

Creating standards-compliant web pages in XHTML

Practical workbook

Aims and Learning Objectives

The aim of this course is to enable you to create a simple but well designed website to XHTML standards.

When you have completed these exercises, you will be able to:

- create a simple but functional website to present information about yourself, department or other interest using essential XHTML elements
- apply fundamental good web design principles to your pages
- transfer your files from your local PC to a web server using the SSH Secure File Transfer program

Document information

Course files

This document and any associated practice files (if needed) are available on the web. To find these, go to www.bristol.ac.uk/is/learning/resources and in the **Keyword** box, type the document code given in brackets at the top of this page.

Related documentation

Other related documents are available from the web at:

<http://www.bristol.ac.uk/is/learning/resources>



This document is licensed under a Creative Commons Attribution-Noncommercial-Share Alike 2.0 UK: England & Wales Licence (<http://creativecommons.org/licences/by-nc-sa/2.0/uk/>). Its “original author” is the University of Bristol which should be acknowledged as such in any derivative work.

Introduction

This course aims to equip you with the basic skills needed to create your own website. You will learn how to design and produce basic web pages using the (X)HTML language and how to integrate them into a well-organised and user-friendly website. The course will also cover useful design tips and techniques to improve your site, as well as how to put your website on line.

Prerequisites

This document assumes that you are familiar with the use of a computer keyboard and mouse, Microsoft Windows based products and the use of a web browser such as Mozilla Firefox or Internet Explorer.

Contents

Introduction to the World Wide Web.....	1
Task 1 Customising HTML-KIT.....	4
Task 2 Creating a basic web page.....	7
Task 3 Structuring content	9
Task 4 Adding formatted lists	11
Task 5 Changing the look and feel of a website, and the implications of bold and italic text	13
Task 6 Creating hyperlinks.....	15
Task 7 Using images.....	19
Task 8 Using tables.....	22
Appendix A XHTML tags quick reference.....	24
Appendix B Using colours on the web	27
Appendix C Design and planning tips	29
Appendix D Glossary of terms	31
Appendix E Useful resources.....	32

Introduction to the World Wide Web

The World Wide Web (WWW) is part of the **Internet**, a network of interconnected computers, in other words the physical infrastructure used to transfer data (for example, emails, web documents etc.) between computers.

The WWW is a body of virtual information stored on **web servers**. A web server is a computer system that runs software to allow people to download content stored on it from their own computers, and where appropriate, upload content from local computer to server. The University has its own web servers connected to the Joint Academic NETwork (JANET). From home, you have to connect (you must be registered first) to the web server of an Internet Service Provider (ISP) to access the Internet.

Publishing information on the web

The HyperText Mark-up Language

(X)HTML (HyperText Mark-up Language) is a document layout and hyperlink specification **mark-up language** used to format text and information for the web; it is **NOT a programming language**. (X)HTML consists of **mark-up elements**. The syntax of a typical element is as follows:

```
<name attribute1="value" attribute2="value">text</name>
```

↑
↑
↑
↑

opening tag
attribute
value
closing tag

At its most basic form, an (X)HTML element usually consists of a **opening tag** (a name placed between angled brackets – <name>) and a corresponding **closing tag** (indicated by a forward slash before the tag name – </name>) wrapped around the content to be structured. Opening tags, may include one or more optional **attributes** carrying **values**, which modify the default behaviour and settings of the element. (X)HTML elements instruct **browsers** and other **user agents** (the generic term for software used to interpret webpages e.g. **screen readers**) on how to structure the content **semantically** (i.e. logically); for example:

- <h1>heading level 1</h1> - is a level 1 heading
- University of Bristol - is a link to the University of Bristol homepage

A few elements do not contain anything and are hence known as **empty elements**; they are simply instructions that either point to a resource (e.g. an image) or insert an object, for example:

- - inserts an image.
-
 - inserts a line break

Note the space and forward slash inserted before the final bracket to close the instruction within the one pair of brackets.

(X)HTML elements are the **building blocks** of the web. This means that (X)HTML is not going away. Since 1990, HTML standards as defined by the World Wide Web Consortium (W3C, see <http://www.w3.org>) have evolved considerably. Until recently, HTML 4.01 was the recommended standard, however it has been superseded by the eXtensible HyperText Mark-up Language (XHTML), which has now become the recommended standard. It can be **parsed** (interpreted) or as two languages simultaneously: HTML and XML; this helps ensure forward-compatibility of documents.

Making the transition from HTML to XHTML is very straightforward. Apart from some **deprecated** (superseded and to be avoided) elements, most HTML code is compliant with XHTML. In XHTML:

- **tags must be in lower case** whereas HTML accepts UPPER as well as lower case
- **attribute values must be enclosed in double quotes** (`alt="Joe Bloggs">` instead of `alt=Joe Bloggs>`)
- **elements must be correctly nested** (for example, `properly nested tags` instead of `incorrectly nested tags`)
- **empty elements must be closed** (for example, `<hr />` instead of `<hr>`, `
` instead of `
`)
- **all attributes must be given explicit values and not minimised** (for example, `<option selected="selected" />` instead of `<option selected>`). There are no examples of such attributes in this course.

(X)HTML files should be saved with the extension **html** rather than **htm** (e.g. `welcome.html`).

Basic XHTML document structure

The following shows how a basic webpage is structured:

```
1 <?xml version="1.0" encoding="iso-8859-1"?>
2 <!DOCTYPE HTML PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
3 <html xmlns="http://www.w3.org/1999/xhtml" lang="en">
4 <head>
5 <title>Joe's website - Home</title>
6 <meta http-equiv="Content-Type" content="text/html;
  charset=iso-8859-1" />
7 <meta name="description" content="Joe Bloggs's homepage" />
8 <meta name="keywords" content="keyword 1, keyword 2" />
9 </head>
10 <body><!-- content starts here -->
11 <h1>Joe Bloggs's homepage</h1>
12 <p>Welcome to my world!</p>
13 <p>Enjoy your visit</p>
14 <p>Updated by <a href="mailto:jbloggs@hotmail.com">Joe
  Bloggs</a></p>
15 </body>
16 </html>
```

Lines 1 and 2 tell the browser what versions of XML and XHTML your pages are written in.

The XML declaration (line 1) is optional, but the XHTML declaration (line 2) is compulsory for your pages to validate against the standards as set by the World Wide Web Consortium (W3C). You will see many web page that do not include a `<!DOCTYPE>` and still work, however it **MUST**

be included at the top of each page to ensure your pages will function properly in all browsers and ensure code validates.

The whole document is contained between the opening `<html>` tag (line 3) and the closing `</html>` tag (line 16).

An (X)HTML document has **two main parts**: the `head` and the `body`. The `head` (enclosed between the opening `<head>` and closing `</head>` tags – lines 4 to 9) contains information **about** the file, including the `title` (between the opening `<title>` and closing `</title>` tags – line 5) of the page – displayed in the bar at the top of the web browser. Other information useful to search engines can also be added to the head. The `body` of the page (set by the `<body>` and `</body>` tags – lines 10 to 15) defines the actual content of the document.

