

HTML Workbook One

Contents

Task 1. Getting Started.....	2
Task 2. Creating a Simple HTML Document	4
Task 3. Your HTML Document	7
Task 4. Headings and Paragraphs.....	8
Task 5. Formatting Text.....	11
Task 6. HTML Comments.....	14
Task 7. Creating Hypertext Links.....	15
Task 8. Linking Files Together.....	16
Task 9. Including Images.....	17
Task 10. Using Image Attributes.....	19
Task 11. Images as Hyperlinks.....	20
Task 12. Best Practice.....	21
Task 13. Resources	22
Task 14. Homework One.....	23
Task 15. Homework 1 Marking	24

Task 1. Getting Started

Task 1.1

Download the zip file `html-workbooks.zip` from **Moodle** and extract the `html-workbooks` folder as instructed in class.

- This workbook uses the folder **workbook01**
- This folder contains all relevant exercise files
- Save all files created during this workbook to the `workbook1` folder.
- Delete the zip file once extracted to avoid confusion.

Task 1.2

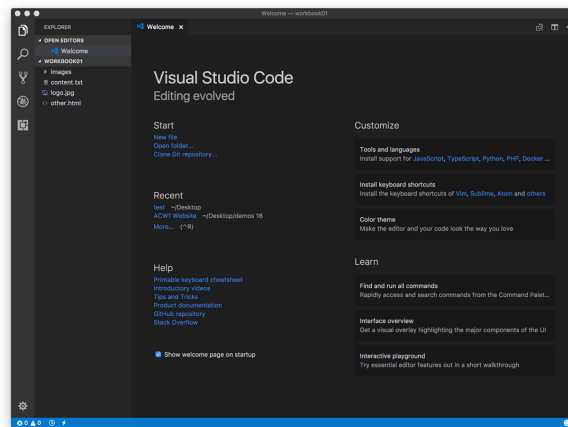
Open a HTML editor.

At MMU you will find a variety of editors installed including Notepad++, Sublime Text, Brackets and Microsoft Visual Studio Code. Installations vary from room to room. Microsoft's **Visual Studio Code** will be installed in all rooms we teach.

The Visual Studio Code editor is open source and free to install on your own Mac or PC. It is also fast becoming an industry favourite.

Follow these steps:

1. Open **Microsoft's Visual Studio Code** editor
2. Select **Open Folder**
3. Navigate to the `workbook01` folder on your H drive (or laptop)
4. Select **Open**
5. The **assets** (the files and sub-folder) in that folder are now displayed on the left pane of the application window.



Task 1.3

Find, select and open a web browser.
We recommend using Chrome or Firefox if possible.



Screen size	<p>If you are using a desktop computer the screen is likely to be a reasonable sized, widescreen monitor. Make use of the screen space – don't maximise applications. On a laptop you may have a number of 'virtual' screens that you can use at any one time.</p> <p>Ultimately, you'll find your own way of working but make sure it is both efficient and effective. Learn keyboard shortcuts like alt-tab (command-tab) to swop between programmes, ctrl-s (command-s) to save and F5 to refresh your web browser.</p>
Web browsers	<p>At MMU Internet Explorer is generally the default browser. You will also find other browsers including Google Chrome available. It is important to view your web pages in different browsers as they may display your pages differently.</p> <p>As a web developer you will become keenly aware of the problems that may arise trying to get web pages to display satisfactorily. There can be a number of factors:</p> <ul style="list-style-type: none"> ▪ The operating system, Mac, PC, iOS, Android etc. ▪ Make of web browser, Internet Explorer (IE), Firefox, Safari or Google Chrome. ▪ Version of web browser, ie. IE8, IE9 or IE10 etc. ▪ Size of users' screen, Laptop, PC, Mac, Smart phone, iPad. <p>By creating standards-based pages we aim to help avoid many of these issues.</p>

Things to know

No spaces	<p>On the web we avoid spaces in both file names and folder names. Why? If we load up a file called hello world.html in a browser what we get is:</p> <p>hello%20world.html – The %20 not only looks ugly, but it is hard to read.</p> <p>So what do we do? Common examples are:</p> <ul style="list-style-type: none"> ▪ hello-world.html using a hyphen ▪ hello_world.html using an underscore ▪ helloworld.html losing the space completely. <p>The secret is to select a method and stick with it. That will ensure consistency throughout your web site. Just remember, NO SPACES.</p>
No CAPS	<p>Only use lowercase letters when naming files and folders. It just makes life simpler for you as the developer, and for your user who then doesn't need to know if that is helloworld.html with a capital H or not. Simple, NO CAPS.</p> <p>MS Windows especially likes to mess this up by putting Capitals on file and folder names by default. Don't worry. It's only when you start uploading files to the web that you need to enforce lower case. Don't write CAPS in your code (e.g. <p> is better than <P>).</p>
.html suffix (or file extension)	<p>Much as MSWord files has the .docx suffix to allow it to identify files as MSWord files, for web pages we use the suffix .html to identify HTML pages.</p> <p>You will recognise file names such as index.html, which you will often find in the URL of the web sites that you currently use. For example:</p> <p>http://www.bl.uk/onlinegallery/index.html</p>
Old school	<p>You may come across .htm as the suffix for web pages but this is just a hangover from the days of MS DOS, when we could only use three characters for the file extension. Always use .html</p>
Plain text	<p>.html files are simple plain text documents. You may have come across .txt files. Unlike .doc, .ppt or other file formats plain text files only carry simple formatting such as paragraph breaks. This means they are light (in file size) and quick to load.</p>

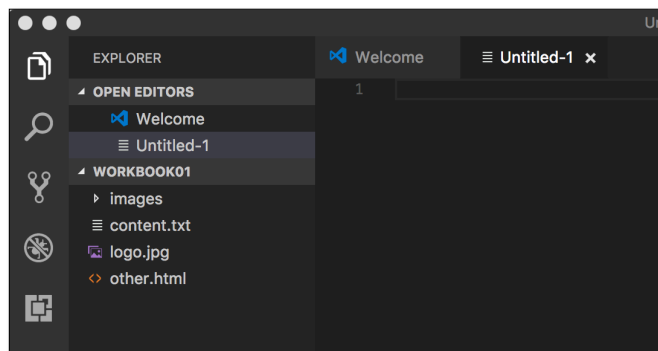
Task 2. Creating a Simple HTML Document

You will create a simple HTML document using a text editor and then view the document using a web browser.

Task 2.1

In the text editor **Visual Studio Code**, create a new document:

1. File > New



New document in Visual Studio Code.

