Web Development Lab – week 7

# CSS Units and Responsive Design

## Step 1:

Download the Lab7Starter ZIP file from Webcourses (Lab Work Week 7).

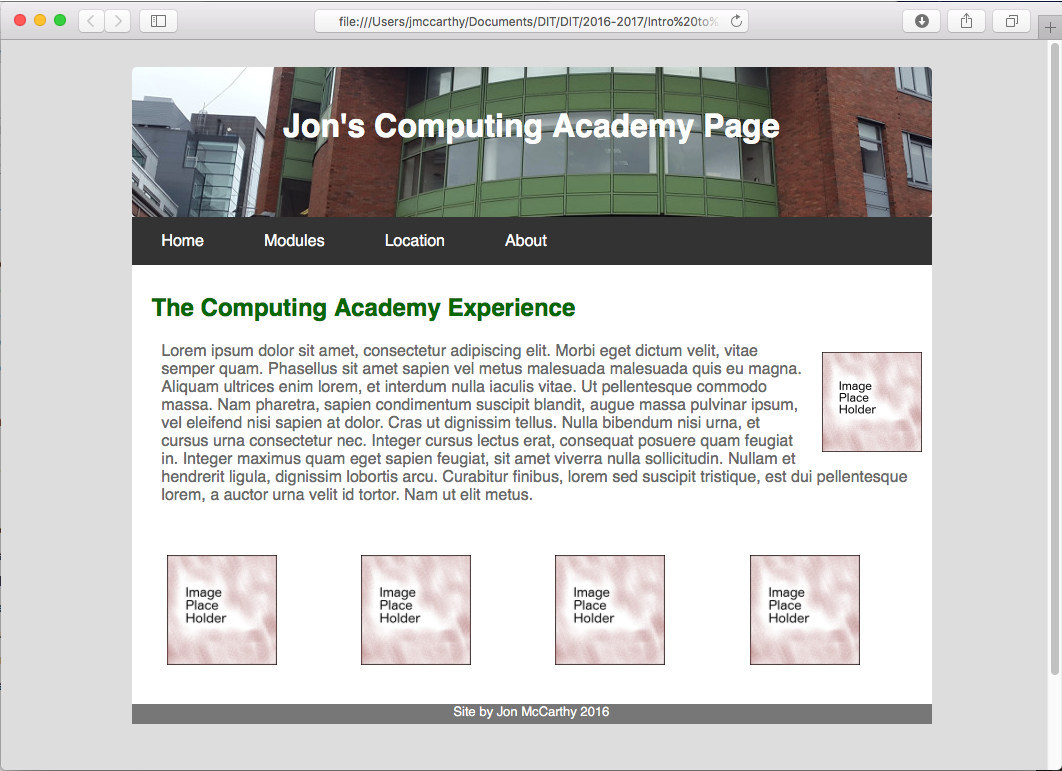
## Step 2:

Extract the contents of the folder.

If contains a index.html file and site.css file.

## Step 3:

View the index.html page. If should look as follows:



The basic layout and look of the page is OK, but it is not very responsive.

If the page is viewed in a smaller browser the page will not scale to match the screensize.

Try it: make your browser window smaller and see how the page displays!!

## Step 4:

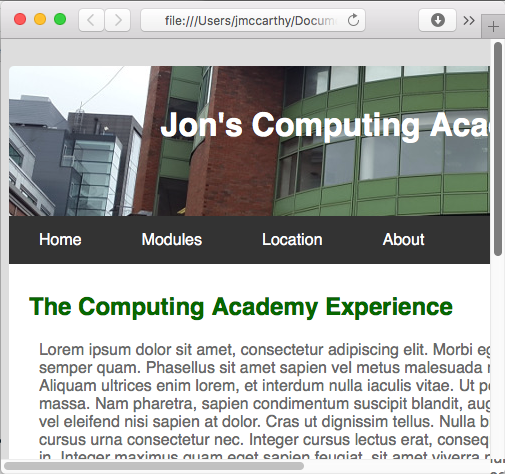
First we will try make the page responsive using CSS Units.

