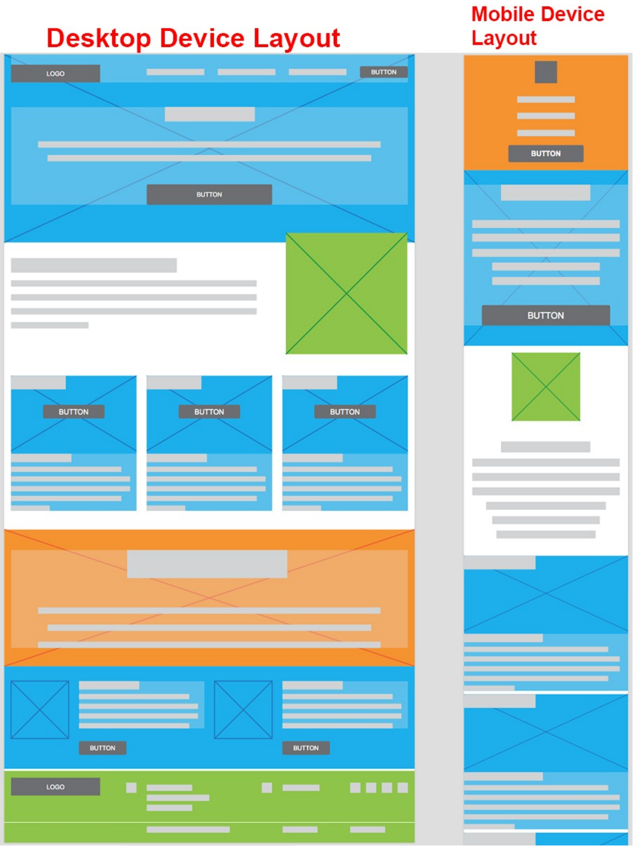
MEDIA QUERIES

Media queries are simple filters that you can apply to CSS styles, which enables you to create a responsive experience. A key concept behind responsive design is fluidity and proportionality, as opposed to fixed-width layouts. In the following example we will use a media query to target devices with a screen width of 768 pixels or wider:

```
//CSS to hide mobile toggle button for desktop device
@media screen and (min-width: 768px) {
  .btn-mobile-toggle {
           display: none;
   }
}
```

Media queries tailor a website’s content presentation to a specific range of output devices, without changing the content itself. Also, media queries make it easy to change styles, based on the characteristics of the device rendering the content.



Media queries look at the capability of the device, and check the:

- Width and height of the viewport
- Width and height of the device
- Page orientation
- Resolution

A media query consists of a media type and zero or more expressions that limit the style sheet's scope by applying media features, such as width, height, and color.

Using relative units for measurements simplifies layouts and prevents you from accidentally creating components that are too big for the viewport. In addition, using relative units allows browsers to render the content, based on the user’s zoom level, without the need for adding horizontal scroll bars to the page.

The most commonly used media queries are:

- `min-width` Rules applied for any browser width over the value defined in the query
- `max-width` Rules applied for any browser width below the value defined in the query
- `min-height` Rules applied for any browser height over the value defined in the query
- `max-height` Rules applied for any browser height below the value defined in the query
- `orientation:portrait` Rules applied for any browser where the height is greater than or equal to the width
- `orientation:landscape` Rules for any browser where the width is greater than the height

Note: There is an important difference between `min-width` and `min-device-width`. The value for `min-width` is based on the size of the browser window, while the value for `min-device-width` is based on the size of the display screen for the device.

Device Breakpoints

Breakpoints are the design element cutoff points you specify in your website design that provide the most effective visual layout for your site’s content. Breakpoints can be divided into two different types; major and minor. Major breakpoints are usually based on the most classic device sizes being used by people, and minor are breakpoints are used to fix content issues between the major breakpoints. It’s imperative that you create web applications that are focused on readable content at any width. You should always incorporate a fluid layout structure by using percentage widths to account for all widths between major breakpoints.

Example of a set of major device breakpoints:

```
/* Extra small devices (phones, up to 480px) */
@media screen and (max-width: 767px) {...}


/* Small devices (tablets, 768px and up) */
@media (min-width: 768px) and (max-width: 991px) {...}


/* tablets/desktops and up  */
@media (min-width: 992px) and (max-width: 1199px) {...}


/* large like desktops and up */
@media screen and (min-width: 1200px) {...}
```

Mobile first

Remember to design your website’s layout for the smallest mobile device first, and progressively enlarge the layout design as more screen area becomes available. For devices with large screens, it’s best to limit the maximum width of the container panel so that it doesn’t consume the whole screen width, as in the following:

```
//css for main container
.container {
  max-width: 62.5rem;
}
```

Avoid using pixels to declare your breakpoints, since this creates a horizontal scrollbar when the user zooms in on your content. Instead of using pixels, use relative units, which allows browsers to adjust the design, based on the user’s zoom level.

Media queries detect your website’s viewport width, color, color index, aspect ratio, device aspect ratio, width, device width, height, device height, orientation, monochrome, resolution, scan, and pixel-density. To optimize media queries, you can use:

1. Pixel-density to conditionally serve larger background images and icon sprites for Retina and other high resolution screens
2. Height to detect the available screen height, and adjust styles accordingly
3. Orientation to detect whether a screen is in portrait or landscape mode. You can also use Orientation to conditionally disable fixed positioning to free up screen area.

Employing conditional loading allows you to prioritize your website’s core content, and doesn’t detract from your website's performance.

Going Forward

Keep these breakpoint guidelines in mind when designing your website’s layout:

1. Optimize the text for reading
2. Use major device breakpoints, and address content with minor breakpoints
3. Treat your website’s layout as an enhancement
4. Use relative units like `rem` or `em`;

It is growing increasingly more popular to build major breakpoints based on major content shifts instead of current devices. Because, the majority mindset is still based on devices before content, many designers tend to build their designs based on devices not just content, also contracts with statements of work are still written to include devices. Ultimately, find the breakpoints that best suite you, your project, and your team.

https://www.w3schools.com/css/css3_mediaqueries.asp
https://developer.mozilla.org/es/docs/CSS/Media_queries
https://lenguajecss.com/p/css/propiedades/responsive-design