Note Be consistent in naming the title of each page. For example, if your site is called **Joe's website** then it would be a good idea to give your CV page the title **Joe's website – CV**.

The line `<!-- content starts here -->` (line 10) is a **comment** and is ignored by browsers. Comments act as **signposts** to help you (or whoever might contribute to your site) make sense of your code, or to give instructions. They begin with `<!--` and end with `-->`.

The `<html>` `</html>`, `<head>` `</head>`, and `<body>` `</body>` tags are **the only required structural elements** of an (X)HTML document. As we have already said, the **content** (text, images, links, etc) of the document **is enclosed between the `<body>` and `</body>` tags**. Other (X)HTML tags are included within the body to structure the content logically, and pull in other resources (images for example).

In our example, the `<h1>` ... `</h1>` tags (line 11) format the text **Joe Bloggs's Homepage** as a level 1 heading (there are 6 levels) - that is by default, in most visual browsers, a bold, larger font size with spacing before and after. For a document to be logically structured, it is important that headings are used in the correct order; for example, an `<h3>` should only be used as a subheading of an `<h2>`.

The `<p>` and `</p>` tags (lines 12 to 14) form paragraphs; a visual browser will insert space before and after the paragraph, to set it apart from other elements. Paragraphs, headings etc. are examples of **block-level elements**, used to structure distinct sections of a document.

Finally, `Joe Bloggs` (line 14) creates a link to an email address. This is an example of an **inline element**, used to format unstructured content within a block (e.g. a phrase within a paragraph).

The emphasis should always be on the document being well-formed, as opposed to what it looks like. Other **user agents** will make use of the code as appropriate: a screen-reader, for example, can interpret the document structure for a blind or partially-sighted user. The **look and feel** of the page is dealt with elsewhere.

Here is how the code above would be rendered in a visual browser:



Joe Bloggs's homepage

Welcome to my world!

Enjoy your visit

Updated by [Joe Bloggs](#)

Task 1 Customising HTML-KIT

Objectives To open and customise HTML-Kit to produce XHTML compliant documents.

Comments In accordance with University Web guidelines, it is important to code pages to XHTML 1.0 standards.

1.1 Open HTML-Kit and close additional panels.

- Go to **Start / Programs / HTML-Kit** and select the launch icon  HTML-Kit

You should see the HTML-Kit start screen with **Open File wizard** embedded within it.

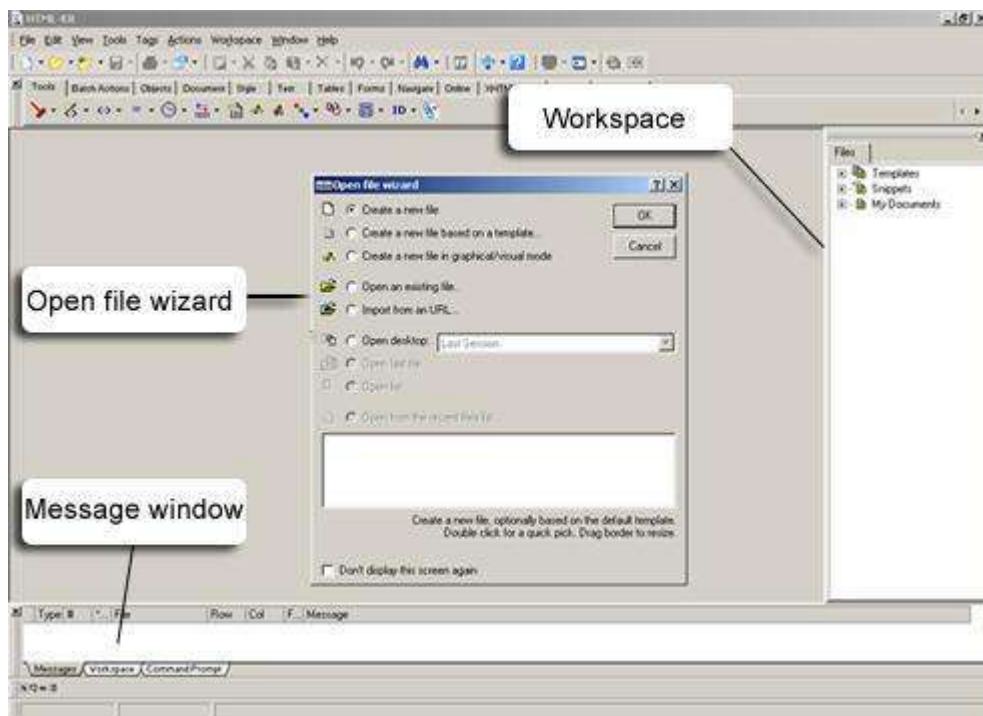


Figure 1 - HTML-Kit default interface showing Open file wizard

- Working in the Open file wizard (see Figure 1), select **Cancel** to close the box.
- Close the Workspace and Message window using the close buttons (see Figure 2 and Figure 3).



Figure 2 - closing the workspace



Figure 3 - closing the message window

1.2 Set the default template to XHTML:

- Select the Edit / Preferences, and select the tab labelled **Startup**.
- Working in the box shown in Figure 4, make sure the checkbox shown is ticked, delete the default text, and replace with an XHTML template using the button labelled **Import From File** (see Figure 4) and selecting file C:\Training\WWW\Web design 1\template.txt



Figure 4 - default text for new documents

1.3 Select the **Editor** tab and, working in the **Word Wrap Options** area, check the box labelled **Wrap at column**, leaving the following boxes with their default values: column set to 0 (auto-wrap) , and the other boxes, labelled **Show wrap column** and **Mark wrapped lines** checked (see Figure 5).

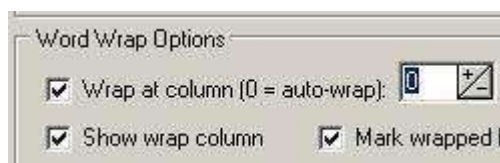


Figure 5 - setting word wrap options

1.4 Select the **Help** tab and uncheck the box labelled **Display after (idle time in milliseconds)**, to disable the **Tags Reminder** (See Figure 6).

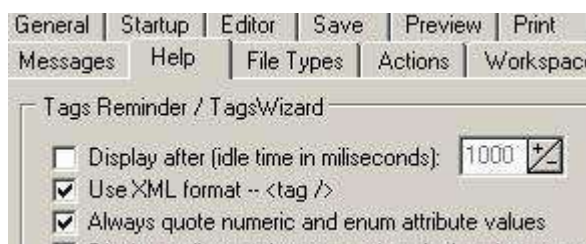


Figure 6 - disabling tags reminder

- 1.5 Select the **Auto Complete** tab and make sure the box labelled **Enable Auto Complete** is unchecked (see Figure 10)

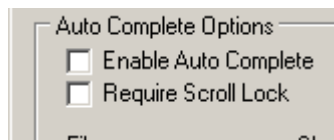


Figure 7 - disabling Auto Complete

The **Auto Complete** option, although useful during the initial construction of a page, can result in the addition of unwanted code during subsequent editing.

