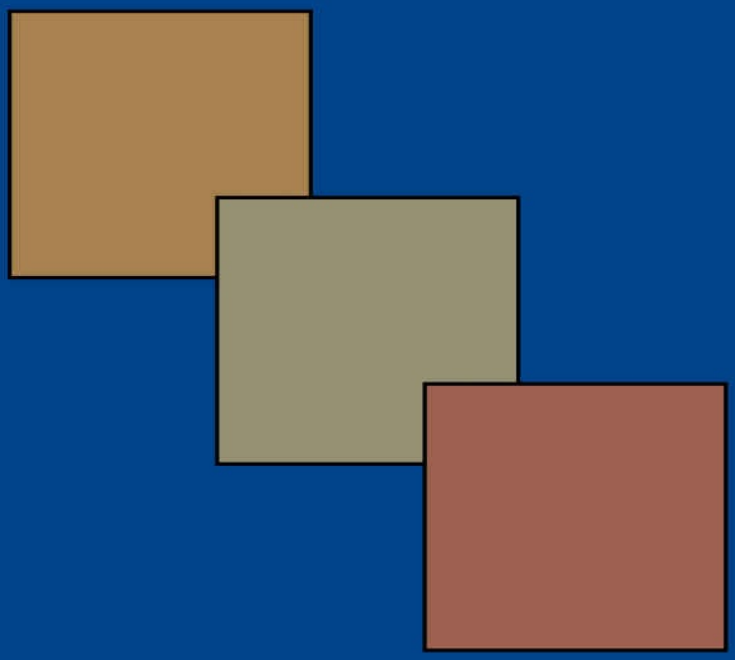
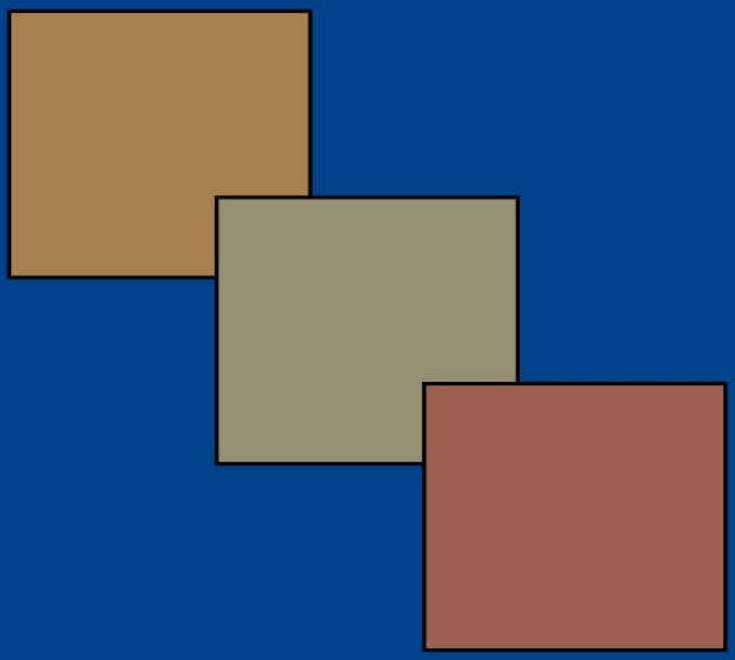


HTML Beginners

Basics of Web Design



Paul Gibbs

HTML Beginners – Basics of Web Design

Paul Gibbs

HTML Beginners – Basics Of Web Design

written by Paul Gibbs

Copyright

Amazon ASIN Number: (eBook – Kindle)

Websites, Source Code and Feedback

http://www.paulvgibbs.com
http://www.withinweb.com

Download all of the source code from:
http://www.paulvgibbs.com
or
http://www.withinweb.com/ebooks/

Email: paul@paulvgibbs.com

Trademarks

Warning and Disclaimer

Preface

This book has been written from a series of college lectures on Web Design. It is has many exercises and examples which can be downloaded from the web site. It starts with an introduction to HTML and then takes the reader through the concepts of styles and how to apply them.

There are also chapters on Graphics, JQuery, setting up a web site and looks at HTML 5 and responsive web design concepts.

The book explains issues that you encounter in real world situations and provides the basic examples from which you can then use to further develop. Many of the tutorials consists of a series of examples and tasks which illustrate each point and concentrate on simplified code so that you can see how they are used.

* Introduction - An introduction to HTML.
* HTML Tags – A look at standard HTML Tags.
* Images and Tables - Using images and creating tables.
* Styles and Stylesheets - Different ways to use styles.
* Further Styles - Definitions and further examples.
* Site Examples - A site layout.
* Lists and Menus - Creating navigation.
* Graphics – Graphic concepts and Photoshop.
* Layouts – Some principles of design.
* Adding functionality - JQuery and other features.
* htaccess file - Examples of using an htaccess file.
* Site set up – Domain names and search optimisation.
* Forms – Formatting and layout of forms.
* HTML 5 Design – Features of HTML 5.
* Responsive Design – A responsive design example.

One of the issues that we have in the computer world is that it is very much influenced by the US designers of software. Hence the word “color” is used instead of the British English spelling of “colour”. In this book I use “color” throughout. I did for a while consider using the word “color” when writing about the style attribute but use “colour” when writing about color generally, but thought that would just confuse things. So British English readers will have to put up with “color” for now.

Using the code examples

All the code examples with answers to the exercises may be downloaded to your computer from the web site:

http://www.paulvgibbs.com
or:
http://www.withinweb.com/ebook

How to copy and paste text from Kindle books

You may want to copy and paste programming code examples from Kindle onto your computer so that you don't have to re-type them.

To do this in Kindle, highlight the text you want to copy and click on the "Highlight" button. Now go on the web to your Kindle account (https://kindle.amazon.com) and click on "Your Highlights" link. You'll see all the text you've highlighted in your Kindle books. From there, you can copy the text and paste it into another program.

Some Web Design Terms

Responsive – Refers to the ability for a web page to responds to different devices and screen resolutions controlled by the selection of different styles.

Cascading – Page elements are styled depending on rules which control which style takes precedence when there are more than one style for an element.

Inheritance – Certain styles are inherited down the document tree. So for example font-family can be placed in the body style and this will mean all text on the page will have that font-family. This makes style sheets simpler.

In-line styles – A method to place styles into a web page directly against the element rather than in the head of a document or in a style sheet.

class – A class can be used to style more than one element.

id – An Id is used to style one particular element.

W3C – The international body that oversees the web specifications.

Elements – Is a component of a web page such as headings <h1>, <h2>, <h3> and when placed on a web page form part of the document tree

Tags – The less than tag is < and the greater than tag is > which are used to define the start and end of an element. For example <h1>, <p> and so on.

Attributes – Elements can have attributes which are properties which provide further information depending on the element type such as the href value for a URL link or an image name for an image element.

POST and GET - This is how you send data to another web page where you can then process the data. POST and GET work with forms where the user enters information.

Cookies - A cookie is small text file that sits on the user's computer. It is often used to

store data so that the user has a better browsing experience should they return to the web site. For example, a shopping cart system may store the items that the user has selected. When they return to the store, the contents of the cart can be displayed and the user can continue shopping.

Sessions - Another way for the programmer to retain information; however, session data is deleted and lost when the user closes the browser window.

Book Version

June 2016 is the current version

Other Books by Paul Gibbs

PHP Tutorials: Programming with PHP and MySQL

Amazon ASIN Number: B00CBLZ1U0 (eBook – Kindle)
ISBN Numbers:
978-0-9928697-0-0 (eBook – PDF)
978-0-9928697-1-7 (eBook - ePub)
978-0-9928697-2-4 (eBook – Kindle)
978-0-9928697-3-1 (Paper Back)

Table of Contents

1 Introduction

- The processing of an HTML page
- A basic HTML page
  - The type Declaration tag
  - The Head tag
  - The Body tag
- Some HTML tags
- HTML 5
- CSS 3
- Folders and Files and Servers
- Content Management Systems and HTML
- Common errors that beginners make
- Some useful web sites
- Some HTML Editors
- Summary

2 HTML Tags

- TASK 1 - Creating Folders
- TASK 2 – Create a new Web Page
- TASK 2 - Saving the Page
- TASK 3 - Inserting headings
- TASK 4 - Inserting and formatting the text
- TASK 5 - Formatting your text - The style attribute
- TASK 6 - The bold and italic <i>tags
- TASK 7 - Copying files into folders and - inserting an image
- TASK 8 - The image tag
- TASK 9 - Borders and aligning images
- TASK 10 - Adding a Horizontal line
- TASK 11 - Other formatting
- TASK 12 - Links to other pages
- TASK 13 - Anchor Links to places on the same page
- TASK 14 - Email links
- TASK 15 - HTML Special Characters
- TASK 16 - Lists and bullets
- TASK 17 - Colors in html
- TASK 18 - Page validation
- Debugging in the browser

Using the inbuilt developer tools in web browsers
Emulating different devices using Google Chrome
Firefox debugger
XHTML validate the web page
Use Dreamweaver to check the document
Google Chrome PageSpeed utility
Some Standard HTML Tags
Summary
3 Images and Tables
A Table example
TASK 1 - Creating a table
Table Headings
Simple Table Exercise
A Gallery of Pictures
TASK 2 - Creating a new images folder
TASK 3 - Adding Images
TASK 4 - Set up the Page
TASK 5 - Add the table
TASK 6 – The table
TASK 7 - Insert your images into the table
TASK 8 - Add descriptions to your images
TASK 9 - Add a horizontal line or rule <hr />
TASK 10 – Complete the formatting of your gallery page
TASK 11 Center the table
TASK 12 – Add in the email link
TASK 13 – Test your pages
Summary
4 Style and Style Sheets
In line styles
Placing styles in the head of the document
Style syntax
Creating styles with Dreamweaver
CSS - Cascading Style Sheets
Inheritance in Styles
TASK 1 – Placing styles in the head of the web page
TASK 2 - Creating your own styles
TASK 3 - Creating a style for p only
CSS id and Class
TASK 4 - An example of a selector id
TASK 5 - An example of a selector class
Some notes about Font Sizes

DIVs and SPAN
Creating a separate style sheet
The @media rule
Summary
5 Further Styles
Style Definition Summary
[1] Built in Tags (selectors)
[2] Classes
[3] ids
Style Exercises
TASK 1 - Set the page to a defined font and size
STYLES
BODY TEXT
TASK 2 - Modify an h1 tag
TASK 3 - Modify a paragraph tag
TASK 4 - Creating a quote block of text
Links
TASK 5 - Remove the underline from a hyperlink
TASK 6 - Make hyperlinks block hover
TASK 7 - Creating two types of styles for links using a div class
TASK 8 - Creating two types of styles using an href class
Images and Styles
TASK 9 - Remove a border or add a border to an image
TASK 10 - Add a shadow to an image
CSS Box Model
Block and Inline elements
Block Elements
Inline Elements
HTML Div and Span
Difference between div and span
TASK 11 - Float left and float right blocks
TASK 12 - Centre a block of text with a div
TASK 13 - Centre a block of text with two divs
TASK 14 - Floating images to the right
TASK 15 - Floating images to the left
Further Tables
TASK 16 - Creating a new table
TASK 17 - Adding in some styles
TASK 18 - Formatting the display further
Summary
6 Site Examples

Create a Business Card Web Site
TASK 1 - Create a new web page
TASK 2 - Create some DIV containers
TASK 3 - Add in main image
TASK 4 - Add in header text
TASK 5 - Add in description text
TASK 6 - Add in opening hours text
TASK 7 - Add in location text
TASK 8 - Add in prices text
TASK 9 - Add in contact text
TASK 10 - Add in footer text
TASK 11 - Format text
TASK 12 - Add in body styles
TASK 13 - Add in header styles
TASK 14 - Add in styles for each block
TASK 15 - Add in footer styles
TASK 16 - Add In Canvas Styles
TASK 17 - Add in Google map
TASK 18 - Add in logo and logo styles
TASK 19 - Add in href anchor links
TASK 20 - Add in the href links styles for the header
TASK 21 - Place styles into a separate style sheet
Files for Business Card Web Site
Minify CSS
[A] - Create a separate style sheet
[B] - Minify the css
Summary

7 Lists and Menus

Unordered lists
Ordered Lists
Using images in lists
All CSS List Properties
Create a menu using an Unordered List
Example of a menu in a web page
CSS Mouseover Buttons
Summary

8 Graphics

Graphic formats used on the web
gif format
jpg format
png format

- bmp format
- tiff format
- Photoshop basic principles
  - TASK 1 - Creating shapes in Photoshop
  - TASK 2 - Using text in Photoshop
  - TASK 3 - Saving for a web page
  - TASK 4 - To create a button with Photoshop
  - TASK 5 - Using Layers
- Checking the size of images
- Creating roll over images
  - Using Dreamweaver to create a roll over image
  - Create two buttons using Photoshop
  - TASK 6 - Creating a roll over using CSS
- Exercise in Background Images
  - TASK 7 - Background body image
  - TASK 8 - Creating a background image
  - TASK 9 - Background position
- Preparing and saving images for the web
  - [1] Bring an image into Photoshop
  - [2] Set Monitor RGB
  - [3] Crop the image
  - [4] Re-size the image
  - [5] Create the re-sized image
  - [6] Filtering the image
  - [7] Sharpen your image
  - [8] Alternative method for sharpening
  - [9] Save the file
  - [10] Add your image to a web page
- Summary

9 Layouts

- Page Sizes
- General layouts concepts
  - Traditional Aspects:
  - Site details:
  - Color theory?
  - How to select color schemes
- Web Design - Good Practice
  - Good Web Pages
  - Bad Web Pages
- Responsive Design
- How do we find a layout for our site?

Some Free Template Sites
Responsive Design Frameworks
A Web Site Project
Web site specification
A web site consisting of 6 pages
General considerations
Layout choice
Using Wire Frame design
Suggested Pages
[1] index.html page
[2] Products or services page
[3] FAQ or Customer Services Information
[4] Contacts page
[5] Gallery page
[6] About us page
Site map page
Site navigation
Site optimisation and submission to search engines
Things to consider
Where do I get my images from?
How many pages should the site have?
What layout should the site have?
How do I research other similar web sites?
Summary
10 Adding Functionality
Web Fonts from Adobe and Google
Google Fonts
Adobe Edge Web Fonts
TASK 1 – Add some fonts to your own web page.
Introduction to JavaScript
What is JavaScript?
What is JavaScript used for?
How do we use JavaScript?
A simple example of JavaScript
Using JQuery
What is JQuery?
Uses of JQuery
Incorporating JQuery
An illustration of JQuery
UI Accordion interface
Menu Interface

Date Picker
Adding movies to a web pages
Adding audio to a web pages
Summary
11 .htaccess Files
Things that can be done with an .htaccess file
To create an .htaccess file
Examples of .htaccess files
301 is permanent redirect
302 is temporary redirect
Defining a custom 404 error page
Stop users displaying the directory listing of a web site
Stop users displaying the details of an .htaccess file
To protect your admin area you can create an .htaccess / .htpasswd file.
Create an .htaccess file
Create the .htpasswd username / password file
Page Status Codes
Summary
12 Site Set Up
Domain Names and Web Space
Domain Name
Considerations for TLD
Web Space and Web Servers
Things to consider when looking for web hosting
After purchase of web space
Notes on Page Set Up
Page details
Site Navigation
Search Engine Optimisation (SEO)
SEO site tasks
Google Analytics
To create a new View (Profile)
Filter data by subfolder
To see Google search engine words
To display real time
To display % of use showing which links are being used
Tools for Analysing and Optimising Pages
Other facilities for testing web pages
Structured data mark-up
An introduction to Microdata
Tool for creating Microdata

Testing your Microdata
Summary
13 Forms
Basic Form Structure
Dreamweaver and Forms
Form elements
[1] Text
[2] Password
[3] Hidden
[4] Textarea
[5] Checkbox
[6] Radio Button
[7] Dropdown List
[8] Email input box (HTML 5)
[9] Submit Button
[10] Reset Button
[11] Image Button
Form Controls
Example HTML email form
Issues with email forms
Forms in HTML 5
HTML 5 and form validation
TASK 1 - The required attribute
TASK 2 - Input type of “number”
Formatting Forms in HTML 5
TASK 3 - Create a form
Summary
14 HTML 5 Design
Exercise in HTML 5 Layout
TASK 1 - Create a new HTML 5 web page
TASK 2 - Add in basic structure
TASK 3 – Add the first styles
TASK 4 - The wrapper and header styles
TASK 5 - The h1 tags
TASK 6 - Add in all text
TASK 7 - Navigation styles
TASK 8 - The section and article schematic mark-up tags
TASK 9 - The figure and hgroup styles
TASK 10 - the aside styles
TASK 11 - Add in the final group of styles to finish off the links, h tags and footer
TASK 12 - Add in htmlshv JavaScript code

Summary

15 Responsive Web Design (RWD)

The importance of mobile design

Media Queries

A Responsive Web Design Framework example

TASK 1 - Download the HTML- KickStart framework

TASK 2 - Copy the file blank.html and call it index.html

TASK 3 - Page title

TASK 4 - Add in the basic grid structures

TASK 5 - Add in text for logo for header menu

TASK 6 - Add in styles

TASK 7 - Add in the header style

TASK 8 - Add in link styles

TASK 9 - Add in Logo styles

TASK 10 - Add in the slide show

TASK 11 - Add the text to the three column blocks

TASK 12 - Add in some styles for the Three blocks

TASK 13 - Add in text and image for the description block

TASK 14 – Add in the image which is part of the description DIV

TASK 15 - Styles for description

TASK 16 – Add in text for Opening Hours

TASK 17 – Add in Styles for Opening Hours

TASK 18 - Add in text for footer

TASK 19 - Add in styles for refs at the bottom of the page

TASK 20 - Finish off the page

Summary

1
Introduction

What's in this chapter:

* HTML Processing.
* A Basic HTML Web Page.
* Some useful websites.
* A list of web editors.

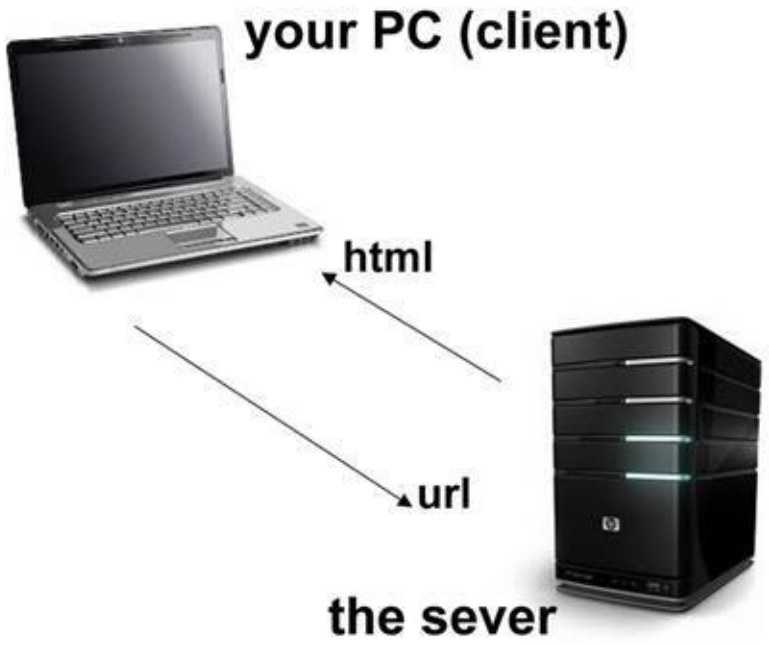
The acronym HTML means "**H**ypertext **M**ark **U**p **L**anguage" which essentially means that text is tagged in a particular way to represent information that a web browser can understand.

A web page is nothing more than an ordinary text file and you can create and edit web pages using a text editor like NotePad or NotePad++. However, there are a number of commercial editors such as Dreamweaver or CofeeCup. There are also a number of free editors such as Kompozer which also has the ability to create and edit HTML documents.

Creating web pages is straight forward and you can display your results in a web page by simply right clicking on the file and opening it using one of your installed web browsers. So you do not need to have access to a web server to develop web sites.

The processing of an HTML page

If you have a site which consists only of HTML web pages, the server merely sends the HTML data to the web browser as shown in the figure below.



This is why HTML pages can be viewed locally in your web browser from your own computer since they do not need to be “served,” whereas dynamically generated pages such as PHP need to be accessed through a server which handles the processing.

A basic HTML page

An HTML page adheres to defined standards which you need to meet or otherwise the page will not display correctly. The point of having these standards is that you know the page will look more or less the same in whatever browser you use. If you don't follow the rules you can't be sure it will look the same on different browsers.

A basic web page consists of:

- declaration
- head
- body

So we start with a simple web page as follows:

```
<!doctype html>
<html>
<head>
<title>A web page</title>



</head>
<body>

The body text goes here


</body>
</html>
```

The type Declaration tag

The <!DOCTYPE> declaration is not an HTML tag; it is an instruction to the web browser about what version of HTML the page is written in.

It always appears at the top of the page. In this book we will be using HTML 5 and the doc type for that is

<!doctype html>

If you look at other web pages, you will come across other type declaration such as:

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

which is the doc type for html 4.01. As you can see, html 5 has simplified this entry.

The Head tag

Within the <head> .. </head> tags we place elements which don't normally display anything on the web page. This includes the title tag, styles for the document, JavaScript

code, meta tags, and others.

The Body tag

The body tag is where the main body of the document is placed and this is where we put your text, graphics and all the web page structure. The body is placed after </head> and finishes before the </html>.

Some HTML tags

HTML is actually XML which is a tagged mark-up language where the tags infer meaning. All tags have a start and an end and here are some examples:

<title>……..</title>

** …… **

** ……. **

<td>…….</td>

As the following shows, the tag consists of the tag name with optional arguments.

< tag_name tag_attributes(optional) >……</ tag_name >

HTML defines the structure of the document such as if it is a heading or if it is a paragraph. Styles (or CSS) define the formatting of the page, like the font type, size or color. Using this method separates the structure of the web page from the formatting and is the basic concept of a web page using HTML and CSS.

HTML 5

For a number of years web standards used various forms of HTML 4.

The HTML 5 specification was finally completed in 2014 although many browsers already incorporated elements of the specification so there was no real delay before developers could start to use it. Now, the latest browsers largely support the specification.

HTML 5 takes over from all the previous specification and is the one that should be used from now on.

HTML 5 incorporates many new features particularly for **multi-media** and **graphical content** and adds a new page structure using "**schematic mark-up elements**" to make layouts easier to understand.

CSS 3

CSS 3 is the latest version of the Cascading Style Sheet specification and is responsible for the presentation of the elements.

Using CSS means that content on the web page can be separated from the presentation making it possible to change the styles so that the same content can be presented with a different look.

CSS 3 is not one single specification but is made up of a number of modules. There are over 50 of these modules each of which are in various states of status and stability. From the point of view of the web developer the majority of attributes that you will want to use are available on the majority of the latest web browsers. However, there will be some features that will not work on certain browsers and there are sites such as:

http://caniuse.com/

which show browser compatibility.

Folders and Files and Servers

When you create your web pages you have to give them file names of course. There are a couple of things that you should be aware of when you decide on your file and folder names.

First, there are essentially two types of web server: Linux and Windows based. The file and folder names on Linux web servers are case sensitive, while windows are not case sensitive.

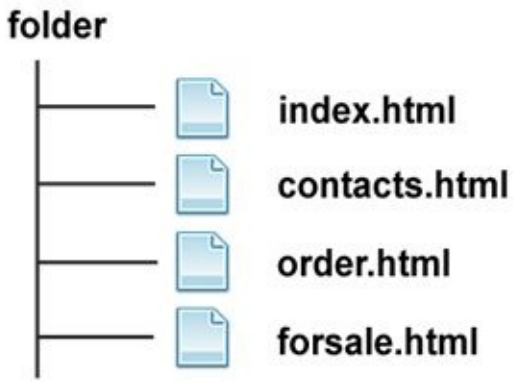
The recommendations for filenames and folders in both types of servers are:

no spaces
lowercase
alphabetic characters (a – z)

so don't use "odd" characters like quote marks, brackets and so on.

The file extension for a web page should be **htm** or **html** You may come across other file extensions such as php for the PHP programming language, or .aspx for the dot net programming language, but for all our work we will be using **htm** or **html**

Web sites can consist of a very large number of files and folders and normally we create folders to make logical structures. A simple web site with one folder may look something like the following:



Note that for **each** folder there will always be a default page and this default page will have a file name of one of the following:

Index.html
or **index.htm**
or **default.html**
or **default.htm**

The default page is the "home" page for that folder or the first page that the visitor goes to. In our example above, when the visitor enters in http://www.sitename.com the server "knows" to send the visitor to index.html even though the visitor did not explicitly identify that in the requested URL

Content Management Systems and HTML

Content Management Systems (CMS) are web applications that are usually database driven and where you create web pages through a web admin interface. Some examples of popular CMS's are WordPress, Joomla, and so on.

To get the most out of a CMS system you do need to have an understanding of HTML and CSS particularly if you want to do more than just place text and graphics on the pages.

Common errors that beginners make

I have taught HTML over a number of years to students of various standards but there are a number of problems that students continuously seem to have:

- They don’t remember the basic structure of a web page and place things in the wrong location. For example, they place the web content into the head of the page instead of the body and so nothing displays.

- They don’t close off tags properly and don’t use the tools in their editor to see where their mistakes are. This can result in text picking up incorrect styles or losing the page structure.

- They are not precise enough when they enter in their styles, missing out a (:) symbol or (“) symbol which causes the styles not to work.

Practice helps to overcome these problems. Also, using a good web editor helps as they provide code hints and display error messages if something is not correct.

Some useful web sites

Microsoft DreamSpark
http://www.dreamspark.com/

This site allows you to download many Microsoft Products for free if you are a student or a teacher. This is done by self-verification where you use the email address of the college or school where you are registered. If you use your student or staff email address you can download Microsoft Software.

HTML Reference
http://www.w3schools.com/html/default.asp

This site has become very popular and has an extensive range of reference material covering HTML, CSS and many programming languages such as PHP.

Can I use
http://caniuse.com/

A useful web site that shows which effects can be used on which browser.

10 minute guide on html
http://www.w3.org/MarkUp/Guide/

Web Design course
http://www.homeandlearn.co.uk/wd/WebDesign.html

Beginners PHP
http://www.homeandlearn.co.uk/php/php.html

CSS Positioning of DIVs
http://www.barelyfitz.com/screencast/html-training/css/positioning/

Float tutorial
http://css.maxdesign.com.au/floatutorial/

Wire framing design
http://webdesign.tutsplus.com/tutorials/workflow-tutorials/a-beginners-guide-to-wireframing/

Web site on “Tools and resources for web professionals”
http://westciv.com/

Google Developer site
https://developers.google.com

HTML 5 Rocks
http://www.html5rocks.com/

Some HTML Editors

BlueGriffon
http://bluegriffon.org/pages/Download

A free editor

Visual Web Developer Express (Free from Microsoft)
http://www.asp.net/vwd

This has an HTML and CSS editor and also a number of other development features for programming.

CoffeeCup (Commercial)
http://www.coffeecup.com/free-editor/

Free version does not have ftp and other extras

HTML editor which has many features similar to Dreamweaver.

