HTML & CSS: Day 5

Deep Dive on Media Queries

Fullstack Academy

Lesson Objectives

Media Queries

1. What a Media Query is
2. Why they're useful
3. Structure of a Media Query
4. Media Types
5. Media Features
6. Operators

What is a Media Query?

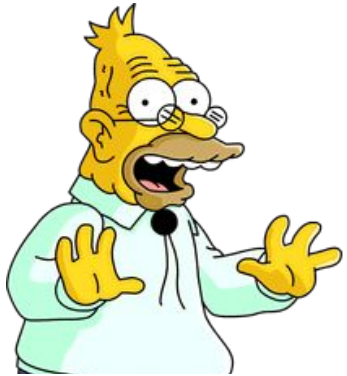
A way to define a page's CSS rules depending on the **type** or **capabilities** of the **medium** on which it is displayed.

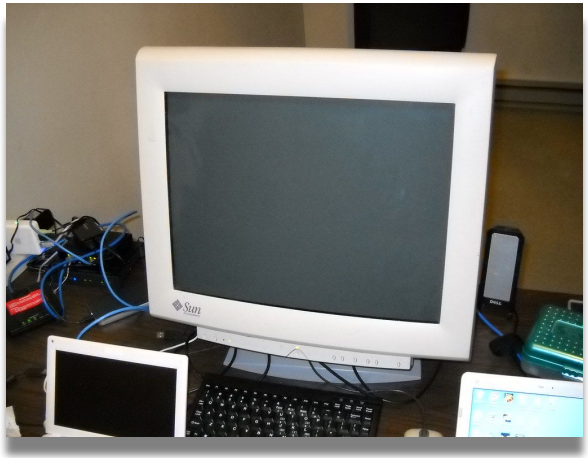
Why They're Useful

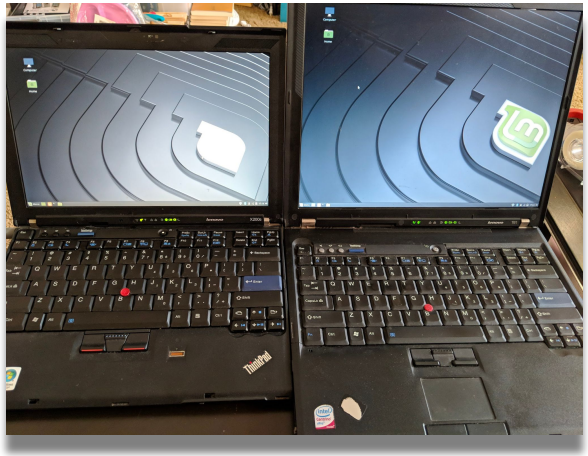
Up until relatively recently, people mostly interacted with websites on one of two kinds of displays: a desktop monitor, or a laptop screen.

They were (mostly) all pretty similar resolutions and aspect ratios.

So websites pretty much looked the same no matter what kind of computer you used.





