HTML&CSS

design and build websites

JON DUCKETT

HTML & CSS

Design and Build Websites

JON DUCKETT

JOHN WILEY & SONS, INC.

HTML & CSS
DESIGN AND BUILD WEBSITES

Published by
John Wiley & Sons, Inc.
10475 Crosspoint Boulevard
Indianapolis, IN 46256
www.wiley.com

ISBN: 978-1-118-00818-8
Manufactured in the United States of America
Published simultaneously in Canada
10 9 8 7 6 5 4 3 2 1

Library of Congress Control Number: 2011932082

CREDITS

For John Wiley & Sons, Inc.

EXECUTIVE EDITOR
Carol Long

MARKETING MANAGER
Ashley Zurcher

PRODUCTION MANAGER
Tim Tate

PRODUCTION EDITOR
Daniel Scribner

VICE PRESIDENT AND
EXECUTIVE GROUP PUBLISHER
Richard Swadley

VICE PRESIDENT AND
EXECUTIVE PUBLISHER
Barry Pruett

ASSOCIATE PUBLISHER
Jim Minatel

PRODUCTION COORDINATOR,
COVER
Katie Crocker

AUTHOR
Jon Duckett

COVER DESIGNER
Emme Stone

DESIGN AND LAYOUT
Jon Duckett
Emme Stone

TECHNICAL EDITOR
Chris Mills

TECHNICAL REVIEWERS
Andy Stone
Angela Shimell
Donna Watson
Martin Callanan
Rob Jacoby
Tony Berry

PHOTOGRAPHY
John Stewardson
johnstewardson.com

ADDITIONAL PHOTOGRAPHY
Hesperian
Joe Robertson
flickr.com/photos/mindfire
Jules Clancy
thestonesoup.com
Kylie Gusset
gusset.net
Michael Stillwell
beebo.org

Try out and download all of the code for this book online at:
http://www.htmlandcssbook.com/code/

CONTENTS

INTRODUCTION

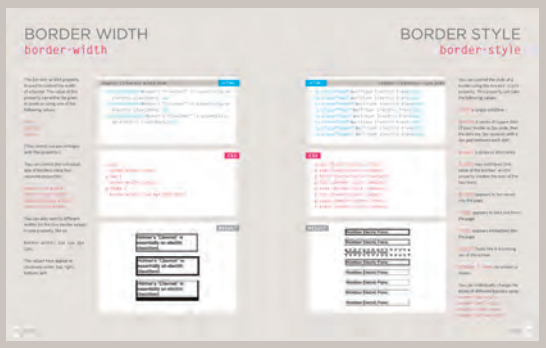
- About this book
- How the web works
- Learning from other pages

Firstly, thank you for picking up this book. It has been written with two very different types of people in mind:

- Those who want to learn how to design and build websites from scratch
- Anyone who has a website (that may be built using a content management system, blogging software, or an e-commerce platform) and wants more control over the appearance of their pages

The only things you need in order to use this book are a computer with a web browser and a text editor (such as Notepad, which comes with Windows, or TextEdit, which comes with Macs).

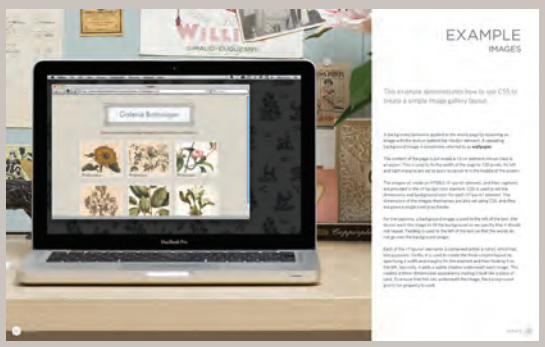
Introduction pages come at the beginning of each chapter. They introduce the key topics you will learn about.



Reference pages introduce key pieces of HTML & CSS code. The HTML code is shown in blue and CSS code is shown in pink.

Background pages appear on white; they explain the context of the topics covered that are discussed in each chapter.

Diagram and infographics pages are shown on a dark background. They provide a simple, visual reference to topics discussed.



Example pages put together the topics you have learned and demonstrate how they can be applied in each.

Summary pages come at the end of each chapter. They remind you of the key topics that were covered in each chapter.

IS IT HARD TO LEARN?

Many books that teach HTML and CSS resemble dull manuals. To make it easier for you to learn, we threw away the traditional template used by publishers and redesigned this book from scratch.

At work, when people look at my screen and see it full of code, it's not unusual to get a comment about it looking very complicated or how clever I must be to understand it. The truth is, it's not that hard to learn how to write web pages and read the code used to create them; you certainly don't have to be a "programmer."

Understanding HTML and CSS can help anyone who works with the web; designers can create more attractive and usable sites, website editors can create better content, marketers can communicate with their audience more effectively, and managers can commission better sites and get the best out of their teams.

I've focussed on the code you need to use 90% of the time and omitted the code that you would rarely see even if writing websites is your full time job. By the end of the book, if you come across the other 10% you will be able to Google it to find out what it means quickly and easily.

I have also added practical information on topics I am commonly asked about, such as how to prepare images, audio and video for the web, how to approach the design and build of a new site, how to improve your rankings in search engines (SEO), and how to use Google Analytics to learn about visitors to your site.

THE STRUCTURE OF THIS BOOK

In order to teach you about creating web pages, this book is divided into three sections:

1: HTML

We will spend the first chapter looking at how HTML is used to create web pages. You will see that you start by writing down the words you want to appear on your page. You then add tags or elements to the words so that the browser knows what is a heading, where a paragraph begins and ends, and so on.

The rest of this section introduces the tags you have at your disposal to create web pages, grouped into chapters on: text, lists, links, images, tables, forms, video audio and flash, and miscellaneous elements.

I should warn you that the examples in the first nine chapters are not exciting to look at, yet they are the foundation of every web page. The following chapters on CSS will show you how to make your pages look a lot more interesting.

2: CSS

We start this section with a chapter that explains how CSS uses rules to enable you to control the styling and layout of web pages. We then go on to look at the wide variety of CSS properties you can use in your CSS rules. These properties generally fall into one of two categories:

Presentation: How to control things like the color of text, the fonts you want to use and the size of those fonts, how to add background colors to pages (or parts of a page), and how to add background images.

Layout: How to control where the different elements are positioned on the screen. You will also learn several techniques that professionals use to make their pages more attractive.

3: PRACTICAL

We end up with some helpful information that will assist you in building better websites.

We look at some new tags that will be introduced in HTML5 to help describe the structure of your pages. HTML5 is the latest version of HTML (still under development at the time of writing). Before learning about these elements, you need a good grasp of how CSS is used to control the design of web pages. There is a chapter that talks you through a design process that you might like to follow when creating a new website.

Finally, we end up looking at topics that will help you once you have built your site, such as putting it on the web, search engine optimisation (SEO) and using analytics software to track who comes to your site and what they are looking at.

HOW PEOPLE ACCESS THE WEB

Before we look at the code used to build websites it is important to consider the different ways in which people access the web and clarify some terminology.

BROWSERS

People access websites using software called a **web browser**. Popular examples include Firefox, Internet Explorer, Safari, Chrome, and Opera.

In order to view a web page, users might type a web address into their browser, follow a link from another site, or use a bookmark.

Software manufacturers regularly release new versions of browsers with new features and supporting new additions to languages. It is important, however, to remember that many computer owners will not be running the latest versions of these browsers. Therefore you cannot rely on all visitors to your site being able to use the latest functionality offered in all browsers.

You will learn how to tell which browsers visitors use to access your website in Chapter 19.

WEB SERVERS

When you ask your browser for a web page, the request is sent across the Internet to a special computer known as a **web server** which hosts the website.

Web servers are special computers that are constantly connected to the Internet, and are optimized to send web pages out to people who request them.

Some big companies run their own web servers, but it is more common to use the services of a **web hosting** company who charge a fee to host your site.

DEVICES

People are accessing websites on an increasing range of devices including desktop computers, laptops, tablets, and mobile phones. It is important to remember that various devices have different screen sizes and some have faster connections to the web than others.

SCREEN READERS

Screen readers are programs that read out the contents of a computer screen to a user. They are commonly used by people with visual impairments.

In the same way that many countries have legislations that require public buildings to be accessible to those with disabilities, many laws have also been passed that require websites be accessible to those with disabilities.

Throughout this book you will see several references to screen readers. These notes will help ensure that the sites you create are accessible to people who use such software.

It is interesting to note that technologies similar to those employed by screen readers are also being used in other areas where people are unable read a screen, such as when they are driving or jogging.

HOW WEBSITES ARE CREATED

All websites use HTML and CSS, but content management systems, blogging software, and e-commerce platforms often add a few more technologies into the mix.

WHAT YOU SEE

When you are looking at a website, it is most likely that your browser will be receiving HTML and CSS from the web server that hosts the site. The web browser interprets the HTML and CSS code to create the page that you see.

Most web pages also include extra content such as images, audio, video, or animations and this book will teach you how to prepare them for use on the web and then how to insert them into your web pages.

Some sites also send JavaScript or Flash to your browser, and you will see how to add JavaScript and Flash in your web pages. Both of these technologies are advanced topics that you can go on to learn more about once you have mastered HTML and CSS, if you want to.

HOW IT IS CREATED

Small websites are often written just using HTML and CSS.

Larger websites — in particular those that are updated regularly and use a content management system (CMS), blogging tools, or e-commerce software — often make use of more complex technologies on the web server, but these technologies are actually used to produce HTML and CSS that is then sent to the browser. So, if your site uses these technologies, you will be able to use your new HTML and CSS knowledge to take more control over how your site looks.

Larger, more complex sites like these may use a database to store data, and programming languages such as PHP, ASP.Net, Java, or Ruby on the web server, but you do not need to know these technologies to improve what the user sees. The skills you'll learn in this book should be enough to help you on that road.

HTML5 & CSS3

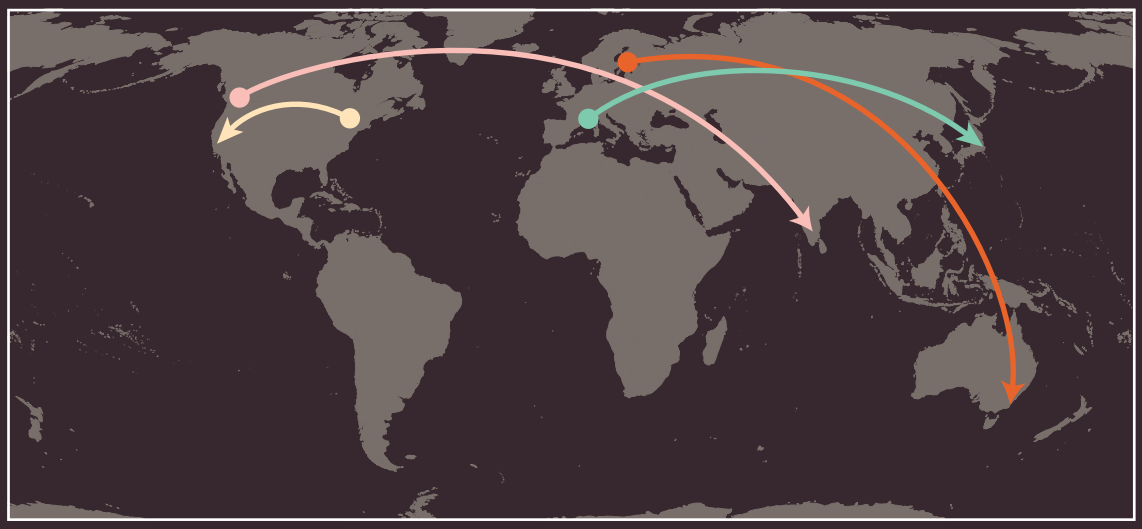
Since the web was first created there have been several versions of HTML and CSS — each intended to be an improvement on the previous version.

At the time of writing this book, HTML5 & CSS3 were still being developed. Although they had not been finalized, many browsers were already supporting some features of these languages and a lot of people were using the latest code on their websites. I have therefore chosen to teach you these latest versions.

Because HTML5 and CSS3 build on previous versions of these languages, learning these means you will also be able to understand the earlier versions of them. I have added clear notes when the code is new and also when it might not work in older browsers.

HOW THE WEB WORKS

When you visit a website, the web server hosting that site could be anywhere in the world. In order for you to find the location of the web server, your browser will first connect to a Domain Name System (DNS) server.



On this page you can see examples that demonstrate how the web server that hosts the website you are visiting can be anywhere in the world. It is the DNS servers that tell your browser how to find the website.

- A user in Barcelona visits `sony.jp` in Tokyo
- A user in New York visits `google.com` in San Francisco
- A user in Stockholm visits `qantas.com.au` in Sydney
- A user in Vancouver visits `airindia.in` in Bangalore

On the right you can see what happens when a web user in England wants to view the website of the Louvre art gallery in France which is located at `www.louvre.fr`. Firstly, the browser in Cambridge contacts a DNS server in London. The DNS server then tells the browser the location of the web server hosting the site in Paris.


1
When you connect to the web, you do so via an Internet Service Provider (ISP). You type a domain name or web address into your browser to visit a site; for example: google.com, bbc.co.uk, microsoft.com.
2
Your computer contacts a network of servers called Domain Name System (DNS) servers. These act like phone books; they tell your computer the IP address associated with the requested domain name. An IP address is a number of up to 12 digits separated by periods / full stops. Every device connected to the web has a unique IP address; it is like the phone number for that computer.
Cambridge
LONDON
3
The unique number that the DNS server returns to your computer allows your browser to contact the web server that hosts the website you requested. A web server is a computer that is constantly connected to the web, and is set up especially to send web pages to users.
PARIS
4
The web server then sends the page you requested back to your web browser.

1

STRUCTURE

- Understanding structure
- Learning about markup
- Tags and elements

We come across all kinds of documents every day of our lives. Newspapers, insurance forms, shop catalogues... the list goes on.

Many web pages act like electronic versions of these documents. For example, newspapers show the same stories in print as they do on websites; you can apply for insurance over the web; and stores have online catalogs and e-commerce facilities.

In all kinds of documents, structure is very important in helping readers to understand the messages you are trying to convey and to navigate around the document. So, in order to learn how to write web pages, it is very important to understand how to structure documents. In this chapter you will:

- See how HTML describes the structure of a web page
- Learn how tags or elements are added to your document
- Write your first web page


Safari File Edit View History Bookmarks Develop Window Help
Sun 12:17
http://www.htmlandcssbook.com/code/chapter-01/example.html
http://www.htmlandcssbook.com/code/chapter-01/example.html
This is the Main Heading
This text might be an introduction to the rest of the page. And if the page is a long one it might be split up into several sub-headings.
This is a Sub-Heading
Many long articles have sub-headings so to help you follow the structure of what is being written. There may even be sub-sub-headings (or lower-level headings).
Another Sub-Heading
Here you can see another sub-heading.
MacBook Pro

HOW PAGES USE STRUCTURE

Think about the stories you read in a newspaper: for each story, there will be a headline, some text, and possibly some images. If the article is a long piece, there may be subheadings that split the story into separate sections or quotes from those involved. Structure helps readers understand the stories in the newspaper.

The structure is very similar when a news story is viewed online (although it may also feature audio or video). This is illustrated on the right with a copy of a newspaper alongside the corresponding article on its website.

Now think about a very different type of document — an insurance form. Insurance forms often have headings for different sections, and each section contains a list of questions with areas for you to fill in details or checkboxes to tick. Again, the structure is very similar online.

The Guardian | Monday 12 September 2011 31

Read more on MediaGuardian.co.uk

Digital economy or bust
Part 33. In which the team turn up the volume with inside track on The X Factor - and get a glimpse of the future

Coming up this week
Monday: Shortlists for Student Media Awards announced
Wednesday to Friday: Coverage of the RTS Cambridge Convention

Interview Rio Caraeff

Vevo revolutionary

Universal's former mobile chief is leading the music industry's fight to shake up online video. He reveals his frustration with MTV, and says why no one need own music if his site succeeds. Interview by **Mark Sweney**

If Rio Caraeff succeeds, perhaps only diehard fans will need to own music. His online music video site, part-owned by the two largest record companies, also hopes to have the same impact as MTV and to be an answer to YouTube. Chuck those goals in with that of making the industry less dependent on the purchase of recordings, and for Caraeff there is clearly plenty to do.

Caraeff is the youthful chief executive of Vevo - launched in late 2009 with the backing of three of the four major groups, Sony Music, Universal Music and EMI - who is taking the venture international with a rollout starting in the UK and continental Europe. "Sex, music and sports are the only entertainment categories on the planet that people love that can build audiences at the scale of billions of people," he says. "I'm in the business of connecting billions of people to music," is his modestly stated aim.

With global CD sales plummeting by $1.5bn last year, Caraeff's mission is clear. "We wouldn't have created Vevo if we didn't need it," he says. "The industry felt it was necessary. If MTV was doing a great job paying royalties, if YouTube [was], there would have been no need. We have invested tens of millions to be responsible for our own destiny. We can't sit back and say 'I hope Apple or whoever figures this out'."

Vevo's relationship with Google, the owner of the world's largest video-sharing platform YouTube, is clearly critical. Michael Grade called the company a "parasite" and Sir Martin Sorrell described it as a "frenemy". Despite the combative relationship the music industry has historically had with players in the digital space, Caraeff prefers to characterise Vevo's dealings with YouTube as "symbiotic" - although "declaration of independence" would be more appropriate.

"We said 'let's figure out how to work with them'," he explains. "There are no duplicate copies [of music videos] on YouTube, there were thousands before, the official versions are only available from us. They don't threaten us. YouTube is a place where people can upload any video in the world, we're not trying to compete." Caraeff points out that 30% of Vevo's traffic comes from YouTube search, and 30% comes from recommendations of videos that users might like to watch that appear on the side of the YouTube web pages when a user is viewing clips.

Video vexations ... Rio Caraeff says 'if MTV was doing a

'We are about access: it is the only scalable model for the music industry; the question is, how do you do that and make money?'

Free access

Vevo's business model is all about providing music videos that fans can access free, funded by advertising - or to put it another way - give consumers an alternative to owning songs. "I believe the future is access, not ownership, not iTunes as it is today," he says. "We're not trying to sell people music; our customers are not the small amount of people that want to buy music. We are about providing access; it is the only scalable model for the music industry; the question is, how do you do that and make money?"

Which raises the question of how well Vevo is actually doing. Caraeff doesn't want to give away too much commercially but says it is already making "hundreds of millions of dollars" in revenue, although there are hosting costs to pay. More than half of gross revenue goes to content owners - the label, artist or licensee - with the remainder being kept by Vevo or paid to partners such as YouTube. He says that Vevo is "significantly ahead" of its original business plan - about 40% ahead to be precise - and is on track to achieve profitability "in the very early part of next year".

Yet there are problems. Caraeff's business is dependent on advertising, and he is frustrated by the low rates that companies pay to run campaigns around music content. His contention is that advertisers treat music content as inferior and that Vevo's role is to "own" the prime content and then be able to position it as a premium product. Think the free-to-access equivalent of BSkyB owning Premier League football.

"The audience that loves music is vast and promising: it should be treated as

Curriculum vitae

Age 36

Education Did not go to u... cause "I started my first c... I was 18"

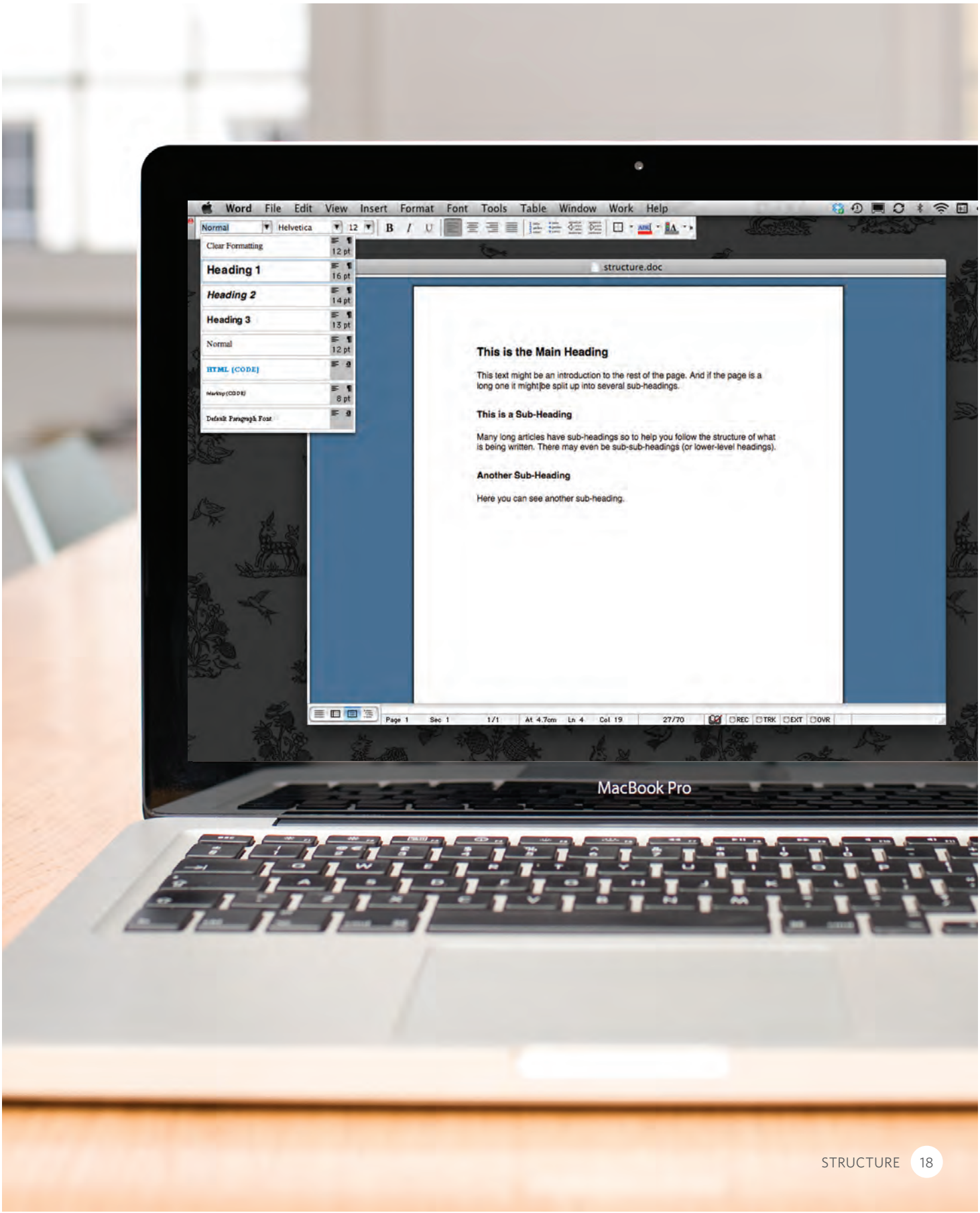
Career
2004 vice-president ... Pictures **2005** gener... Universal Music Mo... vice-president, Uni... division, responsi... and new technolo... tive, Vevo

STRUCTURING WORD DOCUMENTS

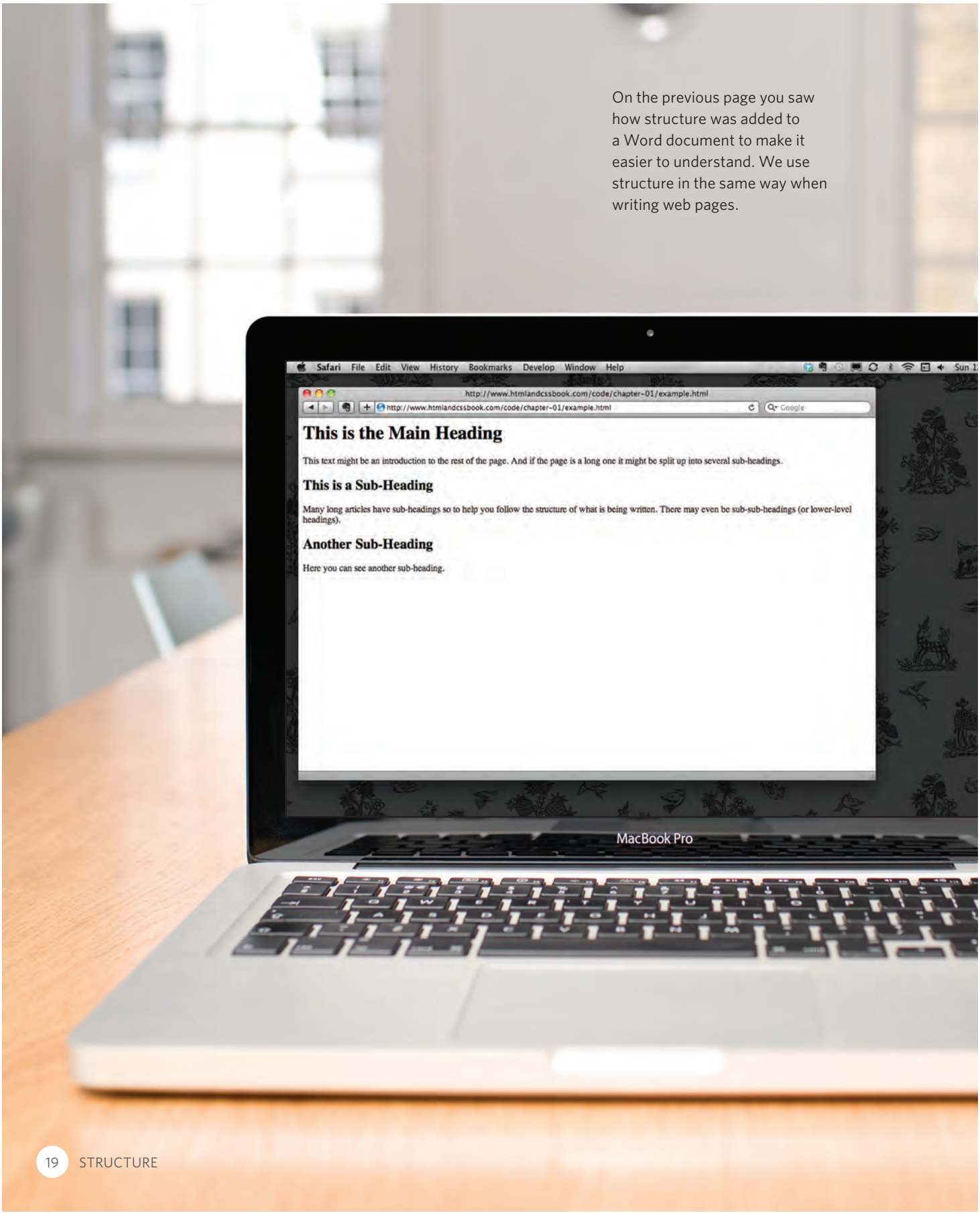
The use of headings and subheadings in any document often reflects a hierarchy of information. For example, a document might start with a large heading, followed by an introduction or the most important information.

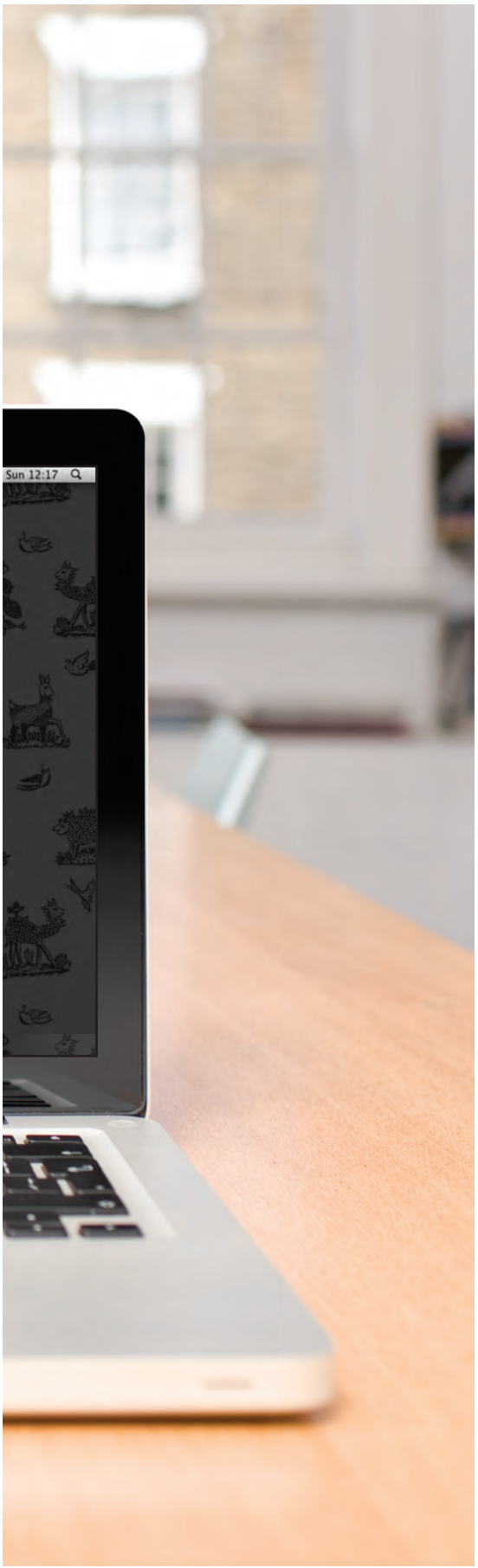
This might be expanded upon under subheadings lower down on the page. When using a word processor to create a document, we separate out the text to give it structure. Each topic might have a new paragraph, and each section can have a heading to describe what it covers.

On the right, you can see a simple document in Microsoft Word. The different styles for the document, such as different levels of heading, are shown in the drop down box. If you regularly use Word, you might have also used the formatting toolbar or palette to do this.


Word File Edit View Insert Format Font Tools Table Window Work Help
Normal
Helvetica
12
Clear Formatting
Heading 1
Heading 2
Heading 3
Normal
HTML (CODE)
Default Paragraph Font
structure.doc
This is the Main Heading
This text might be an introduction to the rest of the page. And if the page is a long one it might be split up into several sub-headings.
This is a Sub-Heading
Many long articles have sub-headings so to help you follow the structure of what is being written. There may even be sub-sub-headings (or lower-level headings).
Another Sub-Heading
Here you can see another sub-heading.
MacBook Pro

On the previous page you saw how structure was added to a Word document to make it easier to understand. We use structure in the same way when writing web pages.





HTML DESCRIBES THE STRUCTURE OF PAGES

In the browser window you can see a web page that features exactly the same content as the Word document you met on the page 18. To describe the structure of a web page, we add code to the words we want to appear on the page.

You can see the HTML code for this page below. Don't worry about what the code means yet. We start to look at it in more detail on the next page. Note that the HTML code is in blue, and the text you see on screen is in black.

```
<html>
  <body>
    <h1>This is the Main Heading</h1>
    <p>This text might be an introduction to the rest of
       the page. And if the page is a long one it might
       be split up into several sub-headings.<p>
    <h2>This is a Sub-Heading</h2>
    <p>Many long articles have sub-headings so to help
       you follow the structure of what is being written.
       There may even be sub-sub-headings (or lower-level
       headings).</p>
    <h2>Another Sub-Heading</h2>
    <p>Here you can see another sub-heading.</p>
  </body>
</html>
```

The HTML code (in blue) is made up of characters that live inside angled brackets — these are called HTML **elements**. Elements are usually made up of two **tags**: an opening tag and a closing tag. (The closing tag has an extra forward slash in it.) Each HTML element tells the browser something about the information that sits between its opening and closing tags.

HTML USES ELEMENTS TO DESCRIBE THE STRUCTURE OF PAGES

Let's look closer at the code from the last page. There are several different elements. Each element has an opening tag and a closing tag.

CODE

Tags act like containers. They tell you something about the information that lies between their opening and closing tags.

DESCRIPTION

The opening `<html>` tag indicates that anything between it and a closing `</html>` tag is HTML code.

The `<body>` tag indicates that anything between it and the closing `</body>` tag should be shown inside the main browser window.

Words between `<h1>` and `</h1>` are a main heading.

A paragraph of text appears between these `<p>` and `</p>` tags.

Words between `<h2>` and `</h2>` form a sub-heading.

Here is another paragraph between opening `<p>` and closing `</p>` tags.

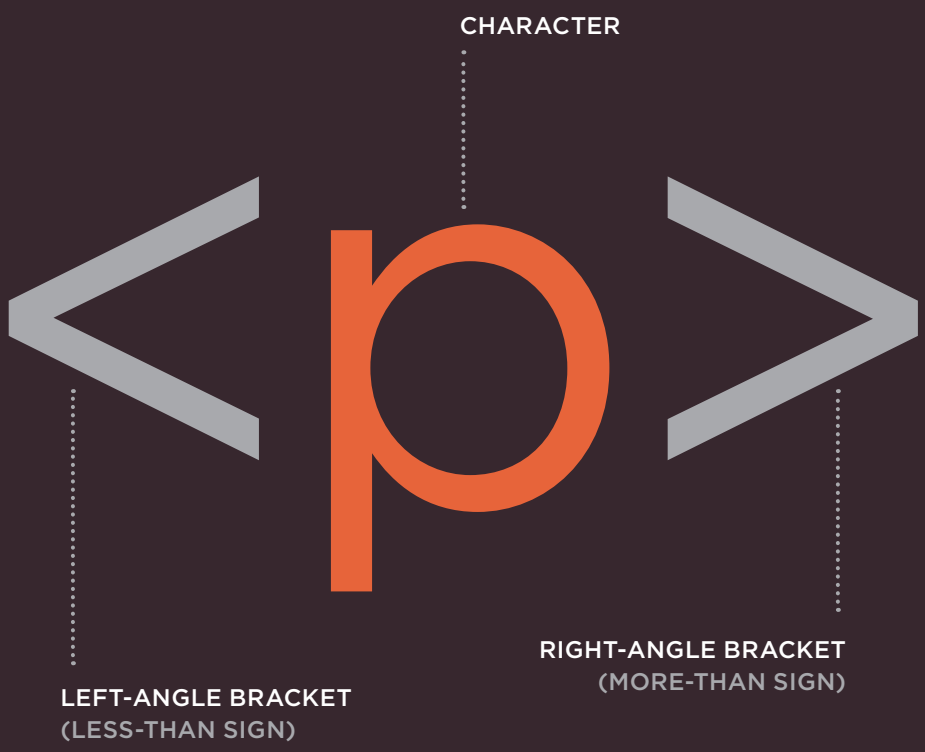
Another sub-heading inside `<h2>` and `</h2>` tags.

Another paragraph inside `<p>` and `</p>` tags.

The closing `</body>` tag indicates the end of what should appear in the main browser window.

The closing `</html>` tag indicates that it is the end of the HTML code.

