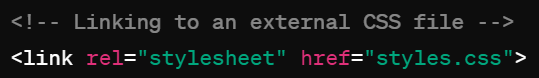
**What is CSS?**

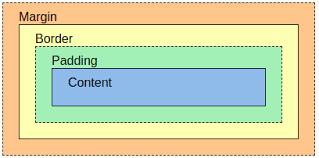
* **Cascading Style Sheet** - A language for **styling HTML documents**.
  + It lets us define the *look* and position of HTML elements.
* CSS uses key/value pairs to indicate which element to style.
* CSS is broken down into **selectors and declarations**
  + **Selector**: Select the HTML element to style.
  + **Declaration**: Defines the style you are imposing on the selector.

**Where can we put CSS?**

* There are **three places you can declare CSS styling**
  + **External Style Sheets** (this tends to be best practice)
    - A separate CSS document that you link to in your HTML
    - 
  + **Internally in the <head>** (uncommon)
  + **Inline of the actual HTML element tag** (not best practice but easy and ok to do in a pinch)
* In case of conflicts? What if you’re trying to style the same element in multiple declarations?
  + **The CSS closest to the element will “win”** in conflicts
    - So inline styling beats head styling which beats external style sheets

\*Showed 2 ways to style elements (not external yet) \*

**CSS Box Model**

* CSS treats each HTML element in the document as a “box”. Each box has its own 4 regions that can all be styled independently of one another.
* 
* **Margin:** The outermost layer. Useful for separating the element from its neighbors. The dimensions are given by the margin-box height/width.
* **Border:** The layer between the margin and padding, its dimensions are given by the width and height of the border. The default border style is no border – you can edit this attribute to get a |visible border|.
* **Padding:** The layer between the content and within the border box. Its dimensions are given by the width and height of the padding box. Useful for making space between the border and content.
* **Content:** This consists of the actual content of the element such as text, images, or any other media. It is bound by the content edge and its dimensions are given by content-box width and content-box height.

\*Showed examples of box model\*

**Other Commonly-Styled CSS Properties**

* There are many, many properties that can be styled with CSS. Some include:
  + **Position:** Determines where on the page an element is displayed
    - Static - The element’s box is arranged normally. This is the default.
    - Relative - Defaults to the normal position, but can change RELATIVE to the other elements’ positioning
    - Absolute - The element’s box is placed in an ABSOLUTE position on the page, such as top right or bottom left. When you scroll, it will follow.
    - Fixed - The element’s box is in a FIXED location in the window, unaffected by user scrolling or other elements.
  + **Color:** Allows you to change the color of an element
    - Includes simple options like “red” but you can use rbg or hex color codes
    - Color is everywhere – most elements can be given a color.
  + **Text Align:** Sets the text inside the content to be of a particular alignment. I.e. left/right justified, or centered
  + **Font:** Chooses the font of text-based elements, with options such as:
    - font-family
    - font-size
    - font-weight (for **bolding**)
    - font-style (for *italics*)
  + **Background:** Configures the styling of the background of the page
    - background-color
    - background-image
    - background-size
    - background-position

**CSS Selectors**

* “Selectors are how we choose (hence SELECTors) which HTML element to style.” -Good starting QC line
* **Element Selectors:** Selects all the instances of an HTML element

|  |
| --- |
| **p {**  **text-align:center;**  **color:blue;**  **}** |

* **ID Selectors:** Selects elements by the ID attribute. IDs should be unique. **Uses the # to select. Better for individual elements**

|  |
| --- |
| **#div1 {**  **text-align:center;**  **color:blue;**  **}** |

* **Class Selectors:** Will style any element with the declared class attribute. **Uses the . to select.** Can be combined with an element selector! **Better for groups of elements.**

|  |
| --- |
| **.intro {**  **text-align:center;**  **color:blue;**  **}** |

* **Sibling Selectors:** Selects an element directly after a certain element. **Uses the +.**

|  |
| --- |
| **div + p {**  **text-align:center;**  **color:red;**  **}** |

//FOR QC you can probably just know the above 4 selects and be fine. Just be good on their syntax and how they work. The rest of them below are nice to know if you want more advanced styling, or for some extra ammo for QC.

