style Syntax

With CSS, you write styles rules. A style rule has three parts: *selector, property,* and *value*. The selector part defines the html element to which the rule is applied. The property part defines the attribute or characteristic, such as size, color, or border, to which the rule is applied. The value part defines the setting of the property.

The rule below sets the style for the <h1> tag to a 12 point font size.

```
h1 {font-size: 12pt}
```

Note in the style rule above, that the property and value parts of the rule are within brackets, and the property and value are separated by a colon.

You can apply multiple properties to a single html tag. The properties are separated by semicolons. The following style definition assigns the <h2> tag a font size of 10 points and a font weight of bold.

```
h2 {font-size: 10pt; font-weight: bold}
```

The rule below sets the left, top, right, and bottom margins for the <body> tag.

```
body {margin: 20px 20px 20px 20px}
```

Note in the rule above, the unit of measurement is *pixels*. Other units of measurement available are cm (centimeters) and in (inches).

The rules below set the font family for the <body> tag to arial with a 12 point font size, and the font family for the <h1> to arial with a 14 point font size.

```
body {font-family: arial; font-size: 12pt}
h1 {font-family: arial; font-size: 14pt; color: blue;}
```

Style sheets are referred to as "cascading" because you can specify style at three different levels; *inline, embedded*, and *linked*. You can specify style at one, two, or all three levels. Regardless of which level you use to specify style, the basic syntax is the same.

Contents

Inline Style

The biggest advantage of CSS is that you can define styles for your entire webpage, or even your entire website, in one place; for example, in the <head> section of your webpage or in an external Style Sheet. But what if you want to override the global style rule for a specific html tag? To do that, you can apply an "inline" style rule.

Specifying inline style is a great way to learn CSS. Let's get some hands-on experience. All the examples in this series of articles will start by opening Windows Notepad, or your favorite plain ASCII text editor, and typing the basic tags that make a web page, as shown below.

```
<html>
<head>
</head>
<body>

</body>
</html>
```

Enter a paragraph of text into the <body> section and save the file with the extension .htm. When you double-click on the name of the file, the text will appear in your Web browser.

First, let's apply some style to the <body> tag. Edit the <body> tag so that it appears as shown below and save the file.

<body style="font-family: verdana; margin:20px">

This time when you open the webpage, the text will display with the verdana font, and the webpage will have a margin of 20 pixels.

Edit the webpage again, typing a title just below the paragraph of text. Then type another two paragraphs of text below that. Place the title, and the two paragraphs inside tags, as shown below. Save the file, then open it in your browser to see how it looks.

Title

Edit the title's paragraph tag so that it appears as shown below.

Edit the paragraph tag of the first paragraph below the title as shown below.

Edit the paragraph tag of the second paragraph below the title as shown below.

This time when you open the webpage, the title text will display with the arial font in bold, the first paragraph below the title will be indented 30 pixels, similar to a blockquote, and the last paragraph will have a small 10 point courier font.

You can use inline style to set style rules for individual html elements on your webpage.

Contents

Embedded Style

The biggest advantage of CSS is that you can define styles for your entire webpage, or even your entire website, in one place. You can define the styles for your entire webpage by embedding a style block in the <head> section of your webapge.

Let's get some hands-on experience. All the examples in this series of articles we'll start by opening Windows Notepad or your favorite plain ASCII text editor and typing in the basic tags to make a web page as shown below.

<html>
<head>
</head>
<body>

```
</body>
</html>
```

Between the <body> tags, type some text. Type a paragraph of text. Type two headlines. Then type another paragraph of text. Add the appropriate html tags as shown below.

```
<body>
We the people of the United States, in order to form a
more perfect union, establish justice, insure domestic
tranquility, provide for the common defense, promote the
general welfare, and secure the blessings of liberty to
ourselves and our posterity, do ordain and establish this
Constitution for the United States of America.

<h1>Article 1</h1>

<h2>Part 1</h2>

<p>All legislative powers herein granted shall be
vested in a Congress of the United States, which shall
consist of a Senate and House of Representatives.</p>
</body>
```

Save the file with the extension .htm. Then double-click on the name of the file to view it as a webpage in your browser.

With the file open in Notepad, in the <head> section of the webpage add the tags for an embedded style block as shown below.

```
<style type="text/css">

</style>
```

Inside this style block, we will put style rules for the html elements that we placed on the webpage. The completed style block is shown below.

```
<style type="text/css">
body { font-family: helvetica; margin:20px }
h1 { font-family: arial; font-weight: bold }
h2 { font-family: courier; font-size: 20pt; }
p { margin-left: 30px; margin-right:30px; }
</style>
```

That's all there is to it. You have created your first embeded style block. Save the file, then open it in your Browser to see how it looks.

The styles defined in the style block apply to all html elements on the web page of the type defined by the rule's selector. To make sure this is true, add more h1 or h2 level headlines or more paragraphs to the webpage. The new elements will follow the style rules defined in the embedded style block.

Contents

External Style Sheet

The biggest advantage of CSS is that you can define styles for your entire webpage, or even your entire website, in one place. In part 4, we learned how to define the styles for an entire webpage by embedding a style block in the <head> section of the webapge. In this article, we learn how to define styles for your entire website by linking to an external style sheet.