Note If you find the **Tags Reminder** feature useful, you can re-enable it. It works by predicting your code as you begin to type. If you choose to enable it, it is important that the **Use XML** format box is checked.

- Click the button labelled **OK**, at the bottom of the **Preferences** dialogue box.

Task 2 Creating a basic web page

Objectives To create a basic web page.

Comments You will create a basic (X)HTML document, add some text, and preview the result in a browser.

2.1 Open a new blank document:

➤ Go to **File / New Document**. You should see the following:

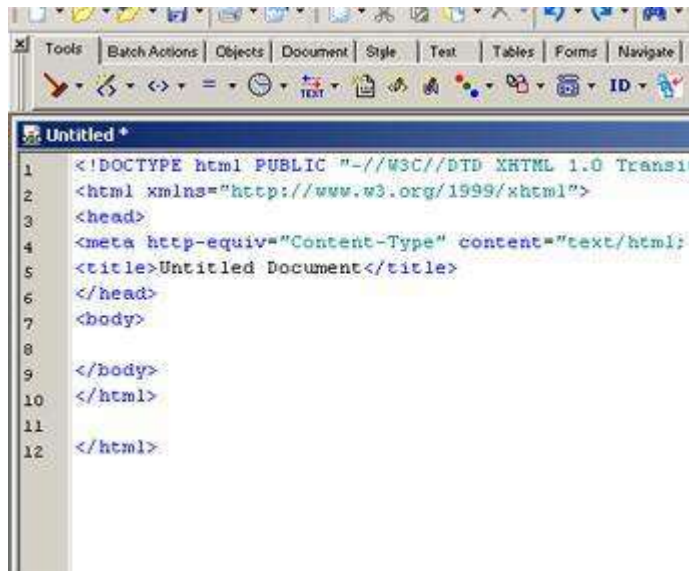


Figure 8 - new blank document

2.2 Add some content:

Between the `<body>` and `</body>` tags, type in the following text (or any text of your choice), starting new lines where indicated:

```
Joe Bloggs's website
Welcome
This site has been hand-coded to demonstrate the
simplicity of XHTML
On it, you will find information about me, a
description of my favourite hobbies and contact
information.
Enjoy your visit!
```

Note Replace **Joe Bloggs** with your own name. The above text can also be cut and pasted from: <http://www.bris.ac.uk/is/info/websupport/resources/cc1icw.html/>

2.3 Give the page a title:

➤ Replace the text **Untitled Document** between the title tags with the following, shown in bold.

```
<title>YourName's website – Home</title>
```

Note Remember that this title is not part of the main content of your page, it is part of the **metadata** (information **about** the page) contained in the `<head>` element. The title will appear in the browser window title bar (right at the top of the screen). It is also the default name for a link to the document (for example when users bookmark a site, the name of that bookmark will be the title text).

Saving a page

It is standard practice to give the home page of your site the default name **index.html**. Thus, once the page is on line you won't need to include it in the address bar or your browser.

- 2.4 Save your new file on your local computer. Select **File/Save As**. In the **Save As** window do the following:
 - Find the correct local directory (**C:\Training\WWW\Web design 1**) in the **Save in** box.
 - In the **File name** text box type **index.html**.
 - Click the **Save** button.

Note You could save your file on your local computer in any directory that you created.

Viewing your page in a browser

- 2.5 HTML-Kit has a 'preview in browser' facility that you can use to preview your pages.

- Click on the **Preview in default browser** icon  on the Toolbar.

Your document should be displayed as shown in Figure 9.

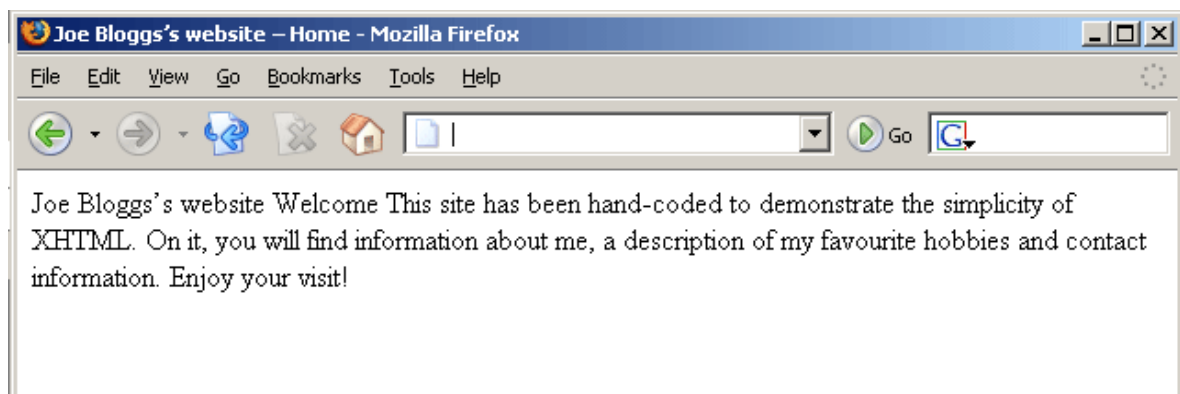


Figure 9 - document displayed as a single paragraph

Notice how the text appears in a single paragraph despite the line breaks in the source (HTML) document.

- Close the browser.

Note You can preview your page without saving. This will create a temporary file which can be deleted later.

- 2.6 View the page independently in a browser:
 - Open Internet Explorer or Mozilla Firefox.
 - Then go to **File / Open** (**File / Open File** in Firefox).
 - Click the **Browse** button and navigate to your file (C:\Training\WWW\Web design 1\index.html in this case) and open it.

Task 3 Structuring content

Objectives To add some content-structuring elements such as paragraphs, line breaks, headings and horizontal rules.

Comments You will be using the following (X)HTML tags for this task:
 Paragraph: `<p>...</p>`
 Headings: `<h1>...</h1>` ... `<h6>...</h6>`

Paragraph - `<p></p>`

3.1 Working in HTML-Kit, enclose each of the sentences between the `<p>` and `</p>` tags to form separate paragraphs as follows:

```
<p>Joe Bloggs' s website</p>
```

```
<p>Welcome</p>
```

```
<p> This site has been hand-coded to demonstrate the  
simplicity of XHTML </p>
```

```
<p>On it, you will find information about me, a brief  
description of my favourite hobbies and contact  
information.</p>
```

```
<p>Enjoy your visit!</p>
```

➤ Save and preview in your web browser (see Figure 10).