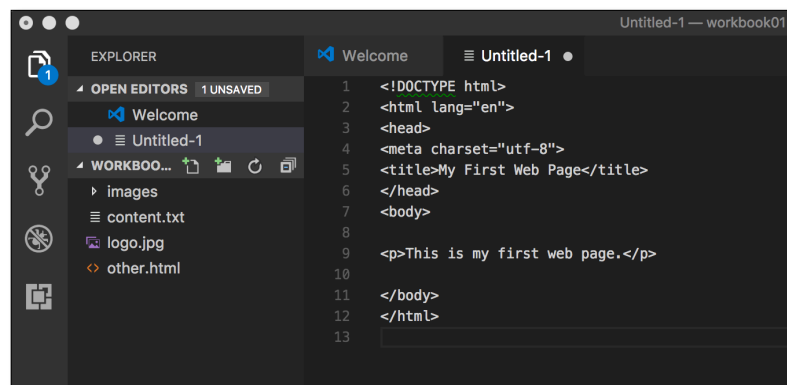
Task 2.2

In the text editor window type in the following:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>My First Web Page</title>
</head>
<body>

<p>This is my first web page.</p>

</body>
</html>
```

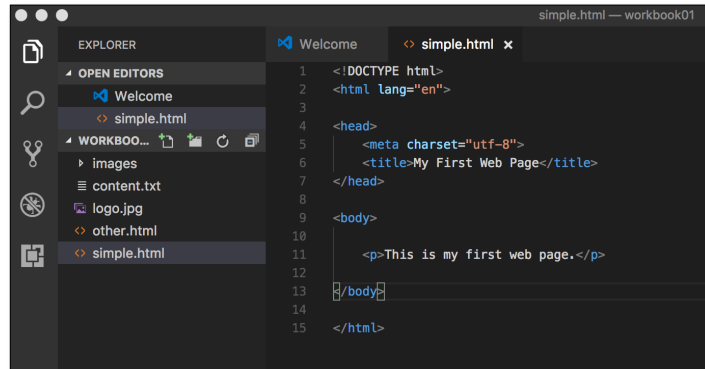


Task 2.3

Save the file to the `workbook1` folder as **`simple.html`**

To do this follow these steps:

1. Select **File > Save** in the **Visual Studio Code** menu
2. You should already be in the `workbook1` folder
3. Overtyping the filename `Untitled-1` with `simple.html` (web pages have the `.html` extension)
4. **Save**



`Simple.html` – note that once saved your code is colour coded, indented and spaced.

Task 2.4

To view **`simple.html`** in your web browser.

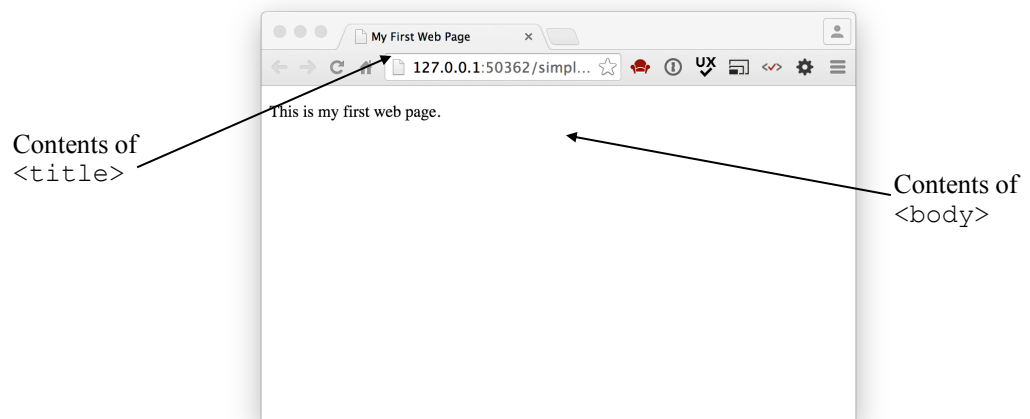
Locate the new file (**`simple.html`**) in the `workbook01` folder on your computer.

Now either:

- Drag the file onto a browser window.
- OR
- Double click on the file

Note: Double clicking on your HTML file will open up the file in the computer's default browser.

You will now see your first web page (**`simple.html`**) displayed in your browser. If you can't see anything, try to work out what is wrong and attempt to fix it **before** asking tutors.





Task 2.5

In developing a web page, you should get comfortable with a cycle of enhancing and amending your web page in your text editor, and viewing changes in your browser.

1. Try it out – add the following text (below, in **red/bold**) to your paragraph in `simple.html` using Visual Studio Code.

```
<p>This is my first web page created for MMU.</p>
```

2. Select **File > Save** in the **Visual Studio Code** menu
3. Now view again in your browser by *refreshing* the page.

To *refresh* you will need to select the refresh button  or  (or press F5) to see this new version of `simple.html` in your browser.

Throughout these workbooks you will be repeating this cycle of edit/save/refresh and view changes.

Notes

Remember - Don't maximise either your editor or your browser to fill your monitor screen – keep the windows at a manageable size.

Structure

These elements supply the underlying structure for our page:

- **<html>** tells the browser that we are starting a new HTML document
- **<head>** tells the browser that we are providing some information about the page. It is here that we put the page title.
- **<title>** tells the user the page title – displayed at the top of the browser and/or on the page tab.
- **<body>** tells the browser that we are starting the text, images and other content that we want the browser to display.
- The **<p>** element surrounds our first paragraph of text.

<head> and **<body>**

Every page has a **<head>** and **<body>**. The **<head>** contains information about the page such as the title, metadata for search engines, scripts and stylesheets. These are all things that are about or affect the content of the page, but are not actually displayed on the page.

Anything you want the user to see (text, graphics etc.) should be contained within the opening and closing **<body>** tags.

Notice that we finish our document by closing the **</body>** & **</html>** elements.

Visual Studio Code

Visual Studio Code has been really helpful. By first selecting the folder in which you are working at the start, when it came to saving `simple.html` you didn't have to find the correct folder – you were already in it! This is your *working* or *root* folder.

Get used to this. One of the common issues new coders have is saving files in locations other than their working folder. This often leads to multiple copies of the same file being worked on, causing confusion and frustration.

Getting Started and the User Guide for *Visual Studio Code* provide more information and advice on this excellent editor:

<https://code.visualstudio.com/docs>

Notes

VS Code will tidy up your code automatically when you save. It will for example indent code for you. VS Code will also try to auto-complete, guessing what tag you are coding. Some of this may get in the way at the start. As you progress you can use these features to your advantage. You can also customise them on your own computer.

Task 3. Your HTML Document

Take time to look at the code for `simple.html` in more detail.

Task 3.1

The **DOCTYPE** indicates to our browser what flavour of HTML we are using. In HTML5 this has been simplified compared to previous versions.

The **<head>** section contains information for the browser *about* the page.

This should always include a page **<title>** and a special **<meta>** tag which tells the browser which character set to use (in this case `utf-8`).