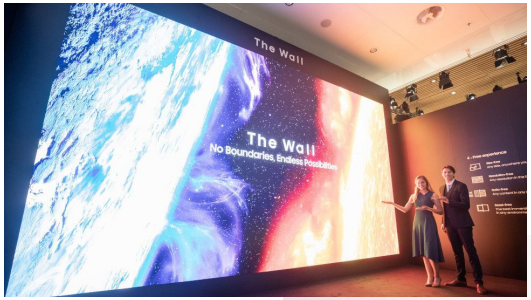


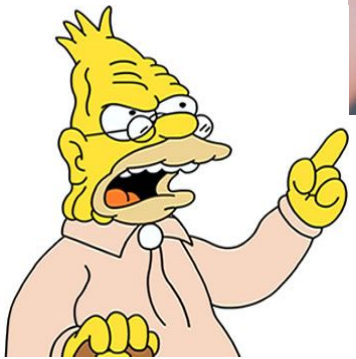
Why They're Useful

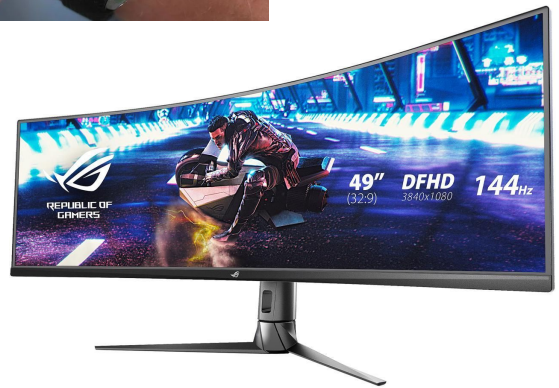
These days, we have a huge variety of display sizes, resolutions, pixel densities, and shapes to account for.

Well over 50% of all website traffic these days is from a mobile device!

So, modern site design needs to consider the kind of display sizes and ratios to support.







 |

Why They're Useful

Additionally, media queries can help accommodate users with some difficulties or medical conditions. Many of these are not widely supported (or recommended), but at least a few are fairly widely supported, even if experimental. For example:

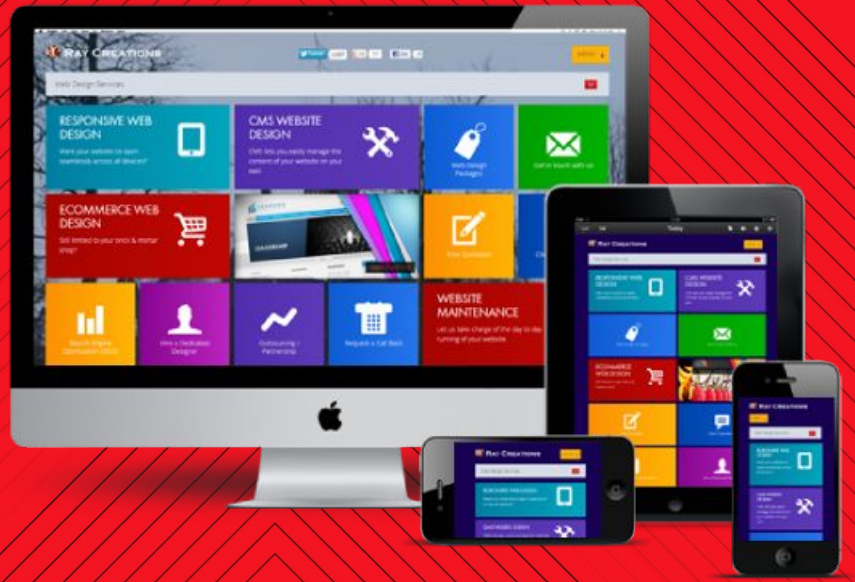
```
@media (prefers-reduced-motion: [no-preference/reduce])
@media (prefers-contrast:[no-preference/more/less])
```

This can help users that have difficulty with strobing or rapidly moving animations. This may include users with ADHD, epilepsy, etc

Resource: https://caniuse.com/?search=prefers-contrast
https://caniuse.com/?search=prefers-reduced-motion

This is called “responsive design,” and is (usually) accomplished with media queries.

Note: You’ll often hear the phrase “mobile first.” This is simply a design philosophy that prioritizes mobile screens in a responsive design.



 |

Structure of a Media Query

Fullstack Academy

This is the general structure of a media query.
Note that you can chain operators and features!

Reference: https://css-tricks.com/a-complete-guide-to-css-media-queries/

Media Types

- OPTIONAL unless using the `not` or `only` operators
  - We'll talk about those in a minute
- Describe the general category of the device
- Only 3 types:
  - `all` (default, suited for all devices)
  - `print`: intended for printing to a page, or viewing in 'print preview' mode
  - `screen`: intended for (you guessed it) screens

Reference: https://css-tricks.com/a-complete-guide-to-css-media-queries/
https://developer.mozilla.org/en-US/docs/Web/CSS/@media#media_types

Media Features

- Completely optional
- Enclosed by parentheses: `([feature]: [value])`
- Describe the specific characteristics of the display/device/environment
- Lots and lots of types, but the following two are probably what you'll be using most:
  - `max-width` : Apply styles only BELOW this screen width
  - `min-width` : Apply styles only ABOVE this screen width