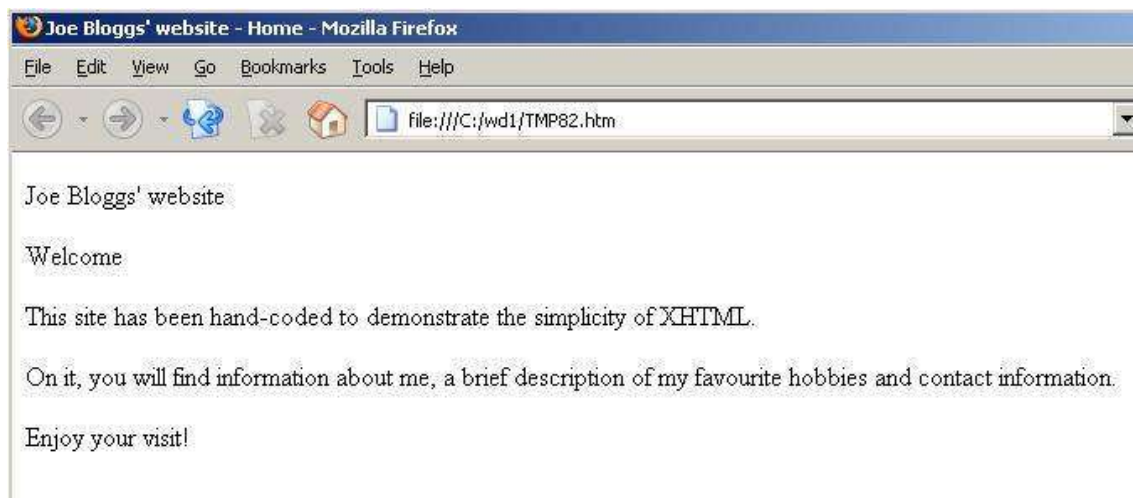


Figure 10 - five separate paragraphs

Note Text placed between the `<p>` and `</p>` tags forms a separate paragraph. A visual browser will typically add a line of space. The `
` tag simply breaks the text flow which continues on the next line. Use of the `
` is discouraged other where line breaks are essential to the document structure, for example: postal addresses or poetry. Under normal circumstances, text should be left to flow to accommodate available space – a concept know as **liquid design**.

➤ Close the browser.

Headings - `<h1></h1>` ... `<h6></h6>`

3.2 Add appropriate headings to your page:

- Make the first paragraph of your document a level 1 heading instead by replacing the `<p></p>` tags with `<h1></h1>` as follows:

```
<h1>Joe Bloggs's website</h1>
```
- In the same way, make the second paragraph a level 2 heading.
- Save and view in your web browser.



Figure 11 - level 1 and 2 headings

Note The University's house style uses sentence case for headings: apart from proper nouns, only the first word should have a capitalised initial letter. A full description of the house style is available here: <http://www.bristol.ac.uk/visualidentity/house-style.html>

Creating another page

- 3.3** Create a second page similar to the first one:
- Select **File / New Document**
 - In the `<head>`, make the content of the `<title>` tag **Joe Bloggs's website – about me**.
 - In the `<body>` type **Joe Bloggs's website** and make it a level 1 heading.
 - On the next line, type **About me** and make it a level 2 heading.
 - Save it as **aboutme.html** in `C:\Training\www\Web design`.

Task 4 Adding formatted lists

Objectives To insert some unordered and ordered lists in your page.

Comments You will be using the following (X)HTML tags for this task:
Unordered (bullet) list: `...`
Ordered (numbered) list: `...`

Ordered list - `...`

- 4.1** An ordered list is simply a numbered list. It consists of the `...` tags to create a new list and `...` tags to insert new list items.

Example:

```
<h3>Favourite websites</h3>
<ol>
  <li>BBC</li>
  <li>Ebay</li>
  <li>Youtube</li>
</ol>
```

will display Firefox as in Figure 12:



Figure 12 - an ordered list

- Add the ordered list, above, to your **aboutme.html** page.
- Save and preview in a browser.

Unordered lists - `...`

- 4.2** An unordered list is simply a bullet list. We still use the `...` tags to insert new list items but this time we enclose them between `...` tags.

Example:

```
<h3>Favourite websites</h3>
<ul>
  <li>BBC</li>
  <li>Ebay</li>
  <li>Youtube</li>
</ul>
```

...will display in Firefox as in Figure 13:

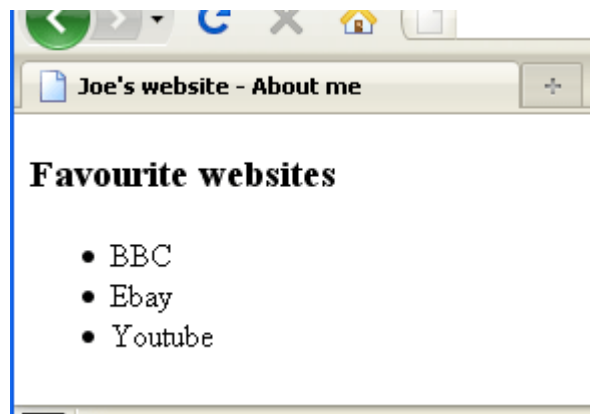


Figure 13 - an unordered list

- Replace the ordered list you created in tasks 4.1 with an unordered list.
- Save and preview in a browser.

Task 5 Changing the look and feel of a website, and the implications of bold and italic text

Objectives To apply some style elements to text.

Comments You will be using the following HTML tags for this task:

Bold: `...`

Italics: `...`

You will also use a simple style sheet to change the font type and colour of text.

Bold and italic text

5.1 Text can be made bold to highlight it.

- In your **index.html** file, choose a word or phrase that you want to make bold and enclose it between the `` and `` tags.

Example:

```
<strong>XHTML</strong>.
```

- Make 2 words/phrases in your page bold.
- Save and preview.

5.2 Italics are used for emphasis.

- Choose a word or phrase that you want to make italics and enclose it between the `` and `` tags.

Example:

```
<em>simplicity</em>.
```

- Make 2 words/phrases in your page italic.
- Save and preview.

Note In addition to rendering text bold or italic respectively, the `` and `` elements attach semantic meaning. This means they can be interpreted by visual and non-visual browsers.

Changing text colour and font type

Cascading Style Sheets (CSS) is used to apply the look and feel to a website; this includes choice of fonts, colour scheme etc. As with (X)HTML CSS can be created in any text editor

5.3 Link your pages to a style sheet called **style.css** in your working folder:

- Add the following line in the head of your document, just above the closing `</head>` tag:

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

- Save and preview. Note a completely new look and feel.
- Apply the style sheet to your **aboutme.html** page.

Note The ability to control look and feel of multiple pages using one external CSS file aids consistency throughout a website. It also removes the need for visual styling within the (x)html code, resulting in much cleaner, accessible content.

5.4 Open **style.css** and make a few changes to it:

- Go to **File / Open File** and open the file **style.css** in C:\Training\WWW\Web design 1.

Refer to the following lines:

```
body {font-family: Arial, Helvetica, sans-serif;
      background-color: #FFFFCC;}

h1 {font-family: Arial, Helvetica, sans-serif;
    color: #FFFFFF;
    background-color: #FF6600;
    border: 2px solid #CC3300;
    padding: 4px;}

h2 {color: #FF6600;
     border-bottom: 1px dashed #CCCCCC;}

h3 {color: #777777;}
```

Note the reference to HTML tags (`body`, `h1`, `h2`, etc) at the beginning of each line; they are known as **selectors** in CSS and are the link between the HTML document and the style. What follows, enclosed between two curly brackets: {}, are a list of **declarations** consisting of two parts: a **property** (for example: **color**, **font-family**, etc.) and one or more **values** (for example: **#FFFFCC**, **Arial**, etc).