* **Universal selector** - selects everything. Uses the \*

|  |
| --- |
| **\* {**  **text-align:center;**  **color:pink;**  **}** |

* **Attribute Selector** - Allows you to select elements that declare a specific attribute. Uses the [] to select.

|  |
| --- |
| **[title] {**  **text-align:left;**  **color:blue;**  **}**  **[class=”warning”]{**  **color:green;**  **}** |

* **Grouping Selectors:** you can group multiple selectors and apply the same style. Simply declare the selectors and separate them with a comma

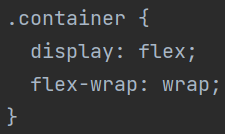
|  |
| --- |
| **H1, #para1 {**  **text-align:left;**  **color:red;**  **}** |

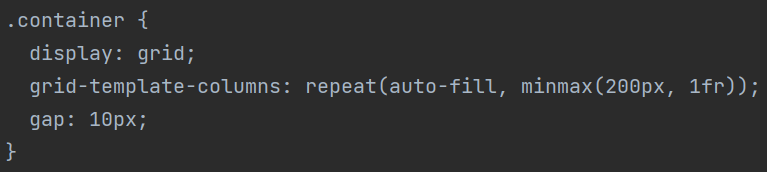
* **Child selectors** - Selects specific elements that are direct children of another specific type of element. Uses the > sign.

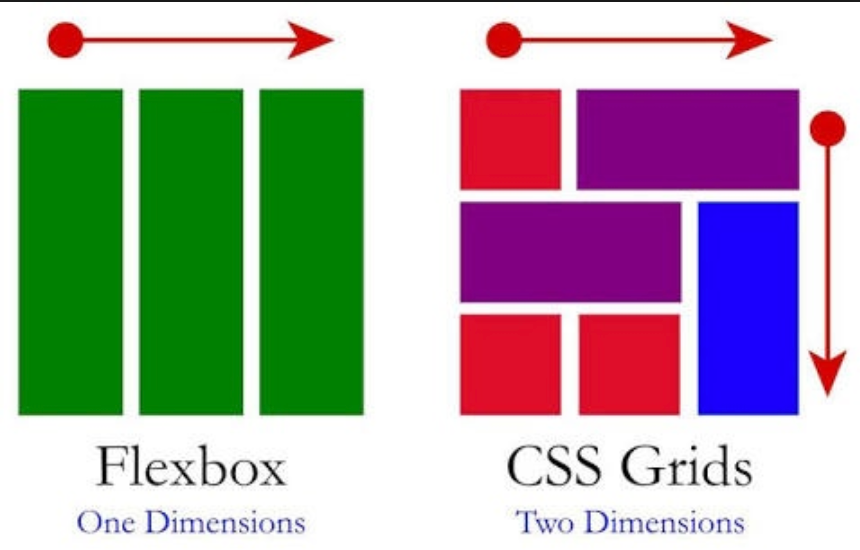
|  |
| --- |
| **div>p {**  **text-align:left;**  **color:red;**  **}** |

* **Conflicts?** If selectors conflict with each other, then the “winning” CSS is determined by **specificity**
  + ID wins over everything
  + Class beats anything under ID
  + Element is the lowest priority
  + If specificity does not resolve the conflict, the determination is made by the order of the declarations. The latest declaration declared will win.

**Responsive Web Design**

* Responsive web design is a design approach that allows websites and pages to render well on a variety of devices and window or screen sizes.
  + It's about using HTML and CSS to automatically resize, hide, shrink, or enlarge components on a website, to **make it look good on all devices** (desktops, tablets, and phones).
    - In other words, the goal is for the webpage to **respond** to the size of the window it’s being viewed on
* **Flexbox and Grid** are two CSS techniques that are fundamental to responsive design.
* **Flexbox** is a one-dimensional layout method for laying out items in rows or columns. **Items flex to fill additional space and shrink to fit into smaller spaces.**
  + Flexbox can do things like:
    - **flex-direction:** define the direction of items in a flexbox
    - **flex-wrap:** define whether items can wrap onto new lines
    - **align-****items:** aligns flexbox items along their row
  + 
    - This is a **flex container,** with a typical use of flexbox (setting content to wrap).

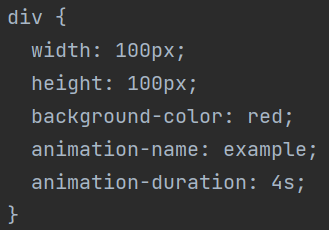
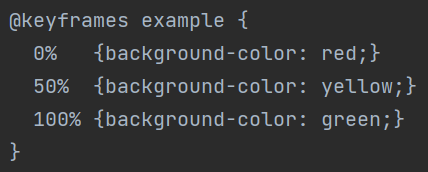
* **CSS Grid** is a two-dimensional layout system that can handle both rows and columns. **It's great for laying out items in two dimensions (...rows and columns).** This is particularly useful for more complex layouts that need to be responsive.
  + 
  + grid-template-columns: repeat(auto-fill, minmax(200px, 1fr)); means that the grid will have as many columns as can fit without any of them dropping below 200px in width.
    - 1fr means that the remaining space will be distributed equally among the columns. fr stands for “fraction”. The columns will take the same an equal fraction of the remaining space.