Let's get some hands-on experience. Open Windows Notepad, or your favorite plain ASCII text editor, and type the style rules shown below.

```
body { font-family: helvetica; margin:20px }
h1 { font-family: arial; font-weight: bold }
h2 { font-family: courier; font-size: 20pt; }
p { margin-left: 30px; margin-right:30px; }
```

You will recognize that these are the exact same style rules we used in the embedded style block, except this time we left out the opening and closing style tags. We don't need the style tags because when we save this file with the .css extension, the browser will know it contains style code. Save the file with the name "mystyles.css".

Open a new file in Windows Notepad and type in the basic tags to make a web page as shown below.

```
<html>
<head>
</head>
<body>

</body>
</html>
```

Between the body tags, type some text. Type a paragraph of text. Type two headlines. Then type another paragraph of text. Add the appropriate html tags as shown below.

```
<body>
We the people of the United States, in order to form
a more perfect union, establish justice, insure
domestic tranquility, provide for the common defense,
promote the general welfare, and secure the blessings
of liberty to ourselves and our posterity, do ordain
and establish this Constitution for the United States
of America.

<h1>Article 1</h1>

<h2>Part 1</h2>

<p>All legislative powers herein granted shall
be vested in a Congress of the United States, which
shall consist of a Senate and House of Representatives.</p>
```

```
</body>
```

Save the file with the extension .htm. Then double-click on the name of the file to view it as a webpage in your browser.

With the file open in Notepad, in the <head> section of the webpage add the tag for a link to the style sheet as shown below.

```
<link rel="stylesheet" type="text/css" href="mystyles.css">
```

In the tag above, the "rel" attribute defines the relationship with the linked file; it's a style sheet. The "type" attribute indicates that the file contains text that is CSS code. The "href" atribute contains the URL, or path to the style sheet. If you save the webpage in the same folder as the style sheet, all that is needed here is the name of the file.

That's all there is to it. You have created your first linked style sheet. Save the file, then open it in your Browser to see how it looks.

Contents

Cascading Rules

In earlier pages, we learned how to specify style rules as an attribute in individual html tags. We learned how to embed a block of style rules in the head of our webpage, and we learned how to link to an external style sheet. What happens if we use several of these methods together and they have conflicting style information?

That's when the cascading rules come into play. Let's learn the cascading rules and get some hands-on experience at the same time. Open Windows Notepad, or your favorite plain ASCII text editor, and type in the basic tags to make a web page, as shown below.

```
<html>
<head>
</head>
<body>

</body>
</html>
```

In the body of the web page, type three headlines as shown below.

```
<h2>Headline Number One</h2>

<h3>Headline Number Two</h3>

<h4>Headline Number Three</h4>
```

Save the file with the extension .htm, and open another file in

Notepad. In the second file, type the style rule shown below.

```
h2 { font-family: arial; color: red; }
h3 { font-family: arial; color: red; }
h4 { font-family: arial; color: red; }
```

This file will be your external style sheet. In the same folder as the web page, save the file with the name mystyles.css.

Now, link the external style sheet to your webpage by typing, or pasting the following line in the <head> section of your webpage.

```
<link rel="stylesheet" type="text/css" href="mystyles.css">
```

When you open the web page, the three headlines will appear with the arial font in the color red, as defined by the external style sheet. Next, type or paste the style block shown below into the <head> section of your webpage.

```
<style type="text/css">
h3 { font-family: courier; color: green; }
h4 { font-family: courier; color: green; }
</style>
```

This time when you open the web page, the first headline will appear with the arial font in the color red, as defined by the external style sheet. However, the second and third headlines will appear with the courier font in the color green. That's because style definitons defined in an embedded style block override style definitions defined in an external style sheet.

Next, in the web page, edit the html for the third headline as shown below.

```
<h4 style="font-family:helvetica; color:blue;">
Headline Number Three</h4>
```

This time when you open the web page, the first headline will appear with the arial font in the color red, as defined by the external style sheet. The second headline will appear with the courier font in the color green as defined by the embedded style block. The third headline will appear with the helvetica font in the color blue.

Styles defined in an html tag override style definitions defined in an embedded style block and style definitions defined in an external style sheet.

Contents

Inheritance

In earlier pages of this ebook we learned that there are three different ways to apply style rules; inline, embedded, and linked. We

learned that we can use several of these methods together, and if they have conflicting style information, the cascading rules come into play.

Understanding the cascading rules helps predict the styles we should see on our webpage; however, there is another rule that complicates things a bit. Similar to a family tree, the elements on a web page follow a heirarchial structure of inheritance. And similar to a family tree, the child elements inherit style from their parents.

Let's get some hands-on experience to demonstrate how inheritance works. Open Windows Notepad, or your favorite plain ASCII text editor, and type in the basic tags to make a web page, as shown below.

```
<html>
<head>
</head>
<body>

</body>
</html>
```

Immediately we can see that the top level parent tag is <html>. The tags for the <head> and <body> sections are child elements of the <html> element. Type or paste the following paragraphs inside the <body> section.

```
<div>
<p>
We the people of the United States, in order to form
a more perfect union, establish justice, insure
domestic tranquility, provide for the common defense,
promote the general welfare, and secure the blessings
of liberty to ourselves and our posterity, do ordain
and establish this Constitution for the United States
of America.
</p>

<p>
The Constitution was framed by a convention of
delegates from twelve of the thirteen original states,
Rhode Island failing to send a delegate. The draft
was submitted to all thirteen states and became
effective when ratified by New Hampshire, the ninth
state to approve.
</p>
</div>
```

Then add style to the <body> tag as shown below.

```
<body style="font-family: arial; color: red;">
```

When you open the web page in your browser, all the text will appear in red arial font. The html (division) element inherited style from its parent element; the <body> element. Each of the <p>

(paragraph) elements inhertied style from their parent element; the .