A CLOSER LOOK AT TAGS

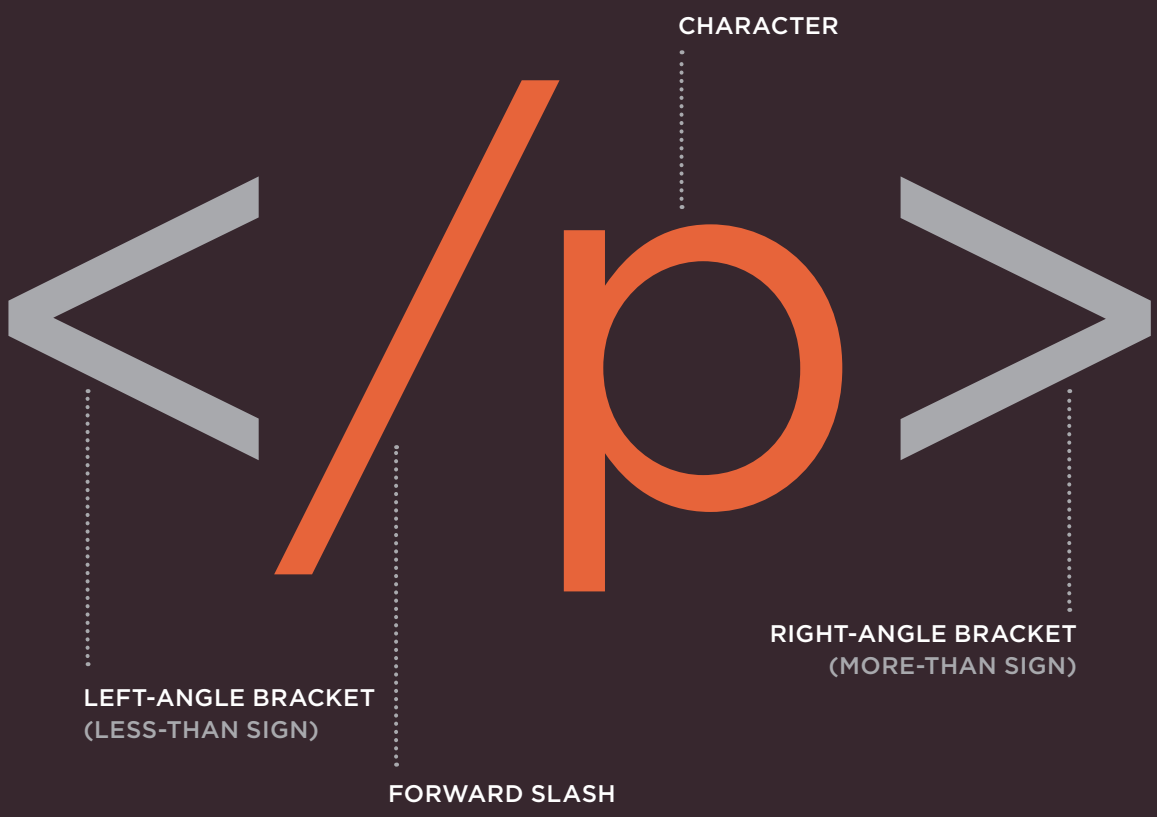


OPENING TAG

The characters in the brackets indicate the tag's purpose.

For example, in the tags above the p stands for paragraph.

The closing tag has a forward slash after the the < symbol.



CLOSING TAG

The terms "tag" and "element" are often used interchangeably. Strictly speaking, however, an element comprises the opening tag *and* the closing tag *and* any content that lies between them.

ATTRIBUTES TELL US MORE ABOUT ELEMENTS

Attributes provide additional information about the contents of an element. They appear on the opening tag of the element and are made up of two parts: a name and a value, separated by an equals sign.

The attribute **name** indicates what kind of extra information you are supplying about the element's content. It should be written in lowercase.

The **value** is the information or setting for the attribute. It should be placed in double quotes. Different attributes can have different values.

Here an attribute called `lang` is used to indicate the language used in this element. The value of this attribute on this page specifies it is in US English.

HTML5 allows you to use uppercase attribute names and omit the quotemarks, but this is not recommended.

The majority of attributes can only be used on certain elements, although a few attributes (such as `lang`) can appear on any element.

Most attribute values are either pre-defined or follow a stipulated format. We will look at the permitted values as we introduce each new attribute.

The value of the `lang` attribute is an abbreviated way of specifying which language is used inside the element that all browsers understand.

BODY, HEAD & TITLE

<body>

You met the <body> element in the first example we created. Everything inside this element is shown inside the main browser window.

<head>

Before the <body> element you will often see a <head> element. This contains information *about* the page (rather than information that is shown within the main part of the browser window that is highlighted in blue on the opposite page). You will usually find a <title> element inside the <head> element.

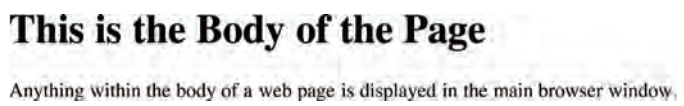
<title>

The contents of the <title> element are either shown in the top of the browser, above where you usually type in the URL of the page you want to visit, or on the tab for that page (if your browser uses tabs to allow you to view multiple pages at the same time).

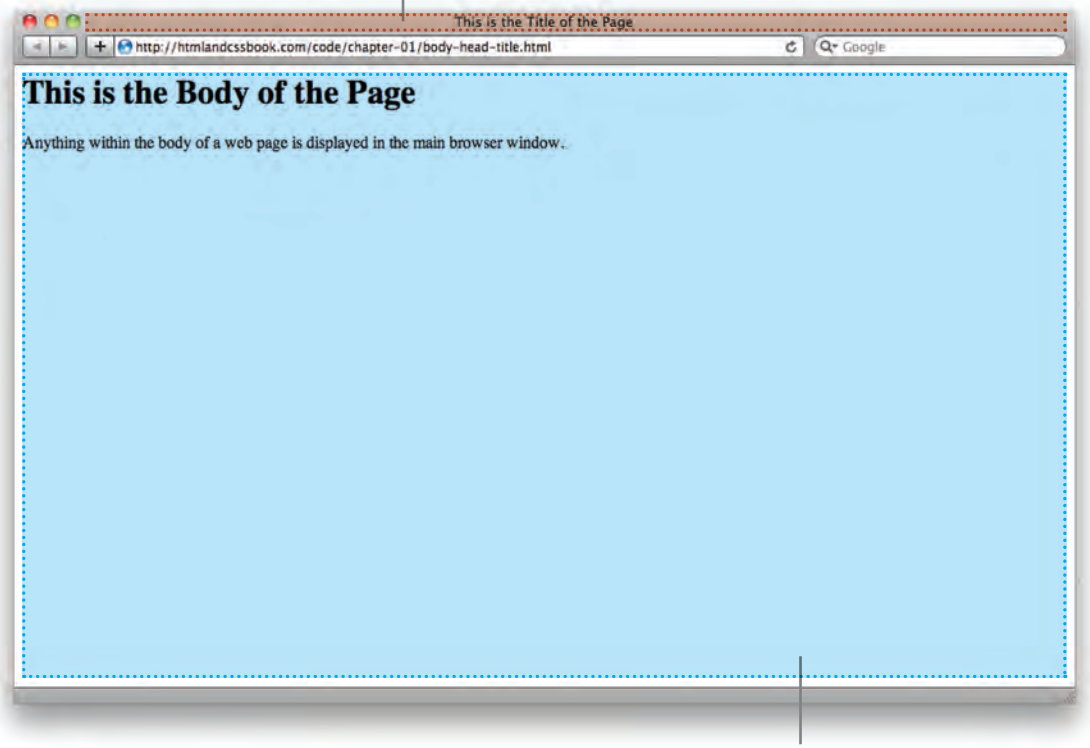
/chapter-01/body-head-title.html **HTML**

```
<html>
  <head>
    <title>This is the Title of the Page</title>
  </head>
  <body>
    <h1>This is the Body of the Page</h1>
    <p>Anything within the body of a web page is
    displayed in the main browser window.</p>
  </body>
</html>
```

RESULT



Anything written between the <title> tags will appear in the title bar (or tabs) at the top of the browser window, highlighted in orange here.



Anything written between the <body> tags will appear in the main browser window, highlighted in blue here.

You may know that HTML stands for HyperText Markup Language. The HyperText part refers to the fact that HTML allows you to create links that allow visitors to move from one page to another quickly and easily. A markup language allows you to annotate text, and these annotations provide additional meaning to the contents of a document. If you think of a web page, we add code around the original text we want to display and the browser then uses the code to display the page correctly. So the tags we add are the markup.

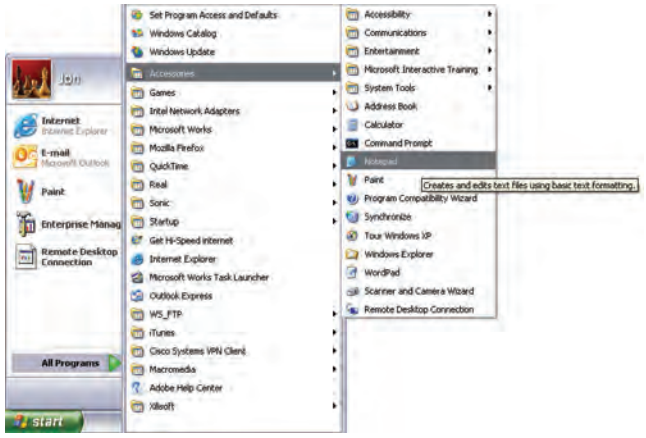
CREATING A WEB PAGE ON A PC

To create your first web page on a PC, start up Notepad. You can find this by going to:

Start
All Programs (or Programs)
Accessories
Notepad

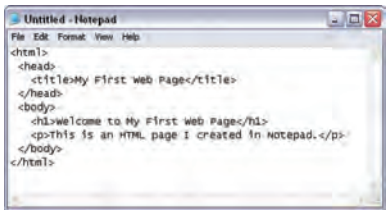
You might also like to download a free editor called Notepad++ from `notepad-plus-plus.org`.

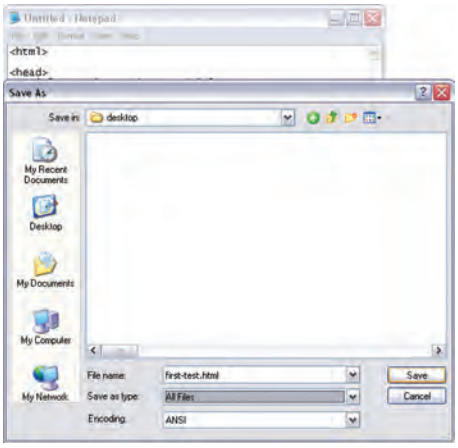
1



Type the code shown on the right.

2

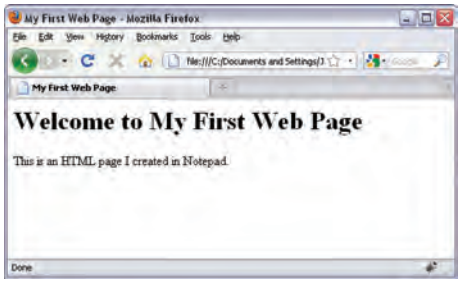




3

Go to the File menu and select ***Save as...*** You will need to save the file somewhere you can remember. If you like, you could create a folder for any examples that you try out from this book.

Save this file as `first-test.html`. Make sure that the ***Save as type*** drop down has ***All Files*** selected.



4

Start your web browser. Go to the ***File*** menu and select ***Open***. Browse to the file that you just created, select it and click on the ***Open*** button. The result should look something like the screen shot to the left.

If it doesn't look like this, find the file you just created on your computer and make sure that it has the file extension `.html` (if it is `.txt` then you need to go back to Notepad and save the file again, but this time put quote marks around the name "`first-test.html`").

CREATING A WEB PAGE ON A MAC

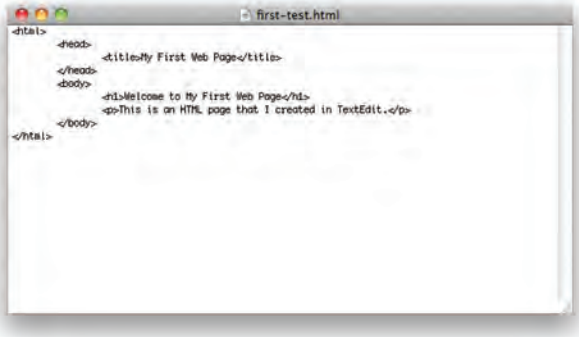
To create your first web page on a Mac, start up TextEdit. This should be in your ***Applications*** folder.

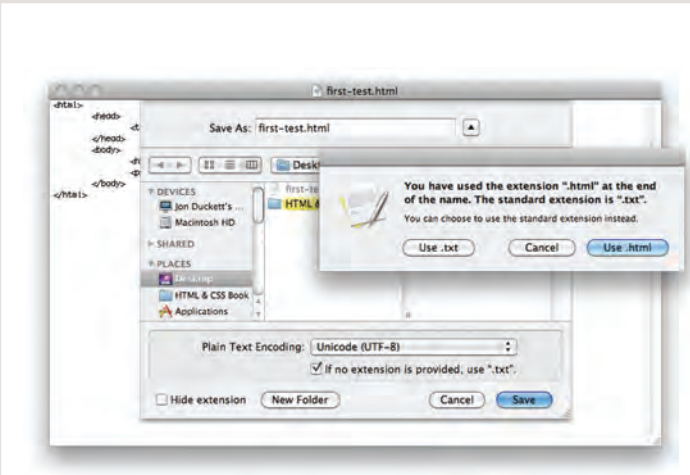
You might also like to download a free text editor for creating web pages called TextWrangler which is available from `barebones.com`.

1

Type the code shown on the right.

2



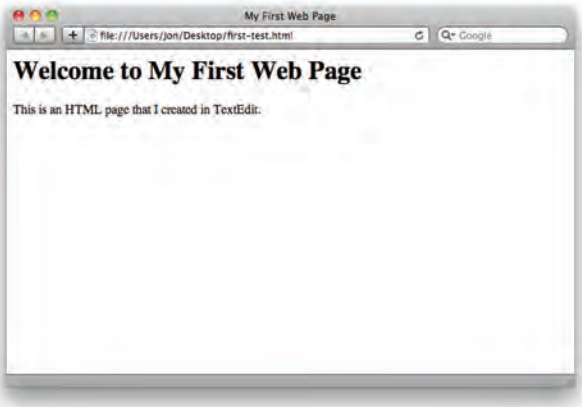


3

Now go to the ***File*** menu and select ***Save as...*** You will need to save the file somewhere you can remember.

If you like, you could create a folder for any examples that you try out from this book. Save this file as `first-test.html`. You will probably see a window like the screen shot to the left.

You want to select the ***Use .html*** button.

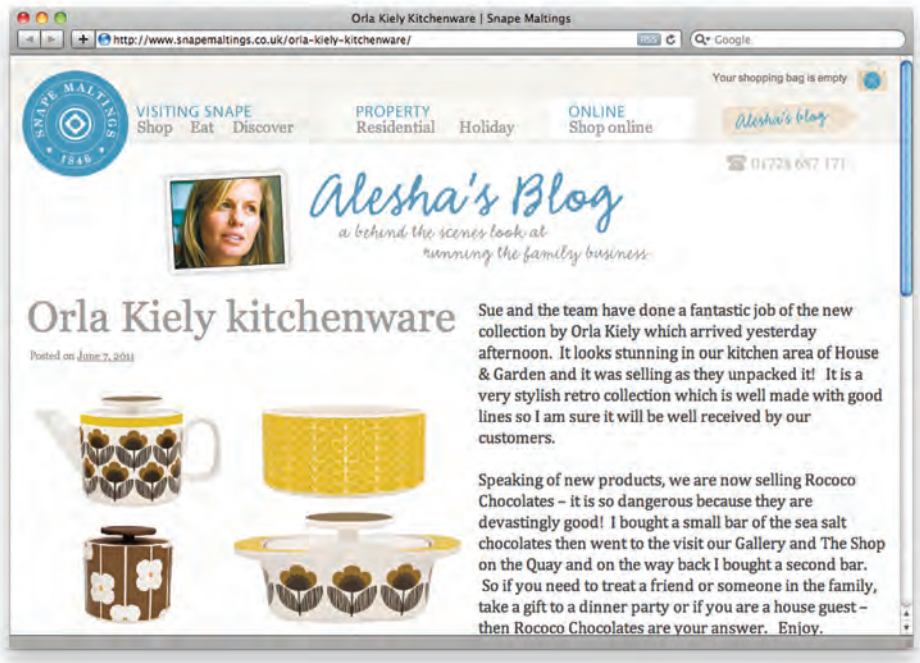


4

Next, start your web browser, go to the ***File*** menu, and select ***Open***. You should browse to the file that you just created, select it and click on the ***Open*** button. The result should look like the screen shot to the left.

If it doesn't look like this, you might need to change one of the settings in TextEdit. Go to the TextEdit menu and select ***Preferences***. Then on the preferences for ***Open and Save***, tick the box that says ***Ignore rich text commands in HTML files***. Now try to save the file again.

CODE IN A CONTENT MANAGEMENT SYSTEM



If you are working with a content management system, blogging platform, or e-commerce application, you will probably log into a special administration section of the website to control it. The tools provided in the administration sections of these sites usually allow you to edit parts of the page rather than the entire page, which means you will rarely see the `<html>`, `<head>`, or `<body>` elements.

Looking at the content management system on the opposite page, you have a box that allows you to enter a title for the page, another box for the main article, a way to enter a publication date, and something to indicate which section of the site this page belongs in.

For an e-commerce store, you might have boxes that allow you to enter a title for the product, a description of the product, its price, and the quantity available.

That is because they use a single 'template' to control all of the pages for a section of the site. (For example, an e-commerce system might use the same template to show all of their products.) The information you supply is placed into the templates.

The advantage of this approach is that people who do not know how to write web pages can add information to a website and it is also possible to change the presentation of something in the template, and it will automatically update every page that uses that template. If you imagine an e-commerce store with 1,000 items for sale, just

altering one template is a lot easier than changing the page for each individual product. In systems like this, when you have a large block of text that you can edit, such as a news article, blog entry or the description of a product in an e-commerce store, you will often see a text editor displayed.

Text editors usually have controls a little like those on your word processor, giving you different options to style text, add links or insert images. Behind the scenes these editors are adding HTML code to your text, just like the code you have seen earlier in this chapter. Many of these editors will have an option that allows you to see (and edit) the code that they produce.

Once you know how to read and edit this code, you can take more control over these sections of your website.

In the example above, you can see that the text editor has a tab for Visual / HTML views of what the user enters. Other systems might have a button (which often shows angle brackets) to indicate how to access the code.

Some content management systems offer tools that also allow you to edit the template files. If you do try to edit template files you need to check the documentation for your CMS as they all differ from each other. You need to be careful when editing template files because if you delete the wrong piece of code or add something in the wrong place the site may stop working entirely.

LOOKING AT HOW OTHER SITES ARE BUILT

When the web was first taking off, one of the most common ways to learn about HTML and discover new tips and techniques was to look at the source code that made up web pages.

These days there are many more books and online tutorials that teach HTML, but you can still look at the code that a web server sends to you. To try this out for yourself, simply go to the sample code for this chapter, at `www.htmlandcssbook.com/code/` and click on the link called "View Source."

Once you have opened this page, you can look for the ***View*** menu in your browser, and select the option that says ***Source*** or ***View source***. (The title changes depending on what browser you are using.)

You should see a new window appear, and it will contain the source code that was used to create this page.

You can see this result in the photograph on the right. The page you see is the window at the top; the code is below.

At first this code might look complicated but don't be discouraged. By the time you have finished the next chapter of this book, you will be able to understand it.

All of the examples for this book are on the website, and you can use this simple technique on any of the example pages to see how they work.

You can also download all of the code for this book from the same website by clicking on the "Download" link.


Safari File Edit View History Bookmarks Develop Window Help
Wed 19:29
Hide Toolbar
Customize Toolbar...
Show Bookmarks Bar
Show Tab Bar
Hide Status Bar
Stop
Reload Page
Actual Size
Zoom In
Zoom Out
Zoom Text Only
View Source
Text Encoding
Enter Reader
View the Source of This Page
http://htmlandcssbook.com/code/chapter-01/view-source.html
How to View the Source of This Page
If you go to your web browser's View menu, you will see an option that allows you to see the source code behind a web page.
When the web was first taking off this was a very popular way for people to learn how web pages had been built.
Today, people still often view the source of pages to learn how a web page has been built.
Source of http://htmlandcssbook.com/code/chapter-01/view-source.html
<html>
<head>
<title>View the Source of This Page</title>
</head>
<body>
<h1>How to View the Source of This Page</h1>
If you go to your web browser's View menu, you will see an option that allows you to see the source code behind a web page.
When the web was first taking off this was a very popular way for people to learn how web pages had been built.
Today, people still often view the source of pages to learn how a web page has been built.
</body>
</html>
MacBook Pro

SUMMARY
STRUCTURE

- HTML pages are text documents.
- HTML uses tags (characters that sit inside angled brackets) to give the information they surround special meaning.
- Tags are often referred to as elements.
- Tags usually come in pairs. The opening tag denotes the start of a piece of content; the closing tag denotes the end.
- Opening tags can carry attributes, which tell us more about the content of that element.
- Attributes require a name and a value.
- To learn HTML you need to know what tags are available for you to use, what they do, and where they can go.

2

TEXT

- Headings and paragraphs
- Bold, italic, emphasis
- Structural and semantic markup

When creating a web page, you add tags (known as markup) to the contents of the page. These tags provide extra meaning and allow browsers to show users the appropriate structure for the page.

In this chapter we focus on how to add markup to the text that appears on your pages. You will learn about:

- **Structural markup:** the elements that you can use to describe both headings and paragraphs
- **Semantic markup:** which provides extra information; such as where emphasis is placed in a sentence, that something you have written is a quotation (and who said it), the meaning of acronyms, and so on


POUDRE D'IRIS
REVOLUTION VELOCIPEDIQUE !
Nouvelle BICYCLETTE
G. DUPONT
SAVON SURFIN
VIOLETTE
BON POINT
Il faut être fort
HELLO! ISN'T IT A LOVELY DAY?
DISTILLERIE DE MONTMORILLON
AMER LÉON
TONIQUE & HYGIÉNIQUE
AU CITRON – SANS ALCOOL
LÉON DARDANT
MONTMORILLON (Vienne)
The Spotty Bird
GIRAUD-DUQUESNE
The Story in the Book
Chapter 1
MacBook Pro
Copperplate

HEADINGS

<h1>
<h2>
<h3>
<h4>
<h5>
<h6>

HTML has six "levels" of headings:

<h1> is used for main headings

<h2> is used for subheadings

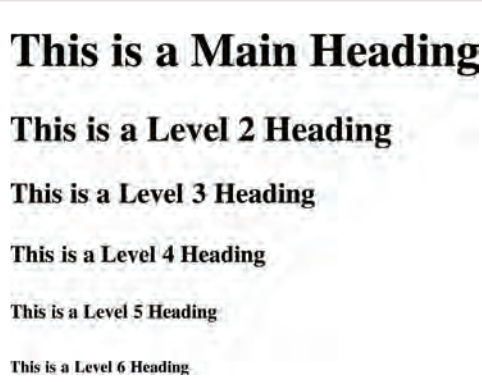
If there are further sections under the subheadings then the <h3> element is used, and so on...

Browsers display the contents of headings at different sizes. The contents of an <h1> element is the largest, and the contents of an <h6> element is the smallest. The exact size at which each browser shows the headings can vary slightly. Users can also adjust the size of text in their browser. You will see how to control the size of text, its color, and the fonts used when we come to look at CSS.

chapter-02/headings.html **HTML**

```
<h1>This is a Main Heading</h1>
<h2>This is a Level 2 Heading</h2>
<h3>This is a Level 3 Heading</h3>
<h4>This is a Level 4 Heading</h4>
<h5>This is a Level 5 Heading</h5>
<h6>This is a Level 6 Heading</h6>
```

RESULT



PARAGRAPHS

HTML chapter-02/paragraphs.html

```
<p>A paragraph consists of one or more sentences
   that form a self-contained unit of discourse. The
   start of a paragraph is indicated by a new
   line.</p>
<p>Text is easier to understand when it is split up
   into units of text. For example, a book may have
   chapters. Chapters can have subheadings. Under
   each heading there will be one or more
   paragraphs.</p>
```

RESULT

A paragraph consists of one or more sentences that form a self-contained unit of discourse. The start of a paragraph is indicated by a new line.

Text is easier to understand when it is split up into units of text. For example, a book may have chapters. Chapters can have subheadings. Under each heading there will be one or more paragraphs.

To create a paragraph, surround the words that make up the paragraph with an opening tag and closing tag.

By default, a browser will show each paragraph on a new line with some space between it and any subsequent paragraphs.

BOLD & ITALIC

By enclosing words in the tags and we can make characters appear bold.

The element also represents a section of text that would be presented in a visually different way (for example key words in a paragraph) although the use of the element does not imply any additional meaning.

chapter-02/bold.html **HTML**

```
<p>This is how we make a word appear <b>bold.</b>
   </p>
<p>Inside a product description you might see some
   <b>key features</b> in bold.</p>
```

RESULT

This is how we make a word appear **bold.**

Inside a product description you might see some **key features** in bold.

By enclosing words in the tags and we can make characters appear italic.

The element also represents a section of text that would be said in a different way from surrounding content — such as technical terms, names of ships, foreign words, thoughts, or other terms that would usually be italicized.

chapter-02/italic.html **HTML**

```
<p>This is how we make a word appear <i>italic</i>.
   </p>
<p>It's a potato <i>Solanum teberosum</i>.</p>
<p>Captain Cook sailed to Australia on the
   <i>Endeavour</i>.</p>
```

RESULT

This is how we make a word appear *italic*.

It's a potato *Solanum teberosum*.

Captain Cook sailed to Australia on the *Endeavour*.

SUPERSCRIPT & SUBSCRIPT

HTML chapter-02/superscript-and-subscript.html

```
<p>On the 4<sup>th</sup> of September you will learn
   about E=MC<sup>2</sup>.</p>
<p>The amount of CO<sub>2</sub> in the atmosphere
   grew by 2ppm in 2009<sub>1</sub>.</p>
```

RESULT

On the 4^{th} of September you will learn about $E=MC^2$.

The amount of CO_2 in the atmosphere grew by 2ppm in 2009_1.

<sup>

The <sup> element is used to contain characters that should be superscript such as the suffixes of dates or mathematical concepts like raising a number to a power such as 2^2.

<sub>

The <sub> element is used to contain characters that should be subscript. It is commonly used with foot notes or chemical formulas such as H_20.

WHITE SPACE

In order to make code easier to read, web page authors often add extra spaces or start some elements on new lines.

When the browser comes across two or more spaces next to each other, it only displays one space. Similarly if it comes across a line break, it treats that as a single space too. This is known as **white space collapsing**.

You will often see that web page authors take advantage of white space collapsing to indent their code in order to make it easier to follow.

chapter-02/white-space.html **HTML**

```
<p>The moon is drifting away from Earth.</p>
<p>The moon        is drifting away from Earth.</p>
<p>The moon is drifting away from

      Earth.</p>
```

RESULT

The moon is drifting away from Earth.

The moon is drifting away from Earth.

The moon is drifting away from Earth.

LINE BREAKS & HORIZONTAL RULES

HTML chapter-02/line-breaks.html

```
<p>The Earth<br />gets one hundred tons heavier
   every day<br />due to falling space dust.</p>
```

RESULT

The Earth
gets one hundred tons heavier every day
due to falling space dust.

As you have already seen, the browser will automatically show each new paragraph or heading on a new line. But if you wanted to add a line break inside the middle of a paragraph you can use the line break tag `<br />`.

HTML chapter-02/horizontal-rules.html

```
<p>Venus is the only planet that rotates
   clockwise.</p>
<hr />
<p>Jupiter is bigger than all the other planets
   combined.</p>
```

RESULT

Venus is the only planet that rotates clockwise.

Jupiter is bigger than all the other planets combined.

<hr />

To create a break between themes — such as a change of topic in a book or a new scene in a play — you can add a horizontal rule between sections using the `<hr />` tag.

There are a few elements that do not have any words between an opening and closing tag. They are known as **empty elements** and they are written differently.

An empty element usually has only one tag. Before the closing angled bracket of an empty element there will often be a space and a forward slash character. Some web page authors miss this out but it is a good habit to get into.

VISUAL EDITORS & THEIR CODE VIEWS

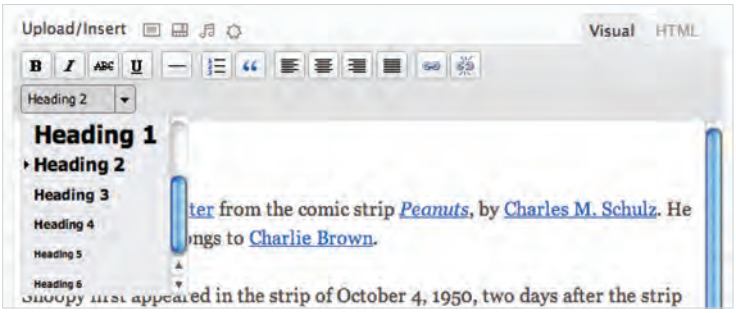
Content management systems and HTML editors such as Dreamweaver usually have two views of the page you are creating: a visual editor and a code view.

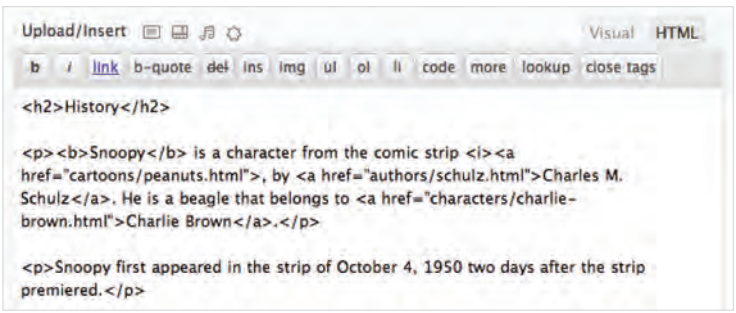
Visual editors often resemble word processors. Although each editor will differ slightly, there are some features that are common to most editors that allow you to control the presentation of text.

- Headings are created by highlighting text then using a drop-down box to select a heading.
- Bold and italic text are created by highlighting some text and pressing a **b** or **i** button.
- New paragraphs are created using the return or the enter key.
- Line breaks are created by pressing the shift key and the return key at the same time.
- Horizontal rules are created using a button with a straight line on it.

If you copy and paste text from a program that allows you to format text (such as Word) into a visual editor, it may add extra markup. To prevent this copy the text into a plain text editor first (such as Notepad on a PC or TextEdit on a Mac) and then copy it from that program and paste it into the visual editor.

Code views show you the code created by the visual editor so you can manually edit it, or so you can just enter new code yourself. It is often activated using a button with an icon that says HTML or has angled brackets. White space may be added to the code by the editor to make the code easier to read.





SEMANTIC MARKUP

There are some text elements that are not intended to affect the structure of your web pages, but they do add extra information to the pages — they are known as semantic markup.

In the rest of the chapter you will meet some more elements that will help you when you are adding text to web pages. For example, you are going to meet the `<em>` element that allows you to indicate where emphasis should be placed on selected words and the `<blockquote>` element which indicates that a block of text is a quotation.

Browsers often display the contents of these elements in a different way. For example, the content of the `<em>` element is shown in italics, and a `<blockquote>` is usually indented. But you should not use them to change the way that your text looks; their purpose is to describe the content of your web pages more accurately.

The reason for using these elements is that other programs, such as screen readers or search engines, can use this extra information. For example, the voice of a screen reader may add emphasis to the words inside the `<em>` element, or a search engine might register that your page features a quote if you use the `<blockquote>` element.

STRONG & EMPHASIS

The use of the element indicates that its content has strong importance. For example, the words contained in this element might be said with strong emphasis.

By default, browsers will show the contents of a element in bold.

chapter-02/strong.html **HTML**

```
<p><strong>Beware:</strong> Pickpockets operate in
   this area.</p>
<p>This toy has many small pieces and is <strong>not
   suitable for children under five years old.
   </strong></p>
```

RESULT

Beware: Pickpockets operate in this area.

This toy has many small pieces and is **not suitable for children under five years old.**

The element indicates emphasis that subtly changes the meaning of a sentence.

By default browsers will show the contents of an element in italic.

chapter-02/emphasis.html **HTML**

```
<p>I <em>think</em> Ivy was the first.</p>
<p>I think <em>Ivy</em> was the first.</p>
<p>I think Ivy was the <em>first</em>.</p>
```

RESULT

I *think* Ivy was the first.

I think *Ivy* was the first.

I think Ivy was the *first*.

QUOTATIONS

HTML chapter-02/quotations.html

```
<blockquote cite="http://en.wikipedia.org/wiki/
  Winnie-the-Pooh">
  <p>Did you ever stop to think, and forget to start
     again?</p>
</blockquote>
<p>As A.A. Milne said, <q>Some people talk to
   animals. Not many listen though. That's the
   problem.</q></p>
```

RESULT

Did you ever stop to think, and forget to start again?

As A.A. Milne said, "Some people talk to animals. Not many listen though. That's the problem."

There are two elements commonly used for marking up quotations:

<blockquote>

The `<blockquote>` element is used for longer quotes that take up an entire paragraph. Note how the `<p>` element is still used inside the `<blockquote>` element.

Browsers tend to indent the contents of the `<blockquote>` element, however you should not use this element just to indent a piece of text — rather you should achieve this effect using CSS.

<q>

The `<q>` element is used for shorter quotes that sit within a paragraph. Browsers are supposed to put quotes around the `<q>` element, however Internet Explorer does not — therefore many people avoid using the `<q>` element.

Both elements may use the `cite attribute` to indicate where the quote is from. Its value should be a URL that will have more information about the source of the quotation.

ABBREVIATIONS & ACRONYMS

<abbr>

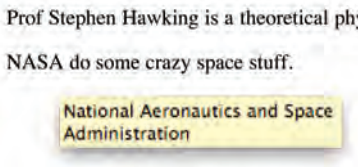
If you use an abbreviation or an acronym, then the <abbr> element can be used. A `title` attribute on the opening tag is used to specify the full term.

In HTML 4 there was a separate <acronym> element for acronyms. To spell out the full form of the acronym, the `title` attribute was used (as with the <abbr> element above). HTML5 just uses the <abbr> element for both abbreviations and acronyms.

chapter-02/abbreviations.html **HTML**

```
<p><abbr title="Professor">Prof</abbr> Stephen
   Hawking is a theoretical physicist and
   cosmologist.</p>
<p><acronym title="National Aeronautics and Space
   Administration">NASA</acronym> do some crazy
   space stuff.</p>
```

RESULT



CITATIONS & DEFINITIONS

HTML chapter-02/citations.html

```
<p><cite>A Brief History of Time</cite> by Stephen
   Hawking has sold over ten million copies
   worldwide.</p>
```

RESULT

A Brief History of Time by Stephen Hawking has sold over ten million copies worldwide.

