MAS240

Cybercultures
3 Credit points

Workshop Manual
Semester 1, 2012

Media, Music, Communication and Cultural Studies
# Content

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content</td>
<td>1</td>
</tr>
<tr>
<td>About This Manual</td>
<td>3</td>
</tr>
<tr>
<td>The Golden Rule of MAS240</td>
<td>3</td>
</tr>
<tr>
<td>Workshop Schedule</td>
<td>4</td>
</tr>
<tr>
<td>Week 3: Introduction to HTML</td>
<td>6</td>
</tr>
<tr>
<td>What is HTML?</td>
<td>6</td>
</tr>
<tr>
<td>Basic HTML tags</td>
<td>9</td>
</tr>
<tr>
<td>Nesting Tags</td>
<td>9</td>
</tr>
<tr>
<td>Creating a List</td>
<td>10</td>
</tr>
<tr>
<td>Creating a hyperlink</td>
<td>10</td>
</tr>
<tr>
<td>Inserting an image</td>
<td>11</td>
</tr>
<tr>
<td>A note about <code>&lt;img&gt;</code></td>
<td>11</td>
</tr>
<tr>
<td>Week 3 Exercise</td>
<td>12</td>
</tr>
<tr>
<td>Week 4: Introduction to CSS</td>
<td>13</td>
</tr>
<tr>
<td>What is CSS?</td>
<td>13</td>
</tr>
<tr>
<td>Introduction to CSS</td>
<td>13</td>
</tr>
<tr>
<td>Why CSS is awesome!</td>
<td>15</td>
</tr>
<tr>
<td>Some styles require a bit more attention</td>
<td>16</td>
</tr>
<tr>
<td>Applying CSS across different HTML tags</td>
<td>17</td>
</tr>
<tr>
<td>Margins and Padding</td>
<td>17</td>
</tr>
<tr>
<td>Week 4 Exercise</td>
<td>19</td>
</tr>
<tr>
<td>Week 5: Creating Custom Styles</td>
<td>20</td>
</tr>
<tr>
<td>What is a custom style?</td>
<td>20</td>
</tr>
<tr>
<td>Class syntax</td>
<td>20</td>
</tr>
<tr>
<td>Applying a class to an HTML tag</td>
<td>21</td>
</tr>
<tr>
<td>What are classes used for?</td>
<td>21</td>
</tr>
<tr>
<td>The <code>&lt;span&gt;</code> tag</td>
<td>24</td>
</tr>
<tr>
<td>Week 5 Exercise</td>
<td>24</td>
</tr>
<tr>
<td>Week 6: Positioning</td>
<td>25</td>
</tr>
</tbody>
</table>
About This Manual

The MAS240 Cybercultures workshop manual is designed to assist your learning in practical workshops. It is not a substitute for attending workshops, but it will provide a reference for your learning and supplement instruction given by tutoring staff.

This manual was put together by Dr Steve Collins, Alex Mesker and John Cook.

This manual will help guide you through the practical workshops of MAS240 Cybercultures. It will assist you in developing Web production skills. No prior knowledge of Web production is assumed, but we do assume basic knowledge of computers and using the Web.

During the course of this semester, you will learn the following:

- How to hand code HTML
- How to use Cascading Style Sheets to:
  - Customise HTML tags
  - Design custom classes
- How to design Web pages using the box model
- How to resize and optimize images for the Web

You will hand code your Web pages rather than visually design them. This will give you a deeper appreciation for how the Web works and prepare those of you intending to pursue further Web design studies.

This manual will occasionally refer you to the unit page online. The address for this is:

http://media.mq.edu.au

The Golden Rule of MAS240

If you don’t understand something, then ask your tutor for clarification. We are more than happy to spend time explaining things to you … it’s what we get paid for!
<table>
<thead>
<tr>
<th>Week</th>
<th>Overview</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>No workshops</td>
</tr>
</tbody>
</table>
| 2    | Introduction to the workshops:  
  - Get familiar with Mac OSX  
  - How to log in  
  - Get familiar with [http://media.mq.edu.au](http://media.mq.edu.au)  
  - Web site assignment requirements |
| 3    | Introduction to HTML  
  - Introduction to HTML tags  
  - Basic text formatting  
  - Displaying images  
  - Making a hyperlink |
| 4    | Introduction to Cascading Style Sheets (CSS)  
  - Understanding the relationship between HTML and CSS  
  - Setting up a style sheet  
  - File paths  
  - Customising HTML tags |
| 5    | Creating Custom Styles  
  - Understanding classes  
  - How to create a class  
  - How to use classes |
| 6    | Positioning  
  - Introduction to the box model  
  - The <DIV> tag  
  - Creating classes to position HTML elements  
  - Floats |
| 7    | Introduction to Design  
  - Using images  
  - Background images  
  - Colours  
  - CSS trickery |
<table>
<thead>
<tr>
<th></th>
<th>Advanced CSS Styling</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>▪ Creating menus from lists</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Photoshop and Image Galleries</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>▪ Resizing images</td>
</tr>
<tr>
<td></td>
<td>▪ Optimising images for the Web</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>10-12</th>
<th>Project Development</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Weeks 10 to 12 provide an opportunity</td>
</tr>
<tr>
<td></td>
<td>for you to work on your Web site</td>
</tr>
<tr>
<td></td>
<td>assignments in class time with access</td>
</tr>
<tr>
<td></td>
<td>to staff expertise for assistance and</td>
</tr>
<tr>
<td></td>
<td>feedback.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>13</th>
<th>Web Site Showcase</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>In your timetabled workshops your Web</td>
</tr>
<tr>
<td></td>
<td>sites will be showcased to your peers.</td>
</tr>
<tr>
<td></td>
<td>This is not a formal presentation but</td>
</tr>
<tr>
<td></td>
<td>an opportunity for you to see what your</td>
</tr>
<tr>
<td></td>
<td>classmates have produced.</td>
</tr>
</tbody>
</table>
What is HTML?

HTML stands for:

HyperText

Markup

Language

HTML is the underlying language of the Web. It’s best explained as a formatting language. HTML uses a series of tags to tell Web browsers how to display documents. Let’s have a look at how it works.

Open your browser and go to http://media.mq.edu.au/mas240 and download the HTML template file. Make sure you save it to your Sites/mas240 folder.

Once the file is downloaded, open it in Text Wrangler.

What you are looking at here is the standard HTML document structure. At the moment this is just a blank page, but the browser requires all these things.

Your tutor will take you through what all these mean.

These are examples of HTML tags. Tags always use angle brackets, e.g. < >. Most have an opening and a closing tag:

<body> is an opening (or starting) tag

</body> is a closing (or ending) tag. The slash indicates closing.
The body section indicated by <body> and </body> is where everything you want to displayed on the Web page goes. Let’s put some text in there:

<body>
Here is some text
Here is some more text
</body>

Save your document by going to File >> Save As ...

When the save dialogue appears, navigate to your Sites/mas240 folder and save the file as text.html (don’t forget to add the .html extension).

Open your Web browser and go to: http://media.mq.edu.au/~your login name/mas240/text.html

You should notice two things:

1. Your text is displayed on the page
2. The browser has completely ignored the line break and put all the text on one line

Browsers are stupid
Browsers don’t understand things like line breaks. They’re quite stupid and require us to use HTML tags to format things like text. There are dozens and dozens of HTML tags and we won’t use them all throughout this unit, but there are several that you will come to know and love (or hate).

Go back to Text Wrangler and use the paragraph tag:

<body>
<p>Here is some text</p>
<p>Here is some more text</p>
</body>
Save your document and switch back to your browser and refresh the page. You should now see this:

Now you can see the paragraph tag at work. Congrats, you just used your first tag. The \texttt{<p>} dictates the start of a paragraph and \texttt{</p>} dictates the end.
Basic HTML tags

Now, we want you to spend a bit of time playing with some tags:

This screen shot demonstrates some of the basic tags used to structure text:

<table>
<thead>
<tr>
<th>Tag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;h1&gt;</td>
<td>The various heading tags are used to describe headings (no surprise there)</td>
</tr>
<tr>
<td>&lt;h2&gt;</td>
<td>of varying appearance.</td>
</tr>
<tr>
<td>&lt;h3&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;h4&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;h5&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;h6&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;p&gt;</td>
<td>Creates a paragraph</td>
</tr>
<tr>
<td>&lt;strong&gt;</td>
<td>Puts enclosed text in bold. If you have any previous HTML experience you</td>
</tr>
<tr>
<td></td>
<td>may have used &lt;b&gt; &lt;/b&gt;, but this is no longer correct. Get used to using</td>
</tr>
<tr>
<td></td>
<td>&lt;strong&gt; &lt;/strong&gt; or you will get marked down in your assignment.</td>
</tr>
<tr>
<td>&lt;em&gt;</td>
<td>Puts enclosed text in italics. If you have any previous HTML experience you</td>
</tr>
<tr>
<td></td>
<td>may have used &lt;i&gt; &lt;/i&gt;, but this is no longer correct. Get used to using</td>
</tr>
<tr>
<td></td>
<td>&lt;em&gt; &lt;/em&gt; or you will get marked down in your assignment.</td>
</tr>
</tbody>
</table>

Nesting Tags

You can nest tags inside of one another, for example:

<p>This a paragraph of text. You can nest tags inside of each other too. For example, <strong>this text is in bold</strong> and <em>this uses italics</em>.</p>
Creating a List
There are two types of lists – unordered and ordered lists.

- Unordered lists
- look like
- this
1. Ordered lists
2. however, look
3. like this

The basic structure is the same though:

<table>
<thead>
<tr>
<th>Unordered List</th>
<th>Ordered List</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;ul&gt;</code></td>
<td><code>&lt;ol&gt;</code></td>
</tr>
<tr>
<td><code>&lt;li&gt;</code>Unordered lists<code>&lt;/li&gt;</code></td>
<td><code>&lt;li&gt;</code>Unordered lists<code>&lt;/li&gt;</code></td>
</tr>
<tr>
<td><code>&lt;li&gt;</code>look like<code>&lt;/li&gt;</code></td>
<td><code>&lt;li&gt;</code>look like<code>&lt;/li&gt;</code></td>
</tr>
<tr>
<td><code>&lt;li&gt;</code>this<code>&lt;/li&gt;</code></td>
<td><code>&lt;li&gt;</code>this<code>&lt;/li&gt;</code></td>
</tr>
<tr>
<td><code>&lt;/ul&gt;</code></td>
<td><code>&lt;/ol&gt;</code></td>
</tr>
</tbody>
</table>

As you can see, each list type is opened using either `<ul>` or `<ol>` and then each individual list item is enclosed in `<li>` `</li>` tags and then the list is closed using `</ul>` or `</ol>`.

Creating a hyperlink
Creating a hyperlink is quite straightforward. This is the basic code:

```html
<a href="http://www.google.com">Click for Google</a>
```

The text “Click for Google” is what you will see on the Web page and when clicked will take you to Google. The `<a>` tag like many other tags is capable of handling properties.

What are properties?
Properties are extra things that can be placed inside of some tags. The `<a>` tag is quite useless on its own unless we use the href (hypertext reference) property which sets the destination for the link.

Some things to note about hyperlinks
If you are linking to a page outside of your own site then you must use http:// at the beginning of the address. If you’re referring to a page within your own site then you can just use the file name:

```html
<a href="next-page.html">Next Page</a>
```

The filename of the destination page and the href value **MUST** be the same.
If you want your hyperlink to open in a new window, then you simply need to amend your code to read:

\[
\text{<a href="next-page.html" target="_blank">Next Page</a>}
\]

You’d only probably use this if you were referring to a page on another Web site, as a general rule, you want internal site pages opening in the same window.

**Inserting an image**

You’ve probably put images in Word documents before? If you have, then you know that Word saves the image inside the document file. This is not the case with HTML documents. We use a special tag to tell the browser to display an image on the page, but we have to be able to tell the browser where the image can be found.

As a general rule of good practice in Web design, it’s best to keep your images in a separate folder to your HTML documents (but both will be inside your site folder) this is why you created an images folder when you defined your site last week.

Go to the MAS240 page (http://media.mq.edu.au/mas240) and download the image for this week and move it to the images folder inside your mas240learn folder.

The following code will get that image onto your Web page:

\[
\text{<img src="images/cheese.jpg"} />\]

**A note about <img>**

The <img> tag is a bit different to other tags in that it’s self-closing. There is no </img>, instead the tag closes itself using a /> as you can see in example above.

The tag uses a src (source) attribute to tell the browser where the image is stored on the server. The image is in a folder called images so therefore we need to tell the browser hence the file path images/cheese.jpg.
**Week 3 Exercise**

We want you to create a page with the following details:

<table>
<thead>
<tr>
<th>Text</th>
<th>Tags</th>
</tr>
</thead>
<tbody>
<tr>
<td>Your name</td>
<td>Heading 1</td>
</tr>
<tr>
<td>The degree you are studying</td>
<td>Heading 2</td>
</tr>
<tr>
<td>Date of birth</td>
<td>Heading 3</td>
</tr>
<tr>
<td>Why you decided to study MAS240 Cybercultures</td>
<td>Paragraphs</td>
</tr>
<tr>
<td>List of the units you are taking this semester with links to their entry in the online handbook</td>
<td>Unordered list, List item, Hyperlink</td>
</tr>
<tr>
<td>A picture of yourself</td>
<td>Image – You can download one from Facebook or take one during class, just ensure that the image is renamed to or saved as <em>me.jpg</em>.</td>
</tr>
</tbody>
</table>

**Dummy Student**

**BA Media**

1st January 1990

I'm studying Cybercultures because I want to learn about the Web and other cool stuff too. I think this will really help me in my degree and future employment.

This semester I'm studying:

- MAS240 Cybercultures
- MAS205 Radio Production
Week 4: Introduction to CSS

What is CSS?
CSS stands for Cascading Style Sheets. A style sheet is a separate document that is linked to by each Web page in a site. The style sheet contains information on how HTML elements are presented, for example:

- Type of font
- Font size
- Font colour
- Background colours
- Height and width of boxes
- Padding and margins
- Borders
- Positioning

Back in the day Web designers used some old HTML tags to control these things, but CSS has replaced all of that. CSS can be used in a few different ways, but for the purposes of this week you’re going to use it to control the appearance of your HTML tags.

You only need one CSS document for a whole Web site.

Introduction to CSS

Step 1
Open Text Wrangler and open your text.html document from last week.

Step 2
Go File >> New >> Text Document. When the new empty document opens, let’s save it straight away in Sites/mas240 as styles.css

You should now have both an HTML document and a CSS document open in Text Wrangler.

Step 3
Now we need to connect the HTML document to the CSS document.

Type the following code on a new line anywhere between the <head> and </head> section of the HTML document:

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

As the <link> tag suggests, this line of code “links” the HTML document to the CSS document.
Step 4
If you open your Web browser and go to
http://media.mq.edu.au/~your login name/mas240learn/text.html
you will see that the various HTML tags you have used have certain styles. For example, the <h1> tag is a big font size and is in bold. <h3> by contrast, is still in bold but a much smaller size. The default appearance for a hyperlink is blue, underlined text.
These appearances can be controlled using CSS to redefine the styles of HTML tags.
The syntax for CSS is different to HTML so it’ll take you a bit of getting used to.

Step 5
Flip back to Text Wrangler and switch to your CSS document and type the following:

```
   h1 {
     font-size: 20px;
     font-weight: normal;
   }
```

Let’s have a look at what this means and how it works:

This is called the selector and it can be an HTML tag or a custom class or ID (more on these next week)

These are properties with associated values. Note two things here. 1) properties always go between the opening { and closing } and 2) each line containing a property must end with a semi-colon.

Save the CSS document and refresh your browser. Your <h1> text should now be smaller and no longer in bold. Let’s add some colour into the mix:

```
   h1 {
     font-size: 20px;
     font-weight: normal;
     color: #FF0000;
   }
```

Note!
HTML and CSS uses American English spellings.
Colours are set using 6-digit codes.
Again, save the CSS document and refresh the browser, your text should now be in red.

CSS can achieve many things, so let’s go a bit crazy with this <h1> styling:

```css
h1 {
  font-size: 20px;
  font-weight: normal;
  color: #FF0000;
  border: 3px solid #CC33FF;
}
```

Some properties, such as border can be used in different ways.

In the above example, border sets all four sides of a box to look the same, but it is possible to affect each border side individually using:

- `border-top: value;`
- `border-bottom: value;`
- `border-left: value;`
- `border-right: value;`

Have a go at giving each border side a different colour.

**Why CSS is awesome!**

One of the many reasons why CSS is awesome is that if every single HTML document in your Web site links to a single CSS document, then you can update the look of the whole site just by editing one CSS document.

The above example would affect every single <h1> tag in the whole Web site. So, if you decide that you don’t like your headings in red, you can change the colour (or any other property) across the whole site just by editing your CSS document.
Some styles require a bit more attention

If you want to change the appearance of your hyperlinks, then you need to be aware of the following:

Hyperlinks have four inbuilt appearances (not that you’d know it from the default settings).

<table>
<thead>
<tr>
<th>State</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Link</td>
<td>This is what the hyperlink looks like when it hasn’t been clicked or the mouse isn’t hovering over it</td>
</tr>
<tr>
<td>Visited</td>
<td>This is what the hyperlink looks like when the destination link has been visited</td>
</tr>
<tr>
<td>Hover</td>
<td>This is what the hyperlink looks like when the mouse is hovering over it</td>
</tr>
<tr>
<td>Active</td>
<td>This is what the hyperlink looks like when the mouse clicks the link</td>
</tr>
</tbody>
</table>

An example of the syntax in action:

```html
a:link {
    color: #FF0000;
}
```

```html
a:visited {
    color: #FF0000;
    text-decoration: underline;
}
```

```html
a:hover {
    background-color: #CCCC33;
}
```

```html
a:active {
    font-weight: bold;
}
```
Applying CSS across different HTML tags

Sometimes you might want to make several tags look the same. You can achieve this by stringing a number of selectors together. If we use our h1 selector as an example:

```
h1, h2, h3 {
    font-size: 20px;
    font-weight: normal;
    color: #FF0000;
    border: 3px solid #CC33FF;
}
```

Save your CSS document and refresh your browser. The <h1>, <h2> and <h3> tags should now all look the same.

You can do the same thing with the hyperlink styling too. I tend to group my a:link and a:visited, and my a:hover and a:active. For example:

```
a:link, a:visited {
    color: #FF0000;
    font-weight: bold;
}
a:hover, a:active {
    text-decoration: underline;
}
```

Margins and Padding

You may have noticed that some tags such as the various heading tags and the paragraph tag have space around them. This can be achieved by using either margins or padding.

Margins create space around the OUTSIDE of the HTML element, whilst padding creates space around the INSIDE of the HTML element.

This is best demonstrated using a style that has a border. Using the method above, apply the same style to the h1, h2 and h3 tags and make sure that the style includes a border.
Now add the following line into that style:

```
margin: 0px;
```

All your headings should now be bunched up together.

Change the value to 30px and see what happens. See how the margin is applied around the OUTSIDE of each heading tag.

Like the border property, you can also specify exactly which sides have margins and which do not. For example:

```
margin-bottom: 30px;
margin-top: 0px;
```

Ok, delete any margins you have in place because now we’re going to look at padding.

Add the following line:

```
padding: 30px;
```

You should see how padding works on the INSIDE of the element. Padding can be side-specific like border and margin.
Week 4 Exercise

This week we would like you to decorate (as tastefully or not as you want) your text.html document using CSS. Refer to the CSS cheat sheet at the back of the manual or ask your tutor for help.

Here’s a tip: <body> represents the Web page. So if you want to colour the background of the whole page you need to use body as a selector for a style.
What is a custom style?
You’ve seen how CSS can be used to redefine the appearance of HTML tags. For example, if you redefine the <p> tag to set the colour property to red then this will affect every single instance of <p> across the whole Web site. What if you don’t want all paragraphs to be red? What if you want the majority of all your paragraphs to be in black but you want to highlight the occasional one in red? This is where custom styles called classes come into play.

A class works exactly the same as when you redefine an HTML tag, but you give it a custom name instead of nominating a tag. That name is then allocated to HTML tags as and when you choose.

Class syntax
Creating a custom class uses the same syntax as when redefining an HTML tag the only difference is that you make up a name for the selector.

Two additional things to remember in CSS documents: 1) class names are ALWAYS prefixed with a full stop, and 2) class names may not contain spaces.

Ok, this is an example of a custom class (this code goes into your CSS document):

```css
.redText {
    color: #FF0000;
}
```

It’s always good practice to give your classes names that are relevant to their function. There’s absolutely nothing to stop you from calling this class “Bob” but when you get onto big projects and you have dozens and dozens of classes and you’re looking for the one to make text red, “redText” seems sensible, but also be aware that at some point you might decide to change your red text to blue text ... suddenly the name redText doesn’t seem so apt! I’m using descriptive class names for learning purposes but for your own projects be sure to use class names that describe the function of the style, e.g. .importantText.
Applying a class to an HTML tag

As you saw last week, when you redefine an HTML tag the effects are immediately apparent because the browser understands tags. Classes however use custom selectors. In the above example, “redText” means nothing to a browser, there is no such HTML tag called <redText>.

Classes need to be applied to HTML tags. The process is quite painless. This example demonstrates how you would apply to the redText class to a <p> tag (in your HTML document):

<p class="redText">This text will be in red.</p>

Note that when you apply a class to an HTML tag you don’t use the prefixing full stop; that is only used when declaring the selector in the CSS document.

What are classes used for?

Classes can be used in a number of different ways.

Sometimes it’s not appropriate to redefine an HTML tag. As I said earlier, if you redefine the <p> tag then it will affect every <p> tag in the Web site. Classes are good for things like if you want to put a whole paragraph of text in a different colour, or give it a border – you can create class with those style properties and then apply it to ANY HTML tag.

Classes are also used with the <div> tag to create boxes for positioning. We’re not going to cover this today, but we’ll be looking at positioning and layout next week.

As I just mentioned, classes can be applied to any HTML tag. Let’s work on an example now to illustrate this.

**Step 1**
Download the HTML template, save it as classes.html, and open it in Text Wrangler.

**Step 2**
Create yourself a new CSS document and save it straight away in your mas240 folder as classes.css.

**Step 3**
Insert the following line of code into the <head> </head> section of your new HTML document:

<link href="classes.css" rel="stylesheet" type="text/css" />
Step 4
Switch back to your HTML document and in the <body> </body> section type out the following:

```html
<h1>This is a heading</h1>
<p>This is a paragraph of text.</p>
<p>This is another paragraph of text.</p>
<p>This is yet another paragraph of text.</p>
```

Save your HTML file and look at it in the browser. You should have something like this:

![Browser output of HTML content](image)

This is a heading
This is a paragraph of text.
This is another paragraph of text.
This is yet another paragraph of text.

Step 5
Switch to your CSS document (classes.css). You’re going to create a class:

```css
.redText {
  color: #FF0000;
}
```

Save your CSS document.
Step 6
Now you’re going to apply that class to the <h1> tag and one of the paragraph tags. Amend your HTML so it reads like this:

```html
<h1 class="redText">This is a heading</h1>
<p>This is a paragraph of text.</p>
<p class="redText">This is another paragraph of text.</p>
<p>This is yet another paragraph of text.</p>
```

Save your HTML file and refresh your browser. You should see something like this:

![Image of a browser showing a heading and paragraphs]

This is a heading

If you’re reading this in the printed manual then you’ll have to use your imagination and pretend that these are red!

As you can see, you can apply that class to any HTML tag you like.
The <span> tag
What about if you only want to highlight specific words in a paragraph? If you apply the class to the <p> tag then it will make all the words in the paragraph red.

This is where the <span> </span> tag comes in handy.

Switch back to your HTML document and amend the code to read:

```html
<h1 class="redText">This is a heading</h1>
<p>This is a paragraph of text.</p>
<p class="redText">This is another paragraph of text.</p>
<p>This is yet <span class="redText">another paragraph</span> of text.</p>
```

Save your file and refresh your browser.

As you can see the words “another paragraph” are now affected by the redText class but the rest of the paragraph remains in default black font.

**Week 5 Exercise**
For this week’s exercise we would like you to demonstrate what you have learned about classes by creating some custom classes of your own design and applying them to some HTML that you have also coded. The subject matter is not important, but demonstrating your grasp of HTML and CSS is.

Remember, if you need any help or anything explained then ask your tutor!
Week 6: Positioning

How are Web layouts structured?
The most efficient approach to Web design is the box model. This involves thinking about design in terms of boxes.

In the 90s, it was common practice to layout web pages using ‘tables’.

This method is both convoluted and outdated. In addition to this, pages laid out in such a way suffer from a number of usability and semantic issues, and hence this method of laying out content is frowned upon in MAS240 and you will lose marks if you use it in your Web site project.

The simple page mock-up below illustrates how we can use a series of boxes to structure Web page content.

Everything on the page is placed inside a large containing box (the surrounding grey area).

All the content – the site logo/heading, menu, content area including an image – is all placed in boxes inside of the containing box.
The basics of boxes
Coding a box intrinsically relies on this tag:

\[<div></div>\]

On its own the \(<div>\) tag doesn’t really do much, but when you create a custom class and apply it to a \(<div>\) tag then you can start to use it to position and structure your page.

In order to code something even as simple as the above example requires patience and understanding of the box model approach. This week you’re going to build a replica of the above mock-up, be introduced to some new HTML tags and style properties and further develop your knowledge of custom classes.

**Step 1**
Save a fresh copy of the HTML template to your Sites/mas240 folder as **position.html** and open it in Text Wrangler. Then create a new CSS document and save it straight away as **position.css**.

**Step 2**
Link your HTML document to your CSS document by adding the following into the \(<head>\) \(</head>\) section of your HTML:

\[<link href=“position.css” rel=“stylesheet” type=“text/css” />\]

Save both documents.

**Step 3**
Add the following code to your HTML document’s \(<body>\) \(</body>\) section:

\[<div class=“container”>\]

\[/div]\]

**Step 4**
Now switch to your CSS and create the following style:

You’ve set a width for the box.

\[
.container { \\
  width: 800px; \\
  height: auto; \\
  background-color: #CCCCCC; \\
}
\]

The value auto for the height means that the box will expand with the content inside it.

Background colour is a light grey. We’re going to give all the boxes different background colours so we can easily see them on the page.
If you save BOTH your HTML and CSS documents and check the HTML file in the browser you won’t see much yet. This is because the height value is set to auto so that containing box will expand to accommodate any content inside. As there is currently no content its height is currently zero.

**Step 5**
Let's create a logo/header box to go at the top of the page inside the containing box.

Amend your HTML so it reads:

```html
<div class="container">
  <div class="header">
    Header goes here
  </div>
</div>
```

Now let’s create a style for the header:

```css
.header {
  width: auto;
  height: 100px;
  background-color: #999900;
  font-size: 24px;
}
```

This time we’re setting the width to auto. This means that the box will be as wide as its containing box. We want to give the header a fixed height. Let’s set the font size for any text inside the header box.

Save both documents and refresh your browser. You should now see something like this:
**Step 6**
Now we create the menu. Amend your HTML to read:

```
<div class="container">
  <div class="header">
    Header goes here
  </div>
  <div class="menu">
    Menu goes here
  </div>
</div>
```

As you can see, the header and menu boxes are nested inside of the container box.

This is the closing `</div>` tag for the container.

Now create the style for the menu:

```
.menu {
  width: auto;
  height: 40px;
  background-color: #3399FF;
}
```

Save BOTH your HTML and CSS documents and refresh the browser. The menu should now be visible beneath the header.
Step 7
You’re on your own for this next step.

Create a box called content underneath the menu and style it with a red background colour.

Think about the CSS: If this is where all the text is going to go, then what values should you give the height and width properties in the style?

If you figure it out, you should end up with something like this:

![Diagram of the content area with header, menu, and content sections]

header goes here

menu goes here

content goes here
Step 8

Ok, in order for the rest of this to make sense, you’re going to need to get some content into your page.

Go to http://media.mq.edu.au/mas240/wk6-dummy-text.html and you’ll see that I’ve put some text up there complete with paragraph tags. You can just copy and paste this into your newly created content box – don’t forget to delete “content goes here” first.

Save your files and refresh the browser and you should now have something like this:

Note how once you insert content the top of your content box is no longer flush with the menu box.

The grey you can see here is the background colour of the container box.

The reason for this is the first <p> tag in the content. Paragraph tags have an in-built style which includes a top margin. We can easily remedy this by creating another style that only affects paragraphs in the content box:

```css
.container p {
  margin-top: 0px;
}
```

Save your CSS document and refresh the browser to see the changes.
**Step 9**
The next thing you’re going to do is insert an image into the content area. Save the coin image from the MAS240 page into your mas240/images folder.

Insert the following into your HTML below the opening <body> tag but above the first <p> tag.

```
<img src="images/coin.jpg" />
```

Save your HTML document and refresh the browser and you should see the image on the page.

**Step 10**
Switch to your CSS document. You’re going to make a new class and use a property you haven’t come across before:

```
.floatRight {
    float: right;
}
```

Now apply that class to the <img> tag in the HTML document:

```
<img src="images/coin.jpg" class="floatRight" />
```

Now save both documents and refresh the document. The image should now be on the right hand side of the page.

**What is “float”?**
Browsers usually display HTML elements in the order they are coded. For example, the HTML on the left produces the screen shot on the right:

```
<h1>This is a heading</h1>
<p>This is a paragraph of text. It appears immediately below the heading.</p>
<img src="images/me.jpg" />
<p>Here is another paragraph.</p>
```

As you can see, the way the browser displays the HTML reflects the logical, top-to-bottom order.

The float property, however, “floats” an element out of the top-down order and places it left or right depending on the value you give the float property.
Step 11
Go back to your HTML document and insert the image another two times so your code looks something like this:

```html
<body>
  <img src="images/coin.jpg" class="floatRight" />
  <img src="images/coin.jpg" class="floatRight" />
  <img src="images/coin.jpg" class="floatRight" />
</body>
<p>Lorem ipsum dolor sit amet, consectetur adipiscing elit ...</p>

Save your document and refresh the browser.

As you can see, the three images are all floated right in a row.

How can we get them into a column instead?

This is quite easy, switch to your CSS document and amend the floatRight style to read:

```css
.floatRight {
  float: right;
  clear: right;
}
```

Save and refresh the browser.

If you want to float to the left, then just change the value of the float property to left.

What is “clear”?

The clear property specifies the side(s) of an HTML element where other floating elements are not allowed. In this case it stops any floating elements to the right and forces them down.
Step 12
Do you think those coins look a bit bunched up together? Why don’t you try and give them a 10px margin on the bottom only?

If you’re successful, you should now have this:

Hmm ... I now think that maybe the text is a bit too close to the left side of the coin images. How do you think you’d go about creating a margin between the text and the coins?
Some important things about floats

You can float any HTML element not just images, but there is something of which you need to be aware.

When you create a style to float an image you don’t need to set a width property because the `<img>` tag knows its width because it’s the width of the image file itself. Other tags such as `<div>` or `<p>` don’t have a defined width by default so if you want to float them you must include a width in the class.

Go back to your HTML document and find the second paragraph; it begins “Donec ante risus.”

Amend its opening `<p>` tag to read:

```html
<p class="floatRight">
```

Save and refresh the browser.

If you can find that paragraph on the page you’ll see that all that’s happened is that it’s moved down the page and appeared halfway through another paragraph, but see what happens if you add a width property with a value of 150px to the floatRight style.

See how it has now moved cleanly beneath the images, but scroll down the page and you’ll see that now we have another hiccup:

See how the newly floated text is breaking out of the content `<div>`?

(I’ve given it a colour so you can clearly see what’s happened.)

This is always something to be careful of, but luckily it’s easily fixed with one line of code in your HTML:

Find the closing `</div>` for the content box (it should be right above the closing `</div>` for the container box).

Insert the following BEFORE the closing `</div>`

```html
<div style="clear:both;"></div>
```

Save and refresh the browser. You should now see the red content box extend to the same height as the floating column.
Floating columns using `<div>`

This is the last thing for this topic, but it’s quite important and there’s a bit of math involved.

This section involves `<div>` tags, floats, margins and padding.

You’re going to replicate the following:

![Image of floating columns](image-url)

We’ll go about this in a slightly different way to the previous example but it’ll be a bit quicker.

**Step 1**
Create a new HTML document and save it as `position2.html` in your mas240 folder.

**Step 2**
Insert the code to link this page to your CSS document. You can copy and paste it from `position.html` as we’ll be using the same CSS document.

**Step 3**
Type out the following HTML in the `<body> </body>` section of your HTML document:

```
<div class="container">
  <div class="topBox">Top Box</div>
  <div class="leftBox">Left Box</div>
  <div class="rightBox">Right Box</div>
</div>
```
If you look at this in the browser it’s very unimpressive. All we’ve done is write the HTML and call some classes that we haven’t yet declared in the CSS document. Let’s do that next.

**Step 4**

```css
.topBox {
    background-color: #00CC66;
}
.leftBox {
    width: 500px;
    height: 200px;
    background-color: #339999;
    float: left;
}
.rightBox {
    width: 300px;
    height: 200px;
    background-color: #663399;
    float: right;
}
.bottomBox {
    background-color: #CC3300;
}
```

Remember all the boxes are nested inside the container box.

You may recall that the container’s styling class has a width property with a value of 800px.

The combined width of any inner boxes cannot exceed the width of the containing element, and you’ll recall from a bit earlier when I said that any floated element (except for `<img>`) must have a defined width.

This is why we have set the widths for leftBox and rightBox:

500px + 300px = 800px (the width of the containing box)

Note: If you include any margins, padding or borders in a style these add to the width and height. For example, if you create a class that has a width of 500px with a height of 400px and you add 10px of padding then the true width and height becomes 520px and 420px (remember that’s 10px of padding on each side of the box). If you do this then you should amend the width and height properties by subtracting the amount of padding or margin: e.g. the width property should 480px + 20px (left and right padding of 10px) = 500px.

If you add a 1px border then you’ll also have to subtract 2px from both the height and width.

Save BOTH your HTML and CSS documents and check position2.html in the browser. You should have successfully replicated the screen shot.
If you change the height values for leftBox and rightBox to auto to allow content to expand then problems start to occur.

**Step 5**

Make the changes to the height values and then create a few paragraphs of text in leftBox.

Save your files and refresh the browser. See how the layout breaks:

![Image showing layout issue](image)

This is easily fixed by adding a clear: both to the style for bottomBox:

```css
.bottomBox {
    background-color:#CC3300;
    clear: both;
}
```

Save your CSS document and refresh the browser and everything should be perfect. You’ll be able to see the grey background of the container box where rightBox hasn’t extended.

Try adding several paragraphs to the rightBox to make it longer than leftBox and you’ll see that the layout remains intact.
Week 7: Introduction to Design

This session introduces you to the basics of design. The second semester unit MAS241 Interactive Web Design offers more detailed approaches to Web design.

The session will take you through some of the different ways images can be used in Web sites design and introduce you to some of the newer CSS tricks. Note that much of the newer CSS properties that you’ll learn today are not supported in older browsers.

Using images

Images can be used in two different ways in Web design:

The first involves using images as part of the content. For example, if your Web site is for a guitar shop then you’ll want to have pictures of guitars as part of the content so customers can see what products you have on offer.

The second (and what we’re concerned with in this session) involves using images as part of the design.

Images can be used as background images for any HTML element. You’re going to learn how to set a background image on the Web page as well as in <div>.

You’ll need to download the two images (greyBG.gif and guitar.jpg) for this week from the MAS240 page and move them to your mas240/images folder.

Background images

Step 1
Create a new HTML document and save it as image.html and then create a new CSS document and save it as image.css.

Step 2
Link your HTML document to the CSS document using the following code in the <head> </head> section of the HTML document:

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

Step 3
You’re going to apply a tiling background image to the whole Web page. Remember the <body> </body> section represents what you see in the browser. The <body> tag by default has some inbuilt styles, one of which is a background colour set to white. You’re going to redefine the <body> tag’s style to include a background image.

Add the following to your CSS document:

```css
body {
    background-image: url(images/greyBG.gif);
}
```

Save your CSS document and check the HTML document in the browser. You should see the background image tiling across the whole page.

Step 4
Now you’re going to create a box and in a moment we’ll set a background image for it. Add the following to <body> </body> of your HTML document:

```html
<div class="imageBox"></div>
```

Now let’s create the imageBox class in the CSS document:

```css
.imageBox {
    width: 500px;
    height: 400px;
    background-color: #CCCCCC;
    border: 2px solid black;
}
```

Save your HTML and CSS documents and refresh the browser. You should now see the box on the page.
**Step 5**
Now let’s add a background image to the imgBox. Amend its class to read:

```css
.imageBox {
    width: 500px;
    height: 400px;
    background-color: #CCCCCC;
    border: 2px solid black;
    background-image: url(images/coin.jpg);
}
```

Save your CSS document and refresh the browser. You should now see the coin image tiling throughout the box. We’re going to introduce some extra properties and values now to show you different ways in which background images can be used.

**Step 6**
Add the following to the imgBox class:

```css
background-repeat: repeat-x;
```

Save your CSS document and refresh the browser. You should now see the coin image tiled in one row along the x axis (i.e. left to right).

Amend the background-repeat property by changing the value to `repeat-y`.

Save and refresh the browser. You should now see the coin image tiled in one column along the y axis (i.e. top to bottom).

**Step 7**
Now amend the background-repeat property by changing the value to `no-repeat`. Save and refresh the browser. You should now see only one instance of the coin image.

It’s also possible to change the position of a background image. You’ll notice that it defaults to top-left. You can change this by using:

```css
background-position: 50px 80px;
```

Note how background-position accepts two values. The first sets the position along the x axis and the second sets the position along the y axis.

Save and refresh the browser to see the result.
As this is a background image you can place text or other images over the top. Switch back to your HTML document and type some text between `<div class="imgBox">` and `</div>`. Save the HTML document and refresh the browser to see the result.

**A note on background images, colours and content**

You’ve already learned how to set a background colour and now you know how to set a background image. Be very careful of how you combined background images and colours when you need to lay text content on top. The text needs to be readable, so if you’re going to use a tiled background image or a background colour make sure that text can still be read.

Don’t create a design like this:

![Background image example](image.png)

Your user needs to be able to read your content. Don’t let your design overshadow your content.
Custom bullet points
You may remember from earlier weeks that unordered lists have bullet points like this:

- This is
- An
- Unordered list

Using CSS it’s possible to redefine the <ul> tag to replace the default bullet point with an image.

You’re going to replace the default bullet with a small image of a guitar.

**Step 1**
Create a new HTML document and save it as bullets.html and create a new CSS document and save it as bullets.css.

Link the HTML document to the CSS document.

**Step 2**
Add the following to the <body> </body> section of the HTML document:

```html
<ul>
  <li>This is a list</li>
  <li>With some items</li>
  <li>and a custom image</li>
  <li>replacing the bullet</li>
</ul>
```

Save the HTML document and check it in the browser. You should see an ordinary unordered list.

**Step 3**
Switch to bullets.css and redefine the <ul> tag:

```css
ul {
  list-style-image: url(images/guitar.jpg);
}
```

Save the CSS document and refresh the browser. The default bullet should now be replaced by the guitar image.

This is a useful technique to deepen the connection between the content and the design.
CSS tricks
CSS has recently been improved with some new properties. As I mentioned earlier these aren’t supported by older browsers.

Also, just to complicate matters there are different ways of achieving these new things depending on which browser you are using. We’ll show you how to implement these properties in Firefox and Webkit browsers such as Safari and Chrome.

We’ll continue to use the bullets.html and bullets.css files for this exercise.

CSS rotation

Step 1
Switch to your HTML document and nest the unordered list INSIDE a <div> with the class transform.

If you’re unsure how to do this, ask your tutor for assistance, but by now, you should have enough grasp of HTML to achieve this.

Save your HTML document.

Step 2
Add the following class to your CSS document:

```css
.transform {
    width: 400px;
    height: auto;
    background-color:#99CC99;
    -moz-transform: rotate(-5deg);
    -webkit-transform: rotate(-5deg);
}
```

Save your CSS document and refresh the browser to see your box slightly rotated counter-clockwise.

Have a play around by changing the -5 value to whatever you want. The value works in degrees so any value between 0 to 360 (for clockwise rotation) and 0 to -360 (counter-clockwise rotation) is accepted.
CSS text shadows
The new text-shadow property allows you to create a text shadow, i.e. a slightly moved, slightly blurry copy of some text.

It is useful as previously the only way to do this was to render the text as an image and include it in your Web page, but whilst it can add a nice effect to your page, be careful not to overdo it.

Step 1
Underneath the box with the bullet points (after closing </div>) create a <p> </p> with some text in it.

Save your HTML document.

Step 2
Add the following to your CSS document:

```
p {
    text-shadow: #6374AB 20px 12px 2px;
}
```

Save your CSS document and refresh the browser and you should see something like this:

You can see the text shadow appearing here.

The text-shadow property accepts a number of values. It’s good for you to know how to control it:

```
text-shadow: #6374AB 20px 12px 2px;
```

#6374AB is the colour of the shadow

20px sets the x position of the shadow relative to the text

12px sets the y position of the shadow relative to the text

2px sets the blur radius of the shadow – the amount of space the shadowtext is ‘stretched’, causing a blur effect. 0 means: no blur. Don’t set this value too high, the shadowtext quickly becomes illegible.

You can, of course, combine the text-shadow with the transform property and indeed any of the properties you’ve covered so far in this unit.
Week 8: Advanced styling

This week you’re going to do some advanced styling. It’s become common practice to construct Web site menu bars using unordered lists, redefining the <ul>, <li> and <a> tags.

**Step 1**
Create a new HTML document and save it in your mas240 folder as menu.html. Then create a new CSS document and save it in your mas240 folder as menu.css.

Link the HTML document to the CSS document.

**Step 2**
You’re going to create three boxes. A container box in which everything else is nested, a menu box and a content box.

You’ll also add in the basic content we need.

Add the following to your HTML document:

```html
<div class="container">
  <div class="menu">
    <ul>
      <li><a href="somepage.html">Page 1</a></li>
      <li><a href="somepage.html">Page 2</a></li>
      <li><a href="somepage.html">Page 3</a></li>
      <li><a href="somepage.html">Page 4</a></li>
    </ul>
  </div>
  <div class="content">
  </div>
</div>
```

These are just dummy names that all link to a fictitious somepage.html.

For the purposes of this week it doesn’t matter that this page doesn’t exist, we just need the hyperlinks to be active.

You’re going to populate the content box using the dummy text you used in week 6. You can copy and paste it from:

http://media.mq.edu.au/mas240/wk6-dummy-text.html
Save your HTML document and check it in the browser. You should see something like this:

Ok, so far it doesn’t look very impressive.

The first thing we’re going to style is the font because I really don’t like Times New Roman on the Web (remember this when you do your Web site assignments 😊).

**Step 3**
Switch to the CSS document and add the following:

```css
body {
    font-family: Georgia;
}
```

Font family is used to set the font. You’ll notice that I’ve created a style to redefine the `<body>` tag. This ensures that Georgia will be used for everything on the page. You can change the font for other things by setting that property for an HTML tag or a class.
Be very careful when setting fonts because if you use a non-standard font then text will display in the default Times New Roman if the user doesn’t have the font on their computer. As a general rule stick to:

Arial

“Times New Roman” – note: must be declared inside quotation marks because it’s more than one word.

Courier (although it doesn’t look very nice)

Georgia

Verdana

Step 4
Let’s create a class for the container:

```css
.container {
    width: 800px;
    height: auto;
}
```

Step 5
Now a class for the menu box:

```css
.menu {
    width: auto;
    height: 40px;
    font-size: 20px;
}
```

Note that by setting the width to auto ensures that the menu will extend to the full width of the containing element, i.e. the 800px of the container box.
**Step 6**
Now a class for the content box:

```css
.content {
    width: auto;
    height: auto;
    padding: 10px;
    border: 1px solid #CCCCCC;
}
```

Now you can save your CSS document and refresh the browser to see the changes.

**Step 7**
Now you’re going to style the menu using a series of styles to redefine HTML tags:

Switch to your CSS document and add the following style:

```css
.menu ul {
    margin: 0px;
    padding: 0px;
    list-style: none;
}
```

Save your CSS document and refresh the browser to see the changes. You’ve taken out the inbuilt margin and padding of the `<ul>` tag and removed the bullet points.

Next you’re going to redefine the `<li>` tag.

**Step 8**
Create a new style for the `<li>` tag. As with the `<ul>` tag style above, this will only affect `<li>` tags nested in the menu box.

```css
.menu li {
    float: left;
}
```
Save your CSS document and refresh the browser, you will see that the <li> elements are all sitting in a horizontal row – it’s starting to look more like a menu bar now. The next step is to redefine the <a> tag and the various states of a:link, a:visited, a:hover and a:active.

For the purposes of this week (and because it’s my personal design preference) I’m going to group the link and visited states into one style and create another for the hover and active states.

Redefining the <a> tag and the various states will involve a number of properties.

**Step 9**
Add the following styles to your CSS document:

```css
.menu a, .menu a:link, .menu a:visited {
    background-color:#CCCCFF;
}

.menu a:hover, .menu a:active {
    background-color: #CC99FF;
}
```

Again, this will only affect hyperlinks contained in the menu box.

Save your CSS document and refresh the browser. Try rolling the mouse over the links and you’ll see the background colour change.

**Step 10**
The next step is to fill out the whole of the menu bar with our links. Amend only the .menu a, .menu a:link, .menu a:visited style so it reads:

```css
.menu a, .menu a:link, .menu a:visited {
    display: block;
    width: 200px;
    height: 40px;
    text-align: center;
    background-color:#CCCCFF;
}
```
You’ve just made a number of changes. Let’s go through them in turn so you understand what’s happened:

```css
display: block;
```

Basically, this turns each of the `<a>` tags into individual boxes. This is necessary so we can give them:

```css
width: 200px;
height: 40px;
```

There are four links and the container box is 800px wide; 4 x 200px = 800px. Turning the `<a>` tags into boxes 200px x 40px means that when the mouse rolls over them, the hover state is invoked. If we had instead placed these properties in the `<li>` tags then you would only see the hover state when the mouse hovered over the hyperlinked text. This method instead expands the `<a>` tag into a box.

```css
text-align: center;
```

This aligns the text to the centre of the box.

Save your CSS document and refresh the browser to see the changes.

**Step 11**

Almost there. Just a few more adjustments and you’ll be there.

CSS allows us to use text-align to centre the text within the box, but it only works on the horizontal axis, there is no vertical align equivalent, but we can achieve this using a bit of padding.

Amend the `.menu a, .menu a:link, .menu a:visited` style so it reads:

```css
.menu a, .menu a:link, .menu a:visited {
  display: block;
  width: 200px;
  height: 31px;
  text-align: center;
  background-color: #CCCCFF;
  padding-top: 9px;
  text-decoration: none;
  color: #000000;
}
```

I’ve added some top padding (9px) and therefore have had to subtract 9 from the height (40 – 9 = 31).

Remember padding (margins and borders) adds to height and width.

How did I come up with 9px of top padding? I just played around. I usually start off using this formula:

\[(\text{height of box} – \text{font size}) / 2 = \text{top padding value}\]

Then usually I end up subtracting 1 from the top padding value and I find that’s about right.

Setting the text decoration to none removes the `<a>` tag’s default underline.

And let’s give it a different colour.
Week 9: Photoshop and Image Galleries

In this session you will learn the basics of optimising images for the Web. This is an important lesson as correctly optimised images can dramatically affect the loading speed of a Web page.

You’ll be using Photoshop to resize and save images in a Web-friendly fashion. You’ll follow this up with learning how to implement the LightBox2 image gallery.

**Why optimising images is important**

Digital cameras are great for many, many reasons, but people have a tendency to get their photos off their cameras and upload them to Web sites. Many cameras’ image quality is very high which results in large file sizes. For example, iPhone photos are usually in the range of 700kb to 1MB. Imagine if you had a dozen such images on your Web site; the user would have to load in over 10MB for one page. With many digital cameras, the file sizes are even higher in fact some high resolution (hi-res) images can be well over 10MB each!

Nobody likes to sit watching photos slowly load in on the page, this is why today you’re going to learn how to optimise images for the Web.

**Get the image**

To start off you’ll be working with the wasp image that is on the MAS240 page for this week.

Save it to your desktop.

Click it once and press Apple and i to bring up the file info.

You can see here that bigwasp.jpg is 1.2MB in file size.

That means it’d take a short time to load in if you were using it on a Web page.

If you click for more info then you’ll also see that the image is 1284px wide and 792px high. This is pretty big. Usually we don’t design Web pages for wider than 1000px (absolute maximum) because that average screen resolution is only 1024px wide.

Imagine if you had a whole gallery of images this size, it’d take ages to load in.
**Step 1**
Open Photoshop. If it’s not in your dock then check your applications folder, it’ll be in a folder called Adobe Photoshop. Be patient, Photoshop usually takes a while to open.

**Step 2**
Open bigwasps.jpg in Photoshop. You can either go File >> Open and browse to the file or you can drag the image from the desktop to the Photoshop icon in the dock.

**Step 3**
It’s really a good idea to resize your images to the size you need them to be. The `<img>` tag can accept height and width properties but these only tell the browser at what size to display the image. For example if you had:

```html
<img src="images/bigwasps.jpg" width="400" height="300" />
```

The browser would still have to load the whole 1.2MB, but would display it at the size you specified. This is why it’s best practice to resize the images to the size you require.
**Step 4**

Now your image is in Photoshop you can resize it by going Image >> Image Size to call up the Image Size dialogue box:

You can set the width and height here.

Make sure that the unit is set to pixels.

You should also ensure that the constrain proportions box is checked. This keeps your image proportionate, so if you alter the width then the height is automatically proportionally match it.

Set the width to 500 and the height should automatically set itself to 308 and then click OK.

You should now see a resized version of the image.

**Step 5**

The final step is to optimise the image for the Web.

Click File >> Save for Web & Devices. The Save for Web & Devices dialogue box offers four tabs:
Select the 4-Up tab. This shows you four different version of the image with different quality settings.

As you can see here, Photoshop gives you the file sizes of each quality setting. In this example, you can see that the newly resized image is 451kb and when the quality is set to 50 its file size is drastically reduced to the 40.5k.

The 4-Up view allows you to draw a compromise between image quality and image file size.

You can manually set the quality by selecting one of the panels and adjusting the quality slider on the right side of the box.

Once you’ve made your decision click Save and save your image.

Building an image gallery

Images galleries are a very popular feature of Web sites and there are many ways to implement them. We’ll cover more advanced gallery plugins in MAS241, but for this unit, we’ll provide you with a basic gallery plugin written especially for you.

This gallery plugin is very easy to implement. Type out the code below just above the closing </head> tag in the HTML document that will be your image gallery:

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.5.1/jquery.min.js"></script>
<script src="http://media.mq.edu.au/mas240/gallery.js"></script>
```

This ensures that all the clever coding that makes the gallery work is included in your HTML page. Now, all you need to do is add a couple of attributes into the <img /> tags of your thumbnail images:

```
<img src="images/th-image1.jpg" href="images/image1.jpg" class="gallery"/>
```

In this example, the src attribute is the path to the thumbnail image. The href attribute is the path to the full size image, and the class is set to gallery to indicate that it is part of the image gallery.

Give it a try out. There’s a zip file on the MAS240 page with some ready-made gallery images.
CSS Cheat Sheet

Unless otherwise stated, all references to value are in pixels (px).

This cheat sheet is not exhaustive but covers everything used in this manual. For a full reference please see: http://reference.sitepoint.com/css

<table>
<thead>
<tr>
<th>Backgrounds</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>background-color</td>
<td>Sets the background color of an element</td>
<td>colour code</td>
</tr>
<tr>
<td>background-image</td>
<td>Sets the background image for an element</td>
<td>url(pathToImage)</td>
</tr>
<tr>
<td>background-position</td>
<td>Sets the starting position of a background image</td>
<td>xPosition yPosition</td>
</tr>
<tr>
<td>background-repeat</td>
<td>Sets how a background image will be repeated</td>
<td>no-repeat, repeat-x, repeat-y</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dimensions</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>height</td>
<td>Sets the height of an element</td>
<td>auto, value</td>
</tr>
<tr>
<td>width</td>
<td>Sets the width of an element</td>
<td>auto, value</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Transformation</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>-moz-transform</td>
<td>Rotates the element in degree</td>
<td>rotate(value) *value is stated in degrees, e.g. -5deg</td>
</tr>
<tr>
<td>-webkit-transform</td>
<td>Rotates the element in degree</td>
<td>rotate(value) *value is stated in degrees, e.g. -5deg</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
<td>Value</td>
</tr>
<tr>
<td>----------------</td>
<td>------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>border</td>
<td>Sets all the border properties in one declaration</td>
<td>[width] value,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>[colour] colour code,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>[style] none, solid, dashed, dotted, groove, double, inset, outset, ridge</td>
</tr>
<tr>
<td>border-bottom</td>
<td>Sets all the bottom border properties in one declaration</td>
<td>[width] value,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>[colour] colour code,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>[style] none, solid, dashed, dotted, groove, double, inset, outset, ridge</td>
</tr>
<tr>
<td>border-bottom-color</td>
<td>Sets the color of the bottom border</td>
<td>colour code</td>
</tr>
<tr>
<td>border-bottom-style</td>
<td>Sets the style of the bottom border</td>
<td>none, solid, dashed, dotted, groove, double, inset, outset, ridge</td>
</tr>
<tr>
<td>border-bottom-width</td>
<td>Sets the width of the bottom border</td>
<td>value</td>
</tr>
<tr>
<td>border-color</td>
<td>Sets the color of the four borders</td>
<td>colour code</td>
</tr>
<tr>
<td>border-left</td>
<td>Sets all the left border properties in one declaration</td>
<td>[width] value,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>[colour] colour code,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>[style] none, solid, dashed, dotted, groove, double, inset, outset, ridge</td>
</tr>
<tr>
<td>border-left-color</td>
<td>Sets the color of the left border</td>
<td>colour code</td>
</tr>
<tr>
<td>border-left-style</td>
<td>Sets the style of the left border</td>
<td>none, solid, dashed, dotted, groove, double, inset, outset, ridge</td>
</tr>
<tr>
<td>border-left-width</td>
<td>Sets the width of the left border</td>
<td>value</td>
</tr>
<tr>
<td>border-right</td>
<td>Sets all the right border properties in one declaration</td>
<td>[width] value,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>[colour] colour code,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>[style] none, solid, dashed, dotted, groove, double, inset, outset, ridge</td>
</tr>
<tr>
<td>border-right-color</td>
<td>Sets the color of the right border</td>
<td>colour code</td>
</tr>
<tr>
<td>border-right-style</td>
<td>Sets the style of the right border</td>
<td>none, solid, dashed, dotted, groove, double, inset, outset, ridge</td>
</tr>
<tr>
<td>border-right-width</td>
<td>Sets the width of the right border</td>
<td>value</td>
</tr>
<tr>
<td>border-style</td>
<td>Sets the style of the four borders</td>
<td>none, solid, dashed, dotted, groove, double, inset, outset, ridge</td>
</tr>
<tr>
<td>border-top</td>
<td>Sets all the top border properties in one declaration</td>
<td>[width] value,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>[colour] colour code,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>[style] none, solid, dashed, dotted, groove, double, inset, outset, ridge</td>
</tr>
<tr>
<td>border-top-color</td>
<td>Sets the color of the top border</td>
<td>colour code</td>
</tr>
<tr>
<td>border-top-style</td>
<td>Sets the style of the top border</td>
<td>none, solid, dashed, dotted, groove, double, inset, outset, ridge</td>
</tr>
<tr>
<td>border-top-width</td>
<td>Sets the width of the top border</td>
<td>value</td>
</tr>
<tr>
<td>border-width</td>
<td>Sets the width of the four borders</td>
<td>value</td>
</tr>
</tbody>
</table>
## Font

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>font-family</td>
<td>Specifies the font family for text</td>
<td>font name</td>
</tr>
<tr>
<td>font-size</td>
<td>Specifies the font size of text</td>
<td>value</td>
</tr>
<tr>
<td>font-style</td>
<td>Specifies the font style for text</td>
<td>italic, normal, oblique</td>
</tr>
<tr>
<td>font-variant</td>
<td>Specifies whether or not a text should be displayed in a small-caps font</td>
<td>small-caps</td>
</tr>
<tr>
<td>font-weight</td>
<td>Specifies the weight of a font</td>
<td>bold, normal</td>
</tr>
</tbody>
</table>

## Margins

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>margin</td>
<td>Sets all the margin properties in one declaration</td>
<td>value</td>
</tr>
<tr>
<td>margin-bottom</td>
<td>Sets the bottom margin of an element</td>
<td>value</td>
</tr>
<tr>
<td>margin-left</td>
<td>Sets the left margin of an element</td>
<td>value</td>
</tr>
<tr>
<td>margin-right</td>
<td>Sets the right margin of an element</td>
<td>value</td>
</tr>
<tr>
<td>margin-top</td>
<td>Sets the top margin of an element</td>
<td>value</td>
</tr>
</tbody>
</table>

## Padding

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>padding</td>
<td>Sets all the padding properties in one declaration</td>
<td>value</td>
</tr>
<tr>
<td>padding-bottom</td>
<td>Sets the bottom padding of an element</td>
<td>value</td>
</tr>
<tr>
<td>padding-left</td>
<td>Sets the left padding of an element</td>
<td>value</td>
</tr>
<tr>
<td>padding-right</td>
<td>Sets the right padding of an element</td>
<td>value</td>
</tr>
<tr>
<td>padding-top</td>
<td>Sets the top padding of an element</td>
<td>value</td>
</tr>
</tbody>
</table>
### Positioning

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>clear</td>
<td>Specifies which sides of an element where other floating elements are not allowed</td>
<td>none, both, left, right</td>
</tr>
<tr>
<td>float</td>
<td>Specifies whether or not a box should float</td>
<td>none, left, right</td>
</tr>
</tbody>
</table>

### Text

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>color</td>
<td>Sets the color of text</td>
<td>colour code</td>
</tr>
<tr>
<td>direction</td>
<td>Specifies the text direction/writing direction</td>
<td>ltr, rtl</td>
</tr>
<tr>
<td>letter-spacing</td>
<td>Increases or decreases the space between characters in a text</td>
<td>value</td>
</tr>
<tr>
<td>line-height</td>
<td>Sets the line height</td>
<td>value</td>
</tr>
<tr>
<td>text-align</td>
<td>Specifies the horizontal alignment of text</td>
<td>center, left, right</td>
</tr>
<tr>
<td>text-decoration</td>
<td>Specifies the decoration added to text</td>
<td>blink, line-through, overline, underline, none</td>
</tr>
<tr>
<td>text-indent</td>
<td>Specifies the indentation of the first line in a text-block</td>
<td>value</td>
</tr>
<tr>
<td>text-shadow</td>
<td>Specifies the shadow effect added to text</td>
<td>[colour] colour code, [xPosition] value, [yPosition] value, [blur] value</td>
</tr>
<tr>
<td>text-transform</td>
<td>Controls the capitalization of text</td>
<td>capitalize, lowercase, none, uppercase</td>
</tr>
<tr>
<td>word-spacing</td>
<td>Increases or decreases the space between words in a text</td>
<td>value</td>
</tr>
</tbody>
</table>

### Lists

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>list-style-image</td>
<td>Specifies an image as the list-item marker</td>
<td>none, url(pathToImage)</td>
</tr>
<tr>
<td>list-style-type</td>
<td>Specifies the type of list-item marker</td>
<td>circle, square</td>
</tr>
</tbody>
</table>