Next, add style to the tag as shown below.

This time, when you open the web page in your browser, all the text will appear in green courier font. The inline style you applied to the overrides the style that it inherited from the <body>. Each of the <p> elements inherited the green courier font style from the .

Next, add style to one of the <p> tags as shown below.

<p style="font-family: helvetica; color: blue;">

This time, when you open the web page in your browser, the text in the paragraph where you added style to the <p> tag will be in blue helvetica font, while the other paragraph will inherit the green courier font style from its parent, the tag.

To continue the experiment, you might add another paragraph of text outside the element. The text in that paragraph will inherit the red arial font style from it's parent, the <body> tag.

You have learned that in order to use the full capability of CSS, you have to understand the rules of cascading and inheritance. Sometimes these rules make it difficult to understand why an element displays a particular style, especially if someone else designed the style. To help yourself and others, you can add comments to your style definitions. Place comments between the characters /* and */. Below is an example of a comment.

```
h1 {color: blue;}
/* overides green for h1 in style sheet */
```

Contents

Typographics : The Font Family

Typographics is the most important component of any webpage. You can use Typographics to communicate the structure of your webpage, the relative importance of information, and to guide the reader through the information. CSS allows you to control the following Typographic properties.

font-family
font-size
font-style
font-weight
line-height

text-decoration

The example below specifies that text within paragraphs should use the verdana font.

```
p {font-family:verdana;}
```

The user must have the specified font installed on their computer, otherwise the browser will use the closest matching font on the user's computer. What the browser chooses as the closest matching font may not be anything close to what you had in mind. To avoid unexpected results, you should specify only the commonly installed fonts listed below.

arial
courier
helvetica
times
verdana

To minimize unexpected results even further, you can specify a list of alternate fonts as shown below.

```
p {font-family:verdana,arial,helvetica;}
```

If the first font in the list is not available on the user's computer, the browser will use the second font in the list. If the second font in the list is not available, the browser will use the third font in the list.

You might get more reliable results by specifying a generic font family, as shown below.

```
p {font-family:serif;}
```

Serif fonts have little details at the end of the letters strokes. This is a traditional font designed for early ink print processes to make sure ink was applied at the ends of the letters strokes. Serif fonts are tiresome to read on a computer screen. You can specify the sans-serif font family to make the browser use a font without serifs.

If you want to get the effect of typewriter text, for example to represent code, you can specify the monospace font family. With the monospace font, every letter has the same width.

Contents

Typographics : Specifying the Font Size, Weight, and Style

Typographics is the most important component of any webpage. You can use Typographics to communicate the structure of your webpage, to communicate the relative importance of information, and to guide the reader through the information. In Part 8, we learned how to specify the font family. With CSS, we can control many other font

characteristics, including font size, weight, and style.

Specifying the Font Size

The example below specifies that the font size for the paragraph should be 12 points.

```
p {font-family: verdana; font-size: 12pt;}
```

You can specify the font size in points (pt), inches (in), centimeters (cm) , or pixels (px). The first three units are fixed dimensions, useful if the format of the page is important when it's printed. Image sizes and most other webpage structures are specified in pixels. Because the users screen resolution can vary, it's best not to mix fixed dimensions with pixel dimensions.

Specifying the Font Weight

The example below specifies that the font for the paragraph should be bold.

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

In some cases, after setting the font-weight to bold, you might need to set it back to normal, for example to override inheritance. To do that, you can apply the "font-weight:normal" rule to an html span, as shown below.

```
<p style="font-weight: bold;">This text is bold.
<span style="font-weight: normal;"> This text is
normal</span> This text inherits the bold weight.</p>
```

You can also specify a specific font weight or "thickness" value as a number from 100 to 900 as shown below.

```
p {font-weight: 700;}
```

This gives you pretty good control of the boldness of your fonts.

Specifying the Font Style

The example below specifies that the font for the paragraph should be italic.

```
p {font-style: italic;}
```

In some cases, after setting the font-style to italic, you might need to set it back to normal, for example to override inheritance. To do that, you can apply the "font-weight:normal" rule to an html span, as shown below.

```
<p style="font-style: italic;">This text is italic.
<span style="font-style: normal;"> This text is
normal</span> This text inherits the italic style.</p>
```

CSS gives you control over font characteristics that are similar to those provided by html. With html, you must specify font characteristics as attributes in every font tag. The advantage of CSS is that you can specify fonts for an entire website in a single style sheet.

Contents

Typographics : Specifying the Line Height and Text Alignment

Line Height

In traditional paper and ink printing, the Typographer was able to separate lines of text by putting lead spacers of various thickness between them, this was called "leading". CSS lets you set line spacing with the line-height property. The example below specifies that lines of text in the paragraph should be separated by a space 50 percent of the height of the text.

```
p {line-height: 50%;}
```

You can also specify line height in points (pt), inches (in), centimeters (cm), or pixels (px). The example below specifies a line height of 12 pixels.

```
p {line-height: 12px;}
```

Text Alignment

In a word processor application, you can align text to the left or right margin. CSS allows you to do the same thing. Let's get some hands-on experience. Start by opening Windows Notepad, or your favorite plain ASCII text editor, and typing the basic tags that make a web page, as shown below.

```
<html>
<head>
</head>
<body>

</body>
</html>
```

Enter or paste the paragraph of text shown below into the <body> section of the webpage and save the file with the extension .htm. When you double-click on the name of the file, the text will appear in your Web browser.

```
<p style="text-align:left;">I stand before you under
indictment for the alleged crime of having voted at the last
presidential election, without having a lawful right to vote.
It shall be my work this evening to prove to you that in
thus doing, I not only committed no crime, but instead simply
```

exercised my citizens rights guaranteed to me and all United States citizens by the National Constitution
... Susan B. Anthony 1873</p>

The text-align style rule can have one of four different values; left, center, right, or justify. Try each value by editing the inline style rule in the paragraph tag and re-opening the webpage in your browser.