<cite>

When you are referencing a piece of work such as a book, film or research paper, the `<cite>` element can be used to indicate where the citation is from.

In HTML5, `<cite>` should not really be used for a person's name — but it was allowed in HTML 4, so most people are likely to continue to use it.

Browsers will render the content of a `<cite>` element in italics.

HTML chapter-02/definitions.html

```
<p>A <dfn>black hole</dfn> is a region of space from
   which nothing, not even light, can escape.</p>
```

RESULT

A black hole is a region of space from which nothing, not even light, can escape.

<dfn>

The first time you explain some new terminology (perhaps an academic concept or some jargon) in a document, it is known as the defining instance of it.

The `<dfn>` element is used to indicate the defining instance of a new term.

Some browsers show the content of the `<dfn>` element in italics. Safari and Chrome do not change its appearance.

AUTHOR DETAILS

<address>

The <address> element has quite a specific use: to contain contact details for the author of the page.

It can contain a physical address, but it does not have to. For example, it may also contain a phone number or email address.

Browsers often display the content of the <address> element in italics.

You may also be interested in something called the hCard microformat for adding physical address information to your markup.

ONLINE EXTRA:
You can find out more about hCards on the website accompanying this book.

chapter-02/address.html **HTML**

```
<address>
  <p><a href="mailto:homer@example.org">
     homer@example.org</a></p>
  <p>742 Evergreen Terrace, Springfield.</p>
</address>
```

RESULT

CHANGES TO CONTENT

HTML chapter-02/insert-and-delete.html

```
<p>It was the <del>worst</del> <ins>best</ins> idea
   she had ever had.</p>
```

RESULT

It was the ~~worst~~ best idea she had ever had.

<ins> <del>

The <ins> element can be used to show content that has been inserted into a document, while the <del> element can show text that has been deleted from it.

The content of a <ins> element is usually underlined, while the content of a <del> element usually has a line through it.

HTML chapter-02/strikethrough.html

```
<p>Laptop computer:</p>
<p><s>Was $995</s></p>
<p>Now only $375</p>
```

RESULT

Laptop computer:

~~Was $995~~

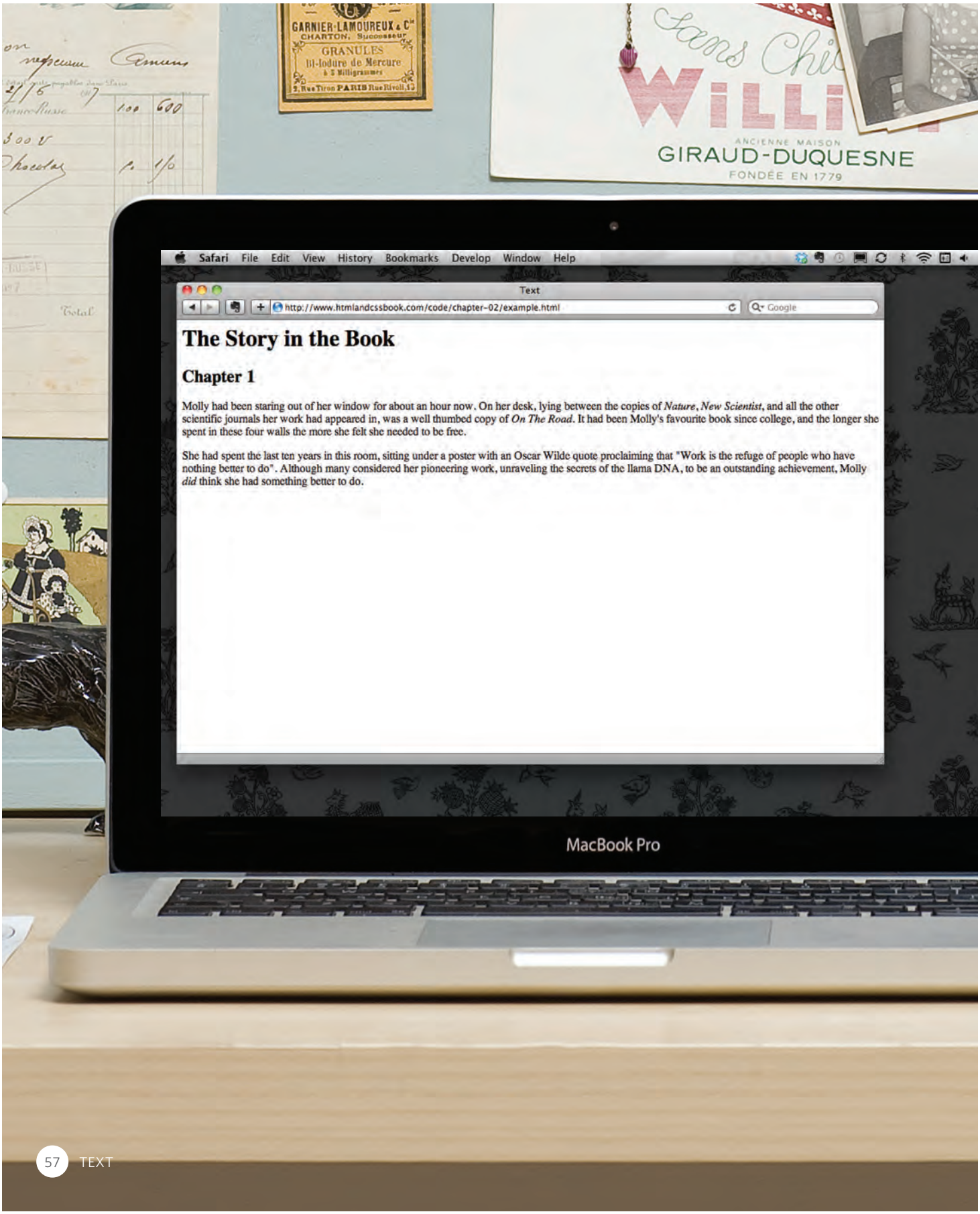
Now only $375

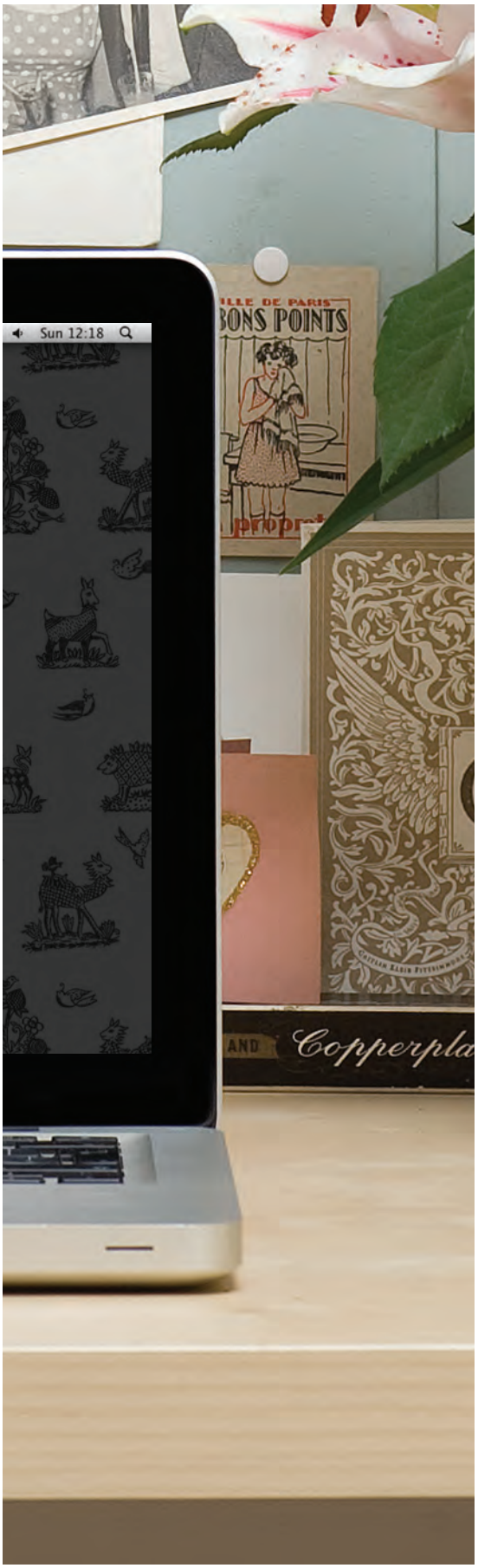
<s>

The <s> element indicates something that is no longer accurate or relevant (but that should not be deleted).

Visually the content of an <s> element will usually be displayed with a line through the center.

Older versions of HTML had a <u> element for content that was underlined, but this is being phased out.


GARNIER-LAMOUREUX & Cie
CHARTON, Successeur
GRANULES
Bi-Iodure de Mercure
Safari File Edit View History Bookmarks Develop Window Help
Text
http://www.htmlandcssbook.com/code/chapter-02/example.html
Google
The Story in the Book
Chapter 1
Molly had been staring out of her window for about an hour now. On her desk, lying between the copies of Nature, New Scientist, and all the other scientific journals her work had appeared in, was a well thumbed copy of On The Road. It had been Molly's favourite book since college, and the longer she spent in these four walls the more she felt she needed to be free.
She had spent the last ten years in this room, sitting under a poster with an Oscar Wilde quote proclaiming that "Work is the refuge of people who have nothing better to do". Although many considered her pioneering work, unraveling the secrets of the llama DNA, to be an outstanding achievement, Molly did think she had something better to do.
MacBook Pro



EXAMPLE
TEXT

This is a very simple HTML page that demonstrates text markup.

Structural markup includes elements such as <h1>, <h2>, and <p>. Semantic information is carried in elements such as <cite> and .

```
<html>
  <head>
    <title>Text</title>
  </head>
  <body>
    <h1>The Story in the Book</h1>
    <h2>Chapter 1</h2>
    <p>Molly had been staring out of her window for about
       an hour now. On her desk, lying between the copies
       of <i>Nature</i>, <i>New Scientist</i>, and all
       the other scientific journals her work had
       appeared in, was a well thumbed copy of <cite>On
       The Road</cite>. It had been Molly's favorite book
       since college, and the longer she spent in these
       four walls the more she felt she needed to be
       free.</p>
    <p>She had spent the last ten years in this room,
       sitting under a poster with an Oscar Wilde quote
       proclaiming that <q>Work is the refuge of
       people who have nothing better to do</q>. Although
       many considered her pioneering work, unraveling
       the secrets of the llama <abbr
       title="Deoxyribonucleic acid">DNA</abbr>, to be an
       outstanding achievement, Molly <em>did</em> think
       she had something better to do.</p>
  </body>
</html>
```

SUMMARY
TEXT

- HTML elements are used to describe the structure of the page (e.g. headings, subheadings, paragraphs).
- They also provide semantic information (e.g. where emphasis should be placed, the definition of any acronyms used, when given text is a quotation).

3

LISTS

- Numbered lists
- Bullet lists
- Definition lists

There are lots of occasions when we need to use lists. HTML provides us with three different types:

- **Ordered lists** are lists where each item in the list is numbered. For example, the list might be a set of steps for a recipe that must be performed in order, or a legal contract where each point needs to be identified by a section number.
- **Unordered lists** are lists that begin with a bullet point (rather than characters that indicate order).
- **Definition lists** are made up of a set of terms along with the definitions for each of those terms.


Safari File Edit View History Bookmarks Develop Window Help
Sun 12:18
Lists
http://www.htmlandcssbook.com/code/chapter-03/example.html
Scrambled Eggs
Eggs are one of my favorite foods. Here is a recipe for deliciously rich scrambled eggs.
Ingredients
2 eggs
1tbs butter
2tbs cream
Method
1. Melt butter in a frying pan over a medium heat
2. Gently mix the eggs and cream in a bowl
3. Once butter has melted add cream and eggs
4. Using a spatula fold the eggs from the edge of the pan to the center every 20 seconds (as if you are making an omelette)
5. When the eggs are still moist remove from the heat (it will continue to cook on the plate until served)
MacBook Pro

ORDERED LISTS

The ordered list is created with the `<ol>` element.

Each item in the list is placed between an opening `<li>` tag and a closing `</li>` tag. (The `li` stands for list item.)

Browsers indent lists by default.

Sometimes you may see a `type` attribute used with the `<ol>` element to specify the type of numbering (numbers, letters, roman numerals and so on). It is better to use the CSS `list-style-type` property covered on pages 333-335.

chapter-03/ordered-lists.html **HTML**

```
<ol>
  <li>Chop potatoes into quarters</li>
  <li>Simmer in salted water for 15-20
      minutes until tender</li>
  <li>Heat milk, butter and nutmeg</li>
  <li>Drain potatoes and mash</li>
  <li>Mix in the milk mixture</li>
</ol>
```

RESULT

1. Chop potatoes into quarters
2. Simmer in salted water for 15-20 minutes until tender
3. Heat milk, butter and nutmeg
4. Drain potatoes and mash
5. Mix in the milk mixture

UNORDERED LISTS

HTML chapter-03/unordered-lists.html

```
<ul>
  <li>1kg King Edward potatoes</li>
  <li>100ml milk</li>
  <li>50g salted butter</li>
  <li>Freshly grated nutmeg</li>
  <li>Salt and pepper to taste</li>
</ul>
```

RESULT

- 1kg King Edward potatoes
- 100ml milk
- 50g salted butter
- Freshly grated nutmeg
- Salt and pepper to taste

The unordered list is created with the <ul> element.

Each item in the list is placed between an opening <li> tag and a closing tag. (The li stands for list item.)

Browsers indent lists by default.

Sometimes you may see a type attribute used with the <ul> element to specify the type of bullet point (circles, squares, diamonds and so on). It is better to use the CSS list-style-type property covered on pages 333-335.

DEFINITION LISTS

<dl>

The definition list is created with the <dl> element and usually consists of a series of terms and their definitions.

Inside the <dl> element you will usually see pairs of <dt> and <dd> elements.

<dt>

This is used to contain the term being defined (the definition term).

<dd>

This is used to contain the definition.

Sometimes you might see a list where there are two terms used for the same definition or two different definitions for the same term.

chapter-03/definition-lists.html **HTML**

```
<dl>
  <dt>Sashimi</dt>
  <dd>Sliced raw fish that is served with
      condiments such as shredded daikon radish or
      ginger root, wasabi and soy sauce</dd>
  <dt>Scale</dt>
  <dd>A device used to accurately measure the
      weight of ingredients</dd>
  <dd>A technique by which the scales are removed
      from the skin of a fish</dd>
  <dt>Scamorze</dt>
  <dt>Scamorzo</dt>
  <dd>An Italian cheese usually made from whole
      cow's milk (although it was traditionally made
      from buffalo milk)</dd>
</dl>
```

RESULT

Sashimi
: Sliced raw fish that is served with condiments such as shredded daikon radish or ginger root, wasabi and soy sauce

Scale
: A device used to accurately measure the weight of ingredients
: A technique by which the scales are removed from the skin of a fish

Scamorze

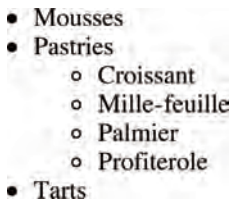
Scamorzo
: An Italian cheese usually made from whole cow's milk (although it was traditionally made from buffalo milk)

NESTED LISTS

HTML chapter-03/nested-lists.html

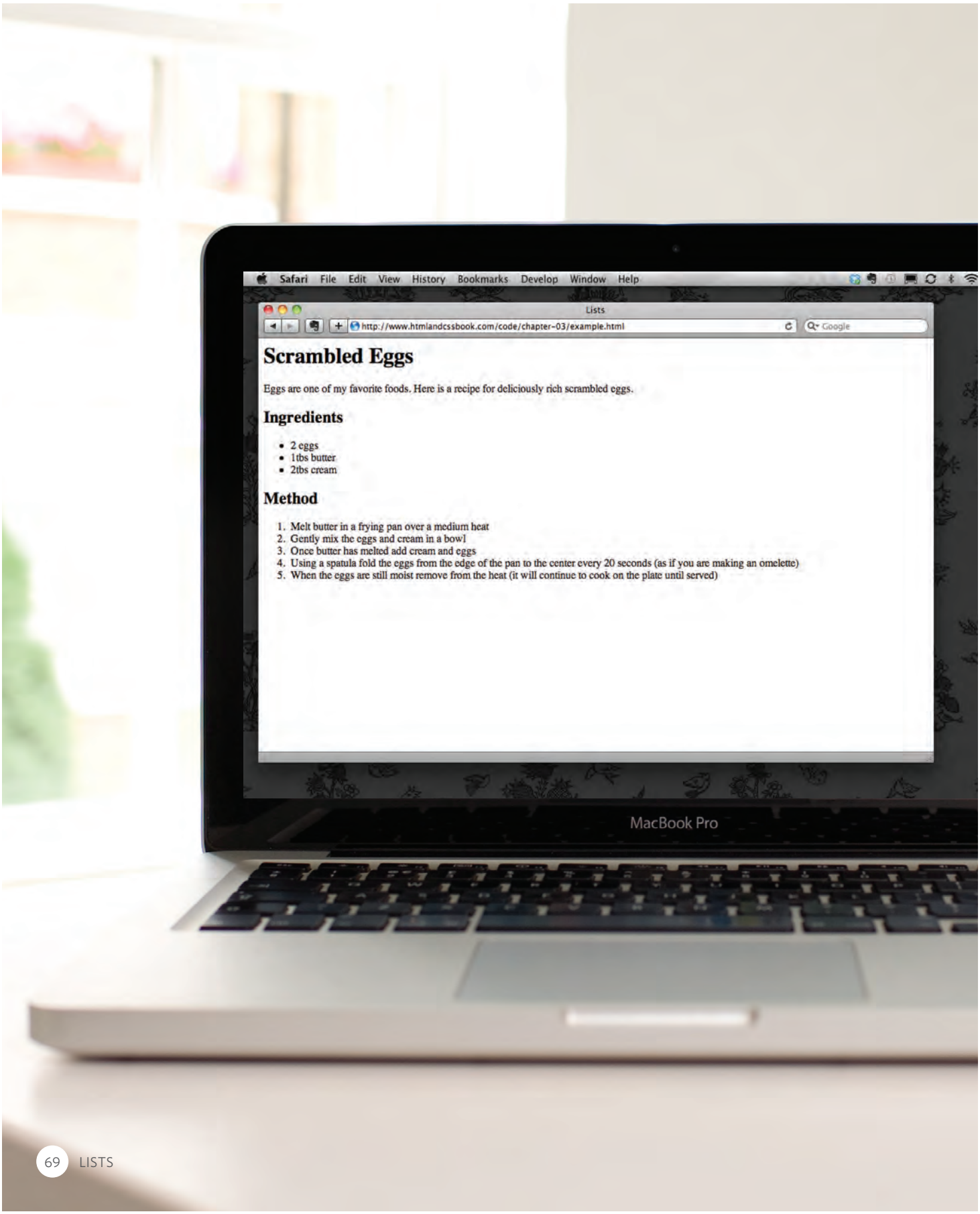
```
<ul>
  <li>Mousses</li>
  <li>Pastries
    <ul>
      <li>Croissant</li>
      <li>Mille-feuille</li>
      <li>Palmier</li>
      <li>Profiterole</li>
    </ul>
  </li>
  <li>Tarts</li>
</ul>
```

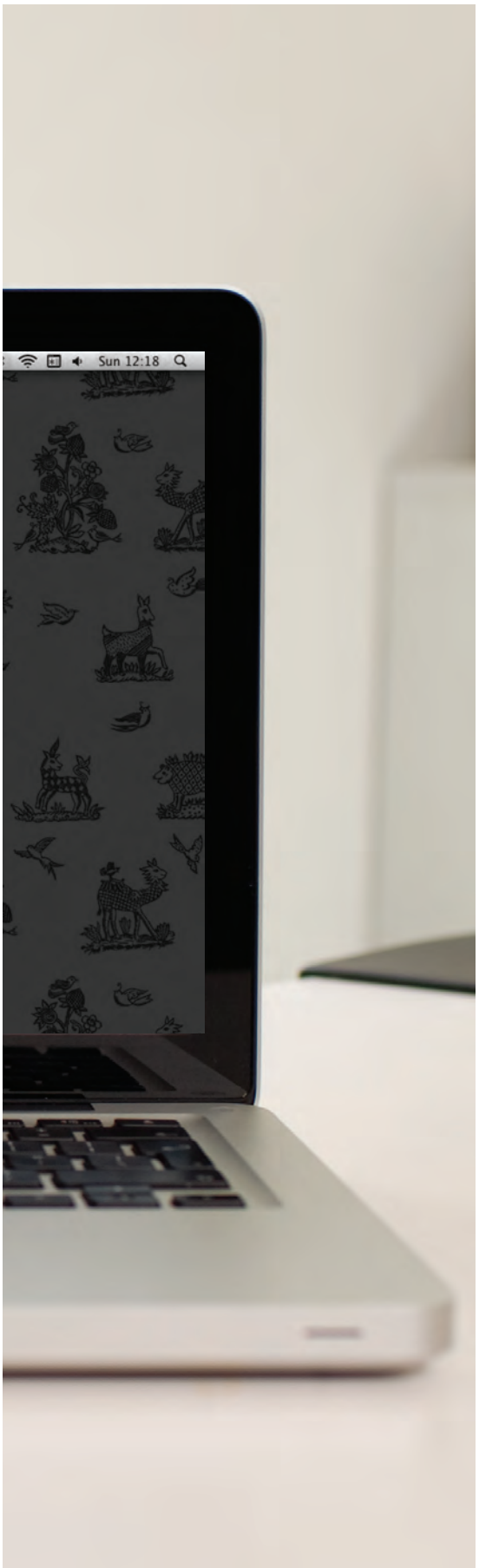
RESULT



You can put a second list inside an `<li>` element to create a sub-list or nested list.

Browsers display nested lists indented further than the parent list. In nested unordered lists, the browser will usually change the style of the bullet point too.


Safari File Edit View History Bookmarks Develop Window Help
Lists
http://www.htmlandcssbook.com/code/chapter-03/example.html
Google
Scrambled Eggs
Eggs are one of my favorite foods. Here is a recipe for deliciously rich scrambled eggs.
Ingredients
2 eggs
1tbs butter
2tbs cream
Method
1. Melt butter in a frying pan over a medium heat
2. Gently mix the eggs and cream in a bowl
3. Once butter has melted add cream and eggs
4. Using a spatula fold the eggs from the edge of the pan to the center every 20 seconds (as if you are making an omelette)
5. When the eggs are still moist remove from the heat (it will continue to cook on the plate until served)
MacBook Pro



EXAMPLE
LISTS

Here you can see a main heading followed by an introductory paragraph. An unordered list is used to outline the ingredients and an ordered list is used to describe the steps.

```
<html>
  <head>
    <title>Lists</title>
  </head>
  <body>
    <h1>Scrambled Eggs</h1>
    <p>Eggs are one of my favourite foods. Here is a
       recipe for deliciously rich scrambled eggs.</p>
    <h2>Ingredients</h2>
    <ul>
      <li>2 eggs</li>
      <li>1tbs butter</li>
      <li>2tbs cream</li>
    </ul>
    <h2>Method</h2>
    <ol>
      <li>Melt butter in a frying pan over a medium
          heat</li>
      <li>Gently mix the eggs and cream in a bowl</li>
      <li>Once butter has melted add cream and eggs</li>
      <li>Using a spatula fold the eggs from the edge of
          the pan to the center every 20 seconds (as if
          you are making an omelette)</li>
      <li>When the eggs are still moist remove from the
          heat (it will continue to cook on the plate
          until served)</li>
    </ol>
  </body>
</html>
```

SUMMARY
LISTS

- There are three types of HTML lists: ordered, unordered, and definition.
- Ordered lists use numbers.
- Unordered lists use bullets.
- Definition lists are used to define terminology.
- Lists can be nested inside one another.

4

LINKS

- Creating links between pages
- Linking to other sites
- Email links

Links are the defining feature of the web because they allow you to move from one web page to another — enabling the very idea of browsing or surfing.

You will commonly come across the following types of links:

- Links from one website to another
- Links from one page to another on the same website
- Links from one part of a web page to another part of the same page
- Links that open in a new browser window
- Links that start up your email program and address a new email to someone


PICCADILLY EXPRESS TRAINS
EALING COMMON
PARK ROYAL
SUDBURY TOWN
SOUTH HARROW
FROM 4TH JULY
sur ses deux oreilles
DUNLOP
Film Folk
Festival Diary
Here are some of the film festivals we will be attending this year.
Please contact us if you would like more information.
January
Sundance Film Festival
Park City, Utah, USA
20 - 30 January 2011
February
Tropfest
Sydney, Australia
20 February 2011
March
South by Southwest
Austin, Texas, USA
11 - 20 March 2011
April
London Independent Film Festival
London, UK
15 - 24 April 2011
May
Cannes International Film Festival
Cannes, France
11 - 22 May 2011
HOLGA 120N
Practical XML for the Web
Beginning
ASP.NET 2.0
Ajax
7 GRIDS AMBROSE / HARRIS
PROFESSIONAL
XML
2ndEdition
Beginning
Web Programming with HTML, XHTML, and CSS
2nd Edition
Ordering Disorder
INTRODUCING HTML5
LANDING PAGE OPTIMIZATION
FLICKR
LEGO

WRITING LINKS

Links are created using the `<a>` element. Users can click on anything between the opening `<a>` tag and the closing `</a>` tag. You specify which page you want to link to using the `href` attribute.

The text between the opening `<a>` tag and closing `</a>` tag is known as link text. Where possible, your link text should explain where visitors will be taken if they click on it (rather than just saying "click here"). Below you can see the link to IMDB that was created on the previous page.

Many people navigate websites by scanning the text for links. Clear link text can help visitors find what they want. This will give them a more positive impression of your site and may encourage them to visit it for longer. (It also helps people using screen reader software.)

To write good link text, you can think of words people might use when searching for the page that you are linking to. (For example, rather than write "places to stay" you could use something more specific such as "hotels in New York.")

LINKING TO OTHER SITES

<a>

Links are created using the `<a>` element which has an attribute called `href`. The value of the `href` attribute is the page that you want people to go to when they click on the link.

Users can click on anything that appears between the opening `<a>` tag and the closing `</a>` tag and will be taken to the page specified in the `href` attribute.

When you link to a different website, the value of the `href` attribute will be the full web address for the site, which is known as an **absolute** URL.

Browsers show links in blue with an underline by default.

chapter-04/linking-to-other-sites.html **HTML**

```
<p>Movie Reviews:
  <ul>
    <li><a href="http://www.empireonline.com">
        Empire</a></li>
    <li><a href="http://www.metacritic.com">
        Metacritic</a></li>
    <li><a href="http://www.rottentomatoes.com">
        Rotten Tomatoes</a></li>
    <li><a href="http://www.variety.com">
        Variety</a></li>
  </ul>
</p>
```

RESULT

Movie Reviews:

- Empire
- Metacritic
- Rotten Tomatoes
- Variety

ABSOLUTE URLS

URL stands for Uniform Resource Locator. Every web page has its own URL. This is the web address that you would type into a browser if you wanted to visit that specific page.

An absolute URL starts with the domain name for that site, and can be followed by the path to a specific page. If no page is specified, the site will display the homepage.

LINKING TO OTHER PAGES ON THE SAME SITE

HTML chapter-04/linking-to-other-pages.html

```
<p>
  <ul>
    <li><a href="index.html">Home</a></li>
    <li><a href="about-us.html">About</a></li>
    <li><a href="movies.html">Movies</a></li>
    <li><a href="contact.html">Contact</a></li>
  </ul>
</p>
```

RESULT

<a>

When you are linking to other pages within the same site, you do not need to specify the domain name in the URL. You can use a shorthand known as a **relative** URL.

If all the pages of the site are in the same folder, then the value of the `href` attribute is just the name of the file.

If you have different pages of a site in different folders, then you can use a slightly more complex syntax to indicate where the page is in relation to the current page. You will learn more about these on the pages 81-84.

If you look at the download code for each chapter, you will see that the `index.html` file contains links that use relative URLs.

RELATIVE URLS

When linking to other pages within the same site, you can use relative URLs. These are like a shorthand version of absolute URLs because you do not need to specify the domain name.

We will take a closer look at relative URLs on pages 83-84 as there are several helpful shortcuts you can use to write links to other pages on your own website.

Relative URLs help when building a site on your computer because you can create links between pages without having to set up your domain name or hosting.

DIRECTORY STRUCTURE

On larger websites it's a good idea to organize your code by placing the pages for each different section of the site into a new folder. Folders on a website are sometimes referred to as directories.

STRUCTURE

The diagram on the right shows the directory structure for a fictional entertainment listings website called ExampleArts.

The top-level folder is known as the **root** folder. (In this example, the root folder is called ***examplearts***.) The root folder contains all of the other files and folders for a website.

Each section of the site is placed in a separate folder; this helps organize the files.

RELATIONSHIPS

The relationship between files and folders on a website is described using the same terminology as a family tree.

In the diagram on the right, you can see some relationships have been drawn in.

The ***examplearts*** folder is a parent of the ***movies***, ***music*** and ***theater*** folders. And the the ***movies***, ***music*** and ***theater*** folders are children of the ***examplearts*** folder.

HOMEPAGES

The main homepage of a site written in HTML (and the homepages of each section in a child folder) is called ***index.html***.

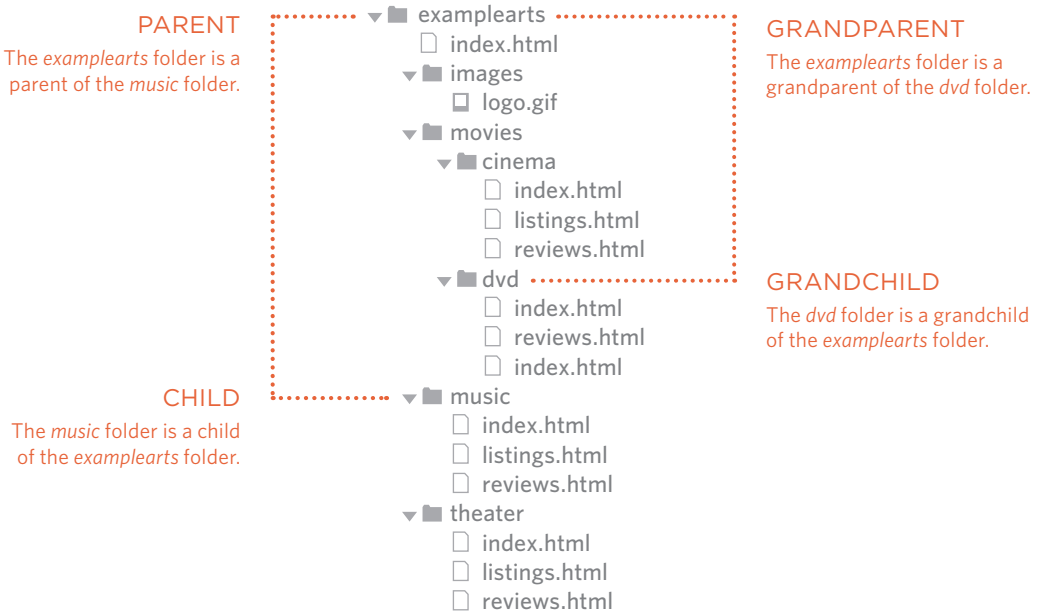
Web servers are usually set up to return the ***index.html*** file if no file name is specified.

Therefore, if you enter `examplearts.com` it will return `examplearts.com/index.html`, and `examplearts.com/music` will return `examplearts.com/music/index.html`.

If you are working with a content management system, blogging software, or an e-commerce system, you might not have individual files for each page of the website.

Instead, these systems often use one template file for each different type of page (such as news articles, blog posts, or products).

Editing the template file would change all of the pages that use that template. Do not change any code that is not HTML or you may break the page.



Every page and every image on a website has a **URL** (or Uniform Resource Locator). The URL is made up of the domain name followed by the **path** to that page or image.

The path to the homepage of this site is `www.examplearts.com/index.html`. The path to the logo for the site is `examplearts.com/images/logo.gif`.

You use URLs when linking to other web pages and when including images in your own site. On the next page, you will meet a shorthand way to link to files on your own site.

The root folder contains:

- A file called *index.html* which is the homepage for the entire site
- Individual folders for the movies, music and theatre sections of the site

Each sub-directory contains:

- A file called *index.html* which is the homepage for that section
- A reviews page called *reviews.html*
- A listings page called *listings.html* (except for the DVD section)

The movies section contains:

- A folder called *cinema*
- A folder called *DVD.*

RELATIVE URLS

Relative URLs can be used when linking to pages within your own website. They provide a shorthand way of telling the browser where to find your files.

When you are linking to a page on your own website, you do not need to specify the domain name. You can use **relative URLs** which are a shorthand way to tell the browser where a page is in relation to the current page.

This is especially helpful when creating a new website or learning about HTML because you can create links between pages when they are only on your personal computer (before you have got a domain name and uploaded them to the web).

Because you do not need to repeat the domain name in each link, they are also quicker to write.

If all of the files in your site are in one folder, you simply use the file name for that page.

If your site is organized into separate folders (or directories), you need to tell the browser how to get from the page it is *currently on* to the page that you are *linking to*.

If you link to the same page from two different pages you might, therefore, need to write two different relative URLs.

These links make use of the same terminology (borrowed from that of family trees) you met on the previous page which introduces directory structure.

RELATIVE LINK TYPE	EXAMPLE (from diagram on previous page)
SAME FOLDER To link to a file in the same folder, just use the file name. (Nothing else is needed.)	To link to music reviews from the music homepage: `<a href="reviews.html">Reviews</a>`
CHILD FOLDER For a child folder, use the name of the child folder, followed by a forward slash, then the file name.	To link to music listings from the homepage: `<a href="music/listings.html">Listings</a>`
GRANDCHILD FOLDER Use the name of the child folder, followed by a forward slash, then the name of the grandchild folder, followed by another forward slash, then the file name.	To link to DVD reviews from the homepage: `<a href="movies/dvd/reviews.html">` Reviews`</a>`
PARENT FOLDER Use `../` to indicate the folder above the current one, then follow it with the file name.	To link to the homepage from the music reviews: `<a href="../index.html">Home</a>`
GRANDPARENT FOLDER Repeat the `../` to indicate that you want to go up two folders (rather than one), then follow it with the file name.	To link to the homepage from the DVD reviews: `<a href="../../index.html">Home</a>`

When a website is live (that is, uploaded to a web server) you may see a couple of other techniques used that do not work when the files are on your local computer.