In this case, the **selectors** simply specify which HTML elements are to be affected by one or more **declaration**.

- Change the values attached to the **color** and **background-color** properties in one or more selectors. To help you choose a colour combination that works, open the following page in a browser:
<http://colorshemesdesigner.com/>
(See Appendix B for more information on colours).
- Change the values attributed to the `font-family` properties of the `h1` selector, to one of the following:
 - Verdana, Arial, Helvetica, sans-serif;
 - "Courier New", Courier, monospace;
 - Georgia, "Times New Roman", Times, serif;

Note CSS specifies five generic font families that can be used in conjunction with individual font names. They are: **serif**, **sans-serif**, **monospace**, **cursive** and **fantasy**. The ability to display content in a specific font relies on that font having been installed on the end-user's computer, hence the inclusion of several options (in order of preference).

- Save your CSS file after each change and preview in a browser to see the changes.

Task 6 Creating hyperlinks

Objectives To create links between and within your pages, to other sites, and to an email address.

Comments You will use the `<a>...` (anchor¹) tags together with the `href` and `id` attributes.

The basic syntax associated with links is:

```
<a href="link-address">link text</a>
```

where `link-address` is the URL (address) of the object you are linking to and `link text` is what will be displayed on the page (browser default is normally blue and underlined).

Linking to an external web page

6.1 Linking to another website is the simplest type of link. The (X)HTML syntax is as follows:

```
<a href="http://www.bbc.co.uk">BBC</a>
```

This links to an **absolute URL**² (that is, the full path `http://www.bbc.co.uk` is used).

- Referring to the example above, return to the list you created earlier in your page `aboutme.html` and make the list items links to the corresponding websites:

Name	Absolute URL
BBC	<code>http://www.bbc.co.uk</code>
Ebay	<code>http://www.ebay.co.uk</code>
Youtube	<code>http://www.youtube.com</code>

- Save, preview and try your links.

Linking to another page in your site

6.2 Linking to an internal page is very similar to linking to an external site. The only difference is that you use the **relative URL**³ of the page you want to link to:

```
<a href="aboutme.html">About me</a>
```

In this example, the file we are linking to (`aboutme.html`) is in the same folder as the file we are writing to.

- Open your **index.html** page and in the second paragraph create the following links:

Text	Relative URL
about me	<code>aboutme.html</code>
favourite hobbies	<code>hobbies.html</code>
contact information	<code>contact.html</code>

Save, preview and try your links – notice that at the moment only your link to **aboutme.html** works, as the other two files do not exist. Also, at the moment, the

¹ See glossary

² See glossary

³ See glossary

only way you can come back to your homepage (**index.html**) is by clicking the **Back** button on the browser.

Note Once a site is live, a link to a page named index.html can be expressed as a trailing backslash (i.e. the final slash at the end of the address). Therefore, the address of an absolute link would be coded: `http://www.bris.ac.uk/mydepartment/` and a relative link to the index page of the current folder simply as a backslash (/)

When linking to and from pages within `http://www.bris.ac.uk`, the University standard practice is to code root-relative links. To do this, begin the link with a forward slash; this will link to the root of the server; then add the subsequent filepath. For example: `http://www.bris.ac.uk/french/` should be coded `/french/`

Linking to a bookmark within a page

Linking to a specific bookmark within a page is particularly useful when you have long pages that don't display on a typical single screen. Forcing your users to scroll up and down the page is an option but you can make life easier for them by providing links that will enable them to jump to specific sections in your page.

There are two stages when creating this type of link:

1. creating a bookmark, that is naming the specific location on your page where you want users to jump to;
2. creating a link to that bookmark.

First of all you need to add some content to your **aboutme.html** page, in order to make the page long enough to see the effect properly:

6.3 Open the following page in a browser:

<http://www.bristol.ac.uk/is/info/websupport/resources/>

- Follow the link: **Introduction to web page creation in XHTML**.
- Scroll down to the **Sample content for 'About me' page** section.
- Select and copy the sample code and text.
- Paste it between the `<body>` and `</body>` tags of your **aboutme.html** page after deleting the current content.
- Replace **Joe Bloggs** with your own name in the `<h1>` tag if you want.
- Save your file and preview it to see the changes.

You are going to link the items of the table of contents (numbered list) at the top of the page to the relevant section further down the page.

Note You will be able to change/edit the content of this page later on.

First you are going to name the **bookmarks**, in this case all level 3 headings, by using the **id** attribute in the `<h3>` tags. The **id** attribute defines a unique name (in other words, no two elements can be named with the same **id**).

6.4 Modify the first `<h3>` section head as follows:

```
<h3 id="bio-info">Biographical information:</h3>.
```

- In the same way, give a unique **id** to the other two `<h3>` section heads using the names in the table below:

Section heading	id
Favourite books	fav-books
Favourite websites	fav-web

- 6.5** Now that you have defined your anchors, you can create links to them. The syntax for linking to an anchor is similar to other links, except that (X)HTML uses the # sign to tell browsers to look for an id rather than a file:

```
<a href="#bookmark-name">Link text</a>
```

- Create a link from the first bullet point in the table of contents to the corresponding section as follows:

```
<li><a href="#bio-info">Biographical  
information</a></li>
```

- Repeat the above, linking the remaining bullet points to their corresponding headings, *Favourite books* and *Favourite websites*.
- Save, preview and try your links (you may need to decrease the size of the browser to see the full effect).

Note Similarly, you can easily create bookmarks and links that will take your visitors back to the top of the page. Standard University of Bristol templates have an id **#top**, so for most University sites there is no need to create this.

Linking to an e-mail address

Another useful type of link is the **mailto:email@address** which typically causes the default e-mail client application of the user's computer to open and makes it possible for users to send emails to the address specified.

- 6.6** Add an email link at the bottom of your two pages.

- First, go to your **index.html** file and add the following just above the closing `</body>` tag:

```
<p>Maintained by:  
<a href="mailto:your.name@bristol.ac.uk">  
your.name@bristol.ac.uk</a></p>
```

- Save and view in your browser.
- Finally, select, copy and paste this block of code into your **aboutme.html** file.

Note Using the email address as the visible link text will increase the visibility of your address to automated spammers. However, you cannot be sure that the end user's computer will be set to open an email client when a mailto link is selected, so this method ensures your address is visible for users to copy and paste into their chosen application. It also means the address is visible on printed copies of the page. The University recommends the use of shared mailboxes for website maintainer and feedback addresses, e.g. mydepartment-webteam@bristol.ac.uk. This links the address to a role, rather than a specific individual.

Create a simple navigation bar

- 6.7** Create a simple navigation bar that you can re-use in all your pages.

- Copy the code headed *Footer navigation bar* on webpage: <https://www.bris.ac.uk/is/info/websupport/resources/cc1icw.html>
- Paste the code into both of your pages, just above the maintainer email paragraph you created in task 6.6.

