Step-by-Step Learning Activity

a) Media Queries

***Key takeaway**: Use media queries to create alternate styles for different sized devices, or orientation on devices.*

A Media query asks the browser if something is true and specifies a style based on the response.

Media queries used to create responsive web designs include:

```
@media only screen and (max-width: 600px) {...}
@media only screen and (min-width: 768px) {...}
@media only screen and (orientation: portrait) {...}
@media only screen and (orientation: landscape) {...}
```

b) Media Queries – small devices (phones, 600px and smaller)

***Key takeaway**: Use media queries to create alternate styles for different sized devices.*

@media only screen and (max-width: 600px) – by asking this question you can create break points in your website. If the screen is only 600px, then specify an image that is smaller or has a different orientation than the one you would use for your desktop. This way you can ensure that your site looks good on any device.

Small devices (phones, 600px and smaller)

```
@media only screen and (max-width: 600px) {
...alt styles
}
```

c) Media Queries – Medium devices (landscape tablets, 768px and larger)

***Key takeaway**: Use media queries to create alternate styles for different sized devices.*

@media only screen and (min-width: 768px) – by asking this question you can create break points in your website. If the screen is over 768px, then specify styles that suit a tablet sized screen.

Medium devices (phones, 768px and larger)

```
@media only screen and (min-width: 768px) {
...alt styles
}
```

d) Media Queries – Desktops (1200px and larger)

***Key takeaway**: Use media queries to create alternate styles for different sized devices.*

@media only screen and (min-width: 1200px) – by asking this question you can create break points in your website. If the screen is over 1200px, then specify styles that are more suitable for a desktop screen.

Large devices (desktops, 1200px and larger)

```
@media only screen and (min-width: 1200px) {
...alt styles
```

```
}
```

e) Add a media query to create alternate styles for phones 600px wide or smaller

***Key takeaway**: Use max-width to declare styles for devices smaller than the specified widths.*

*The media query should be at the **bottom** of the CSS document so that it is the last rendered styles. If not, styles following the media query may overwrite the styles according to Cascading style rules.*

Selectors and their properties are defined within the curly brackets of the media query.

Use the provided mediaquery.css file (this is inspired be the completed Lesson 5 activity):

Step 1: In this CSS document, add a media query targeting phones

```
@media only screen and (max-width: 768px) {
}
```

Step 2: Save the mediaquery.css file.

f) Reduce text-size for elements on a smaller screen

***Key takeaway**: If necessary, reduce text size for smaller screens.*

Step 1: In the CSS document, add the h1 selector inside the media query and set the font-size to 28px.

```
@media only screen and (max-width: 768px) {
    h1 {
        font-size: 28px;
    }
}
```

Step 2: Add the h2 selector inside the media query. Set the font-size to 20px.

```
@media only screen and (max-width: 768px) {
    h1 {
        font-size: 28px;
    }
    h2 {
        font-size: 20px;
    }
}
```

Step 3: Save the mediaquery.css file.

Step 4: Preview the mediaquery.html page in the live server.

Step 5: Resize the web browser to see the effect of the media query.

Step 6: Switch back to HTML document to proceed to next segment.

g) Remove margins for elements on a smaller screen

***Key takeaway**: To increase screen real estate remove extraneous margins.*

Step 1: In the CSS document, add the global selector inside the media query. Set the margins to 0px.

```
* {
    margin:0px;
}
```

Step 2: Save the mediaquery.css file.

Step 3: Preview the mediaquery.html page in the live server.

Step 4: Resize the web browser to see the effect of the media query.

Step 5: Switch back to HTML document to proceed to next segment.

h) Reduce padding for elements on a smaller screen

***Key takeaway**: To increase screen real estate, reduce padding.*

Step 1: In the CSS document, add the figure selector inside the media query. Set the padding to 0px.

```
@media only screen and (max-width: 768px) {
    figure {
        padding:0px;
    }
}
```

Step 2: Add the .mycolumn class selector inside the media query. Set the padding to 15px.

```
@media only screen and (max-width: 768px) {
    .mycolumn {
        padding: 15px;
    }
}
```

Step 3: Save the mediaquery.css file.