For example, you may see the name of a child folder without the name of a file. In this case the web server will usually try to show the homepage for that section.

A forward slash will return the homepage for the entire site, and a forward slash followed by a file name will return that file providing it is in the root directory.

EMAIL LINKS

mailto:

To create a link that starts up the user's email program and addresses an email to a specified email address, you use the `<a>` element. However, this time the value of the `href` attribute starts with `mailto:` and is followed by the email address you want the email to be sent to.

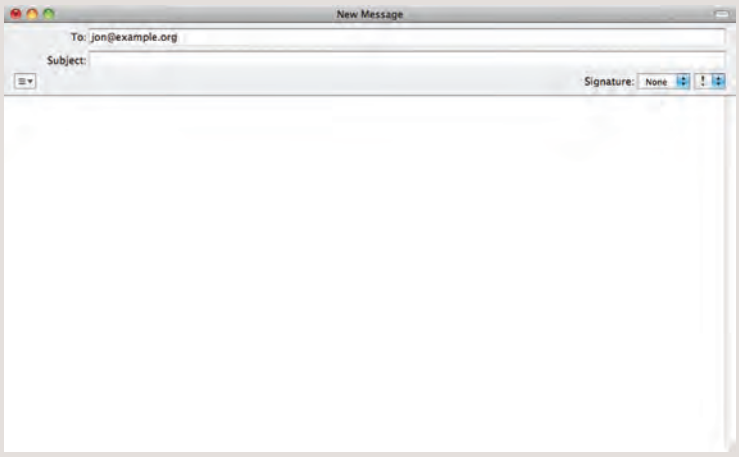
On the right you can see that an email link looks just like any other link but, when it is clicked on, the user's email program will open a new email message and address it to the person specified in the link.

chapter-04/email-links.html **HTML**

```
<a href="mailto:jon@example.org">Email Jon</a>
```

RESULT

Email Jon



OPENING LINKS IN A NEW WINDOW

HTML chapter-04/opening-links-in-a-new-window.html

```
<a href="http://www.imdb.com" target="_blank">
Internet Movie Database</a> (opens in new window)
```

RESULT

Internet Movie Database (opens in new window)

target

If you want a link to open in a new window, you can use the `target` attribute on the opening `<a>` tag. The value of this attribute should be `_blank`.

One of the most common reasons a web page author might want a link to be opened in a new window is if it points to another website. In such cases, they hope the user will return to the window containing their site after finishing looking at the other one.

Generally you should avoid opening links in a new window, but if you do, it is considered good practice to inform users that the link will open a new window before they click on it.

LINKING TO A SPECIFIC PART OF THE SAME PAGE

At the top of a long page you might want to add a list of contents that links to the corresponding sections lower down. Or you might want to add a link from part way down the page back to the top of it to save users from having to scroll back to the top.

Before you can link to a specific part of a page, you need to identify the points in the page that the link will go to. You do this using the `id` attribute (which can be used on every HTML element). You can see that the `<h1>` and `<h2>` elements in this example have been given `id` attributes that identify those sections of the page.

The value of the `id` attribute should start with a letter or an underscore (not a number or any other character) and, on a single page, no two `id` attributes should have the same value.

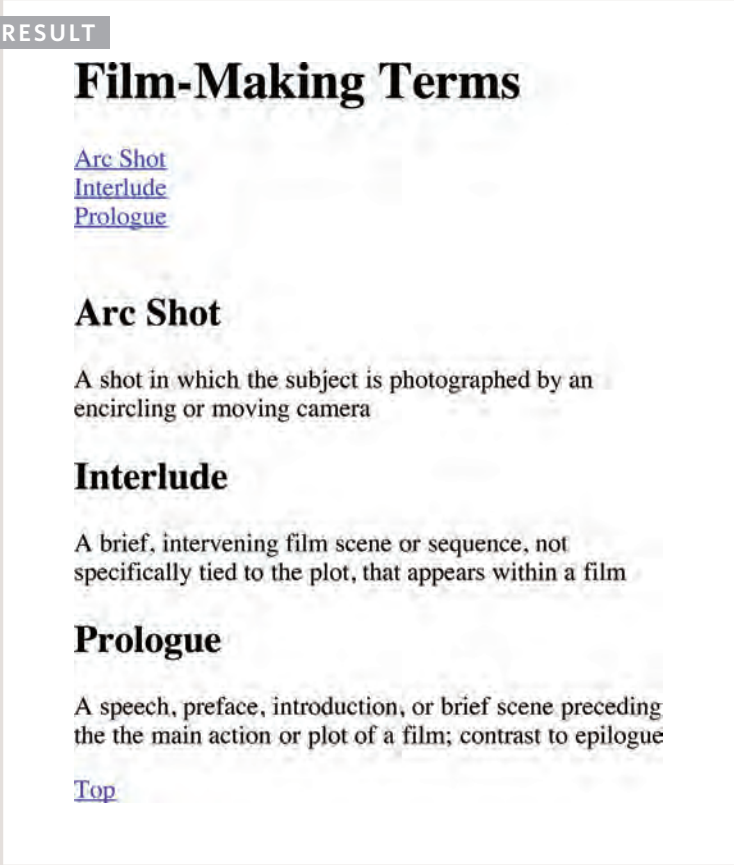
To link to an element that uses an `id` attribute you use the `<a>` element again, but the value of the `href` attribute starts with the # symbol, followed by the value of the `id` attribute of the element you want to link to. In this example, `<a href="#top">` links to the `<h1>` element at the top of the page whose `id` attribute has a value of `top`.

chapter-05/linking-to-a-specific-part.html **HTML**

```
<h1 id="top">Film-Making Terms</h1>
<a href="#arc_shot">Arc Shot</a><br />
<a href="#interlude">Interlude</a><br />
<a href="#prologue">Prologue</a><br /><br />
<h2 id="arc_shot">Arc Shot</h2>
<p>A shot in which the subject is photographed by an
   encircling or moving camera</p>
<h2 id="interlude">Interlude</h2>
<p>A brief, intervening film scene or sequence, not
   specifically tied to the plot, that appears
   within a film</p>
<h2 id="prologue">Prologue</h2>
<p>A speech, preface, introduction, or brief scene
   preceding the the main action or plot of a film;
   contrast to epilogue</p>
<p><a href="#top">Top</a></p>
```

LINKING TO A SPECIFIC PART OF ANOTHER PAGE

RESULT



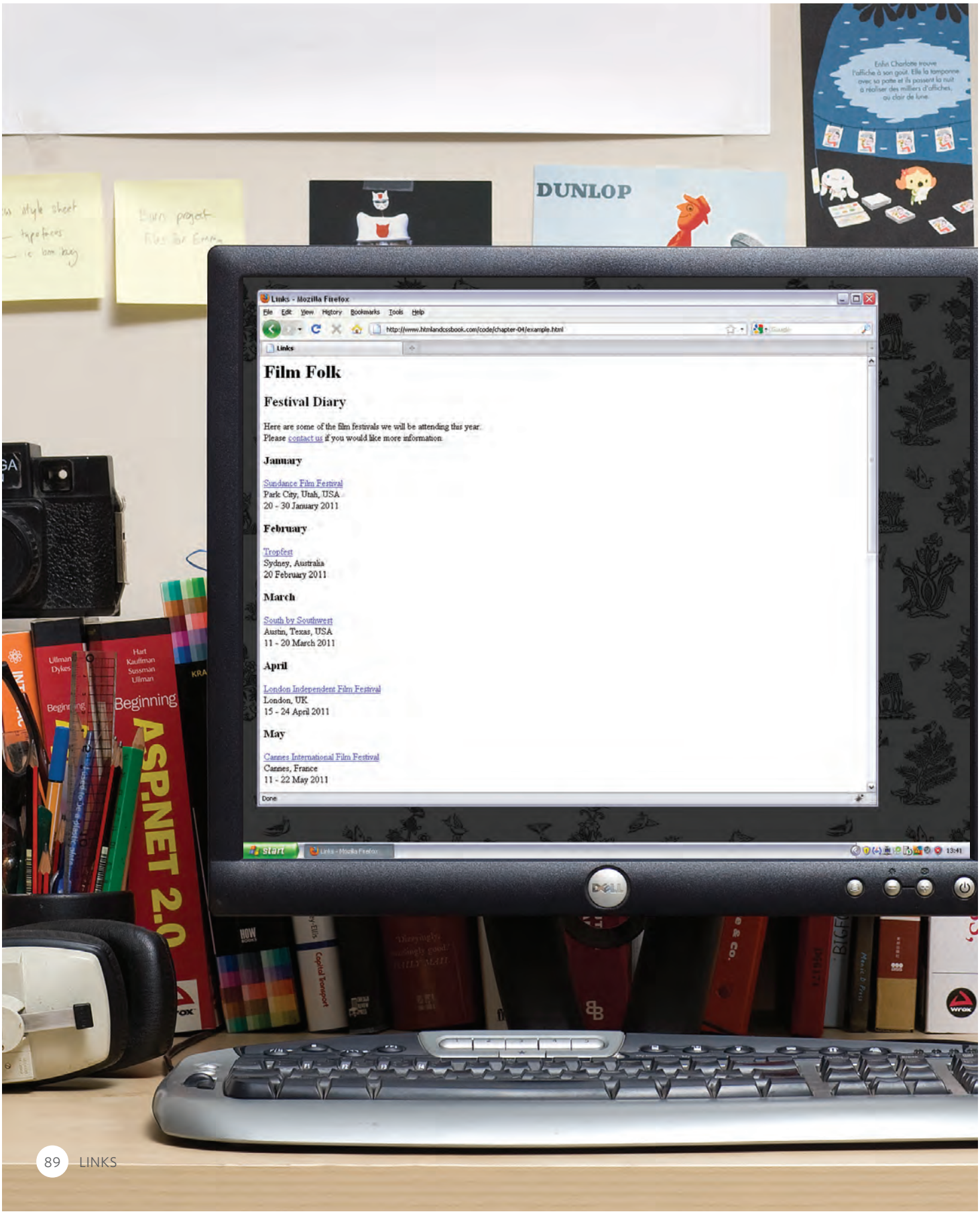
If you want to link to a specific part of a different page (whether on your own site or a different website) you can use a similar technique.

As long as the page you are linking to has `id` attributes that identify specific parts of the page, you can simply add the same syntax to the end of the link for that page.

Therefore, the `href` attribute will contain the address for the page (either an absolute URL or a relative URL), followed by the # symbol, followed by the value of the `id` attribute that is used on the element you are linking to.

For example, to link to the bottom of the homepage of the website that accompanies this book, you would write:

```
<a href="http:/www.
htmlandcssbookcom/
#bottom">
```


Links - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://www.htmlandcssbook.com/code/chapter-04/example.html
Links
Film Folk
Festival Diary
Here are some of the film festivals we will be attending this year.
Please contact us if you would like more information.
January
Sundance Film Festival
Park City, Utah, USA
20 - 30 January 2011
February
Tropfest
Sydney, Australia
20 February 2011
March
South by Southwest
Austin, Texas, USA
11 - 20 March 2011
April
London Independent Film Festival
London, UK
15 - 24 April 2011
May
Cannes International Film Festival
Cannes, France
11 - 22 May 2011
Done
DUNLOP
Beginning
ASP.NET 2.0

EXAMPLE
LINKS

This example is of a web page about film.

The <h1> element is used with an id attribute at the top of the page so that a link can be added to take readers from the bottom of the page to the top. There is an email link to allow readers to contact the author of the web page. There are also a number of links to qualified URLs. These link to various film festivals. Below this list is a link to a relative URL which is an "about" page that lives in the same directory.

```
<html>
  <head>
    <title>Links</title>
  </head>
  <body>
    <h1 id="top">Film Folk</h1>
    <h2>Festival Diary</h2>
    <p>Here are some of the film festivals we
       will be attending this year.<br />Please
       <a href="mailto:filmfolk@example.org">
       contact us</a> if you would like more
       information.</p>
    <h3>January</h3>
    <p><a href="http://www.sundance.org">
       Sundance Film Festival</a><br />
       Park City, Utah, USA<br />
       20 - 30 January 2011</p>
    <h3>February</h3>
    <p><a href="http://www.tropfest.com">
       Tropfest</a><br />
       Sydney, Australia<br />
       20 February 2011</p>
    <!-- additional content -->
    <p><a href="about.html">About Film Folk</a></p>
    <p><a href="#top">Top of page</a></p>
  </body>
</html>
```

SUMMARY
LINKS

- Links are created using the `<a>` element.
- The `<a>` element uses the `href` attribute to indicate the page you are linking to.
- If you are linking to a page within your own site, it is best to use relative links rather than qualified URLs.
- You can create links to open email programs with an email address in the "to" field.
- You can use the `id` attribute to target elements within a page that can be linked to.

5

IMAGES

- How to add images to pages
- Choosing the right format
- Optimizing images for the web

There are many reasons why you might want to add an image to a web page: you might want to include a logo, photograph, illustration, diagram, or chart.

There are several things to consider when selecting and preparing images for your site, but taking time to get them right will make it look more attractive and professional. In this chapter you will learn how to:

- Include an image in your web pages using HTML
- Pick which image format to use
- Show an image at the right size
- Optimize an image for use on the web to make pages load faster

You can also use CSS to include images in your pages using the `background-image` property, which you will meet on pages 413-420.


Safari File Edit View History Bookmarks Develop Window Help
Sun 12:18
Images
http://www.htmlandcssbook.com/code/chapter-05/example.html
Google
From A to Zucchini
chocolate islands
This recipe for individual chocolate cakes is so simple and so delectable!
More Recipes:

CHOOSING IMAGES FOR YOUR SITE

A picture can say a thousand words, and great images help make the difference between an average-looking site and a really engaging one.

Images can be used to set the tone for a site in less time than it takes to read a description. If you do not have photographs to use on your website, there are companies who sell **stock images**; these are images you pay to use (there is a list of stock photography websites below). Remember that all images are subject to copyright, and you can get in trouble for simply taking photographs from another website.

If you have a page that shows several images (such as product photographs or members of a team) then putting them on a simple, consistent background helps them look better as a group.

IMAGES SHOULD...

- ✔ Be relevant
- ✔ Convey information
- ✔ Convey the right mood
- ✔ Be instantly recognisable
- ✔ Fit the color palette

STOCK PHOTOS

`www.istockphoto.com`
`www.gettyimages.com`
`www.veer.com`
`www.sxc.hu`
`www.fotolia.com`

ONLINE EXTRA

We have provided an online gallery that helps you choose the right image for your website. You can find it in the tools section of the site accompanying this book.

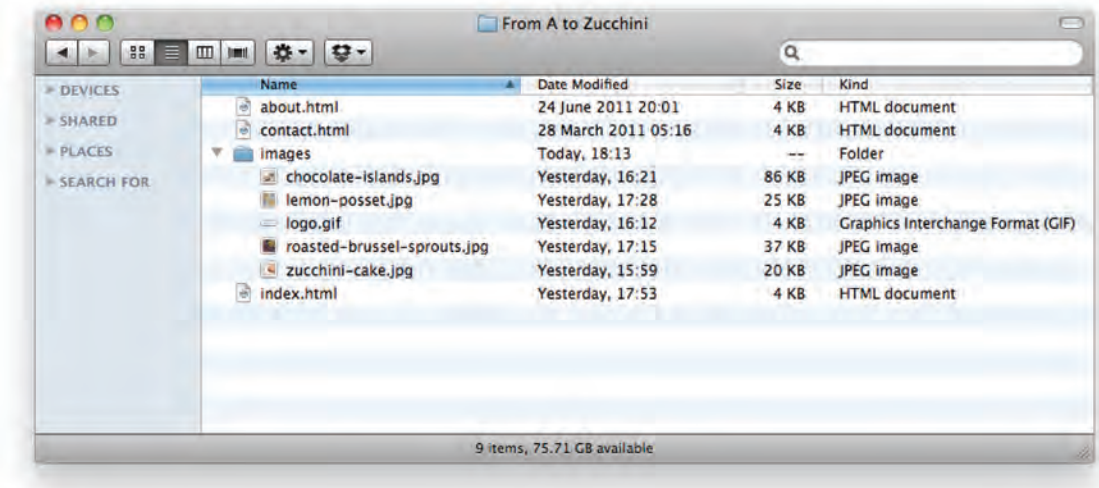
STORING IMAGES ON YOUR SITE

If you are building a site from scratch, it is good practice to create a folder for all of the images the site uses.

As a website grows, keeping images in a separate folder helps you understand how the site is organized. Here you can see an example of the files for a website; all of the images are stored in a folder called ***images***.

On a big site you might like to add subfolders inside the ***images*** folder. For example, images such as logos and buttons might sit in a folder called ***interface***, product photographs might sit in a page called ***products***, and images related to news might live in a folder called ***news***.

If you are using a content management system or blogging platform, there are usually tools built into the admin site that allow you to upload images, and the program will probably already have a separate folder for image files and any other uploads.



ADDING IMAGES

To add an image into the page you need to use an `<img>` element. This is an empty element (which means there is no closing tag). It must carry the following two attributes:

src

This tells the browser where it can find the image file. This will usually be a relative URL pointing to an image on your own site. (Here you can see that the images are in a child folder called ***images*** — relative URLs were covered on pages 83-84).

alt

This provides a text description of the image which describes the image if you cannot see it.

title

You can also use the `title` attribute with the `<img>` element to provide additional information about the image. Most browsers will display the content of this attribute in a tootip when the user hovers over the image.

chapter-05/adding-images.html **HTML**

```
<img src="images/quokka.jpg" alt="A family of
   quokka" title="The quokka is an Australian
   marsupial that is similar in size to the
   domestic cat." />
```

RESULT

The text used in the `alt` attribute is often referred to as **alt text**. It should give an accurate description of the image content so it can be understood by screen reader software (used by people with visual impairments) and search engines.

If the image is just to make a page look more attractive (and it has no meaning, such as a graphic dividing line), then the `alt` attribute should still be used but the quotes should be left empty.

HEIGHT & WIDTH OF IMAGES

HTML chapter-05/height-and-width-of-images.html

```
<img src="images/quokka.jpg" alt="A family of
   quokka" width="600" height="450" />
```

RESULT

You will also often see an `<img>` element use two other attributes that specify its size:

height

This specifies the height of the image in pixels.

width

This specifies the width of the image in pixels.

Images often take longer to load than the HTML code that makes up the rest of the page. It is, therefore, a good idea to specify the size of the image so that the browser can render the rest of the text on the page while leaving the right amount of space for the image that is still loading.

The size of images is increasingly being specified using CSS rather than HTML — see pages 409-410 for more information about this.

WHERE TO PLACE IMAGES IN YOUR CODE

Where an image is placed in the code will affect how it is displayed. Here are three examples of image placement that produce different results:

1: BEFORE A PARAGRAPH

The paragraph starts on a new line after the image.

2: INSIDE THE START OF A PARAGRAPH

The first row of text aligns with the bottom of the image.

3: IN THE MIDDLE OF A PARAGRAPH

The image is placed between the words of the paragraph that it appears in.

chapter-05/where-to-place-images.html **HTML**

```
<img src="images/bird.gif" alt="Bird" width="100"
   height="100" />
<p>There are around 10,000 living species of birds
   that inhabit different ecosystems from the
   Arctic to the Antarctic. Many species undertake
   long distance annual migrations, and many more
   perform shorter irregular journeys.</p>
<hr />
<p><img src="images/bird.gif" alt="Bird" width="100"
   height="100" />There are around 10,000 living
   species of birds that inhabit different
   ecosystems from the Arctic to the Antarctic. Many
   species undertake long distance annual
   migrations, and many more perform shorter
   irregular journeys.</p>
<hr />
<p>There are around 10,000 living species of birds
   that inhabit different ecosystems from the
   Arctic to the Antarctic.<img
   src="images/bird.gif" alt="Bird" width="100"
   height="100" />Many species undertake long
   distance annual migrations, and many more perform
   shorter irregular journeys.</p>
```

RESULT

There are around 10,000 living species of birds that inhabit different ecosystems from the Arctic to the Antarctic. Many species undertake long distance annual migrations, and many more perform shorter irregular journeys.

There are around 10,000 living species of birds that inhabit different ecosystems from the Arctic to the Antarctic. Many species undertake long distance annual migrations, and many more perform shorter irregular journeys.

There are around 10,000 living species of birds that inhabit different

ecosystems from the Arctic to the Antarctic. Many species undertake long distance annual migrations, and many more perform shorter irregular journeys.

Where you place the image in the code is important because browsers show HTML elements in one of two ways:

Block elements always appear on a new line. Examples of block elements include the `<h1>` and `<p>` elements.

If the `<img>` is followed by a block level element (such as a paragraph) then the block level element will sit on a new line after the imageas shown in the first example on this page.

Inline elements sit within a block level element and do not start on a new line. Examples of inline elements include the `<b>`, `<em>`, and `<img>` elements.

If the `<img>` element is inside a block level element, any text or other inline elements will flow around the image as shown in the second and third examples on this page.

Block and inline elements are discussed in greater depth on pages 185-186.

OLD CODE: ALIGNING IMAGES HORIZONTALLY

align

The `align` attribute was commonly used to indicate how the other parts of a page should flow around an image. It has been removed from HTML5 and new websites should use CSS to control the alignment of images (as you will see on pages 411-412).

I have discussed it here because you are likely to come across it if you look at older code, and because some visual editors still insert this attribute when you indicate how an image should be aligned.

The `align` attribute can take these horizontal values:

left

This aligns the image to the left (allowing text to flow around its right-hand side).

right

This aligns the image to the right (allowing text to flow around its left-hand side).

chapter-05/aligning-images-horizontally.html **HTML**

```
<p><img src="images/bird.gif" alt="Bird" width="100"
   height="100" align="left" />There are around
   10,000 living species of birds that inhabit
   different ecosystems from the Arctic to the
   Antarctic. Many species undertake long distance
   annual migrations, and many more perform shorter
   irregular journeys.</p>
<hr />
<p><img src="images/bird.gif" alt="Bird" width="100"
   height="100" align="right" />There are around
   10,000 living species of birds that inhabit
   different ecosystems from the Arctic to the
   Antarctic. Many species undertake long distance
   annual migrations, and many more perform shorter
   irregular journeys.</p>
```

RESULT

There are around 10,000 living species of birds that inhabit different ecosystems from the Arctic to the Antarctic. Many species undertake long distance annual migrations, and many more perform shorter irregular journeys.

There are around 10,000 living species of birds that inhabit different ecosystems from the Arctic to the Antarctic. Many species undertake long distance annual migrations, and many more perform shorter irregular journeys.

This looks a lot neater than having one line of text next to the image (as shown on the previous example).

When you give the `align` attribute a value of `left`, the image is placed on the left and text flows around it.

When you give the `align` attribute a value of `right`, the image is placed on the right and the text flows around it.

When text flows right up to the edge of an image it can make it harder to read. You will learn how to add a gap between text and images on pages 313-314 using the CSS `padding` and `margin` properties.

OLD CODE: ALIGNING IMAGES VERTICALLY

As you saw on the last page, the `align` attribute is no longer used in HTML5, but it is covered here because you may see it used in older websites and it is still used in the code created by some visual editors.

You can see how to use CSS to achieve the same effects on pages 285-286.

There are three values that the `align` attribute can take that control how the image should align vertically with the text that surrounds it:

top

This aligns the first line of the surrounding text with the top of the image.

middle

This aligns the first line of the surrounding text with the middle of the image.

bottom

This aligns the first line of the surrounding text with the bottom of the image.

chapter-05/aligning-images-vertically.html **HTML**

```
<p><img src="images/bird.gif" alt="Bird" width="100"
   height="100" align="top" />There are around
   10,000 living species of birds that inhabit
   different ecosystems from the Arctic to the
   Antarctic. Many species undertake long distance
   annual migrations, and many more perform shorter
   irregular journeys.</p>
<hr />
<p><img src="images/bird.gif" alt="Bird" width="100"
   height="100" align="middle" />There are around
   10,000 living species of birds that inhabit
   different ecosystems from the Arctic to the
   Antarctic. Many species undertake long distance
   annual migrations, and many more perform shorter
   irregular journeys.</p>
<hr />
<p><img src="images/bird.gif" alt="Bird" width="100"
   height="100" align="bottom" />There are around
   10,000 living species of birds that inhabit
   different ecosystems from the Arctic to the
   Antarctic. Many species undertake long distance
   annual migrations, and many more perform shorter
   irregular journeys.</p>
```

RESULT

There are around 10,000 living species of birds that inhabit different ecosystems from the Arctic to the Antarctic. Many species undertake long distance annual migrations, and many more perform shorter irregular journeys.

There are around 10,000 living species of birds that inhabit different ecosystems from the Arctic to the Antarctic. Many species undertake long distance annual migrations, and many more perform shorter irregular journeys.

There are around 10,000 living species of birds that inhabit different ecosystems from the Arctic to the Antarctic. Many species undertake long distance annual migrations, and many more perform shorter irregular journeys.

The value of `top` places the first line of text near the top of the image and subsequent lines of text appear under the image.

The value of `middle` places the first line of text near the vertical middle of the image and subsequent lines of text appear under the image.

The value of `bottom` places the first line of text near the bottom of the image and subsequent lines of text under the image.

When text flows right up to the edge of an image it can make it harder to read. You will learn how to add a gap between text and images on pages 313-314 using the CSS `padding` and `margin` properties.

If you would like all of the text to wrap arond the image (rather than just one line of text), you should use the CSS `float` property discussed on pages 370-372.

In older code, you may see the `align` attribute used with the values `left` or `right` to achieve the same effect (as described on the previous page), although its use is no longer recommended.

THREE RULES FOR CREATING IMAGES

There are three rules to remember when you are creating images for your website which are summarized below. We go into greater detail on each topic over the next nine pages.

1

SAVE IMAGES IN THE RIGHT FORMAT

Websites mainly use images in jpeg, gif, or png format. If you choose the wrong image format then your image might not look as sharp as it should and can make the web page slower to load.

2

SAVE IMAGES AT THE RIGHT SIZE

You should save the image at the same width and height it will appear on the website. If the image is smaller than the width or height that you have specified, the image can be distorted and stretched. If the image is larger than the width and height if you have specified, the image will take longer to display on the page.

3

USE THE CORRECT RESOLUTION

Computer screens are made up of dots known as pixels. Images used on the web are also made up of tiny dots. Resolution refers to the number of dots per inch, and most computer screens only show web pages at 72 pixels per inch. So saving images at a higher resolution results in images that are larger than necessary and take longer to download.

TOOLS TO EDIT & SAVE IMAGES

There are several tools you can use to edit and save images to ensure that they are the right size, format, and resolution.

The most popular tool amongst web professionals is **Adobe Photoshop**. (In fact, professional web designers often use this software to design entire sites.) The full version of Photoshop is expensive, but there is a cheaper version called Photoshop Elements which would suit the needs of most beginners.

OTHER SOFTWARE

Adobe Fireworks
Pixelmator
PaintShop Pro
Paint.net

ONLINE EDITORS

`www.photoshop.com`
`www.pixlr.com`
`www.splashup.com`
`www.ipiccy.com`

ONLINE EXTRA

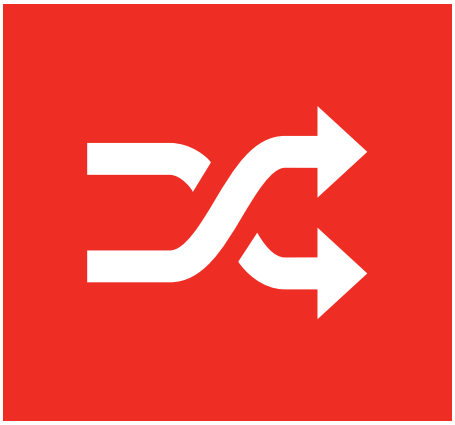
Watch videos that demonstrate how to resize images and save them in the correct format using both of these applications.

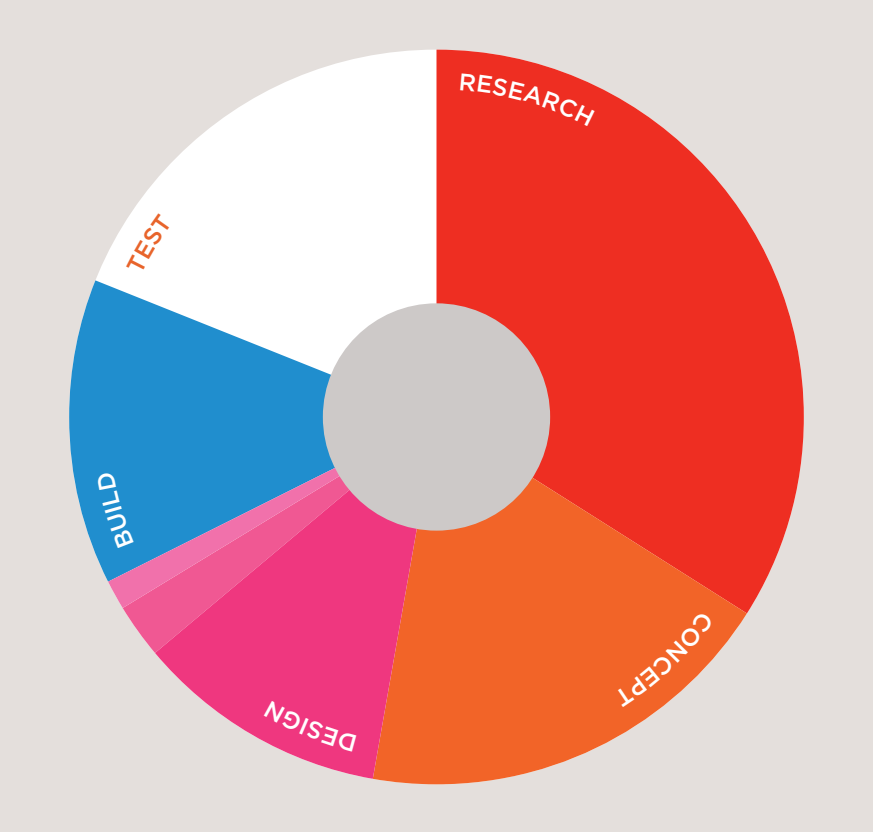
IMAGE FORMATS: JPEG

Whenever you have many different colors in a picture you should use a JPEG. A photograph that features snow or an overcast sky might look like it has large areas that are just white or gray, but the picture is usually made up of many different colors that are subtly different.

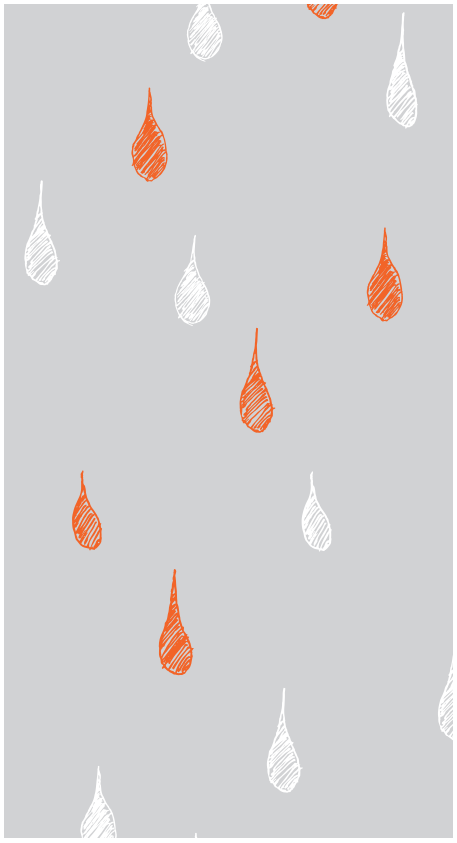

de
SOUK

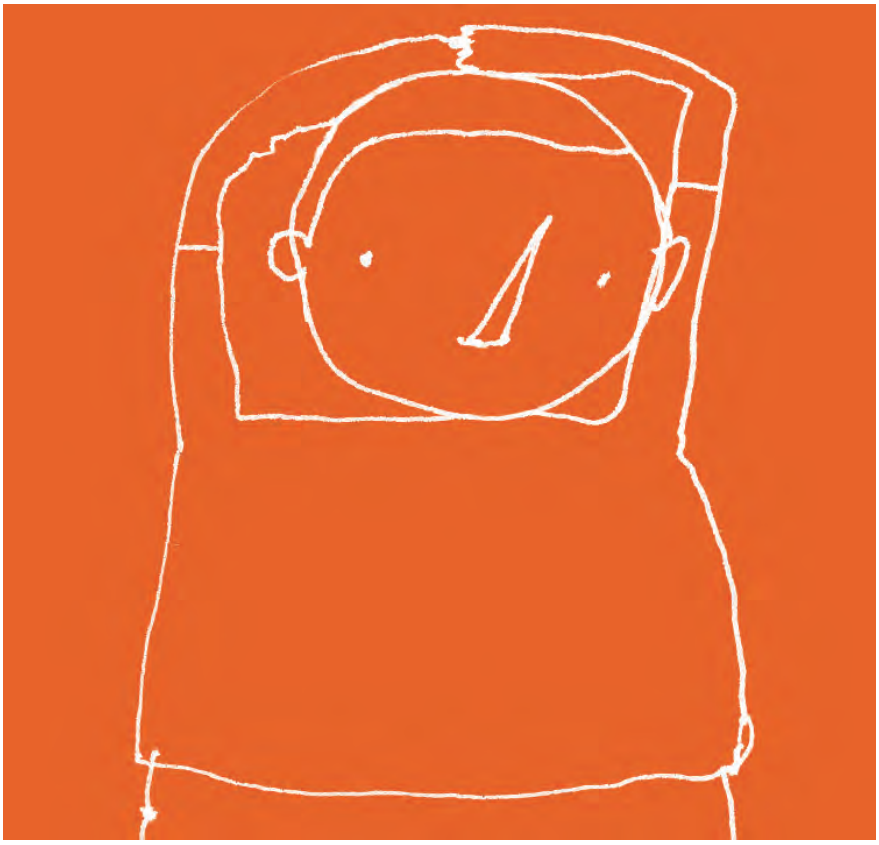
IMAGE FORMATS: GIF





Use GIF or PNG format when saving images with few colors or large areas of the same color.





When a picture has an area that is filled with exactly the same color, it is known as flat color. Logos, illustrations, and diagrams often use flat colors. (Note that photographs of snow, sky, or grass are not flat colors, they are made up of many subtly different shades of the same color and are not as suited to GIF or PNG format.)

IMAGE DIMENSIONS

The images you use on your website should be saved at the same width and height that you want them to appear on the page.