```
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="utf-8">

  <title>My First Web Page</title>
</head>

<body>

  <p>This is my first web page</p>

</body>

</html>
```

<body> tags enclose the content (text, media etc) to be displayed on the page.

<html> tags enclose the whole page. The 'lang' attribute defines the language of the page as English.

DOCTYPE

```
<!DOCTYPE html>
```

The **DOCTYPE (Document Type Definition or DTD)** defines what 'flavour' of HTML we are using to mark up our document. In this case we are using the 'new, simpler' DOCTYPE introduced in HTML5. It tells the browser how to display the page and what language you have used to mark it up.

The important bit is *what happens if we miss it out*. Failure to provide a DOCTYPE means that some browsers will *presume your page is old-fashioned, invalid or has strange (quirky) markup*. Some browsers will display your page in **Quirks Mode** and your page may not display as you intended.

Flavour of HTML

HTML5 is the current version of HTML. HTML specifications are managed by the World Wide Web (W3C) consortium - <https://www.w3.org/>. An excellent resource for learning more about HTML (and other web technologies) is MDN docs.

<https://developer.mozilla.org/en-US/docs/Learn/HTML>

HTML

```
<html lang="en">
```

The **<html>** element gives the browser and assistive technologies such as screen-readers the 'heads up' that we are opening a new html document. Likewise, the `lang` attribute gives clues to the language used. This helps a screen reader decide how to pronounce words. As Bruce and Remy point out in their excellent book *Introducing HTML5*, `six` is pronounced very differently in English and French.

We can be more specific and use `<html lang="en-gb">` for British English.

Character set

```
<meta charset="utf-8">
```

The *why* is not particularly important but again, adding the correct character set helps to ensure we have a *valid* HTML document. In your document we are using UTF-8 (8-bit Unicode Transformation Format), which is perfect for web pages as it supports a large character set allowing for umlauts, accents and different writing systems (e.g. Japanese, Cyrillic) on your web page.

Task 4. Headings and Paragraphs

To add text to a simple web page and structure the document for display in a browser.

You will add content to your file using your text editor and use HTML `<h#>` and `<p>` elements to structure the page output.

Notes HTML documents are just plain text files. All the basic structural and display information is contained within the HTML elements. These include instructions for handling headings, paragraphs and line breaks. White space and carriage returns in the *source* document (i.e. the HTML file you create in your editor) are not used to define the way the page is displayed in the browser.

Structure & Semantics Using HTML markup to purely define the structure is described as **Structural Markup**. **Semantics** relates to using the elements as intended, e.g. `<h#>` for headings, `<p>` for paragraphs and not vice versa. The presentational elements of a page are provided by Cascading Style Sheets (CSS).

How this task works... If you look at the code in Task 4.1 you will notice that the top section and the bottom closing tags are highlighted in a **red and bold** font style. In this and all further tasks we want you to add the **red coding and sometimes content** to the existing text, (which in this case is from the file `content.txt`).

Task 4.1 Open `content.txt` in the editor (just click on the file name in **Visual Studio Code**) and add the following HTML (**shown in red**) to your document (see **note** below before you start):

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>A Simple HTML Document</title>
</head>
<body>
```

Introduction to HTML

A simple HTML document can be...

Hypertext and Hyperlinks

HTML stands for HyperText Markup Language...

The W3C

The World Wide Web Consortium (W3C) coordinates the definition and development of HTML. Current standards and guidelines can be found at the official W3C HTML home page.

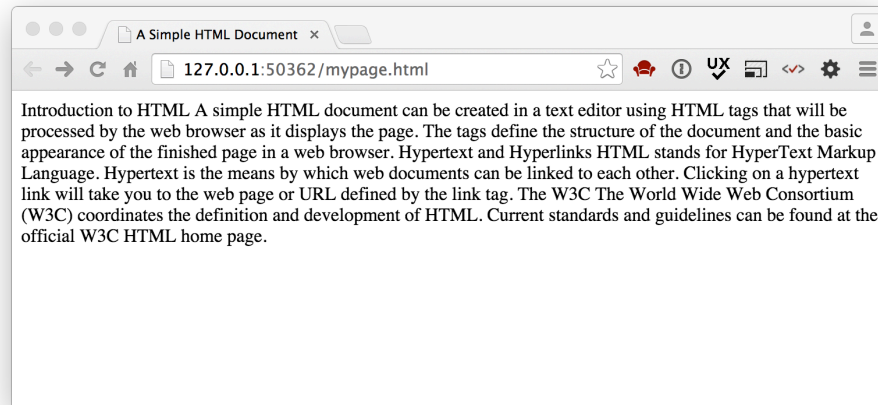
```
</body>
</html>
```

Save the file as **mypage.html** by selecting File > Save As.

Note Try copy & pasting from `simple.html` (and edit as required) rather than re-typing this code. As coders we don't like to 're-invent the wheel'. If we've already coded something in one document, why type it all out again in a new document? Cut & Paste is your friend. **However**, take note that re-using your own code is different to 'borrowing' other people's code without attribution to the original. As a student you might be accused of plagiarism; in the commercial world you may end up in court.

Task 4.2

Change to your web browser and open `mypage.html` (see **Task 2.4** for instructions if you are still unsure how). The following screen will be displayed:

**Note**

Notice the way the text has been displayed as one block. The browser does not process the white space in the source HTML file - **the HTML elements we are about to add will provide all the structural information.**

We are now going to introduce **Block Level Elements**. These are elements that create space by always having space above and below them. In this case both the `<h#>` and `<p>` elements will re-introduce the white space.

Task 4.3

Return to `mypage.html` in the editor.

Add the following HTML elements as **shown in red** below.

