LC 101

Unit 3 - Responsive Design

March 30, 2017

History

- In the early days of the web, not much though was given to design
  - Webpage creators were supposed to just worry about content and let the browser display it however it wanted
    - As a result, webpages were generally ugly
- Then the graphic designers got involved
  - With most coming from a print design background, webpages were designed the same way
    - Generally designed around a fixed width
    - Worked because mostly viewed on desktops with similar screens
- Then came the mobile revolution
  - Old, fixed-width websites designed for desktops often do not work well on mobile devices
    - And thus comes *responsive design*

Responsive Design

- The goal of responsive design is **not** to make a website look the **same** on all devices but rather to look **good** on all devices
  - This often means making the website look **different** on different screen sizes

- This generally involves changing the size and layout of elements on the page based on the screen size

Fluid Grids

- Instead of specifying widths in pixels or other fixed dimensions, use percents
  - Can specify a fixed min-width and/or max-width if necessary

```
.sidebar {
  width: 20%;
  min-width: 100px;
  max-width: 200px;
}
```

Fluid Images

- Instead of specifying a fixed size for images, let the image scale
  - Specify a max-width of 100% instead of a width
    - This will allow the image to scale down (get smaller) but not get larger than its actual size

```
img {
  max-width: 100%;
}
```

CSS Media Queries

- Styles can be applied selectively based on media queries
  - Allows us to pick styles based on browser characteristics, such as width or orientation
    - **`min-width`** is most commonly used

In CSS:

```
@media (min-width: 600px) {
  /* Styles for screens over
     600px wide go here */
}
```

Or in HTML:

```
<link rel="stylesheet" media="(min-width: 600px)" href="min-600px.css">
```

Style Breakpoints

- Useful to think of having *breakpoints*, specific widths where we change from one layout to another

```
/* default styles for < 768px */
@media (min-width: 768px) { /* styles for between 768px and 991px */ }
@media (min-width: 992px) { /* styles for between 992px and 1199px */ }
@media (min-width: 1200px) { /* styles for 1200px and up */ }
```

Mobile First

- It is better to design for the smallest screens first and then add features for larger sizes rather than the other way around
  - Easier to enhance a smaller design than cut down a larger design
  - Web use has increasingly moved to mobile devices
- Use min-width to define breakpoints

```
/* default styles for < 768px */
@media (min-width: 768px) { /* styles for between 768px and 991px */ }
@media (min-width: 992px) { /* styles for between 992px and 1199px */ }
@media (min-width: 1200px) { /* styles for 1200px and up */ }
```

CSS Pixels vs Physical Pixels

- A pixel (`px` unit) in CSS is defined as 1/96th of an inch
  - This means that on high resolution devices there will be multiple physical pixels in a single CSS pixel
  - So, specifications like `min-width: 768px` are theoretically device resolution independent and are based on screen size
    - Though you can also use `resolution` or `min-resolution`, etc. in media queries

Bootstrap

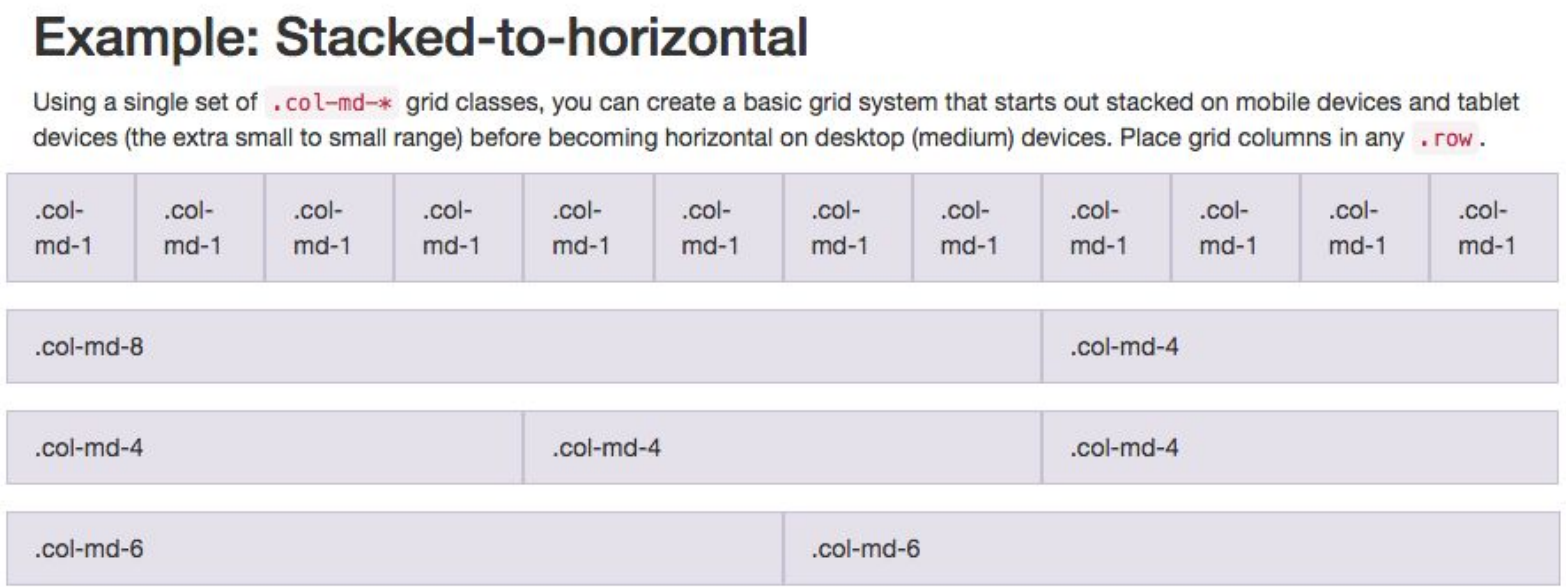
- While we can apply styles ourselves using media queries, there are frameworks that can help
  - http://getbootstrap.com/
- Bootstrap allows you to simply apply special classes to your elements
  - It then uses a combination of CSS and JavaScript to format things for you

Including Bootstrap

- You can download the Bootstrap files and serve them from your own web server or use their CDN
- Get the basic starting template for Bootstrap from here:
  - http://getbootstrap.com/getting-started/#download

Bootstrap Grid System

- Bootstrap is based on a grid system that is 12 columns wide
  - http://getbootstrap.com/css/#grid



Bootstrap Grid System

- Put all of the body in a **`div`** with class **`container-fluid`**
- Nest column divs inside rows
- Specify how many columns (out of 12) each column div should use

```
<div class='container-fluid'>
  <div class='row'>
    <div class='col-md-6'>half row</div>
    <div class='col-md-6'>other half</div>
  </div>
</div>
```

Bootstrap Images

- Add the **`img-responsive`** class to **`img`** elements
  - Will scale the image to the parent element

```
<img class='img-responsive' src='someImage.jpg'>
```

- Other images classes that can be added

`img-rounded`
`img-circle`
`img-thumbnail`

Bootstrap Buttons

- Buttons should be styled with the btn class

```
<button class='btn'>
```

- Some other button classes that can be added

```
btn-block
btn-primary
btn-info
btn-danger
btn-default
```

Font Awesome

- Font Awesome is a collection of scalable icons
  - http://fontawesome.io/

- Use by including the `fa` class and the class for a specific icon on an empty inline element
  - By convention the `i` element is used (though `span` would be more correct)

```
<i class='fa fa-info-circle'></i>
```