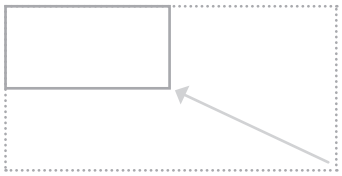
For example, if you have designed a page to include an image that is 300 pixels wide by 150 pixels tall, the image you use should be 300 x 150 pixels. You may need to use image editing tools to resize and crop the image. When sourcing images, it is important to understand how you can alter the dimensions of an image; imagine that you had designed a web page to include an image that is 300 pixels wide by 150 pixels tall:

ONLINE EXTRA

Visit the tools section of the website accompanying this book to watch a video guide to resizing images in Photoshop and GIMP.

REDUCING IMAGE SIZE

You can reduce the size of images to create a smaller version of the image.



Example: If your image is 600 pixels wide and 300 pixels tall, you can reduce the size of the image by 50%.

Result: This will create an image that is quicker to download.

INCREASING IMAGE SIZE

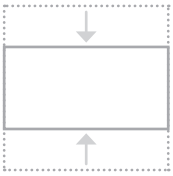
You can't increase the size of photos significantly without affecting the image quality.

Example: If your image is only 100 pixels wide by 50 pixels tall, increasing the size by 300% would result in poor quality.

Result: The image will look blurry or blocky.

CHANGING SHAPE

Only some images can be cropped without losing valuable information (see next page).



Example: If your image is 300 pixels square, you can remove parts of it, but in doing so you might lose valuable information.

Result: Only some images can be cropped and still make sense.

CROPPING IMAGES

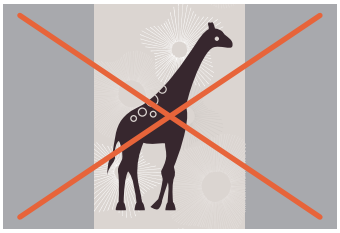
When cropping images it is important not to lose valuable information. It is best to source images that are the correct shape if possible.

PORTRAIT

Here you can see an illustration of a giraffe that is best suited to appearing in **portrait**.

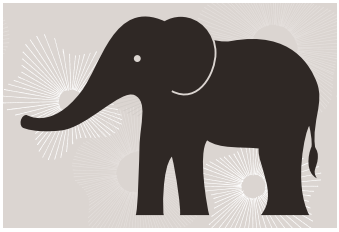
LANDSCAPE

If we **crop** this illustration to make it landscape we lose the head and feet.



If we **add extra space** to the left and right of the illustration the background is not continued.

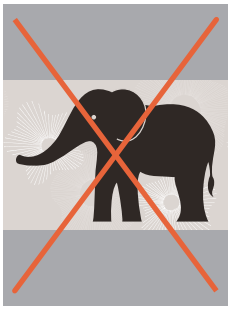
LANDSCAPE



Here you can see an illustration of an elephant that is best suited to appearing in **landscape**.

PORTRAIT

If we **crop** this illustration to make it portrait we lose the trunk and the hindquarters.



If we **add extra space** to the top and bottom of the illustration the background is not continued.

IMAGE RESOLUTION

Images created for the web should be saved at a resolution of 72 ppi. The higher the resolution of the image, the larger the size of the file.

JPGs, GIFs, and PNGs belong to a type of image format known as **bitmap**. They are made up of lots of miniature squares. The **resolution** of an image is the number of squares that fit within a 1 inch x 1 inch square area.

Images appearing on **computer** screens are made of tiny squares called **pixels**. A small segment of this photograph has been magnified to show how it is made up of pixels. The web browsers on most desktop computers display images at a resolution of **72** pixels per inch (ppi). Images in **print** materials (such as books and magazines) are made up of tiny circles called **dots**. These images are usually printed at a resolution of **300** dots per inch (dpi).

For this image:
JPEG at 300 dpi = 1,526kb
JPEG at 72 ppi = 368kb

Due to the fact that computer displays are capped at a resolution of 72 ppi, using images on the web with a higher resolution will not result in better image quality — only in larger file sizes, which will increase the time needed to load them and therefore slow down viewing of your web pages.

VECTOR IMAGES

Vector images differ from bitmap images and are resolution-independent. Vector images are commonly created in programs such as Adobe Illustrator.

When an image is a line drawing (such as a logo, illustration, or diagram), designers will often create it in vector format. Vector formatted images are very different to bitmap images.

Vector images are created by placing points on a grid, and drawing lines between those points. A color can then be added to "fill in" the lines that have been created.

The advantage of creating line drawings in vector format is that you can increase the dimensions of the image without affecting the quality of it.

The current method of using vector images for display on websites involves saving a bitmap version of the original vector image and using that.

Scalable Vector Graphics (SVG) are a relatively new format used to display vector images directly on the web (eliminating the need to create bitmap versions of them), however its use is not yet widespread.

ANIMATED GIFS

Animated GIFs show several frames of an image in sequence and therefore can be used to create simple animations.

Below you can see the individual frames that make up an animated GIF that shows an orange dot revolving around a circle — like the kind of animation you might see when a web page is loading.

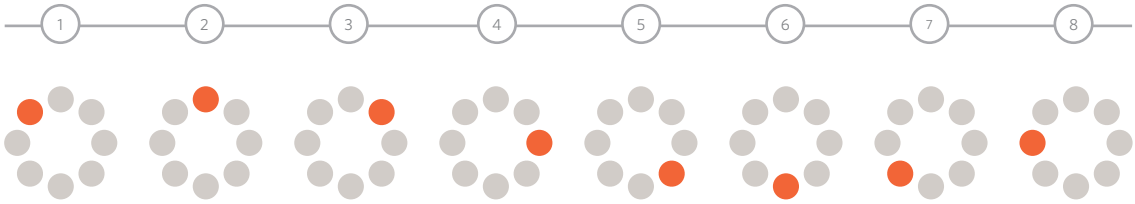
Some image editing applications such as Adobe Photoshop allow you to create animated GIFs. There are several tutorials about how to do this on the web. There are also several websites that allow you to upload the graphics for the individual frames and create the animated GIF for you.

IT IS IMPORTANT TO REMEMBER:

Each extra frame of the image increases the size of the file, and can therefore add to the time it takes for an image to download (and web users do not like waiting a long time for images to download).

Because GIFs are not an ideal format for displaying photographs, animated GIFs are really only suitable for simple illustrations.

Some designers frown on animated GIFs because they remember a lot of amateur web designers overusing them in the 1990's.



TRANSPARENCY

Creating an image that is partially transparent (or "see-through") for the web involves selecting one of two formats:

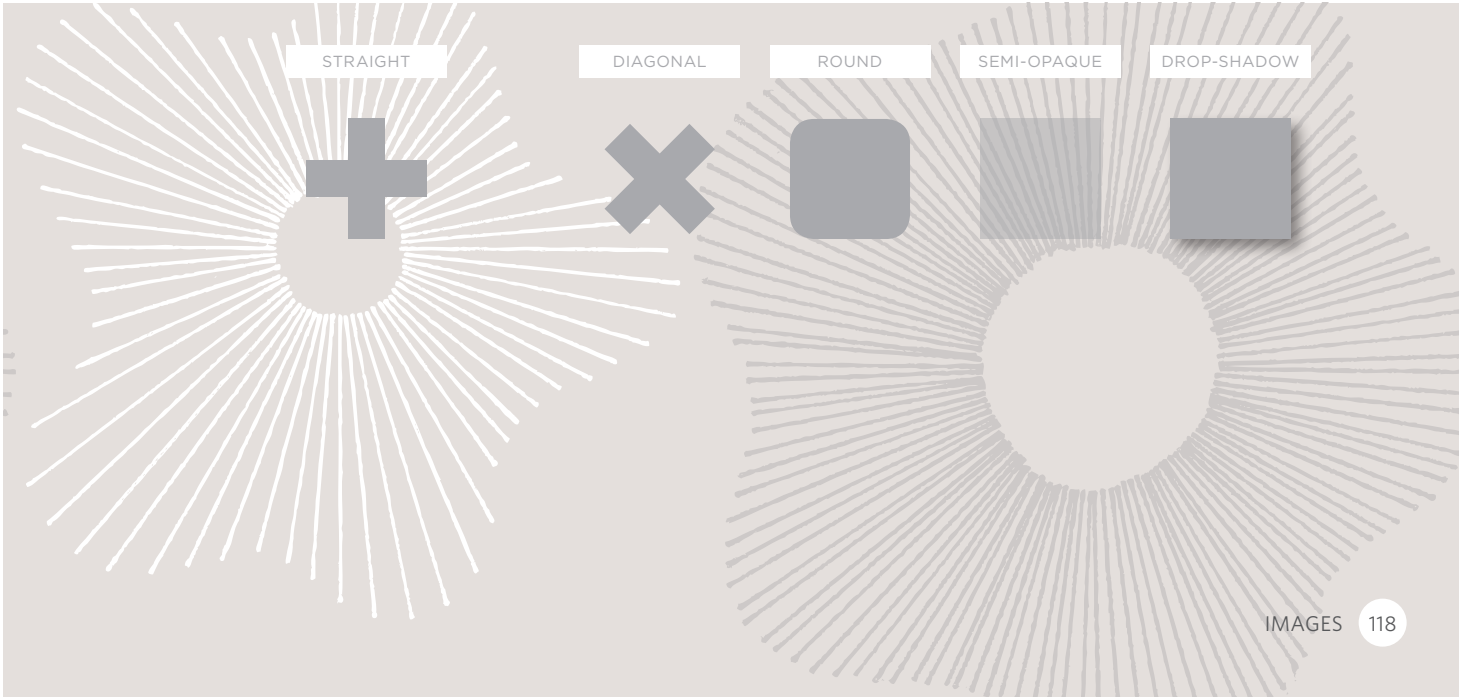
TRANSPARENT GIF

If the transparent part of the image has straight edges and it is 100% transparent (that is, not semi-opaque), you can save the image as a GIF (with the transparency option selected).

PNG

If the transparent part of the image has diagonal or rounded edges or if you want a semi-opaque transparency or a drop-shadow, then you will need to save it as a PNG.

Transparent PNGs are not fully supported in older browsers, most notably Internet Explorer 6 (IE6). There is some JavaScript you can use to get around this issue. The details of this script can be found in the tools section of the website accompanying this book.



EXAMINING IMAGES ON THE WEB

CHECKING THE SIZE OF IMAGES

If you are updating a website, you might need to check the size of an existing image before creating a new one to replace it. This can be achieved by right-clicking on the image and making a selection from the pop-up menu that appears. (Mac users will need to hold down the control key and click rather than right-click.)

DOWNLOADING IMAGES

If you want to download images from a website, you can do so by accessing the same pop-up menu. (Please remember however that all images online are subject to copyright and require explicit permission to reuse.)

On the left you can see how to check the size of images and how to download them using Safari. Below is a brief overview of what to select in the pop-up menu to perform these functions in various browsers.

CHROME

Size: ***Open Image in New Tab***
Size appears in new tab
Download: ***Save Image As***

FIREFOX

Size: ***View Image Info***
Size appears in pop-up window
Download: ***Save Image As***

INTERNET EXPLORER

Size: **Properties**
Size appears in pop-up window
Download: **Save Image**

SAFARI

Size: **Open Image in New Tab**
Size appears in title bar
Download: **Save Image As**

HTML5: FIGURE AND FIGURE CAPTION

HTML chapter-05/figure-and-figure-caption.html

```
<figure>
  <img src="images/otters.jpg" alt="Photograph of
       two sea otters floating in water">
  <br />
  <figcaption>Sea otters hold hands when they
       sleep so they don't drift away from each
       other.</figcaption>
</figure>
```

RESULT

Sea otters hold hands when they sleep so they don't drift away from each other.

<figure>

Images often come with captions. HTML5 has introduced a new `<figure>` element to contain images and their caption so that the two are associated.

You can have more than one image inside the `<figure>` element as long as they all share the same caption.

<figcaption>

The `<figcaption>` element has been added to HTML5 in order to allow web page authors to add a caption to an image.

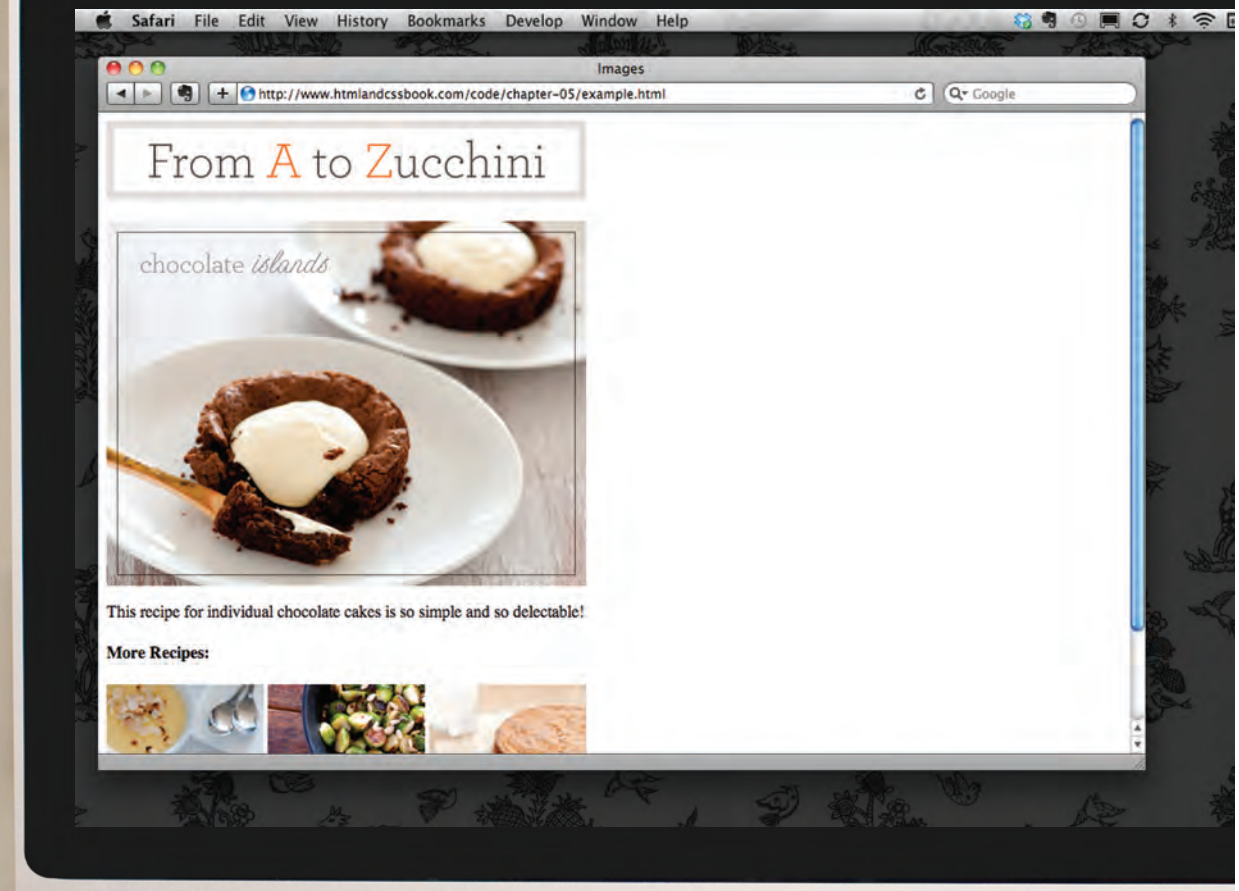
Before these elements were created there was no way to associate an `<img>` element with its caption.

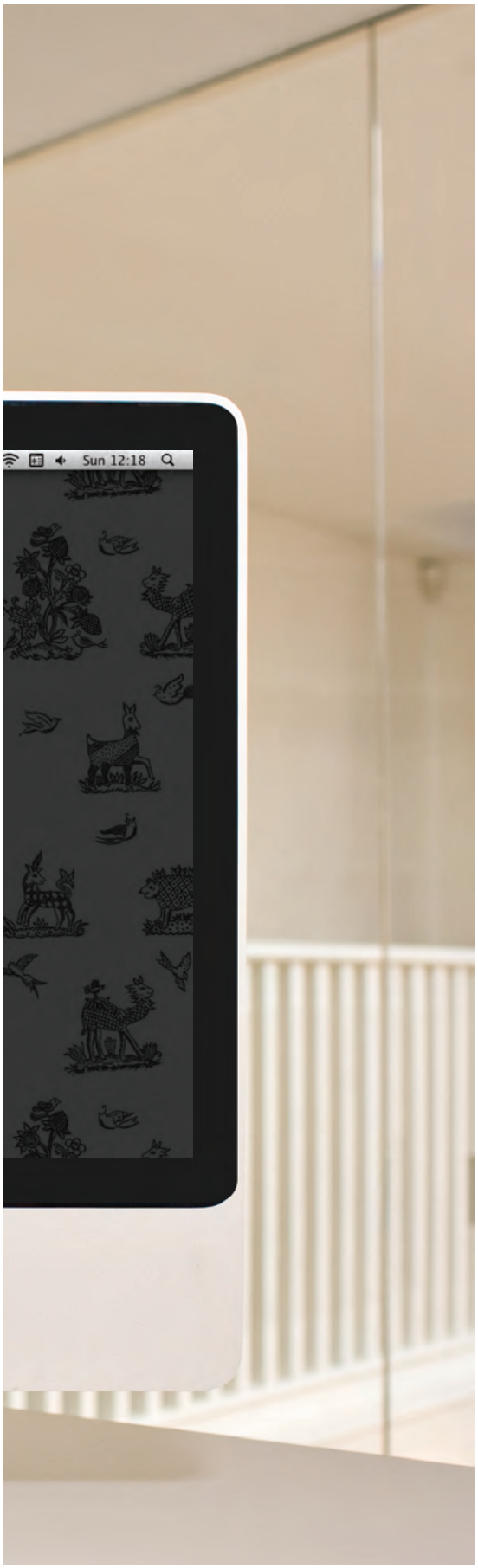
Older browsers that do not understand HTML5 elements simply ignore the new elements and display the content of them.

In this example, the logo is a GIF because it uses flat colors, while the photographs are JPEGs. The main photo is placed inside the HTML5 `<figure>` element and has its own caption.

The `alt` attribute on each image provides a description for those using screen readers and the `title` attribute provides additional information. (This is shown in the tooltip.)

This example does not use the `height`, `width`, or `align` attributes as these are being phased out and you are encouraged to use CSS properties instead.





EXAMPLE
IMAGES

```
<html>
  <head>
    <title>Images</title>
  </head>
  <body>
    <h1>
      <img src="images/logo.gif"
           alt="From A to Zucchini" />
    </h1>
    <figure>
      <img src="images/chocolate-islands.jpg"
           alt="Chocolate Islands"
           title="Chocolate Islands Individual Cakes" />
      <p>
        <figcaption>
          This recipe for individual chocolate
          cakes is so simple and so delectable!
        </figcaption>
      </p>
    </figure>
    <h4>More Recipes:</h4>
    <p>
      <img src="images/lemon-posset.jpg"
           alt="Lemon Posset"
           title="Lemon Posset Dessert" />
      <img src="images/roasted-brussel-sprouts.jpg"
           alt="Roasted Brussel Sprouts"
           title="Roasted Brussel Sprouts Side Dish" />
      <img src="images/zucchini-cake.jpg"
           alt="Zucchini Cake"
           title="Zucchini Cake No Frosting" />
    </p>
  </body>
</html>
```

SUMMARY
IMAGES

- The `<img>` element is used to add images to a web page.
- You must always specify a `src` attribute to indicate the source of an image and an `alt` attribute to describe the content of an image.
- You should save images at the size you will be using them on the web page and in the appropriate format.
- Photographs are best saved as JPEGs; illustrations or logos that use flat colors are better saved as GIFs.

6

TABLES

- How to create tables
- What information suits tables
- How to represent complex data in tables

There are several types of information that need to be displayed in a grid or table. For example: sports results, stock reports, train timetables.

When representing information in a table, you need to think in terms of a grid made up of rows and columns (a bit like a spreadsheet). In this chapter you will learn how to:

- Use the four key elements for creating tables
- Represent complex data using tables
- Add captions to tables


DUDLEY HOUSE
BOARDING & DAY SCHOOL
LETRASET ART SHEET AA21
AVEC
METAPOL
Pâte ou liquide
FER·FONTE·ACIER
retrouvent
le Poli du neuf
LION NOIR
Letratone.
LT1000
HIGH TEA
of Highgate
LETRASET ART SHEET AA22
You Rock!
Oil Painting Primer

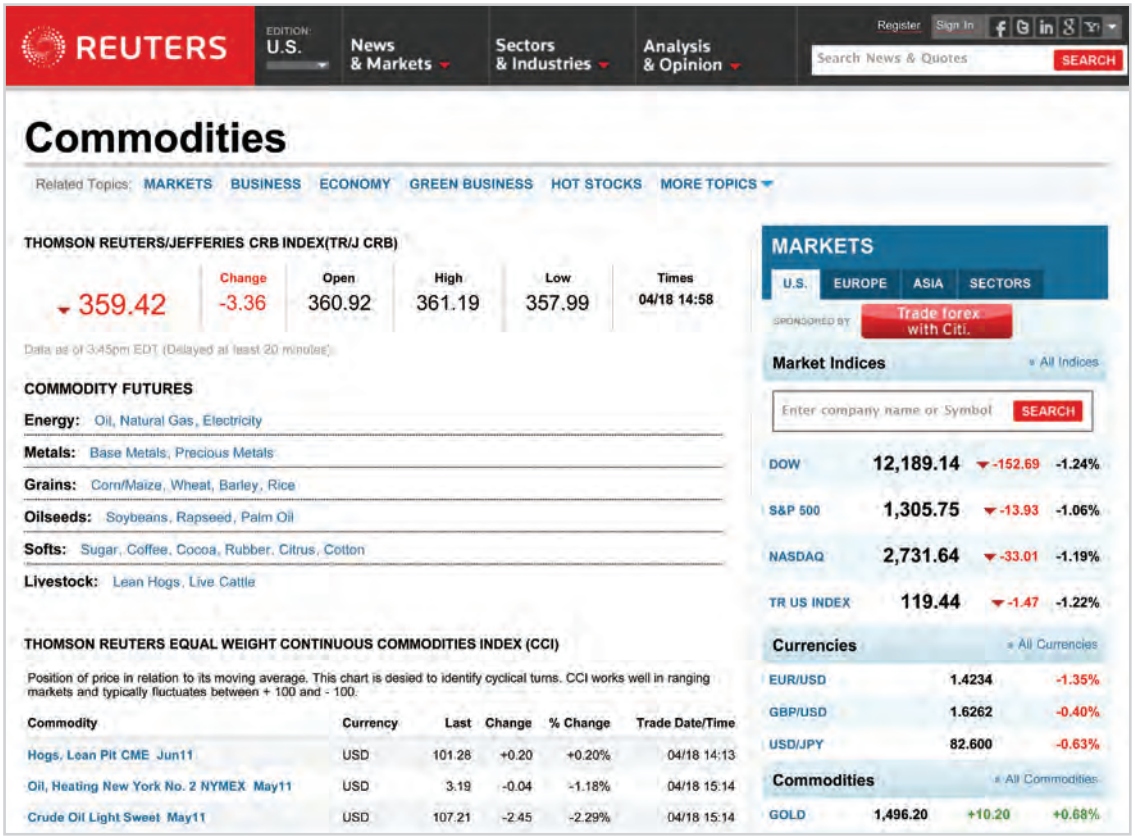

THE INDEPENDENT MONDAY 18 APRIL 2011
Shares & Markets Business
Using Oyster on National Rail
Pay as you go and Travelcard season tickets on Oyster can be used on all National Rail services London in the zones paid for except on:
Express
between Heathrow and
between
Adult fares
Pay as you go
Peak[1]*
Off-Peak[2]*
Zone 1 only £2.00 £1.50
Zones 1-2 £2.20 £1.70
Zones 1-3 £2.80 £2.00
Zones 1-4 £3.30 £2.30
Zones 1-5 £4.20 £2.60
Zones 1-6 £5.20 £3.20
Zones 2, 3, 4, 5 or 6 £1.50 £1.30
Zones 2-3, 3-4, 4-5, or 5-6 £1.90 £1.50
Zones 2-4, 3-5 or 4- 6 £2.30 £1.70
Zones 2-5 or 3-6 £2.90 £2.00
Zones 2-6 £3.40 £2.20
1*Peak Oyster single fares apply from 0630 to 0930 and from 1600 to 1900 Monday to Friday (excluding public holidays).
2*Off-Peak Oyster single fare applies at all other times.
15
No time restriction
When you need to make a claim, of course we hope your pet gets better quickly. But if they don't with M&S Pet Insurance there's no need to worry because there's no time limit on treatment. As long as you renew your policy and we can continue to offer cover, your vet's fees will be covered up to the limit stated. For full details of your cover have a look at your policy booklet.
Keep our claims helpline number, safe in case you need it, 0800 980 8750.
Cover you can depend on
Take a look at the table below to see just how much cover M&S Pet Insurance gives you. As well as cover for vets' fees, you'll also receive money towards the cost of advertising for your pet if it goes missing and a reward for its safe return. You're also covered for emergency kennel or cattery costs. With such comprehensive cover, you can rest assured that if the unexpected happens, M&S Pet Insurance is here to help.
Ask about the 5% discount offered on premiums for additional pets
Benefits
Up to
Premier
Standard
Veterinary fees for illness and injury £7,000 a year £4,000 a year
- Behaviourist £250 Not Covered
- Prescription food £200 Not Covered
- Complementary treatments £1,000 Not Covered
Limit for each illness/injury No Limit £1,000
Advertising for missing pet £1,250 £600
Rewards £750 £600
Stolen or missing pet £1,500 £600
Agency boarding/pet minder £750 total £600p/w
Farm injury £1,500 £600 total
Dental illness (up to age 9) £1,500 £600
Yes Yes
Yes £2,000,000 Yes
£3,000 £1,000,000
£2,000 Not Covered
Not Covered

WHAT'S A TABLE?

A table represents information in a grid format. Examples of tables include financial reports, TV schedules, and sports results.

Grids allow us to understand complex data by referencing information on two axes.

Each block in the grid is referred to as a **table cell**. In HTML a table is written out row by row.



BASIC TABLE STRUCTURE

<table>

The <table> element is used to create a table. The contents of the table are written out row by row.

<tr>

You indicate the start of each row using the opening <tr> tag. (The tr stands for table row.)

It is followed by one or more <td> elements (one for each cell in that row).

At the end of the row you use a closing </tr> tag.

<td>

Each cell of a table is represented using a <td> element. (The td stands for table data.)

At the end of each cell you use a closing </td> tag.

Some browsers automatically draw lines around the table and/or the individual cells. You will learn how to control the borders of tables using CSS on pages 309-312 and 337-340.

chapter-06/basic-table-structure.html **HTML**

```
<table>
  <tr>
    <td>15</td>
    <td>15</td>
    <td>30</td>
  </tr>
  <tr>
    <td>45</td>
    <td>60</td>
    <td>45</td>
  </tr>
  <tr>
    <td>60</td>
    <td>90</td>
    <td>90</td>
  </tr>
</table>
```

RESULT

15 15 30
45 60 45
60 90 90

TABLE HEADINGS

HTML chapter-06/table-headings.html

```
<table>
  <tr>
    <th></th>
    <th scope="col">Saturday</th>
    <th scope="col">Sunday</th>
  </tr>
  <tr>
    <th scope="row">Tickets sold:</th>
    <td>120</td>
    <td>135</td>
  </tr>
  <tr>
    <th scope="row">Total sales:</th>
    <td>$600</td>
    <td>$675</td>
  </tr>
</table>
```

RESULT

	Saturday	Sunday
Tickets sold:	120	135
Total sales:	$600	$675

<th>

The `<th>` element is used just like the `<td>` element but its purpose is to represent the heading for either a column or a row. (The `th` stands for table heading.)

Even if a cell has no content, you should still use a `<td>` or `<th>` element to represent the presence of an empty cell otherwise the table will not render correctly. (The first cell in the first row of this example shows an empty cell.)

Using `<th>` elements for headings helps people who use screen readers, improves the ability for search engines to index your pages, and also enables you to control the appearance of tables better when you start to use CSS.

You can use the `scope` attribute on the `<th>` element to indicate whether it is a heading for a column or a row. It can take the values: `row` to indicate a heading for a row or `col` to indicate a heading for a column.

Browsers usually display the content of a `<th>` element in bold and in the middle of the cell.

SPANNING COLUMNS

Sometimes you may need the entries in a table to stretch across more than one column.

The `colspan` attribute can be used on a `<th>` or `<td>` element and indicates how many columns that cell should run across.

In the example on the right you can see a timetable with five columns; the first column contains the heading for that row (the day), the remaining four represent one hour time slots.

If you look at the table cell that contains the words 'Geography' you will see that the value of the `colspan` attribute is 2, which indicates that the cell should run across two columns. In the third row, 'Gym' runs across three columns.

You can see that the second and third rows have fewer `<td>` elements than there are columns. This is because, when a cell extends across more than one column, the `<td>` or `<th>` cells that would have been in the place of the wider cells are not included in the code.

I added some CSS styles to this example so that you can see how the cells span more than one column. You will learn how to do this on pages 250, 337-340.

chapter-06/spanning-columns.html **HTML**

```
<table>
  <tr>
    <th></th>
    <th>9am</th>
    <th>10am</th>
    <th>11am</th>
    <th>12am</th>
  </tr>
  <tr>
    <th>Monday</th>
    <td colspan="2">Geography</td>
    <td>Math</td>
    <td>Art</td>
  </tr>
  <tr>
    <th>Tuesday</th>
    <td colspan="3">Gym</td>
    <td>Home Ec</td>
  </tr>
</table>
```

RESULT

	9am	10am	11am	12am
Monday	Geography		Math	Art
Tuesday	Gym			Home Ec

SPANNING ROWS

HTML chapter-06/spanning-rows.html

```
<table>
  <tr>
    <th></th>
    <th>ABC</th>
    <th>BBC</th>
    <th>CNN</th>
  </tr>
  <tr>
    <th>6pm - 7pm</th>
    <td rowspan="2">Movie</td>
    <td>Comedy</td>
    <td>News</td>
  </tr>
  <tr>
    <th>7pm - 8pm</th>
    <td>Sport</td>
    <td>Current Affairs</td>
  </tr>
</table>
```

RESULT

	ABC	BBC	CNN
6pm - 7pm	Movie	Comedy	News
7pm - 8pm		Sport	Current Affairs

You may also need entries in a table to stretch down across more than one row.

The `rowspan` attribute can be used on a `<th>` or `<td>` element to indicate how many rows a cell should span down the table.

In the example on the left you can see that ABC is showing a movie from 6pm - 8pm, whereas the BBC and CNN channels are both showing two programs during this time period (each of which lasts one hour).

If you look at the last `<tr>` element, it only contains three elements even though there are four columns in the result below. This is because the movie in the `<tr>` element above it uses the `rowspan` attribute to stretch down and take over the cell below.

I have added some CSS styles to this example so that you can see how the cells span more than one row. You will learn how to apply these CSS styles to tables on pages 250, 337-340.

LONG TABLES

There are three elements that help distinguish between the main content of the table and the first and last rows (which can contain different content).

These elements help people who use screen readers and also allow you to style these sections in a different manner than the rest of the table (as you will see when you learn about CSS).

<thead>

The headings of the table should sit inside the `<thead>` element.

<tbody>

The body should sit inside the `<tbody>` element.

<tfoot>

The footer belongs inside the `<tfoot>` element.

By default, browsers rarely treat the content of these elements any differently than other elements however designers often use CSS styles to change their appearance.

chapter-06/long-tables.html **HTML**

```
<table>
  <thead>
    <tr>
      <th>Date</th>
      <th>Income</th>
      <th>Expenditure</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <th>1st January</th>
      <td>250</td>
      <td>36</td>
    </tr>
    <tr>
      <th>2nd January</th>
      <td>285</td>
      <td>48</td>
    </tr>
    <!-- additional rows as above -->
    <tr>
      <th>31st January</th>
      <td>129</td>
      <td>64</td>
     </tr>
  </tbody>
  <tfoot>
    <tr>
      <td></td>
      <td>7824</td>
      <td>1241</td>
    </tr>
  </tfoot>
</table>
```

RESULT

Date	Income	Expenditure
1st January	250	36
2nd January	285	48
3rd January	260	42
4th January	290	38
5th January	310	115
6th January	168	14
7th January	226	20
8th January	253	37
9th January	294	33
10th January	216	46
11th January	244	29
12th January	297	32
13th January	328	86
14th January	215	38
15th January	254	30
16th January	256	27
17th January	311	68
18th January	212	39
19th January	234	36
20th January	221	43
21st January	259	38
22nd January	246	31
23rd January	248	17
24th January	229	45
25th January	263	34
26th January	258	41
27th January	283	22
28th January	256	30
29th January	278	47
30th January	251	15
31st January	129	64
	7824	1241

Some of the HTML editors that come in content management systems offer tools to help draw tables. If the first row of your table only contains `<th>` elements then you may find that the editor inserts a `<thead>` element automatically.

Part of the reason for having separate `<thead>` and `<tfoot>` elements is so that, if you have a table that is taller than the screen (or, if printed, longer than one page) then the browser can keep the header and footer visible whilst the contents of the table scroll. This is intended to make it easier for users to see which column the data is in (however this functionality is not implemented by default in any current browser).

I have added some CSS styles to this example so that you can see the contents of the `<thead>` and `<tfoot>` being treated differently than the rest of the rows. You will learn how to apply these CSS styles to tables on pages 309-312 and 337-340.

OLD CODE: WIDTH & SPACING

There are some outdated attributes which you should not use on new websites. You may, however, come across some of them when looking at older code, so I will mention them here. All of these attributes have been replaced by the use of CSS.

The `width` attribute was used on the opening `<table>` tag to indicate how wide that table should be and on some opening `<th>` and `<td>` tags to specify the width of individual cells. The value of this attribute is the width of the table or cell in pixels.

The columns in a table need to form a straight line, so you often only see the `width` attribute on the first row (and all subsequent rows would use that setting).

The opening `<table>` tag could also use the `cellpadding` attribute to add space inside each cell of the table, and the `cellspacing` attribute to create space between each cell of the table. The values for these attributes were given in pixels.

I added CSS styles to this example so that you can see the width of the table cells more clearly. If you want to control the width or spacing of tables and cells you should use CSS as shown on pages 303, 337-340.

chapter-06/width-and-spacing.html **HTML**

```
<table width="400" cellpadding="10" cellspacing="5">
  <tr>
    <th width="150"></th>
    <th>Withdrawn</th>
    <th>Credit</th>
    <th width="150">Balance</th>
  </tr>
  <tr>
    <th>January</th>
    <td>250.00</td>
    <td>660.50</td>
    <td>410.50</td>
  </tr>
  <tr>
    <th>February</th>
    <td>135.55</td>
    <td>895.20</td>
    <td>1170.15</td>
  </tr>
</table>
```

RESULT

	Withdrawn	Credit	Balance
January	250.00	660.50	410.50
February	135.55	895.20	1170.15

OLD CODE: BORDER & BACKGROUND

HTML chapter-06/border-and-background.html

```
<table border="2" bgcolor="#efefef">
  <tr>
    <th width="150"></th>
    <th>Withdrawn</th>
    <th>Credit</th>
    <th width="150" bgcolor="#cccccc">Balance</th>
  </tr>
  <tr>
    <th>January</th>
    <td>250.00</td>
    <td>660.50</td>
    <td bgcolor="#cccccc">410.50</td>
  </tr>
  <tr>
    <th>February</th>
    <td>135.55</td>
    <td>895.20</td>
    <td bgcolor="#cccccc">1170.15</td>
  </tr>
</table>
```

RESULT

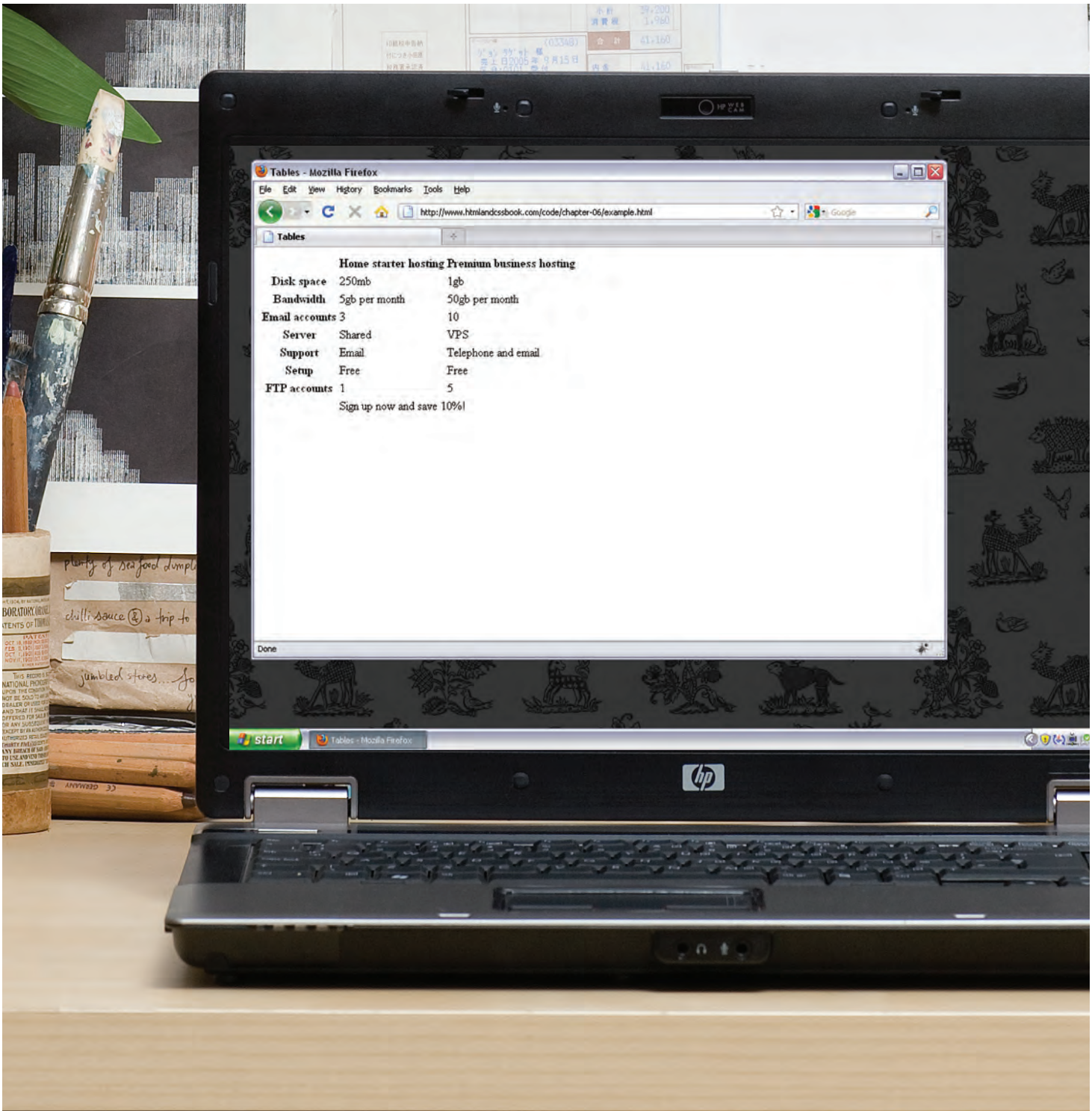
	Withdrawn	Credit	Balance
January	250.00	660.50	410.50
February	135.55	895.20	1170.15

The `border` attribute was used on both the `<table>` and `<td>` elements to indicate the width of the border in pixels.

The `bgcolor` attribute was used to indicate background colors of either the entire table or individual table cells. The value is usually a hex code (which we discuss on pages 249-252).

This example uses the HTML `border` and `bgcolor` attributes. No CSS attributes were utilized in this example.

When building a new website you should use CSS to control the appearance of the table rather than these attributes. They are only covered here because you may come across them if you look at the code of older websites.



	Home starter hosting	Premium business hosting
Disk space	250mb	1gb
Bandwidth	5gb per month	50gb per month
Email accounts	3	10
Server	Shared	VPS
Support	Email	Telephone and email
Setup	Free	Free
FTP accounts	1	5
	Sign up now and save 10%!	

This example shows a table for customers to compare website hosting packages. There are table headings in the first row and first column of the table.

The empty cell in the top left still has a `<th>` element to represent it. Each cell of the table must be accounted for by a `<th>` or `<td>` element. The `<th>` elements use the `scope` attribute to indicate whether they are headings for a row or column. The final row uses the `colspan` attribute to spread across all three columns.

EXAMPLE
TABLES

```
<html>
  <head>
    <title>Tables</title>
  </head>
  <body>
    <table>
      <thead>
        <tr>
          <th></th>
          <th scope="col">Home starter hosting</th>
          <th scope="col">Premium business hosting</th>
          </tr>
      </thead>
      <tbody>
        <tr>
          <th scope="row">Disk space</th>
          <td>250mb</td>
          <td>1gb</td>
        </tr>
        <tr>
          <th scope="row">Bandwidth</th>
          <td>5gb per month</td>
          <td>50gb per month</td>
        </tr>
        <!-- more rows like the two above here -->
      </tbody>
      <tfoot>
        <tr>
          <td></td>
          <td colspan="2">Sign up now and save 10%!</td>
        </tr>
      </tfoot>
    </table>
  </body>
</html>
```

SUMMARY
TABLES

- The `<table>` element is used to add tables to a web page.
- A table is drawn out row by row. Each row is created with the `<tr>` element.
- Inside each row there are a number of cells represented by the `<td>` element (or `<th>` if it is a header).
- You can make cells of a table span more than one row or column using the `rowspan` and `colspan` attributes.
- For long tables you can split the table into a `<thead>`, `<tbody>`, and `<tfoot>`.

7

FORMS

- How to collect information from visitors
- Different kinds of form controls
- New HTML5 form controls

Traditionally, the term 'form' has referred to a printed document that contains spaces for you to fill in information.

HTML borrows the concept of a form to refer to different elements that allow you to collect information from visitors to your site.

Whether you are adding a simple search box to your website or you need to create more complicated insurance applications, HTML forms give you a set of elements to collect data from your users. In this chapter you will learn:

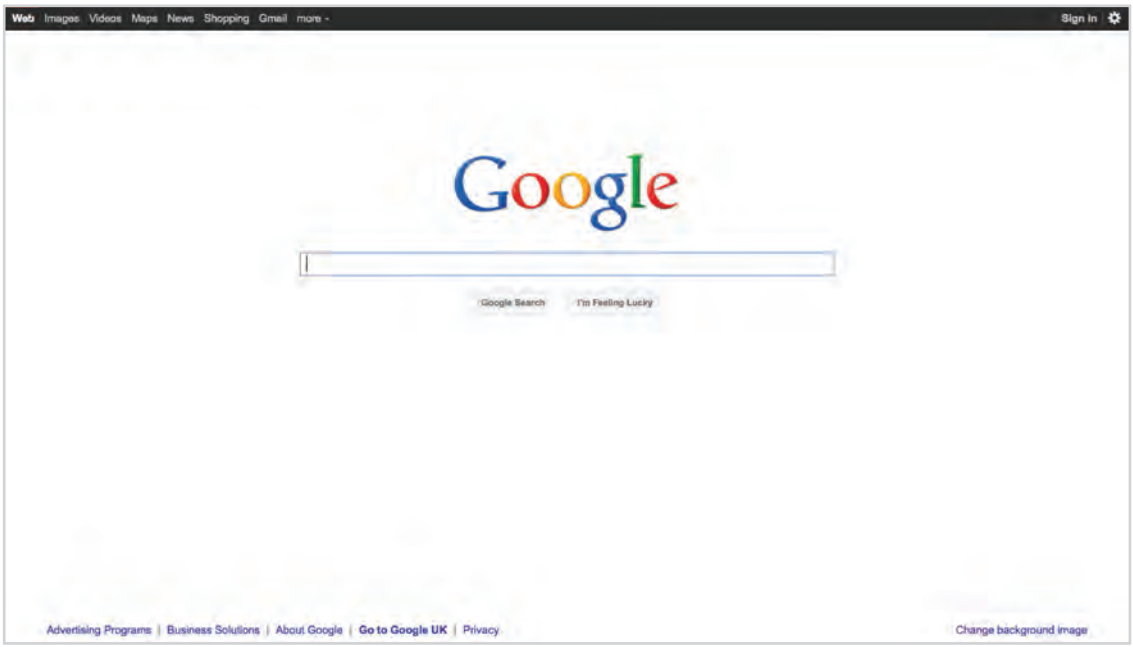
- How to create a form on your website
- The different tools for collecting data
- New HTML5 form controls


Safari File Edit View History Bookmarks Develop Window Help
Sun 12:19
Forms
http://www.htmlandcssbook.com/code/chapter-07/example.html
Your Details:
Name:
Email:
Your Review:
How did you hear about us? Google
Would you visit again?
Yes No Maybe
Comments:
Sign me up for email updates
Submit review
MacBook Pro

WHY FORMS?

The best known form on the web is probably the search box that sits right in the middle of Google's homepage.

In addition to enabling users to search, forms also allow users to perform other functions online. You will see forms when registering as a member of a website, when shopping online, and when signing up for newsletters or mailing lists.



FORM CONTROLS

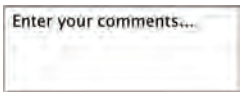
There are several types of form controls that you can use to collect information from visitors to your site.

ADDING TEXT:

Text input (single-line)
Used for a single line of text such as email addresses and names.

Password input
Like a single line text box but it masks the characters entered.

Text area (multi-line)
For longer areas of text, such as messages and comments.



MAKING CHOICES:

Radio buttons
For use when a user must select one of a number of options.

Checkboxes
When a user can select and unselect one or more options.

Drop-down boxes
When a user must pick one of a number of options from a list.

SUBMITTING FORMS:

Submit buttons
To submit data from your form to another web page.

Image buttons
Similar to submit buttons but they allow you to use an image.

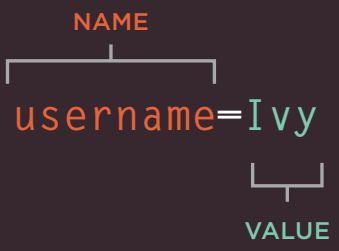
UPLOADING FILES:

File upload
Allows users to upload files (e.g. images) to a website.

HOW FORMS WORK

A user fills in a form and then presses a button to submit the information to the server.

A form may have several form controls, each gathering different information. The server needs to know which piece of inputted data corresponds with which form element.



To differentiate between various pieces of inputted data, information is sent from the browser to the server using name/value pairs. In this example, the form asks for the visitor's username and also for their favorite jazz musician. The name/value pairs sent to the server are:

`username=Ivy`

If the form control allows the user to enter text, then the value of the form control is whatever the user has typed in.

`vote=Herbie`

If the form control allows you to choose from a fixed set of answers (e.g. radio buttons, checkboxes or a drop down list), the web page author will add code that gives each option an automatic value.

You should never change the name of a form control in a page unless you know that the code on the server will understand this new value.

FORM STRUCTURE

<form>

Form controls live inside a <form> element. This element should always carry the action attribute and will usually have a method and id attribute too.

chapter-07/form-structure.html HTML

```
<form action="http://www.example.com/subscribe.php"
  method="get">
  <p>This is where the form controls will appear.
    </p>
</form>
```

RESULT

This is where the form controls will appear.

action

Every <form> element requires an action attribute. Its value is the URL for the page on the server that will receive the information in the form when it is submitted.

method

Forms can be sent using one of two methods: get or post.

With the get method, the values from the form are added to the end of the URL specified in the action attribute. The get method is ideal for:

- short forms (such as search boxes)
- when you are just retrieving data from the web server (not sending information that should be added to or deleted from a database)

With the post method the values are sent in what are known as HTTP headers. As a rule of thumb you should use the post method if your form:

- allows users to upload a file
- is very long
- contains sensitive data (e.g. passwords)
- adds information to, or deletes information from, a database

If the method attribute is not used, the form data will be sent using the get method.

id

We look at the id attribute on page 183, but the value is used to identify the form distinctly from other elements on the page (and is often used by scripts — such as those that check you have entered information into fields that require values).

TEXT INPUT

HTML chapter-07/text-input.html

```
<form action="http://www.example.com/login.php">
  <p>Username:
    <input type="text" name="username" size="15"
      maxlength="30" />
  </p>
</form>
```

RESULT

size

The `size` attribute should not be used on new forms. It was used in older forms to indicate the width of the text input (measured by the number of characters that would be seen).

For example, a value of 3 would create a box wide enough to display three characters (although a user could enter more characters if they desired).

In any new forms you write, CSS should be used to control the width of form elements. The `size` attribute is only mentioned here because you may come across it when looking at older code.

<input>

The `<input>` element is used to create several different form controls. The value of the `type` attribute determines what kind of input they will be creating.

type="text"

When the `type` attribute has a value of `text`, it creates a single-line text input.

name

When users enter information into a form, the server needs to know which form control each piece of data was entered into. (For example, in a login form, the server needs to know what has been entered as the username and what has been given as the password.) Therefore, each form control requires a `name` attribute. The value of this attribute identifies the form control and is sent along with the information they enter to the server.

maxlength

You can use the `maxlength` attribute to limit the number of characters a user may enter into the text field. Its value is the number of characters they may enter. For example, if you were asking for a year, the `maxlength` attribute could have a value of 4.

PASSWORD INPUT

<input>

type="password"

When the `type` attribute has a value of `password` it creates a text box that acts just like a single-line text input, except the characters are blocked out. They are hidden in this way so that if someone is looking over the user's shoulder, they cannot see sensitive data such as passwords.

name

The `name` attribute indicates the name of the password input, which is sent to the server with the password the user enters.

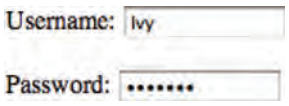
size, maxlength

It can also carry the `size` and `maxlength` attributes like the the single-line text input.

chapter-07/password-input.html **HTML**

```
<form action="http://www.example.com/login.php">
  <p>Username:
    <input type="text" name="username" size="15"
      maxlength="30" />
  </p>
  <p>Password:
    <input type="password" name="password" size="15"
      maxlength="30" />
  </p>
</form>
```

RESULT



Although the password is hidden on the screen, this does not mean that the data in a password control is sent securely to the server. You should never use these for sending sensitive data such as credit card numbers.

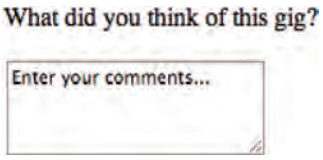
For full security, the server needs to be set up to communicate with users' browsers using Secure Sockets Layer (SSL). The topic of SSL is beyond the scope of this book, however there are links to learn more about it on the accompanying website.

TEXT AREA

HTML chapter-07/textarea.html

```
<form action="http://www.example.com/comments.php">
  <p>What did you think of this gig?</p>
  <textarea name="comments" cols="20" rows="4">Enter
    your comments...</textarea>
</form>
```

RESULT



If you are creating a new form, you should use CSS to control the width and height of a `<textarea>`. However, if you are looking at older code, you may see the `cols` and `rows` attributes used with this element.

The `cols` attribute indicates how wide the text area should be (measured in numbers of characters). The `rows` attribute indicates how many rows the text area should take up vertically.

<textarea>

The `<textarea>` element is used to create a mutli-line text input. Unlike other input elements this is not an empty element. It should therefore have an opening and a closing tag.

Any text that appears between the opening `<textarea>` and closing `</textarea>` tags will appear in the text box when the page loads.

If the user does not delete any text between these tags, this message will get sent to the server along with whatever the user has typed. (Some sites use JavaScript to clear this information when the user clicks in the text area.)

RADIO BUTTON

<input>

type="radio"

Radio buttons allow users to pick just one of a number of options.

name

The `name` attribute is sent to the server with the value of the option the user selects. When a question provides users with options for answers in the form of radio buttons, the value of the `name` attribute should be the same for all of the radio buttons used to answer that question.

value

The `value` attribute indicates the value that is sent to the server for the selected option. The value of each of the buttons in a group should be different (so that the server knows which option the user has selected).

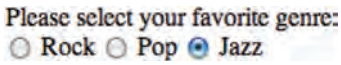
checked

The `checked` attribute can be used to indicate which value (if any) should be selected when the page loads. The value of this attribute is `checked`. Only one radio button in a group should use this attribute.

chapter-07/radio-button.html **HTML**

```
<form action="http://www.example.com/profile.php">
  <p>Please select your favorite genre:
    <br />
    <input type="radio" name="genre" value="rock"
      checked="checked" /> Rock
    <input type="radio" name="genre" value="pop" />
      Pop
    <input type="radio" name="genre" value="jazz" />
      Jazz
  </p>
</form>
```

RESULT



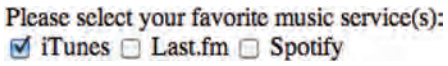
Please note: Once a radio button has been selected it cannot be deselected. The user can only select a different option. If you are only allowing the user one option and want them to be able to deselect it (for example if they are indicating they agree to terms and conditions), you should use a checkbox instead.

CHECKBOX

HTML chapter-07/checkbox.html

```
<form action="http://www.example.com/profile.php">
  <p>Please select your favorite music service(s):
    <br />
    <input type="checkbox" name="service"
      value="itunes" checked="checked" /> iTunes
    <input type="checkbox" name="service"
      value="lastfm" /> Last.fm
    <input type="checkbox" name="service"
      value="spotify" /> Spotify
  </p>
</form>
```

RESULT



<input>

type="checkbox"

Checkboxes allow users to select (and unselect) one or more options in answer to a question.

name

The `name` attribute is sent to the server with the value of the option(s) the user selects. When a question provides users with options for answers in the form of checkboxes, the value of the `name` attribute should be the same for all of the buttons that answer that question.

value

The `value` attribute indicates the value sent to the server if this checkbox is checked.

checked

The `checked` attribute indicates that this box should be checked when the page loads. If used, its value should be `checked`.

DROP DOWN LIST BOX

<select>

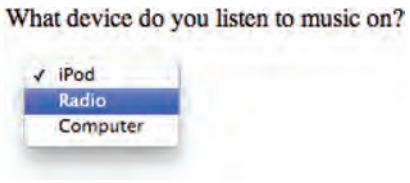
A drop down list box (also known as a select box) allows users to select one option from a drop down list.

The `<select>` element is used to create a drop down list box. It contains two or more `<option>` elements.

chapter-07/drop-down-list-box.html **HTML**

```
<form action="http://www.example.com/profile.php">
  <p>What device do you listen to music on?</p>
  <select name="devices">
    <option value="ipod">iPod</option>
    <option value="radio">Radio</option>
    <option value="computer">Computer</option>
  </select>
</form>
```

RESULT



name

The `name` attribute indicates the name of the form control being sent to the server, along with the value the user selected.

<option>

The `<option>` element is used to specify the options that the user can select from. The words between the opening `<option>` and closing `</option>` tags will be shown to the user in the drop down box.

value

The `<option>` element uses the `value` attribute to indicate the value that is sent to the server along with the name of the control if this option is selected.

selected

The `selected` attribute can be used to indicate the option that should be selected when the page loads. The value of this attribute should be `selected`.

If this attribute is not used, the first option will be shown when the page loads. If the user does not select an option, then the first item will be sent to the server as the value for this control.

The function of the drop down list box is similar to that of the radio buttons (in that only one option can be selected). There are two key factors in choosing which to use:

1. If users need to see all options at a glance, radio buttons are better suited.

2. If there is a very long list of options (such as a list of countries), drop down list boxes work better.

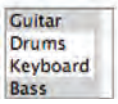
MULTIPLE SELECT BOX

HTML chapter-07/multiple-select-box.html

```
<form action="http://www.example.com/profile.php">
  <p>Do you play any of the following instruments?
    (You can select more than one option by holding
    down control on a PC or command key on a Mac
    while selecting different options.)</p>
  <select name="instruments" size="3"
    multiple="multiple">
    <option value="guitar" selected="selected">
      Guitar</option>
    <option value="drums">Drums</option>
    <option value="keyboard"
      selected="selected">Keyboard</option>
    <option value="bass">Bass</option>
  </select>
</form>
```

RESULT

Do you play any of the following instruments? (You can select more than one option by holding down control on a PC or command key on a Mac while selecting different options.)



<select>

size

You can turn a drop down select box into a box that shows more than one option by adding the `size` attribute. Its value should be the number of options you want to show at once. In the example you can see that three of the four options are shown.

Unfortunately, the way that browsers have implemented this attribute is not perfect, and it should be tested throroughly if used (in particular in Firefox and Safari on a Mac).

multiple

You can allow users to select multiple options from this list by adding the `multiple` attribute with a value of `multiple`.

It is a good idea to tell users if they can select more than one option at a time. It is also helpful to indicate that on a PC they should hold down the ***control*** key while selecting multiple options and on a Mac they should use the ***command*** key while selecting options.

FILE INPUT BOX

<input>

If you want to allow users to upload a file (for example an image, video, mp3, or a PDF), you will need to use a file input box.

type="file"

This type of input creates a box that looks like a text input followed by a ***browse*** button. When the user clicks on the ***browse*** button, a window opens up that allows them to select a file from their computer to be uploaded to the website.

When you are allowing users to upload files, the `method` attribute on the `<form>` element must have a value of `post`. (You cannot send files using the HTTP `get` method.)

When a user clicks on the ***browse*** button, the presentation of the window that allows them to browse for the file they want to upload will match the windows of the user's operating system. You cannot control the appearance of these windows.

chapter-07/file-input-box.html **HTML**

```
<form action="http://www.example.com/upload.php"
  method="post">
  <p>Upload your song in MP3 format:</p>
  <input type="file" name="user-song" /><br />
  <input type="submit" value="Upload" />
</form>
```

RESULT

SUBMIT BUTTON

HTML chapter-07/submit-button.html

```
<form action="http://www.example.com/subscribe.php">
  <p>Subscribe to our email list:</p>
  <input type="text" name="email" />
  <input type="submit" name="subscribe"
    value="Subscribe" />
</form>
```

RESULT

<input>

type="submit"

The submit button is used to send a form to the server.

name

It can use a `name` attribute but it does not need to have one.

value

The `value` attribute is used to control the text that appears on a button. It is a good idea to specify the words you want to appear on a button because the default value of buttons on some browsers is 'Submit query' and this might not be appropriate for all kinds of form.

Different browsers will show submit buttons in different ways and tend to fit the visual presentation of the browser. If you want to control the appearance of a submit button, you can either use CSS (as you will learn on page 343), or you can use an image for the button.

IMAGE BUTTON

<input>

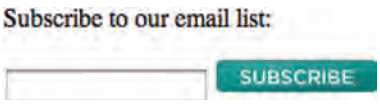
type="image"

If you want to use an image for the submit button, you can give the `type` attribute a value of `image`. The `src`, `width`, `height`, and `alt` attributes work just like they do when used with the `<img>` element (which we saw on pages 99-100).

chapter-07/image-button.html **HTML**

```
<form action="http://www.example.org/subscribe.php">
  <p>Subscribe to our email list:</p>
  <input type="text" name="email" />
  <input type="image" src="images/subscribe.jpg"
    width="100" height="20" />
</form>
```

RESULT



BUTTON & HIDDEN CONTROLS

HTML chapter-07/button-and-hidden-controls.html

```
<form action="http://www.example.com/add.php">
  <button><img src="images/add.gif" alt="add"
    width="10" height="10" /> Add</button>
  <input type="hidden" name="bookmark"
    value="lyrics" />
</form>
```

RESULT

<button>

The <button> element was introduced to allow users more control over how their buttons appear, and to allow other elements to appear inside the button.

This means that you can combine text and images between the opening <button> tag and closing </button> tag.

<input>

type="hidden"

This example also shows a hidden form control. These form controls are not shown on the page (although you can see them if you use the ***View Source*** option in the browser). They allow web page authors to add values to forms that users cannot see. For example, a web page author might use a hidden field to indicate which page the user was on when they submitted a form.

LABELLING FORM CONTROLS

<label>

When introducing form controls, the code was kept simple by indicating the purpose of each one in text next to it. However, each form control should have its own <label> element as this makes the form accessible to vision-impaired users.

The <label> element can be used in two ways. It can:

1. Wrap around both the text description and the form input (as shown on the first line of the example to your right).

2. Be kept separate from the form control and use the for attribute to indicate which form control it is a label for (as shown with the radio buttons).

chapter-07/labelling-form-controls.html **HTML**

```
<label>Age: <input type="text" name="age" /></label>
<br/ >
Gender:
<input id="female" type="radio" name="gender"
  value="f">
<label for="female">Female</label>
<input id="male" type="radio" name="gender"
  value="m">
<label for="male">Male</label>
```

RESULT

Age:
Gender: ○ Female ○ Male

for

The for attribute states which form control the label belongs to. Note how the radio buttons use the id attribute. The value of the id attribute uniquely identifies an element from all other elements on a page. (The id attribute is covered on page 183.)

The value of the for attribute matches that of the id attribute on the form control it is labelling. This technique using the for and id attributes can be used on any form control. When a <label> element is used with a checkbox or radio button, users can click on either the form control or the label to select. The expanded clickable area makes the form easier to use. The position of the label is very important. If users do not know where to enter information or what information to enter, they are less likely to use the form correctly.

As a rule of thumb, here are the best places to place labels on form controls.

ABOVE OR TO THE LEFT:
- Text inputs
- Text areas
- Select boxes
- File uploads

TO THE RIGHT:
- Individual checkboxes
- Individual radio buttons

GROUPING FORM ELEMENTS

HTML chapter-07/grouping-form-elements.html

```
<fieldset>
  <legend>Contact details</legend>
  <label>Email:<br />
  <input type="text" name="email" /></label><br />
  <label>Mobile:<br />
  <input type="text" name="mobile" /></label><br />
  <label>Telephone:<br />
  <input type="text" name="telephone" /></label>
</fieldset>
```

RESULT

<fieldset>

You can group related form controls together inside the `<fieldset>` element. This is particularly helpful for longer forms.

Most browsers will show the `fieldset` with a line around the edge to show how they are related. The appearance of these lines can be adjusted using CSS.

<legend>

The `<legend>` element can come directly after the opening `<fieldset>` tag and contains a caption which helps identify the purpose of that group of form controls.

HTML5: FORM VALIDATION

You have probably seen forms on the web that give users messages if the form control has not been filled in correctly; this is known as **form validation**.

Traditionally, form validation has been performed using JavaScript (which is beyond the scope of this book). But HTML5 is introducing validation and leaving the work to the browser.

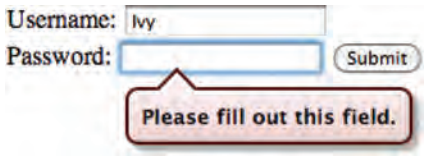
Validation helps ensure the user enters information in a form that the server will be able to understand when the form is submitted. Validating the contents of the form before it is sent to the server the helps:

- Reduce the amount of work the server has to do
- Enables users to see if there are problems with the form faster than if validation were performed on the server.

chapter-07/html5-form-validation.html **HTML**

```
<form action="http://www.example.com/login/"
    method="post">
  <label for="username">Username:</label>
  <input type="text" name="username"
    required="required" /></title><br />
  <label for="password">Password:</label>
  <input type="password" name="password"
    required="required" />
  <input type="submit" value="Submit" />
</form>
```

RESULT



At the time of writing, only Chrome and Opera supported HTML5 validation, although other browsers are expected to follow. In order to support older browsers (that do not understand HTML5), web page authors are likely to continue using JavaScript to validate forms.

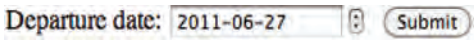
An example of HTML5 form validation is the `required` attribute, which can be used on any form element that the user is expected to fill in. This HTML5 attribute does not need a value, but in HTML 4 all attributes must have a value. So, some people give this attribute a value of `required`.

HTML5: DATE INPUT

HTML chapter-07/html5-date-input.html

```
<form action="http://www.example.com/bookings/"
    method="post">
  <label for="username">Departure date:</label>
  <input type="date" name="depart" />
  <input type="submit" value="Submit" />
</form>
```

RESULT



<input>

Many forms need to gather information such as dates, email addresses, and URLs. This has traditionally been done using text inputs.

HTML5 introduces new form controls to standardize the way that some information is gathered. Older browsers that do not recognize these inputs will just treat them as a single line text box.

type="date"

If you are asking the user for a date, you can use an `<input>` element and give the `type` attribute a value of `date`. This will create a date input in browsers that support the new HMTL5 input types.

This example shows what the date input looks like in the Opera browser. The appearance of the date input changes across different browsers.

HTML5: EMAIL & URL INPUT

<input>

HTML5 has also introduced inputs that allow visitors to enter email addresses and URLs. Browsers that do not support these input types will just treat them as text boxes.

type="email"

If you ask a user for an email address, you can use the email input. Browsers that support HTML5 validation will check that the user has provided information in the correct format of an email address. Some smart phones also optimize their keyboard to display the keys you are most likely to need when entering an email address (such as the @ symbol).

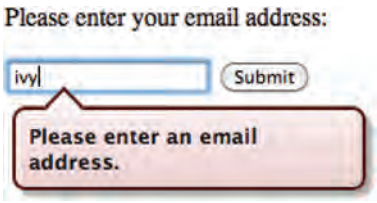
type="url"

A URL input can be used when you are asking a user for a web page address. Browsers that support HTML5 validation will check that the user has provided information in the format of a URL. Some smart phones also optimize their keyboard to display the keys you are most likely to need when entering a URL.

chapter-07/html5-email-input.html **HTML**

```
<form action="http://www.example.org/subscribe.php">
  <p>Please enter your email address:</p>
  <input type="email" name="email" />
  <input type="submit" value="Submit" />
</form>
```

RESULT



chapter-07/html5-url-input.html **HTML**

```
<form action="http://www.example.org/profile.php">
  <p>Please enter your website address:</p>
  <input type="url" name="website" />
  <input type="submit" value="Submit" />
</form>
```

RESULT

HTML5: SEARCH INPUT

HTML chapter-07/html5-search-input.html

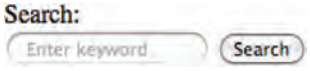
```
<form action="http://www.example.org/search.php">
  <p>Search:</p>
  <input type="search" name="search" />
  <input type="submit" value="Search" />
</form>
```

RESULT

HTML chapter-07/html5-placeholder.html

```
<form action="http://www.example.org/search.php">
  <p>Search:</p>
  <input type="search" name="search"
    placeholder="Enter keyword" />
  <input type="submit" value="Search" />
</form>
```

RESULT



<input>

If you want to create a single line text box for search queries, HTML5 provides a special type of input for that purpose.

type="search"

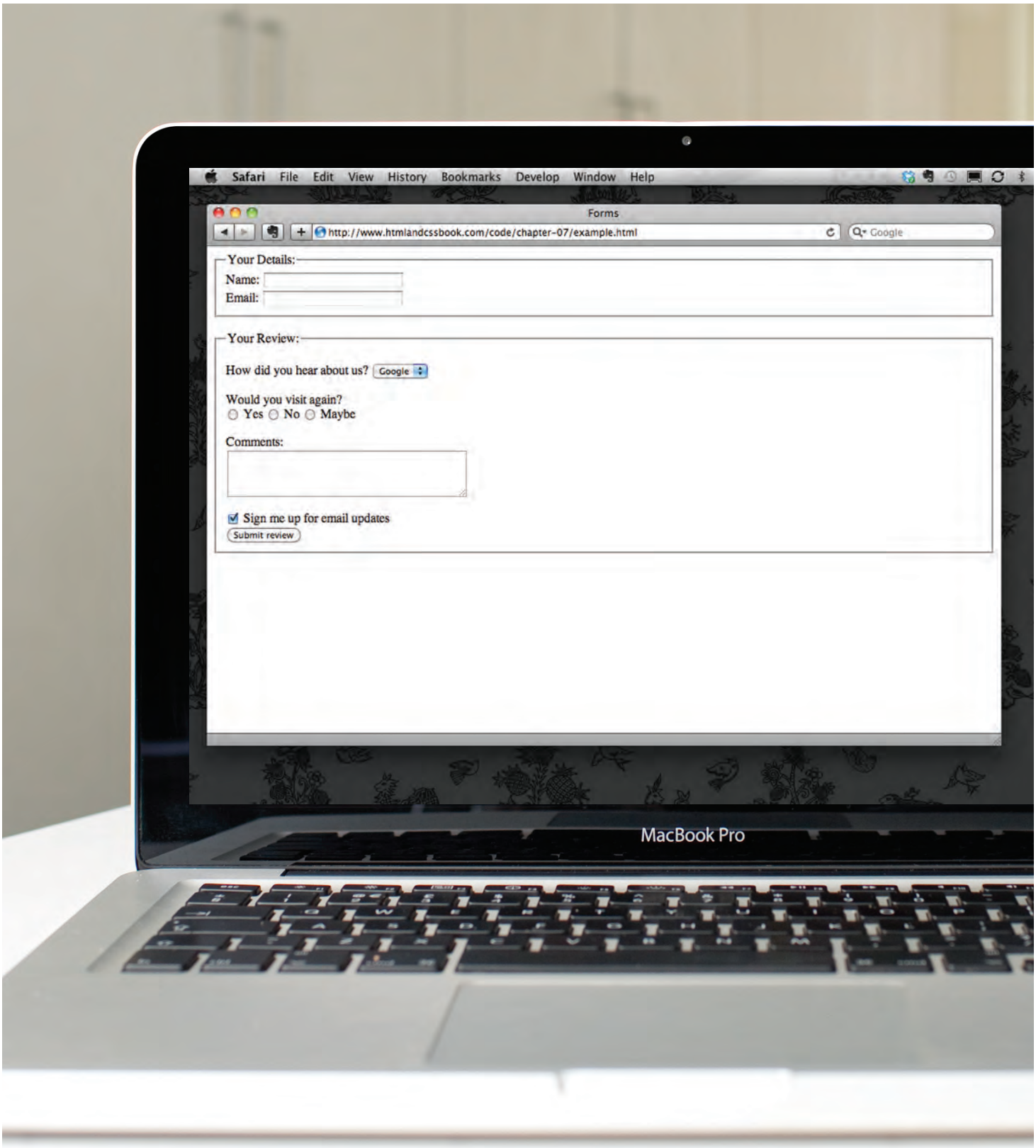
If you want to create a single line text box for search queries, HTML5 provides a special search input.

