Responsive Web Design (RWD)

Tasks:
1. Duplicate the a4 folder
2. Setting recommended CSS default values
3. Setting the viewport
4. Add media queries
5. Incorporating fluid design into your layout
6. Setting image sizing to be scalable
7. Add fallbacks for older browsers
8. Validate webpage and CSS
9. Test “Responsive” layouts
10. Upload using DW and submit a zip file thru Canvas

Assignment Overview:
In this assignment we will be learning about Responsive Web Design (RWD). We will be adding RWD techniques to the web template created in A4. You will first need to duplicate your A4 folder and then rename it to a5. The goal of RWD is to create web pages that respond or adapt to various devices with varying screen resolutions. RWD typically uses the following 3 techniques: fluid design, media queries, and flexible images. Media queries are an important part of CSS 3.0 that enables styling various web page components uniquely for different screen resolutions.
RWD definition

“Responsive Web Design (RWD) is an approach to web design aimed at crafting sites to provide an optimal viewing and interaction experience—easy reading and navigation with a minimum of resizing, panning, and scrolling—across a wide range of devices …

A site designed with RWD adapts the layout to the viewing environment by using

**fluid, proportion-based grids, flexible images, and CSS3 media queries, …**

The fluid grid concept calls for page element sizing to be in relative units like percentages, rather than absolute units like pixels or points. Flexible images are also sized in relative units, so as to prevent them from displaying outside their containing element. Media queries allow the page to use different CSS style rules based on characteristics of the device the site is being displayed on, most commonly the width of the browser.” -Wikipedia
Background on Web Design layouts

The following are the three most common types of layouts currently being used in Web Design:

1. Fixed design
   ...uses fixed layouts using exact pixel dimensions like 960px

2. Fluid design
   ...uses layouts that adjust to a screen using % sizing like 100%

3. Responsive Web Design (RWD)
   ...RWD typically uses three techniques for making web pages adapt well across many screen resolutions:
   fluid design, media queries, and flexible images
**Fixed design** is designing with a fixed dimension like pixels and also referred to as non-responsive. Smartphones / tablets display the fixed design, scaling to fit the page to the devices screen resolution. Users typically can zoom in and out in order to better view the content. Fixed design does not scale amongst various resolutions.

The following is an example of fixed design technique and was used for our previous web page layouts.

```html
<div class="container">
  layout tool
</div>
```

CSS style for a typical fixed-width layout.

```css
.container {
  width:960px;
  background:#ccc;
  margin:0 auto;
}
```

The style being used with the div (in the above example) can be named whatever you want. Some designers prefer to style using id's (#idname) instead of a class.
**Fluid design** is designing with a fluid or adjusting dimension. It uses layouts that adjust to a screen based on a %.

CSS style for a fluid layout:

```css
.container {
    width: 100%;
    background: #ccc;
    margin: 0 auto;
}
```

Another RWD technique known as flexible images essentially uses this same % technique with images. We will demonstrate flexible images later in our assignment.
Responsive Web Design (RWD) uses 3 specific techniques:

1. fluid grids (using % for layouts)
2. CSS3 media queries
3. flexible images (using % for sizing)

Read the following article that helps to define Responsive Web Design or RWD
http://alistapart.com/article/responsive-web-design

Media queries were added in CSS 3, enabling styling and layout to be customized to various resolutions. Different resolution ranges can be given different styling, all defined in the style sheet. An example of a media query is shown below. Code goes inside the CSS file, it specifies specific styles based on the resolution of the device displaying the page.

Media query example:

h1 { color:black; }

@media screen and (max-width: 640px) {
  h1 { color:green; }
}

@media screen and (min-width: 641px) and (max-width: 800px) {
  h1 { color:red; }
}

In the above example, when the screen is scaled between 0 and 640px the h1 text color will be green. Between 641 and 800px the h1 text color will be red. Over 800px it will be black the default h1 style defined in the first h1 style.
The following steps explain how to add RWD techniques to the a4 template.

**Step 1 – Duplicate the a4 folder**

First we need to make a copy of your A4 folder and rename it a5. Within DW select the a4 folder control+click (mac) or rt+click (pc) and select `edit>duplicate` and then rename the new folder (a4 copy) to a5.

**Step 2 – Setting recommended CSS default values**

Before we get started lets make sure we add the following to our CSS in order to get the best results from our RWD examples. Note: some of these default values may already be in the reset file used in a4, go ahead and add them to the top of the CSS file to be certain.