Contents

Units of Measurement

So far in this eBook, we have been creating style rules using various units of measurement without a detailed understanding of CSS units of measurement. CSS provides a vaiety of measurement units including "absolute", "relative", and "percentages".

Absolute units, like "in" (inch) and "cm" (centimeter), should be used when you're concerned about the appearance of the webapage when it's printed. Relative units like "em" (the width of a capital M) and "px" (pixel) allow you to build webpages that scale to the user's configured font size and display size. Percentages allow you to scale elements relative to other elements on the webpage.

Absolute Units

cm Centimeter
mm Millimeter
in Inch
pt Point
pc Pica

The centimeter is a metric unit. There are approximately 2 1/2 centemeters in an inch. There are exactly 10 millimeters in a centimeter. The inch is an English unit. The point and pica are units left over from traditional printing where movable type was used. There are 72 points in an inch, 6 pica in an inch.

Unless you are primarily concerned with the printing of the web page, you should avoid using absolute units for webpages because they cannot be scaled to the users screen size. What might appear as an inch on your computer screen will appear as two inches on the user's screen if it is twice as large.

Relative Units

em The width of a capital M
ex The height of a small x
px Pixel

The em unit is useful for setting font sizes and margin sizes so they they maintain the same relative appearance regardless of what default font size the user configures. Below is an example of a

headline and a paragraph defined with the em unit.

```
<h3 style"font-size: 2em;">The Constitution for the
United States of America</h3>

<p style="margin: 1em">We the people of the United
States, in order to form a more perfect union, establish
justice, insure domestic tranquility, provide for the
common defense, promote the general welfare, and secure
the blessings of liberty to ourselves and our posterity,
do ordain and establish this Constitution for the United
States of America.</p>
```

With the style properties defined above, the title font size will always be two times the default font size, and the paragraph margin will always the width of a capital M in the default font size.

If you want to control the appearance of your webpage relative to images on the page, you should use the px (pixel) unit. An image consists of a two-dimensional array of colored dots on the computer screen. For that reason, the dimensions of images are always in pixel units. If the size of your fonts and other elements on the webpage are not also in pixel units, the relative size, and possibly location, of elements on the webpage will vary depending upon the size and resolution of the users screen.

Contents

Specifying Color

So far in this eBook, we have been creating style rules to control color without having a detailed understanding of how to specify color. The CSS "color" property specifies the foreground color of an element on a webpage. This property can be used to set the color for text and/or the border of an html element.

The value for the color property can be specified by one of four different methods. The first method is to use the color name as shown below.

```
h1 {color: red;}
```

The problem with this method is, unless your using a basic color, it's difficult to remember the color names defined in the CSS specification.

Colors are created on a computer by mixing varying amounts of the basic colors red, green, and blue, referred to as "rgb" colors. You can specify the amount of each basic color by percentage. The example below defines the color red by specifying 100% for the red component, and 0% for the green and blue components.

```
h1 {color: rgb(100%,0%,0%);}
```

The computer has to convert these percentages to 16 bit binary values. The highest 16 bit binary value is 255; therefore, 50% would be converted to 127. Your webpage will load slightly faster if you enter color values from 0 to 255 rather than as percentages, as shown blow.

```
h1 {color: rgb(255,0,0);}
```

The computer still has to convert these decimal values to binary numbers. Binary numbers can be represented in hexadecimal notation. Whereas a decimal digit can be one of ten different values from 0 to 9, a hexadecimal digit can be one of 16 different values from 0 to f (after 9 - a, b, c, d, e and f are used).

The example below shows how a pound sign (#) is used to indicate that the color is being specified in hexadecimal notation.

```
h1 {color: #ff0000;}
```

Once the color value is specified, it can be used to set the color of an element on a webpage. The example below sets the color property to green for both the text in the paragraph, and the border around the paragraph.

```
<p style="color: green; border: solid;">
We the people of the United States, in order to form
a more perfect union, establish justice, insure
domestic tranquility, provide for the common defense,
promote the general welfare, and secure the blessings
of liberty to ourselves and our posterity, do ordain
and establish this Constitution for the United States
of America.</p>
```

To set a different color for the text and the border, you could place the paragraph inside a div and set a different color property for the paragraph and div, as shown below.

```
<div style="color: blue; border: solid;">
<p style="color: green;">
We the people of the United States, in order to form
a more perfect union, establish justice, insure
domestic tranquility, provide for the common defense,
promote the general welfare, and secure the blessings
of liberty to ourselves and our posterity, do ordain
and establish this Constitution for the United States
of America.</p></div>
```

The "background-color" property specifies the background color of an element on a webpage. The example below sets the background-color property for the paragraph to yellow.

```
<p style="background-color: yellow;">
We the people of the United States, in order to form
a more perfect union, establish justice, insure
```

domestic tranquility, provide for the common defense,
promote the general welfare, and secure the blessings
of liberty to ourselves and our posterity, do ordain
and establish this Constitution for the United States
of America.