To create the HTML5 search box the `<input>` element should have a `type` attribute whose value is `search`. Older browsers will simply treat it like a single line text box.

Recent browsers add some features that improve usability. For example, Safari on a Mac adds a cross to clear the search box when you have started to enter information. Safari also automatically rounds the corners on the search input field.

placeholder

On any text input, you can also use an attribute called `placeholder` whose value is text that will be shown in the text box until the user clicks in that area. Older browsers simply ignore this attribute.


Safari File Edit View History Bookmarks Develop Window Help
Forms
http://www.htmlandcssbook.com/code/chapter-07/example.html
Google
Your Details:
Name:
Email:
Your Review:
How did you hear about us? Google
Would you visit again?
Yes No Maybe
Comments:
Sign me up for email updates
Submit review
MacBook Pro

EXAMPLE
FORMS

This example shows a feedback and newsletter sign-up form. It uses a variety of form controls.

The `<form>` element uses the `action` attribute to indicate the page that the data is being sent to. Each of the form controls sits inside the `<form>` element. Different types of form control are suited to collecting different types of data. The `<fieldset>` element is used to group related questions together. The `<label>` element indicates the purpose of each form control.

EXAMPLE
FORMS

```
<html>
  <head>
    <title>Forms</title>
  </head>
  <body>
    <form action="http://www.example.com/review.php" method="get">
      <fieldset>
        <legend>
          Your Details:
        </legend>
        <label>
          Name:
          <input type="text" name="name" size="30" maxlength="100">
        </label>
        <br />
        <label>
          Email:
          <input type="email" name="email" size="30" maxlength="100">
        </label>
        <br />
      </fieldset>
      <br />
      <fieldset>
        <legend>
          Your Review:
        </legend>
        <p>
          <label for="hear-about">
            How did you hear about us?
          </label>
          <select name="referrer" id="hear-about">
            <option value="google">Google</option>
            <option value="friend">Friend</option>
            <option value="advert">Advert</option>
            <option value="other">Other</option>
          </select>
        </p>
        <p>
```

EXAMPLE
FORMS

```
        Would you visit again?
        <br />
        <label>
          <input type="radio" name="rating" value="yes" />
          Yes
        </label>
        <label>
          <input type="radio" name="rating" value="no" />
          No
        </label>
        <label>
          <input type="radio" name="rating" value="maybe" />
          Maybe
        </label>
      </p>
      <p>
        <label for="comments">
          Comments:
        </label>
        <br />
        <textarea rows="4" cols="40" id="comments">
        </textarea>
      </p>
      <label>
        <input type="checkbox" name="subscribe" checked="checked" />
        Sign me up for email updates
      </label>
      <br />
      <input type="submit" value="Submit review" />
    </fieldset>
  </form>
 </body>
</html>
```

SUMMARY
FORMS

- Whenever you want to collect information from visitors you will need a form, which lives inside a `<form>` element.
- Information from a form is sent in name/value pairs.
- Each form control is given a name, and the text the user types in or the values of the options they select are sent to the server.
- HTML5 introduces new form elements which make it easier for visitors to fill in forms.

8

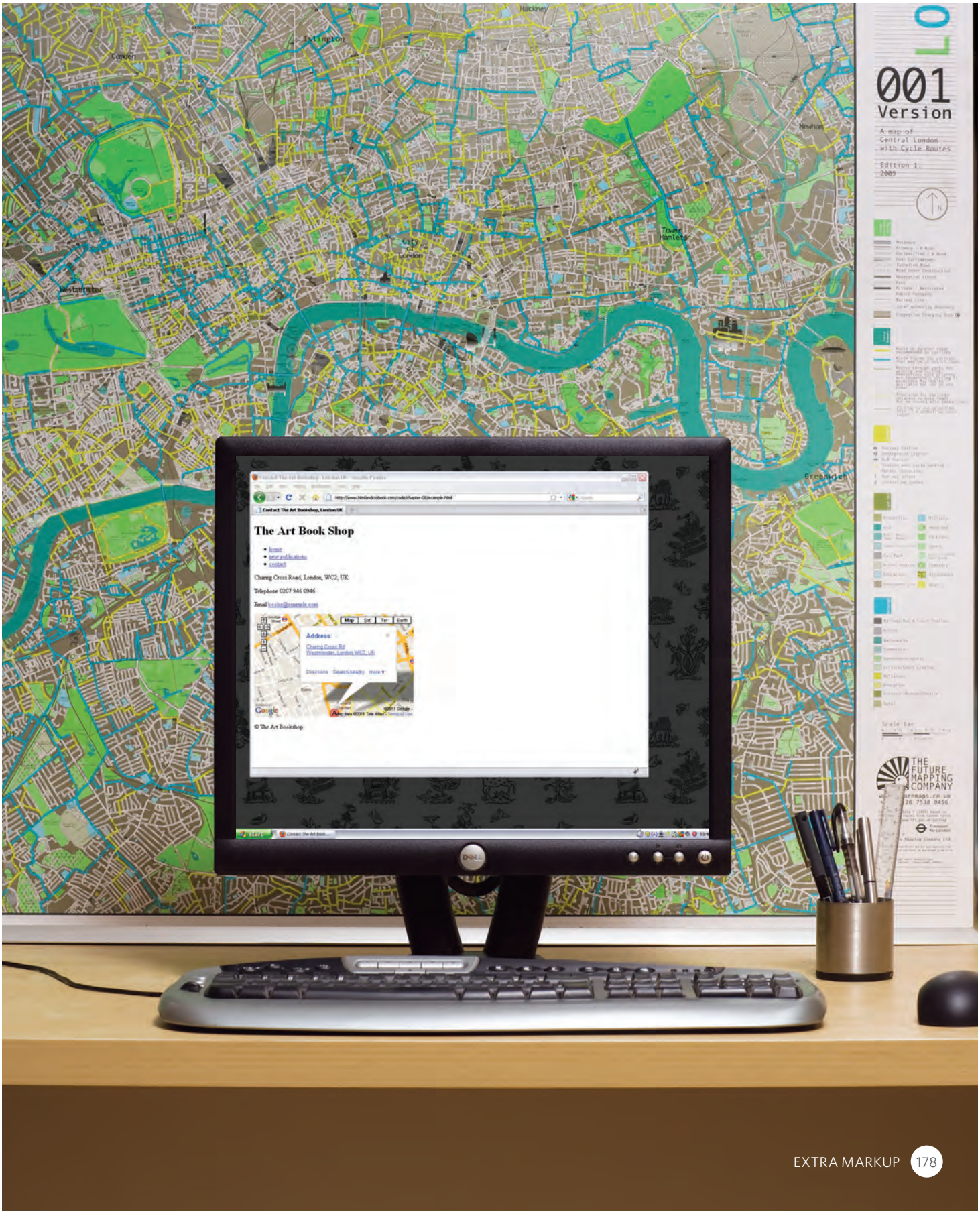
EXTRA MARKUP

- Specifying different versions of HTML
- Identifying and grouping elements
- Comments, meta information and iframes

At this point, we have covered the main tags that fit nicely into groups and sections.

In this chapter, we will focus on some helpful topics that are not easily grouped together. You will learn about:

- The different versions of HTML and how to indicate which version you are using
- How to add comments to your code
- Global attributes, which are attributes that can be used on any element, including the `class` and `id` attributes
- Elements that are used to group together parts of the page where no other element is suitable
- How to embed a page within a page using iframes
- How to add information about the web page using the `<meta>` element
- Adding characters such as angled brackets and copyright symbols


001
Version
A map of
Central London
with Cycle Routes
Edition 1:
2009
The Art Book Shop
home
new publications
contact
Charing Cross Road, London, WC2, UK
Telephone 0207 946 0946
Email books@example.com
Address:
© The Art Bookshop
THE
FUTURE
MAPPING
COMPANY

THE EVOLUTION OF HTML

Since the web was first created, there have been several different versions of HTML.

Each new version was designed to be an improvement on the last (with new elements and attributes added and older code removed).

There have also been several versions of each browser used to view web pages, each of which implements new code. Not all web users, however, have the latest browsers installed on their computers, which means that not everyone will be able to view all of the latest features and markup.

Where you should be particularly aware of browsers not supporting certain features, I have made a note of this (as you have seen with some of the HTML5 elements introduced in the Forms chapter — and as you will see in the CSS chapters).

HTML 4

RELEASED 1997

With the exception of a few elements added in HTML5 (which have been highlighted), the elements you have seen in this book were all available in HTML 4.

Although HTML 4 had some presentational elements to control the appearance of pages, authors are not recommended to use them any more. (Examples include the `<center>` element for centering content on a page, `<font>` for controlling the appearance of text, and `<strike>` to put a line through the text — all of these can be achieved with CSS instead.)

XHTML 1.0

RELEASED 2000

In 1998, a language called XML was published. Its purpose was to allow people to write new markup languages. Since HTML was the most widely used markup language around, it was decided that HTML 4 should be reformulated to follow the rules of XML and it was renamed XHTML. This meant that authors had to follow some new, more strict rules about writing markup. For example:

- Every element needed a closing tag (except for empty elements such as `<img />`).
- Attribute names had to be in lowercase.
- All attributes required a value, and all values were to be placed in double quotes.
- Deprecated elements should no longer be used.
- Every element that was opened inside another element should be closed inside that same element.

The examples in this book all follow these strict rules of XML.

One of the key benefits of this change was that XHTML works seamlessly with other programs that are written to create and process XML documents.

It could also be used with other data formats such as Scalable Vector Graphics (SVG) — a graphical language written in XML, MathML (used to mark up mathematical formulae), and CML (used to mark up chemical formulae).

In order to help web page authors move to this new syntax, two main flavors of XHTML 1.0 were created:

- **Strict XHTML 1.0**, where authors had to follow the rules to the letter
- **Transitional XHTML 1.0**, where authors could still use presentational elements (such as `<center>` and `<font>`).

The transitional version of XHTML was created because it allowed authors to continue to follow older practices (with a less strict syntax) and use some of the elements and attributes that were going to be removed from future versions of HTML.

There was also a third version of XHTML 1.0 called **XHTML 1.0 Frameset**, which allowed web page authors to partition a browser window into several "frames," each of which would hold a different HTML page. These days, frames are very rarely used and are being phased out.

HTML5

RELEASED 2000

In HTML5, web page authors do not need to close all tags, and new elements and attributes will be introduced. At the time of writing, the HTML5 specification had not been completed, but the major browser makers had started to implement many of the new features, and web page authors were rapidly adopting the new markup.

Despite the fact that HTML5 is not yet completed, you can safely take advantage of the new features of the language as long as you endeavour to ensure that users with older browsers will be able to view your pages (even though some of the extra features will not be visible to them).

DOCTYPES

Because there have been several versions of HTML, each web page should begin with a DOCTYPE declaration to tell a browser which version of HTML the page is using (although browsers usually display the page even if it is not included). We will therefore be including one in each example for the rest of the book.

As you will see when we come to look at CSS and its box model on page 316, the use of a DOCTYPE can also help the browser to render a page correctly.

Because XHTML was written in XML, you will sometimes see pages that use the XHTML strict DOCTYPE start with the optional XML declaration. Where this is used, it should be the first thing in a document. There must be nothing before it, not even a space.

HTML5 — HTML

```
<!DOCTYPE html>
```

HTML 4

```
<!DOCTYPE html PUBLIC
  "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
```

Transitional XHTML 1.0

```
<!DOCTYPE html PUBLIC
  "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/
   xhtml1-transitional.dtd">
```

Strict XHTML 1.0

```
<!DOCTYPE html PUBLIC
  "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/
   xhtml1-strict.dtd">
```

XML Declaration

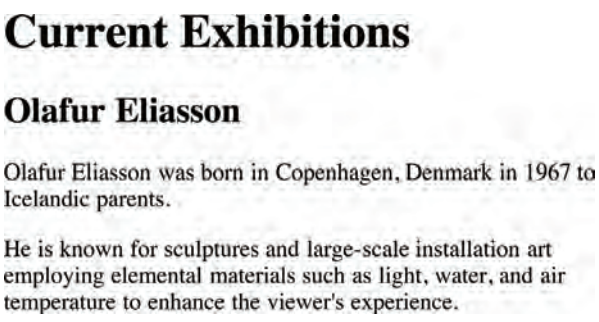
```
<?xml version="1.0" ?>
```

COMMENTS IN HTML

HTML chapter-08/comments-in-html.html

```
<!-- start of introduction -->
<h1>Current Exhibitions</h1>
<h2>Olafur Eliasson</h2>
<!-- end of introduction -->
<!-- start of main text -->
<p>Olafur Eliasson was born in Copenhagen, Denmark
   in 1967 to Icelandic parents.</p>
<p>He is known for sculptures and large-scale
   installation art employing elemental materials
   such as light, water, and air temperature to
   enhance the viewer's experience.</p>
<!-- end of main text -->
<!--
  <a href="mailto:info@example.org">Contact</a>
-->
```

RESULT



<!-- -->

If you want to add a comment to your code that will not be visible in the user's browser, you can add the text between these characters:

```
<!-- comment goes here -->
```

It is a good idea to add comments to your code because, no matter how familiar you are with the page at the time of writing it, when you come back to it later (or if someone else needs to look at the code), comments will make it much easier to understand.

Although comments are not visible to users in the main browser window, they can be viewed by anyone who looks at the source code behind the page.

On a long page you will often see comments used to indicate where sections of the page start or end, and to pass on notes to help anyone who is looking at the code understand it.

Comments can also be used around blocks of code to stop that code from being displayed in the browser. In the example on the left, the email link has been commented out.

ID ATTRIBUTE

Every HTML element can carry the `id` attribute. It is used to uniquely identify that element from other elements on the page. Its value should start with a letter or an underscore (not a number or any other character). It is important that no two elements on the same page have the same value for their `id` attributes (otherwise the value is no longer unique).

As you will see when you come to look at CSS in the next section, giving an element a unique identity allows you to style it differently than any other instance of the same element on the page. For example, you might want to assign one paragraph within the page (perhaps a paragraph containing a pull quote) a different style than all of the other paragraphs. In the example on the right, the paragraph with the `id` attribute whose value is `pullquote` is made uppercase using CSS.

If you go on to learn about JavaScript (a language that allows you to add interactivity to your pages), `id` attributes can be used to allow the script to work with that particular element.

The `id` attribute is known as a **global attribute** because it can be used on any element.

chapter-08/id-attribute.html **HTML**

```
<p>Water and air. So very commonplace are these
   substances, they hardly attract attention - and
   yet they vouchsafe our very existence.</p>
<p id="pullquote">Every time I view the sea I feel
   a calming sense of security, as if visiting my
   ancestral home; I embark on a voyage of seeing.
   </p>
<p>Mystery of mysteries, water and air are right
   there before us in the sea.</p>
```

RESULT

Water and air. So very commonplace are these substances, they hardly attract attention - and yet they vouchsafe our very existence.

EVERY TIME I VIEW THE SEA I FEEL A CALMING SENSE OF SECURITY, AS IF VISITING MY ANCESTRAL HOME; I EMBARK ON A VOYAGE OF SEEING.

Mystery of mysteries, water and air are right there before us in the sea.

CLASS ATTRIBUTE

HTML chapter-08/class-attribute.html

```
<p class="important">For a one-year period from
   November 2010, the Marugame Genichiro-Inokuma
   Museum of Contemporary Art (MIMOCA) will host a
   cycle of four Hiroshi Sugimoto exhibitions.</p>
<p>Each will showcase works by the artist
   thematically contextualized under the headings
   "Science," "Architecture," "History" and
   "Religion" so as to present a comprehensive
   panorama of the artist's oeuvre.</p>
<p class="important admittance">Hours: 10:00 - 18:00
   (No admittance after 17:30)</p>
```

RESULT

FOR A ONE-YEAR PERIOD FROM NOVEMBER 2010, THE MARUGAME GENICHIRO-INOKUMA MUSEUM OF CONTEMPORARY ART (MIMOCA) WILL HOST A CYCLE OF FOUR HIROSHI SUGIMOTO EXHIBITIONS.

Each will showcase works by the artist thematically contextualized under the headings "Science," "Architecture," "History" and "Religion" so as to present a comprehensive panorama of the artist's oeuvre.

HOURS: 10:00 - 18:00 (NO ADMITTANCE AFTER 17:30)

Every HTML element can also carry a `class` attribute. Sometimes, rather than uniquely identifying one element within a document, you will want a way to identify several elements as being different from the other elements on the page. For example, you might have some paragraphs of text that contain information that is more important than others and want to distinguish these elements, or you might want to differentiate between links that point to other pages on your own site and links that point to external sites.

To do this you can use the `class` attribute. Its value should describe the class it belongs to. In the example on the left, key paragraphs have a `class` attribute whose value is `important`.

The `class` attribute on any element can share the same value. So, in this example, the value of `important` could be used on headings and links, too.

By default, using these attributes does not affect the presentation of an element. It will only change their appearance if there is a CSS rule that indicates it should be displayed differently.

In this example, CSS has been applied to make elements with a `class` attribute whose value is `important` uppercase, and elements with a `class` attribute whose value is `admittance` red.

If you would like to indicate that an element belongs to several classes, you can separate class names with a space, as you can see in the third paragraph in the example above.

BLOCK ELEMENTS

Some elements will always appear to start on a new line in the browser window. These are known as **block level** elements.

Examples of block elements are `<h1>`, `<p>`, `<ul>`, and `<li>`.

chapter-08/block-elements.html **HTML**

```
<h1>Hiroshi Sugimoto</h1>
<p>The dates for the ORIGIN OF ART exhibition are as
   follows:</p>
<ul>
  <li>Science: 21 Nov - 20 Feb 2010/11</li>
  <li>Architecture: 6 Mar - 15 May 2011</li>
  <li>History: 29 May - 21 Aug 2011</li>
  <li>Religion: 28 Aug - 6 Nov 2011</li>
</ul>
```

RESULT

Hiroshi Sugimoto

The dates for the ORIGIN OF ART exhibition are as follows:

- Science: 21 Nov - 20 Feb 2010/11
- Architecture: 6 Mar - 15 May 2011
- History: 29 May - 21 Aug 2011
- Religion: 28 Aug - 6 Nov 2011

INLINE ELEMENTS

HTML chapter-08/inline-elements.html

```
Timed to a single revolution of the planet around
the sun at a 23.4 degrees tilt that plays out the
rhythm of the seasons, this <em>Origins of Art</em>
cycle is organized around four themes: <b>science,
architecture, history</b> and <b>religion</b>.
```

RESULT

Timed to a single revolution of the planet around the sun at a 23.4 degrees tilt that plays out the rhythm of the seasons, this *Origins of Art* cycle is organized around four themes: **science, architecture, history** and **religion**.

Some elements will always appear to continue on the same line as their neighbouring elements. These are known as **inline** elements.

Examples of inline elements are `<a>`, `<b>`, `<em>`, and `<img>`.

GROUPING TEXT & ELEMENTS IN A BLOCK

The element allows you to group a set of elements together in one block-level box.

For example, you might create a element to contain all of the elements for the header of your site (the logo and the navigation), or you might create a element to contain comments from visitors.

In a browser, the contents of the element will start on a new line, but other than this it will make no difference to the presentation of the page.

Using an id or class attribute on the element, however, means that you can create CSS style rules to indicate how much space the element should occupy on the screen and change the appearance of all the elements contained within it.

It can also make it easier to follow your code if you have used elements to hold each section of the page.

chapter-08/grouping-block-elements.html **HTML**

```
<div id="header">
  <img src="images/logo.gif" alt="Anish Kapoor" />
  <ul>
    <li><a href="index.html">Home</a></li>
    <li><a href="biography.html">Biography</a></li>
    <li><a href="works.html">Works</a></li>
    <li><a href="contact.html">Contact</a></li>
  </ul>
</div><!-- end of header -->
```

RESULT

Since there may be several other elements inside a element, it can be helpful to add a comment after the closing tag.

This allows you to clearly see which opening tag it is supposed to correspond to, as shown at the end of the example here.

GROUPING TEXT & ELEMENTS INLINE

HTML chapter-08/grouping-inline-elements.html

```
<p>Anish Kapoor won the Turner Prize in 1991 and
   exhibited at the <span class="gallery">Tate
   Modern</span> gallery in London in 2003.</p>
```

RESULT

Anish Kapoor won the Turner Prize in 1991 and exhibited at the TATE MODERN gallery in London in 2003.

The element acts like an inline equivalent of the element. It is used to either:

1. Contain a section of text where there is no other suitable element to differentiate it from its surrounding text

2. Contain a number of inline elements

The most common reason why people use elements is so that they can control the appearance of the content of these elements using CSS.

You will usually see that a class or id attribute is used with elements:

- To explain the purpose of this element
- So that CSS styles can be applied to elements that have specific values for these attributes

IFRAMES

<iframe>

An iframe is like a little window that has been cut into your page — and in that window you can see another page. The term iframe is an abbreviation of inline frame.

One common use of iframes (that you may have seen on various websites) is to embed a Google Map into a page. The content of the iframe can be any html page (either located on the same server or anywhere else on the web).

An iframe is created using the `<iframe>` element. There are a few attributes that you will need to know to use it:

src

The `src` attribute specifies the URL of the page to show in the frame.

height

The `height` attribute specifies the height of the iframe in pixels.

width

The `width` attribute specifies the width of the iframe in pixels.

chapter-08/iframes.html **HTML**

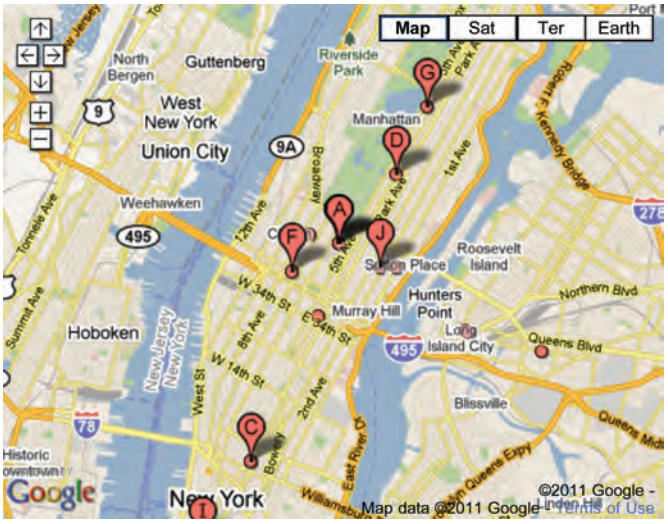
```
<iframe
  width="450"
  height="350"
  src="http://maps.google.co.uk/maps?q=moma+new+york
  &amp;output=embed">
</iframe>
```

RESULT

HTML chapter-08/iframes-continued.html

```
<iframe
  src="http://maps.google.co.uk/maps?q=moma+new+york
  &amp;output=embed"
  width="450"
  height="350"
  frameborder="0"
  scrolling="no">
</iframe>
```

RESULT



scrolling

The `scrolling` attribute will not be supported in HTML5. In HTML 4 and XHTML, it indicates whether the iframe should have scrollbars or not. This is important if the page inside the iframe is larger than the space you have allowed for it (using the `height` and `width` attributes). Scrollbars allow the user to move around the frame to see more content. It can take one of three values: `yes` (to show scrollbars), `no` (to hide scrollbars) and `auto` (to show them only if needed).

frameborder

The `frameborder` attribute will not be supported in HTML5. In HTML 4 and XHTML, it indicates whether the frame should have a border or not. A value of `0` indicates that no border should be shown. A value of `1` indicates that a border should be shown.

seamless

In HTML5, a new attribute called `seamless` can be applied to an iframe where scrollbars are not desired. The `seamless` attribute (like some other new HTML5 attributes) does not need a value, but you will often see authors give it a value of `seamless`. Older browsers do not support the `seamless` attribute.

INFORMATION ABOUT YOUR PAGES

<meta>

The <meta> element lives inside the <head> element and contains information about that web page.

It is not visible to users but fulfills a number of purposes such as telling search engines about your page, who created it, and whether or not it is time sensitive. (If the page is time sensitive, it can be set to expire.)

The <meta> element is an empty element so it does not have a closing tag. It uses attributes to carry the information.

The most common attributes are the name and content attributes, which tend to be used together. These attributes specify properties of the entire page. The value of the name attribute is the property you are setting, and the value of the content attribute is the value that you want to give to this property.

In the first line of the example on the opposite page, you can see a <meta> element where the name attribute indicates an intention to specify a description for the page. The content attribute is where this description is actually specified.

The value of the name attribute can be anything you want it to be. Some defined values for this attribute that are commonly used are:

description

This contains a description of the page. This description is commonly used by search engines to understand what the page is about and should be a maximum of 155 characters. Sometimes it is also displayed in search engine results.

keywords

This contains a list of comma-separated words that a user might search on to find the page. In practice, this no longer has any noticeable effect on how search engines index your site.

robots

This indicates whether search engines should add this page to their search results or not. A value of noindex can be used if this page should not be added. A value of nofollow can be used if search engines should add this page in their results but not any pages that it links to.

HTML chapter-08/meta.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>Information About Your Pages</title>
    <meta name="description"
      content="An Essay on Installation Art" />
    <meta name="keywords"
      content="installation, art, opinion" />
    <meta name="robots"
      content="nofollow" />
    <meta http-equiv="author"
      content="Jon Duckett" />
    <meta http-equiv="pragma"
      content="no-cache" />
    <meta http-equiv="expires"
      content="Fri, 04 Apr 2014 23:59:59 GMT" />
  </head>
  <body>
  </body>
</html>
```

The `<meta>` element also uses the `http-equiv` and `content` attributes in pairs. In our example, you can see three instances of the `http-equiv` attribute. Each one has a different purpose:

author

This defines the author of the web page.

pragma

This prevents the browser from caching the page. (That is, storing it locally to save time downloading it on subsequent visits.)

expires

Because browsers often cache the content of a page, the `expires` option can be used to indicate when the page should expire (and no longer be cached). Note that the date must be specified in the format shown.

ESCAPE CHARACTERS

There are some characters that are used in and reserved by HTML code. (For example, the left and right angled brackets.)

Therefore, if you want these characters to appear on your page you need to use what are termed "escape" characters (also known as escape codes or entity references). For example, to write a left angled bracket, you can use either `<` or `&#60;`. For an ampersand, you can use either `&` or `&#38;`.

There are also special codes that can be used to show symbols such as copyright and trademark, currency symbols, mathematical characters, and some punctuation marks. For example, if you want to include a copyright symbol on a web page you can use either `&copy;` or `&#169;`.

When using escape characters, it is important to check the page in your browser to ensure that the correct symbol shows up. This is because some fonts do not support all of these characters and you might therefore need to specify a different font for these characters in your CSS code.

ONLINE EXTRA
You can find a more complete list of escape codes in the tools section of the website accompanying this book.

< **Less-than sign**
`<`
`&#60;`

> **Greater-than sign**
`>`
`&`

& **Ampersand**
`&`
`&#38;`

" **Quotation mark**
`&quot;`
`&#34;`

¢ **Cent sign**
`&cent;`
`&#162;`

£ **Pound sign**
`&pound;`
`&#163;`

¥ **Yen sign**
`&yen;`
`&#165;`

€ **Euro sign**
`&euro;`
`&#8364;`

© **Copyright symbol**
`&copy;`
`&#169;`

® **Registered trademark**
`&reg;`
`&#174;`

TM **Trademark**
`&trade;`
`&#8482;`

‘ **Left single quote**
`&lsquo;`
`&#8216;`

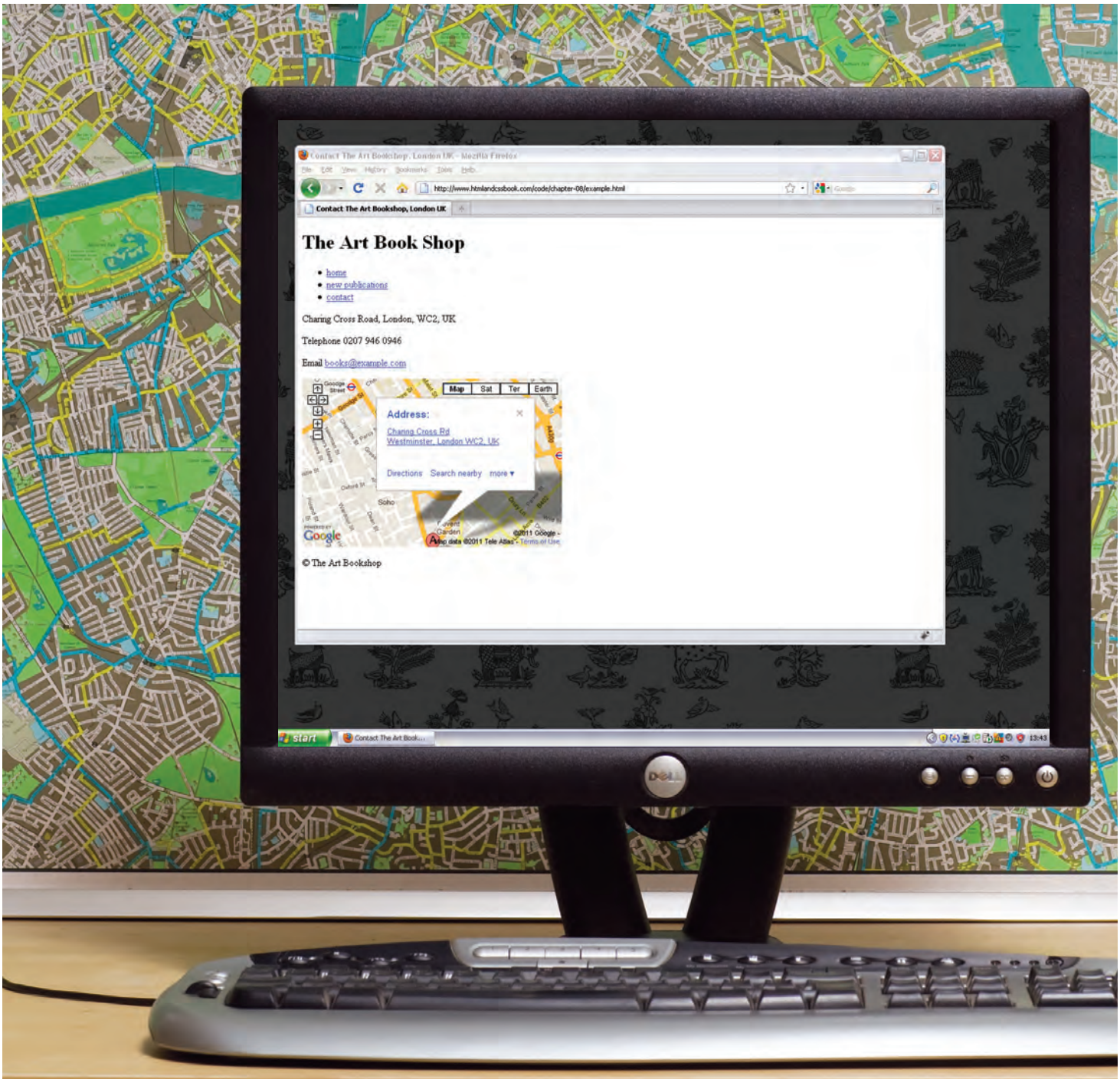
’ **Right single quote**
`&rsquo;`
`&#8217;`

“ **Left double quotes**
`&ldquo;`
`&#8220;`

” **Right double quotes**
`&rdquo;`
`&#8221;`

× **Multiplication sign**
`&times;`
`&#215;`

÷ **Division sign**
`&divide;`
`&#247;`



This example starts by using a `DOCTYPE` to indicate that this is an HTML 4 web page. In the `head`, you can also see a `<meta>` tag describing the page's content. Several elements use the `id` and `class` attributes to identify their purpose. The copyright symbol has been added using an escape code.

Parts of the page have been grouped using `<div>` elements, and comments have been added to indicate what the `</div>` elements are closing.

EXAMPLE
EXTRA MARKUP