Step 4: Preview the mediaquery.html page in the live server.

Step 5: Resize the web browser to see the effect of the media query.

Step 6: Switch back to HTML document to proceed to next segment.

i) Remove floats on a smaller screen

***Key takeaway**: Floating elements should be removed for small devices to allow them to break into rows of content.*

Step 1: In the CSS document, in the figure inside the media query. Set the float to none.

```
@media only screen and (max-width: 768px) {
    figure {
        float: none;
        padding:0px;
    }
}
```

Step 2: In the .mycolumn class selector inside the media query. Set the float to none.

```
@media only screen and (max-width: 768px) {
    .mycolumn {
        float: none;
        padding: 15px;
    }
}
```

Step 3: Save the mediaquery.css file.

Step 4: Preview the mediaquery.html page in the live server.

Step 5: Resize the web browser to see the effect of the media query.

Step 6: Switch back to HTML document to proceed to next segment.

j) Resize elements for a smaller screen

***Key takeaway**: Floating elements should be removed for small devices to allow them to break into rows of content. Phones do not have the room to display content side by side*

Step 1: In the CSS document, add the img selector inside the media query. Set the width to 100% and the height to auto.

```
@media only screen and (max-width: 768px) {
    img {
        width: 100%;
        height: auto;
    }
}
```

Step 2: In the .mycolumn class selector inside the media query. Set the width to 100%.

```
@media only screen and (max-width: 768px) {
    .mycolumn {
        float: none;
        width: 100%;
        padding: 15px;
    }
```

```
    }
```

Step 3: Save the mediaquery.css file.

Step 4: Preview the mediaquery.html page in the live server.

Step 5: Resize the web browser to see the effect of the media query.

Step 6: Switch back to HTML document to proceed to next segment.

k) Revaluate element styles for a smaller screen

***Key takeaway**: Once you have resized elements for a smaller screen you may want to adjust other styles to ensure the styles suit the device screen size.*

Step 1: In the CSS document, in the figure inside the media query. Set the border to none and add a margin to the top and bottom of the figure.

```
@media only screen and (max-width: 768px) {
    figure {
        float: none;
        padding:0px;
        border: none;
        margin: 15px 0px;
    }
}
```

Step 2: Save the mediaquery.css file.

Step 3: Preview the mediaquery.html page in the live server.

Step 4: Resize the web browser to see the effect of the media query.

Step 5: In the CSS document, add the figcaption selector inside the media query. Set the bottom to 0px and the width to 100%.

```
@media only screen and (max-width: 768px) {
    figcaption {
        bottom: 0px;
        width: 100%;
    }
}
```

Step 6: Save the mediaquery.css file.

Step 7: Preview the mediaquery.html page in the live server.

Step 8: Resize the web browser to see the effect of the media query.

Step 9: Switch back to HTML document to proceed to next segment.

l) Hide extraneous elements on a smaller screen

***Key takeaway**: Hide nonessential elements for smaller screens.*

When designing for smaller screen sizes you want to be sure that you are not bogging down your user's bandwidth with unnecessary elements that take a long time to load. Hide nonessential elements for smaller screens.

Step 1: In the CSS document, add the id selector #myvideo inside the media query. Set display property to none.

```
@media only screen and (max-width: 768px) {
    #myvideo {
        display: none;
    }
}
```

Step 2: Save the mediaquery.css file.

Step 3: Preview the mediaquery.html page in the live server.

Step 4: Resize the web browser to see the effect of the media query.

Step 5: Switch back to HTML document to proceed to next segment.

m) Style for mobile first

***Key takeaway**: Design for mobile first to help the loading time on smaller screens.*

The reason for this is rendering rules for Cascading Style sheets. The browser will render the code from top to bottom. If mobile styles are listed first, they will be rendered before all others. If the desktop styles are rendered as a media query, they will not be rendered at all for mobile devices.

We will discuss this in future sessions.