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>A Simple HTML Document</title>
</head>
<body>
<h1>Introduction to HTML</h1>
```

```
<p>A simple HTML document can be created in a text editor
using HTML tags that will be processed by the web browser
as it displays the page. The tags define the structure of
the document and the basic appearance of the finished page
in a web browser.</p>
```

```
<h2>Hypertext and Hyperlinks</h2>
```

```
<p>HTML stands for HyperText Markup Language. Hypertext is
the means by which web documents can be linked to each
other. Clicking on a hypertext link will take you to the
web page or URL defined by the link tag.</p>
```

```
<h2>The W3C</h2>
```

```
<p>The World Wide Web Consortium (W3C) coordinates the
definition and development of HTML. Current standards and
guidelines can be found at the official W3C HTML home
page.</p>
</body>
</html>
```

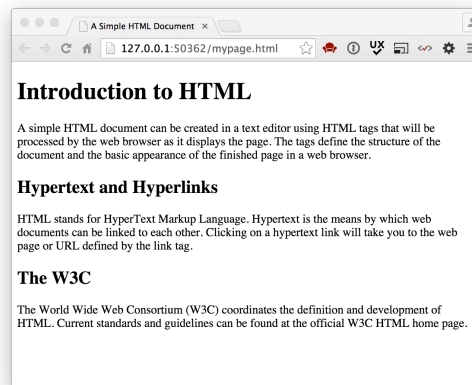
You now have `<h1>`, `<h2>` and `<p>` elements (in **bold**) to add to the document to create structure.

Task 4.4

Save the file.

Task 4.5

Return to your web browser and reload `mypage.html`. Notice that the individual headings and paragraphs are now displayed correctly.

**Headings**

Text headings are defined using the `<h#>` and `</h#>` elements where # is a number from 1 to 6 (i.e. giving six different levels of heading, `<h1>` to `<h6>`).

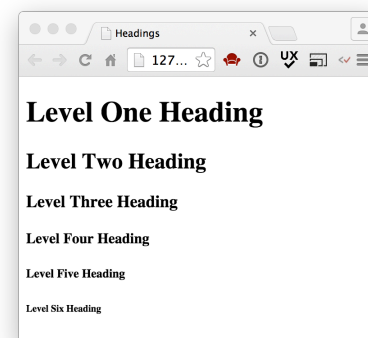
Header elements should not be nested i.e. only one heading style can be applied to a piece of text. **Wrong** - `<h1>Manchester <h2>Metro University</h2> </h1>`

Strictly speaking the heading elements are only structural, identifying the relevance of each piece of text within the document. By default, graphical browsers render the elements using default styling as a hierarchy of larger, bold text styles, as shown below.

The code:

```
<h1>Level One Heading</h1>
<h2>Level Two Heading</h2>
<h3>Level Three Heading</h3>
<h4>Level Four Heading</h4>
<h5>Level Five Heading</h5>
<h6>Level Six Heading</h6>
```

NOTE: Semantically there should only be one main heading `<h1>` per page (although HTML5 allows more). Only use **ONE** main heading `<h1></h1>` in your work for us – as a matter of good practice.



Things to know

Don't get wrapped up in the idea that the headings are displayed (by default) in different sizes. When you are introduced to Cascading Style Sheets (CSS) you will see that **any** level of heading can be 'styled' to **any** font size, font style or colour.

Don't place text in a `<h#>` element if it is not a heading. For example, don't be tempted to place a paragraph of text in a `<h1>` element to simply make the text larger.

Paragraphs are defined in HTML using the `<p> . . . </p>` tags. You shouldn't add a number inside the paragraph tags – all paragraphs in a text are equal in meaning.

Task 4.6

By default, text on web pages wraps within `<h#>` and `<p>` elements to fit the available display space. Line breaks can be forced by using the `
` element. (see below). There is no separate closing element for a line break. This is known as a self-closing element.

```
<p>
This paragraph will have 2 forced<br>
line-breaks<br>
in it.
</p>
```

Task 5. Formatting Text

You will use a text editor to add the `<hr>` `` and `` elements to your document.

Task 5.1

Add the following element (shown in **red/bold**) to `mypage.html` after the first paragraph:

```
<hr>
<h2>Hypertext and Hyperlinks</h2>
<p>HTML stands for HyperText ...
```

Save and reload your page. You should see a horizontal rule added to your page, above the heading.

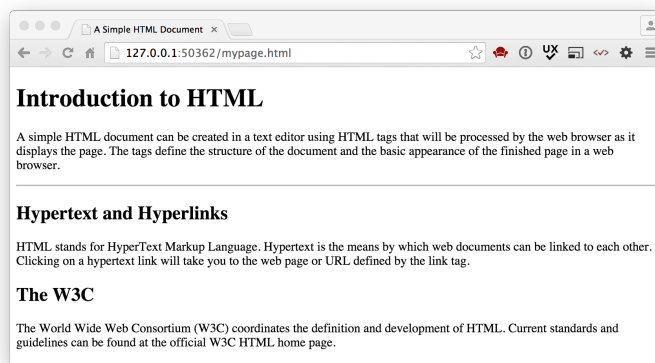


Figure 5-1 Adding a Horizontal Rule

Inline Level Elements

You have met **block level elements** in the form of paragraphs and headings (there are others including `<hr>`). You now understand that the browser renders space above and below each block level element.

The other type of element in HTML is the **inline level element**. As perhaps you can guess, the **inline level element** doesn't force a line break. This means we can use inline level elements within block level elements without disrupting the flow of text.

Task 5.2

Add the inline element `` to `mypage.html` as shown below:

```
...
<hr>
<h2>HTML, Hypertext and Hyperlinks</h2>
<p><strong>HTML</strong> stands for HyperText Markup
Language. Hypertext is the means by which web documents
can be linked to each other. ...
```

Save and reload your page. Check that the text `HTML` is now rendered in **bold**.

Task 5.3

Add the inline element `` as shown below:

```
...
<p><strong>HTML</strong> stands for <em>HyperText Markup
Language</em>. Hypertext is the means by which web
documents can be linked to each other. ...
```

Save and reload your page. Check that the text `HyperText Markup Language` is now rendered in *italics*.

``
``
`<i>`
``

In the world of HTML there has always been confusion regarding the use of either `` or `` (which browsers render both **bold** by default); and `` and `<i>` (which browsers render both in *italic*).

HTML5 has attempted to add semantic meaning (and differentiate) these elements:

If you use the `` element over a span of text it should be merely to draw attention to that text, rather than indicating any level of importance.

`<p>It was a dark and stormy night. The wind howled and twigs and leaves scuffled and rattled past the house. Mr and Mrs White sat in the parlour of their cosy home, in front of a blazing fire. </p>`

It was a dark and stormy night. The wind howled and twigs and leaves scuffled and rattled past the house. Mr and Mrs White sat in the parlour of their cosy home, in front of a blazing fire.

Using the `` element over a span of text is to indicate its importance.

`<p>Beware: Bad coding loses marks!</p>`

Beware: Bad coding loses marks!

If you use the `<i>` element over a span of text it should be to indicate an alternative voice or mood.

`<p>Captain Kirk sailed the galaxy in the spaceship <i>Enterprise</i>. </p>`

Captain Kirk sailed the galaxy in the spaceship *Enterprise*.

Using `` element over a span of text is to add linguistic stress emphasis.

`<p>I think Chris Hoy won.</p>`

I *think* Chris Hoy won.

`<p>I think Chris Hoy won.</p>`

I think *Chris Hoy* won.

`<p>I think Chris Hoy won.</p>`

I think Chris Hoy *won*.

In our examples in **Tasks 5.2 & 5.3** we have used `` to emphasise the importance of the word HTML in this sentence. We have wrapped the phrase *HyperText Markup Language* in `` to emphasise those words in speech.

You could also write the same paragraph in this way.

...
`<p>HTML stands for <i>HyperText Markup Language</i>. Hypertext is the means by which web documents can be linked to each other. ...`

In this case we merely want to draw attention to the word HTML and offset the phrase HyperText Markup Language.

Reading

You can read more about the semantics of these elements:

- developer.mozilla.org/en/docs/Web/HTML/Element/strong
- developer.mozilla.org/en-US/docs/Web/HTML/Element/em

CSS

Stylistically you will later learn to add boldness and italics to your text by using Cascading Style Sheets (CSS). This can be a much more efficient method.

Nesting Block level elements usually cannot be nested inside inline level elements; for example, you cannot have a `<p>` element nested within `` `` tags.

Correct:

`<p>`The strong element nested within the containing p tags in this paragraph - ``me`` - is fine.`</p>`

Wrong:

`<p>`This is not fine.`</p>`

Nesting – the exception HTML5 has brought about a change in that you are now allowed to wrap block level elements inside the `<a>``` inline level element. You get to meet `<a>` in Task 7.

Example:

`<h1>`The BBC Homepage`</h1>`

Avoid this until you are more experienced. This was introduced for a certain reason in HTML5 and may still cause some problems.

First In, Last Out This is also a suitable point to look at the over lapping of elements. The correct approach is often described as the **First In, Last Out** approach.

Correct:

First in. `<p>`This paragraph is correct as we start with the p tag, ``We have the em tags opening and closing``, before we then close the paragraph.`</p>`

Wrong:

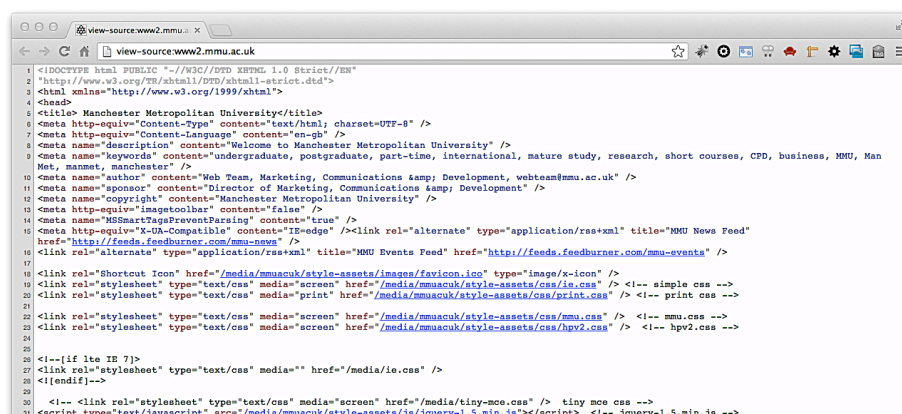
`<p>`This paragraph fails as you can see that the ``first tag is not the last to close.`</p>`

Last out.

View Source All web browsers have the ability to show you the coding behind most web pages. Look for *View Source* in your browser menu. This will usually open a new window or tab that contains the code of the page you are viewing. This is a great way to look at your own code or other people's work when learning. Just beware that looking at the code of more sophisticated/complicated sites (the BBC for example) can be quite frightening, even to the more experienced coder.

Learn the keyboard shortcut to *view source* in your favourite browsers.

Read <https://www.computerhope.com/issues/ch000746.htm>



View Source in Google Chrome (Mac)

Task 6. HTML Comments

You will use the `<!-- -->` elements to create a HTML comment.

Notes

Like most forms of coding it is handy at times to add comments and hide elements. In HTML this can be done with HTML comments. Anything between the opening `<!--` and the closing `-->` will not be displayed by the browser.

This is handy to add comments to our code such as reminders and notes about the code. They can also be used to hide whole blocks of code, essentially switching off elements of the page. This is sometimes handy when testing your pages.

Don't forget, your user can always see the comments or commented areas by 'viewing source' in the browser so don't comment out stuff you *really* don't want people to see such as passwords etc.

Task 6.1

Add a HTML comment just below the `<h1>` heading in `mypage.html`. Type the following text (**shown in red/bold**):

```
<h1>Netskills - Introduction to HTML</h1>
```

```
<!-- HTML comment. It will not be displayed in the  
browser -->
```

```
<p>A simple HTML document can be created in a text  
editor using HTML tags that will be ...
```

Save the file.

Task 6.2

Refresh and view `mypage.html` in your browser. You should **not** be able to see the comment if your syntax correct.

Task 6.3

Try commenting out one of the paragraphs using the opening `<!--` and closing `-->` comment elements. Save and view the file to see if it worked.

```
<!--
```

```
<p>The World Wide Web Consortium (W3C) coordinates the  
definition and development of HTML. Current standards and  
guidelines can be found at the official W3C HTML home  
page.</p>
```

```
-->
```

Task 6.4

Remove this comment to bring the paragraph back for the next task.

Save the file & refresh in the browser to check the paragraph is back.

Task 7. Creating Hypertext Links

You will use the `<a>` elements to create a hyperlink from your web page to the W3C site.

Notes Hyperlinks are what makes the web what it is. They provide a means by which web resources can be connected together and accessed using a web browser.

Task 7.1

Return to `mypage.html` in your editor and add the following text shown in **red/bold**:

```
<h2>The W3C</h2>

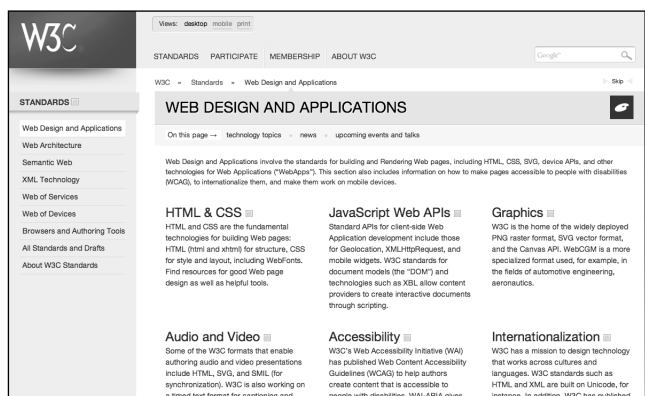
<p>The World Wide Web Consortium (W3C) coordinates the
definition and development of HTML. Current standards and
guidelines can be found at
<a href="https://www.w3.org/standards/webdesign/">the
official W3C HTML home page</a>.</p>
```

Save your file and then go to your web browser and Refresh your page.

Notice how the phrase the official W3C HTML home page is displayed differently from the rest of the text (by default blue and underlined).

Task 7.2

Click on the highlighted phrase the official W3C HTML home page with the left-mouse button. You will be taken to the HTML page, as shown below:



Task 7.3

Select **View > Source** (This varies on different browsers / browser versions). As we've said, this allows you to examine the HTML of the W3C homepage you are viewing. Take a look at the source code then close the HTML source window or tab when you have finished.

Notes The basic format for making a hyperlink is as follows. This is an example of a HTML element with an **attribute**:

```
<a href="document">Link text</a>
```

`href` is an abbreviation for **H**ypertext **R**EFerence, and is the 'attribute' within the HTML element.

`document` is the name of the file you wish to link to. This file can be another `.html` file within your web site or a full URL (web address) for a page on another web site.

`Link text` is the trigger text in the page. In a browser clicking on the text with your mouse will take you to the page specified by `href`.

Task 7.4

Add another link - Try creating a link on `mypage.html` to another web site such as Facebook, Twitter or the BBC. Test the link in your browser to make sure it works.

Task 8. Linking Files Together

To create a hyperlink to a locally delivered web page. You will use the `<a>` element again to link to one of your own web pages (rather than another web site).

Notes Hypertext links can be used both to link to other pages on the web and to your own files. The `<a>` element applies equally to both, but the address of the page may simply be a filename instead of a full URL.

Task 8.1

In the editor window add the following lines (in **red/bold**) to `mypage.html`:

```
<h2>The W3C</h2>
```

```
<p>The World Wide Web Consortium (W3C) ... found at  
<a href="http://www.w3c.org/MarkUp/">the official W3C HTML  
home page</a>.</p>
```

```
<h3>Linking to Local Files</h3>
```

```
<p>Here is a <a href="other.html">link to my other  
page</a></p>
```

Compare this link with the previous task and see how the text between the quotes is the name of a file rather than a URL.

Save the file.

Important Note

In this exercise you are linking to another file (`other.html`) in the same folder. You should have a copy of `other.html` in your `workbook1` folder.

Task 8.2

Switch to your web browser and refresh `mypage.html`.

Find the link you have created with the words link to my other page.

Select this link and see that your file `other.html` is now displayed.

If this doesn't work, go back and check that filename in the link is the same as the file you are linking to!

Task 8.3

Return to your editor and open `other.html`.

Add the following line to link back to `mypage.html`:

```
<p><a href="mypage.html">Go back to my first page</a></p>  
</body>  
</html>
```

Task 8.4

Save the file and then go to your web browser and **Refresh** the page `other.html`

You should now be able to switch between your two pages by selecting the relevant links.

Note

Linking between files in this way - using just the filename - requires that the two files are located in the same folder. How to link to files in sub-folders will be covered later.

Important Note

Although you've only just created your first links it's never too early to think about **link text**. A simple example is link text such as click here, which you've probably come across on a number of web sites. OK, so it tells you what to do (but you probably know that already). However, it tells you nothing about the page you are clicking to. More later, but for now **always make sure your link text is descriptive (of where the link is going to take the user)** on any pages you create.

Task 9. Including Images

You will use the `` element to embed an image. The `src` and `alt` attributes are introduced.

Notes The `` element specifies the name and location of an image.

Important Note

In this exercise you are linking to the image file (**logo.jpg**) in the same folder. You should have a copy of **logo.jpg** in your **workbook1** folder.

Task 9.1

Open `mypage.html` in your editor and insert the following text after the `<body>` tag, as shown below:

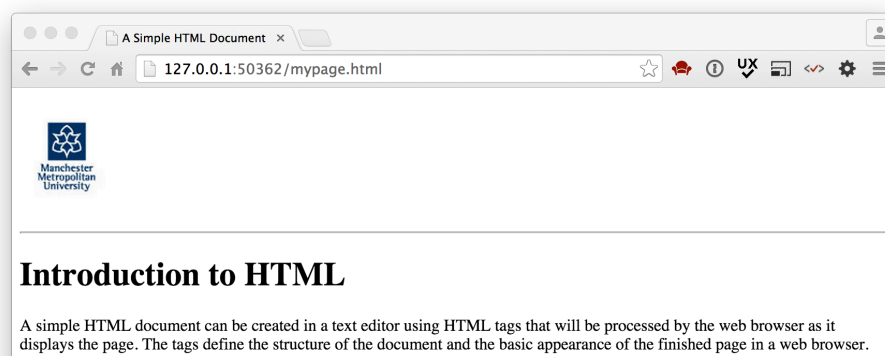
```
<body>
<p>
  
</p>
<hr>
```

Save the file and then go to your web browser.

Task 9.2

Reload `mypage.html`.

The MMU logo will appear in your document, as shown below.



Alt Text - As our image contains text, the alt (alternative) text simply replicates that text. If our image was of a "cat sat in a basket" then that would be our alt text, to describe the image contents.

“Alternative text is text associated with an image that serves the same purpose and conveys the same essential information as the image. In situations where the image is not available to the reader, perhaps because they have turned off images in their web browser or are using a screen reader due to a visual impairment, the alternative text ensures that no information or functionality is lost.” Source: en.wikipedia.org

“Screen readers are software programs that allow blind or visually impaired users to read the text that is displayed on the computer screen with a speech synthesizer or braille display. A screen reader is the interface between the computer's operating system, its applications [such as a web browser], and the user.” Source: afb.org

SEO -Search Engine Optimisation. A search engine like Google can't 'see' your image(s). Providing an alternative description (alt text) also helps Google better index the content of your site.

Task 9.3

You can test the `alt` attribute by editing the `src` attribute of the image element you have just created to introduce a deliberate error, e.g.