Note The `<div></div>` element is a generic container with no semantic meaning. It is used to contain user-defined sections of a page, for example: navigation bars, news boxes etc.

- Save and view both web pages. You should see the following (Figure 15):

[UoB home](#) | [Home](#) | [About me](#) | [Hobbies](#) | [Contact](#)

Maintained by: j.bloggs@bristol.ac.uk

Figure 15 – navigation bar

Styling the navigation bar with CSS classes

- 6.8** Return to your style sheet and look at the following code

```
/* Styling for navigation bar */
.navbar {padding: 8px;
         border: 1px solid #FF6600;
         background-color: #FFFFFF;}
```

Note The dot preceding the word *navbar* indicates a **CSS class**. Classes are user-named styles which can be applied to opening (X)HTML tags to add further styling to specific instances of that element; for example, you may wish to style two level 2 headings in different colours. For this reason, you will need to override the default set for the standard `h2` in your style sheet. Classes are covered in greater detail on another course.

- Add the class *navbar* to the opening `<div>` element in the navigation bar code you inserted on both pages as shown in bold:

```
<div class="navbar">
<a href="http://www.bris.ac.uk">UoB home</a> |
<a href="index.html">Home</a> |
<a href="aboutme.html">About me</a> |
<a href="hobbies.html">Hobbies</a> |
<a href="contact.html">Contact</a>
</div>
```

- Save your style sheet and view both web pages in your web browser. You may need to refresh/reload these to see changes. You should see the following (Figure 16):

[UoB home](#) | [Home](#) | [About me](#) | [Hobbies](#) | [Contact](#)

Maintained by: j.bloggs@bristol.ac.uk

Figure 14 – navigation bar styled using a CSS class

Note It is good practice to remove links to the page currently being viewed (i.e. a page should not link to itself). To achieve this, simply remove the `<a>... ` tags where appropriate, leaving just the text. For example, on the home page remove `` and the corresponding `` leaving just the text *Home*.

Task 7 Using images

Objectives To add images to your web pages.

Comments You will use the `` syntax to include an image. The `src` attribute of the `` tag specifies the source (that is, name and location) of an image. You will normally use `GIF` (used for simple illustrations, logos, cartoons etc.) or `JPEG` (used for colour photographs and complex illustrations,) image formats in your (X)HTML documents. The `PNG` format is also supported.

Creating an images folder

7.1 Create a new folder in your working folder called **images**:

- Go to **Start / Programs / Accessories / Windows Explorer**. Find your working folder (C:\Training\WWW\Web design 1).
- Then go to **File / New / Folder** and call it **images**.

Note It is standard practice to keep your images in a separate folder. This will make it easier to maintain your site as it grows. Note that the folder name is written in lower case. This is also good practice as web servers are case-sensitive, whereas local computers are not.

Inserting an image on a page

There are two ways to obtain images: either you create an image yourself (based on a scanned photograph or from scratch) using a graphics package (for example, **Adobe Illustrator, PhotoShop, Paintshop Pro** etc), or you use an existing image already optimised for the web (that is, **GIF, JPEG** (usually saved with .jpg file extension) or **PNG** format).

7.2 Re-open the page <http://www.bristol.ac.uk/is/info/websupport/resources/> in a browser (you may still have it open on your desktop).

- Click on **Introduction to web page creation in XHTML** and scroll to the bottom of the page where you will find two images: an orange University logo and a Computer Key labelled *Joe Bloggs*.
- Right-click on each image and select **Properties**.
- Make a note of the image dimensions, expressed in the order **width** and **height**.
- Download the University logo: **uob-logo-orange.gif** into the **images** folder you have just created, by right-clicking and choosing **Save Image As** (Firefox) / **Save Picture As** (Internet Explorer), browsing to the correct location and saving. Repeat for the image of the computer key: **comp-key.jpg**.

The element used to insert an image is:

```
<img />
```

Four attributes should always be included: the **source** (location) of the file (`src`) **width** (`width`), the **height** (`height`) and **alternative text** (`alt`). Height and width values are, by default, interpreted as pixels. The alternative text should provide a brief, meaningful description of the image for screen readers and other cases where images are not displayed; for example some users disable images in visual browsers to reduce page download time.

- Go back to your **index.html** file and insert the following code just below the `<body>` tag:

```

```

- In the same way, insert the image **comp-key.jpg** just below the line `<h2>Welcome</h2>`, giving it what you consider to be an appropriate `alt` attribute.
- Save and preview in a browser. You should see something like this (Figure 17):



Figure 17 - adding images

Note Providing alternative text for images is one of the basic requirements to ensure that your pages are accessible to people with disabilities (for example, blind or partially-sighted people using a screen-reader)

Controlling the position of images with CSS classes

7.3 User-defined **CSS classes** can be applied to `` tags to make images float left or right of your page, with text wrapping around.

- Look at the following at the bottom of your style sheet:

```
/* Classes to enable flexible image positioning */
.leftpic {float: left;
margin-right: 12px;
margin-bottom: 12px;}
.rightpic {float: right;
margin: 8px;}
```

- Add the classes `rightpic` to the logo image, by adding the code shown in bold:

```

```

- Add the class `leftpic` to the computer key image.
- Save your style sheet, and refresh `index.html` in your browser.
- Working in your style sheet, increase the padding applied to the `h1` element to `8px` in order to accommodate the floated logo image. If the background colour of

the h1 element does not match the image, set it back to #FF6600. Your page should now look something like this:



Figure 15 - images floated alongside content

Making an image a link

As a rule of thumb, users expect logos to link to appropriate URLs.

7.4 Link the University logo to the University home page.

- Enclosing the image of the University logo between the `...` tags as follows:

```
<a href="http://www.bristol.ac.uk"></a>
```

Note Since the University of Bristol logo is now a link, it is appropriate to change the alternative text to `alt="University of Bristol home"`, which conveys the **function** of the image (link to the University of Bristol homepage). It is more meaningful in this context than `alt="University of Bristol logo"`, which merely describes the **content** of the image (University of Bristol logo).

7.5 Save and view in your browser. Notice that a border has appeared around your image now that it is a link.

- Add a property to the `.rightpic` class to remove the border:

```
.rightpic {float: right;
margin: 8px
border: 0;}
```

- Save your style sheet and refresh the webpage in your browser.

Task 8 Using tables

Objectives To create a table for displaying tabular data.

Comments You will use the following tags:

table: `<table>...</table>`

table row: `<tr>...</tr>`

table data (cell): `<td>...</td>`

table head (cell): `<th>...</th>`

Displaying tabular data in a standard table

Creating a table in (X)HTML is relatively straightforward, provided everything is nested correctly. A table is defined with the `<table>` element. Each table has a set of **rows**, and each row is made up of **cells** containing the **table data** and **headers**. They are represented using the `<tr>` and `<td>` (`<th>` for headers) elements.

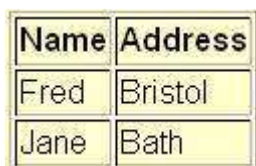
A simple, three-row, two-column table would be coded as follows:

```
<table border="1" summary="Staff addresses">
  <tr>
    <th scope="col">Name</th>
    <th scope="col">Address</th>
  </tr>

  <tr>
    <td>Fred</td>
    <td>Bristol</td>
  </tr>

  <tr>
    <td>Jane</td>
    <td>Bath</td>
  </tr>
</table>
```

... and look like Figure 16 in a browser.



Name	Address
Fred	Bristol
Jane	Bath

Figure 16 - a simple table

Note that:

- Tables should only be used to format tabluar data, not for layout purposes.

Note The **border** attribute in the opening `<table>` tag creates a border around the table (default is no border). This should now be added using CSS, whereas this exercise uses the older html method for simplicity.

- The **summary** attribute is an accessibility requirement and is used to describe the purpose of the table.
- This is the default table layout: the table is aligned to the left of the browser window, cell content is aligned to the left and cells stretch to accommodate the largest item in the cell.
- In a typical browser, the table header tags `<th>` centre-align the cell content and render the text bold. For structural reasons, it is important to define headers using this tag. Note also the `scope` attribute. This tells assistive technology (e.g. screen reader) whether the header defines a column or row of data. Values can be set to `row` or `col` (for column headers).

8.1 Open your **aboutme.html** file and replace the unordered list in the **favourite websites** section with a table. Follow the example code above to create a four-row, two-column table as illustrated in Figure 17, using the page URLs to provide both the text and link in the address column, for example:

```
<a href="http://www.bbc.co.uk">http://www.bbc.co.uk</a>
```

Favourite websites:

Website	Address
BBC	http://www.bbc.co.uk
Ebay	http://www.ebay.co.uk
Youtube	http://www.youtube.com

Figure 17 – table showing website names and addresses

Note Columns line up automatically. Width is set by the widest cell-content in each column, unless a width property is added.

Appendix A XHTML tags quick reference

Skeletal Tags

<html> ... **</html>** defines file as HTML document

<head> ... **</head>** invisible, descriptive part of page

<body> ... **</body>** visible part of page

Header Tags

<meta /> meta tags add information to head

Attributes:

name = "type" (for example, keywords/description)

content = "string"

<title> ... **</title>** title of page – appears in title bar

Hyperlinks

<a> ... **** anchor – defines link

Attributes:

href = url

title = "title text" – spoken out by screen readers and appears on screen as a tool tip when the mouse hovers over the link.

Lists

**** ... **** unordered list

**** ... **** ordered list

**** ... **** list item, used to define the content of each bullet point within either of the above lists.

Attributes (when used in conjunction with **** only):

value = "starting value"

<dl> ... **</dl>** definition list

<dd> ... **</dd>** item to be defined

<dt> ... **</dt>** definition

Tables

<table> ... **</table>** table

Attributes:

align="left/right/center" (position of table on page) – *should now be set via CSS*

border="pixels" (width)

cellpadding="pixels" (space around cell contents) – *should now be set via CSS*

cellspacing="pixels" (space between cell contents) – *should now be set via CSS*

cols="number of columns"

width="pixels or percentage of browser window" – *should now be set via CSS*

summary="string" (brief summary of table content)

<caption> ... </caption>

Attributes:

align="top/bottom/middle/right" - *should now be set via CSS*

<thead> ... </thead> table header

Wraps around header row to define headers:

`<thead><tr><th> ...</th></tr></thead>`

<tbody> ... </tbody> table header

Opens before first row of data cells and closes after the final row:

`<tbody><tr> ...</tr></tbody></table>`

A **<tfoot> ... </tfoot>** can be used to define a row containing a footnote. The code should be placed just before the `<tbody>` tag.

<tr> ... </tr> table row

Attributes:

align="left/right/center" (cell contents – horiz.) - *should now be set via CSS*

valign="top/middle/bottom" (cell contents – vert.) - *should now be set via CSS*

<td> ... </td> table cell

Attributes:

colspan="number of columns cell is to span"

rowspan="number of rows cell is to span"

width="pixels or percentage of table width" - *should now be set via CSS*

<th> ... </th> table head cell (bold & centred in visual browsers)

Attributes:

as for `<td>` element, along with:

scope="row/col" (direction of cells defined by header; either a row or column)

Text Markup Tags (inline elements)

** ... ** emphasised (usu. Italic)

<pre> ... </pre> preformatted fixed-width text

<small> ... </small> rel. smaller font

<strike> ... </strike> strikethrough

** ... ** strong (usu. bold)

_{...} subscript

^{...} superscript

** ... ** used to apply styling to inline content

Attributes:

class = "class_name" (defined in a style sheet)

id = "id_name" (defined in a style sheet)

Layout Formatting Tags (block-level elements)

<blockquote> ... **</blockquote>** quoted section

**
** line break: should only be used where semantically appropriate; for example, at the end of a line of poetry. It should not be used for visual formatting, as content should be allowed to flow and wrap naturally

<div> ... **</div>** division

Attributes:

class = "class-name" (defined in a style sheet; prefixed by a dot)

id = "id-name" (defined in a style sheet; prefixed by a hash: #)

<h1> ... **</h1>** heading (n=1-6)

<hr /> horizontal rule

<p> ... **</p>** new paragraph

Images

**** image

Attributes:

alt="description of image"

height="pixels"

width="pixels"

src="location of image file"

Appendix B Using colours on the web

The RGB Color model

Computer monitors generate colours by mixing the primary colours of light (called *additive primaries*): Red, Green and Blue. This is known as the **RGB Color model**. Each colour channel is 8-bit (one bit being 0 or 1), that is to say it has 256 different brightness settings (from 0 to 255). This means that by mixing the various brightness values in the red, green and blue colour channels we obtain more than 16 million (24-bit or 256 x 256 x 256) colours! This is more than enough colours to realistically represent the world as we see it. When all three colours are turned on at full intensity the result is white, when they are all turned off the result is black (i.e. no light at all).

The Hexadecimal code

(X)HTML uses the **hexadecimal code** (base-16 counting system) to represent colours on a web page.

Decimal	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Hexadecimal	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Decimal to hexadecimal conversion

It is made up of a combination of six letters and/or numbers (0 to 9 and A to F) divided into three pairs, each pair representing one channel of the RGB values, and preceded by a hash (#) symbol.

(X)HTML Hex code	Red (#)	Green (#)	Blue (#)	Colour
#000000	00 (0)	00 (0)	00 (0)	Black
...
#F8F400	F8 (248)	F4 (244)	00 (0)	yellow
#8DCB41	8D (141)	CB (203)	41 (65)	apple green
#E3372E	E3 (227)	37 (55)	2E (46)	red
...
#FFFFFF	FF (255)	FF (255)	FF (255)	White

Examples of colours expressed as hexadecimal codes (RGB numbers)

Web-safe colours

While modern 24-bit, or "True Color" monitors can display more than 16 million colours, there are still a small number of web surfers out there using older 8-bit or 16-bit monitors limited to displaying 256 or 65,536 colours.

When a browser is called upon to display a colour that is out of the monitor's range, it has two options: it can *dither* (mix two colours to produce the missing one) or it can *shift* the missing colour to one in the set available to the monitor. Sometimes the effect is hardly noticeable; sometimes it's a disaster.

This is less and less of a problem though, as the number of 8-bit or 16-bit monitors decreases every day.

However, there is no harm in using the web-safe colour palette; it simply limits your choices to 216 colours (40 are set aside for the operating system).

If you want to use web-safe colours, stick to a combination of values: 00 33 66 99 CC FF

Foreground / background contrast

When choosing a colour scheme for your site, it is far more important to choose colours that combine and contrast well, so that you don't make it difficult (if not impossible) for visually impaired users (colour blind people in particular) to view your site.

The following sites will help you choose colour combinations that work:

Color Schemes - <http://colorschemedesigner.com/>

Colour Selector - <http://www.limov.com/colour/?ID=0P44BK6DL2P0010>

Appendix C Design and planning tips

Directory names and structure and filenames

There is a good chance your website will continue to grow, so it is important to think carefully about the structure. Remember that directory and file names become part of the URL for a page.

- Keep directory and file names brief and sensible.
- Leave no blank spaces in names.
- Don't use underscore (_) as it is not always visible in underlined hypertext or the location bar on the browser.
- Use only lower case. Browsers are case sensitive, and it just makes things simpler. It is becoming the standard.
- If you are using relative links, make sure you don't move files without updating the links.

File Size

Large files take a long time to download and may cause viewers to give up on your website.

- Split large files into smaller ones and create links.
- Use images sparingly and only where they 'add value'. Use thumbnails where possible, and link to larger images if required.

Navigation

Analyse the information you are providing and identify logical routes through it and links between sections. Think about how people are going to want to look at your pages.

- Don't assume people will enter through your home page, so always include a link to it.
- Don't create dead end pages; always have a link back to a logical place.
- Don't rely on people using the 'Back' and 'Forward' buttons.
- Include a local search if you can.
- Include a navigation bar.

Consistent design

Be consistent in your design. This includes:

- page layout
- navigation
- colour schemes
- images and logos
- fonts and heading styles

This will help viewers navigate your pages, make them attractive, and save you time if you create a 'template' instead of redesigning every page.

Include these elements:

- signature line – institution, author/contact details, date of last update;
- link to home page;
- alternative text for images (use the `alt` attribute in the `` tag.

Accessibility

You must make sure that your pages are accessible to all users whatever their disability (for example, visual impairment, motor impairment, dyslexia etc).

Detailed guidelines are available (<http://www.techdis.ac.uk>), but follow these guidelines:

- Use colour sensibly, with high contrast.
- Use the ALT attribute to describe images.
- Provide captioning and transcripts of audio, and descriptions of video.
- Use meaningful hyperlink text.
- Provide alternative versions if required.
- Validate your pages using the W3C Validator (validator.w3.org).
- Use metadata to provide a description of your page. For search engines:

```
<meta name="description" content="Joe Bloggs's homepage" />
```

Printing web pages

Web pages that do not have print-specific style sheets often print badly. There are no page breaks, and a lot depends on the browser settings. If you want web pages to be printable, you need to create an appropriate style sheet, or might consider publishing your pages as Word/RTF documents or .pdf files to be downloaded.

Maintenance

Don't leave your pages languishing once you have created them! Keep them up to date. Frequently check that:

- the content is up to date;
- the links are still working;
- the images are still linked;
- the date in the signature reflects the latest check, even if nothing on the page has changed.

Appendix D Glossary of terms

Anchor	The anchor (<a>) tag is the HTML element used for defining both the source and the destination of a hyperlink.
Absolute link	A hypertext link that refers to the full path of a URL (for example, http://www.bbc.co.uk).
Block	Semantically-structured section of a document, for example: paragraph, heading, list item (see <i>Inline</i>)
Boilerplate	A template document that you use as a skeleton for all your pages. It is the framework - page layout and content outline - into which you add the detail and character for each page. It is a way of re-using HTML code which ensures a consistent look and feel across your pages and saves time.
Bookmark	An anchor on a page used as a target for a hypertext link on the same or a different page within your website.
Domain name	The name given to a web server. The University of Bristol domain name is www.bristol.ac.uk ; ac denotes "academic" and uk the country.
File access permissions	Access permissions determine who can read, write or execute programs in your files and directories. For example, you can set permissions so that others can view your files but not make changes to them.
FTP	File Transfer Protocol – a program for transferring files from your computer and a web server.
Home page	Your site's home page is the first page (also the welcome page) that will be accessed by users. The file should always be named <code>index.html</code> .
Hyperlink	A link to an object or location on the web.
Hypertext	A non-linear way of presenting information. Chunks or pages of text are linked together via paths, or hyperlinks.
Inline content	Unstructured section of a block, for example: words within a paragraph. Examples of inline formatting include the rendering of bold or italic text and creating hyperlinks: elements that do not affect a document's structure. (See <i>Block</i>)
Pixel	A pixel is one of the many tiny dots that make up the display on your computer. Display sizes vary but the most common one currently is 800 pixels wide by 600 pixels high (800 by 600).
Relative link	A hypertext link that refers to files within the same website (for example, news/article.html).
URL	Universal Resource Locator. The address of a resource on the web.
WYSIWYG editor	'What You See Is What You Get' – HTML editing software that enables web designers to create pages in a browser-like view, and creates the HTML coding in the background.

Appendix E Useful resources

Books:

Castro Elizabeth (2007), HTML, XHTML and CSS: Visual QuickStart Guide, Sixth Edition, Visual Quickstart Guide, Peachpit Press

An excellent beginner/intermediate level book, this will also prove a great reference as you progress. It is clear, concise and reasonably priced.

On the Web:

(X)HTML tutorials:

W3Schools.com – <http://www.w3schools.com/>.

Lots of free web building tutorials starting with basic HTML tutorials and leading to more advanced techniques.

HTML-Kit download and support:

<http://www.chami.com/html-kit/>

Download HTML Kit free of charge

HTML-Kit tutorials:

<http://www.ironspider.ca/hktutorials/>

Usability and Accessibility:

University of Bristol Web Accessibility Policy and Guidelines – http://www.bris.ac.uk/university/how_run/policies/access.html

If you are going to provide information on a University hosted web server, you must be aware of this policy and familiarise yourself with the guidelines.

TechDis – <http://www.techdis.ac.uk/>.

A JISC-funded service supporting the further and higher education community in all aspects of technology and disabilities and/or learning difficulties.

useit.com: Jakob Nielsen's website – <http://www.useit.com/>.

Jakob Nielsen is the usability guru. Check his 'Alertbox column' in particular; it contains useful 'How to' articles (for example, Top ten guidelines for homepage usability).

Validation tools:

W3C HTML Validation service – <http://validator.w3.org/>.

An extremely useful tool to check that your site conforms to W3C HTML standards. Enter the URL of your site and validate.

WAVE – <http://wave.webaim.org/>.

A tool to help web authors assess whether their pages are compliant with the W3C Web Content Accessibility Guidelines, and identify what changes are needed to make them compliant.