Making a Digital Project Responsive

In today’s modern web industry we have to think about the different devices that access the internet. In the past ten years we’ve seen web consumption go from traditional desktops and laptops to smart phones, tablets, smart watches and more. Thus, in order to make sure that our design can adjust for multiple devices. Responsive web design is the practice of making our designs adjust across multiple platforms.

Overview

In this lab we will be taking the website for Brunch and Munch and making it responsive. When thinking about a design that goes across different devices there are many things to consider. How much space will there be for columns? Are people using a mouse or touch interface with their finger like on an tablet or smartphone? Is there device going to be in landscape or portrait mode? Can all of the information fit into the device? Is all the information necessary?

Take a moment to look over the mockup of the design. You’ll see that there are major sections such as a logo section, article section and social media section. You’ll also see how the design changes or breaks depending on the width of the webpage.

When the design is above 955px in width, which is optimal for desktop and laptop browsers the design has some areas with three columns and some with two.

As the width lowers to between and 480px we see that eating categories go from three to two columns. We also get the inclusion of a menu icon instead of the social media navigation bar.

As the device gets below 480px or a size optimal for smartphones you'll see that the design goes from two columns to one column. You'll also notice that unnecessary article images are removed. You'll also see that links are made more touch friendly by being made larger.

HTML Overview and Font Icon

Take a moment to look over Brunch and Munch HTML and CSS. You'll notice that each major part of the design is divided into *<section>* tags. You'll also see that icon images have been used for the social media bar. We are going to use a font icon Font Awesome to add in icons. The reason is it will make adding CSS to the icons much easier.

Add the following code into your *<head>* tag.

```
<link rel="stylesheet" href="https://cdnjs.cloudflare.
com/ajax/libs/font-awesome/4.7.0/css/font-awesome.
min.css">
```

This is link to a content delivery network or CDN. The CDN's job is to host the CSS for other developers such as yourself. Now that we are linked to it we can use the CSS from the stylesheet.

Begin to replace the PNG icons with their font awesome equivalents you can look up the CSS from the Font Awesome website, https://fontawe-some.com/?from=io.

Replace the

- Facebook
- Twitter
- Pinterest
- Instagram
- Youtube

Icons with the font awesome CSS code.

Responsive Text

One way to make your design more responsive is to use more fluid measurements. This means not using finite values like pixels or inches, but things like percentages instead. For text, instead of us-ing the finite value of points we'll use something called an 'em'.

An 'em' pronounced 'M' is the size of the default text of the device. So for instance, if the default text size is 12pt, than 1em = 12pt. If the default text size for the device is 16pt, then 1em = 16pt. This system makes sure that text can scale in proportion.

To figure out an em use the formula

Target
------   = em
Context

Target is what you measurement the text to be in points. The context is what you *think* the default size is for the average person. For this lab we'll say we think the default text size is 14pt. Let's measure out the size of the text in the <body>

```
Body {
Font-size: 14pt;
Line-height: 20pt;
}
```

14pt is the target and the context is 14pt, thus 14/14 = 1em, or our font-size will be 1em. Let's apply this to the line-height. 20pt is the value and the target is 14pt. 20pt/14pt = 1.4em.

Go through the rest of the CSS and replace all measurements for type to ems.

Add Responsive CSS

Believe it or not you can have more than one CSS document affect an HTML document. We are going to add additional CSS that is related to responsiveness. To do this let's create a new CSS document in our coding program called *responsive.css*. Add this CSS document in the <head> section using the <link> as per usual.

Add a Viewport

Essentially the viewport is the user's visible area of a web page. The viewport varies with the device, and will be smaller on a mobile phone than on a computer screen. When we started surfing the internet using tablets and mobile phones, fixed size web

pages were too large to fit the viewport. To fix this, browsers on those devices scaled down the entire web page to fit the screen.
This is most of the time not ideal, so we need to fix this. To fix this add the following code into the *<head>* part of the document.

```
<meta name=”viewport” content=”width=device-width, ini-
tial-scale=1.0”>
```

This code states that the page will be rendered ‘as is’ and won’t be scaled up or down.

Responsive Grids

If you open *brunch-munch.html* and open it into a browser and make the browser smaller you notice that as the browser gets shorter there is just too much content for the browser. We’ll need a way to reorganize it, or in otherwords, we’ll need to change the design of the website, but only if the browser gets below a certain size.

Go to your responsive.css document and add the following:

```
@media screen and (min-width:480px) and (max-width:955px){
}
```

This is called a media query and it tells the brows-er two things. The *@media* screen tells the browser what CSS to use depending on the type of device that is reading the webpage. For instance you can have different CSS for a printer than you do for a web browser.
@media screen means if the HTML page is being

viewed on a computer screen like a Desktop computer or smartphone. In this class we will only deal with @media screen.

Min-width:480px and max-width:955px are perimeters of WHEN the CSS will be applied. In this case it means ‘the following CSS will be applied WHEN the webpage is ABOVE 480px in width BELOW 956px in width. As soon as webpage falls above or below that range the CSS will no longer apply to the HTML page.

What we want to change for this range is the dining options to go from three columns to two columns.

If you look at the /* === layout === */ section you’ll see that the <main> is divided into a total of 6 columns and the #brunch, #dinner, and #drinks each take up a 1/3rd of the width or two columns each. Let’s set it that between min-width:480 and max-width:955 that #brunch and #dinner are three columns each and that #drinks is six columns. So, it should look something like this:

```
#brunch {
Grid-column: ¼;
}
#dinner {
Grid-column: 4/7;
}
#drinks {
Grid-column: 1/7;
}
```

Save and test it, you should see a switch from three columns to two columns when the page gets below 955px.

Add a Menu Icon

With responsive design sometimes we may need user interface elements, but only at certain sizes. For instance, at smaller screen sizes we may want to add menus that allow the user to add and subtract content from the webpage. This is usually done with a menu icon.

We will add the menu icon inside the #social-media section but outside of the #social-list. Add the following Font Awesome icon:

```
<i class=”fa fa-navicon” id=”menu”></i>
```

Note that we added an id to make it easy to call in the CSS. If you preview you’ll see both the social media list and the menu icon. To only show one we’ll use the display:none CSS option. display: none / display: inherit allows you to turn off and on certain HTML elements, like for instance the social media list.

Let’s turn off the #social-media list but only for sizes between 480px and 955px. So add the following:

```
#social-media {
display: none;
}
```

This now creates a new issue though. The menu icon appears at large screen sizes above 955px.

So to fix that let’s add a section @media for screen sizes ABOVE 955px.

```
@media screen and (min-width: 955px){ }
```

Set the #menu to display:none for browser above 955px.