Getting Started With Website Design

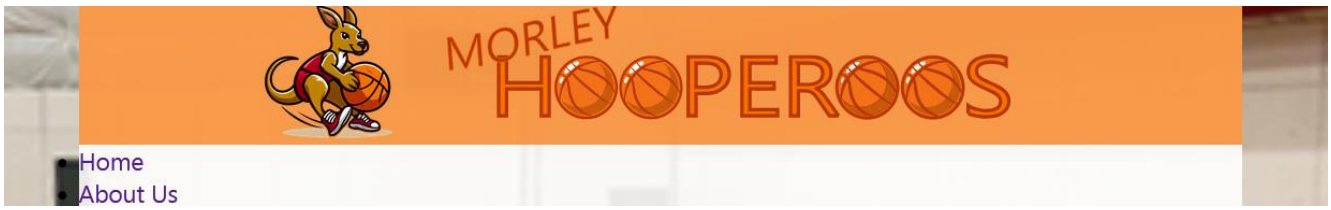
5 – More CSS Layout

We've made some adjustments to our typography as well as setting some styles for our main section and the page in general. Now we'll add some style rules to format / layout the other main sections of the page.

Exercise 1 – Formatting the Header Section

1. Add the following css rule to the header section in our css file.

```
/* Header */
header {
  background-color: rgba(248,114,2,0.70);
  text-align: center;
}
```



Exercise 2 – Formatting the Footer Section

1. Add the following css rule to the footer section in our css file.

```
footer {
  clear: both;
  font-size: 0.8em;
  text-align: center;
  background-color: rgba(248,114,2,0.70);
}
```

Exercise 3 – Formatting the Navigation Section

For the navigation section we currently have a pretty plain looking unordered list. We are going to use a combination of css properties to make the unordered list look more like a proper navigation bar with navigation buttons.

1. Add some navigation rules in the navigation section of the css as follows.

```
nav ul {                    this formatting will only apply to unordered lists inside the nav section
  margin: 0;                removes margin space outside our list
  padding: 0;               removes padding space inside the edge of our list
  list-style-type: none;    displays the list without dot points
  display: flex;            creates a css flexbox that our list items will be arranged in
}
nav li {                    this formatting will only apply to list items that are inside the nav section
  font-size:1.9rem;         text size of the nav buttons
  display:inline;           list items will be inline text rather than blocks of text
  text-align:center;        text will be in the centre of each nav button
  flex:1;                   each list item will be equal size
}
nav a {                     this formatting will only apply to links that are inside the nav section
  padding: 3px 0;           each link will have padding space above and below but no side padding
  background-color: #B10022; link background colour
  color: white;             white text
  display: block;           Links will be formatted as blocks rather than inline text
  font-weight: bolder;
  border: 3px outset;
  text-decoration: none;
}
nav a:hover {               this formatting will only apply to hover links that are inside the nav section
  background-color: #F87202;
  color: #B10022;
  cursor: pointer;          This will make the mouse pointer change to a hand shape
  text-decoration: none;
}
```

Save and preview the pages to test the navigation bar. The list will now use the css flex layout to make the buttons sit next to each other rather than the list's normal vertical layout. The remaining formatting will make the normal links appear like buttons.

CSS Shorthand

Remember that according to the CSS box model, each block element on the page can have padding, border and margins. Each of these can be on the top, right, bottom or left of a shape. You can set each of these individually but css also allows for shorthand, allowing you to set properties in a simpler way. Look at the following examples.

Setting width (thickness), style and colour of a border all at once

This	Can also be written as
border-width: thin; **border-style: solid;** **border-color: black;**	**border: thin solid black;**

When it comes to borders, padding and margins, we can set each side individually or we can set them together. You will have already seen some examples in the css we have already added.

This	Can also be written as
margin-top: 4px; **margin-righ: 2px;** **margin-bottom: 3px;** **margin-left: 5px;**	**margin: 4px 2px 3px 5px;**

If 4 measurements are given then they are for top, right, bottom, left.

If 2 measurements are given, they are for top/bottom, left/right.

If one measurement is given it sets all sides equally.

Shorthand can also be used for things like font formatting.

This	Can also be written as
font-style: italic; **font-variant: small-caps;** **font-weight: bold;** **font-size: 12pt;** **line-height: 20pt;** **font-family: Arial, sans-serif;**	**font: italic small-caps bold 12pt/20pt Arial, sans-serif;**

Exercise 4 – Additional Page Formatting

At the moment the text on the page might be sitting right up against the edge of the margins so we will add a bit of padding to the side of the main section.

1. Add a selector for the main element in the css as follows.

```
/* Main */

main {

  padding: 0 3px;

}
```

This will add zero padding to the top and bottom of the main section while adding 3 pixels padding on either side.

Notice that on pages that have a bullet list (unordered list) the bullets themselves sit outside the main area. This is because of our fist lines of CSS that remove the default margin around each of our elements. List items usually have a left margin to allow room for the bullet or number. The DD part of a definition list also usually has a left margin. We will add those left margins to list items and definition data items that are within the main part of the page (we don't want it to apply to the list items elsewhere such as the navigation area).

2. Add the following rule in the main part of your CSS.

```
main li, main dd {

        margin-left: 1em;

}
```

List items should now have enough space on the left to accommodate the bullet point.

On the memberships page we have an aside element following the first paragraph.

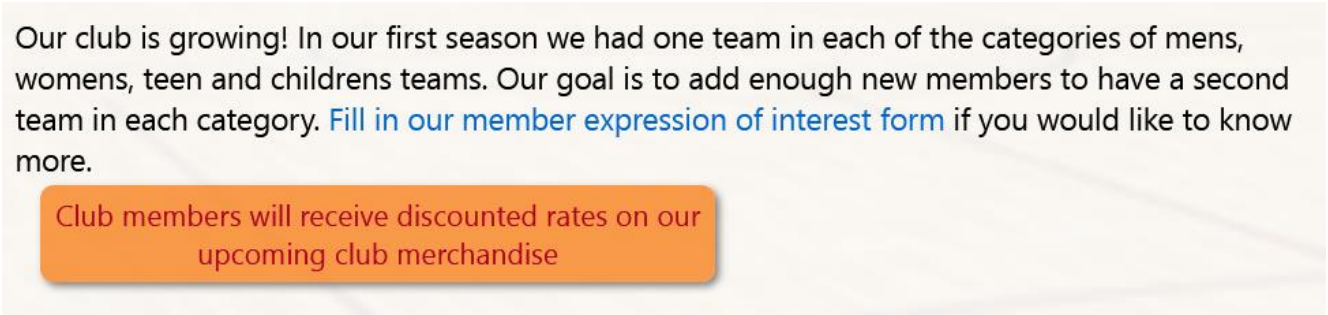
```
<aside>Club members will receive discounted rates on our upcoming club merchandise</aside>
```

We'll add a CSS rule to format this as well as any additional asides that we add to our site.

3. Add a css rule like the following in the main section.

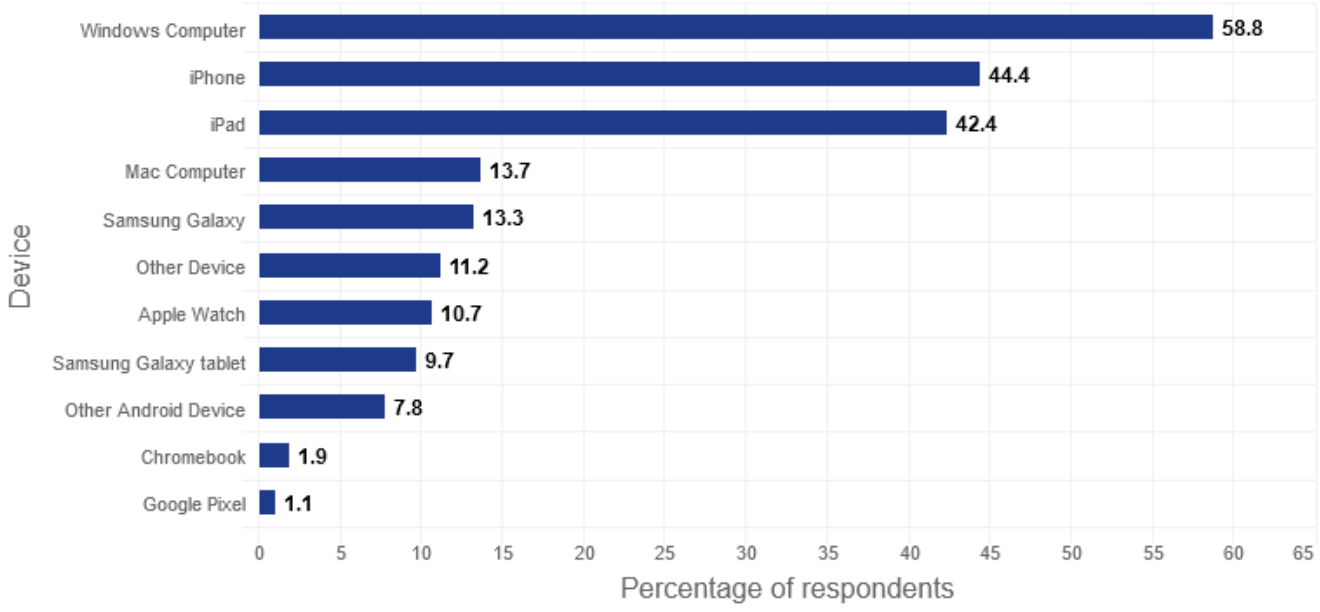
```
aside {
 background-color: rgba(248,114,2,0.70);
 color: #B10022;
 font-size: 1.7rem;
 padding: 6px;
 margin-left: 20px;
 text-align: center;
 width: 25vw;
 border-radius: 8px;
 box-shadow: 2px 2px 4px gray;
}
```

4. Save and preview the changes.



Responsive Design

When the World Wide Web began, the vast majority of people using it were using desktop computers. In recent years, that has no longer been the case as more and more people use mobile devices such as phones and tablets to access the web. The chart below shows devices that were used to access the Internet in Australia[1] based on recent information.



[1] Source - https://www.visionaustralia.org/

It is no longer sufficient to simply design websites to work well on desktop screens when so many people viewing your website will be viewing it from a smaller screen on a mobile device. Not only does this effect the size your page will be viewed at. It also makes a difference to how people will use your site. Some web-based features that relied on users with a computer mouse have fallen out of favour as more and more people use touch screens to access websites.

There are several approaches to making your web pages responsive. In our web page, we will make some adjustments to our css that will make the pages adjust to different screen sizes. Some of our css has already been intended to suit different screen sizes, such as setting our body width with both % with and max width properties.

Media Rules

In CSS we can create media rules. These allow you to set a group of CSS rules that will only apply when the screen being used to view your website is a certain size. For example, you could create a media rule with css that applies to screen widths less than 1000 pixels (such as tablets) and another media rule with css that applies to screens widths less than 400 pixels (such as phones).

In our website, we will add a media rule containing css rules intended for users viewing our web page on a mobile device.

Exercise 5 – Adding a Media Rule

1. Add the following at the end of your css file. Any CSS rules we place in here will only apply to people viewing the site from a screen that is no wider than 1100 pixels and no smaller than 800 high. This should include most mobile phone screens where the screen is usually much taller than it is wide. These rules will over-ride any similar rules that have already been defined.

```
/* Smaller Screens */

@media screen and (max-width: 1100px) and (min-height: 800px) {


}
```

We will add a new rule for the body selector. There is already a body rule near the top of our css file that applies to our whole document. Where there are conflicts between the first one and the second one we will put in this section, this one will be the one that is used for smaller screens.

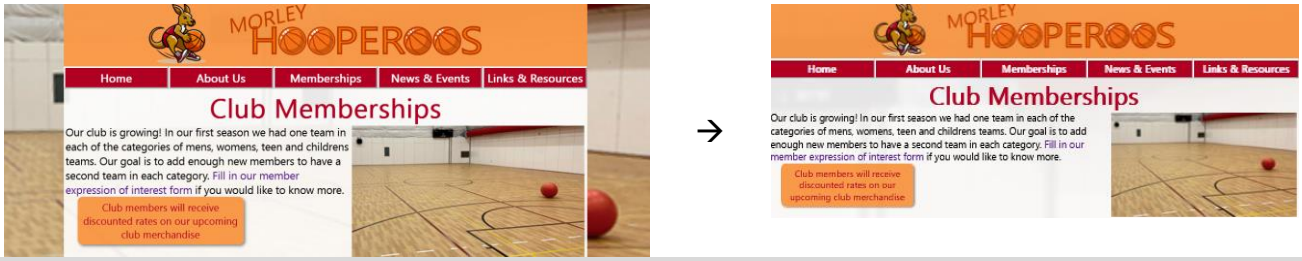
2. Add a new css rule in the media section like the following.

```
@media screen and (max-width: 1100px) and (min-height: 800px) {
        body  {
                width:100%;
        }
}
```

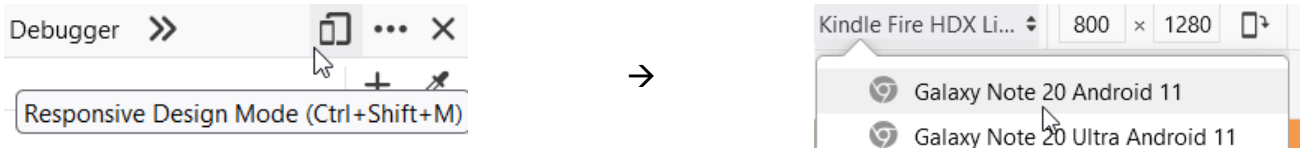
Now on smaller screens, the body will take up the whole width when the web page is being viewed in a smaller area. Notice that we have indented this rule to indicate that it is underneath the media rule.

3. Save and preview the changes.

When your web browsers is a larger size you will see no difference. If you resize your browser window, then when the viewing size is fits the dimensions specified in our media rule, the body will adjust to fit the whole width with no space on the side margins.



Tip In most modern web browsers you can access *developer tools*. Usually by **right-clicking** on a page and selecting **Inspect** or using the **Ctrl Shift I** shortcut. These developer tools (which will normally appear at the bottom or side of your window), will typically have an option that allows you to preview the page the way it would look on different devices. This is useful for testing CSS changes on different screen sizes.



The font sizes in our website might be fine for reading on a large desktop screen but would be too small for reading on a smaller phone screen. We will add some rules in the media section with larger font sizes that will apply to small screens.

4. Add the following css rules. Make sure these are all inside the media section.

```
body  {
  width:100%;
}
h1 {
  font-size: 2em;
}
h2 {
  font-size: 1.7em;
}
h3 {
  font-size: 1.4em;
}
main {
  font-size: 3rem;
}
```

These will all increase the size of text when the screen is smaller.

5. Save and preview the page at different screen sizes. Adjust the sizes if you prefer.

Exercise 6 – Responsive Navigation

At the moment our navigation is fine on larger screens, but on smaller screens it looks small and would be difficult to use on a touch screen. We will use css in our media section to change it so that on smaller screens the buttons will be larger and stacked vertically instead of side by side.

1. Add the following rules in the media section.

```
nav ul {
  display: block;          The list will now display as blocks of text rather than flex items
}
nav li {
  width: 100%;             Items in the list will now fit the whole width
}
nav a {
  font-size: 1.6em;        Text for list items in the navigation will be larger
}
```

2. Save and preview the changes.

Exercise 7 – Additional Smaller Screen Adjustments

1. Add the following rules to the media section of the css.

```
aside {
  font-size: 2.2rem;
  width: 90%;
}
#main_image {
  width: 70%;
  display: block;
  margin: auto;
  float: none;
}
```

2. **Add the following line to the media section to hide the footer section of the page on smaller screens**

```
footer {
  display: none;
}
```

3. Save and preview changes.
4. Use what you have learned to adjust the rules you have created and even add some additional rules until you like the way the page looks.

Take a look at the news page. At the moment we have used an inline style to set the column colours. That would mean that if we wanted to change the colour of those columns later, we would need to change each one. And if we made a new table, we would have to add the column colour as well.

5. See if you can modify your tables in the news page so that if you need to change the column colour later, you can just do it from one place in your css file (hint – you could create a new class in your css class with the background colour and then replace the inline style in the html with an attribute that refers to your new class).