Have a look at the CSS file and try flag any potential issues you see (we will fix these issues in the following steps….)

## Step 5:

First we will address the width of the content with respect to the browser width.

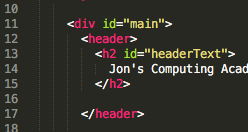
If the browser size is reduced the page displays as follows:



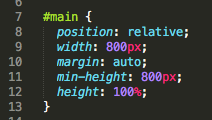
The main container for the page needs to be dynamic with respect to the browser width.

Percentages would be a good solution for this!!

Have a look at the HTML file. All the content for display on the page is contained within a div with an id of main.



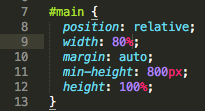
In the CSS file the CSS rules for main are as follows:



The width of the container has been set to 800px. This is a fixed value and the page will always display at 800px, wo matter how wide the browser is.

If this is changed to a percentage value the container should size with respect to the browser width and then should be always visible on the page.

Update the height to look as follows:



This should allow the main container to scale and always display on the page. Reload the page and see the difference!!

## Step 6:

The page is now somewhat dynamic, and will render reasonably well in a variety of different browsers. Lets try apply some better CSS Units to the page content.

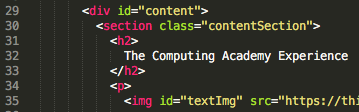
## Step 7:

The image that is to the right of the text needs attention.

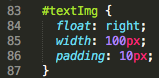
It is always the same size, no matter how big or small the browser is.

Lets try find this element in the html and css and update its style rule.

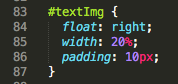
HTML



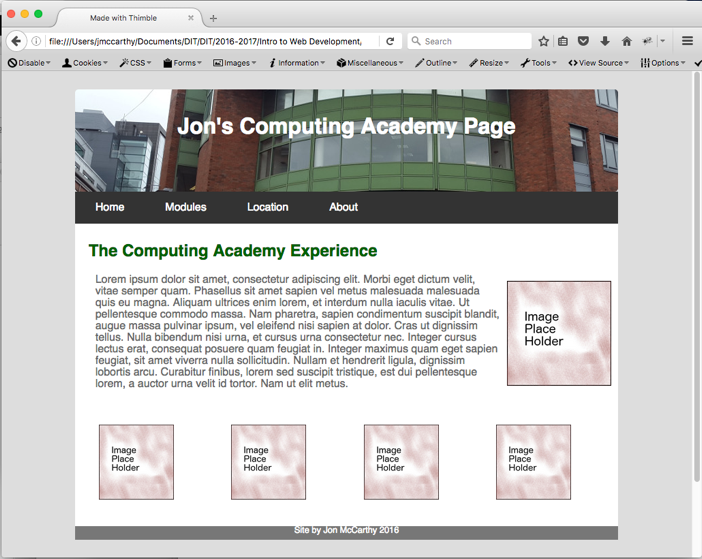
CSS



The pain issue we have with this image is its style rule has a width of 100px. This is a fixed value and the image will always display at this size. In certain circumstances this might be how we want the image to display, but no here. Lets update the width property to use percentages. Set the width to 20%:



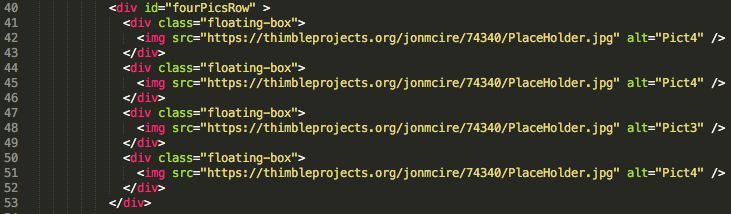
The page should now look as follows:



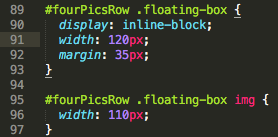
The image to the right of the text should increase in size for bigger browser widths.

## Step 8:

The next set of elements to be addressed are the block of images:

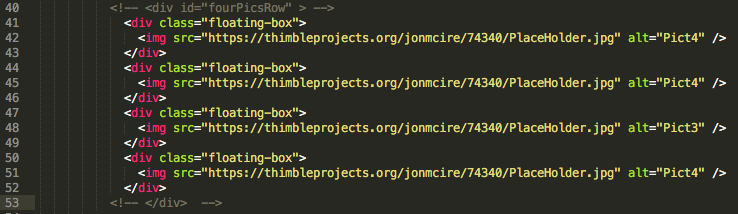


CSS



The main issue here is that the width of the container and the images all have fixed pixel values. Update the html and css for the block to look as follows

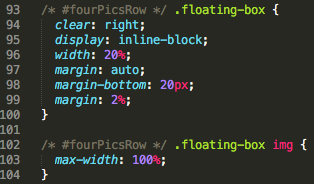
HTML



Comment out or remove the div with an ID of rourPicsRow.

In the example above the div has been commented out with <!—comment -->

CSS



In the CSS code, update the CSS rules to look as follows.

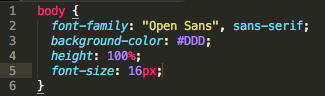
In the example above the div has been commented out with /\* comment \*/

See how dynamic the page is now when the browser width changes.

## Step 9:

The page is starting to scale well, but it would be nice if the text displayed smaller on smaller sized browsers (Phones and Tablets etc…)

Set the body’s font-size to 16px, we will use media queries to control this later.



## Step 10:

Next we will size the text on the page with respect to the font size of th body, which is our root element. We will use the rem css font.



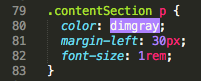
For the page header text, we are using 2rem as the font size, this will currently be 2 \* 16px = 32px, as the font of the root(body) has a size of 16px.

Set the padding top to 3vh (see notes for description on vh)

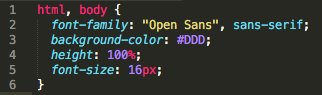
Change the font color to yellow.

## Step 11:

Update the paragraph text to have a font size of 1 rem



We must update the body tag styling to incorporate the html root tag.



## Step 12:

Add the following media queries to the bottom of the CSS file:

You should be able to cut and paste the text from this example.

/\* Wide Screens \*/

@media only screen and (max-width : 1200px) {

html, body {

font-family: "Open Sans", sans-serif;

background-color: #DDD;

height: 100%;

font-size: 20px;

}

}

/\* Desktops \*/

@media only screen and (max-width : 992px) {

html, body {

font-family: "Open Sans", sans-serif;

background-color: #DDD;

height: 100%;

font-size: 16px;

}

}

/\* Tablets \*/

@media only screen and (max-width : 768px) {

html, body {

font-family: "Open Sans", sans-serif;

background-color: #DDD;

height: 100%;

font-size: 12px;

}

}

/\* Phones \*/

@media only screen and (max-width : 480px) {

html, body {

font-family: "Open Sans", sans-serif;

background-color: #DDD;

height: 100%;

font-size: 10px;

}

}

/\* Custom \*/

@media only screen and (max-width : 320px) {

html, body {

font-family: "Open Sans", sans-serif;

background-color: #DDD;

height: 100%;

font-size: 8px;

}

}

As the page width (browser) changes, watch the different media query rules change the size of the text.

## Step 13:

Try use the media queries to change the other content on the page for different page sizes.

## Step 14:

Create a new Thimble project and add the site content above. Publish your site and open it on a phone, tablet etc. How does it display?

## Step 15:

Try make one of your previous lab pages responsive.