Html elements like paragraphs , divisions , and spans define rectangular areas, or boxes, on a webpage.

Contents

The Box Model : Border

In this eBook, we have been applying style rules to many diferent kinds of html elements. Almost every html element is contained within a rectangular area on the webpage. A paragraph of text is contained within a box. An image is contained within a box. Even a list is contained within a box. In addition, html defines some specific box elements like the and the .

Let's get some handson experience with CSS boxes. Open Windows Notepad and enter the basic tags that make a web page as shown below.

```
<html>
<head>
</head>
<body>

</body>
</html>
```

Enter the code for an html span into the <body> section as shown below.

```
<span>This is my text</span>
```

Save the file with the extension .htm. When you double-click on the name of the file, the text "This is my text" will appear in your Web browser.

At this point, it may be difficult to believe the span is a box. Apply a style rule to the tag as shown below, then save the file and re-open it in your Web browser.

```
<span style="border-style: solid;">This is my text</span>
```

Now that the span has a border, it is easy to see that it is a box. The default border-style is none: therefore, if you don't specify a border-style, you don't get a border. Below are the possible border-styles you can specify.

none

dotted
dashed
solid
double
grove
ridge
inset
outset

One thing you may have noticed is that there is no space between the text and the border. To fix that problem, you can define some padding, as shown below.

```
<span style="border-style: solid; padding: 5px;">
This is my text</span>
```

Similarly, if you want some space outside the border between the span and other elements on the web page, you can define a margin.

These examples use in-line style. In previous parts of this article, we used embedded style. To use embedded style, remove the style rule from the tag and instead place a block of style code in the <head> section of the webpage as shown below.

```
<html>
<head>

<style type="text/css">
span
{
border-style: solid;
padding: 5px;
margin: 10px;
}
</style>

</head>
<body>

<span>This is my text</span>

</body>
</html>
```

When you open the file in your Web browser, you'll see the page display exactly as it did before. Next, add rules to define the border width and border color of the span as shown below.

```
span
{
border-style: solid;
padding: 5px;
margin: 10px;
border-width: 2px;
border-color: blue;
}
```

Of course, the border-color can also be specified using the rgb function or hexadecimal notation as described in the last article.

Go ahead and experiment with different values for border-style, border-width, border-color, and so on.

Contents

The Box Model : Background

CSS provides the following properties that allows you to define the background of a box.

background-attachment: fixed or scroll
background-color: color name, rgb function or hexadecimal notation
background-position: x and y postion, pixels or %
background-image: url("imagename.jpg");
background-repeat: repeat, no-repeat, repeat-x, or repeat-y

Let's get some hands-on experience with CSS boxes. Open Windows Notepad and enter the basic tags that make a web page as shown below.

```
<html>
<head>
</head>
<body>

</body>
</html>
```

Enter the code for an html span into the <body> section as shown below.

```
<span>This is my text</span>
```

Save the file with the extension .htm. When you double click on the name of the file, the text "This is my text" will appear in your Web browser. Next, apply a style rule to the tag as shown below. Save the file and open in your Web browser.

```
<span style="background-color:lightblue;">
This is my text</span>
```

Now the span appears with a light blue background. Of course, the background color can also be specified using the rgb function or hexadecimal notation as described in a previous article.

Next, locate an image that you would like to use for the backgound of the span. Edit the style rule of the span as shown below to display the image as the background of the span.

```
<span style="background-image:url('background.jpg');">
```

This is my text

You could apply the same style rule to the webpage's <body> tag. When you open the webpage in your browser, you will find that the image has been tiled to provide a backgound for the webpage. You can prevent the tiling effect by using the "background-repeat:no-repeat;" rule, as shown below.

<body style="background-image:url('background.jpg');
background-repeat:no-repeat;">

You can create a graphic margin for your webpage by specifying that the background image should repeat only in the "y" direction, as shown below.

<body style="background-image:url('background.jpg');
background-repeat:repeat-y">

Similarly, you can create a graphic header for your webpage by specifying that the backgound image should repeat in the "x" direction. In this case, you might also apply the "background-attachment:fixed;" rule. This will cause the background image to remain stationary while the webpage text and other elements scroll.

<body style="background-image:url('background.jpg');
background-repeat:repeat-x;
background-attachment:fixed;">

Note: To view this effect, you'll need to add lines of text and/or adjust the size of your webpage until a vertical scroll bar appears.

You can precisely position the backgound image using the "background-position: x y;" rule. The example below places image of the background upper-left corner 20 pixels from the right side and 50 pixels from the top side of the browser window (or whatever element you use the rule with).

<body style="background-image:url('background.jpg');
background-position: 20px 50px;">

Of course, you can specify the position in any valid CSS units, including percent. When applied to the <body> tag, specifying the position as a percentage will cause the position of the graphic to change as the browser window is resized.

Although this method can be used to specify the position of a background graphic within an html element, CSS provides much more powerful postioning rules.

Contents

The Box Model : Positioning

Let's get some hands-on experience with CSS positioning. Open Windows Notepad and enter the basic tags that make a web page as shown below.

```
<html>
<head>

</head>
<body>

</body>
</html>
```

Enter the code for an html span into the <body> section as shown below.

```
<span>This is my text</span>
```

Save the file with the extension .htm. When you double click on the name of the file, the text "This is my text" will appear in your Web browser.