HTML Kit (Commercial, but free if you install previous version)
http://www.htmlkit.com/

Kompozer (Free to download)
http://www.kompozer.net/

Code anywhere
https://codeanywhere.com/

On line editor but has no visual design tools

Sublimetext
http://www.sublimetext.com/

This text editor is free with an unlimited trial version but you have to pay for the version with ftp.

It has a color coded display which is good for coding in html and php but has no facility to run the code in a web server in the way that Dreamweaver does, nor does it have any layout facilities like Dreamweaver.

WebMatrix
Microsoft free code editor, good for PHP but has no real design tools. The latest version is WebMatrix 3 - It is really intended as a programmer's tool and is not really all that good for page layout.

Microsoft Expression Web (Free)

http://www.microsoft.com/en-us/download/details.aspx?id=36179

This editor will not be further developed by Microsoft - up to a few months ago it was a product you had to purchase but now is free to download.

Apanta Studio
http://www.aptana.com/

Apanta is an IDE which is used for HTML, CSS, and programming as well. It is a free to download application and has a good support community.

Brackets
http://brackets.io/

Is an open source code editor developed by Adobe. The editor has the ability to preview changes that you make in the web page as you make them.

Summary

This chapter gives an introduction to web pages using HTML. You can create web pages using a text editor but there are many commercial and free editors that make page development much easier. The chapter introduced the basic structure of a page and illustrated tag structure which will be looked at in more detail in the next chapter.

2
HTML Tags

What's in this chapter:

* A basic web page
* Looks at the basic HTML tags
* Debugging using a browser
* Some standard HTML tags

This chapter is an introduction to many of the basic element tags that are available in HTML and which make up a web page.

In the last chapter we looked at the way in which an HTML element is defined:

< tag_name tag_attributes(optional) >……</ tag_name >

Note that each element consists of a tag name enclosed in < > brackets. It is a requirement of a valid HTML document that all tags are properly written.

TASK 1 - Creating Folders

The first thing to do is to create a set of folders where you can store all your files. I would suggest that you create a new folder for each chapter, e.g. chapter01, chapter02 and so on.

You may want to us a flash pen drive as the files are quite small and it will mean you can work on the on different computers.

TASK 2 – Create a new Web Page

The following illustrates a basic web page:

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Untitled Document</title>
</head>

<body>

</body>
</html>
```

Code file: 02_HTML_Tags/files/basic_html_page_01.html

In Dreamweaver, to create a new web page, select File -> New which will list a set of templates. These templates are just layouts that Dreamweaver have kindly created for us. However, we just want to start with a Blank Page, so make sure that the page type of HTML is selected and a layout of none.

You should also select a Doc Type of HTML5.

The Doc Type is simply a way to tell the browser what specification the web page will be using so that the browser will be able to interpret the tags correctly.

Note that the web page content goes in between the **body** tags. One common error that is often made is to insert the web contents in the head of the page.

Note that the above page is completely blank as it has no body content.

HTML consists of tags defined with a start and end tag. Different tags may have attributes such as width and height for an image.

The Head and Body Elements of a web page:

<!doctype html>

<head> tag -

<body> tag -

<title> tag -

TASK 2 - Saving the Page

Save the file with the name (**basic_html_page_01.html**) and the <title>A basic web page</title>.

TASK 3 - Inserting headings

Now we modify our page by adding in some heading tags and paragraph tags.

The Heading tags <h1>…</h1> is the biggest <h6>…</h6> is the smallest

So we can try out h1, h2, h3, h4, h5, h6 and see how they appear on the page.

Insert headings onto your page between the opening <body> and closing </body> tags so that it looks something like the following:

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
```

```
<title>A basic web page</title>
</head>

<body>

<h1>This is the main heading using h1</h1>

<p>This might be an introduction to the rest of the page.</p>

<h2>This is a subheading using h2</h2>

<p>More text to describe the subheading</p>

<h3>Another sub sub heading using h3</h3>

<p>Some more text describing the subheading</p>

<h1>This another main heading using h1</h1>

</body>
</html>
```

Code file: 02_HTML_Tags/files/html_02.html

View your page in a browser. If you are using Dreamweaver, press F12 to display the page in the browser.

TASK 4 - Inserting and formatting the text

This task is an exercise in taking some text and formatting it for the web. You will copy some text from a text file and then add in <p> and <h1> tags. You will need to have access to the book download files.

[1] Create a new HTML 5 web page and save it with a file name of **article.html** into a folder called chapter02.

[2] Change the <title> of the web page to “Giant Snail Found”

[3] Copy and insert the text from the file **/02_HTML_Tags/files/html_text_1.txt** from the downloaded files for the book and place the text below the <body> tag.

[4] Format the text so that the heading has <h1> tag and the paragraphs have <p> tags

The paragraph tag is as follows:

Some text

Important - Always close tags, to ensure well-formed HTML.

All HTML has start and end tags and all must be closed.

However, there are some tags which have the closing tag built in so for example

is a line break. This is called a **singleton**. It is correct HTML but the format is slightly different to a tag such as …

Format the imported html text using heading tags and paragraph tags. Note that when text is imported into the code view, the text will just wrap without formatting. You have to add your tags to make the page look correct.

[5] Save your file and display it in a browser. You can see this page in the file **article.html**

Code file: 02_HTML_Tags/files/article.html

TASK 5 - Formatting your text - The style attribute

HTML is a mark-up language which consists of tags that define the structure of the document. Styles (which are often placed in separate style sheets) define formatting attributes of tags, such as color or font size. We will quickly show what a style is but we will look at it in more detail later on.

[1] In **article.html** modify the color, font-family and font-size of an h1 tag by adding in some style attributes.

Example:

<h1 style="color:blue; font-family:Verdana,Arial; font-size:20pt">Florida experiences Giant Snail infestation</h1>

Modify an <h1> tag as shown above in the code view.

The above <h1> tag has been modified to add attribute tags. This method of adding styles is called "In Line Styles". Later on we will see that we can add styles to the top of the page which is a better method.

The structure of an in-line style is:

style=

and then there is the name of the style, followed by the value.

Note that when you define a font face, the browser will only display it if it is installed on that computer. Normally we restrict ourselves to Arial, Helvetica, Verdana, Times New Roman and Courier although there are ways of adding other fonts which we will see in a later chapter.

You can see this page in the file **article_02.html**

Code file: 02_HTML_Tags/files/article_02.html

TASK 6 - The bold and italic tags

In **article.html** try out bold and italic tags.

To make text bold, use …..

To make text italic, use …

TASK 7 - Copying files into folders and - inserting an image

In this task we will make use of an image.

First create a new folder called “images” inside the folder where the file **article.html** is located, which you have called chapter02.

Copy the image snail.jpg image into the “images” folder. You will have to have access to the book download files.

TASK 8 - The image tag

[1] In **article.html**, add in the snail.jpg image under the <h1> main heading

The html tag for an image is:

So insert the image file – “images/snail.jpg” This assumes that the image is in the images folder.

The image tag starts with img and then it has a number of attributes. src, title width, height, alt and so on.

In code view, you can type in the above code.

alt and title attributes

Note the extra attributes of alt and title.

alt means “alternative” and this is used by audio browsers for people who have poor visibility. The alt attribute text is read out by the audio browser.

title is a tool tip text where the text is displayed as you hover over the image.

Note that if you are using an editor such as Dreamweaver, it will probably have the ability to drag and drop images on to the page where it automatically creates the code for you.

You can see this in the file **article_03.html**

Code file: 02_HTML_Tags/files/article_03.html

TASK 9 - Borders and aligning images

Amend the image to add a border.

You can add a border to an image by adding the border attribute to the image element as follows:

```
<div align = “center”>
<img src = “images/snail.jpg” title = “A snail” alt = “A snail” border=”5” />
</div>
```

You can see this in the file **article_04.html**

Code file: 02_HTML_Tags/files/article_04.html

TASK 10 - Adding a Horizontal line

Horizontal lines are sometimes used to separate parts of a page.

The basic form of a horizontal line is: <hr/>

Or we might do:

<hr size=“5” width=“100%” />

Place a horizontal line at the bottom of your page as shown in the file **article_05.html**.

Code file: 02_HTML_Tags/files/article_05.html

TASK 11 - Other formatting

I have added a few extra formatting, some of which we have not covered yet.

For the heading I have used:

style=”color:blue; font-family:Verdana, Arial; font-size:20pt; background-color:#D0CDCD;text-align:center;”

This gives a background color of light grey to the heading.

I have also used a div to center a block:

…..

And added around the image to align the content.

Note the difference between align=center for the image and the centering the whole div

above.

We have not looked at what a div is yet but it is actually a container for our HTML code which we can use to position on the page. We will see later that divs are used to control the layout of pages and give the structure.

You can see this in the file **article_06.html**

Code file: 02_HTML_Tags/files/article_06.html

TASK 12 - Links to other pages

In this task we will create hypertext links to other web pages.

We use the anchor element <a> to define a hyperlink to move from one page to another. The <a> tag uses the href attribute which is the hyperlink reference to the required page which can either be the full url of the page if it is not on the same server, or it can be a relative url.

Links to web pages on other web sites:

This is a link to my web site

Or we can add a target attribute as follows:

This is a link to my web site

Links to web pages on the current web site:

This is a link to a page on this site

You can do this either typing the code manually or if you are using Dreamweaver you can highlight the text and point to the file.

NOTE that you can link from other things other than text. So you can have a hyperlink around an image as well.

If you try placing the snail.jpg image on your page, you can create a link around it in the same way:

<img scr="snail.jpg">

In our previous web page **article.html** we can add a link at the bottom referencing an external web site.

<p>For information on keeping snails as pets, refer to

 http://www.petsnails.co.uk

You can see this in the file **article_07.html**

Code file: 02_HTML_Tags/files/article_07.html

TASK 13 - Anchor Links to places on the same page

Sometimes you want to have a hyperlink to another point on the same page. This is done by anchor links.

At the point where you want to move to in the document put something like:

This is the anchor tag.

To reference the tag you put a hyperlink using the # character as follows:

Back to top

You can see this in the file **article_08.html**

Code file: 02_HTML_Tags/files/article_08.html

TASK 14 - Email links

Add an email link onto your page with some appropriate text:

Email me and give feedback

You can see this in the file **article_09.html**

Code file: 02_HTML_Tags/files/article_09.html

TASK 15 - HTML Special Characters

Certain characters are not available on a normal keyboard. These are such things as mathematical symbols, Greek symbols and other special characters such as copyright and register symbols.

HTML character codes are listed here:

http://www.ascii.cl/htmlcodes.htm

You can either type in the HTML number or the HTML name. So for copyright symbol you can either type in © or ©

In Dreamweaver select Insert -> Html -> Special Characters

Code file: 02_HTML_Tags/files/article_10.html

TASK 16 - Lists and bullets

We look at an example of lists in our next web tutorial page but we can show the technique here.

[1] Create a new web page with a file name of **lists.html**

[2] Try out these examples of lists, either creating them by hand or using Dreamweaver or other editor.

An ordered Lists:

```
<ol>
<li>first item </li>
<li>second item </li>
<li>third item </li>
</ol>
```

An unordered lists

```
<ul>
<li>first item </li>
<li>second item </li>
<li>third item </li>
</ul>
```

You can change the type of bullet in the list using styles:

```
<ul>
<li style="list-style-type:circle">first item</li>
<li style="list-style-type:circle">second item</li>
<li style="list-style-type:circle">third item</li>
</ul>
```

You can see this in the file **lists.html**

Code file: 02_HTML_Tags/files/lists.html

TASK 17 - Colors in html

Colors are identified either by hexadecimal (hex) values or by color names. Not all colors have color names so you will have to get use to hexadecimal values.

Hex values are in the order Red, Green Blue and are represented by a # symbol followed by three sets of numbers. A hex number is to base 16.

#XXYYZZ

XX is the hex value for Red, YY is the hex value for Green and zz is the hex value for blue.

So #000000 is black

#ffffff is white

#ff0000 is red

Each color can have 256 variations so the red has 256 variations, as does green and blue. So the total number of colors possible in html is over 16 million combinations.

In the early days of the web, there was the concept of “web safe” colors which were 256 colors that could be represented. The limitation was because graphics and video cards could not display more than 256 colors. The term is not really relevant now although you may come across the limited palette of web safe colors.

Color names:

Weblink: http://html-color-codes.info

HTML editors usually have a way to select the correct color code and in Dreamweaver that is done using the page properties windows.

TASK 18 - Page validation

http://validator.w3.org/

Use the above link to validate your html code.

You can also validate pages within Dreamweaver

First make sure your page is saved to your PC.

Select File -> Validate -> Markup

Make sure that the Validation tab is displayed.

Click on the Green arrow to select the current document or the local site.

Any HTML errors will be displayed on the Validation Tab.

If you display a page in your browser and deliberately cause an HTML error such as removing a </p> then this should get flagged as an error.

Debugging in the browser

How do you debug a problem with a web page if the layout does not appear in the way you want it to or a style is not correct? These issues are quite difficult to do but now there are in-built debugging facilities in browsers that help.

Using the inbuilt developer tools in web browsers

Most modern web browsers have a developer's tool system which is normally accessed by pressing F12.

Internet Explorer, Firefox and Google Chrome all have this built into the web browser, however each are slightly different and have slightly different features.

So in Google Chrome, display a web page in the browser and press F12.

* Make sure that the 'Elements' tag is selected,

* If you select a particular element on the web page and right click and then click "Inspect Element", it will show the details of that particular element. On the right hand side of the page is the styles for that element.

* Note that some styles may be crossed out.

Styles can be quite complicated and consist of inherited styles, so this display is an attempt to show what styles are applicable to a particular element taking into account other style definitions in the web page.

If an element is not displaying a particular style, then this may be the way that you can see if the style is available for that element. You can also edit and add styles on the page by click on the (+) new element style rule. However, these changes will not be saved to the page.

Emulating different devices using Google Chrome

Google chrome can be used to display the web page using emulation. First press F12 and then press the **Escape** key which displays the Emulation mode. In this screen we can choose a Device model.

An interesting point that F12 shows is that you can have HTML which is incorrect but the browser automatically corrects it. So with the following table example there is a </td> missing. If you show this in the browser and press F12, you will see the browser automatically adds in the </td> tag for you so that the table is not broken.

```
<table>
<tr>
    <td>1</td>
```

```
      <td>2</td>
</tr>
<tr>
    <td>3</td>
      <td>4
</tr>
</table>
```

Weblink: https://developer.chrome.com/devtools/index

In particular see https://developer.chrome.com/devtools/docs/ network which describes the meaning of the network color bar displays.

Firefox debugger

Firebug is a Firefox extension which can be found here:

Weblink: https://addons.mozilla.org/en-US/firefox/addon/firebug/

It provides very similar facilities to those found on Google Chrome allowing you to inspect and debug and change values.

XHTML validate the web page

You can validate a web page at http://validator.w3.org/ which will indicate page errors in the HTML such as incorrect HTML syntax, missing arguments and so on.

* Validate by URI
* Validate by file upload
* Validate by direct input

Validation is the process of verifying if the web page actually follows the rules of the mark-up language. So it will check if it matches the required HTML syntax. When you submit the code to the validator, it is able to understand what version of HTML it is using and hence provide the appropriate tests.

Use Dreamweaver to check the document

Most web editors environments like Dreamweaver or CoffeeCup will have some way to check that you have the correct number of tags and have closed tags correctly.

Web editors do this in different ways, some may highlight it as you type.

Dreamweaver has a way to validate the mark up – If you create your page and then in

Code view select File -> Validate -> Mark Up

Or press Shift-F6 will check your html code.

You can test this by entering a tag without a closing tag. So if you enter and then press Shift-F6 it should display the location of the error.

Google Chrome PageSpeed utility

A useful feature of Google Chrome is the PageSpeed utility.

Load up a web page in Google Chrome and Click on ‘PageSpeed’ tag.

Click on Analyse and Google will return a list of suggestions to improve the page loading times.

Some Standard HTML Tags

The following is a list of some of the common HTML tags that which we have used so far in this chapter:

<!doctype html>	HTML 5 Doc type declaration. Not actually an HTML tag but included for completeness.
<title>……</title>	Defines title of the web page
<h1>, <h2>. <h3>, <h4>. <h5>. <h6>	Standard heading tags
…	Paragraph tag
 	A line break
….	Bold text
…..	Italics
link text	Links
	Images
  - Coffee - Tea - Milk 	Un-ordered list
  - first value - second value - third value 	Ordered list

Summary

This chapter introduced many standard HTML tags that you will use in a web page. It used a number of web examples to show these work. The chapter has introduced some concepts of styles but only briefly. Styles will be further explained in the next chapter.

3
Images and Tables

What's in this chapter:

* Crating a table example.
* Table headings
* A table of images.

Tables are an important part of web design. Tables were used for general page layout but that has been taken over by DIVs and stylesheets.

However, tables are still very much used where there is a requirement for displaying data in rows and columns. With CSS3 we can add in styles that greatly improve the look of tables.

A Table example

TASK 1 - Creating a table

[1] Create a new web page called **table_01.html** in a folder called chapter03

[2] Create a table using the following HTML code:

```
<table width="200" border="1">
  <tr>
    <td> </td>
    <td> </td>
    <td> </td>
  </tr>
  <tr>
    <td> </td>
    <td> </td>
    <td> </td>
  </tr>
  <tr>
    <td> </td>
    <td> </td>
    <td> </td>
  </tr>
</table>
```

[2] If you are using Dreamweaver you can create a table in DESGIN view by **Insert -> Tables.** This will display a dialog box which if you press OK will create a table of 3 columns by 3 rows similar to the above.

From the above we can see that <tr> is used to define **ROWs** and <td> is used to define **COLUMNs**.

The ** ** is just a not breaking space to pad it out.

Note that setting border="1" helps us to see the structure of the table which we may set to border="0" once the table has been completed.

[3] Try creating other tables with more rows or columns and change the border width to see what happens.

[4] Try changing the width of the table to 100% so that it extends over the full width of the page.

[5] Try merging cells together using the colspan attribute as follows:

```
<tr>
   <td colspan="3"> </td>
</tr>
```

In this example we are merging all three columns.

Table Headings

Table headings <th> is the first row of a table and contain the titles of the columns. Table headings are optional and are formatted to be bold and centred by default.

```
<table>
   <tr>
     <th>Column 1 heading</th>
     <th> Column 2 heading</th>
     <th> Column 3 heading</th>
   </tr>
```

Table headings give a way of applying a different style to the first row of a table.

You can see an example of table headings and a basic table layout in the file **html_01.html**

Code file: 03_Images_and_Tables/files/table_01.html

Simple Table Exercise

This exercise is to create a table that looks like the following:

Comparison from 1999 to 2014	
1999	**2014**
Pentium III 600 MHz processor	Intel i5 2.4 GHz Processor
128 Mbyte Ram	8 Gbyte Ram
15 Gbyte Hard Drive	500 Gbyte Hard Drive
3.5 inch disc drive (1.4 Mbyte)	
CD-Rom drive	DVD Read / Writer

[1] Make a table with 7 rows by 2 columns.

[2] Merge the top row together using colspan or use Dreamweaver merge table row facility.

[3] Replace the td for the top two row by **th** which automatically centres and bolds the text. If you are using Dreamweaver, the top row may already be set as **th** depending on your settings.

[4] Set the width of the table to 800px as follows:

<table width=”800px”>

and set the two element widths to 400px each:

<th width=”400px”></th>
<th width=”400px”></th>

[5] Enter some text into the fields - there is some sample text provided but you can put in any text you want.

The following is the complete html for the table:

```
<table width=”800px”>
  <tr bgcolor=”#CCCCCC”>
     <th colspan=“2”>Comparison from 1999 to 2014</th>
  </tr>

  <tr>
```

```
    <th width=“400px”>1999</th>
    <th width=“400px”>2014</th>
  </tr>
  <tr>
    <td>Pentium III 600 MHz processor</td>
    <td>Intel i5 2.4 GHz Processor</td>
  </tr>
  <tr>
    <td>128 Mbyte Ram</td>
    <td>8 Gbyte Ram</p></td>
  </tr>
  <tr>
    <td>15 Gbyte Hard Drive</td>
    <td>500 Gbyte Hard Drive</td>
  </tr>
  <tr>
    <td>3.5 inch disc drive (1.4 Mbyte)</td>
    <td></td>
  </tr>
  <tr>
    <td>CD-Rom drive</td>
    <td>DVD Read / Writer</td>
  </tr>
</table>
```

Code file: 03_Images_and_Tables/files/table_02.html

A Gallery of Pictures

This is an exercise in creating a gallery of images using a structure of a table.

TASK 2 - Creating a new images folder

This exercise will be in a folder called **tutorial03** which you may have already created. You will also need a folder INSIDE this folder called **images**

TASK 3 - Adding Images

Copy six images of your choice from the chapter03 **images** folder and place them into your new **images** folder**.** Rename them if you wish in order to make them more easily identifiable.

TASK 4 - Set up the Page

Create a new web page called **gallery.html** and add a title to the <title> tag.

<title>Image Gallery**</title>**

Save the page.

TASK 5 - Add the table

Now add the table to your page.

In Dreamweaver, you can insert a table by **Insert** -> **Table** Set the **border thickness** to 0 and select the **header** as none. You need a table of **7 rows by 5 columns**.

Refer to the file **gallery_base_table.txt** which shows the basic table if you wish.

Code file: 03_Images_and_Tables/files/gallery_base_table.txt

TASK 6 – The table

	col 1	col 2	col 3	col 4	col 5
row 1					
row 2					
row 3	1		2		3
row 4					
row 5	4		5		6
row 6					
row 7					

The top row is where the heading should go. Merge the row into one row and add a centred title as follows:

<h1 align = “center”>Flower Gallery</h1>

TASK 7 - Insert your images into the table

Insert and add your chosen gallery images to the **3rd** and **5th** Rows, in the **1st**, **3rd** and **5th** columns.

The HTML will look something like this. For each image you will want to enter the alt text and the title text.

```
<tr>
<td> <img src=“images/file_name.jpg” alt =”?” title=”?” /> </td>
<td>   </td>
<td> <img src=“images/file_name.jpg” alt =”?” title=”?” /> </td>
<td>   </td>
<td> <img src=“images/file_name.jpg” alt =”?” title=”?” /> </td>
</tr>
```

You may want to change the width of the images. Try adding **width=”200”** to each of the images. If you leave out the width, the image will automatically be adjusted to maintain the correct aspect ratio.

****

NOTE that you can set the properties of an image in Dreamweaver, if you are displaying the **Properties** window at the bottom of Dreamweaver.

TASK 8 - Add descriptions to your images

Add an image description for each of the images on the **4th** and the **6th** rows.

```
<tr>
<td> <p> Your Description </p> </td>
<td>   </td>
<td> <p> Your Description </p> </td>
<td>   </td>
<td> <p> Your Description </p> </td>
</tr>
```

TASK 9 - Add a horizontal line or rule <hr />

In the second from bottom row, merge the row using colspan and add a horizontal line.

```
<tr>
```

<td colspan="5"> **<hr />** </td>
</tr>

TASK 10 – Complete the formatting of your gallery page

Format and centre align the images **and** the image descriptions.

<tr>
<td> <**div align="center"**> <img src="images/ file_name.jpg" alt="?" title="?" > ** **
</td>
<td> </td>
<td> ** ** <img src="images file_name.jpg" alt=" ? " title="?" > ** **
</td>
<td>   </td>
<td> ** ** <img src="images/ file_name.jpg" alt=" ? " title="?" >
** ** </td>
</tr>

<tr>
<td> ** ** Your Description </**div**> </td>
<td> </td>
<td> ** ** Your Description ** ** </td>
<td> </td>
<td> ** ** Your Description ** ** </td>
</tr>

TASK 11 Center the table

You may also want to set the width of the table to **65%** and centre the table using align="center"

<table align="center" border="0" width="65%">

TASK 12 – Add in the email link

Add and email link and copyright information (** © All rights reserved. **) to the bottom row and centre and align. You will have to merge the bottom row together to make one long row.

TASK 13 – Test your pages

Test your pages in a web browser.

Code file: 03_Images_and_Tables/files/gallery.html

Summary

This chapter illustrates a HTML tables, their structure and layout. HTML tables provide a really good way to show certain types of data and are relatively easy to construct.

4
Style and Style Sheets

What's in this chapter:

* In line styles.
* Styles in the head.
* Cascading and inheritance.
* Classes and IDs
* External style sheets.

We have mentioned before that HTML defines the structure of the web page, whereas styles define the formatting of the web page. This is a basic concept of web pages where there is separation between the layout and format.

Up until now we have used some styles because it is difficult to illustrate particular points without mentioning them. Also, if you are using Dreamweaver you will find that it does sometimes generate styles for you so you may have seen them as you work with a web page.

A style is a "rule" that defines particular attributes. The latest definition is **CSS3** which includes a number of new features such as rounded corners, animation, fading and transparency which can improve the appearance of your design.

Styles can be:

* in line which are applied directly to the element itself.
* in style definitions located in the head of the web page.
* in a separate style sheet referenced in the web page.

In line styles

In line styles is a way of quickly applying styles to an element without getting involved in creating named styles. For example:

<p style=“border:thin #000 solid; padding:5px; border-radius:10px;”>This is an example of a box around a paragraph**</p>**

Normally styles are given classes and are named which we will show later.

You can apply styles like this to any html element, such as h1, h2, p, img and so on.

Some examples of inline styles

```
<h1 style=“font-size:36pt;”>36 point heading</h1>
<p style=“font-size:16pt;”>16 point paragraphTest</p>
<p style=“background-color:green;font-size:2em;”>Green 2em size paragraph.</p>
```

Placing styles in the head of the document

The following illustrates how a style is placed into the head of a web page.

```
<!DOCTYPE HTML>
<html>
<head>
<meta charset=“utf-8”>
<title>Style in the head of a document</title>
<style type=“text/css”>

h1 {
    color:#2E7DD0;
    font-size:36px;
}

p
{
    padding-left: 20px;
    padding-right: 20px;
    text-align: justify;
    font-family: Arial;
    font-size: 16px;
}

</style>

</head>
<body>


<h1>A main heading</h1>

<p>This is an example of a paragraph</p>

<p>This is another example of a paragraph</p>

<p>This is yet another example of a paragraph</p>
</body>
</html>
```

Code file: 04_Styles_and_Stylesheets/files/01_head01.html

With CSS, you can set many properties once, either in the **head** section of each HTML document or in a remote file, and have it control every instance of that tag on your page.

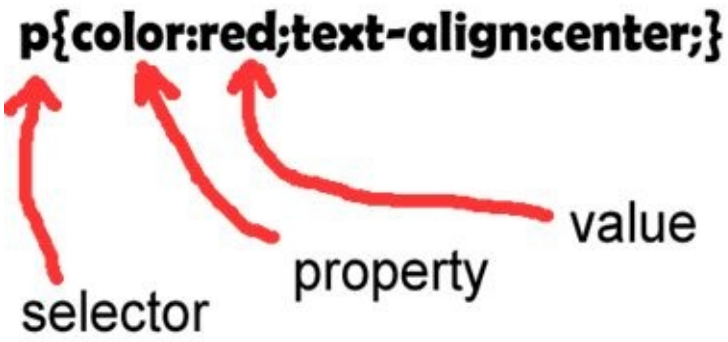
Note that styles are placed between the tags <style type="text/css"> and </style>

The example above applies styles to an h1 tag and a p tag and when you display this in your browser you can see the result.

Style syntax

We can define style syntax in the following way:

p { color:red; text-align:center; }



p is the **Selector** in this example. So we are creating a 'rule' to target specific HTML elements on the web page which in this example are p the elements.

In the brackets are the declarations which consist of a property and a value.

So in this example:

Property is color
Value is red

Property is text-align
Value is center

They are always structured in this form:

property:value;property:value; and so on.

so the general form of a style is

selector { property:value;property:value; }

Note how the styles are defined in the head of the document using:

<style type="text/css">

……..

</style>

Creating styles with Dreamweaver

Creating styles can be difficult if you have to enter them in by hand as it can be easy to mistype a style.

Dreamweaver does have a style designer which can help you in this.

[1] On the right hand side of Dreamweaver, at the top of the tab group is a set of tabs. Click on the CSS Designer tab.

[2] Add a CSS Source by clicking on the + symbol and select “Define in Page”.

[3] In the Selectors section click on the + symbol and enter the selector required.

[4] For example, type in h1 and then in the properties section, you can choose different styles. (Make sure that “show set” has been ticked).

CSS - Cascading Style Sheets

Cascading Style Sheet (CSS) are statements (also known as **rules**) that specify how the content of an element will appear.

Cascading in the name “Cascading Style Sheets” refer to the way in which rule takes precedence over another.

1. Later properties override earlier properties
2. More specific selectors override less specific selectors
3. Specified properties override inherited properties

CSS can be thought of as a **priority** scheme which determines which style will apply when there is more than one rule for a particular element.

Web pages can have multiple style sheets with styles applying to the same element and it is this priority system that works out which specific stylesheet rule applies to which piece of HTML.

The rule cascades down from the more general rule to the specific rule. The most specific rule is chosen. So you could have an enclosing div with font type, and paragraphs within the div with different font types and it will be the more specific paragraph rule that would be chosen.

The following illustrates this. <p> will be at 200% not 100% because it is more specific while the body tag is more general.

```
<!DOCTYPE html>
<head>
<meta content=“text/html; charset=ISO-8859-1” http-equiv=“content-type”>
<title>An example webpage</title>

<style type=“text/css”>
body {
    font-size:100%;
}
p {
    font-size:200%;
}

</style>
</head>
<body>
```

```
<p>This is an example of a paragraph text</p>
<br>

</body>
</html>
```

Code file: 04_Styles_and_Stylesheets/files/cascading_styles_example1.html

The above is illustrated in the file **cascding_styles_example1.html.**

A further example is shown in **cascading_styles_example2.html** file.

So the advantage with this method is that you could have a main style sheet and then have a separate style sheet with styles that overrides some of those styles. You can create a set of styles that define the generic characteristics of your page and then add in more specific styles so simplifying your style sheets.

Inheritance in Styles

Inheritance is where some style properties are inherited by their child elements.

So **font-family** in the body style means that all child elements inherit this property. This means that you don't have to include the font family in all other styles making the style sheet much simpler.

However, not all elements are inherited such as border properties. If border properties were inherited then every element including the p elements would have a border around them making the page very messy.

In the following style example all text takes the font family as defined in the body style which shows that font family styles are inherited. However, the border style is only applied to the wrapper and not to the h1 and p elements.

```
<!DOCTYPE HTML>
<html>
<head>
<meta charset="utf-8">
<title>Inheritance in styles</title>
<style type="text/css">

body{
font-family:Verdana;
color:#666666;
}
.wrapper{
border: 1px solid #ff0000;
background-color:#eeeeee;
}



</style>

</head>
<body>



<div class="wrapper">
<h1>Main title</h1>
<p>A paragraph</p>
<p>Another paragraph</p>
</div>
```

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

Code file: 04_Styles_and_Stylesheets/files/inheritance.html

NOTE: You can force a property to inherit from its parent by placing the word **inherit** after the property.

TASK 1 – Placing styles in the head of the web page

As mentioned previously, styles are placed in the head of the document within:

<style text=”text/css”>

</style>

We will do a couple of examples showing some basic principles.

Create a new web page called **head01.html** as follows. It should have a paragraph with   tags and use the p style in the head of the document. It should also have <h1> tag with style.

```
<!DOCTYPE HTML>
<html>
<head>
<meta http-equiv=“Content-Type” content=“text/html; charset=utf-8”>
<title>Style in the head of a document</title>
<style type=“text/css”>

h1 {
    color:#2E7DD0;
    font-size:36px;
}

p
```

```
{
    padding-left: 20px;
    padding-right: 20px;
    text-align: justify;
    font-family: Arial;
    font-size: 16px;
}

</style>

</head>
<body>


<h1>A main heading</h1>

<p>This is an example of a paragraph</p>

<p>This is another example of a paragraph</p>

<p>This is yet another example of a paragraph</p>
</body>
</html>
```

Code file: 04_Styles_and_Stylesheets/files/01_head01.html

Note that all paragraphs now have the same style.

Modify this further to add in a body style:

```
<!DOCTYPE HTML>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>Style in the head of a document</title>
<style type="text/css">

body {
    font-family: Arial;
    font-size: 16px;
```

```
}

h1 {
    color:#2E7DD0;
    font-size:36px;
}

p
{
    padding-left: 20px;
    padding-right: 20px;
    text-align: justify;
}

</style>

</head>
<body>


<h1>A main heading</h1>

<p>This is an example of a paragraph</p>

<p>This is another example of a paragraph</p>

<p>This is yet another example of a paragraph</p>
</body>
</html>
```

Code file: 04_Styles_and_Stylesheets/files/01_head02.html

Note that the font family and the font size have been placed into the body style. The body style then becomes where you place the default styles for the page.

TASK 2 - Creating your own styles

Naming your own **class** of code is really quite easy. Suppose you want to have a special style to highlight a paragraph. To do this we create a class which is defined using the dot notation before the name of your style.

In the following example our class is defined using .highlight

Create a new web page called **class_01.html** as follows:

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Example of a class style</title>
</head>

<style type="text/css">

body {
    font-family:Arial, Verdana;
    font-size:16px;
}

.highlight {
    color:#D02210;
    font-weight:bold;
    border:thin solid #000000;
    padding:10px;
}

</style>


<body>

<p>Some paragraph text</p>

<p class="highlight">Another paragraph text</p>

<p>Yet another paragraph text</p>

</body>
</html>
```

Code file: 04_Styles_and_Stylesheets/files/class_01.html

TASK 3 - Creating a style for p only

The above example shows how to create a **class** that can be applied to any element.

However, we may want a style that can only be used for a particular selector such as p.

Suppose you want to have a special style with a smaller font and less padding. This is shown in the example **class_02.html**

```
<!DOCTYPE HTML>
<html>
<head>
<meta http-equiv=“Content-Type” content=“text/html; charset=utf-8”>
<title>Example of a class style for p</title>
<style type=“text/css”>
<!—

p {
    padding-left: 20px;
    padding-right: 20px;
    text-align: left;
    font-family: Arial;
    font-size: 16px;
    }

p.smallerpara {
    padding-left: 3px;
    padding-right: 3px;
    text-align: left;
    font-size: 12px;
    }
—>
</style>

</head>
<body>

<h1>This is a heading</h1>

<p>This is an example of a paragraph</p>

<p class=“smallerpara”>This is an example of a paragraph</p>
```

```
<p>This is an example of a paragraph</p>

</body>
```

Code file: 04_Styles_and_Stylesheets/files/class_02.html

Here we have defined a style of p.smallerpara

Note that styles that are defined in the p element are inherited to those in the p.smallerpara element. So p.smallerpara has the same font family even though p.smallerpara does not define it.

Also note that the smallerpara class can only be applied to p elements.

CSS id and Class

So far we have set styles for HTML elements.

We can also create our own **Selectors** which can be **id** or **class**.

We have seen how to define a class selector and they can be used as many times as you like on the page.

The id selector is used once for a unique element and is defined using the # character. An id name is used only once on a web page whereas a class selector is used to set the style for many items on a web page. Id selectors provide for **inheritance** while class selectors do not have inheritance.

The following page gives an explanation of the differences:

```
#wrapper{
padding-left:20px;padding-right:20px;
}
```

the class selector is used for many elements and is defined using the . character (a full stop)

```
.bluepara{
color:blue;
}
```

TASK 4 - An example of a selector id

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Untitled Document</title>
<style type="text/css">

#wrapper{
padding-left:20px;padding-right:20px;
}
```

```
#wrapper p{
color:green;
}

</style>

</head>
<body>
<div id=“wrapper”>This is an example of id <p>A paragraph</p></div>
</body>
</html>
```

Code file: 04_Styles_and_Stylesheets/files/selector_01.html

Note the way that we can control the elements with the id using the syntax of #wrapper p

TASK 5 - An example of a selector class

```
<!doctype html>
<html>
<head>
<meta charset=“utf-8”>
<title>Untitled Document</title>
<style type=“text/css”>

.bluepara{
color:blue;
}

</style>

</head>
<body>
<p>This is an example of a paragraph</p>
<p class=“bluepara”>This is an example of a blue paragraph</p>
</body>
</html>
```

Code file: 04_Styles_and_Stylesheets/files/selector_02.html

Some notes about Font Sizes

Should I use pixels, points, ems or something else for font sizes?

Font sizes can be Points (pt), Picas (pc), Pixels(px), Ems(em) or Percent(%)

Points (pt) and Picas (pc): Points and Picas are traditionally used in print media (anything that is to be printed on paper, etc.). One point is equal to 1/72 of an inch. Points and Picas are much like pixels, in that they are fixed-size units and cannot scale in size. Pica is another print size and is 1/6 of an inch. Avoid using these sizes because monitors are not always able to accurately set these correctly. Examples are:

```
p { font-size:10pt; }
p { font-size:10pc; }
```

Pixels (px): Pixels are fixed-size units that are used in screen media (i.e. to be read on the computer screen). One pixel is equal to one dot on the computer screen (the smallest division of your screen's resolution). Many web designers use pixel units in web documents in order to produce a pixel-perfect representation of their site as it is rendered in the browser. One problem with the pixel unit is that it does not scale upward for visually-impaired readers or downward to fit mobile devices. An example is:

```
p { font-size:10px; }
```

"Ems" (em): The "em" is a scalable unit that is used in web document media. An em is equal to the current font-size, for instance, if the font-size of the document is 12pt, 1em is equal to 12pt. In fact 1em is the hight of the letter "M". Ems are scalable in nature, so 2em would equal 24pt, .5em would equal 6pt, etc. Ems are becoming increasingly popular in web documents due to scalability and their mobile-device-friendly nature. The default font setting with be 1em for a font size. **"Rems" (rem)** is a variation of ems. The size is relative to the root HTML definition. Its effect is to reset the size and can be useful in some situations when you don't want children elements to inherit the size. Some examples are:

```
p { font-size:1em; }
p { font-size:0.9em; }
```

Percent (%): The percent unit is much like the "em" unit, save for a few fundamental differences. First and foremost, the current font-size is equal to 100% (e.g. 12pt = 100%). While using the percent unit, your text remains fully scalable for mobile devices and for accessibility. An example is:

```
p { font-size: 90%; }
```

So don’t use Points and Picas or Pixels, but use either percentages or ems both of which are scalable. Scalable fonts are becoming much more important with responsive design where you are creating layouts for different size browsers.

DIVs and SPAN

DIVs and SPANs are the way in which CSS uses to divide up pages and apply different styles.

A SPAN defines a section which you can then apply styles to - I don’t tend to use spans that much but there may be situations where you want to do that.

A DIV defines a block or a container which you can then apply styles to.

Creating a separate style sheet

In these examples we can also go back to our “Giant Snail” (**article.html**) exercise to try out the external style sheet.

We can place the styles into a separate style sheet so that we can use the same styles in different web pages.

The code to do this goes in the <head> …. </head>

<link href=“styles.css” rel=“stylesheet” media=“screen” type=“text/css” />

The reference styles.css can be either relative to this location or a full URL.

The web page **example.html** looks like this with a reference to a style sheet.

```
<!DOCTYPE HTML>
<html>
<head>
<meta http-equiv=“Content-Type” content=“text/html; charset=utf-8”>
<title>Untitled Document</title>
<link href=“styles.css” rel=“stylesheet” media=“screen” type=“text/css” />

</head>
<body>



<p>This is an example of a paragraph</p>
</body>
</html>
```

The style sheet **styles.css** looks like this

```
p {padding-left: 20px;
padding-right: 20px;
text-align: justify;
font-family: Arial;
font-size: 16px;}
```

Code file: 04_Styles_and_Stylesheets/files/example.html

Note that you can have multiple style sheets in a web page. Some designers have separate

style sheets, so one sheet might contain font, table styles and another contains page layout styles.

Exercise

Create the web page **example.html** as above and the style sheet **styles.css** as above. Test that you external style sheet is working correctly.

Modify the style sheet to include a body style. Define a style for font family in the body style and test your web page.

The @media rule

@media is used to define different style rules for different media, such as screen, screens with different width, printer, speech and others.

In particular, it can be used to create styles specifically for a printer version of a page. You might create styles which always have black for the font color and may also make certain images not display.

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

.bluepara{
  color:blue;
}


/* print styles placed at the end of the main styles  */
@media print {

  .bluepara{
    color:black;
  }

}

</style>
```

In this example, when you print the page, the style will have a color of black.

The @media rule is extensively used in responsive design layouts.

Summary

This chapter has looked at some of the features of styles that make them so useful. It has looked at using external style sheets which means that many different web pages can use the same styles. It has also looked at the concept of cascading. A good understanding of how styles cascade will mean that you will have fewer styles and they will be easier to maintain and modify.

5
Further Styles

What’s in this chapter:

* Summary of style definitions.
* Adding styles to the head of the page.
* Hyperlinks in web pages.
* The CSS box model.
* DIVS and SPANS.
* Further tables.

Style Definition Summary

We first look at the three main categorisation of styles which are: tags, classes and ids

[1] Built in Tags (selectors)

Styles for tags (selectors) define the styles for built in standard tags like <h1>, , <img and so on.

<h1>This is a heading</h1>

Style is defined as:

```
h1 {
background-color:#cccccc;
}
```

When you create a style for these, **ALL** h1 tags will then have those styles.

This refines an html element.

[2] Classes

Classes are user defined styles that we can apply to any tag (selector) to change the style for one or more tag. You can choose which ones to apply the styles to. The name of the class can be anything that you want and in this example I have used “quote” as the class name.

* This is a paragraph *

Style is defined as

.quote { font-size:16px }

Note the full stop in front of the style definition name

This can apply to any HTML element.

[3] ids

ids are tags (selectors) which we use **ONCE** on the page and are often used with div (divisions) which define the layout of a page.

* This is a div *

Style is define as:

#wrapper { background-color:blue; }

Note the # This applies to only one HTML element.

Style Exercises

The following Tasks illustrate many style features that you will want to control in your web page.

For these tasks, you may want to create a new web page for each task, or you may want to use just the one web page, adding on each style as you go – it doesn't matter which method you choose.

You should start by creating a web page and in the **head** of the web page, enter in the styles.

TASK 1 - Set the page to a defined font and size

Create a new web page and create:

[1] A body style.
[2] A font size.
[3] A font family.
[4] A background color.

Modify the values to see the effects as they are changed.

Note that placing the styles into the body tag defines the default styles for the page.

Code file: 05_Further_Styles/files/styles01.html

STYLES

```
body{
    font-size:16px;
    background-color:#CCCCCC;
    font-family:Arial, Helvetica, sans-serif;
}
```

BODY TEXT

```
<p>This is some text</p>
```

TASK 2 - Modify an h1 tag

Create a new web page and create:

A new class style

[1] Define a font.
[2] Define a border.
[3] Define a background color.

Modify the values to see the effects as they are changed.

Code file: 05_Further_Styles/files/styles02.html

STYLES

```
.box {
background-color:#FFFF99;
font-family:Geneva, Arial, Helvetica, sans-serif;
border-style:solid;
border-width:1px;
}
```

BODY TEXT

```
<h1 class=“box”>This is a heading</h1>
```

Try changing the border style and other attributes of the box class.

TASK 3 - Modify a paragraph tag

Create a new web page and create:

A new class style

[1] A font size.
[2] A margin left value.
[3] A text indent value.

Apply the class style to a paragraph tag.

Code file: 05_Further_Styles/files/styles03.html

STYLES

```
.myPara {
```

```
    font-size:120%;
    margin-left:20px;
    text-indent: 20px;
}
```

BODY TEXT

```
<p class=“myPara”>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Pellentesque ullamcorper, metus id tempor dictum, magna mi ultrices ligula,
sagittis porttitor metus sapien sit amet sapien. Cras eu enim in ante sagittis ornare quis non augue. Mauris pellentesque quam vitae
pulvinar ullamcorper. Donec tempus eleifend tortor, eu accumsan odio lobortis et. Donec sed hendrerit ipsum. Sed sed ex ut lectus consectetur
condimentum eget aliquet neque. Mauris nisl tortor, pretium et fermentum eu, bibendum at odio. Praesent ligula arcu,
</p>
```

TASK 4 - Creating a quote block of text

Create a new web page and create:

A new class style

[1] A font size and font family.
[2] A margin and padding for the text.
[2] A thin border box.
[3] A background color.

Apply the class style to a paragraph tag.

Code file: 05_Further_Styles/files/styles04.html

STYLES

```
.quote {
    font-size:90%;
    font-style:italic;
    font-family:Arial, Helvetica, sans-serif;
    margin: 30px 0px 30px 0px;
    padding: 20px 20px 20px 20px;
    border:thin #000000 solid;
    background-color:#FFFFCC;
}
```

BODY TEXT

```
<p class=“quote”>Ut eleifend, lorem sit amet viverra sollicitudin, ex massa venenatis nisl, quis faucibus velit ipsum at odio. Cras luctus neque orci,

ut molestie libero tincidunt eu. Aliquam vel lorem non turpis facilisis laoreet. Vivamus ante mauris, condimentum non porttitor non,

lobortis nec est. Vivamus ac risus id odio venenatis tempus. Ut malesuada tortor nec facilisis condimentum.</p>
```

Try changing the attributes of the quote class, margin, padding etc. to see the changes.

NOTE: We can easily look up the attributes of margin by going to Google and doing a search with something like the following: ‘html margin’. This will return with a set of URLs which you can go to and describe this particular attribute.

Links

Links or Hyperlinks is the way in which you move from one web page to another. We can apply styles to give different type of effects as someone moves their mouse over the link.

a:link	Defines the style for normal unvisited links.
a:visited	Defines the style for visited links.
a:active	Defines the style for active links. This is the style the moment the link is clicked.
a:hover	Defines the style for hovered links.

link, visited, active and hover are called **Pseudo Classes** which add an effect to some selectors. In this case they modify the link display.

Note that the order should be a:link, a:visited, a:hover for it to work

TASK 5 - Remove the underline from a hyperlink

Create a new web page to demonstrate how the underline can be removed from a hyperlink.

[1] Add styles for the a:link and the a:hover Pseudo Classes.

Code file: 05_Further_Styles/files/styles06.html

STYLES

```
a:link {text-decoration:none;}
a:hover {text-decoration:none;}
```

BODY TEXT

```
<a href=“http://www.withinweb.com”>My site</a>
```

TASK 6 - Make hyperlinks block hover

Create a new web page and add in the styles:

[1] a:link, a:visited, a:hover and a:active Pseudo Classes.

Code file: 05_Further_Styles/files/styles07.html

STYLES

```
a:link { color: #036; }
a:visited { color: #066; }
a:hover, a:active { color: #fff; background-color: #036; }
```

Note: :active MUST come after :hover (if present) in the CSS definition in order to be effective.

BODY TEXT

```
<a href="http://www.withinweb.com">My site</a>
```

As an additional exercise, add in the **text-decoration:none** to the a:link and a:hover examples above.

TASK 7 - Creating two types of styles for links using a div class

Create a new web page and add in the following styles to illustrate how to have two different hyperlink styles.

Code file: 05_Further_Styles/files/styles08.html

STYLES

```
a:link { color: #036; }
a:visited { color: #066; }
a:hover, a:active { color: #fff; background-color: #036; }
.section a:link { color: #036; }
.section a:visited { color: #066; }
.section a:hover, a:active { color: #fff; background-color:#CCCCCC }
```

BODY TEXT

```
<p><a href=“http://www.withinweb.com”>My site</a></p>

<div class=“section”>
<p><a href=“http://www.withinweb.com”>My other site</a></p>
</div>
```

TASK 8 - Creating two types of styles using an href class

Create a web page with the following styles to illustrate using a class for the href link.

Code file: 05_Further_Styles/files/styles09.html

STYLES

```
a:link {color:#FF0000;}      /* unvisited link */
a:visited {color:#00FF00;}  /* visited link */
a:hover {color:#FF00FF;}  /* mouse over link */
a:active {color:#0000FF;}  /* selected link */

a.topbar:link{ color:#0099FF; }
a.topbar:visited{ color:#FF9933; }
a.topbar:hover{ color:#FFFFCC; }
a.topbar:active{ color:#99CC66; }
```

BODY TEXT

```
<br/>
<a href=“http://www.wiltshire.ac.uk”>Test link</a>

<br/>
<a href=“http://www.wiltshire.ac.uk” class=“topbar”>Test link</a>
```

Images and Styles

TASK 9 - Remove a border or add a border to an image

Create a web page with a new class and with:

[1] a border style.

Code file: 05_Further_Styles/files/styles10.html

STYLES

```
.imgborder {
border:10px;
border-color:#CCCCCC;
border-style:solid;
}
```

BODY TEXT

```
<img src=“snail.jpg” class=“imgborder” />
```

TASK 10 - Add a shadow to an image

Create a web page with a class and style to add a shadow to an image

Code file: 05_Further_Styles/files/styles11.html

STYLES

```
.imgborder {
box-shadow: 10px 10px 5px #888;
}
```

BODY TEXT

```
<img src=“snail.jpg” class=“imgborder” />
```

CSS Box Model

One of the most important things to understand about CSS is the Box Model. Every element that we use in the page is considered to be a rectangular box.

The properties for this box are as follow:

content, **padding**, **border** and **margin**.

You can understand how this element interacts from the following image:

As you can see the padding is applied around the content area. So we can use the padding to create a gutter around the content. Then we have the border, shown in blue in the diagram, which if applied creates a line around the padding element. We can define the thickness in pixels of the line, the style of the line such as solid, dashed, dotted.

After the border there is the margin. This is transparent and we can use it to create and control spacing between different boxes. Padding, borders and margins are optional and they are set to zero as default setting. If we want to them set back to zero we can use the universal selector and the following code:

***{ margin:0; padding:0; }**

If we assign a width and height to a box, it refers to the **content** of the box only. The padding, borders, and margins are then added to this value and summing all together gives the total box width. So if I want a total box to be 100px wide, one way is to use the following values:

***{ margin: 10px; padding: 10px; width: 60px; }**

This consists of a total margin of 20px and a total padding of 20px plus the width of the content.

So in summary, the padding adds in space between the content and the border, and the margin adds in space between the border and other content.

There are two web page examples of this:

Code File: 05_Further_Styles/files/CSS_Box_Model_Margin.html

This shows a box with a margin

Code file: 05_Further_Styles/files/CSS_Box_Model_Padding.html

This shows a box with padding

Block and Inline elements

Block Elements

Some elements will always appear to start in a new line in the browser. These are called **block level** elements and example of these are <h1>, <h2>, <p> ,

Inline Elements

Some elements always appear to continue on the same line as neighbouring elements. These are known as **inline elements**. Examples of these would be <i>,   and .

It is quite easy to demonstrate the difference between **block** and **inline** elements using a simple web page and adding borders to the styles.

If we look at the web page:

Code file: 05_Further_Styles/files/block_and_inline.html

we will see that it is made up of a mixture of block level and inline elements. We can show this more clearly by applying border styles.

Code file: 05_Further_Styles/files/block_and_inline.html

All block level elements have been identified by a border of orange, and all in line elements have been identified with a border of green.

HTML Div and Span

In the following tasks we will be using the example text in the file **div_and_span_example_text.txt**

Difference between div and span

div (division or container) is a block element, **span** is inline.

This means that divs should be used to wrap sections of a document, while spans should be used to wrap small portions of text, images, etc. divs are used to "group" together elements.

Example of a span:

[1] Create and save a new web page called **div01.html**

[2] In the body of the web page, add the text from the file **div_and_span_example_text.txt**

[3] For each paragraph, put start and end tags around the paragraphs.

[4] Now we can demonstrate a span element:

```
<span style="background-color:#00ff00;">
    Some text here
</span>
```

Place the example span around some text in one of the paragraphs and view in a browser.

View the page in the browser.

Code file: 05_Further_Styles/files/div01.html

TASK 11 - Float left and float right blocks

An example of float left and float right of a div:

[1] Create and save a new web page called **div02.html**

[2] In the body of the web page, add the text from the file **div_and_span_example_text.txt**

[3] We need to float left the first block and float right the **second** block of text and then view the page in the browser to see the result.

```
<div style="float:left;width:300px;height:auto;">
    First block of text here
</div>
```

```
<div style="float:right;width:300px;height:auto;">
    Second block of text here
</div>
```

Code file: 05_Further_Styles/files/div02.html

You may want to try modifying the width of the blocks to see what happens.

The normal flow of a div (or in fact any element) is one under the other. Float means that you take the element from the normal flow and place on either the left or the right side of the container.

TASK 12 - Centre a block of text with a div

You must define width and margins as auto as below (you can use this method to centre anything in a div):

[1] Create and save a new web page called **div03.html**

[2] In the body of the web page, add the text from the file **div_and_span_example_text.txt**

[3] We need to apply the following style to the div with defining the width and margins as auto and then view the page in a browser.

```
<div style="width:400px;margin: 0 auto;>
    Text to be centered here
</div>
```

Code file: 05_Further_Styles/files/div03.html

TASK 13 - Centre a block of text with two divs

This shows how to centre two divs. This uses one div which is centred and the other two divs use float left and float left. You might find this useful when you want to put two sets of text together.

[1] Create and save a new web page called **div04.html**

[2] In the body of the web page, add the text from the file **div_and_span_example_text.txt**

[3] We need to use an outer div to centre all the text and one text block is floated left, the

other floated right.

```
<div style="width:60%;margin-left:auto;margin-right:auto;height:auto;">
<div style="float:left;width:40%;height:auto;">
    First block of text here
</div>
<div style="float:right;width:40%;height:auto;">
    Second block of text here
</div>
</div>
```

Code file: 05_Further_Styles/files/div04.html

TASK 14 - Floating images to the right

It is quite common to want to have text wrapping around an image. We can do this by floating the image to the right but with all the text contained within an outer div as follows.

[1] Create and save a new web page called **div05.html**

[2] In the body of the web page, add the text from the file **div_and_span_example_text.txt**. We also have to add image using the snail.jpg file with a width of 300px in this example.

[3] We create a div around the image to float the image right. But we also have to create an outer div around all the items like this:

```
<div style="float:left;width:550px;">
<div style="width:300px;float:right;"><img src="snail.jpg" width="300px" /></div>
    Enter the text here
</div>
```

Code file: 05_Further_Styles/files/div05.html

TASK 15 - Floating images to the left

This is essentially the same as the previous example but with the image floating to the left.

[1] Create and save a new web page called **div06.html**

[2] In the body of the web page, add the text from the file

div_and_span_example_text.txt. We also have to add image using the snail.jpg file with a width of 300px in this example.

[3] We create a div around the image to float the image left. But we also have to create an outer div around all the items like this:

```
<div style="float:left;width:550px;">
<div style="width:300px;float:left;"><img src="snail.jpg" width="300px" /></div>
Enter the text here
</div>
```

Code file: 05_Further_Styles/files/div06.html

Further Tables

In a previous chapter we looked at tables. Tables are used for tabular data of course. Tables can also be used to layout areas of a web page but you should try to restrict the use of tables for tabular data because it is often cleaner and easier to use DIVs. If we go back a number of years, tables were used to define web page layout, nowadays using tables for page structure is frowned upon.

HTML 5 and CSS 3 enable the creation of really nice looking tables which we will look at here.

TASK 16 - Creating a new table

[1] Create a new HTML 5 web page called **table_01.html** in a folder.

[2] In the body of the web page we need to create a basic table as follows:

```
<table>
<tr>
    <th>Name</th>
    <th>Job</th>
    <th>Dept</th>
    <th>Salary</th>
</tr>
<tr>
        <td>Fred Blogs</td>
      <td>Programmer</td>
      <td>IT</td>
      <td>£35000</td>
</tr>
<tr>
    <td>Pete Smith</td>
      <td>Systems Engineer</td>
      <td>IT</td>
      <td>£28000</td>
</tr>
<tr>
    <td>Joe Soap</td>
      <td>HR Admin</td>
      <td>HR</td>
      <td>£16000</td>
</tr>
</table>
```

If you are using Dreamweaver, you should be able to create this fairly easily in design view.

Note that the first row defines header columns identified as **th**. The header column automatically is defined as bold.

The **th** table header element is something that has always been available in HTML and allows you to define the headers of a table. One of the reasons for using **th** is to do with audio screen readers which will understand that a **th** is actually a table header and not table data.

Code file: 05_Further_Styles/files/table_01.html

TASK 17 - Adding in some styles

```
body {
    font-family:Arial, Helvetica, sans-serif;
}
table {
    width:550px;
    border-collapse:collapse;
}
th, td
{
    padding: 5px 10px 5px 10px;
}
tr:nth-of-type(odd) {
    background-color:#efefef;
}

tr:hover {
    background: #cccccc;
}
```

Code file: 05_Further_Styles/files/table_02.html

TASK 18 - Formatting the display further

Add in the following style to format the header:

```
th
{
    background: #cccccc;
    text-align: left;
}
```

Add in the following class to format the column used for displaying the salary:

```
.money {
    text-align:right;
}
```

Then add class=”money” to the <th> and <td> for the salary column.

Code file: 05_Further_Styles/files/table_03.html

Summary

This chapter looks at how to use styles in more detail. It looks at placing styles into external style sheets and gives various examples of the some of the techniques that you may want to use. It also looked at the box model and the difference between block level and in line elements.

6
Site Examples

What’s in this chapter:

* Create a single page web site.
* Minify the CSS.

Create a Business Card Web Site

The concept of this is a single page ‘business card’ web site where all the details of the company / shop are on one web page. This could also be used as the basis for a CV type web site.

As you work through the exercise, test each stage and you will see how the page is built up as the styles are added.

TASK 1 - Create a new web page

Create a new folder called **business**.

Create a new HTML 5 web page called **index.html**

Change the title of the page to **Archibald’s Barbers Shop**

Code file: 06_Site_Examples\files\BusinessCardLayout\index_task01.html

TASK 2 - Create some DIV containers

Create a div with id name of **container**. This is the overall container

Inside that div create other divs called header, logo image, description, openinghrs, location, prices, map-canvas and footer as follows:

```
<body>
<div id = “container”>
    <div id=“header”>
        </div>

    <div id=“logo”>
        </div>

    <div id=“image”>
    </div>

    <div id =“description”>
    </div>

    <div id=“openinghrs”>
    </div>

    <div id=“location”>
    </div>
```

```
    <div id="prices">
    </div>
        <div id="contactus">
    </div>

    <div id="map-canvas">
    </div>

    <div id="footer">
    </div>

</div>
</body>
```

Code file: 06_Site_Examples\files\BusinessCardLayout\index_task02.html

TASK 3 - Add in main image

Insert **image03.jpg** inside the image div and set width="100%" in the <img tag

```
<img src="image03.jpg" width="100%">
```

This will constrain the image to the size of the web page.

Code file: 06_Site_Examples\files\BusinessCardLayout\index_task03.html

TASK 4 - Add in header text

In the header type in:

Home | Opening Hours | Location | Prices | Contact Us

Create a new style called #header with a background-color of black.

```
<style type="text/css">
#header {
```

```
Background-color:#000000;
}
</style>
```

Code file: 06_Site_Examples\files\BusinessCardLayout\index_task04.html

TASK 5 - Add in description text

In the description div copy in:

Welcome to Archibald’s Barber in Ilfracombe Devon

You don’t need an appointment and we are open late on Thursday evening to accommodate busy schedules.

Try our professional hairdresser today.

Code file: 06_Site_Examples\files\BusinessCardLayout\index_task05.html

TASK 6 - Add in opening hours text

In the opening hours div copy in:

Come Visit Us

Opening Hours

Monday Closed All Day
Tuesday 8.00 am to 5.30 pm
Wednesday 8.00 am to 5.30 pm
Thursday 11.00 am to 6.30 pm
Friday 11.00 am to 6.30 pm
Saturday Closed all day
Sunday Closed all day

Code file: 06_Site_Examples\files\BusinessCardLayout\index_task06.html

TASK 7 - Add in location text

In the location div copy in:

Location

North Field Road
Ilfracombe

Devon
EX34 8AL

Code file: 06_Site_Examples\files\BusinessCardLayout\index_task07.html

TASK 8 - Add in prices text

In the prices div copy in:

Prices

Adults £7.00
Children £6.00
Senior Citizens £4.00

Code file: 06_Site_Examples\files\BusinessCardLayout\index_task08.html

TASK 9 - Add in contact text

In the contactus div copy in:

Contact Us

07824728621

Code file: 06_Site_Examples\files\BusinessCardLayout\index_task09.html

TASK 10 - Add in footer text

In the footer div, copy in:

&copy 2016

Code file: 06_Site_Examples\files\BusinessCardLayout\index_task10.html

TASK 11 - Format text

For each text area add in **<h1>..</h1>** tag for the title and paragraph ** .. ** tags around the main text and **
** line break at the end of each line to force a new line.

Code file: 06_Site_Examples\files\BusinessCardLayout\index_task11.html

TASK 12 - Add in body styles

```
#body {
        font-family: Arial, Helvetica, sans-serif;
    color: #030;
    margin:0px;
}
```

Code file: 06_Site_Examples\files\BusinessCardLayout\index_task12.html

TASK 13 - Add in header styles

```
#header {
    background-color: #000000;
    color: #FFF;
    margin:0px;
    padding: 20px;
    font-size: 120%;
}
```

Code file: 06_Site_Examples\files\BusinessCardLayout\index_task13.html

TASK 14 - Add in styles for each block

For the description div:

```
#description {
   padding-top: 20px;
    padding-right: 250px;
    padding-bottom: 10px;
    padding-left: 250px;
    text-align: center;
}
#description h1 {
   font-size:120%;
    color:#F30;
}
```

and then for each of the other divs **openinghrs**, **location** and **prices** use the same styles.

Note that we could do multiple styles like this:

```
#prices h1, #location h1 {
}
```

Code file: 06_Site_Examples\files\BusinessCardLayout\index_task14.html

TASK 15 - Add in footer styles

```
#footer {
   background-color: #000;
    color: #FFF;
    margin:0px;
    padding: 20px;
    }
```

Code file: 06_Site_Examples\files\BusinessCardLayout\index_task15.html

TASK 16 - Add in Canvas Styles

```
#map-canvas {
   width: 100%;
   height: 400px;
   }
```

Code file: 06_Site_Examples\files\BusinessCardLayout\index_task16.html

TASK 17 - Add in Google map

Google provides JavaScript code that can be just pasted into a web page and which will then display a Google map.

The JavaScript is shown below already configured and should be copied and pasted between the **<head>….</head>** tags.

```
<script src=“https://maps.googleapis.com/maps/api/js”></script>
   <script>
   function initialize() {
      var mapCanvas = document.getElementById(‘map-canvas’);
```

```
    var mapOptions = {
      center: new google.maps.LatLng(51.208656, -4.126427),
      zoom: 16,
      mapTypeId: google.maps.MapTypeId.ROADMAP
  }
  var map = new google.maps.Map(mapCanvas, mapOptions)
      var myLatlng = new google.maps.LatLng(51.208656, -4.126427);
      var marker = new google.maps.Marker({
      position: myLatlng,
      map: map,
      title: 'Archibalds Barbers Shop'
});
  }
  google.maps.event.addDomListener(window, 'load', initialize);
  </script>
```

The entries that are configured for this web page are:

- the name of the div where the map will appear which in this example is 'map-canvas'
- the latitude and longitude where you want the map to be focused on
- the latitude and longitude of the marker
- The title of the marker

For further information, refer to Google documentation which should explain all the options and other options available:

https://developers.google.com/maps/tutorials/fundamentals/adding-a-google-map

To add a marker to the point of interest:

https://developers.google.com/maps/documentation/javascript/examples/marker-simple

Code file: 06_Site_Examples\files\BusinessCardLayout\index_task17.html

TASK 18 - Add in logo and logo styles

In the logo div, add in some text which represents the title of the site:

Archibald's Barber Shop

```
#logo {
    background-color: #000;
    color: #FFF;
    font-size: 240%;
    padding-top: 30px;
    padding-right: 40px;
    padding-bottom: 40px;
    padding-left: 120px;
    border-top-width: thick;
    border-top-style: solid;
    border-top-color: #CCC;
}
```

This should create a black background with white text, and create a top border to give a separation.

If you wanted to you could replace the text with a graphic image.

Code file: 06_Site_Examples\files\BusinessCardLayout\index_task18.html

TASK 19 - Add in href anchor links

Add in href anchor links and links for styles.

The top header text is intended to be a list of href links. When a link is selected, the page should scroll down rather than go to another page.

To do this we use the href link but using a named anchor text:

Place **** next to the div <div =”location”>

Then in the header text we put **Location**

Notice that the href link has the # character followed by the name of the <a name

Now when we click on the ‘Location’ link at the top of the page, it should move down to that position on the page.

Code file: 06_Site_Examples\files\BusinessCardLayout\index_task19.html

TASK 20 - Add in the href links styles for the header

The following styles will change the href links to display colors correctly.

```
#header a:visited{
    color: #FFF;
    text-decoration: none;
```

```
}
#header a:link{
    color: #FFF;
    text-decoration: none;
}
#header a:hover{
    color:  #F00;
    text-decoration: none;
}
```

Code file: 06_Site_Examples\files\BusinessCardLayout\index_task20.html

TASK 21 - Place styles into a separate style sheet

Create a new folder called **CSS**

Create a new file called **styles.css** and copy all the styles from the web page into **styles.css** and replace the styles in the web page with a <link to the stylesheet.

Files for Business Card Web Site

All the files for this project are located in Chapter 6 under the folder **BusinessCardProject**.

Minify CSS

One of the issues of modern web pages is that they can create large CSS style sheets particularly with responsive layouts that have to work with different display devices which operate over slower network speeds.

Minify is one way to overcome the size of CSS style sheets. Essentially, the idea is to remove unwanted spaces, comments and other text from the style sheet which reduces the file size and hence speeds up the time to download the page.

There are many web sites which help you to compress and minimise files to speed up the speed of a page, and this example shows a simple way of minimising CSS.

[A] - Create a separate style sheet

For the example web page that we have just created for Archibald's Barbers, extract the styles and place them into a separate style sheet called **styles.css.**

This style sheet should be placed in a new folder called **stylesheets**

Now in your web page create a link to the **styles.css** which will look something like:

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

Display the web page in your browser to make sure the styles are working as expected.

-

[B] - Minify the css

Go to the web page:

http://www.feedthebot.com/tools/css/#minify

where you should copy in the styles to minify the css.

Copy the result back into the style sheet and test that the web pages still work.

Summary

This chapter has tried to bring together many of the ideas that have been explained in previous chapters. It brings together Divs to define the page layout and then applying styles and classes to control the look and feel of the page. As you worked through the exercises you should have seen the changes that take place to give the final result.

7
Lists and Menus

What's in this chapter:

* Unordered and ordered lists.
* Menus and list.
* CSS menu lists.

This chapter looks at using lists and menus. We have already looked at lists briefly where we showed that you can create an ordered list or an unordered list. Adding styles to lists and making the list go in a horizontal direction is a way in which menus are created.

Unordered lists

Unordered lists are defined by <ul> with each element using <li> as shown in the following example:

```
<ul>
    <li>One large tomato</li>
    <li>A pinch of salt</li>
    <li>3 kg of potatoes</li>
</ul>
```

The **list-style-type** CSS property can have values to give different appearance to the bulleted listing:

So

list-style-type:square

displays a square bullet list:

```
<ul style=”list-style-type:square”>
    <li>One large tomato</li>
    <li>A pinch of salt</li>
    <li>3 kg of potatoes</li>
</ul>
```

Code file: 07_Lists_and_Menus/files/lists.html

Here is a list of some of the more common **list-style-type**.

Disc	Solid circle.
circle	Hollow circle.
square	Solid square.
decimal	1, 2, 3, 4, etc.
decimal-leading-zero	01, 02, 03 … 10, 11, etc.
lower-roman	i, ii, iii, iv, etc.

upper-roman	I, II, III, IV, etc.
lower-greek	Greek characters.
lower-latin	a, b, c, d, etc.
upper-latin	A, B, C, D, etc.
armenian	Armenian numbering.
georgian	Georgian numbering.
lower-alpha	Equivalent of lower-roman.
upper-alpha	Equivalent of upper-roman.
none	No marker.

Ordered Lists

Ordered lists are defined by <ol> with each element using

```
<ol>
    <li>One large tomato</li>
    <li>A pinch of salt</li>
    <li>3 kg of potatoes</li>
</ol>
```

To add style to the order list we can add the type attribute:

Type	
type=“1”	The list items will be numbered with numbers (default)
type=“A”	The list items will be numbered with uppercase letters
type=“a”	The list items will be numbered with lowercase letters
type=“I”	The list items will be numbered with uppercase roman numbers
type=“i”	The list items will be numbered with lowercase roman numbers

```
<ol type=”A”>
    <li>One large tomato</li>
    <li>A pinch of salt</li>
    <li>3 kg of potatoes</li>
</ol>
```

Code file: 07_Lists_and_Menus/files/lists.html

Using images in lists

Another CSS style that can be used with lists is the **list-style-image** which allows us to place an image of our choice against the list item.

To illustrate this, create a new HTML web page called **listimages.html**

```
<p>HTML 5 enables you to:</p>

<ul>
<li>Easier Audio and Vision support</li>
<li>Simplier and cleaner HTML code</li>
<li>Better forms and tables styles</li>
</ul>

<br/>
```

An image can be added by using the style:

```
<style type=“text/css”>
ul
{
    list-style-image:url(lightbulb.png);
}
</style>
```

Code file: 07_Lists_and_Menus/files/listsimages.html

All CSS List Properties

Property	
list-style	Sets all the properties for a list in one declaration
list-style-image	Specifies an image as the list-item marker
list-style-position	Specifies if the list-item markers should appear inside or outside the content flow
list-style-type	Specifies the type of list-item marker

Create a menu using an Unordered List

This is an example of a menu using CSS and an unordered list.

Unordered lists are usually lists with icons but in a menu we use the style **list-style-type: none;** to disable any icon.

We can use the features of an unordered list with different colors in the various link states to give our effects.

Create a new web page called **index_menu.html** and copy the following unordered list into the body.

```
<div class="menu">
<ul>
<li><a href="index.html">Home</a></li>
<li><a href="about.html">About</a></li>
<li><a href="location.html">Location</a></li>
<li><a href="contact.html">Contact Us</a></li>
<li><a href="shop.html">Shop</a></li>
</ul>
</div>
```

Now view this in the web page.

Now we add the styles to format the menu as follows:

```
<style type="text/css">
.menu ul {
    margin: 0;
    padding: 0;
    width:185px;
    list-style-type: none;
}

.menu ul li a {
    text-decoration: none;
    color: white;
    padding: 10.5px 11px;
    background-color: #005555;
    display:block;
}
```

```
.menu ul li a:visited {
    color: white;
}

.menu ul li a:hover, .menu ul li .current {
    color: white;
    background-color: #5FD367;
}
</style>
```

Code file: 07_Lists_and_Menus/files/index_menu.html

Example of a menu in a web page

To demonstrate this in a web page there is an example layout in 06-Navigation in the **layout** folder which we can use.

In 07_Lists_and_Menus in the **layout** folder:

[1] Download the index.html page and view it in a browser. This page is a basic layout with blank images and no styles applied.

Code file: 07_Lists_and_Menus/files/layout/index.html

[2] Download the images in the images folder and place them all in a folder called **images**.

[3] Download the **styles.css** page.

[4] Now add the link to the styles.css page to the index.html page.

View this page in the browser and you should see that it has been formatted by the styles.

You should see that the navigation is not styled as we have not added in the styles for that yet.

Code file: 07_Lists_and_Menus/files/layout/index2.html

[5] Use the styles from the previous exercise and put them in the **menu.css** stylesheet then add this stylesheet to the index.html page.

Code file: 07_Lists_and_Menus/files/layout/index3.html

[6] View the page in the browser.

[7] Now we can copy the index.html to create files called about.html, contact.html, location.html and shop.html and when we click on the navigation, the pages should work correctly.

Note the use of class=”current” to define which menu item is highlighted.

This is shown in the folder **07_Lists_and_Menus** within the **layout_multi** folder.

CSS Mouseover Buttons

This method gives a simple horizontal menu with submenus. It uses lists and CSS to configure the lists.

The html for the menu is as follows:

```
<ul id=“menu”>
<li><a href=”#”>Home</a></li>
<li><a href=”#”>About Us</a>

    <ul>
    <li><a href=”#”>The Team</a></li>
    <li><a href=”#”>History</a></li>
    <li><a href=”#”>Vision</a></li>
    </ul>

    </li>
    <li><a href=”#”>Products</a>

    <ul>
    <li><a href=”#”>Cozy Couch</a></li>
    <li><a href=”#”>Great Table</a></li>
    <li><a href=”#”>Small Chair</a></li>
    <li><a href=”#”>Shiny Shelf</a></li>
    <li><a href=”#”>Invisible Nothing</a></li>
    </ul>

    </li>
    <li><a href=”#”>Contact</a>

    <ul>
    <li><a href=”#”>Online</a></li>
    <li><a href=”#”>Right Here</a></li>
    <li><a href=”#”>Somewhere Else</a></li>
    </ul>
    </li>
    </ul>
```

The CSS for this is

```
ul {
```

```
    font-family: Arial, Verdana;
    font-size: 14px; margin: 0;
    padding: 0;
    list-style: none;
}
ul li {
    display: block;
    position: relative;
    float: left;
}
li ul {
    display: none;
}
ul li a {
    display: block;
    text-decoration: none;
    color: #ffffff;
    border-top: 1px solid #ffffff;
    padding: 5px 15px 5px 15px;
    background: #2C5463;
    margin-left: 1px;
    white-space: nowrap;
}
ul li a:hover {
    background: #617F8A;
}
li:hover ul {
    display: block;
    position: absolute;
}
li:hover li {
    float: none;
    font-size: 11px;
}
li:hover a {
    background: #617F8A;
}
li:hover li a:hover {
    background: #95A9B1;
}
```

The advantage with this method is that it requires no clever programming.

Exercise

[1] Create a new web page and in the **body** of the web page copy the above list. Save the web page as **cssmenu.html** in the folder **07_Lists_and_Menus.**

[2] In the **head** of the web page, create a style section and add the above css code into it.

[3] When you display the web page in your browser you should see a horizontal menu representing the buttons which is created using html lists.

Summary

This chapter illustrates various methods that can be used to create dropdown menu lists. Using lists and css to create menus is very flexible and is the method that you will come across over and over again. Modifying and adding new menu items is very straight forward as it is just a case of editing the text.

8
Graphics

What's in this chapter:

* Graphic formats for the web.
* Basic principles of Photoshop.
* Creating roll over images.
* Background images.
* Saving images to the web.

This chapter looks at graphics for web pages. No discussion of graphics can really be done without looking at Photoshop which is the most popular in the industry. However, I try to concentrate on features that are needed for web pages, and there are other image editing programs that are available which you can use.

One is an online editor at http://pixlr.com/ which has many of the features of Photoshop and is free. It is best to use FireFox rather than IE.

Another application can be downloaded from http://www.gimp.org which is also free.

The type of features that you need to look for in graphic editors are:

- Create graphics in different formats
- Use tools to create line, brush and fill effects
- Use layers to hide and show parts of images
- Save images in formats suitable for web pages.

Graphic editors usually consists of:

- A main drawing area
- A toolbox
- A set of layers

Graphic formats used on the web

There are a number of image types that you will see on web pages, each have pros and cons which we will look at.

* gif
* jpg
* png
* bmp
* tiff

gif format

Pros of GIF:

- Can support transparency
- Can do small animation effects
- ‘Lossless’ quality–they contain the same amount of quality as the original, except of course it now only has 256 colors
- Great for images with limited colors, or with flat regions of color

Cons of GIF:

- Only supports 256 colors
- It’s the oldest format in the web, having existed since 1989. It hasn’t been updated since, and sometimes, the file size is larger than PNG.

jpg format

Pros of JPEG:

- 24-bit color, with up to 16 million colors
- Rich colors, great for photographs that needs fine attention to color detail
- Most used and most widely accepted image format
- Compatible in most OS (Mac, PC, Linux)

Cons of JPEG:

- They tend to discard a lot of data
- After compression, JPEG tends to create artefacts
- Cannot be animated
- Does not support transparency

png format

Pros of PNG

- Lossless, so it does not lose quality and detail after image compression
- In a lot ways better then GIF. To start, PNG often creates smaller file sizes than GIF
- Supports transparency better than GIF

Cons of PNG:

- Not good for large images because they tend to generate a very large file, sometimes creating larger files than JPEG.
- Unlike GIF however, it cannot be animated.
- Not all web browsers can support PNG.

bmp format

Pros of BMP:

- Works well with most Windows programs and OS, you can use it as Windows wallpaper

Cons of BMP:

- Does not scale or compress well
- Again, very huge image files making it not web friendly
- No real advantage over other image formats

tiff format

Pros of TIFF:

- Very flexible format, it supports several types of compression like JPEG, LZW, ZIP or no compression at all.
- High quality image format, all color and data information are stored
- TIFF format can now be saved with layers

Cons of TIFF:

- Very large file size–long transfer time, huge disk space consumption, and slow loading time.

Photoshop basic principles

The Photoshop display consists of:

- Left hand side is the tool bar.
- Right hand side is the layers.

Two important tool bar elements are the Pointer icon, and the Move icon.

Pointer icon: 

Move icon: 

When you create a new object, that object will be selected. To deselect it, you click on the Pointer icon.

To select something in the toolbar, click on the toolbar object, right click on your mouse and it will display other options.

Photoshop uses layers where each layer is a separate drawing area. So you can create one layer with one shape and then create another layer with another shape and place one on top of another. Layers can be hidden, or merged with other layers.

To display the layer panel select **Wndows -> Layers**

To create an effect, double click on the layer which will display a set of layer styles.

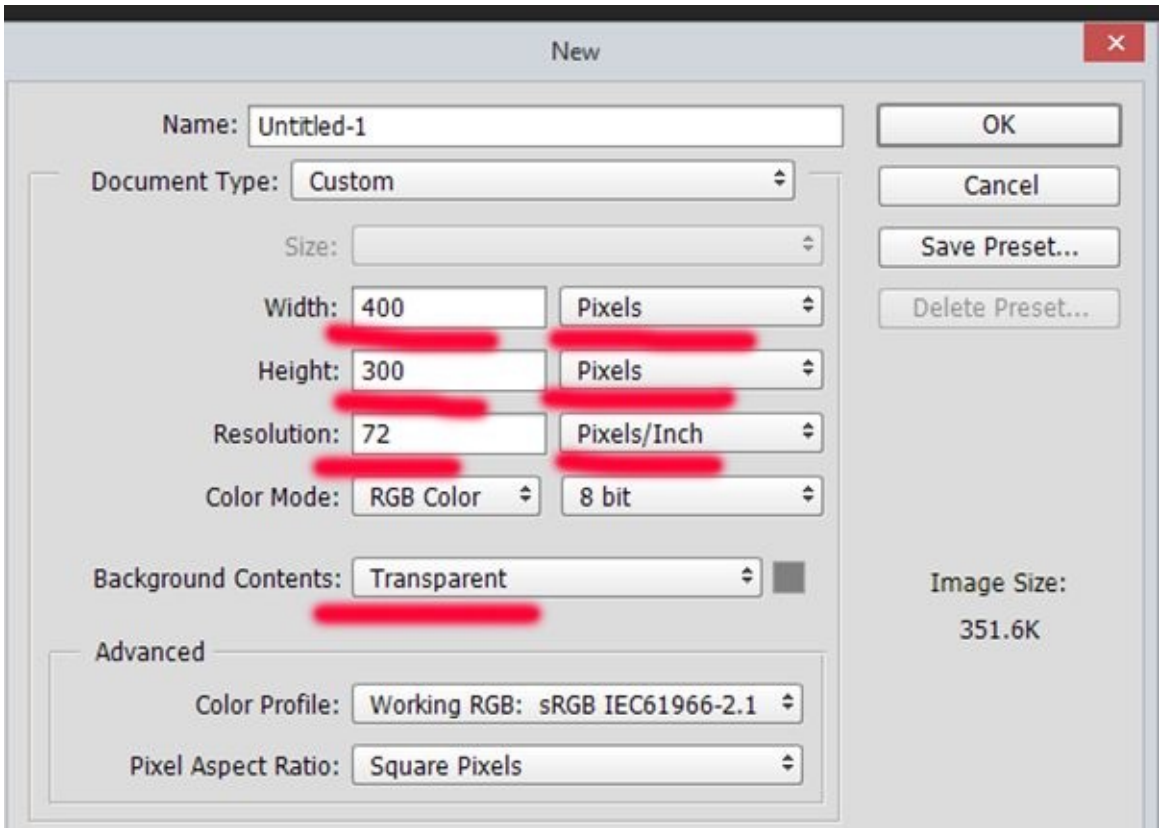
Tick a box to use that style and clicking on the style name will then display a set of options for that particular style.

Remember that standard screen sizes are now around 1280px and 800px so those sizes give an indication of the size of your images on the web.

TASK 1 - Creating shapes in Photoshop

[1] In Photoshop, create a new page, say, 400 pixels by 400 pixels and 72 pixels per inch.

NOTE: Make sure that you have selected **pixels** as the size and choose **transparency** for the background contents as shown in the screen shot.



[2] Create a new layer by clicking on the menu Layer -> New

[3] Select the layer, select a shape and draw it on the page. Note that Photoshop works in a series of layers a bit like working with transparency paper and placing images one on top of the other.

[4] Create another new layer and create another shape on the layer.

TASK 2 - Using text in Photoshop

[1] In Photoshop, create a new page, say 400 pixels by 300 pixels, 72 pixels per inch for web page work

NOTE: Make sure that you have selected **pixels** as the size and choose **transparency** for the background contents as shown in the image above.

[2] Click on the text element

[3] Enter some text and note that it creates a new layer. Some example text might be something like:

Gibbs' Plumbers

[4] For web work, you want a crisp text image and not too small

[5] Make the text a reasonable size so that it fits the width of the area and set it to Black color. You should also choose a font that is thick rather than one of the thinner fonts or otherwise you may not see the effects that we will apply later on.

[6] Double click on the layer or click on fx at the bottom of the layer panel to add a style layer. Click on the "Bevel and Emboss" check box. Open the Bevel and Emboss

properties by clicking on the check box, and then adjust the “Depth” entry. You will also want to select the “Drop Shadow” properties.

[7] Try the same process with **pixlr.com**

Code file: 08_Graphics/files/gibbs_plumbers.psd

TASK 3 - Saving for a web page

In Photoshop, the default file type is psd. This format cannot be used on web pages and so we have to export to a different format.

If you want to create an image for a web, click on **File -> Export As**, or you can select **File -> Quick Export As png**. Or you can do **File -> Save for web (legacy)** which opens up a window of different optimised images. Earlier versions of Photoshop will have slightly different menu to get to the **Save for web**.

Choose either jpg or png and select a suitable image.

Note the file size of the image when you save it. It should be as small as possible so that the web page can load quickly.

If you use jpg format you can reduce the quality of the image and hence reduce the file size of the image. On a web page, reducing the image quality does not make too much visible difference.

A file size of less the 20 kbytes to 40 kbytes is what you should be aiming for.

This is explained in more detail further on this chapter in the section about “Preparing and saving images for the web”.

TASK 4 - To create a button with Photoshop

[1] Create a new image, 100 pixels by 25 pixels with a white or transparent background.

[2] Select a suitable base color for your button. Say #bed4d2

[3] Select the rounded corner shape and draw a button in the page area.

[4] Note that if you try to fill the button shape, or try to fill the shape with color, it will not allow it.

[5] In the Layers window double click on the button shape.

[6] Select a suitable bevel, such as Drop Shadow and if you wish, select a suitable gradient color.

[7] Save the page as a psd

[8] Use the text layer to create a button with the text 'Register'

[9] Save as a web image - jpg or png

[10] Create a new web page and place your image on to the web page.

Note that it is usual to create a new folder where you place all your images. This is often called 'graphics' or 'images'

Code file: 08_Graphics/files/button_register.psd

TASK 5 - Using Layers

Photoshop uses layers to represent images.

Images and shapes can be created on layers and individually manipulated.

The separate layers can then be hidden, modified, or merged together to give particular effects.

This can be illustrated using the file:

layers_exercies.psd

Code file: 08_Graphics/files/layers_exercises.psd

[1] Open the above file in Photoshop.

[2] Shift the layers up and down the layer stack.

[3] Turn the eye icons on and off - the eye icons are on the left sides of the layers. When you can't see the eye icon, you can't see the layer.

[4] Experiment with opacity at the top of the Layers panel.

[5] Select the Eraser tool, and erase parts of the layers.

[6] If you want to start over, go to Edit > Revert.

Using the eye icon, select two layers and deselect all other layers. In the drop down list for blending options, select **Multiply** and see what happens. Select other blending options

and see what happens.

Double click on the layer to display the layer styles, and select some options. The selected options, modify that particular layer and are non-destructive, that is you can just undo the layer styles to revert to the original image.

Checking the size of images

Knowing the details of an image on a page is very useful when you trying to create another image to fit the same area.

To check the size of images on a web page:

[1] Right click on the image and then select Properties. The Details tab will display the image size in pixels and the size of the file itself.

[2] If you want to download the file, then right click on it and click on “save image as …” or “save picture as …”

Creating roll over images

Roll over images have been used for years to add interaction to a web page, particularly with buttons. There are various ways to create roll over images, and we will look at several of those.

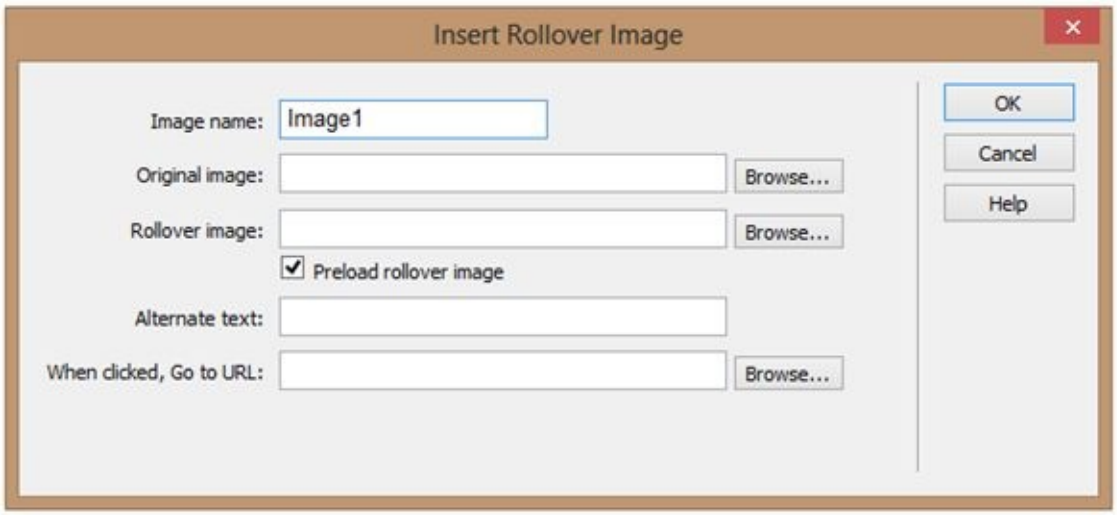
Using Dreamweaver to create a roll over image

Dreamweaver allows you to create roll over images which automatically creates JavaScript in your web page.

In Dreamweaver:

Insert -> HTML -> Rollover Image

Image name:	Any name
Original image:	Enter the url of the original image
Rollover image:	Enter the url of the rollover image
Preload rollover image:	Tick the box means that the JavaScript that is created will include preloading of the images so that it gives no delay when the image is displayed.
Alternative text:	alt text in the buttons
When clicked, Go to URL:	The destination URL



Create two buttons using Photoshop

[1] In Photoshop, create a new file 70 px by 30 px with a transparent background.

[2] Use the rounded rectangle tool to create a button effect that fits the entire image area.

[3] In the layer, double click on the square image to display the color pallet.

[4] Choose a suitable color such as a light grey.

[5] Double click on the shape to display the Layer Style.

[6] Check Bevel and Emboss and check Contour.

[7] Select the Text tool and type in the word “home” using a suitable sized font and style.

[8] Save this as a Photoshop file.

[9] Select Save for web & Devices and save this as B**utton_Original.png**

[10] Go back into the Styles Layer and select Bevel and Emboss and change the angle to -30 degrees

[11] Select Save for web & Devices and save this as B**utton_Rollover.png**

Code file: 08_Graphics/files/Button_Original.png

Code file: 08_Graphics/files/Button_Rollover.png

Code file: 08_Graphics/files/Button_Rollover.psd

Code file: 08_Graphics/files/button_rollover.html

Now that you have the two buttons you can go into Dreamweaver and create a rollover button effect by selecting the **Original image** and **Rollover image**.

If you look at the code that this has been created you will find quite a considerable amount of JavaScript. This is one technique which gives the effect but it is probably a little dated. The following method uses CSS give the same result and in some ways is more concise.

TASK 6 - Creating a roll over using CSS

This method uses only CSS styles which has some advantages to the above method that uses JavaScript. It is simpler to implement and the web page does not have to download any JavaScript code.

The button that we are going to use is **buy_now.png** which if you look at it consists of the “over” and “normal” image on one file.

The CSS works by shifting the image horizontally and hence we have to know the image style which is 110px by 35px.

In a web page you will need the following style:

```
<style type=“text/css”>

a.rollover {
```

```
    display: block;
    width: 110px;
    height: 35px;
    text-decoration: none;
    background: url("buy_now.png");
    }

a.rollover:hover {
    background-position: -110px 0;
    }

</style></style>
```

And in the body you will need the class "rollover"

Code file: 08_Graphics/files/buy_now.html

Code file: 08_Graphics/files/Buy_Now.psd

Code file: 08_Graphics/files/buy_now.png

Exercise in Background Images

The CSS **background-image** property allows you to add a background image to any HTML element which includes a page using the <body> tag.

This can be an entire page, or it can be a part of page. By default, a background image will repeat to fill the entire element.

TASK 7 - Background body image

[1] Create a new HTML 5 web page called **background01.html**

[2] Find the example image **flowers.gif** in the course material for this Chapter.

[3] In the head of the HTML page, create a style as follows:

```
<style type="text/css">
    body {
    background-image: url("flowers.gif");
    color: black;
    padding: 20px;
    font-family:Baskerville, "Palatino Linotype", Palatino, "Century Schoolbook L", "Times New Roman", serif;
    }
</style>
```

[4] In the main body of the web page, copy the body text from the text file **grasses.txt** from the course material for this Chapter.

[5] Format the text with <h1> tag for the main title, <h2> tag for the secondary titles and <p> for each of the paragraphs.

[6] View the resulting web page in your browser or upload to the web server and test it on your web server.

Code file: 08_Graphics/files/background01.html

TASK 8 - Creating a background image

In this exercise we will create an image which is thin strip and which is repeated in a horizontal direction. Background images can be made to repeat only in one direction or not to repeat at all.

[1] In Photoshop, start a new image width of 5 pixels and height of 100 pixels. It should have a white background.

[2] Select the gradient tool which should go from transparent to a solid color such as blue. Drag the gradient tool down the image so that you have white at the top and the solid color at the bottom. Instead of a gradient, you could put a pattern by double clicking on the layer and selecting Pattern Overlay. You should be able to select different patterns from the pattern box. You may also want to set the opacity of the image to something like 70%.

[3] Save the image as a web png file saving it as **vert.png**

[4] Copy the web page **background01.html** to a new web page called **background02.html**

Change the style to the following:

```
<style type="text/css">
    body {
    background-image: url("vert.png");
    background-repeat: repeat-x;
    background-color:#3399cc;
    color: black;
    padding: 20px;}
</style>
```

[5] In this example, I have set the page background to the same solid color as the gradient color. Then I have set the repeat direction to the horizontal direction only. Display the result in the browser.

TASK 9 - Background position

When an image is not being repeated, you can use the background position property to specify where in the browser window the image will be placed.

[1] Find the example image **html5_logo.jpg** from the course material for this chapter.

[2] Copy **background02.html** page to a new web page called **background03.html**

[3] Replace the style sheet with the following:

```
<style type="text/css">
    body {
    background-image: url("html5_logo.jpg");
    background-repeat: no-repeat;
    background-position: center top;
    color: #665544;
```

```
    padding: 20px;}
</style>
```

[4] Display the results in your browser.

The position of the background in this case is defined by **background-position**, when the image is not being repeated.

Experiment with **background-position** to set the position as appropriate. Note also, that you can set the position as % so:

background-position: center top;

is the same as:

background-position: 50% 50%;

Preparing and saving images for the web

Using images on a web page is essentially an optimising process. Different displays have many factors which change the appearance of the image: contrast, depth of colors, viewing angle and so on. We can apply filters and make adjustments to an image so that it looks reasonable the same on different devices.

These exercises use Photoshop but you can do them just as well in other editors such as http://pixlr.com which seems to work best in Firefox.

[1] Bring an image into Photoshop

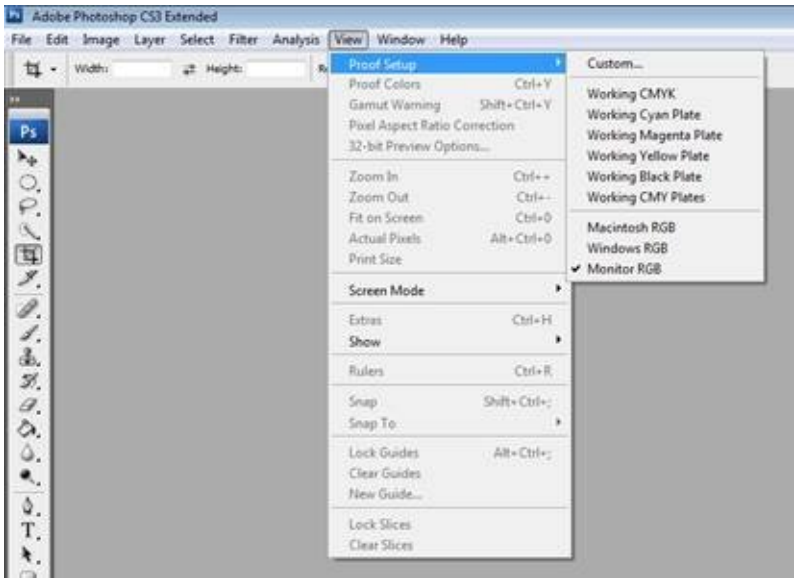
Load an image into Photoshop which should be one that has been taken from a camera.

[2] Set Monitor RGB

Certain color profiles, like CMYK are made to work better in print, while RGB works best for the web.

Open up the view menu and click on **Proof Colors** to turn on the proof colors function.

Now select **View -> Proof Setup -> Monitor RGB**



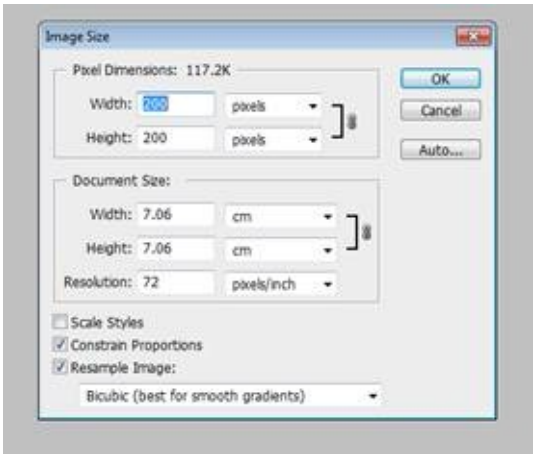
You will need to do this each time you create a new file.

[3] Crop the image

Crop the image using **crop** tool to remove unwanted parts of the image. Drag the crop tool around the area that you want to keep.

[4] Re-size the image

Selecting **Image** -> **Image size** shows size of image



Photos for the web must be of reasonable size so that viewers are able to see your art work, but the file size must not be large as they will take longer to download.

We can consider the "size" of your photo to be measured in terms of dimensions, resolutions and file size:

- **Dimensions**, usually expressed in pixels, for example 500 pixels x 400 pixels.

- R**esolution**, expressed in ppi (pixels per inch). We are aiming for a resolution of 72ppi because computer screens do not normally have higher resolution.

- F**ile size**, usually expressed in kilobytes (KB). The size is depends on the dimensions, resolution, detail and colors the image.

Generally we want a photo to be viewed on a computer screen without scrolling. So a good maximum size would be 800px wide by 600px high.

If you are creating images for other devices such as tablets or phones you obviously have to take into account their target sizes. Quite often with smaller devices, one way to deal with them is to create cropped versions of the larger size image, rather than trying to shrink the whole picture.

With mobile phones, size is going to be even more important as phones do not have as high data rates as a desktop PC.

[5] Create the re-sized image

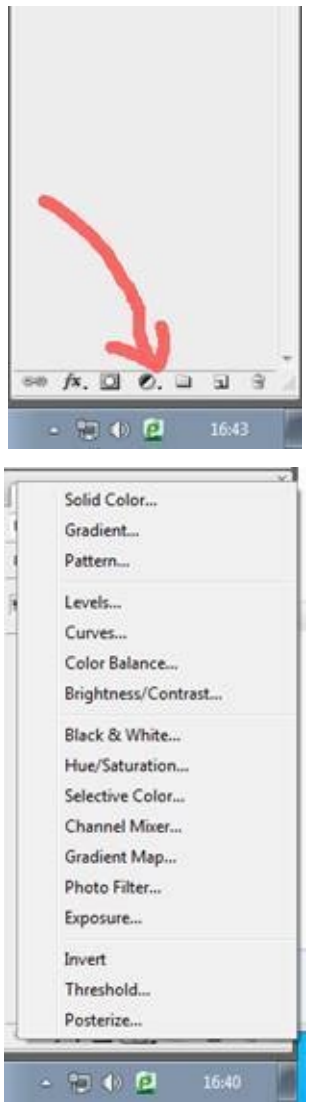
In the Image Size box as shown above, first, make sure that the "**Constrain Proportions**" and "**Resample Image**" boxes are checked. Usually select **Bicubic** or **Bicubic Sharper** as the rendering engine for shrinking the images. **Bicubic Sharper** is a good choose for sharpening the image. Make sure that "Resolution" is at 72 pixels/inch if not already set.

Last, set the image dimensions and as the "Constrain Proportions" box is checked, Photoshop automatically chooses the corresponding height.

Press OK to resize the image.

[6] Filtering the image

The following four points may or may not be required depending on the image. These apply filters and adjustments to the image to modify contrast and levels.



[a] Click on Moon symbol on layers and select curves - it creates a layer and you can adjust the contrast. Normally you adjust it to look like an S-curve.

[b] Click on Photo Filter… and add a warming filter or other filter to the image.

[c] Click on levels, which allow you to make some adjustments to the levels.

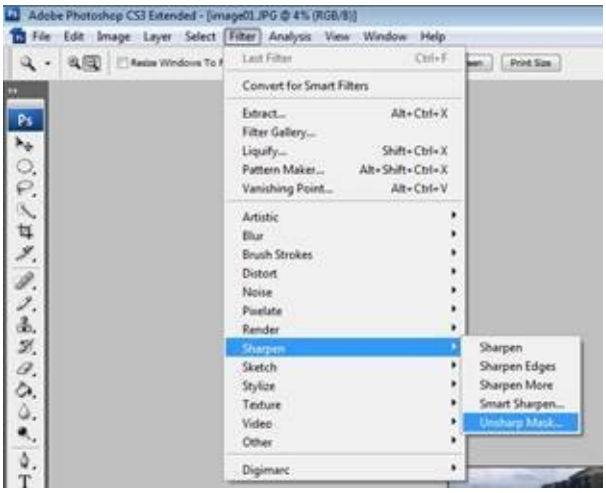
[d] Click on Brightness/Contrast which allows you to make some adjustments to the brightness and contrast.

[7] Sharpen your image

When you optimise a file for a web page, it usually becomes much smaller than the original and in the compression process you may find some blurring.

You can sharpen the image using a Photoshop filter.

From the toolbar, select **Filter**, then **Sharpen**, then **Unsharp Mask**



This displays the Unsharp Mask window.

Set “Amount” to 200%, “Radius” to 0.2 pixels, and “Threshold” to 0 levels.

You will often want to apply the Unsharp Mask more than once, each time sharpening the image until it looks correct. You may apply it twice, three or even four times, but make sure that it does not become too jaggy as the result does not look good.

[8] Alternative method for sharpening

As with all methods in Photoshop, there is another way to sharpen an image but is a more advanced technique which uses layer masking. This method can be more selective rather than the general approach described above but is more time consuming.

First duplicate the layer and then apply an Unsharp Mask to the layer and sharpen the image.

Now create a layer mask using the marque tool, and using the paintbrush tool mask out areas that show over-sharpening. Using the paintbrush tool at reduced opacity, reduce the sharpening effect on selected areas. Or set the paintbrush opacity to 100% to remove the sharpening effect completely from selected areas. You can continue to fine tune the image to give the desired effect and give the best quality.

[9] Save the file

Select **File -> Save for Web** which will display a set of screen images at different configurations:

For later versions of Photoshop, this comes under the **File** -> **Export** sub menu.

The **Save for Web** allows you to make many configuration changes which change the file size of the image. As you make the changes you can monitor how it affects the file size.

In this example we will select JPEG format as the file type. You can now select the "Quality" of the image which is a number up to 100. Notice that if you select PNG file type you will not have that configuration option because you cannot change the quality of a PNG file type.

As you change the Quality value, the file size reduces but the image quality reduces as well. This is important as smaller file sizes will be quicker to download on to the device. The resolution and the color rendition of different devices usually mean that any reduction in quality will not be noticed.

I usually try for a quality of 60 upwards. Most images should not be much more than 40 Kbytes in file size unless it is a big image. You also have to take into account how many images are on the web page as the total download size will be greater the more images you have.

The other factor you will want to take into account is what device the image will be displayed on. If it is a mobile phone device then you will certainly need an image of low file size as the bandwidth for mobile phones will be less.

The **Save for Web** window has the facility to show 4 versions of the image which allows you to preview your selection.

Once you have finalised the image, click on "Save" to save the image to your computer.

[10] Add your image to a web page

You can now add the images to one of your web pages to test the results in a browser. You may want to test your images on different devices, desk top computer, tablet and smart phone.

Summary

This chapter has looked at creating images, buttons and backgrounds using Photoshop. If you looking to make a career in web development, you will find that Photoshop is the standard in graphic design. It is always worth trying to do a course on Photoshop if you can.

9
Layouts

What's in this chapter:

* Layout concepts.
* Good and bad design practices.
* A web site project.

Web site design requires a certain mind set to understand what a visitor will find attractive and hence will keep them on your site. This includes concepts such as color harmony, attractive positioning of images and content, good navigation and site organisation. This chapter looks at these ideas.

Page Sizes

- Pages are usually aimed at 960 to 1000 pixels when designing for desk top computers.
- Recent trends are to **responsive web design** which use layouts that adjust to browser size and device and where elements are modified to optimise for the particular device.
- A fixed width layout - elements can be accurately controlled but can get large gaps particularly around the edges of the web page and it will not work well on devices that are not able to display the defined width.
- A liquid layout - pages fill the size as the browser changes size and is good for desktop displays but only really works if used with responsive techniques.

Mobile devices present particular problems in design:
- the size of the image in pixel height and width needs to be appropriate for the device size.
- the physical size of images in bytes needs to be kept as low as possible as mobile download speeds are not as good as fixed lines.
- menus can be an issue particularly with drop down types on touch screens.

General layouts concepts

Traditional Aspects:

There are many concepts to layout but we can consider the following:

- **A grid layout system** – positioning items on a web page can be difficult if you have no inbuilt structure on your page. However, there are a number of grid system style sheets that you can download from the internet such http://960.gs/. These grid systems make placement of text and images much easier. Responsive design frameworks have a grid system built in.

- **A standard color scheme** – the idea with defining a color scheme is to give harmony throughout the design. The colors that you select can give a certain impression such as warm or cool. There are a number of web sites which have interactive applications where you start with a base color and the application then shows a set of complimentary colors.

  An example of a color wheel is shown here:

  http://paletton.com/

  Choose the color scheme and then choose cold or warm and then click export.

- **A hierarchy of information** - your pages need to be logical in the way that they are grouped using menu and sub menus. The user needs to easily be able to see where they are in the hierarchy of the web site. This can be achieved using a menu system which highlights the current location. It may also have a "breadcrumb" system. This becomes more important the larger the web site.

Site details:

- A choice of clean and legible fonts.
- Stylized icons.
- CSS buttons with or without rounded corners.
- Subtle gradients in buttons and backgrounds.
- Links to Facebook, Twitter, email etc.

- Contact forms with border around groups of input boxes.
- Use of rounded corners.
- Clear strong photos and images.

An example of this kind of site would be http://getflow.com

Color theory?

We can think of color theory as:

Complementary - the way we see colors in relationship with others.
Contrast - the way we see how one color compares with another.
Vibrancy - defines the brightness or loudness of the colors.

How to select color schemes

There are various approaches to getting a list of suitable colors. Web sites such as http://paletton.com/ look at the following:

Triadic - this is where we have three colors at separate ends of the spectrum
Complementary - we select a range of complementary colors
Analogous - these are selections with the same area of the color scheme

For each we move the selections and export our results.

Web Design - Good Practice

Good Web Pages

- Contains content that is useful and relevant to the subject and does not duplicate or copy material from other sources.
- Have changing and up to date information in the form of blogs or news items.
- Provides content and material aimed at the subject audience of the web site.
- Includes contact information, links to social media and ways for the visitor to interact with the site owner.
- Have a page about the company or individual who owns the site.
- Have pages that are well organized and easy to use with good navigation to help the visitor.
- Uses images, photos and icons to compliment the text.
- Uses fonts that are of a good size and contrast.
- Have all pages that are consistent in their design and use of styles and layouts.

Bad Web Pages

- Uses font colors and backgrounds which render them illegible.
- Have links to pages that are out of date or don’t exist.
- Have too much or too little contact on a page.
- Overuse the placement of search keywords in the text.
- Have confusing hierarchy of menus which hide the structure of the site.
- Do not have a clear purpose for the site.
- Have titles and heading that are misleading.
- Contain out of date information
- Have no contact information.
- Break copyright laws.

Responsive Design

If you go back 10 years, most web sites were viewed on the similar sized devices. Nowadays the devices are different sizes and orientation.

Responsive design is the method to try to overcome the problem of different device sizes without having to create a web site for each device. Responsive web pages automatically adjust to suit the particular width of the device using fluid layouts, resizable images and JavaScript. Twitter Bootstrap for example has the ability to resize images within the element by the addition of a special class to the image. Image resolution is important to help with download times in mobile communications.

We look at Responsive Design in a later chapter in more detail.

How do we find a layout for our site?

We have created layouts from scratch in our previous exercises, but quite often this is just not practical and we find a template on the internet, either purchasing it or downloading it from a free web site. We then modify the template for our own purposes.

When you look at templates that have been developed correctly, you will see that the CSS can be very complex. It is not always realistic for you to try to develop something from scratch particularly when it comes to understanding the slight differences in the way that particular browsers render different styles.

There are now many sets of "frameworks" that you can download particularly for Responsive Web Layouts which you can base you site on, modifying it and adding additional styles for you own particular purposes.

By carefully considering the way you set up your site, you should be able to add additional styles without altering those styles that have already been defined.

So for example, you could have a web page that uses a style sheet given to you in a download. You can then set up another style sheet called **additional.css.** You place this style sheet underneath the first one and then you add your new styles or overriding styles to **additional.css**. The advantage is that if the downloaded template styles change you can update those files without overwriting **additional.css**.

Some Free Template Sites

If you are looking for CSS layouts to base you website on, there are many sites that allow you to download the code for free. If you search on Google for something like 'css templates' it should return quite a few that you can look at.

Here are some that give a good selection.

http://www.free-css.com/

This has free layouts but some are a little old so you may have to make sure that they work with modern browsers.

As an exercise, go to the free template list and find "simple company" template which is a responsive design layout based on Twitter Bootstrap. Download it, then extract the zip file and look at the index.html file.

Note that you can do a search for "simple company" with a Responsive layout.

http://www.templateworld.com/index.html

This site has a good range of templates which you can download.

http://www.freehtml5templates.co.uk/

This site is all HTML 5 templates – These may have a link back to their site but you can remove it.

Responsive Design Frameworks

Twitter Bootstrap is a framework layout system

http://getbootstrap.com

for version 3

HTML Kick Start is a simpler responsive design framework

http://www.99lime.com/elements/

HTML 5 Up is another framework type system which is also a responsive css template.

http://html5up.net/

flypixel - Awesum design freebies by kickass designers from all over the world

http://flypixel.com/

A Web Site Project

This section looks at how you might tackle working to a specification and designing a web site to meet particular needs.

The outcomes are:

[1] Create layout, structure and styles for a website consisting of a number of pages depending on the requirements.
[2] Use website software tools to prepare content for a website.
[3] Publish a website.

Web site specification

Choose a subject for a web site, e.g. a site about:

* A one page job CV.
* A personal home site with a blog.
* A portfolio of your work from courses that you have attended with an index page.
* Your dog.
* A favourite film.
* A holiday.
* And so on….

As an example, the site could have the following specification. However, you can change this to fit in to your requirements depending on the kind of site you are developing.

- At least 6 pages.

- Each page should have the same layout.

- The layout should be a maximum of 750 pixels wide and centered in the browser. The 750 pixel wide area should be divided into two columns so that the right column can be used as a vertical navigation bar and the left area will be the main content area. You may want to change this layout depending on the type of site you are working on.

- The header should have a logo of some sort, either built from a Photoshop image or using a font from Google which we look at in a later chapter. The header should be the same on all pages.

- Navigation between pages may use buttons with an animated mouse-over effect or constructed using a CSS rollover system. Every page should have the same navigation system.

- The footer should use a smaller font than the rest of the page.

- Define a font and use the same for everything.

- Use a suitable color scheme.

A web site consisting of 6 pages

General considerations

The web site should have a good file structure in that the images, style sheets and associated files should be located in relevant folders with sensible names.

Each web page and folder should be named so that they match ‘standard conventions’ (no spaces, all lower case etc.)

Layout choice

You need to decide a layout using DIVs which will fit into your requirements. In this project, a 750 pixel wide area has been suggested but, you may modify this as you see fit. Or you may want to look at downloading a free template from the many web sites that provide these.

Using Wire Frame design

A wire frame is a way of laying out web pages using a kind of flow diagram. It is an approach that allows you to see how pages are linked together and to test the navigation.

There are a number of online tools available such as Lumzy

http://www.lumzy.com/app/

This allows you to create simple layouts and prototype the web pages. The more advanced programs allow you to have multiple developers and have a certain amount of interaction so that you can demonstrate your concepts to a customer before spending time on the final version.

Suggested Pages

[1] index.html page

This is the home page or initial page of the web site and will describe the purpose of the site. This will be either index.html, index.htm, default.html, default.htm.

[2] Products or services page

If you are selling items then you may need to incorporate buy now buttons from PayPal which can be easily created through their web sites.

[3] FAQ or Customer Services Information

This may contain copyright details of products you may be selling, product returns policy, privacy statements, data protection statement and so on.

[4] Contacts page

This page should contain all the ways in which a customer can contact you, e.g. email address, telephone numbers and postal address if required etc.

[5] Gallery page

You may want a gallery page which can be implemented with one of the many free JavaScript libraries that can be downloaded from the web.

[6] About us page

A general page which describes yourself or your company.

Site map page

Many sites have a site map page that is used to define the URL of each page of the site. The site map can be created in a standard format and then submitted to search engines.

Site navigation

There should be consistent navigation around the site. The navigation should be clear as to which page you are on and should match your color scheme.

Site optimisation and submission to search engines

Identify keywords that you think visitors will use to get to your site. Create titles, <h1> tags and create text that reflects the purpose of the site. Submit to search engines, set up Google analytics and use web master tools to monitor the web site.

Things to consider

It does make it easier if you have a client that requires a web site rather than you being the client and designer. Having a client means that they can tell you what they require.

Where do I get my images from?

Google has endless images which can be used, however you have to be aware of copyright – you can search Google for images with no copyright by first loading up Google and clicking on Images button. Then do a search and in the results a further menu will appear. Click on “Search tools” and a menu bar will appear where one of the options is to select usage rights. From there you can filter by usage rights.

There are other sources of images such as flickr and stocksnap web sites.

Note that you need to check the licence for the photos to make sure that they are free to download, modify and use as you want usually under one of the creativecommons.org licences.

How many pages should the site have?

If the site grows you will need to work out how to accommodate the pages in a menu system or you may want to just keep to a few pages?

What layout should the site have?

You may want to create your own layout or modify an existing one.

How do I research other similar web sites?

You can search Google to look at examples of other people’s work.

Summary

This chapter has looked as some issues to do with deciding on a layout. There are many examples on the internet to base you design on and you can download templates which you can then modify.

10
Adding Functionality

What's in this chapter:

* Adding Web Fonts, Google and Adobe
* Introduction to JavaScript.
* Introduction to JQuery.
* Adding movies and audio to Web Pages.

This chapter is a miscellaneous group of topics that can be used to improve the appearance and functionality of web pages. It looks at using external fonts, JavaScript and JQuery.

Web Fonts from Adobe and Google

There are limited fonts that can be used on web pages because there is only a relatively small set of core fonts that are available on all computers. There are ways to add fonts to pages: you can create them as a graphic image and use any font you want. However, the method is not practical if you have to do it for every h1 element on your web site.

One solution that has become popular is the use of external fonts that are made available using CSS. Both Adobe and Google have a set of these fonts that you can add to your web pages for no cost.

Google Fonts

Google fonts can be found at http://www.google.com/fonts/

Using the fonts is very easy:

- Choose a particular font that you want
- Click on the “Quick use” button
- Copy the give stylesheet and add into <head> of your web page.
- Copy the font family definition name and add to the style.

For example:

The font Carrois Gothic SC gives a style sheet as:

<link href=‘http://fonts.googleapis.com/css?family=Carrois+Gothic+SC’ rel=‘stylesheet’ type=‘text/css’>

Integrate your fonts into the styles with something like:

```
h1 {
   font-family: ‘Carrois Gothic SC’, sans-serif;
    font-size: 36px;
}
```

Code file: 10_Adding_Functionality/files/google_font_example.html

Note that it may not work in all browsers but it will drop back to a default appearance if the browser is not able to support it.

You should be aware that any font will increase the time to load the page, so you would not add too many different fonts onto the same web page.

Google documentation

https://developers.google.com/fonts/docs/getting_started

This page provides you with a description of Google fonts and how to use them.

In particular, note the beta features:

https://developers.google.com/fonts/docs/getting_started#Effects

These are font effects that are available, but they do not work on all browsers. Each font has a list of compatible browsers.

Adobe Edge Web Fonts

Adobe has a similar system which is located at https://edgewebfonts.adobe.com/

- Choose the font that you want.
- Copy the given JavaScript file to the <head> of your web page.
- Copy the font family definition to the style.

For example:

For the ‘asset’ font family, the JavaScript is:

<script src=”//use.edgefonts.net/asset.js”></script>

The font family is then added to the style sheet like the following:

```
<style type=“text/css”>
h1 {
    font-family: font-family: asset, serif;
    font-size: 36px;
}
</style>
```

Code file: 10_Adding_Functionality/files/adobe_edge_font_example.html

As with Google fonts there will be a payload added to the web page, so you would not add too many different fonts.

TASK 1 – Add some fonts to your own web page.

Find a web page that you have been working on and add a web font to it. You may find that it is better to use the web fonts for main logo text or heading rather than for paragraph elements.

Introduction to JavaScript

In this book we will not be teaching **JavaScript** but it is an important to have an idea of its purpose because it is used on many web pages especially in **JQuery**.

What is JavaScript?

JavaScript is a programming language which runs on your PC, table or smart phone. The program script is downloaded with a web page onto your computer. You create **JavaScript** on the web page and you can have separate **JavaScript** files.

What is JavaScript used for?

JavaScript is used to give the effects that we like on modern web pages such as gallery slideshows, giving feedback on forms when you enter data, controlling HTML elements on the web page and so on.

How do we use JavaScript?

JavaScript code is written in the web page itself. We often use external **JavaScript** files (js files) to add the scripts to the web page.

A simple example of JavaScript

[1] Create a new web page called **javascript.html**

[2] Give it an appropriate <title>

[3] Underneath the <title>…</title> tag inside the <head>…</head> tags enter:

```
<script language="javascript" type="text/javascript">
    alert("A JavaScript Pop-up");
</script>
```

When you put JavaScript into the head of a web page, you always have to identify it as a script in this way.

[4] In the <body>…</body> tags you can enter any text that you want.

[5] Run your web page in the browser and you should get an alert message on your page.

```
<!DOCTYPE HTML>
<html>
```

```
<head>
<meta http-equiv=“Content-Type” content=“text/html; charset=utf-8”>
<title>A JavaScript Page</title>
<script language=“javascript” type=“text/javascript”>
    alert(“A JavaScript Pop-up”);
</script>
</head>
<body>
<h1>An example of Javscript</h1>
</body>
</html>
```

Code file: 10_Adding_Functionality/files/javascript.html

Using JQuery

What is JQuery?

JQuery is based on **JavaScript** which is a client side programming language that runs in the web browser. In the past, this has caused problems as different browsers implement browser standards differently and hence you had to write and test code for different browsers.

JQuery is a **JavaScript** framework which uses a set of pre-defined functions compatible with modern browsers. When you use **JQuery** you can be confident that it will work on the majority of browsers that are being used today, and as new versions of browsers are released the base framework will be updated to ensure it works with those as well.

This section gives a brief introduction to **JQuery**

Uses of JQuery

* Creating forms to do calculations on the client PC
* Validating data before data is submitted to a database
* Creating displays that give a better user experience, for example, displaying error messages in data validation
* Using the user interface plug in such as accordion, date picker, progress bar and so on as shown in http://jQueryui.com/ and other web sites such as http://jquery-plugins.net/
* Using the many plug ins that have been developed by other developers
* Developing AJAX applications

Incorporating JQuery

There are two ways to incorporate JQuery script into a web page, the first method is to download the JavaScript file from the JQuery web site and include in as a linked file, and the second method is to use a hosted JavaScript file which is linked onto your web page.

So for example to include jquery from an external hosted system, use:

<script src="http://code.jquery.com/jquery-1.8.3.js"></script>

An illustration of JQuery

In this tutorial we will not go into the details of JQuery, however it is possible to create working examples very easily from the JQuery user interface web site.

The site

http://jqueryui.com/

includes a number of common User Interfaces which you can incorporate into your own web site with minimum effort and very little programming. In fact you don't actually need to understand JQuery to be able to use these interfaces.

You should by now know enough about the structure of a web page, and be able to read example web pages and then place that code into your own web pages. These JQuery examples require you to do that.

UI Accordion interface

As an example, we will use the Accordion interface which is illustrated below. Just copy and save as an html page called **accordion.html**. This example comes from http://jqueryui.com/accordion/ which lists all the various options available.

Click headers to expand/collapse content that is broken into logical sections, much like tabs. Optionally, toggle sections open/closed on mouseover.

The underlying HTML mark-up is a series of headers (H3 tags) and content divs so the content is usable without JavaScript.

```
<!doctype html>
<html lang="en">
<head>
    <meta charset="utf-8" />
    <title>JQuery UI Accordion - Default functionality</title>
    <link rel="stylesheet" href="http://code.jquery.com/ui/1.9.2/themes/base/jquery-ui.css" />
    <script src="http://code.jquery.com/jquery-1.8.3.js"></script>
    <script src="http://code.jquery.com/ui/1.9.2/jquery-ui.js"></script>
    <script>
    $(function() {
    $( "#accordion" ).accordion();
    });
    </script>
</head>
<body>

<div id="accordion">
    <h3>Section 1</h3>
```

```
    <div>
    <p>
Mauris mauris ante, blandit et, ultrices a, suscipit eget, quam. Integer ut neque. Vivamus nisi metus, molestie vel, gravida in, condimentum sit amet, nunc. Nam a nibh. Donec suscipit eros. Nam mi. Proin viverra leo ut odio. Curabitur malesuada. Vestibulum a velit eu ante scelerisque vulputate.</p>
    </div>
    <h3>Section 2</h3>
    <div>
    <p>
Sed non urna. Donec et ante. Phasellus eu ligula. Vestibulum sit amet    purus. Vivamus hendrerit, dolor at aliquet laoreet, mauris turpis porttitor  velit, faucibus interdum tellus libero ac justo. Vivamus non quam. In suscipit faucibus urna.</p>
        </div>
        <h3>Section 3</h3>
        <div>
          <p>
Nam enim risus, molestie et, porta ac, aliquam ac, risus. Quisque lobortis.
Phasellus pellentesque purus in massa. Aenean in pede. Phasellus ac libero
ac tellus pellentesque semper. Sed ac felis. Sed commodo, magna quis lacinia ornare, quam ante aliquam nisi, eu iaculis leo purus venenatis dui.</p>
          <ul>
            <li>List item one</li>
            <li>List item two</li>
            <li>List item three</li>
          </ul>
        </div>
        <h3>Section 4</h3>
        <div>
          <p>Cras dictum. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Aenean lacin mauris vel est.
</p>
<p>
Suspendisse eu nisl. Nullam ut libero. Integer dignissim consequat lectus.
Class aptent taciti sociosqu ad litora torquent per conubia nostra, per        inceptos himenaeos.</p>
        </div>
    </div>


</body>
</html>
```

Code file: 10_Adding_Functionality/files/accordion.html

Menu Interface

jqueryui includes a menu system as described at http://api.jqueryui.com/menu/

Copy the following text into an html page called **menu.html**

```
<!doctype html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>menu example</title>
<link rel="stylesheet" href="http://code.jquery.com/ui/1.10.2/themes/smoothness/jquery-ui.css">
<style>
.ui-menu {
width: 200px;
}
</style>
<script src="http://code.jquery.com/jquery-1.9.1.js"></script>
<script src="http://code.jquery.com/ui/1.10.2/jquery-ui.js"></script>
</head>
<body>
<ul id="menu">
<li><a href="#">Item 1</a></li>
<li><a href="#">Item 2</a></li>
<li><a href="#">Item 3</a>
<ul>
<li><a href="#">Item 3-1</a></li>
<li><a href="#">Item 3-2</a></li>
<li><a href="#">Item 3-3</a></li>
<li><a href="#">Item 3-4</a></li>
<li><a href="#">Item 3-5</a></li>
</ul>
</li>
<li><a href="#">Item 4</a></li>
<li><a href="#">Item 5</a></li>
</ul>
<script>
$( "#menu" ).menu();
</script>
</body>
</html>
```

Code file: 10_Adding_Functionality/files/menu.html

Date Picker

On the web site http://jqueryui.com/datepicker/ there is an example on creating a date picker. Create a web page and create the date picker, then modify the script to add in the dateFormat to give a UK format rather than the default US format.

You will need to look up the API documentation at http://api.jqueryui.com/datepicker/ and look at the dateFormat method to see how to do it.

Adding movies to a web pages

In HTML 5 the process of adding movies to web pages has become very easy.

Previous versions of HTML required plugins such as flash but HTML 5 introduces the video tag.

Use the following syntax:

```
<video width=“320” height=“180” controls>
  <source src=“movie.mp4” type=“video/mp4”>
  <source src=“movie.ogg” type=“video/ogg”>
  <source src=“movie.webm” type=“video/webm”>
Your browser does not support the video tag.
</video>
```

The video tag has a width and height which you should always try to define, otherwise your page may change its appearance if the video is of a different size than you expected.

The **controls** attribute will add in the video controls for you.

The other attribute is **loop** which causes the video to loop continuously.

Currently, the formats that are supported are by HTML 5 mp4, ogg and webm.

Note that not all browsers support all formats so normally you would provide multiple versions of the video in the different formats.

The full html 5 page is:

```
<!DOCTYPE HTML>
<html>
<head>

<meta http-equiv=“Content-Type” content=“text/html; charset=utf-8”>
<meta http-equiv=“X-UA-Compatible” content=“IE=Edge”/>

<title>Movie example</title>

</head>

<body>

<video width=“320” height=“180” controls>
  <source src=“movie.mp4” type=“video/mp4”>
  <source src=“movie.ogg” type=“video/ogg”>
Your browser does not support the video tag.
</video>
```

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

Code file: 10_Adding_Functionality/files/movie.html

Note for it to work in IE9 you have to put

<meta http-equiv="X-UA-Compatible" content="IE=Edge"/>

In the head of the web page.

Adding audio to a web pages

In HTML 5 the process of adding audio to web pages has become very easy.

Previous versions of HTML required plugins such as flash but HTML 5 introduces the audio tag.

Use the following syntax:

```
<audio controls>
  <source src="mp3_03.mp3" type="audio/mpeg">
     <source src="mp3_03.mp3" type="audio/ogg">
Your browser does not support the audio element.
</audio>
```

The **controls** attribute adds in the audio controls for you.

Currently, the formats that are supported are by HTML 5 mp3, wav and ogg.

Note that not all browsers support all formats so normally you would provide multiple versions of the audio in the different formats.

The full html 5 page is:

```
<!DOCTYPE HTML>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<meta http-equiv="X-UA-Compatible" content="IE=Edge"/>
<title>Audio example</title>
</head>

<body>



<audio controls>
  <source src="mp3_03.mp3" type="audio/mpeg">
  <source src="mp3_03.mp3" type="audio/ogg">
Your browser does not support the audio element.
</audio>



</body>
```

```
</html>
```

Code file: 10_Adding_Functionality/files/audio.html

Summary

This chapter introduced methods to add functionality to your web page by looking at external font systems and JQuery. These add an extra touch to your site which make it look much more professional. You don't have to be a programmer to use JQuery as developers have created applications that can be added to your web site by cutting and pasting.

11
.htaccess Files

What's in this chapter:

* What can be done with an .htaccess file.
* Example of .htaccess files.
* Password protect with a .htpasswd file.
* Page status codes.

This chapter is not actually about HTML or anything to do with HTML pages but is about a special file that is used on Apache web servers. It is a very useful file and provides many features such as redirecting pages from one location to another or defining a custom 404 error page. The great thing about it is that it is a text file and can be easily created.

The file is .htaccess

NOTE the full stop (.) at the front of the file name.

NOTE that .htaccess files only work on Linux type web servers, not on Windows web servers.

One use of an .htaccess file is that it can be used for 301 redirects which tell a web browser that the page has a different file name or folder name - it is used in situations where you move a page and you want Google to correctly understand the move.

As an Apache server configuration file, .htaccess is very powerful but even the slightest syntax error like a missing bracket can result in your web site not displaying correctly or at all.

Since .htaccess is a configuration file, make sure your FTP client is configured to show hidden files. This is usually an option in the program's preferences/options.

When you create the .htaccess file, you will need to upload it to your web server. Often the file is located at the root of the web server, but there may be situations where you only want it to effect a particular folder in which case it will go into that folder.

Things that can be done with an .htaccess file

- Permanently redirect or temporarily redirect one URL to another - useful for informing search engines about which files to redirect to
- Prevent browsers displaying a list of folders and files on your web site.
- Password protect a folder of a web site.
- Create a custom 404 error page.

To create an .htaccess file

[1] Create a new file called .htaccess on your local PC using a text editor. Note that an .htaccess file has a dot at the first character.

[2] For a Windows PC you may not be able to create a file with .htaccess, so create a file called .htaccess with the intention of uploading it to the server and then changing the file name on the server.

Examples of .htaccess files

301 is permanent redirect

```
# Redirect old file path to new file path
Redirect 301 /oldpage.html http://www.yoursite.com/newpage.html
```

302 is temporary redirect

```
# Redirect old file path to new file path - not normally used
Redirect 302 /oldpage.html http://www.yoursite.com/newpage.html
```

Defining a custom 404 error page

```
# Custom 404 error page
ErrorDocument 404 /errordocs/error404.htm
```

Stop users displaying the directory listing of a web site

```
# Stop directory listings
IndexIgnore *
```

Stop users displaying the details of an .htaccess file

```
# Stop displaying of htaccess file
```

```
<Files .htaccess>
order allow,deny
deny from all
</Files>
```

To protect your admin area you can create an .htaccess / .htpasswd file.

One feature that is quite handy is the ability to password protect an area of a web site. This can be done using .htaccess / .htpasswd files. The system using “Basic Authentication” and provides some protection to your folder.

Create an .htaccess file

Go to http://www.htaccesstools.com/htaccess-authentication/

In the first box enter some optional text which gets displayed in the login box

In the second box you need to enter the file path name to a second file htpasswd file. The htpasswd file is where the list of users / passwords is to be located.

Click on the button, copy the text and place it into a text file. Name this file .htaccess and upload it to the folder that you want to protect. In this case this would be the /admin/ folder.

An example of this is:

```
AuthType Basic
AuthName “Protected Area”
AuthUserFile /admin/.htpasswd
Require valid-user
```

Create the .htpasswd username / password file

Go to http://www.htaccesstools.com/htpasswd-generator/

In the first box enter the username.

In the second box enter the password.

Click the button, copy the text and place it into a text file. Name this file .htpasswd and upload it to a suitable location on your server. This can be anywhere on your server, but usually in a folder that is outside the web space. The location is the same location as was entered in (1) above.

The folder as defined in (1) should now be protected by the username / password as defined in (2)

paul:$apr1$t/B1YqZj$S6wiKCAcJu.4.eoohNpEF.

Page Status Codes

You may be wondering what 404 pages are and what are page status codes.

When you make a HTTP request to a web server for a web page, the response from the server will be one of the pages status codes. Normally the status will be 200 which means that the request was successful but there are many error codes.

The codes are divided up into:

1xx Informational
2xx Success
3xx Redirection
4xx Client Error
5xx Server Error

The following is a brief description of many of the page error codes although the only ones you are likely to see are 404 (page not found) and 500 (internal server error).

Code	**Meaning**	**Description**
100	Continue	Confirms the client about the arrival of the first part of the request and informs to continue with the rest of the request or ignore if the request has been fulfilled
101	Switching Protocols	Informs the client about the server switching the protocols to that specified in the Upgrade message header field during the current connection.
200	OK	Standard response for successful requests
201	Created	Request fulfilled and new resource created
202	Accepted	Request accepted, but not yet processed
203	Non-Authoritative Information	Returned meta information was not the definitive set from the origin server.
204	No Content	Request succeeded without requiring the return of an entity-body
205	Reset Content	Request succeeded but require resetting of the document view that caused the request
206	Partial Content	Partial GET request was successful
300	Multiple Choices	Requested resource has multiple choices at different locations.
301	Moved Permanently	Resource permanently moved to a different

		URL.
302	Found	Requested resource was found under a different URL but the client should continue to use the original URL.
303	See Other	Requested response is at a different URL and can be accessed only through a GET command.
304	Not Modified	Resource not modified since the last request.
305	Use Proxy	Requested resource should be accessed through the proxy specified in the location field.
306	No Longer Used	Reserved for future use
307	Temporary Redirect	Resource has been moved temporarily to a different URL.
400	Bad Request	Syntax of the request not understood by the server.
401	Not Authorized	Request requires user authentication
402	Payment Required	Reserved for future use.
403	Forbidden	Server refuses to fulfil the request.
404	Not Found	Document or file requested by the client was not found.
405	Method Not Allowed	Method specified in the Request-Line was not allowed for the specified resource.
406	Not Acceptable	Resource requested generates response entities that has content characteristics not specified in the accept headers.
407	Proxy Authentication Required	Request requires the authentication with the proxy.
408	Request Timeout	Client fails to send a request in the time allowed by the server.
409	Conflict	Request was unsuccessful due to a conflict in the state of the resource.
410	Gone	Resource requested is no longer available with no forwarding address

411	Length Required	Server doesn’t accept the request without a valid Content-Length header field.
412	Precondition Failed	Precondition specified in the Request-Header field returns false.
413	Request Entity Too Large	Request unsuccessful as the request entity is larger than that allowed by the server
414	Request URL Too Long	Request unsuccessful as the URL specified is longer than the one, the server is willing to process.
415	Unsupported Media Type	Request unsuccessful as the entity of the request is in a format not supported by the requested resource
416	Requested Range Not Satisfiable	Request included a Range request-header field without any range-specifier value
417	Expectation Failed	Expectation given in the Expect request-header was not fulfilled by the server.
422	Un-processable Entity	Request well-formed but unable to process because of semantic errors
423	Locked	Resource accessed was locked
424	Failed Dependency	Request failed because of the failure of a previous request
426	Upgrade Required	Client should switch to Transport Layer Security
500	Internal Server Error	Request unsuccessful because of an unexpected condition encountered by the server.
501	Not Implemented	Request unsuccessful as the server could not support the functionality needed to fulfill the request.
502	Bad Gateway	Server received an invalid response from the upstream server while trying to fulfill the request.
503	Service Unavailable	Request unsuccessful to the server being down or overloaded.
504	Gateway Timeout	Upstream server failed to send a request in the time allowed by the server.

505	HTTP Version Not Supported	Server does not support the HTTP version specified in the request.

Summary

This chapter gives an overview of the htaccess file that is used on Linux type web servers. This type of file allows you to do all kinds of functions that are useful with web sites, such as redirecting pages, defining a protected area and so on.

12
Site Set Up

What's in this chapter:

* Domain names and web space.
* Page set up.
* Search Engine Optimisation.
* Testing tools.
* Structured Data Mark-up and Microdata

Domain Names and Web Space

This section looks at what you will need to consider when you purchase a domain name and web space for your up and coming web site.

There are essentially two issues:

1. Domain name for your site.
2. The Web space for your site.

Domain Name

Go to a domain name listing site to identify a domain name that has not been taken and is available to register. There are a number of these sites such as www.who.is

You can use these sites to research an available domain name.

Enter in your suggested domain name e.g. Alfiethedog.com

Try other TLD (Top Level Domains) of say:

.uk
.eu
.biz
.com
.org
.net
.edu

etc.

The list of TLDs is expanding all the time.

Considerations for TLD

- What type of site it is.
- What type of service it is (.co.uk for UK sites if it is specifically only a UK service.)

All domains are available worldwide but there is often a concept of localisation with some TLD. So if you are a plumber in Trowbridge, you would want a domain of .co.uk or .com.uk or similar rather than .org

Some TLDs are restricted e.g. .edu or .ac.uk

Recently there have been a release of many more TLDs e.g.

.wales
.london
.shop

At the moment these are often more expensive that then standard ones.

Domain names usually cost about £10 a year. If you don't pay each year, the name becomes inaccessible to you and your site stops working. The domain name is then made available after a few months to anyone who wants to register it.

Web Space and Web Servers

It may be preferable to purchase the domain name from the web hosting company at the same time as purchasing the web space. Hosting companies manage all the required configuration for the Domain Name to ensure that it is available over the internet.

Hosting companies often do a special offer of giving you a free domain name when you buy the web space although this may restrict you in staying with that hosting company.

Servers come in essentially two flavours:

- Linux (Unix type)
- Windows

There are other types but these two types account for over 80% of the world's web servers.

Linux is open source and there are a number of implementations of this but essentially we can consider them to be all 'Linux' based. As the software is open source, the web hosting is usually cheaper than Windows servers.

Linux servers will usually come with PHP scripting language and MySQL databases as standard and they will often have a large amount of disc space available. You will get a control panel to administer your web site and uploading of files will by with ftp.

Linux type servers may be a better choose if you intend to use WordPress / Joomla

ftp (file transfer protocol) is a method to connect to a remote server and is how you upload and download files to your web space. You will need a program on your computer to do this such as FileZilla or Dreamweaver.

Windows servers are obviously based on the Windows Server operating system and usually have dot net programming language. They sometimes, but not always, will have PHP scripting language and may have mySQL as well. However, Microsoft tends to support their own version of databases called SQL Server which can be an extra cost.

Windows servers will also have ftp access and you will need an ftp program for that.

There are many hosting companies so you will have quite a choice. You should look for Basic Web Hosting rather than business web hosting which is usually more expensive.

Things to consider when looking for web hosting

Size of disk space – This defines the number of files and images you can upload. In practice even the cheapest web sites will provide plenty of space.

Bandwidth – This will define the number of different users that can access your site at the same time. Most hosting companies now seem to have unlimited bandwidth.

Backups – Provides security for you files in case you delete something or the server fails. This may be available at an extra cost so if you have a cheap hosting package you may have to make sure your files are store on your local computer.

Email accounts - Most hosting accounts will allow you to use your domain name for email although there may be some limit on the size of the inbox. However, you can usually redirect the mail to another account.

Server side languages and databases – This would be a decision between using languages like PHP/MySQL and dot net/SQL server. Usually a PHP/MySQL based host would be preferable for smaller web sites.

After purchase of web space

You will get access to the web space you have purchased and will be given ftp host, username and password.

It may take a few hours for your site to appear worldwide as the name servers are updated around the world.

Notes on Page Set Up

This section looks at some of the page elements that you need to be aware of to provide good content to search engines.

Page details

Make sure that you use the page <title> tags
The page title tag is important and appears in the search results. There is no particular rule for number of words but it is generally accepted that it should be around 5 to 10 words. You can use keywords in the title but don't overdo it or otherwise it will be considered as spam content.

Make sure the page Meta description tag is used
meta description. Google can display the title meta tag in the search results sometimes. The description should be a summary of the page contents and is usually longer that the title, up to a paragraph in length.

<meta name="description" value="…….." >

Note that the keyword meta tag has no real use now a days.

Headings and other tags

- Use <h1> <h2> etc. tags in the correct order to define the page structure.
- Make sure that the page is as HTML compliant as possible.
- href link text should be keyword related but not excessive that it looks like spam.
- Images should have text in the alt tags.
- Try to create links to 'good' quality web sites - not spamy web sites.
- Make your web pages to load as fast as possible.
- Don't duplicate title tags on other pages.
- Don't duplicate content on different pages.
- Web page file name should be short and relevant to the content of the page
- Domain names do not have to be long to match the site content.

Have a site map linked from your main home page which should make sure that Google indexes the page when you add new pages. You can also request Google to re-index although you have no real control on that.

Site Navigation

Make sure that there is good hierarchy and structure to your site navigation:

- At the top level is your home page. This will have the main navigation menu and will be the most popular of all pages, and hence has to be optimized for the most

general and competitive phrases.

- The second level comes from the main navigation menu will be your top-level products or service pages which are often optimised for competitive keywords in their category.
- Any third level will be the more specific product or service pages and can be optimized for the names of the products or services.
- Any fourth level will be extra content such as blog posts, articles, videos, podcasts, etc. They help establish the quality of your web site providing that your articles are original and relevant to your web site of course.

Page Content

Google wants sites to have 'good' content. The situation that Google wants to achieve is that the search results should return sites that are relevant, good quality and authoritative. If you create a web site that no one has an interest in like bellybuttontoday.com then you won't get many visits to that site no not matter what optimisation you do.

There used to be all kinds of 'tricks' that you could do which where "black hat" or "grey hat" methods but Google has come down on those now since the Penguin update a few years back e.g. putting hidden text for text that is not relevant to the page. Google are constantly modifying the system so that pages which are deliberately using spammy methods are penalised.

Google Webmaster accounts

The Google Webmaster is a must for all web developers. It allows you look at stats, 404 page not found errors, duplicate titles and so on.

The robots.txt file

robots.txt file is used to tell a search engine specific things, usually which files are not to be indexed and will look similar to the following:

An example robots.txt file is:

Sitemap: http://www.wiltshire.ac.uk/sitemap.xml

User-agent: *
Disallow: /dbhelp/
Disallow: /download/
Disallow: /error/
Disallow: /include/
Disallow: /induction/

The file is located in the root of your web server.

Google adverts

Google, and other search engines allow you to create adverts to advertise products on your web site.

Your advert appears on Google search results when someone uses particular search words. When someone clicks on the advert you then are charged for that click through. The amount you are charged depends on your bid and other competitors bids for search keywords.

So you have to judge if running a campaign will result in improved profits. There is no point in having a click through price that ends up in you spending more than the cost of the product that you are trying to sell.

For example:

Say each click is averaging £0.20 per click.

And you need 200 clicks to get one purchase.

So each purchase costs you £40.

Search Engine Optimisation (SEO)

There are essentially two types of search systems:

1 Indexed types such as Google and Bing.
2 Human directory powered types.

SEO is the process where you make modifications to your web pages so that Google and other search engines will return your page as far up the top of the search results as possible. As Google is the most wildly used search engine, many of the concepts are associated with Google.

SEO also involves having links from good quality web sites pointing back to your web pages that are relevant to your content.

We can split SEO into two processes:

On-Page Techniques
There is where you enter your keywords in the text of the web page.

Off-Page Techniques
This is where you get relevant links to your site from other web sites.

SEO site tasks

[1] If you have an idea for a web site, search the domain web sites and see if you can identify a suitable domain name for your project.

[2] Using the Google tools and other tools, decide on a set of keywords and phrases which you think people will search on to get your web site.

[3] Modify your web page titles to improve their keyword placement but don't overdo the keywords or it may appear spammy.

[4] Check that all pages have a META description and modify them as appropriate.

[5] Add in some internal links - link from one page to another on your site to ensure pages can be found. Check that your navigation hierarchy.

[6] Look at your web page file names and modify them to reflect the content.

[5] Check that you have alt tags for your images.

[6] Check your Webmaster Tools account.

[7] Use link anchors within the web pages to go straight to specific locations of the web site.

[8] Change your content regularly using articles and blogs.

[9] Add in <h1> headlines.

[10] Review your results by looking at some form of page statistics.

[11] Set up social media such as Twitter, Facebook and Google+ as minimum and put links on your web site for visitor to easily get to.

[12] Regularly do updates to your social media and try to engage communications with your visitors.

Google Analytics

Google analytics http://www.google.co.uk/analytics/ is a really good resource for monitoring your traffic. You have to put JavaScript code onto each page of your web site and you need a Google Account of course.

Google Analytics does change occasionally, they add in new features and move things around so it can be difficult write a guide. However, there are some functions that you may want to do.

To create a new View (Profile)

One of the issues with Google Analytics is the terminology that it uses, and it can be difficult to know what you have to do even though you know what it is you want.

The first thing you need to do when you create your account is to create a "View" of your data. A view is going to be a set of data that is displayed in graphs and tables. Normally you create a View which displays all the site data, but you can create many views, each one being a filter of all the data. So you could create a view of just one sub folder.

[1] To create a view, log into your Google account and click on **Admin**.

[2] You will be presented with a drop down list of all your accounts.

There are three parts to an account. There is the **Account** itself and then there is the **Property** and then the **View** (or Profile).

The Account defines the overall account, the Property defines the website and gets the tacking code, and there is the View (or Profile) and with any filters.

[3] Select the account that you want and then in the list of views, select "Create new view".

[4] If you accept the details as they are, the view will display all data. However, you can apply filters to this view by clicking on "Filters".

Now if you go back to the Google Analytics Home screen it will list that new view which you can click on and look at the data.

Filter data by subfolder

One feature that you may want to work with is the ability to see traffic that is going to a particular sub folder. This is fairly easy to do on an ad-hoc basis. So say you have a folder called /phpregister/

Log in to your Google account and go to http://www.google.co.uk/analytics/

Go to the site that you want to look at and select:

Behaviour -> Content -> Site Content -> All Pages

In the search box put in /phpregister/ for a subfolder name.

This displays the data for that subfolder assuming that there is a subfolder on your web site called phpregister of course.

To see Google search engine words

Behaviour -> Site Search -> Search Terms

To display real time

Click on “Real Time” and then Overview.

To display % of use showing which links are being used

Behaviour -> In-page analytics

Tools for Analysing and Optimising Pages

These tools are all free but you will need to create a login for most to fully work.

[1] Google Analytics http://www.google.co.uk/analytics/
Most important tool of all – it is used to monitor your site traffic using graphs and lists and you can use it to measure the success of particular parts of your web site. Requires a Google account and needs JavaScript code to be added to very web page.

[2] Google Webmaster Tools https://www.google.com/webmasters/
Another important tool which offers a load of useful information. It enables you to diagnose site problems such as page not found, and monitor Google search indexing for your site. Requires a Google account and usually used with Google Analytics.

[3] xml site map generator tool http://www.web-site-map.com/xml_sitemap.php
This is a free site map generator tool which can create an XML site map for sites of up to 3500 page. A xml site map for you web site is useful for search engines to find new content, although it may not provide a great improvement in indexing.

[4] Broken link checker http://www.brokenlinkcheck.com/link-checker.php
This is a free broken link checker which is capable of checking web sites with up to 3000 pages.

[5] Google Keyword Planner Tool https://adwords.google.co.uk/KeywordPlanner
This tool needs a Google account to use properly, but you can use it without a Google account. It enables you to research keywords by entering them with reference to particular categories.

It is really intended as a tool to be used to with the Google AdWords system but is very useful as a way to get keyword phrase ideas and see how popular they are.

[6] Collection of tools http://www.searchenginegenie.com/seo-tools.htm
This is a great collection of different tools available for different search engines.

[7] Screaming Frog http://www.screamingfrog.co.uk/seo-spider/
This is a program that you download onto your computer. The report shows you Title tags, URLs, Meta Descriptions, Canonical tags, etc. plus, it tells you about pages that may have 404 errors, redirects, and other things. Requires you to download a program and install it on your local computer. The free version is limits the number of web pages it can spider.

[8] Rex Swain's HTTP Header Viewer http://www.rexswain.com/httpview.html
This shows what data is sent back from a web server for a particular page. It identifies the headers and is good for seeing what sort of redirect any page may have. For SEO purposes we want to see 301 redirects rather than 302s or any other kind. It will also show if there are multiple redirects for any URL. Requires no login account.

[9] Google Keywords Tool https://adwords.google.com/
Use this for keyword research. Use it to learn about the types of phrases your target audiences use at Google when they're looking for what you offer on your website.

Requires a Google account.

[10] NoFollow Available on the Google Chrome web store - search for "NoFollow"
This outlines all links on a page that have the Nofollow attribute on them and will pop up a window if the page you're looking at has the Noindex tag on it. This is useful as you may not have meant the page to have a Noindex tag which will stop search engines indexing it. Installs into Google Chrome as an add in.

[11] SEO for Chrome Available on the Google Chrome web store search for "SEO"
Another SEO plug for Google Chrome which displays various information about a page.

[12] Awesome Screenshot (combined with Evernote) Available on the Google Chrome web store - search for "Awesome Screenshot"
Use to quickly mark up a web page - arrows and words, draw circles around items, etc. Requires Google Chrome to use this as it has to be installed as an add in.

[13] Web Developer ToolBar http://chrispederick.com/work/web-developer/
This is an extension which is available for different browsers and when installed provides many tools for web development.

[14] Google Chrome Developer Tools (DevTools)
This is built into Google Chrome and you can access it by displaying a web page, right click on the page, and then select 'Inspect Element'.

It is very extensive and is fully described at

https://developers.google.com/chrome-developer-tools/

It includes tools for inspecting CSS and page elements, network traffic, debugging JavaScript, emulating different sized browsers and so on which are accessed by pressing F12

[15] Other Chrome Extensions
https://chrome.google.com/webstore/category/extensions

[16] The moz tool bar http:// www.moz.com
This web site has a number of free tools one of which is a tool bar for Google Chrome. It is a really good tool bar that displays Page Authority and Domain Authority.

[17] The webcoe web site https://www.webceo.com/
You need to sign up for this site but has some very good utilities that look at comparing your site to other similar ones and send some nice graphs at regular intervals.

Other facilities for testing web pages

Most web browsers are equipped with in built facilities to help web development.

Internet explorer, Google Chrome and Firefox, Press F12 key will take you to a set of tools useful for web page development.

Google Chrome is particularly good for this and has many features which gives advice on how to improve your web site.

The site:

https://developers.google.com/speed/pagespeed/

provides information which matches the descriptions when you do a “Page Speed” test.

Also note the web site:

https://varvy.com/

is a good resource for modern SEO concepts which are linked closely with Google principles of good web site design.

There is also a pdf:

http://static.googleusercontent.com/media/www.google.co.uk/en/uk/webmasters/docs/searc
engine-optimization-starter-guide.pdf

which is a Google document describing their ideas of web design.

Structured data mark-up

Structured data mark-up is a way that a web designer can add information to a web page that provides more information to a search engine.

So for example, you could have an h1 tag for you heading which may be:

<h1>eSeller</h1>

However, this does not give any information to a search engine as to what eSeller is. It may be referring to a company name or it might be a software product. Structure data can classify and add information which can then be understood by the search engines.

There are a few different ways to implement mark-up data. The following two methods are the most common:

1. Microdata – this is HTML 5 which uses HTML tags and attributes
2. JSON-LD – uses JavaScript

Google prefers Microdata.

An introduction to Microdata

Microdata consists of 3 elements: **itemscope**, **itemtype** and **itemprop**.

Use **itemscope** to identify that the content item as follows:

```
<div itemscope>
PHP Tutorials eBook
</div>
```

The **itemtype** is used as an attribute to define the type.

```
<div itemscope itemtype="http://schema.org/Book">
PHP Tutorials Book
</div>
```

http://schema.org/Book indicates that this item is a book

We can then use the **itemprop** to add properties such as name.

```
<div itemscope itemtype="http://schema.org/Book">
<div itemprop="name"><strong>PHP Tutorials: Programming with MySQL and PHP</strong></div>
</div>
```

A complete list of the entire hierarchy schema is available at

http://schema.org/docs/full.html

A more complete example would look like this:

```
<div itemscope itemtype="http://schema.org/Book">
<a itemprop="url" href="http://www.paulvgibbs.com/#ebooks"><div itemprop="name"><strong>PHP Tutorials: Programming with MySQL and PHP</strong></div>
</a>
<div itemprop="description">This book was written from a set of courses I teach in a Further Education college. It contains of a number of code samples and examples which you can download from this site.

It provides an introduction to web programming, how to display data from a database and update data to the database. It explains issues that you encounter in real world situations and provides the basic code from which you can then use to further develop.</div>
<div itemprop="author" itemscope itemtype="http://schema.org/Person">
Written by: <span itemprop="name">Paul Gibbs</span></div>
<div>Edition: <span itemprop="bookEdition">V2</span></div>
<div>Available in <link itemprop="bookFormat" href="http://schema.org/Ebook">Ebook <link itemprop="bookFormat" href="http://schema.org/Paperback">Paperback </div>
</div>
```

Tool for creating Microdata

Adding the structure data to a web page is not straight forward and it needs to be done manually to each web page. You do not have to update all you web pages and it does not in itself improve your search engine rankings but it does give precise information to the search engine and so should display more relevant content to the user.

The above is a very simple Schema and in practice they do become very complicated so the best way is to use an application. On such application is at:

http://schema-creator.org/

Google webmasters also has a helper program at:

https://www.google.com/webmasters/markup-helper/

Testing your Microdata

You can test and validate your Microdata here:

https://developers.google.com/structured-data/testing-tool/

Summary

This chapter describes the requirements for a good web page so that it meets the requirements of modern search engines. There is no point in having a web site if the site cannot be found. SEO (Search Engine Optimisation) is a very extensive concept but Google does give basic guidelines for creating pages that meet their requirements.

13
Forms

What's in this chapter:

* Form structure and elements.
* Example HTML Form.
* Validation and formatting.

Forms are used on the web to allow users to input data and then have that data processed in some way. For example, a form could be for collecting comments from a user.

The processing can only be done using a script which runs on the server, which might be a PHP script that stores the data in a database, or it might send an email to someone.

We are not covering PHP in this book so we can't really do a great deal with the data. However, we can create an email form that sends the data by email and show the principle of forms, and we can look at validation and layouts.

Basic Form Structure

The basic structure of a form is:

```
<form action="…." method="…">

   FORM ELEMENTS HERE

</form>
```

Action is where the data is to be sent to. Usually this is a URL of a PHP script.

Method is the either **post** or **get** which defines how the data is to be sent to the URL specified in Action.

With **get** the form data is appended as name / value pairs in the URL string and will look something like the following:

http://www.servername.com/processpage.php**?person=fred&age=35**

In the above example:

Name	**Value**
person	fred
age	35

The **get** method that uses appended name / value pairs is limited to around 3000 characters and also appears in the web browser address bar which means the data is not really secure.

On the other hand the **post** method embeds the data in the http request for the page and hence the data is not displayed on the address bar. Also, there is no limitation for the data length.

So a form consists of the <form> … </form> elements and then a number of input data boxes, checkboxes, radio buttons, dropdown lists and a submit button. The person enter in their data into the form, clicks on the submit button which then sends the data to a script on the server where it is actioned.

Dreamweaver and Forms

When working with Dreamweaver we can insert Form elements using the menu system. If we select:

Insert -> Form

then we can select the element from the list.

Form elements

Form elements consist of a form element type which defines the type of the input field. This will be “text”, “checkbox” and so on. Then there will be name which defines the name of the data field. Optionally there will be a value which is the default value of the input box.

The main form elements are as follows:

[1] Text

```
<input type=“text” name=“username” />
```

[2] Password

```
<input type=“password” name=“userpassword” />
```

[3] Hidden

```
<input type=“hidden” name=“recid” />
```

[4] Textarea

```
<textarea name=“comments” cols=“20” rows=“3”></textarea>
```

[5] Checkbox

```
<input type=“checkbox” name=“employed” value=“1” />
```

Check boxes are not dependent on other checkboxes so if there are a number of checkboxes you can check as many as you want.

[6] Radio Button

```
Are you in employment
<input name="employed" type="radio" value="Yes" />
<input name="employed" type="radio" value="No" />
```

Radio buttons are dependent on each other which is different to a check box. In the above example you can only select Yes or No and not both. Note that the name is the same in each radio button to show that they are grouped.

[7] Dropdown List

```
<select name="types" size="3">
    <option name="fulltime">Full time</option>
     <option name="parttime">Part time</option>
  <option name="he">HE</option>
</select>
```

[8] Email input box (HTML 5)

```
<input name="email" type="email">
```

This is an HTML 5 input box which validates the email address that has been entered. There are a number of other HTML 5 specific input boxes and we will look at a few more further on.

[9] Submit Button

```
<input type="submit" name="Submit" value="submit" />
```

[10] Reset Button

```
<input type="reset" name="reset" value="Reset" />
```

[11] Image Button

```
<input type=“image” src=“image.jpg” width=“100” height=“25” />
```

Form Controls

[1] To illustrate the form controls, create a new web page called **form.html**

[2] In the body of the page enter:

```
<form action="" method="post">


</form>
```

[3] Inside the <form> … </form> add in a radio button:

```
<p>Gender</p>
<p>
<input name="gender" type="radio" value="Male" />
<input name="gender" type="radio" value="Female" />
</p>
```

Note that with radio buttons you can only select one item and this is controlled by using the same name. In the above example the name it uses is "gender".

[4] Display the page in the browser to see the effect of a radio button.

[5] Add in a drop down list:

```
<p>Age</p>
<p>
<select name="age">
        <option value="0">Make a selection</option>
        <option value ="1">Under 18</option>
        <option value ="2">18 to 25</option>
       <option value ="3">26 Upwards</option>
  </select>
</p>
```

[6] Display the page in the browser to see the effect of the drop down list.

[7] Add in a submit button.

```
<input type="submit" name ="submit" value="submit" />
```

Code file: 13_Forms/files/form.html

You might want to add on any other form elements and see how they work.

Example HTML email form

Create a new HTML 5 web page called **mail.html**

The purpose of this form is to post data to a PHP page that has already been created on a remote server. The user goes to the **mail.html** form, enters various details and when they press the submit button, the data is posted to the PHP page. The PHP page then processes the data and in this case sends an email to you.

Within the <body> of the web page create a form section on the page similar to the following:

```
<form action="…." method="post">


</form>
```

Note that the action is the url where the PHP script is located. I have include a PHP script that can be used in this example but you will need a web server that has PHP and which sends emails.

[1] Add in a text box for the name, use a name of **name**

```
<form action="…." method="post">
Name:<br>
<input type="text" name="name" value="your name"><br>
</form>
```

[2] Add in a text box for the email address, use a name of **toaddress**

```
<form action="…." method="post">
Name:<br>
<input type="text" name="name" value="your name"><br>
To address:<br>
<input type="text" name="toaddress" value="your email"><br>
</form>
```

[3] Add in a textarea for the comments, use a name of **comment:**

```
<form action="…." method="post">
Name:<br>
<input type="text" name="name" value="your name"><br>
To address:<br>
<input type="text" name="toaddress" value="your email"><br>
Comment:<br>
<input type="text" name="comment" value="your comment" size="50"> <br><br>
</form>
```

[4] Add in a submit button, use a value of **Send** and a name of **submit**

```
<form action="…." method="post">
Name:<br>
<input type="text" name="name" value="your name"><br>
To address:<br>
<input type="text" name="toaddress" value="your email"><br>
Comment:<br>
<input type="text" name="comment" value="your comment" size="50"><br><br>
<input type="submit" value="Send" name="submit">
</form>
```

[5] Add in the **Reset** button

```
<form action="…." method="post">
Name:<br>
<input type="text" name="name" value="your name"><br>
To address:<br>
<input type="text" name="toaddress" value="your email"><br>
Comment:<br>
<input type="text" name="comment" value="your comment" size="50"><br><br>
<input type="submit" value="Send" name="submit">
<input type="reset" value="Reset">
</form>
```

[6] Add a hidden field with a name of **fromaddress**

```
<form action="…." method="post">
Name:<br>
<input type="text" name="name" value="your name"><br>
To address:<br>
<input type="text" name="toaddress" value="your email"><br>
Comment:<br>
<input type="text" name="comment" value="your comment" size="50"><br><br>
<input type="submit" value="Send" name="submit">
<input type="reset" value="Reset">
<input type="hidden" name="fromaddress" value="paulvgibbs@yahoo.co.uk">
<br>
</form>
```

Your page should look similar to the following:

```
<!DOCTYPE html>
<html>
<body>

<h3>Send e-mail to someone@example.com:</h3>

<form action="…." method="post">
Name:<br>
<input type="text" name="name" value="your name"><br>
To address:<br>
<input type="text" name="toaddress" value="your email"><br>
Comment:<br>
<input type="text" name="comment" value="your comment" size="50"><br><br>
<input type="submit" value="Send" name="submit">
<input type="reset" value="Reset">
<input type="hidden" name="fromaddress" value="youremailaddress@sever.com"><br>
</form>

</body>
</html>
```

Note that the hidden field has been added to define the “from” address. CHANGE THIS ENTRY TO A SUITABLE FROM EMAIL ADDRESS.

Display the page in your browser, fill in the form and submit the details.

The following php file is a simple email processing script that can be used with this example. You will need to upload it to a web server to allow an email to be sent.

```
<?php

//————————
if ( isset($_POST[“name”]) )
{
    $name = $_POST[“name”];
}
else
{
    echo(“Fill in your name”);
    exit();
}

//————————
if ( isset($_POST[“toaddress”]) )
{
    $toaddress = $_POST[“toaddress”];
}
else
{
    echo(“Fill in the to address”);
    exit();
}

//————————
if ( isset($_POST[“fromaddress”]) )
{
    $fromaddress = $_POST[“fromaddress”];
}
else
{
```

```
    echo(“There is no from email address”);
    exit();
}

//————————
if ( isset($_POST[“comment”]) )
{
    $comment = $_POST[“comment”];
}
else
{
    echo(“Enter your comment”);
    exit();
}

//————————
if ( isset($_POST[“submit”]) )
{
    $submit = $_POST[“submit”];
    if ($submit == “Send”)
    {
    //Send email
    $result = “Your name: ” . $name . “  Your comments: “. $comment;
    $headers .= “From:” . $fromaddress . “\r\n”;
    //$headers = “From: paulvgibbs@yahoo.co.uk\r\n”;
    mail( $toaddress, “Comments”, $result, $headers );
    echo(“TO ADDRESS : $toaddress<br/>”);
    echo(“FROM ADDRESS : $fromaddress<br/>”);
    }
}

?>
```

Code file: 13_Forms/files/processform.php

Issues with email forms

All input boxes must be validated to check that sensible data is being submitted. This is often done with JavaScript on the client code, and PHP on the server. It is important to have server side validation as well as client side validation.

Forms may need a CAPTCHA script added to make sure that the form is being submitted by a real person. A CAPTCHA script is often a display with a text box where you enter in some characters to prove that you are a human rather than some sort of automated script.

Most forms need server side code for them to be of any value. You can create forms which just do calculations in JavaScript. However, you may want data to be stored in a database or you may want to send an email and that can only be done with coding on the server using PHP or some other programming language such as dot net.

Client side form validation is an extensive subject but there are many systems around now such as JQuery that make form validation easier for a Web Developer to work with.

Forms in HTML 5

HTML 5 has added a number of new form fields. However, the support of the new fields is quite patchy on different browsers, and so it may be a few years before we are able to use them to any extant.

We have shown that standard HTML has form fields for text, radio buttons, checkboxes, drop down lists, password and hidden fields.

HTML 5 has extended these to include the following:

- email - entry of an email address.
- url – entry of a URL.
- number – number entry with range.
- tel – telephone number
- date – calendar date picker

and so on.

Using these input types makes form validation much easier.

On devices such as smart phones and tablets, these fields are displayed to allow easier input for the small screen.

The above input types are used as follows:

<input type=”date” name=”date” />

which would be how you use the date input field.

HTML 5 also adds in some validation methods built into the text box themselves.

HTML 5 and form validation

HTML 5 includes a number of extra attributes that can be used to validate form entry on the client bowser. Not all browses support all the features and also, validation should also be done on the server as well using a programming language like PHP or dot net.

TASK 1 - The required attribute

Create an HTML 5 form called **form01.html** with the following fields and values.

The following has an input box of type “text”, an input box of type “email” and an input type box of type “date”.

The email type and date type are two of the new HTML 5 input types.

```
<!doctype html>
<html>
<head>
<meta charset=“utf-8”>
<title>Untitled Document</title>
</head>

<body>

<form method=“post” action=””>

<fieldset>
<legend>Contact Details</legend>
<p>
<label for=“Name”>Name:</label>
<input type=“text” name=“Name” required />
</p>

<p>
<label for=“Email”>Email:</label>
<input type=“email” name=“Email” required />
</p>

<p>
<label for=“Start Date”>Start Date:</label>
<input type=“date” name=“startdate” required />
</p>

<p>
```

```
<input type="submit" name="Submit" />
</p>

</fieldset>

</form>

</body>
</html>
```

Code file: 13_Forms/form_required.html

When this is run in the browser, and you press submit, notice the affect. There will be differences in the behaviour with different browsers.

TASK 2 - Input type of "number"

The HTML 5 number input type has max, min and value entries which can be used to validate.

Add the following onto the form and see what happens:

```
<p>
<label for="age">Your Age</label>
<input type="number" name="age" min="18" max="99" value="21">
</p>
```

Code file: 13_Forms/files/form_required_number.html

form_with_styles.html is the same form but with styles added into the document.

Code file: 13_Forms/files/form_with_styles.html

Formatting Forms in HTML 5

HTML 5 has the **fieldset, legend** and **label** elements specifically for laying out forms.

Labels are a way of improving usability to a web page. The label is often used to create the name for the <input> element similar to the following:

<label for="name">Your Name</label>
<input type="text" name="name" value="your name">

The label element does not in itself display anything special for the user.

The label can be bound to an element by using the "for" attribute, or by placing the element inside the <label> element.

The **fieldset** provides a way to break up a form into sections and **legend** is used to title a **fieldset**.

Create a new HTML 5 web page called **form_format.html** and in the body of the page either copy the following or create the elements by typing them in manually:

```
<form name="form01" method="post">
<fieldset>
    <legend>Departments</legend>
       <p>
    <label ID="lblModerator" class="title">Moderator</label>
    <input type="text" ID="txtModerator" />
    </p>

    <p>
    <label ID="lblDept_Title" class="title">Dept Title</label>
    <input type="text" ID="txtDept_Title" />
    </p>

    <p>
    <label ID="lblDept_Order" class="title">Dept Order</label>
    <input type="text" ID="txtDept_Order" />
    </p>

    <p>
    <input type="submit" ID="cmdSubmit" value="Create" class="submit" />
    </p>

</fieldset>

</form>
```

Add in the following styles to the top of the page:

```
<style type="text/css">
body {
    font-family:Arial, Helvetica, sans-serif;
}
fieldset {
    width:500px;
}
fieldset p {
    margin: 10px;
    padding-bottom:10px;
    width:460px;
    clear:both;
}
.title {
    float:left;
    width:100px;
    text-align:right;
    padding-right:10px;
}
.submit {
    text-align:left;
    margin: 10px;
}
legend {
    font-size:20px;
}
</style>
```

Test the web page out in different browsers and note the difference in displays. You may find that IE displays a little different to Chrome

Code file: 13_Forms/files/form_format.html

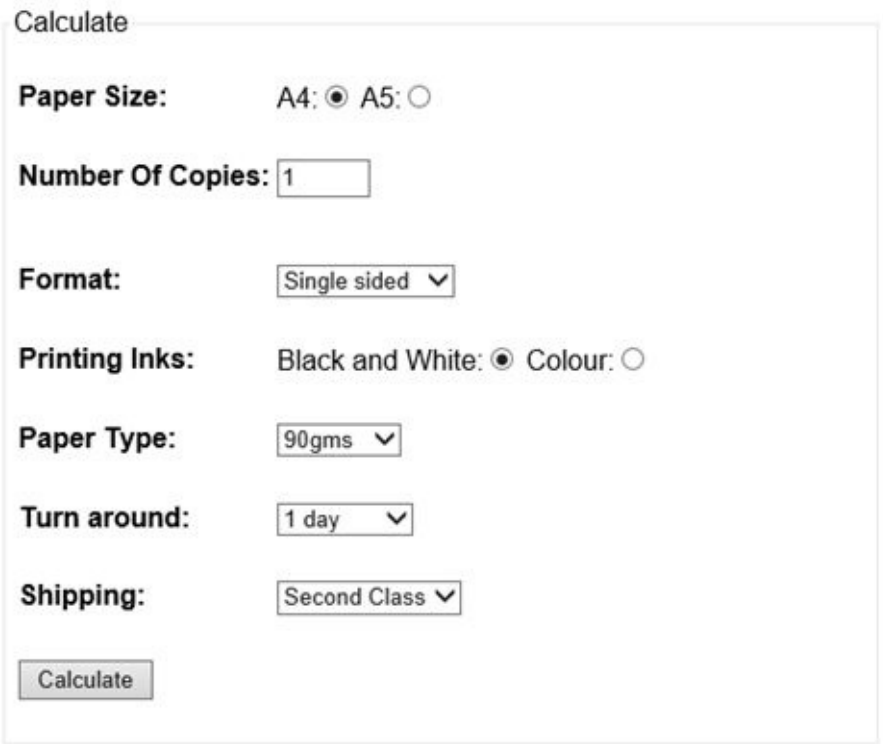
TASK 3 - Create a form

You have to create a web form that is to calculate the costs of small run printing. Your job is to create the front end side of the form which will post the data to a server and a web programmer will then program the code to do the calculations. You have to make the web form to look as good as possible with styles as appropriate.

The following are the fields that need to be created on the web form called **printcalc.html**:

Field	**Description**
Paper Size	A radio button of either A4 or A3
Number of copies	A number field which must be in the range 1 to 99
Format	A drop down list of either Single sided or Double sided
Printing Inks	A radio button of either color or black and white
Paper type	A drop down list of either 90gms, 110gms or 130gms
Turn around	A drop down list of either standard or 1 day
Shipping	A drop down list of either Second class or first class

This is what the final display will look like:



Code file: 13_Forms/files/printcalc.html

Summary

This chapter has looked at forms and form elements. The use of forms is very extensive and you see them on many web sites to collect information and submit data to the owner of the web site. The pitfall of forms is that the data has to be properly validated before it gets sent to the script that will process it and the data also has to be validated on the server as well. HTML 5 has introduced many new features that make validation much easier but the features are not supported on all browsers.

14
HTML 5 Design

What's in this chapter:

* The HTML 5 Layout.
* HTML 5 example.

HTML 5 has introduced extra mark-up tags to define the structure of a web page.

Up until now we have used DIVs to define the page structure; however, it can be difficult to understand what these DIVs refer to within the page layout unless you look at the class name.

So:

```
<div id="footer">

</div>
```

would be used as a footer.

HTML 5 has introduced a number of schematic Mark-up tags which expresses the meaning and purpose more clearly to the developer. These elements have names such as <header>, <footer> and so on. Some of the semantic HTML 5 elements are:

<header> ….. </header>
<footer> ….. </footer>
<section> ….. </section>
<article> ….. </article>
<aside> ….. </aside>
<nav> ….. </nav>
<figure>…. </figure>
<figcaption>….</figcaption>

You do not have to use these in an HTML 5 web page; you can still use DIVs for the structure as before if you want.

You still have to apply styles to these elements in the same way as you do with DIVs but

you would have something like:

```
footer
{
   padding:5px;
   text-align:center;
   font-weight:bold;
}
```

Headers <header> and footer <footer>
The <header> and <footer> elements are used at the top or bottom of each web page but can also be used as header and footer of individual <article> and <section> elements. For example, in a blog, a blog post could be an article element and each blog post could have its own header and footer element.

Navigation <nav>
The <nav> element is used for the major navigation elements of a web page.

Article <article>
The <article> element is used to define content that can be considered as standalone information such as a comment or a blog post or an item for sell.

Aside <aside>
The <aside> element can be placed inside an article element in which case it should have content that relates to the article but is not an essential part of the article. It can also be used outside of an article element, in which case it should contain content that is related to the entire web page.

Section <section>
The <section> element is a grouping of different sections of a web page, so there may be sections of news, products, comments, blogs and so on.

Heading groups <hgroup>
The <hgroup> element is used to group together headings <h1> to <h6> and its purpose is when you have a heading and then a subheading.

Figures <figure> and <figcaption>
The <figure> element is for content of images, videos and so on. The figure caption <figcaption> is a text description of the image.

**Section elements **
The element is used where there is no HTML 5 schematic mark-up that could be used in its place. So for example, you might have a DIV element that is a wrapper for the entire web content.

The htmlshiv code
For internet explorer browsers that are below IE 9, we need to add in a bit of JavaScript code so that they correctly recognise the HTML 5 schematic mark-up elements. We add

in the following code to the <head> of the web page:

```
<!—[if lt IE 9]>
<script src=“http://html5shiv.googlecode.com/svn/trunk/html5.js”></script>
<![endif]—>
```

Exercise in HTML 5 Layout

This is an example of creating an HTML 5 layout which incorporates a number of HTML 5 schematic elements

TASK 1 - Create a new HTML 5 web page

First create a new HTML 5 web page **index.html** and enter the title:

```
<title>Antique Tractors - The Olde Tractor Store</title>
```

TASK 2 - Add in basic structure

In the <body> of the web page add in the basic page structure:

```
<div class="wrapper">
    <header>
        <h1>The Olde Tractor Store</h1>
    </header>

    <section class="products">
        <article>
        </article>
        <article>
        </article>
    </section>

    <aside>
    </aside>

    <footer>
        &copy; The Olde Tractor Store
    </footer>
</div>
```

Code file: 14_HTML5_Design/files/index_01.html

This uses the HTML 5 mark up of header, footer, section, article, nav, aside, hgroup, figure and figcaption.

TASK 3 – Add the first styles

In the head of the document create the style start and end tag.

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

```
</style>
```

We have to make all the schematic mark-up styles as display: block. Block level elements are those that start on a new line and will be the basic building structure of the layout. We have to define these as block level because older browsers that do not recognise the HTML 5 elements will treat them as inline elements.

We then create the body style which contains a background image.

```
header, section, footer, aside, nav, article, figure, figcaption {display: block;}
body {
	color:#514F4F;
	background-color: #f9f8f6;
	background-image: url("images/lgren033.jpg");
	background-position: center;
	font-family:Arial, Verdana;
	line-height: 1.3em;
	margin: 0px;}
```

Code file: 14_HTML5_Design/files/index02.html

TASK 4 - The wrapper and header styles

The next styles add in the wrapper which defines the width of the content and adds in the header image.

```
.wrapper {
	width: 960px;
	margin: 10px auto 10px auto;
	border: 2px solid #000000;
	background-color: #ffffff;}
header {
	height: 160px;
	background-image: url("images/tractorheader.jpg");}
```

Code file: 14_HTML5_Design/files/index03.html

TASK 5 - The h1 tags

```
h1 {
```

```
    text-indent: -9999px;
    width: 960px;
    height: 130px;
    margin: 0px;}
```

In this example, the h1 tag has been set with a text-index of -9999px. This is so that the text itself is shifted off the screen to the left , but will still be available for text screen readers. The header image tractorheader.jpg displays the same text.

Code itle: 14_HTML5_Design/files/index04.html

TASK 6 - Add in all text

To make things easier for this exercise, we will copy in all the text into the body.

To make it easier, the following text contains all the body text for this exercise. Copy this text and overwrite all the text in the body.

So now we have a page which is complete except for the styles.

```
<div class="wrapper">
    <header>
    <h1>The Olde Tractor Store</h1>
    <nav>
    <ul>
    <li><a href="" class="current">HOME</a></li>
    <li><a href="">SPARES</a></li>
    <li><a href="">FAIRS</a></li>
    <li><a href="">ABOUT</a></li>
    <li><a href="">CONTACT</a></li>
                </ul>
    </nav>
    </header>
    <section class="products">
    <article>
    <figure>
    <img src="images/masseyferg_290.jpg" alt="Tractor" />
    <figcaption>1979 Massey Ferguson 575</figcaption>
    </figure>
    <hgroup>
    <h2>1979 Massey Ferguson 575</h2>
                    <h3>&pound;4,500 Plus VAT</h3>
    </hgroup>
    <p>1979 Massey Ferguson 2wd Tractor with muti-power, comes with MF loader (not fitted), done 5555 hours,
```

```
with PUH and 2 double spool valves, overall in resonable condition for its age.</p>
    </article>
    <article>
    <figure>
    <img src=“images/ford_290.jpg” alt=“Tractor” />
    <figcaption>Fordson Major 6 Cylinder, 90HP</figcaption>
    </figure>
    <hgroup>
    <h2>Fordson Major 6 Cylinder, 90HP</h2>
    <h3>&pound;2,650 Plus VAT</h3>
    </hgroup>
    <p>This has had a Ford 6 cylinder fitted and repainted a while ago. Starts, runs and drives well, very useful to run a grain blower, or other stationery P.T.O. application.</p>
    </article>
    </section>
    <aside>
    <section class=“spares-list”>
    <h2>Spares Needed</h2>
    <a href=””>Starter Motor - Perkins Engine</a>
    <a href=””>Radiator - David Brown</a>
    <a href=””>McCormick CX100 Filter Kit</a>
    <a href=””>Case IH Clutch Kit</a>
    </section>
    <section class=“contact-us”>
    <h2>Contact</h2>
    <p>The Olde Tractor Store<br />
    The Warehouse<br />
    Chippenham<br />
    Wiltshire SN15 2JP</p>
    </section>
    </aside>
    <footer>
    &copy; The Olde Tractor Store
    </footer>
</div>
```

Code file: 14_HTML5_Design/files/index05.html

TASK 7 - Navigation styles

The navigation styles follow the standard unordered list <ul> configured as a horizontal

list.

The styles for the navigation list should be added as follows:

```
nav, footer {
    clear: both;
    color: #ffffff;
    background-color:#42573C;
    height: 30px;}
nav ul {
    margin: 0px;
    padding: 5px 0px 5px 30px;}
nav li {
    display: inline;
    margin-right: 40px;}
nav li a {
    font-weight:bold;
    color: #ffffff;}
nav li a:hover, nav li a.current {
    color:#FB9294;}
```

Code file: 14_HTML5_Design/files/index06.html

TASK 8 - The section and article schematic mark-up tags

In this design, the section tag is intended to contain all the information about the products which in this case are tractors.

Each product is contained within an article.

```
<section class=“products”>
    <article>
    </article>
    <article>
    </article>
</section>
```

Add in the styles for the section and article elements:

```
section.products {
    float: left;
    width: 669px;
    border-right: 1px solid #eeeeee;}
article {
    clear: both;
    overflow: auto;
    width: 100%;}
```

Code file: 14_HTML5_Design/files/index07.html

TASK 9 - The figure and hgroup styles

Each article in this example contains a <figure> and <hgroup> element which look like the following:

```
<article>
    <figure>
        <figcaption></figcaption>
    </figure>
    <hgroup>
    </hgroup>
</article>
```

The styles for this can now be added:

```
hgroup {
    margin-top: 40px;}
figure {
    float: left;
    width: 290px;
    height: 220px;
    padding: 5px;
    margin: 20px;
    border: 1px solid #eeeeee;}
figcaption {
    font-size: 90%;
    text-align: left;}
```

Code file: 14_HTML5_Design/files/index08.html

TASK 10 - the aside styles

Add in the following styles for the aside element:

```
aside {
    width: 240px;
    float: left;
    padding: 0px 0px 0px 20px;}
aside section a {
    display: block;
    padding: 10px;
    border-bottom: 1px solid #eeeeee;}
aside section a:hover {
    color: #985d6a;
    background-color: #efefef;}
aside h2 {
    padding: 30px 0px 10px 0px;
    color: #de6581;}
```

Code file: 14_HTML5_Design/files/index09.html

TASK 11 - Add in the final group of styles to finish off the links, h tags and footer

```
a {
    color: #de6581;
    text-decoration: none;}
h1, h2, h3 {
    font-weight: normal;}
h2 {
    line-height:1.2em;
    margin: 10px 0px 5px 0px;
    padding: 0px;}
h3 {
    margin: 0px 0px 10px 0px;
    color: #de6581;}
```

```
footer {
    font-size: 80%;
    padding: 8px 0px 0px 20px;}
```

Code file: 14_HTML5_Design/files/index10.html

TASK 12 - Add in htmlshv JavaScript code

Finally, to ensure that browsers below Internet Explorer 9 work correctly we add in the following code into the head of the web page.

```
<!—[if lt IE 9]>
<script src=“http://html5shiv.googlecode.com/svn/trunk/html5.js”></script>
<![endif]—>
```

Summary

This Chapter has looked at how to work with HTML 5 layout elements. These provide a way to define the page layout using named elements which more clearly represent the structure of a page. HTML 5 is the standard that we should use for all our pages from now on and provide new elements and attributes particularly for multimedia and graphics.

15
Responsive Web Design (RWD)

What's in this chapter:

* Description of RWD.
* Media queries
* An example RWD framework.

Responsive web design (RWD) is the 'in thing' at the moment.

If we go back a few years we only had one way to display web pages which was using a VDU. These were based on the old style Television (TV) sets which used cathode ray tubes (CRT). These were physically very large and heavy.

With only one type of device to display our web pages on, it was quite acceptable to design web pages that fitted to a fixed width of say 960px.

Gradually new technology came in which used LCD and LED displays. These displays became very popular very quickly because they are very thin and light and they use much less power. It wasn't long before this new display technology was incorporated in mobile devices, phones and tablets which changed the way we interact with the internet.

So this introduced devices that display at different resolution and on different sizes.

So how do we cope with these different sized devices? There are really two ways to do this:

[1] Create multiple web sites at different resolution and size and have a script which redirects to different sites after detecting the device type.

[2] Create one site but different CSS depending on the screen size.

Clearly the second method is preferred as you are not duplicating the content which would become difficult to maintain.

The importance of mobile design

Google and other search engines have really taken on board the idea of creating web pages for mobile devices. In fact Google will prefer web sites that are mobile friendly and will rate them higher in the search results.

Google has a tool that in their webmasters pages that allow you to test if the site passes for mobile use https://www.google.com/webmasters/tools/mobile-friendly/

which is from the Google site https://developers.google.com/webmasters/mobile-sites/

Media Queries

So how do we develop a web site that will work in multiple devices?

The answer is really the @media query which is used in style sheets to select different styles depending on the device. The @media query can change styles based on the characteristics of the device, such as the display type, width height, orientation and resolution.

The @media query can be used to control output for printer friendly output as follows:

```
@media print {
  / * print styles */
}
```

In addition, and more importantly for response layouts, we can have specific styles based on the view port size. In the example below we can see that we can use different images based on the screen size.

```
@media screen {
    #firstimage {
    background-image: url(/graphics/firstimage980.jpg);
    height:100px;
    }
}

@media screen and (max-width: 980px) {
    #firstimage {
    background-image: url(/graphics/firstimage980.jpg);
    height:100px;
    max-width:980px;
    }
}

@media screen and (max-width: 650px) {
    #firstimage {
    background-image: url(/graphics/firstimage650.jpg);
    height:95px;
    max-width:650px;
    }
}
```

```
@media screen and (max-width: 480px) {
    #firstimage {
    background-image: url(/graphics/firstimage480.jpg);
    height:80px;
    max-width:480px;
    }
}

@media screen and (max-width: 320px) {
    #firstimage {
    background-image: url(/graphics/firstimage320.jpg);
    height:60px;
    max-width:320px;
    }
}

@media screen and (max-width: 240px) {
    #firstimage {
    background-image: url(/graphics/firstimage240.jpg);
    height:50px;
    max-width:240px;
    }
}
```

As an example to show how the @media query can change the styles for different viewports:

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Responsive example</title>

<style type="text/css">
  @media (min-width: 500px) and (max-width: 600px) {
    h1 {
    color: fuchsia;
    }
  }
</style>
```

```
</head>

<body>

<h1>This demonstrates the way that styles can change as the size of the page changes.</h1>

</body>
</html>
```

Code file: 15_Responsive_Design/files/responsive/responsive.html

So first display the page in your browser and then change the size of the browser. As you reduce the size down to 600px, the color of the <h1> tag will change. Then as you reduce the size further down to under 500px, the color of the </h1> tag changes back.

A Responsive Web Design Framework example

In this exercise we are going to create a web site using the Responsive Web Design Framework **HTML-KickStart** which can be download from:

http://www.99lime.com/

We will use this framework as it described as 'light weight' and is not as extensive as others. There are other similar frameworks which are available, in particular, **Twitter bootstrap** which is located at http://getbootstrap.com/

Another one is called http://foundation.zurb.com/

All of these frameworks are all open source and free to use.

Google also has a web template system at https://developers.google.com/web/starter-kit/?hl=en

NOTE: if you are using Internet Explorer, in the "Compatibility View Setting", make sure that "Display internet sites in Compatibility View" is unticked.

Microsoft created the compatibility view for Internet Explorer 8 to give web designers time to convert their pages over from the non-standard techniques used in IE 7 and IE 6, to a more web standard compliant version. It tells the browser to render the page as if it were IE version 7 rather than IE version 8.

We are going to create another version of the Archibald's web site which we did in an earlier chapter. You will find that this version will look better and it will be easier to maintain because it uses a grid system.

TASK 1 - Download the HTML- KickStart framework

Copy the **HTML-KickStart-master.zip** file onto your local computer drive, unzip it and then copy the web files to an appropriate location.

To make it a bit easier, I have located this zip file in the files folder with the downloads for the book.

Code file: 15_Responsive_Design/files/responsive/ HTML-KickStart-master.zip

TASK 2 - Copy the file blank.html and call it index.html

Make a copy the **blank.html** file and rename it **index.html**

Remove all the code between the **<body>** and the **</body>** tags as we are going to start again.

Code file: 15_Responsive_Design/files/responsive/index.html

TASK 3 - Page title

Change the page title **<title> …. </title>** to Archibald's Shop

Code file: 15_Responsive_Design/files/responsive/index_task3.html

TASK 4 - Add in the basic grid structures

All of these frameworks work on the principle of a grid which is based on up to 12 columns.

The following is entered inside the **<body>** … **</body>** tags.

The text for this is located in a file:

grid_structure.txt in the downloads so that you do not have to enter the text by hand.

```
<div class="grid">

  <div id="logo" class="col_12">
      Logo or text
  </div>

  <div id="header" class="col_12">
      Header menu
  </div>

  <div id="slideshow" class="col_12">
      Images
  </div>

  <div id="blocks" class="col_12">

    <div class="col_4 visible column">
     Column 1 text
  </div>

    <div class="col_4 visible column">
  Column 2 text
  </div>
```

```
    <div class="col_4 visible column">
        Column 3 text
    </div>

    </div>


    <div id="description" class="col_12">

         <div class="col_6">
            Description
        </div>


    <div class="col_6">
          Image
    </div>

     </div>


      <div id="openinghrs" class="col_12">

        <div class="col_4 visible column">
    </div>
        <div class="col_4 visible column">
    Some Text
    </div>
        <div class="col_4 visible column">
    </div>

    </div>

     <div id="refs" class="col_12">
        Footer
    </div>


</div>
```

Code file: 15_Responsive_Design/files/responsive/index_task4.html

TASK 5 - Add in text for logo for header menu

The logo text is:

<h1>Archibald’s Shop</h1>

The text for the header menu is the same as what we did for the original Archibald’s exercise:

Home | Opening Hours | Location | Prices | Contact Us

Code file: 15_Responsive_Design/files/responsive/index_task5.html

TASK 6 - Add in styles

In the header of the web page create the styles tag as follows:

```
<style type=“text/css”>

</style>
```

Code file: 15_Responsive_Design/files/responsive/index_task6.html

TASK 7 - Add in the header style

```
#header {
background-color:#000;
color:#FFF;
padding-top:10px;
padding-bottom:10px;
    padding-left: 20px;
    font-size: 150%;
}
```

Code file: 15_Responsive_Design/files/responsive/index_task7.html

TASK 8 - Add in link styles

```
#header a:link {
```

```
color:#fff;
text-decoration: none;
}
#header a:visited {
color:#fff;
text-decoration: none;
}

#header a:hover {
color: #ff0000;
}
```

Code file: 15_Responsive_Design/files/responsive/index_task8.html

TASK 9 - Add in Logo styles

```
#logo{
        background-color:#000;
    width:100%;
    margin-left:0;
    margin-bottom:0px;
}
#logo h1{
    font-family: 'Fredoka One', cursive;
    color:#FFFFFF;
    font-size:300%;
    padding-left:100px;
}
```

Code file: 15_Responsive_Design/files/responsive/index_task9.html

TASK 10 - Add in the slide show

For the slide show we use the images **image03.jpg**, **image03.jpg** and **image05.jpg**

Enter the slide show code into the slideshow div:

```
 <!— Slideshow —>
<ul class="slideshow">
<li><img src="image03.jpg"  /></li>
```

```
<li><img src="image04.jpg"  /></li>
<li><img src="image05.jpg"  /></li>
</ul>
```

Code file: 15_Responsive_Design/files/responsive/index_task10.html

TASK 11 - Add the text to the three column blocks

COLUMN 1:

<h1>Barbers</h1>

We are a well-established Ilfracombe Barbers, occupying the small corner shop on North Field Road opposite the well-known tourist attraction of the Tunnels Beaches.

COLUMN 2:

<h1>Alterations</h1>

Ask about Clothes Alterations and Repairs such as adjusting hems on trouser and dresses.

COLUMN 3:

<h1>Silk flowers</h1>

We also sell high quality handmade silk flower headbands which are ideal for Weddings and festivals.

Code file: 15_Responsive_Design/files/responsive/index_task11.html

TASK 12 - Add in some styles for the Three blocks

```
 #blocks h1 {
font-size: 170%;
color: red;
text-align: center;
}
```

Code file: 15_Responsive_Design/files/responsive/index_task12.html

TASK 13 - Add in text and image for the description block

<h1>Welcome to Archibald’s Barber in Ilfracombe Devon</h1>

You don’t need an appointment and we are open late on Thursday evening to accommodate busy schedules.

Try our professional hairdresser today.

Code file: 15_Responsive_Design/files/responsive/index_task13.html

TASK 14 – Add in the image which is part of the description DIV

Add in the Images from eStudy – these images should already be in the download zip file.

For the shop image we are going to use two images which will display depending on the size of the browser.

We have two images **shop02.jpg** and **shop03.jpg**.

So in the web page where the word **Image** is located, enter the following picture tag.

```
<picture>
 <!—[if IE 9]><video style=“display: none;”><![endif]—>
 <source srcset=“shop02.jpg” media=”(min-width: 1024px)”>
 <source srcset=“shop03.jpg” media=”(min-width: 480px)”>
 <!—[if IE 9]></video><![endif]—>
 <img srcset=“shop03.jpg” alt=“Shop”>
 </picture>
```

NOTE: The media tag is defining what size browser will be displayed.

NOTE: The picture tag is not available in IE and so when this is displayed in IE, no image will appear. To overcome this we use the picturefill.js script which enables support for the picture element and associated features in browsers that do not yet support them.

```
<script>
   //old ie script
   document.createElement( “picture” );
</script>
<script src=“js/picturefill.js” async=“true”></script>
```

Place the above script into the <head> of the web page pointing to the JavaScript in the js folder. This should make the <picture> element work in IE and in any other browser that does not support the picture element.

Code file: 15_Responsive_Design/files/responsive/index_task14.html

TASK 15 - Styles for description

```
#description h1 {
font-size: 170%;
color: red;
text-align: center;
}
#description p {
font-size: 110%;
}
```

Code file: 15_Responsive_Design/files/responsive/index_task15.html

TASK 16 – Add in text for Opening Hours

```
<h1>Come Visit Us</h1>
<p><strong>Opening Hours</strong></p>
<p>Monday Closed All Day<br/>
Tuesday 8.00 am to 5.30 pm<br/>
Wednesday 8.00 am to 5.30 pm<br/>
Thursday 11.00 am to 6.30 pm<br/>
Friday 11.00 am to 6.30 pm<br/>
Saturday Closed all day<br/>
Sunday Closed all day</p>
```

Code file: 15_Responsive_Design/files/responsive/index_task16.html

TASK 17 – Add in Styles for Opening Hours

```
#openinghrs h1 {
  font-size: 170%;
color: red;
text-align: center;
}
```

Code file: 15_Responsive_Design/files/responsive/index_task17.html

TASK 18 - Add in text for footer

© Your Name Here 2016

Code file: 15_Responsive_Design/files/responsive/index_task18.html

TASK 19 - Add in styles for refs at the bottom of the page

```
  #refs {
background-color: black;
color:white;
margin-top: 20px;
margin-bottom: 20px;
padding-bottom: 20px;
padding-top: 20px;
      text-align: center;
}
```

Code file: 15_Responsive_Design/files/responsive/index_task19.html

TASK 20 - Finish off the page

This is the basic site layout and you can finish it off by adding in a Google map to identify the location and you may want to add in other text.

Summary

This chapter has looked at Responsive Web Design concepts and a simple framework which can be used to base your web pages on. Web sites have had to become more sophisticated in the way they approach the problems of multiple devices and different screen sizes. There are now many well documented techniques for this and as HTML 5 becomes more established, we will probably see much more use of different media such as video as well as graphics.