Responsive Layouts

MMP 350 Fall 2020

Responsive Design

Responsive design is the practice of designing websites that change style and layout to fit a variety of devices with different dimensions, screen densities and functionalities to create an optimal viewing experience based on a single template. In responsive design, the rules will move fluidly between sizes, applying new properties for each style rule.

Adaptive Design

Like responsive design, adaptive design changes based on screen size and density, but adaptive design generally pinpoints specific breaking points or thresholds and creates designs for those points.

Progressive Enhancement

Both concepts are rooted in the idea of progressive enhancement, the idea that websites should be designed to provide the basic content to any device, while progressively introducing enhanced design possibilities for modern devices.

Progressive enhancement starts with the content of the site, then adds visual design, and finally user interaction on top.

Viewport

We need to add a new <meta> tag to set the viewport inside the <head></head> section of our HTML.

This tells the browser not to resize or scale the content based on the device. Some browsers do this by default to load the desktop version of the site at a viewable scale. We're telling the browser that we are handling this in our CSS.

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

Media Queries

To implement Mobile and Tablet sized wireframe designs, we will use media queries in CSS to set breakpoints in terms of pixel dimensions (in device width) to change the layout.

Media Queries are like extra style sheets that let us rewrite the original style rules based on device properties. You can test the size of the screen and change the design around the content that will be on the site. These are called break points.

```
@media (max-width: 400px) {
        /* mobile style rules */
}

@media (min-width: 401px) and
(max-width: 600px) {
        /* tablet style rules */
}

@media (min-width: 601px) {
        /* monitor rules */
}
```

Break Points

Let's create 2 break points: One for mobile, smaller than 400px, and one for tablet, between 401 and 600px. The relationship between rules matter. For example, in the second media query, we need to start at min-width and define a range to max-width. Without that beginning point, this media query would replace the one that comes directly before it.

The first media query only needs a max-width rules because there is nothing smaller targeted. Anything larger than 600px will be handled by the original style rules.

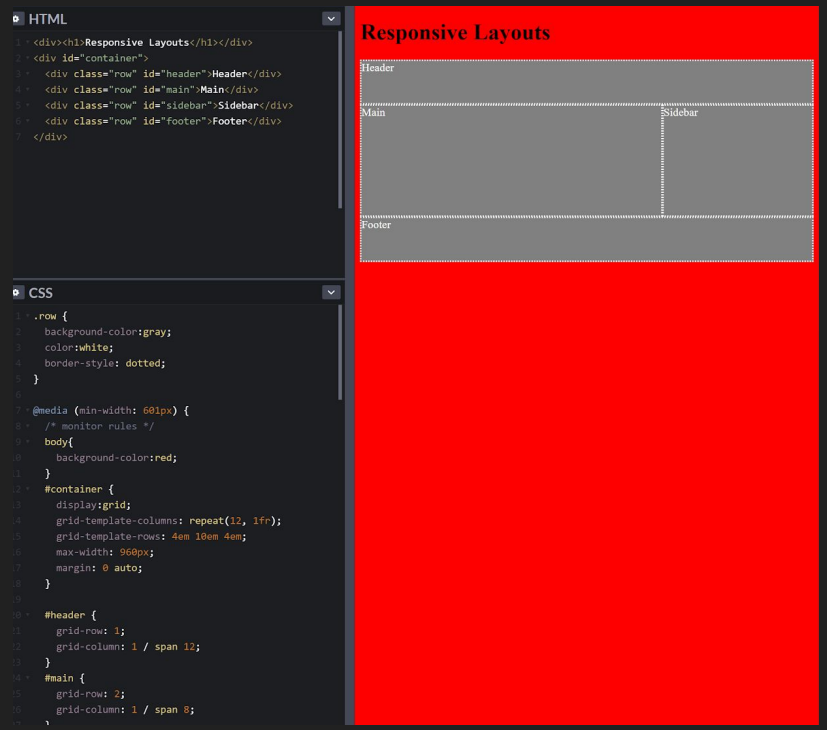
```
@media (max-width: 400px) {
        /* mobile style rules */
 body{
    background-color:blue;
  }
}

@media (min-width: 401px) and
(max-width: 600px) {
        /* tablet style rules */
 body{
    background-color:green;
  }
}
```

grid-template-areas

Another way to change our grid is using grid-template-areas. This adds a semantic dimension to creating layouts. Template grids allow us to position the content on the page irrespective of the order it appears in HTML.

This could be achieved with grid and row numbers, but template-areas allow us to do this using named areas and moving them around without necessarily have to count through all of the units each time.

[Demo for responsive layouts]



Testing Responsive Layouts

It's not enough to just drag the browser width in and out to test responsive layouts. We can emulate mobile and tablet devices using the mobile preview in Chrome. Open View > Developer > Developer Tools, or hit CTRL + Shift + I on PC, or Command + Option + I on Mac.

Click on the icon in the Toolbar that looks like a mobile phone. You will see the content size change and a new menu appear with mobile parameters to set the width and height or choose specific devices.