Reference: https://css-tricks.com/a-complete-guide-to-css-media-queries/
https://developer.mozilla.org/en-US/docs/Web/CSS/@media#media_types

Media Operators

- Use these to combine media queries logically

- 4 operators:
  - `and` : Requires each feature to be TRUE to apply styles
  - `not` : Requires the query to be FALSE to apply styles
  - `,` : if any queries separate by a comma are TRUE, the whole statement returns TRUE (logical or operator)
  - `only` : Requires the whole query to match to apply styles. Primarily useful for preventing older browsers from applying selected styles

Reference: https://css-tricks.com/a-complete-guide-to-css-media-queries/
https://developer.mozilla.org/en-US/docs/Web/CSS/@media#media_types

 |

Media Operators

Fullstack Academy

AND

```
@media (min-width: 320px) and (max-width: 768px) {
  .element {
    /* Styles! */
  }
}
```

Reference: https://css-tricks.com/a-complete-guide-to-css-media-queries/

Media Operators

Fullstack Academy

NOT

```
@media not all and ( orientation: portrait ) {
  header, footer {
    display: none;
  }
}
```

Reference: https://css-tricks.com/a-complete-guide-to-css-media-queries/

Media Operators

Fullstack Academy

, (OR)

```
@media (orientation: landscape), (min-width: 1200px) {
  .element {
    /* Styles! */
  }
}
```


Reference: https://css-tricks.com/a-complete-guide-to-css-media-queries/

Media Queries: Not Just CSS!

Fullstack Academy

You can also use Media Queries in HTML and JavaScript!

This is accomplished in HTML using the `media=` attribute, and in JavaScript using the `window.matchMedia()` method.

Reference: https://css-tricks.com/a-complete-guide-to-css-media-queries/

Media Operators: Not Just CSS!

Fullstack Academy

HTML

```
<html>
  <head>
    <link rel="stylesheet" href="all.css" media="all" />
    <link rel="stylesheet" href="small.css" media="(min-width: 20em)" />
    <link rel="stylesheet" href="large.css" media="(min-width: 90em)" />
</html>
```

(brownie points if anyone can tell me what this does)

Reference: https://css-tricks.com/a-complete-guide-to-css-media-queries/

Media Operators: Not Just CSS!

Fullstack Academy

JavaScript

```
const mediaQuery = window.matchMedia( '( max-width: 1280px )' )


if ( mediaQuery.matches ) {
  document.location = “/tinywindow.html”
}
```

(brownie points if anyone can tell me what this does)

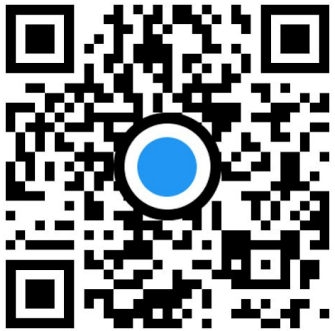
Reference: https://css-tricks.com/a-complete-guide-to-css-media-queries/

Media Queries: Feature, Type, or Operator?

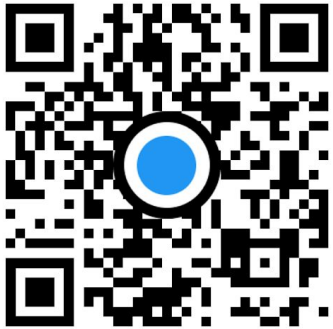
(Bonus: Tell us what it does)

1. `print`
2. `aspect-ratio`
3. `,`
4. `max-width`
5. `orientation`

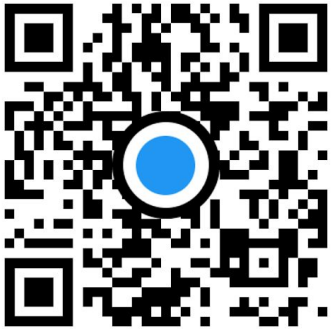
Feature



Type



Operator



Fullstack Academy

BREAK TIME

DEMO TIME

