GENERAL ASSEMBLY

HTML & CSS

DAY 5

Kim Goulbourne

AGENDA

- Review
- Responsive Design
- Fluid vs Fixed
- Em Typography
- Media Queries
- Handing responsive in JS
- Lab Time

REVIEW

RESPONSIVE DESIGN


I heard you want to be a web developer
Here are a few devices to test your site

WHAT IS RESPONSIVE DESIGN?

Responsive web design (RWD) is an approach to web design aimed at crafting sites to provide an optimal viewing and interaction experience across a wide range of devices (from desktop computer monitors to mobile phones).

We are going to learn how to BUILD responsive sites.

"*One site for every screen.*"

RESPONSIVE EXAMPLES

http://thenextweb.com/
http://elespacio.net
http://time.com/
http://foundermantras.com

What makes these responsive?

WHAT'S HAPPENNING?

- Columns are Changing Size
- Images are Scaling
- Dimensions are Changing Columns
- Navigation Items are Being Rearranged
- Elements Are Being Hidden and Shown
- Typography is Changing Size / Ratios

FIXED VS FLUID

WHAT IS A “FIXED” LAYOUT?

A “fixed” layout uses pixels to set box model values (height, width, margin, padding).

It doesn’t care what size the browser is, it will always honor the pixel value set.

WHAT IS A “FLUID” LAYOUT?

A “fluid/flexible” layout uses percentages to set box model values (width, height, margin, padding) to achieve relative sizes based on the browser size.

Fluid layout is our first step toward Responsive Design.

* Note: You cannot set a % border-width.

LET'S SEE THE DIFFERENCE

Fixed: https://jsfiddle.net/kimgoulb/x0bcwjoh/1/
Fluid: https://jsfiddle.net/kimgoulb/x0bcwjoh/3/

Combo: https://jsfiddle.net/kimgoulb/8Lzmb8hv/1/ (uses pixels for gutter and spacing between elements)

FLUID LAYOUTS EXERCISE

Convert the following layout to use percentage widths

MEDIA QUERIES

WHAT IS A MEDIA QUERY?

A media query is use to define different style rules for different media types/ devices. It's like an "if" statement for CSS.

Media queries can be used to check many things, such as:

- width and height of the viewport or device
- orientation (is the tablet/phone in landscape or portrait mode?)
- resolution and much more

http://www.w3schools.com/cssref/css3_pr_mediaquery.asp

USING THE @MEDIA KEYWORD

```
@media only screen and (check size/resolution/orientation) {
    /* style blocks */
}
```

- “only screen” targets just screens

```
@media all and (check size/resolution/orientation) {
    /* style blocks */
}
```

- “all” targets both screen and print

BROWSER WIDTH/HEIGHT MEDIA QUERIES

@media only screen and (min-width: 1024px) { }
- targets browsers greater than or equal to 1024px wide

@media only screen and (max-width: 1024px) { }
- targets browsers less than or equal to 1024px wide

@media only screen and (min-height: 600px) { }
- targets browsers greater than or equal to 600px in height

@media only screen and (max-height: 600px) { }
- targets browsers less than or equal to 600px in height

```
// all sizes
p {
  font-size: 16px;
}

// what will this media query target?
@media only screen and (min-width: 1024px) {
   p {
     font-size: 24px;
   }
}
```

USING MULTIPLE CONDITIONS

Separate multiple condition statements with “and”:

```
@media only screen and (min-width: 320px) and (max-width: 480px) {
  /* These styles will only apply on devices between 320 and 480 pixels wide. */
}
```

** Note: both min-width and max-width are inclusive so it will match the numbers stated vs being 1 less.*

WHAT ARE BREAKPOINTS?

Breakpoints are target widths used to change the layout. Classic breakpoints to target are:

1440px - desktop large
1280px - desktop
1024 - tablet landscape
768 - tablet portrait
568px - small tablet / large mobile
320px or 375px - mobile (based on iphone5 or 6)

TARGETING TABLET PORTRAIT

```
p {
  font-size: 16px;
}

@media only screen and (min-width: 768px) and (max-width: 1023px) {
   p {
     font-size: 24px;
   }
}
```

TARGETING TABLET PORTRAIT

```
.element {
  width:30%;
}

@media only screen and (min-width: 768px) and (max-width: 1023px) {
   .element {
     width:50%;
   }
}
```

BREAKPOINTS EXERCISE

We need to update the layouts on mobile and tablet using the 320px and 768px breakpoints.

EM TYPOGRAPHY

WHAT IS AN “EM”?

An “em” is a unit of measurement. Just like pixels, ems can determine the size of elements on a web page. Unlike pixels, which are fixed, ems are relative to their parent’s font size.

1em = inherited font size (default on body is 16px)

** If the font size of a is set to 16px, 1em within that is equivalent to 16px.*
** If the font size of that changes to 20px, 1em within that is equivalent to 20px.*

WHAAAAATT???

Check out this demo: https://jsfiddle.net/kimgoulb/gurh7upb/

- The body is set to a font-size of 40px so on this page:
- 1em = 40px //// 2em = 80px //// 3em = 120px

```
.element {
  font-size: 20px;
  line-height: 1.2em; //i.e. 24px
}
```

HOW TO USE EMS

Set a font-size on the <body> using pixels.

Set your typography based on ems.

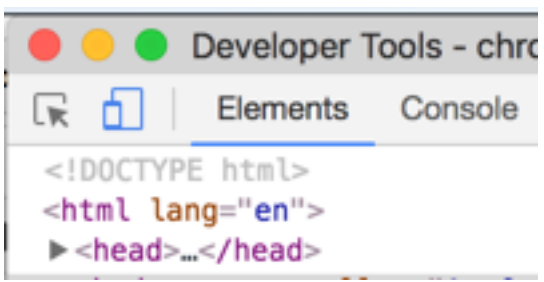
Then, in your media queries at smaller sizes, set that <body> font size to be smaller. The other values will cascade down.

RESPONSIVE TLDR;

IMPORTANT RESPONSIVE TAKEAWAYS

- Use percentages on box model styles (width, height) to create a more fluid layout
- Use media queries to update your layout a different breakpoints
- Use ems on font styles (i.e font size and line height)

Use can test your styles at different breakpoints using the device tool in chrome dev tools (highlighted blue icon).



SOME DESIGN THINGS TO CONSIDER

- Consider designing mobile first to acknowledge the most important elements and then designing desktop.
- Consider the size of the screen and how elements can be played out on a smaller screen. Eg. Grid layouts can typically be stacked or set in a carousel format to be viewing on smaller devices.
- Consider how a user will interact with your navigation. A nav typically takes up too much space to fit on a mobile device outright. It's typically hidden behind a button such as a hamburger or the word menu/nav.

BROWSER PREFIXES

BROWSER/VENDOR PREFIXES IN CSS

Browser prefixes are needed as the browsers experiment and test out their implementations of the newer CSS3 properties. Sometimes all the prefixes are not always needed, but it usually does not hurt to include them, as long as you make sure to put the non-prefixed version last.

```
-webkit-  /* Safari, Chrome and browsers using the Webkit engine */
-moz-      /* Firefox and other browsers using Mozilla's browser engine */
-o-         /* Opera */
-ms-        /* Typically Internet Explorer */
```

USAGE EXAMPLE

```
.box {
  -moz-transition: width 2s;
  -webkit-transition: width 2s;
  -o-transition: width 2s;
  transition: width 2s;
}
```

* Check out properties you may need to prefix here: http://shouldiprefix.com/

RETINA & NON-RETINA IMAGES

RASTER IMAGES

Save your png or jpg images (like icons) at “1x” (non-retina) and “2x”(retina) and serve each based on the pixel density of the device, using media queries.

VECTOR IMAGES

Or you can simple save things like icons as svgs which are vector based and stay sharp across retina and non-retina (so only the 1x version is required).

Note: Make sure to remove xml tags and sometimes the width and height of your sag after saving it to your project folder.

RASTER IMAGES EXAMPLE: THE LOGO

Files: logo1x.png (50px x 50px) and logo2x.png (100px x 100px)

```
/* CSS for devices with normal screens */
.logo {
   background-image: url(logo1x.png);
   background-repeat: no-repeat;
}
```

```
/* CSS for devices with retina screens */
@media only screen and (-webkit-min-device-pixel-ratio: 2),
               only screen and (   min--moz-device-pixel-ratio: 2),
               only screen and (     -o-min-device-pixel-ratio: 2/1),
               only screen and (       min-device-pixel-ratio: 2),
               only screen and (              min-resolution: 192dpi),
               only screen and (              min-resolution: 2dppx) {
                    .logo {
                       background-image: url(logo2x.png);
                       background-size: 50px 50px;
                    }
                }
```