Add these styles to the CSS stylesheet

```html
html { 
 font-size:100%;
}

html, body, ul, li { 
 margin: 0;
 padding: 0;
}
```
Step 3 – Setting the viewport

Use meta viewport element to identify desired screen size. The viewport is used with mobile browsers in order to render web pages, scaling them so they fit nicely inside the phone browser’s visible screen area or viewport. The user can also zoom in on areas of interest. The viewport meta tag tells the device at what scale to render the page.

Add the following HTML code inside the head.

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

The concept of the viewport is important to understanding how pages render differently on mobile devices vs desktop browsers. I highly recommend reading more about the role of viewport in RWD in the following articles...

Mozilla viewport recs

https://developer.mozilla.org/en-US/docs/Mozilla/Mobile/Viewport_meta_tag

Apple viewport recs

Step 4 – Add media queries

Add media queries for the desired screen resolutions using min-width and max-width or both. The following pages provide different examples, please use your own desired properties and values. Feel free to customize and decide which resolutions to target.

3 common methods of using media queries:

@media screen and (max-width: 000px) { style { property:value} }

@media screen and (min-width: 000px) { style { property:value} }

@media screen and (min-width: 000px) and (max-width: 000px) { style { property:value} }

For official specs on media queries
http://www.w3.org/TR/css3-mediaqueries/

Example of an actual media query is shown on the next page.
Examples of media queries with properties and values, goes inside CSS file. The first two styles are regular tag styles for the h1 and body tags, which will be used to define the tags outside of the media query resolutions. Rem sizing used below is further explained later in the assignment.

```css
h1 { color:black; }
body { font-size:1.5rem; }

@media screen and (max-width: 640px) {
  body { font-size:1rem; }
  h1 { color:green; }
}

@media screen and (min-width: 641px) and (max-width: 800px) {
  body { font-size:1.2rem; }
  h1 { color:red; }
}

You can also hide items or change how they are displayed

@media screen and (max-width: 640px) {
  footer { display: none; }
  img { display: block; }
}
Ideal resolutions to target? (This is very debatable)

/* small phone */
@media (max-width: 750px) { ... }

/* medium */
@media (min-width: 751px) and (max-width: 1199px) { ... }

/* Large */
@media (min-width: 1200px) and (max-width: 1500px) { ... }

/* Largest */
@media (min-width: 1501px) { ... }
Step 5 – Incorporating fluid design into your layout

Fluid design uses layouts that adjust to a screen using % sizing, for example 85%

```css
.container {
  width: 85.3%;
  background:#ccc;
  margin:0 auto;
  padding:1.67%;
}
```

The following is just an example of how you might choose a % based on your desired display vs actual display. This is not required, just suggested on how best to pick a %.

desired / actual = % or target screen size for the container
1024px / 1200px = 85.3%

This above simply states the actual width resolution on the device is 1200px wide. The desired container size at this resolution is 1024. Dividing 1024 by 1200 = 85.3 So, an equivalent container width in % would be 85.3% on devices that have a 1200px resolution.
Step 6 – Setting image sizing to be scalable

The following technique is known as flexible images. To be honest, I am not really sure why they came up with the name but we will go with it. The following style will make all images scale to display at 100%. The browser will attempt to display all images at 100%, but images will be scaled to fit a particular layout, device or resolution. This technique allows you to use large resolution images, like 1600x1200px, that will be scaled down appropriately to fit a particular display. The images adjust size based on the resolution of the device they are being viewed at. The maximum resolution the image would ever be displayed in the above example would be 1600x1200px. If the device cannot accommodate this resolution then the image is scaled down to fit within the available space.

All images to be displayed at 100%

```css
img {
  max-width: 100%;
  height: auto;
}
```

Alternatively you could define this style to be applied to only certain images

```css
.imgResponsive {
  max-width: 100%;
  height: auto;
}
```
Step 7 – Add fallbacks for older browsers

To make sure our RWD works in future browsers, we need to implement CSS Device Adaptation (2nd part of this code is also needed for ie10). The code below is to be placed in the top of the CSS style sheet, not the HTML document.

```css
@viewport{
  zoom: 1.0;
  width: extend-to-zoom;
}
-webkit-viewport{
  width: extend-to-zoom;
  zoom: 1.0;
}
```

In addition, add the meta tags (shown below) into the HTML code in the head for some browser issues.