```
<!DOCTYPE html PUBLIC
  "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
<html>
  <head>
    <meta name="description" content="Telephone, email
      and directions for The Art Bookshop, London, UK" />
    <title>Contact The Art Bookshop, London UK</title>
  </head>
  <body>
    <div id="header">
      <h1>The Art Book Shop</h1>
      <ul>
        <li><a href="index.html">home</a></li>
        <li><a href="index.html">new publications</a>
            </li>
        <li class="current-page">
            <a href="index.html">contact</a></li>
      </ul>
    </div><!-- end header -->
    <div id="content">
      <p>Charing Cross Road, London, WC2, UK</p>
      <p><span class="contact">Telephone</span>
         0207 946 0946</p>
      <p><span class="contact">Email</span>
         <a href="mailto:books@example.com">
         books@example.com</a></p>
      <iframe width="425" height="275" frameborder="0"
        scrolling="no" marginheight="0" marginwidth="0"
        src="http://maps.google.co.uk/maps?f=q&amp;
        source=s_q&amp;hl=en&amp;geocode=&amp;
        q=charing+cross+road+london&amp;output=embed">
        </iframe>
    </div><!-- end content -->
    <p>&copy; The Art Bookshop</p>
  </body>
</html>
```

SUMMARY
EXTRA MARKUP

- `DOCTYPES` tell browsers which version of HTML you are using.
- You can add comments to your code between the `<!--` and `-->` markers.
- The `id` and `class` attributes allow you to identify particular elements.
- The `<div>` and `<span>` elements allow you to group block-level and inline elements together.
- `<iframes>` cut windows into your web pages through which other pages can be displayed.
- The `<meta>` tag allows you to supply all kinds of information about your web page.
- Escape characters are used to include special characters in your pages such as <, >, and ©.

9

FLASH, VIDEO & AUDIO

- How to add Flash movies into your site
- How to add video and audio to your site
- HTML5 `<video>` and `<audio>` elements

Flash is a very popular technology used to add animations, video, and audio to websites. This chapter begins by looking at how to use it in your web pages.

We then focus on how to add video and audio to your site, using either the new HTML5 <video> and <audio> elements or a hosted service (such as YouTube or SoundCloud). In this chapter you will learn:

- How to use Flash in your web pages
- How to use HTML5 `<video>` and `<audio>` elements
- When to host your own video and audio and when to use a service such as YouTube


Safari File Edit View History Bookmarks Develop Window Help
Sun 12:19
Flash, Video and Audio
http://www.htmlandcssbook.com/code/chapter-09/example.html
MacBook Pro

HOW FLASH WORKS

Since the late 1990s, Flash has been a very popular tool for creating animations, and later for playing audio and video in websites.

Whether you are creating an animation or a media player in Flash, the files you put on your website are referred to as **Flash movies**.

If you want to create your own Flash movie, you need to purchase the Flash authoring environment from Adobe.

There are, however, several companies that offer Flash animations and slideshows, as well as video and audio players that you can use without purchasing this tool.

When you create a Flash file in the Flash authoring environment, it is saved with the `.fla` file extension. In order to use this file on a web page it has to be saved in a different format known as SWF. (It has the `.swf` file extension.)

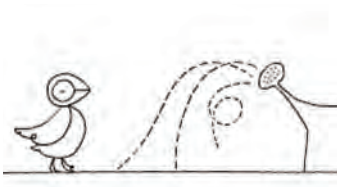
When you export the movie into SWF format, Flash creates code that you can use to embed the Flash movie in your page. Traditionally, this code used the HTML `<object>` and `<embed>` tags. However, now it is more common to use JavaScript.

To view Flash, browsers need to use a plugin (an extra piece of software that runs in the browser) called the Flash Player. Statistics commonly indicate that 98% of browsers on desktop computers have the Flash plugin installed. (The percentage of mobiles and tablets with it is much less.)

There is not space in this book to teach you how to create Flash movies (there are many books devoted to that one topic), but this chapter will show you how to add Flash movies to your site.

The Flash authoring environment is used to create Flash Movies.

The `.fla` file is exported to `.swf` format to use in a web page.



The `.swf` file is included in your web page using JavaScript.

USE OF FLASH

Since 2005, a number of factors have meant that fewer websites are written in Flash or even use elements of Flash in their pages.

When Flash was first released, it was developed to create animations. The technology quickly evolved, however, and people started to use it to build media players and even entire websites.

Although Flash is still very popular, in recent years people have been more selective about when they use it (and now rarely consider building an entire website in Flash).

Despite this, Flash does have a future on the web because there are some things it does very well, such as creating animations.

There are several reasons why fewer websites are using Flash these days, including:

In 2005-6, a set of JavaScript libraries were launched (including Prototype, script.aculo.us, and JQuery) which made it easier for people to create animated effects using JavaScript.

When Apple launched the iPhone in 2007 and later the the iPad in 2010, they took the decision not to support Flash.

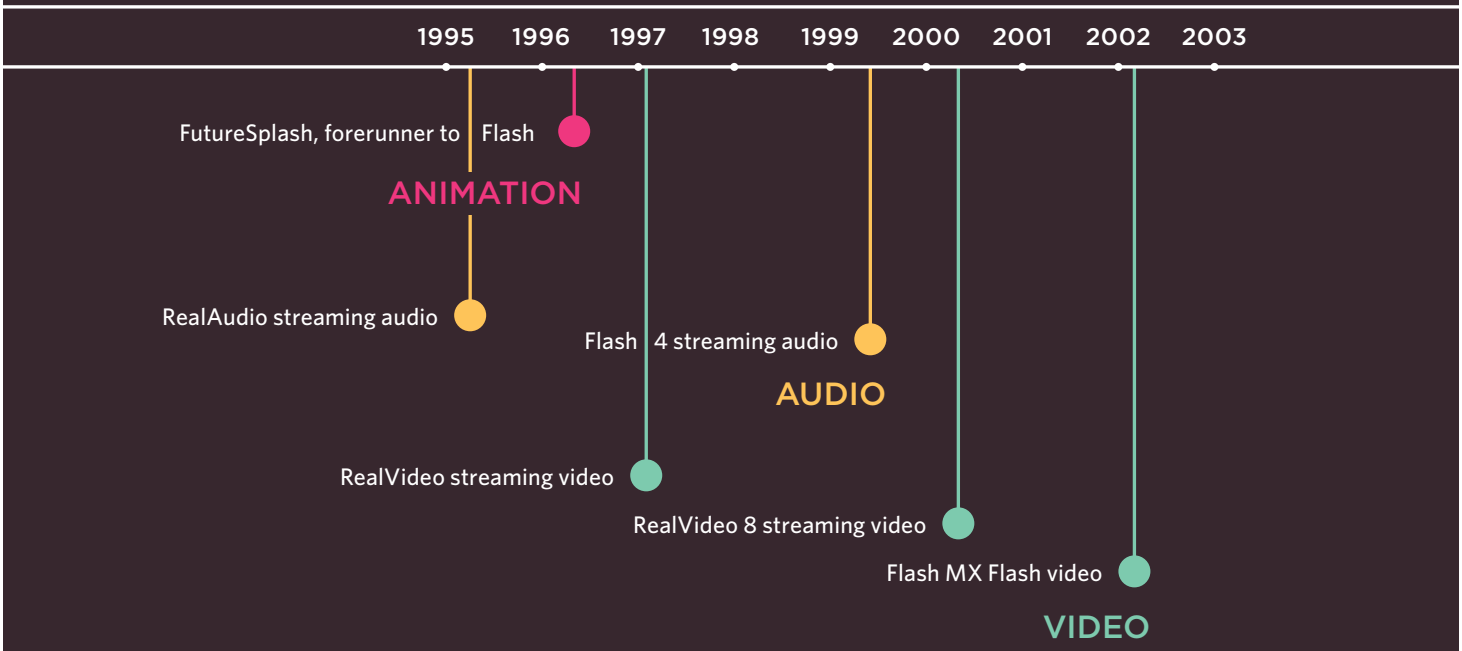
There have been laws introduced to ensure that websites are usable by those with visual or physical impairments — and Flash has been criticized because Flash content does not always meet accessibility requirements.

In 2008, browsers started to support HTML5 `<video>` and `<audio>` tags. At the time of writing, Flash is still a popular way of playing video and audio on the web but more and more people are switching to HTML5.

(You will see how to use these elements later in the chapter.)

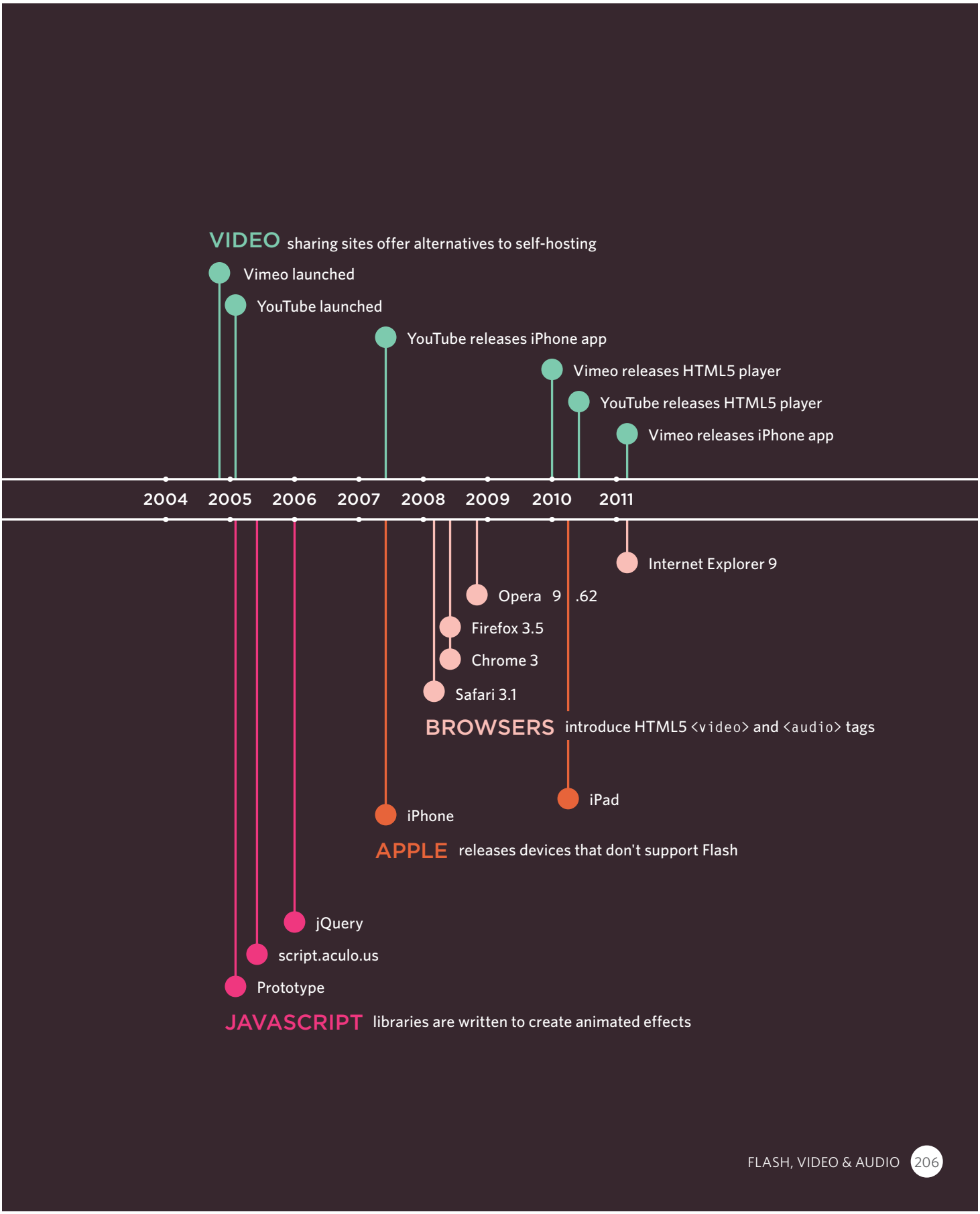
TIMELINE: FLASH, VIDEO & AUDIO

Web technologies change quickly. Here you can see some of the changes in how animation, video, and audio are created on the web.



On this page you can see the first major players to provide web animation, audio, and video.

On the facing page, you can see some of the technologies and events replacing them.


VIDEO sharing sites offer alternatives to self-hosting
Vimeo launched
YouTube launched
YouTube releases iPhone app
Vimeo releases HTML5 player
YouTube releases HTML5 player
Vimeo releases iPhone app
2004
2005
2006
2007
2008
2009
2010
2011
Internet Explorer 9
Opera 9 .62
Firefox 3.5
Chrome 3
Safari 3.1
BROWSERS introduce HTML5 <video> and <audio> tags
iPad
iPhone
APPLE releases devices that don't support Flash
jQuery
script.aculo.us
Prototype
JAVASCRIPT libraries are written to create animated effects

ADDING A FLASH MOVIE TO YOUR WEB PAGE

The most popular way of adding Flash into a web page is using JavaScript. There are several scripts that allow you to do this without an in-depth understanding of the JavaScript language.

The script we will be looking at here is called SWFObject. You can obtain a copy of it for free from Google, and you can see how we use it on the next page.

One advantage to using this technique is that it allows browsers to show alternative content for users whose browsers are not capable of showing Flash.

This technique uses a `<div>` element to create a space where the Flash movie should sit. The `<div>` element has an `id` attribute whose value is used by the SWFObject script. In this example, the value of the `id` attribute is `bird`.

Inside the `<div>` element you can place the alternative content for users who are not able to play Flash.

chapter-09/adding-a-flash-movie.html **HTML**

```
<!DOCTYPE html>
<html>
  <head>
    <title>Adding a Flash Movie</title>
    <script type="text/javascript"
      src="http://ajax.googleapis.com/ajax/libs/
      swfobject/2.2/swfobject.js"></script>
    <script type="text/javascript">
      swfobject.embedSWF("flash/bird.swf",
      "bird", "400", "300", "8.0.0");</script>
  </head>
  <body>
    <div id="bird"><p>An animation of a bird taking
      a shower</p></div>
  </body>
</html>
```

The SWFObject script will check to see if the user's browser can play the Flash movie. If it can, the script will replace the content of the `<div>` with the `.swf` file.

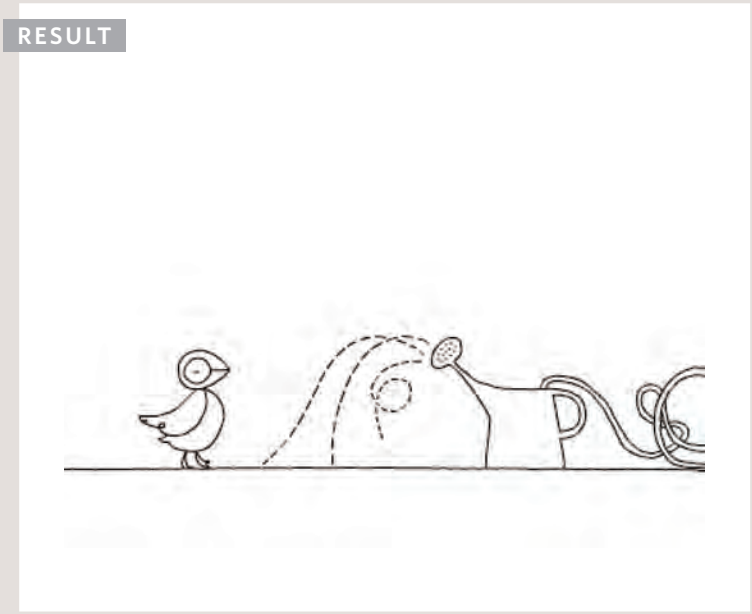
For users who cannot see the Flash movie, you could show a still from the movie instead. You might also like to consider using a text description of the Flash file.

If you use a text description as alternative content, then you can achieve two further benefits:

1. The text can be accessed by those with visual and/or physical impairments who are not able to interact with the Flash file.

2. The text can be indexed by search engines (which are not as effective at indexing SWF files), increasing the chance that your content will be found.

RESULT



In this example, the SWFObject script is hosted on Google's servers. We **include** the script in this web page using the first of the two `<script>` elements.

The type attribute is used on the `<script>` element to indicate that the script inside is written in JavaScript. The `src` attribute tells the browser where to find the script.

The second `<script>` element is used to tell the browser about the Flash movie, as well as which element it should replace. This element is actually telling the SWFObject script **five** pieces of information, which are in the brackets:

1. The **location** of the `.swf` file:
`flash/bird.swf`

2. The element that the Flash movie should **replace**, specified by the `value` of the `id` attribute on the `<div>` element:
`bird`

3. The **width** of the Flash movie:
`400 px`

4. The **height** of the Flash movie:
`300 px`

5. The minimum **version** of the Flash player needed to view the movie:
`Flash Player 8`

UNDERSTANDING VIDEO FORMATS AND PLAYERS

To add video to your site, there are two key issues to understand: file formats and video players/plugins.

FORMATS

Movies are available in many formats (BluRay, DVD, VHS, to name a few). Online, there are even *more* video formats (including AVI, Flash Video, H264, MPEG, Ogg Theora, QuickTime, WebM, and Windows Media).

Just as your DVD player won't play a VHS cassette, browsers differ in what video formats they do and don't support.

In order for users to view your video online, you may need to convert it to another format.

The process of converting a video into another format is sometimes referred to as "encoding" the video.

There are several apps available on the web that enable you to encode videos (such as `www.mirovideoconverter.com`).

PLAYERS / PLUGINS

Browsers were initially designed to show text and images only. For this reason, browsers built prior to 2010 generally required another program called a player or plugin to to be installed in order to play video content.

These players and plugins only supported certain video formats.

Recently browsers have evolved to support the HTML5 `<video>` tag (which renders players and plugins obsolete).

Unfortunately, however, you cannot rely on every visitor to your website having a recent browser that supports this new HTML5 element and the browsers that do recognize the `<video>` element require the video to be encoded in different formats.

APPROACH

The easiest way to add video to your site is to use a hosted service such as YouTube or Vimeo.

However, there are some cases where using these services is not appropriate (as you will see on the next page) and you will want to host the video on your own site.

At the time of writing, to ensure most people can play your video content, it is considered best practice to use the HTML5 `<video>` element for browsers that support it, and also Flash video for those that do not.This means you would need to upload any videos you want to show in at least two different formats: WebM and MP4.

USING HOSTED VIDEO SERVICES

The easiest way to add a video to your site is to upload the video to a site like YouTube or Vimeo and use the features provided on their site to embed the video in your page.

ADVANTAGES

Hosted video sites (such as YouTube) provide players that work with the majority of web browsers.

You do not need to worry about encoding your video since these sites allow you to upload your content in a range of formats. Once uploaded, they automatically convert your video into the various formats required by different browsers.

Web hosting companies often charge extra if you use a lot of bandwidth, and video files can be quite large. Therefore, it can cost you extra to host the videos on your own site. If your video is hosted on a site like YouTube or Vimeo, however, you do not need to pay for the bandwidth.

DISADVANTAGES

Your video will be available on the site of the hosted service, so if you want the content to be exclusively available on your site (and not visible on other sites), you need to host the video on your own server and add your own player into the page.

Some services will limit what your video is allowed to include. For example, most prohibit the use of advertising within the video you upload (which prevents you from monetizing that content).

Some hosted services will play their own adverts before your video will begin, or even overlay them over the screen as your video is playing. The quality of video on some hosted services can also be limited.

THE ALTERNATIVE

If you want to host video on your own site - rather than a hosted service - a lot more work is involved in setting up your site to play the video.

We will be looking at two different ways that you can host your own videos: using both Flash Video and the HTML5 `<video>` element.

In order to ensure that the maximum number of visitors to your site can see the video, you will need to use a combination of both of these techniques.

PREPARING A FLASH VIDEO FOR YOUR SITE

There are three steps you need to follow to add a Flash Video to your web page:

1

CONVERT YOUR VIDEO INTO FLV FORMAT

To play a Flash Video, you need to convert your video into FLV format. Since Flash 6, the Flash authoring environment has come with a Flash Video Encoder to convert videos into FLV format.

Some Flash video players also support a format called H264 (and some video editing programs export video in this format).

Googling "FLV or H264 converters" will allow you to find alternative encoding software.

I have provided a sample FLV file that you can use with the download code on the website (It is in a separate folder because the video files are large.)

2

FIND AN FLV PLAYER TO PLAY THE VIDEO

You'll need a **player** written in Flash to play the FLV file. Its purpose is to hold the FLV movie and add controls such as play/pause. Here are two sites that offer FLV players:

`www.osflv.com`
`www.longtailvideo.com`

You do not need the Flash authoring environment to use either of these on your website.

In the following example, we will use the OS FLV player, which is a free, open-source Flash Video player. This is included in the download code. It only supports the FLV format (not H264).

3

INCLUDE THE PLAYER & VIDEO IN YOUR PAGE

You can include the player in your page using a JavaScript technique such as SWFObject, which was mentioned earlier in this chapter.

You will also need to tell the player where it can find the video file that you want it to play. (Some players have advanced features such as the ability to create playlists of multiple videos, or add a still picture before the video plays.)

In the following example, we will also be using the SWFObject JavaScript technique mentioned on pages 207-208.

ADDING A FLASH VIDEO TO YOUR PAGES

HTML chapter-09/adding-a-flash-video.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>Adding a Flash Video</title>
    <script type="text/javascript"
      src="http://ajax.googleapis.com/ajax/libs/
      swfobject/2.2/swfobject.js"></script>
    <script type="text/javascript">
      var flashvars = {};
      var params = {movie:"../video/puppy.flv"};
      swfobject.embedSWF("flash/splayer.swf",
      "snow", "400", "320", "8.0.0",
      flashvars, params);</script>
  </head>
  <body>
    <div id="snow"><p>A video of a puppy playing in
      the snow</p></div>
  </body>
</html>
```

RESULT

This example uses the OS FLV player to display a video called `puppy.flv`, which has already been convered into FLV format.

You have already seen how to use SWFObject to embed a basic animation in a page, but sometimes Flash movies need information in order for them to work. In this example, the video player needs to know the path to the video it has to play, so SWFObject uses JavaScript variables to pass this information to the Flash movie. These are provided in the two lines of code that start with `var`.

This particular player is not expecting any information in the `flashvars` variable, so that is left empty.

The path to the movie is supplied in the variable called `params`.

```
var params = {movie:
"../videos/puppy.flv"};
```

The line after the variables is the one that tells the script to replace the HTML element with the video player. It is very similar to the one you saw in the earlier example that introduced SWFObject.

Different video players usually require information such as the path to the video in slightly different formats, but they usually come with examples and documentation to help you understand how to use them.

HTML5: PREPARING VIDEO FOR YOUR PAGES

Despite the HTML5 <video> element being a very recent addition, it is enjoying widespread use. Here are some of the key issues to be aware of:

SUPPORT

The new HTML5 <video> element is only supported by recent browsers, so you cannot just use this one technique if you want everyone to be able to see your video (you need to combine this HTML5 with Flash Video).

DIGITAL RIGHTS

At the time of writing, the <video> element does not support any type of Digital Rights Management (DRM — sometimes referred to as copy protection). But a dedicated pirate will usually find a way around DRM.

FORMATS

Not all browsers support the same video formats. Therefore, you need to supply your video in more than one format.

To reach as many browsers as possible, you should provide the video in the following formats:

H264: IE and Safari
WebM: Android, Chrome, Firefox, Opera

Chrome, Firefox, and Opera have indicated that they will support a format called WebM. (Some Flash players also support H264, and WebM - which will save on the number of conversions).

CONTROLS

The browser supplies its own controls for the player, and these can vary from browser to browser. You can control the appearance of these controls using JavaScript (but that is beyond the scope of this book).

IN THE BROWSER

One of the problems with players such as the Flash Player is that they can behave inconsistently when elements such as menus drop over them, or the window is scaled up or down. The HTML5 option solves these issues.

On page 222 you will see how to combine this HTML5 video technique with Flash Video to achieve wider reach.

I have provided a sample video in H264 and WebM format for you to try with the code downloads.

If you look at this example in Firefox and Opera you will see different controls when you hover over the video.

HTML5: ADDING VIDEO TO YOUR PAGES

HTML chapter-09/adding-html5-video.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>Adding HTML5 Video</title>
  </head>
  <body>
    <video src="video/puppy.mp4"
      poster="images/puppy.jpg"
      width="400" height="300"
      preload
      controls
      loop>
      <p>A video of a puppy playing in the snow</p>
    </video>
  </body>
</html>
```

In HTML5 you do not need to supply values for all attributes, such as the controls, autoplay, and loop attributes used with the `<video>` element. These attributes are like on/off switches. If the attribute is present, it turns that option on. If the attribute is omitted, the option is turned off.

If the browser does not support the `<video>` element or the format of video used, it will display whatever is between the opening `<video>` and closing `</video>` tags.

preload

This attribute tells the browser what to do when the page loads. It can have one of three values:

none

The browser should not load the video until the user presses play.

auto

The browser should download the video when the page loads.

metadata

The browser should just collect information such as the size, first frame, track list, and duration.

<video>

The `<video>` element has a number of attributes which allow you to control video playback:

src

This attribute specifies the path to the video. (The example video is in H264 format so it will only work in IE and Safari.)

poster

This attribute allows you to specify an image to show while the video is downloading or until the user tells the video to play.

width, height

These attributes specify the size of the player in pixels.

controls

When used, this attribute indicates that the browser should supply its own controls for playback.

autoplay

When used, this attribute specifies that the file should play automatically.

loop

When used, this attribute indicates that the video should start playing again once it has ended.

HTML5: MULTIPLE VIDEO SOURCES

<source>

To specify the location of the file to be played, you can use the <source> element inside the <video> element. (This should replace the src attribute on the opening <video> tag.)

You can also use multiple <source> elements to specify that the video is available in different formats.

(Due to a bug on the iPad, you should provide the MP4 video as the first format. Otherwise, it might not play.)

src

This attribute specifies the path to the video.

type

You should use this attribute to tell the browser what format the video is. Otherwise, it will download some of the video to see if it can play the file (which will take time and bandwidth).

codecs

The codec that was used to encode the video is supplied within the type attribute. Note the use of single quotes, as well as double quotes in the type attribute, when it is supplied.

chapter-09/multiple-video-sources.html **HTML**

```
<!DOCTYPE html>
<html>
  <head>
    <title>Multiple Video Sources</title>
  </head>
  <body>
    <video poster="images/puppy.jpg" width="400"
      height="320" preload controls loop>
      <source src="video/puppy.mp4" type='video/
        mp4;codecs="avc1.42E01E, mp4a.40.2"' />
      <source src="video/puppy.webm" type='video/
        webm;codecs="vp8, vorbis"' />
      <p>A video of a puppy playing in the snow</p>
    </video>
  </body>
</html>
```

RESULT

If the browser does not support the <video> element or the format of video used, it will display whatever is between the opening <video> and closing </video> tags.

ONLINE EXTRA

We have provided links to tools that help you encode videos and audio into the correct formats in the Tools section of the website.

HTML5: COMBINING FLASH & HTML5 VIDEO

By offering your videos in both HTML5 and Flash Video formats, you will ensure that it can be viewed by the majority of users on your site.

You may choose to offer HTML5 as the first option, and Flash video as a fallback for people whose browser does not support HTML5 video. Or you may work the other way around.

Because some of the video players built in Flash support H264 encoding, if you use a player that supports this format you would only need to provide the video in H264 and WebM formats. (You would not need it in FLV format as well.) You will see this demonstrated in the example at the end of the chapter.

If you start to work with HTML5 video in depth, you can also:

- Create your own playback controls
- Provide different versions of the video for browsers that have different sized screens (so you can provide lower resolution content for handheld devices)
- Tell different parts of a page to change when the video reaches a certain point

ADDING AUDIO TO WEB PAGES

By far the most popular format for putting audio on web pages is MP3. As with video, there are three routes commonly taken:

1

USE A HOSTED SERVICE

There are several sites that allow you to upload your audio, and provide a player which you can embed in your page, such as `SoundCloud.com` and `MySpace.com`.

Some people ask how to get music to play consistently even when visitors move from one page to another on a website.

2

USE FLASH

There are several Flash movies that allow you to play MP3 files; from simple buttons that play one track to complex players that allow you to create playlists and juke boxes.

This is actually quite difficult to achieve and would rely on techniques like using AJAX to load page content or developing the entire site in Flash.

3

USE HTML5

HTML5 has introduced a new `<audio>` element. Browsers that support this element provide their own controls — much as they do for the video files we just looked at.

This is why some sites offer audio players in new windows, so that listeners are not interrupted when they move between pages.

ADDING A FLASH MP3 PLAYER

HTML chapter-09/adding-a-flash-mp3-player.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>Adding a Flash MP3 Player</title>
    <script type="text/javascript"
      src="http://ajax.googleapis.com/ajax/libs/
      swfobject/2.2/swfobject.js"></script>
    <script type="text/javascript">
      var flashvars = {};
      var params = {mp3: "audio/test-audio.mp3"};
      swfobject.embedSWF(
        "flash/player_mp3_1.0.0.swf",
        "music-player", "200", "20", "8.0.0",
        flashvars, params);</script>
  </head>
  <body>
    <div id="music-player">
      <p>You cannot hear this track because this
         browser does not support our Flash music
         player.</p>
    </div>
  </body>
</html>
```

RESULT

There are many MP3 players that have already been written in Flash, such as:

`flash-mp3-player.net`
`musicplayer.sourceforge.net`
`www.wimpyplayer.com`

Each of these players has different functionality, so check their features before choosing one for your site.

This example uses a free player from `flash-mp3-player.net` which is embedded in the page using the SWFObject technique we met on pages 208-208. The player is told the path to the MP3 file using a parameter called `mp3`.

After the second `<script>` tag, you can see that we have created two JavaScript variables; the first called `flashvars`, the second called `params`. Even though we are not using the `flashvars` variable, the SWFObject script expects it before the `params` variable so we need it there.

```
var flashvars = {};
var params = {
  mp3: "music/noise.mp3"};
```

These variables are then added at the end of the line that embeds the MP3 player in the page (just before the second closing `<script>` tag).

HTML5: ADDING HTML5 AUDIO TO YOUR PAGES

<audio>

HTML5 introduced the `<audio>` element to include audio files in your pages. As with HTML5 video, browsers expect different formats for the audio.

The `<audio>` element has a number of attributes which allow you to control audio playback:

src

This attribute specifies the path to the audio file.

controls

This attribute indicates whether the player should display controls. If you do not use this attribute, no controls will be shown by default. You can also specify your own controls using JavaScript.

autoplay

The presence of this attribute indicates that the audio should start playing automatically. (It is considered better practice to let visitors choose to play audio.)

chapter-09/adding-html5-audio.html **HTML**

```
<!DOCTYPE html>
<html>
  <head>
    <title>Adding HTML5 Audio</title>
  </head>
  <body>
    <audio src="audio/test-audio.ogg"
      controls autoplay>
      <p>This browser does not support our audio
      format.</p>
    </audio>
  </body>
</html>
```

RESULT

preload

This attribute indicates what the browser should do if the player is not set to `autoplay`. It can have the same values we saw on page 214 for the `<video>` element.

loop

This attribute specifies that the audio track should play again once it has finished.

This example only works in browsers that support the Ogg Vorbis audio format (Firefox, Chrome, and Opera). For it to work in Safari 5 and IE 9, the audio would need to be in MP3 format (or use the `<source>` element covered on the next page to offer different formats).

HTML5: MULTIPLE AUDIO SOURCES

HTML chapter-09/multiple-audio-sources.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>Multiple Audio Sources</title>
  </head>
  <body>
    <audio controls autoplay>
      <source src="audio/test-audio.ogg" />
      <source src="audio/test-audio.mp3" />
      <p>This browser does not support our audio
      format.</p>
    </audio>
  </body>
</html>
```

RESULT

src

The `<source>` element uses the `src` attribute to indicate where the audio file is located.

type

At the time of writing, the `type` attribute was not commonly being used on the `<source>` element in the same way it was for the `<video>` element.

<source>

It is possible to specify more than one audio file using the `<source>` element between the opening `<audio>` and closing `</audio>` tags (instead of the `src` attribute on the opening `<audio>` tag).

This is important because different browsers support different formats for audio files.

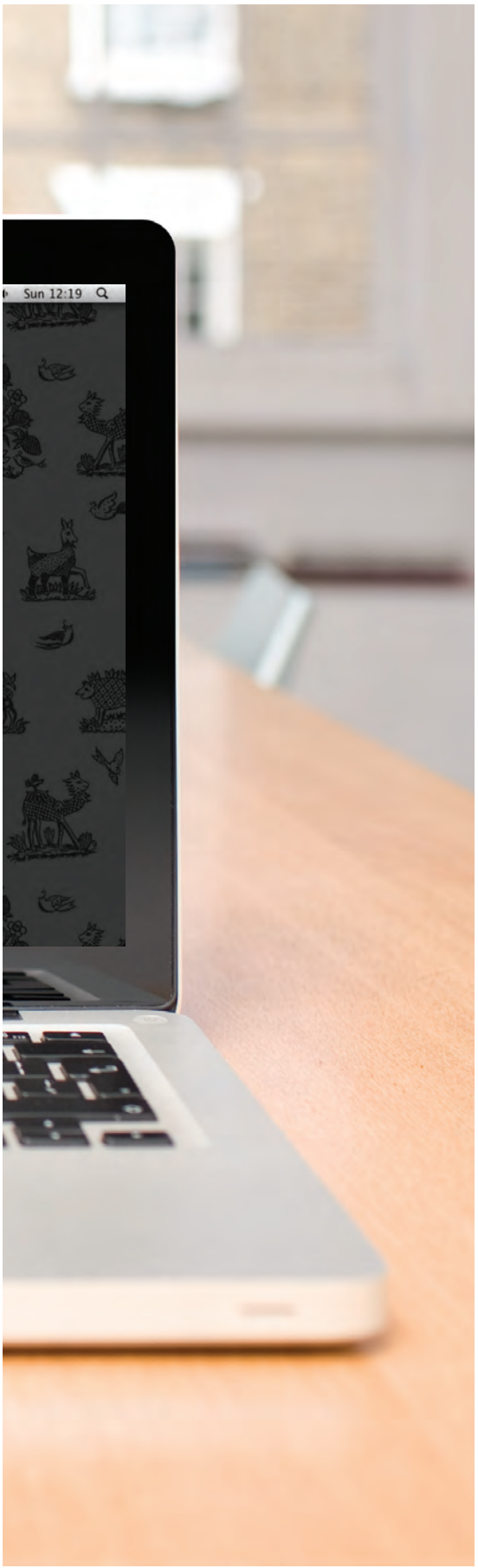
MP3: Safari 5+, Chrome 6+, IE9

Ogg Vorbis: Firefox 3.6, Chome 6, Opera 1.5, IE9

So you would need to supply two audio formats to get coverage across all recent browsers that support the `<audio>` element. You could also provide a Flash alternative for older browsers that do not support the `<audio>` element.

The HTML5 `<audio>` tag has not gained as widespread adoption as the `<video>` tag, and there have been some issues with audio quality in the first browsers to implement it.


Safari File Edit View History Bookmarks Develop Window Help
Flash, Video and Audio
http://www.htmlandcssbook.com/code/chapter-09/example.html
Google
00:18
-00:01
MacBook Pro



EXAMPLE
FLASH, VIDEO & AUDIO

This example uses HTML5 to show a video.

The video has been encoded in H264 and WebM formats to reach as many browsers as possible. A Flash player has been added to the page for browsers that do not support HTML5 video. The Flash player is embedded using SWFObject. If the browser does not support HTML5 video or Flash, then a plain text message will be shown to the user.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Flash, Video and Audio</title>
    <script type="text/javascript"
      src="http://ajax.googleapis.com/ajax/libs/
      swfobject/2.2/swfobject.js"></script>
    <script type="text/javascript">
      var flashvars = {};
      var params = {movie: "../video/puppy.flv"};
      swfobject.embedSWF("flash/osplayer.swf", "snow",
      "400", "320", "8.0.0", flashvars, params);</script>
  </head>
  <body>
    <video poster="images/puppy.jpg" width="400"
      height="320" controls="controls">
      <source src="video/puppy.mp4" type='video/mp4;
        codecs="avc1.42E01E, mp4a.40.2"' />
      <source src="video/puppy.webm" type='video/webm;
        codecs="vp8, vorbis"' />
      <div id="snow">
        <p>You cannot see this video of a puppy playing
          in the snow because this browser does not
          support our video formats.</p>
      </div>
    </video>
  </body>
</html>
```

SUMMARY
FLASH, VIDEO & AUDIO

- Flash allows you to add animations, video and audio to the web.
- Flash is not supported on iPhone or iPad.
- HTML5 introduces new `<video>` and `<audio>` elements for adding video and audio to web pages, but these are only supported in the latest browsers.
- Browsers that support the HTML5 elements do not all support the same video and audio formats, so you need to supply your files in different formats to ensure that everyone can see/hear them.