Taken from: https://css-tricks.com/snippets/css/retina-display-media-query/

ICON FONTS

OPTIMIZATION

HTML5 SHIM / HTML5 SHIV

Ensures that older (i.e. IE) browsers view and render HTML5 elements as proper HTML elements.` *Download and learn more here:* http://code.google.com/p/html5shiv/

* You just need to download the javascript (html5shiv.js) file and include it in the <head> of your html.
* Example usage:

```
<!--[if lt IE 9]>
  <script src="js/html5shiv.js"></script>
<![endif]-->
```

CSS & JS

Decrease load time by minifying(making your file smaller and more compact) your JS and CSS. You should have your original source file (styles.css) and a .min version (styles.min.css) where the link tag points to. You can do this online using:

- http://cssminifier.com/
- http://jscompress.com/

* Or using build tools like gruntjs.com or gulpjs.com. A bit of a learning curve.

MEDIA: IMAGES & VIDEO

Save your images “for the web” using Photoshop or Illustrator or online tools such as https://tinypng.com.

* Images: You should always aim to have even your largest image (like fullscreen images) under 500kb and anything else under 200kb. If you’re even close to a 1mb, your site will slow down.

* Videos: they need to be under 5mb is you are locally hosting them vs using a service like Youtube or Vimeo.

RESPONSIVE IMAGES

Various techniques exist to deliver different sized image files based on the browser size(to decrease load time). Check out a few here:

- https://css-tricks.com/which-responsive-images-solution-should-you-use/
- http://alistapart.com/article/responsive-images-in-practice

ICONS (USING SPRITES)

CSS sprites are used in web design as a way to improve performance by combining numerous small images or icons into a larger image called a sprite sheet or tile set, and selecting which icon to show on the rendered page using CSS background-image and background-position.

You can create one on Photoshop or Illustrator or use an online tool like https://draeton.github.io/stitches/.

OR use icon fonts using services like https://icomoon.io/app/

ICONS (USING FONTS)

A collection of awesome icons that you can use with just CSS. They are embedded as a font, so you can set colors, font-sizes, and they are infinitely-scalable! This is preferred over pngs or jpgs because you would need two sized (original and retina) to support high resolution devices.

http://fontawesome.io/
https://icomoon.io

* or you can just use single svgs for your icons as well (it is also scalable)

TESTING

CHECKING CSS CAPABILITIES

You've done lots of cool stuff with HTML5 and CSS3. Great! However, some (especially older) browsers are going to act-up because of that (and browser prefixes aren't enough).

Use this tool to guide you: http://caniuse.com/

CROSS BROWSER/DEVICE TESTING

Look at your site on as many different devices as possible, ensuring that all layout is acceptable / all animations and interactions work. Or, at the very least that things aren't "broken".

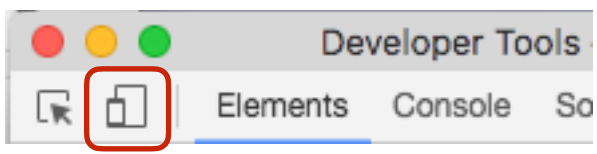
It's always best at the start of a project to know what browsers you will be targeting. For instance, do you care about users on older versions of IE (you shouldn't…)

WHERE TO TEST

See quick snapshots of your site on tons of different browsers using Browser Stack - https://www.browserstack.com

You can also use the device emulator in Chrome (the little icon phone & tablet icon in the top left corner) and Firefox Developer Tools

Chrome:



BROWSER HACKS

Attempt to achieve cross-browser-ness by refactoring to try to fix the issue.

Where you have a problem you can't refactor away, add a "hack":

- http://css-tricks.com/snippets/css/browser-specific-hacks/
- http://browserhacks.com/

IE SPECIFIC HACKS

To target IE specifically, use an IE stylesheet.
Target all IE:

```
<!--[if IE]>
<link rel="stylesheet" type="text/css" href="all-ie-only.css" /> <![endif]-->
```

Target IE 7 and below:

```
<!--[if lte IE 7]>
<link rel="stylesheet" type="text/css" href="ie7-and-down.css" />
<![endif]-->
```

LAB TIME

LET'S MAKE A HAMBURGER MENU

LUNCHTIME

LAB

STARTUP MATCHMAKER

STEPS

1. Create a GIT repository
2. Push your first commit
3. Create an index.html file
4. Create a style.css file and place in the CSS folder
5. Set up your index.html file with the basics
6. Link your external stylesheets to your index.html file
7. Continue….