Next, type or paste the style block shown below into the <head> seaction of your webpage.

```
<style type="text/css">
span
{
border-style: solid;
}
</style>
```

Save the file and open in your Web browser. The style rule specifies that the span should have a solid border. Note that the span is positioned in the upper-left corner of the webpage. To the style block, add a rule as shown below to position the span 100 pixels from the left side of the webpage. Save the file and open in your Web browser.

```
span
{
border-style: solid;
position:relative; left:100px; top:0px;
}
```

Next, type some text into the <body> section of the webpage just before the span.

```
Just type in any old text before the span
<span>This is my text</span>
```

Now, when you open the webpage in your browser, the span will appear 100 pixels from the left of the text, rather than left border the webpage. This is because we defined the position as "relative". The browser positions the span relative to other objects on the webpage. In the style block, change the word "relative" to "absolute" and open the webpage again. Now the span will appear 100 pixels from the left

side of the webpage.

Next, remove the text that you typed before the span and add a second span just below the first span. Give the first span the id "first" and the second span the id "second" as shown below.

```
<span id="first">This is the first span</span>

<span id="second">This is the second span</span>
```

Edit the style block as shown below. Note that the style selectors have been changed to the id's of the spans, prefixed with a pound (#) sign. The second span has an absolute position 100 pixels from the left border and 10 pixels from the top of the browser border.

```
#first
{
border-style: solid;
background-color:yellow;
position:absolute; left:100; top:0;
}
#second
{
border-style: solid;
border-color:blue;
background-color:lightblue;
position:absolute; left:110; top:10;
}
```

A style rule for a blue border color has been added to the second span. A style rule for a background color of yellow has been added to the first span. A style rule for a background color of lightblue has been added to the first span.

The border and background color rules have been added so you can tell which span is which. When you open the webpage in your browser, the second span will partially cover the first span. The layer position of a webpage element is called it's z-index. Webpage elements that overlap have default z-indexes in the order in which the code appears.

Add a z-index rule to the styles for both spans. Give the first span a z-index of 2, and the second span a z-index of 1, as shown below.

```
#first
{
border-style: solid;
background-color:yellow;
position:absolute; left:100; top:0;
z-index:2;
}
#second
{
border-style: solid;
border-color:blue;
background-color:lightblue;
```

```
position:absolute; left:110; top:10;
z-index:1;
}
```

Now when you open the webpage in your browser, the first span will be on top.

You can control the sizes of spans (or other elements) by specifying width and height with the position as shown below.

```
position:absolute; left:100; top:0;
width:200px; height:100px;
```

With this knowledge, you could design your own CSS webpage template.

Contents

Simplification Through Grouping

If you want to assign the same style to different types of tags, you could write the style rules as shown below.

```
h1
{
font-size: 12pt
font-weight: bold;
color: blue;
}
h2
{
font-size: 12pt;
font-weight: bold;
color: blue;
}
h3
{
font-size: 12pt;
font-weight: bold;
color: blue;
}
```

Another way to do the same thing is to group the selectors as shown below.

```
h1, h2, h3
{
font-size: 12pt;
font-weight: bold;
color: blue;
}
```

If you want to assign several style rules to one (or several as above)

types of tags, you could group the formatting specifications as shown below.

```
h1
{
font-size: 15pt;
line-height: 17pt;
font-weight: bold;
font-family: "Arial"
font-style: normal
}
```

Another way to do the same thing is to group the style values as shown below.

```
h1
{
font: 15pt/17pt bold "Arial" normal
}
```

Note: when you group style values as shown above, the order is important. The font size comes first (with the optional line height after a forward slash), followed by the font weight, font name, and font style.

Contents

Simplification With Descendant Selectors

Something similar to grouping is "descendant selectors". A descendant selector uses the hierarchial parent - child relationship of html elements on a web page to specify selectors.

Let's get some hands-on experience to demonstrate how descendant selectors work. Open Windows Notepad, or your favorite plain ASCII text editor, and type in the basic tags to make a web page, as shown below.

```
<html>
<head>
</head>
<body>

</body>
</html>
```

In the <body> type or paste the paragraph of text shown below.

```
<p>Four score and seven years ago our fathers brought forth
on this continent, <b> a new nation conceived in Liberty
</b>, and dedicated to the proposition that all men are
created equal.</p>
```

Note that within this paragraph of text we used the and tags to make the words "a new nation conceived in Liberty" display in bold.

Just below the paragraph of text, type or paste the text shown below.

Abraham Lincoln's Gettysburg Address

Type or paste the style block shown below into the <head> section of the webpage.

```
<style type="text/css">
p b {color: blue;}
</style>
```

Note that the selectors in this style rule are not separated by a comma. This indicates that it is not just a grouping of selectors. It would be interpreted like this; apply the color blue to all b (bold tags) that are within p (paragraph tags). So the rule would not apply to the last line of text that you added because, even though it's within tags, the tags are not descendants of <p> tags.

When you open the webpage in your browser, you will see that the bold text within the paragraph is blue, while the bold text outside the paragraph is the default color, black.