```

```

Save and reload the page.

You should see the following as the `alt` attribute is shown instead of the missing graphic:



Figure 9-1 - Alternative Image Text

Task 9.4

Correct your image element so that the image is displayed again.

Notes

The example above uses a local file *in the same folder as the HTML document*. In practice most images in your pages will be delivered using a locally stored image

Images can be stored in any folder within your web site and referenced by using a *relative* path (i.e. specifying where they are in relation to the current web page). You can also include images using an *absolute* reference (i.e. a URL). Some examples are shown below.

Value of `src` attribute

`src="logo.jpg"`

`src="images/logo.jpg"`

`src="../images/logo.jpg"`

`src="http://www.mmu.ac.uk/resources/images/logo.gif"`

Location of `logo.gif`

Same folder as web page

In a folder called `images` one level *down* from the one containing the web page (i.e. a sub-directory)

In a folder called `images` one level *up* from the one containing the web page.

At the URL specified i.e. not related to the position of the web page. This is known as an *absolute* reference.

Avoid linking to images using an *absolute* reference.

Task 9.5

Try linking to an image in a sub-folder called `images`.

Inside your `workbook1` folder you will see a folder called `images`.

Move the file `logo.jpg` into the `images` folder.

Change your link to the image to reflect the path to the `images` folder.

Clue: see option 2 in the examples above.

Save and reload `mypage.html`.

If it looks like Figure 9-1 (above) you have NOT got the path correct.

SEO

Search engines also read image filenames and count them towards SEO (search engine optimisation) — you should therefore give your image a descriptive filename (`dinosaur.jpg` is better than `img835.jpg`.)

Using images

You cannot Google and download ANY OLD IMAGE to add to your web site. Most images belong to someone. That means you require their permission to use their property. That said, there are many image libraries on the web, both free and commercial ones. Always make sure you check out copyright statements and conditions of use. In addition, never point your `src` attribute at an image hosted on someone else's website that you don't have permission to link to. You also have no control over whether the image is removed or replaced with something embarrassing.

Copyright

Your coursework: As a student you should not use any images, text or media that you have not produced yourself unless they specifically have a licence that allows you free usage. You will need to reference any such licence. We encourage you to create your own images. If in any doubt talk to your tutors as you may otherwise be accused of plagiarism.

Task 10. Using Image Attributes

To further specify the display characteristics of an embedded image. You will add the `height` and `width` attributes to the `` element. We will also add the `title` attribute.

Notes Most HTML elements have attributes that can be added to further define their behaviour in the web browser. Some attributes may be shared between elements; others are specific to one element or group of elements.

Task 10.1

Add `width` and `height` attributes (as shown in **red/bold**) to the `` element in `mypage.html`.

```

```

Save the file and then go to your web browser and reload your file.

There should be no noticeable difference in the appearance of the image in the page but you have now told the browser how much space to allocate for the image as it loads thus speeding up download of the whole page.

Task 10.2

Now change the size of the image using the `height` and `width` attributes. Add the values shown in bold to the appropriate line.

```

```

In HTML5 the `width` and `height` must be defined in pixels. Use `"46"` **not** `"46px"`.

Save the file and view it in your browser. You should see the image displayed at half the previous size.

Now try doubling the original values - 182 x 206. Save & view. You should see the image stretched to twice its normal size. Note that the image quality suffers.

Task 10.3

Reset your image to display at 91 x 103

The `width` and `height` attributes actually specify the amount of display space to allocate to the image, allowing the browser to assemble the rest of the page around it while it waits for the image(s) to download.

The attributes **do not** alter the size of the file itself. i.e. if your image is 3000x3000 pixels and 3 Mb in size and you use `width="100" height="100"`, the image will be displayed at 100x100 but the browser still has to download all 3 Mb of image!

Do not resize images using these attributes. If you wish to make your images physically smaller you will need to use a graphics editor such as Fireworks or Photoshop to physically resize the image file.

Always keep copies of your original images. Once a digital image has been reduced in size in an editor such as Photoshop it cannot be returned to the original size.

Task 10.4

Insert another image .



Insert the image `richard.jpg` which is already in the `images` folder.

Add `richard.jpg` to `mypage.html` above one of the `<h2>` headings.

Add `height` and `width` - `width="600" height="400"`

Add descriptive alt text – 5 to 8 words.

Show the tutor in class if you can to discuss your alt text.

Task 11. Images as Hyperlinks

To use an image as the trigger for a hyperlink. You will combine `<a>` and `` elements to turn an image into a hyperlink.

Notes In the 1990s using images instead of text for hyperlinks was common practice for creating button bars and graphical menus. This is now done with CSS and text. However, it is still useful for page elements such as company logos that link to the homepage.

Task 11.1

Add the following text (shown in **red/bold**) to `mypage.html`.

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

</a>
</p>
```

Save the file and view it in your browser. Your MMU logo should look no different, but now it is *clickable*.

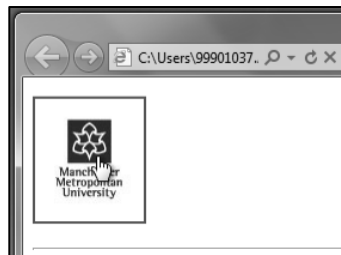


Figure 11-1 Images as Hyperlinks

Select the hyperlink (by clicking on the image) to navigate to the MMU homepage.

Note In some browsers (as above) you may notice that the image has an ugly blue border around it. Don't worry: you will learn how to remove this with Cascading Style Sheets later.

If you can't wait just add the following style (shown in **bold**) to `mypage.html`.

```

```

Task 11.2

Change the alt text attribute to reflect that the logo is now a link to the MMU homepage.

```

```

What text we use is reliant on the context. If this page was not on the MMU web site it might be better to not use the MMU acronym, but to spell out Manchester Metropolitan University in full.

Think about the screen-reader user. Will they understand the acronym, MMU? Do they want (or need) Manchester Metropolitan University read out in full to them?

Consider your alt text

Most screen reader users will prefer `alt="MMU Homepage"` to `alt="Manchester Metropolitan University Homepage"` for speed. As long as it is clear to them what the acronym MMU means.

DO NOT use `alt="Link to MMU Homepage"` as the screen reader user will already know that this image is acting as a link.

It's not an exact science, but do consider your alt text carefully.

Task 12. Best Practice

Since the move to HTML5, many in industry have chosen to carry forward the ‘best practices’ developed whilst using XHTML. We are doing the same. All coursework will be assessed to these standards and practices.

Notes Many books and tutorials on HTML5 follow the same practices. Some mix them up and some don’t follow them at all. **Always be aware of what we require when completing our coursework.**

Task 12.1

Best practices (things that get you marks/make life easier)

Use `<!DOCTYPE html>` or `<!doctype html>`

Use `
` instead of `
` because it is simpler and quicker.

For all coding:

1. Lower case for all elements & attributes – This means that there is no confusion, e.g. should or shouldn’t it be a Cap?

☒ ``

☐ ``

Note: when we are talking about CAPS, we are talking about the **elements** and **attributes**. We are not talking about any **text such as alt or title text**. Hence in the example above with still capitalize MMU and Homepage. When writing alt/titles, use correct grammar – it still needs to be legible.

2. Closing all open elements

☒ `<p>Some text...</p>` ☐ `<p>Some text...`

3. Quotes around most attribute values – Ensures your code is easier to read e.g. easier to identify quoted attribute values on the page.

