Web Design and Publishing - Responsive Web Design

How to build a responsive website

1. Preview the current state of your website on a mobile device and Chrome's developer tools.

a. If you have a mobile device, open your mobile browser and enter your URL.

b. Next, open your site on your desktop in Chrome. Right click on your page and select *Inspect Element*. This will open Chrome's Developer Tools. Much like opening the hood of your car shows you how your car is built, this panel displays the inner-workings of your website. There are many features in this tool, but we'll focus on just a few.

c. One feature relevant to today's lesson is the device toolbar. Click on the icon in the upper left corner of the Developer Tools, which looks like a phone and a tablet.

d. Toggle the different viewports in this view. If you click the *Responsive* option and move the side of the panel in and out, you can get an idea how the pixel width varies as your browser shrinks and expands. This will be helpful when we discuss CSS media queries later.

2. Return to your index.html and other html files in your project directories. Enter the following code in the `<head>` sections of your files.

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

3. Refresh your browser and view your site in the device toolbar again with a mobile device selected.

Do you notice a change? Your site should appear to be more zoomed and show less of your content. Are you able to scroll right on your site? If so we'll need to apply CSS media queries to tell our content to stack or wrap as opposed to extending off inline. It is very rare that you would ever want give your users an ability to scroll horizontally. Scrolling vertically is more natural for mobile experiences. Overall, we want to give our users an optimized viewing experience based on the alloted amount of space for the device they're using.

4. Use the provided templates to sketch out a mobile, tablet, and desktop version to your site.

You can keep this very general. Use simple shapes and abbreviated text if you'd like. The idea here is to use more of a stacked, concise approach on mobile and use more inline, horizontal elements on desktop making use of the greater amount of space.

Use this opportunity to revamp the layout of your site if you want to start making your site more professional in appearance. Try to mimic layouts you see on the web. While you want to make your site unique, often times the best designed sites are

those that stick to layouts that are both familiar and natural. Content is what truly differentiates your site from others on the web.

5. Return to your project directory. We will now use the methods discussed in today's lesson to make our sketches come to life.

Mobile viewport

a. Start with your mobile sketch.

b. Preview your site in Chrome's mobile view and begin to code your site based on your sketch. From top to bottom use a combination of CSS and rearranging your HTML if necessary to achieve the layout you've created on paper.

Mobile CSS tips

- `display: block` will create a stacked layout
- `display: none` will completely hide an element. This can be useful for hidden, non-essential elements until there is more room to display them.
- `max-width: 100%` allows you set a width on an image or element without letting it overflow in your browser. For instance, if you add `max-width: 100%` to an image it will never be more than 100% of the browser window.
- If your site is scrolling horizontally and you can't tell why, check if an element with margin or padding is causing this behavior.
- Use the style section in the Developer Tools in Chrome to play with different properties and preview them instantly. Then copy the styles that worked into your CSS file in your project directory.

Tablet viewport

c. Once you've completed the base layout for your mobile view, expand your device to a wider viewport. You can use the iPad preset or just make sure you're looking at a viewport larger than about 768px, but smaller than about 992px.

d. We'll now begin to use media queries to set specific rules for this view. Type the following media query into the bottom of your CSS file.

```
@media (min-width: 768px) {
    body {
        background: red;
    }
}
```

e. As you expand and shrink down your browser, you should see the red background kick in the moment your browser is 768px or greater. *Congrats! You just wrote your first CSS media query.*

f. Remove that background rule and the body selector if you don't need it and begin to follow the same process you did in the mobile viewport; only this time, all of your CSS will be nested in the media query.

Desktop viewport

g. Now we'll work our way up to desktop. The media query for small-medium sized desktops, (think laptops), is approximately between **992px and 1200px**. Anything **larger than 1200px** is a larger desktop, (think of the wide monitors in the lab).

h. Use the following media queries in your CSS to recreate your sketch for desktop:

Medium Desktop

```
@media (min-width: 992px) {
    body {
        background: green;
    }
}
```

Large Desktop

```
@media (min-width: 1200px) {
    body {
        background: blue;
    }
}
```

6. Try out this full example.

Often times, you'll want to stack your list items on mobile and with more real estate on desktop you want to display them all in a line. The following example does just that:

```
<!--index.html-->
<nav class="my-nav-class">
    <ul>
        <li><a href="index.html">Home</a></li>
        <li><a href="interests.html">Interests</a></li>
        <li><a href="contact.html">Contact</a></li>
        <li><a href="skills.html">Skills</a></li>
        <li><a href="assignments.html">Assignments</a></li>
    </ul>
</nav>
```

```
/* To start out, make nav items stack
with 10px of margin beneath each item*/
.my-nav-class ul {
    list-style: none;
    padding: 0;
}
.my-nav-class li {
    margin-bottom: 10px;
}
/* On desktop, make nav inline with 5px
of left and right margin on the list items */
@media (min-width: 992px) {
    .my-nav-class li {
        display: inline-block;
        margin: 0 5px;
    }
}
```

7. Starting with mobile, expand your browser slowly, carefully checking each element of your website. Make sure there isn't a breakpoint where your site looks broken or has overlapping text.

While it seems as though we're only accounting for only three distinct viewports, we still want to make sure that at any different screen width our site will remain looking fully intact. They key to responsive web design is not coding for an iPhone or a specifically-sized laptop; it is coding for *all* devices.

You should now have a base for a fully responsive site. Double check your site adding fixes where they are needed. Don't forget to push your changes to your live site via FTP **before you leave class**.