Contents

Style Classes

Let's assume that you want to use embedded style to define style rules for your entire webpage in one style block, or you want to link to an external style sheet to provide style rules for entire website. You also want some of your h3 level headlines to be 10 point arial font in black, some to be 12 point verdana font in red, and some to be 12 point sans-serif font in blue. Are you forced to use inline style to apply style rules to all the h3 tags directly?

No, you can use style "classes" to define different style rules for a single type of html tag. In the example above, you could define three classes as shown below.

```
h3.black {font-family: arial; font-size: 10pt; color: black}
h3.red {font-family: verdana; font-size: 12pt; color: red}
h3.blue {font-family: sans-serif; font-size: 12pt; color: blue}
```

Note the dot after the h3 selector. The dot indicates that this is a class name. You could then use the a class name in each h3 tag to specify which set of style rules apply to that particular h3 heading as shown below.

<h3 class="black">This Heading Will Be Black</h3>

```
<h3 class="red">This Heading Will Be Red</h3>

<h3 class="blue">This Heading Will Be Blue</h3>
```

You can use a style class to define style rules to diffent types of html tags so that they all have the same style. Let's assume we want all quotes to be indented eight pixels and use the arial font, italic, and in blue. We could define a style class as shown below.

```
.quote
{
margin-left: 8px;
font-family: arial;
font-style: italic;
color: blue;
}
```

You could then use the quote class to apply style to a paragraph tag as shown below.

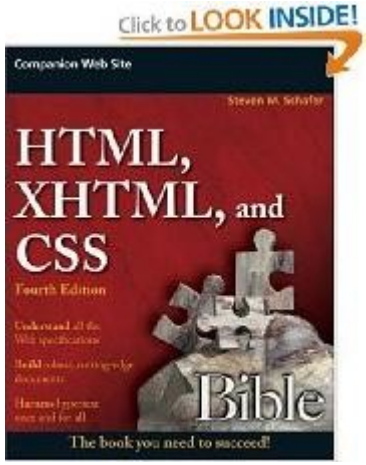
```
<p class="quote">We the people of the United States,
in order to form a more perfect union, establish justice,
insure domestic tranquility, provide for the common defense,
promote the general welfare, and secure the blessings of
liberty to ourselves and our posterity, do ordain and
establish this Constitution for the United States of
America.</p>
```

You can use the quote class to apply style to a division tag as shown below.

```
<div class="quote">Congress shall make no law
respecting an establishment of religion, or prohibiting an
establishment of religion, or prohibiting the free exercise
thereof; or abridging the freedom of speech, or of the
press; or the right of the people peaceably to assemble,
and to petition the Government for a redress of grievances.
</div>
```

You can use the quote class to apply style to any html element that has any of the properties of margin, font, or color that rules are defined for in the class.

Contents



Decipher the code, use the right tools, and conquer the online world of the World Wide Web. This comprehensive guide demystifies HyperText Markup Language (HTML) and Cascading Style Sheets (CSS) so you can create sophisticated and interactive Web pages, robust applications, and as many other ways of interacting on the Web as you can think of. You'll even learn to code cool content for many mobile devices that include a browser. Inside, find all the tools, tips, and techniques you need to succeed.

• Explore the underlying structure of all Web pages
• Learn the basics of text structure, meta tags, links, and more
• Write scripts, master dynamic HTML, and use CSS editing tools
• Create Web pages for mobile devices with XHTML Basic
• Harness new Web 2.0 features with microformats
• Add colors, backgrounds, multimedia, and interactivity
• Clean up, test, and validate your code

Reader Lori Smart of Eugene, OR, says,"We've been in the web development business for over 11 years and have watched so much change in that time. We're often asked what tools are vital to learning how to work in our field. Many from the 'bible' series books have been on our office shelves over the years. This one is the latest to actually take up residence on each of our desks as a regular reference tool. Buy it, use it, use those nifty post-it flags and highlighters all over it, and when the pages start falling out, tape them back in and keep using it. ... It takes an entire library of reference books to stay up with all the changes and aspects of this industry, and as you grow beyond the levels found here add to your collection, but this one really is important to your library!"

Partial Contents:

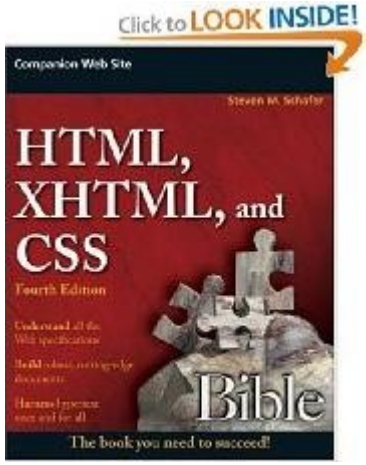
• CSS Tips and Tricks
• Formatting HTML Tables
• Dynamic HTML with CSS
• Pseudo Elements
• Generated Content
• CSS Floating and Positioning
• CSS Inheritance
• CSS Cascade
• HTML Tips and Tricks
• Creating Mobile Web Pages

- Using XML
- Using Scripts
- Internationalization
- Using Multimedia
- Using Forms
- Using Frames
- Character Formatting
- Text Structuring
- Special Characters
- Plus Much More...

Reader David J. Lauridsen Jr says,"I have used this book as a textbook for an 'Introduction to HTML' class I taught. I looked at several references prior to choosing on one, and this was by far the best formatted and most appropriate for those with little to no existing knowledge of HTML.

"The appendix contains a more or less comprehensive listing of all HTML tags and their usage, etc. The chapters are well organized, easy to read, and comprehensive. If this book spreads itself a little thin at times trying to cover so much ground, it is necessary due to the inherently connected nature of HTML, XHTML, and CSS. Covering only HTML would not be useful for beginners who want to gain a basic understanding of these technologies.

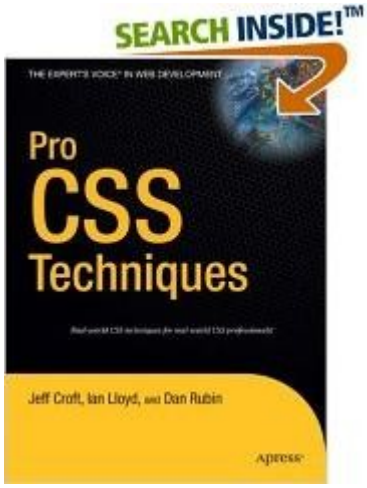
"I highly recomend this book to anyone wishing to learn HTML."



Click here to learn more...

Contents

Pro CSS Techniques



Review by Frank Stepanski (Philadelphia):

There are many CSS books on the market now, so distinguishing yourself from the many is getting harder and harder to do. Since there are less intermediate to advanced books compared to beginner CSS books this is a start. Another thing this book focuses on that is different than the majority of other CSS book is that is stresses creating semantic markup throughout the design and development process. Semantic markup means understanding the meaning of the XHTML that you write. What this means is that the code is free of presentational information using only and when absolutely necessary.

These two tags can be very useful in creating intricate CSS designs but they have no semantic meaning. Many blog posts have called this "divitis" because their web page is just a whole bunch of tags with identifiers (id or class) and that's it. While this visually helps the designer quickly create the site, to screen-readers, or PDAs or cell phones may not render the page exactly as you may think it should and using semantic markup helps the browser for that particular device render that page that makes the most sense. It is a hard concept to grasp or fully explain (as I probably am not), but it is used more and more in current web practices.

Now this book is written by a couple different authors which seems like the norm nowadays. [...] They all have contributed to various blogs (including their own) about web design techniques and many of them are here in this book. ... This is a great book for anybody wanting to further their CSS knowledge and experience by learning techniques that are used by some of the top designers out there. A must buy!

Review by Nathan Smith (SonSpring.com):

"This book is a collection of proven, professional, modern techniques that you can use every day to get the most out of the time you put into your projects... This book is not an introduction to CSS. Although we'll provide an overview of the basics, we'll assume you have a simple understanding of CSS and how it works."

Because the devil is in the details when it comes to CSS, this is

exactly the type of book that is needed. CSS is like chess, simple in principle yet complex in application. It's like the old adage: "A day to learn, a lifetime to master." I've never met a web developer who has had trouble mastering the concepts behind CSS. Agony is caused by multi-browser implementation of advanced layouts.

Don't get me wrong, I think introductory books are necessary, and in fact one of my favorite ones is Eric Meyer's Definitive Guide to CSS. Not every book needs to be the Encylopedia Britannica of programming languages. Pro CSS Techniques is more in line with CSS Mastery in the approach that it takes. ...

This is a great book, and would make an excellent addition to the arsenal of any client-side developer. The appendices are worth the price alone, containing CSS Reference, Specificity and Browser Grading charts. These provide a great way to see side-by-side comparisons of which CSS techniques are well supported, and those we can salivate over until they hit the mainstream. Bottom line, if CSS puts bread on your table, this book will make you more productive. Learn More ...

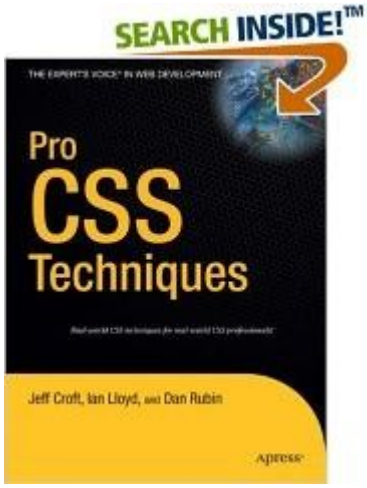
Review by Nate Klaiber (nateklaiber.com):

Pro CSS Techniques by Jeff Croft, Dan Rubin, and Ian Lloyd is a very thorough book on practical CSS. Having just read Simon Collison's Beginning CSS Web Development I found this to be a great continuation of the learning process. This book picks up where Simon's book ended, and even briefly covers some of the same topics. This book is full of great information and author's each had a fun personality (and sense of humor) with their chapters.

The book jumps right in and walks you through specificity and the cascade and how this will help you keep your markup neat and tidy (without any superfluous markup). This topic can cause confusion for many beginning CSS and even those who are advanced. Having a strong understanding of the cascade and specificity will greatly help you write cleaner code and solve any debugging issues that may arise. This chapter was full of examples, charts, and interactive walkthroughs to help you understand the process. ...

Finally, the book wraps up with an exhaustive list of CSS References including: allowed values, element type, and initial/inherited values. Also covered is the CSS specificity chart and the browser grading chart. The appendix of this book makes a nice desk reference to help solve an issue.

Overall I really enjoyed this book. If you are still dipping your feet into CSS, then this book will help you understand the inner workings of CSS and help you avoid many frustrating hours of debugging. Yes, there were some advanced topics that weren't discussed that could have been (hasLayout), but overall the book covered everything exhaustively. Do yourself a favor and pick up a copy of this book. Learn More ...



Contents