☒ `` ☐ ``

Also means there are no problems when there is more than one attribute value:

``

Task 12.2

Other standards – Many older web pages will have been written in either HTML 4.01 or XHTML 1.0. It's therefore worth you seeing how the doctype and head differ.

For HTML 4.01

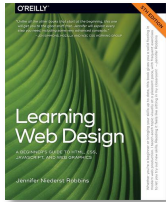
```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
<head>
<meta http-equiv="content-type" content="text/html;
charset=utf-8">
<title>My First Web Page</title>
</head>
```

As you can see, the main difference when compared to HTML5 is the very large DOCTYPE declaration and the additional attributes. For more information - http://www.w3schools.com/tags/tag_doctype.asp

Task 13. Resources

We recommend a number of books. They all work in their own way, some you may find easier to use than others. If coding is a mystery to you, the Jon Duckett book is beautifully and clearly illustrated. Jennifer's book is the most current and will help through both terms and further.

There is a reading list for each unit in Moodle. Many of the books we list will be in the library. Web design and development is a subject that requires a lot of reading to understand and keep up with current trends and practices.



Learning Web Design, 5th Edition by Jennifer Robbins

Just updated in 2018, this book covers everything we have time for and more. This edition now includes CSS Flexbox and Grid for sophisticated and flexible page layout, Responsive Web Design and a look at the command line, Git, and other tools in the modern web developer's toolkit plus SVG graphics.



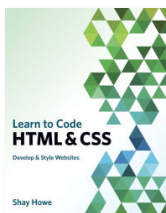
HTML and CSS: Visual QuickStart Guide, 8th Edition by Elizabeth Castro and Bruce Hyslop

We have been using this series since it started. Again it covers our course and more in detail. The book works through a variety of tasks, step by step.



HTML & CSS: Design and Build Web Sites by Jon Duckett

A fantastic book that is beautifully presented and illustrated throughout. This book clearly explains the main elements and attributes of both HTML and CSS. A great reference tool to have at your side. Do note that this book now a little dated. It mixes XHTML and HTML5 in its examples. Jon is working on a new edition.



Learn to Code HTML and CSS by Shay Howe.

This book goes into more detail than we have time for in class. Those interested should use this book as a starting point to develop their skills with HTML and CSS. You should note that it mixes both HTML and CSS from the start. Therefore, it is recommended you complete our workbooks first to get a grounding. This book is also available online, for free - <https://learn.shayhowe.com/>



Lynda.com

MMU has purchased a site licence for Lynda.com, which provides a multitude of online training videos, many related to web design. We will be providing links from Moodle to relevant videos each week. You are also free to explore Lynda.com. Videos also cover many of the applications you will come across on your course such as Photoshop.

How to log in to lynda.com

1. Go to www.lynda.com
 2. Click on Sign In, then choose 'Sign in with your organization portal'
 3. Enter **www.mmu.ac.uk** and click 'Go'
 4. Enter your MMU ID and password and then 'Login'
-

Task 14. Homework One

Now to put into practice what you've learnt. For next week's class we want you to develop a single web page that complies with the following brief.

The brief Select a web site that particularly inspires you. Is it the good design (colour, layout, typography) that you like? Is it easy, fun to use? Has it got great content? It may have some or all of these elements. It also might have some faults. Create a page as directed, link to your chosen site and write a few paragraphs about the site, as you see it, as a user.

Before you start: Use only the elements and techniques demonstrated in this workbook. If you know more elements already, don't use them. If you know some CSS, don't use it. Stick to what has been taught. Pay particular attention to the various tips and techniques explained. Read each task carefully. A big part of web design is following the brief – building *what the client requires*, not just what *you think* is required.

A

Using the folder called `workbook01` – Create a new web page and save it as `homework-one.html` to the `workbook01` folder. A simple technique would be to open `mypage.html` in your editor and Save As `homework-one.html` to the `workbook01` folder. You can then edit this file to fulfil the rest of this brief.

OK, you are now ready to get coding. *Tip - remember that cut & paste are your friends.*

B

Add/edit your page title using this format (**using name of your chosen website**):

```
<title>Homework One - BBC News</title>
```

C

Insert a clickable (linking to MMU homepage) logo at the top of your page using `new-logo.jpg`. *Tip – if you used `mypage.html` as your starting point you've already got some of this code. This new version of the logo is width 400, height 153.*

D

Next - on your page insert an `<h1>` with the name of your chosen site e.g. `<h1>BBC News</h1>` into the body of your page. *Tip – if you used `mypage.html` as your starting point re-use or delete the existing headings and paragraphs.*

E

Create a working hyperlink `<a>` to your chosen web site. *Tip - put this inside a paragraph. Think about your link text.*

F

Create three paragraphs `<p></p>` of original text about your chosen site, a mini critique about the design, content and usability. Use a `<h2>` for each sub-heading as illustrated. **Remember – spelling and grammar DO count.**

```
<h2>Design</h2>
<p>your text</p>
```

```
<h2>Content</h2>
<p>your text</p>
```

```
<h2>Usability</h2>
<p>your text</p>
```

G

And finally (because you need the practice) - insert one of the Geoffrey Manton building images from `images` folder including height & width and alt attributes. *Tip – both are `width="300" height="450"` Save it (`homework-one.html`).*

IMPORTANT: You have been continuously saving the page, haven't you?

Ask yourself - Have I actually followed the brief? Have I done what was asked? Have I done it to the standards and practices illustrated in this workbook? Does it all work?

Next Week

Have this work available at MMU ready to mark in next week's class.

Task 15. Homework 1 Marking

Name		MMU ID	
Degree			
Name of your chosen website -			
A	HTML5 file		
	Correct filename	homework-one.html <input type="checkbox"/>	
B	Metadata – <title>		
	Title - accurate syntax /spelling, formatted as instructed		5
C	Place new-logo within document (linked to mmu.ac.uk)		
	Inserted new-logo.jpg image including correct height and width size attributes		10
	Appropriate ALT description (for an image acting as a button)		5
	Working hyperlink to www.mmu.ac.uk		5
D	Heading		
	<h1> Heading as instructed		5
E	Hyperlink		
	Appropriately placed, working hyperlink to your chosen site		5
	Relevant link text – following good practice		5
F	Sub-headings and paragraphs		
	<h2> Heading as instructed - Design		5
	Well written paragraph, as per brief		10
	<h2> Heading as instructed - Content		5
	Well written paragraph, as per brief		10
	<h2> Heading as instructed - Usability		5
	Well written paragraph, as per brief		10
G	Geoffrey Manton images		
	manton-inside.jpg or manton-outside.jpg		
	Inserted image including correct height & width size attributes		10
	Appropriate ALT description		5
Marker's comments			
			Date: