Lab 6 – CSS Responsive Layout and Accessibility

Aims:

- To learn how to specify and apply different CSS rules for different devices, e.g., desktops and mobile phones;
- To get familiar with the Web Accessibility Initiative;
- To learn how to test web pages for compliance and improve the quality of web pages and introduce some assistive technologies.

Task 1: CSS Responsive Layout (2 Marks)

We are going to create two CSS styles for one registration form, one will be used for a desktop device and the other for a mobile phone.

Step 1:

1.1 Create a new folder 'lab06' under the unit folder on the mercury server. Upload today's work to this lab06 folder.

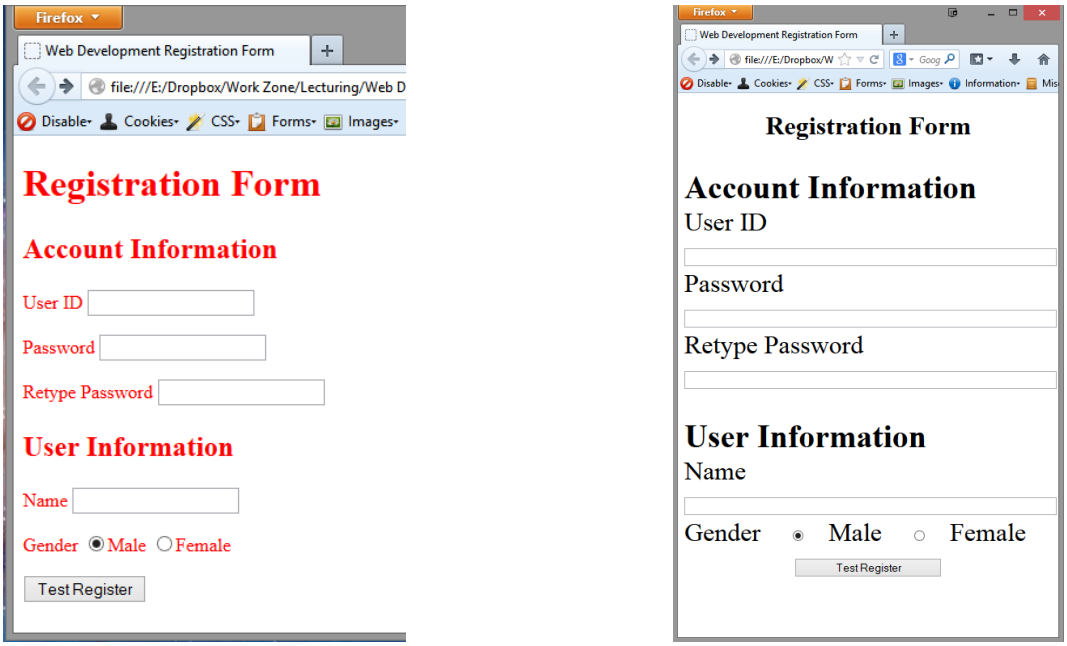
Step 2:

2.1 Download and open the text file regform.html.

2.2 Use NotePad++ (or Sublime Text for Mac users) to open and edit regform.html. Changes will need to be made in order to allow for CSS styling.

Step 3:

3.1 Design a mock up on what the form should look like in a desktop and a mobile phone. Figure 1 presents a possible mock up for both desktop and mobile phone.



a) Desktop Version **b) Mobile Version**

Figure 1: Sample Form Layouts

Step 4:

4.1 Using NotePad++ (or Sublime Text for Mac users), create two CSS files, desktop.css and mobile.css, that will present the forms shown in Figure 1.

4.2 To apply the two CSS files to regform.html, you need to add the following code to the <head> part of regform.html.

```
<link href="desktop.css" rel="stylesheet" media="screen and (min-width:768px)" />
<link href="mobile.css" rel="stylesheet" media="screen and (max-width:480px)" />
```

By doing so, desktop.css will be applied to the regform.html when the width of the browser displaying regform.html is greater than 768 pixels, which is the usually the minimum width of a PC monitor. If the width of the browser is lower than 480 pixels, mobile.css will be applied to regform.html, displaying the web page in a mobile-friendly manner, e.g., larger font and compact layout.

Step 5:

5.1 Specify only one CSS rule in desktop.css:

```
body      {
  color: red;
          }
```

Using this CSS rule, when all the text on the web page are rendered red (see Figure 1a), we know *desktop.css* is taking effect, not *mobile.css*.

5.2 Open *mobile.css*. Following the comments below, complete and apply CSS rules to present *regform.html* as shown in Figure 1b. You might need to specify the id or the class attributes of applicable HTML elements so that they can be properly selected in the CSS file for CSS application.

[IMPORTANT] Complete one CSS rule at a time and test the webpage to understand the effect of that CSS rule. If you cannot see a difference when testing your webpage, try reducing the width of your browser window.

When does mobile.css kick in? Why?

```
      _____ {
        _________:_________;          /* Increase the font size in the form to 200% */
}

{
_________:_________;                  /* Center the text for all <h1> elements */
}

_____ {
_________:_________;                  /* Change the font size of <h2> to 40 pixels  */
_________:_________;                  /* Remove the bottom margin of <h2>  */
}

_____ {
   _________:_________;               /* Make all the <input> elements occupy the entire width  */
}

_____ {
_________:_________;                  /* Remove all margins around all <p> elements */
```

```
}

_____ {
_______:_________;              /* Set width of all radio buttons to 15% */
}

_____ {
_______:_________;              /* Set width of all buttons to 40% */
}

_____ {
_______:_________;              /*Center the test button by centering the appropriate <div> */
}
```

Step 6:

Merge desktop.css and mobile.css into one CSS file named style.css.

6.1 Create a file named style.css.

6.2 Add the following CSS rule to style.css:

```
@media screen and (min-width: 1024px) {
        body {
                font-size: 100%;
      }
}
```

6.3 Apply the media query below to all your code from **Step 5.2**:

```
@media screen and (max-width: 480px) {
…
…
}
```

6.4 Remove the following code from regform.html.

```
<link href="desktop.css" rel="stylesheet" media="screen and (min-width:768px)" />
<link href="mobile.css" rel="stylesheet" media="screen and (max-width:480px)" />
```

6.5 Add the following code to the <head> part of regform.html.

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

Now only one CSS file is applied to regform.html. Based on the width of the browser window, different CSS rules will kick in and present regform.html in two different ways. Test it and see for yourselves.

Step 7. Test and view web pages.

7.1 Using WinSCP, upload your files, including regform.html, desktop.css, mobile.css and style.css onto Mercury.

7.2 Now you can even use your mobile phone to test your webpages.

7.3 To view the pages through http, use any Web browser and type in the following address,

http://mercury.ict.swin.edu.au/<your unit code>/*s<your Swinburne ID>*/<folder>/<filename>

Please refer to the following examples to identify the URLs of your web pages.

Folder on Mercury Web Server	URL
~/cos10024/www/htdocs/index.html	http://mercury.swin.edu.au/cos10024/s1234567/index.html
~/cos60002/www/htdocs/lab06/regform.html	http://mercury.swin.edu.au/cos60002/s1234567/lab06/regform.html

Note: You can copy the URLs in the table, but remember to replace the unit codes and student id in the above examples with yours to obtain the URLs of your web pages on Mercury.

[IMPORTANT] When the browser authorization request dialog pops up, use your SIMS username *and* password to confirm access, NOT your mercury username and password.

Step 8: HTML and CSS Validation

To validate the HTML file, either add in the Web Developer toolbar and use 'Tools'/ 'Validate Local HTML' or use the validator at http://validator.w3.org and for webpages pages on the server validate via 'URL'.

To validate the CSS file, either use the Web Developer toolbar and use 'Tools'/ 'Validate Local CSS' or use the CSS validator at http://jigsaw.w3.org/css-validator/ and for CSS on the server validate by 'URL'.