```html
<meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1">
<meta name="HandheldFriendly" content="true">
```

In the above example IE=edge means IE should use the latest (edge) version of its rendering engine. Chrome=1 means IE should use the Chrome rendering engine if installed. This tells the browser if the page has been built to be viewable on a small screen. Really just a true / false event, offers no granular control over the width of the document. This code is used by older, obscure phones and possibly for the old Windows phone now discontinued.
Rem sizing

CSS3 introduces a few new units, including the rem unit, which stands for "root em". The rem unit is relative to the root—or the html—element. That means that we can define a single font size on the html element and define all rem units to be a percentage of that without compound problems associated with em.

html { font-size: 62.5%; }
body { font-size: 1.4rem; } /* =14px */
h1 { font-size: 2.4rem; } /* =24px */

Rem sizing fallback

It is also recommended that you add a fallback for older browsers that do not understand rem sizing by adding px sizing to each property that uses rem. List fallback before the rem styling as shown below. The example below also shows a comment tag /*fallback*/ it simply enables you to add comments into your CSS code and are ignored by the browsers.

Add rem sizing for your body and also add a fallback inside your existing body style for font-size. Add a fallback property directly above any use of rem sizing. For example, in the media queries we used rem sizing so make sure to add fallback values. You do not have to use these exact size values.

body {
  font-size: 14px; /*fallback*/
  font-size: 1rem;
}
Step 8 – Validate webpage and CSS
Next, we will attempt to validate our page to see if there are any problems with our code.
  • Validate your CSS file

http://jigsaw.w3.org/css-validator/

Step 9 – Test “Responsive” layouts
Another important component of responsive design, media queries and fluid design is to successfully test your pages. Save and then test using the following link:

http://mattkersley.com/responsive/

If the above test is successful, upload website and then check to see if it is also validated as mobile friendly by google.

Step 10 – Upload using DW and submit a zip file thru Canvas

In the final step you will update the course management page and make sure you have uploaded all the a5 files to your web host account using DW. In addition create a zip file for the a5 folder and submit thru Canvas.

- Open the main index.html (course management page) located at the root directory not the file inside the a5 folder.
- Add a link for a5, linking to the index.html page within the a5 folder
- Save your pages and files, *File > Save All*
- Upload all pages and necessary files (*images, CSS file*) to your web host.
- Exit DW and find the a5 folder. Select folder and make a zip file in order to turn in through Canvas using the assignment 5 upload.
Grading Criteria

- Duplicate the a4 folder renaming to a5
- index.html, ext CSS within the a5 directory
- Set CSS default values
- Set the viewport
- Add media queries
- Code images to be scalable
- Add fallbacks
- Validated CSS
- Test the “Responsive” layout
- Published a5 to your web host account
- Submitted zip file via Canvas A5 link

-end

The next several pages provide reference in regards to RWD and related topics, they are optional reading.
Responsive is not nec RWD?

Read the following explaining why the term “responsive” is evolving. RWD and responsive are argued to mean two different things.

http://alistapart.com/column/what-we-mean-when-we-say-responsive

“The definition of “responsive" to be evolving ...something has the quality of being responsive. ...distinguish between Responsive Web Design and “responsive.” ...Responsive Web Design is constrained to the three specific techniques for making sites that adapt well across many browser environments: fluid layouts, flexible images (and media objects), and media queries. ...there's no consensus about what "responsive" means. ...How we make things “responsive" is up to us. “Responsive Web Design is constrained to the three specific techniques for making sites that adapt well across many browser environments: fluid layouts, flexible images (and media objects), and media queries."
Screen resolutions

Which screen resolutions should you target?

Checkout the following examples of various iOS device resolutions.

http://www.iosres.com/

Common display resolutions on iOS mobile devices

640
750
768
800
1024
1080
1242
1280
1536
1960
References used in assignment

**RWD**
Making a website responsive in 3 easy steps
http://www.catswhocode.com/blog/making-a-website-responsive-in-3-easy-steps

What’s responsive Web design all about?
http://arstechnica.com/business/2012/05/17/whats-responsive-web-design-all-about/

**Viewport**
Mobile Web Application Best Practices
http://www.w3.org/TR/mwabp/#bp-viewport

Viewport explained by Apple

Good explanation about the viewport declaration by Mozilla
https://developer.mozilla.org/enUS/docs/Mozilla/Mobile/Viewport_meta_tag

For official specs on the viewport checkout
http://www.w3.org/TR/css-device-adapt/

Simple explanation
http://webdesign.tutsplus.com/tutorials/quick-tip-dont-forget-the-viewport-meta-tag--webdesign-5972