**Other CSS3 Concepts on the Curriculum**

**Animations**

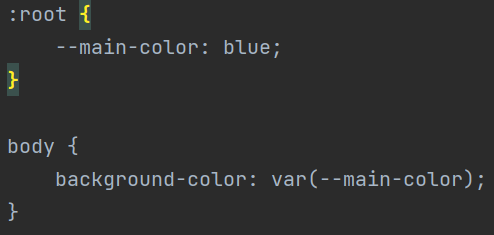
* CSS animations make it possible to animate transitions from one CSS style configuration to another. They consist of two components:
  1. a **style** describing the CSS animation.
  2. a **@keyframe** that indicates the start and end states of the animation's style, as well as possible intermediate waypoints.

Some in depth examples here: <https://www.w3schools.com/css/css3_animations.asp>

**CSS Variables**

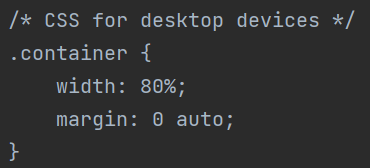
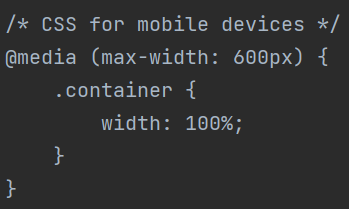
* CSS variables allow you to **define reusable values in your stylesheets.** They are set using custom property notation (e.g., --main-color: blue;).
  + Once you have defined a CSS variable, you can use it elsewhere in your CSS by using the **var()** function.
  + Variables are typically declared in the **:root selector**, making the variable globally available to the entire document.



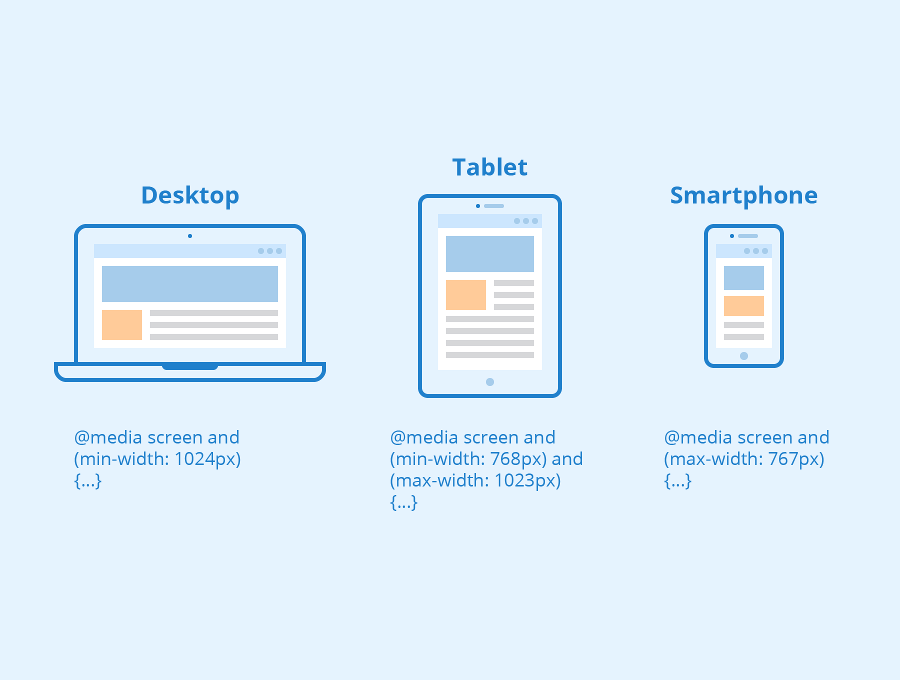
In this example, we’re setting a variable called “--main-color" to “blue”. We can then use it throughout the entire stylesheet. (in the overall body in this case).

**Media Queries**

* Media queries enable you to apply different styles depending on the characteristics of the viewport where your website is being viewed.
  + This is used to create **responsive designs** that adapt to different screen sizes, especially in the context of browser vs mobile styling.
  + We define them with **@media** and a condition. The browser will already know the size of the viewport, letting it evaluate these conditions.

* When the screen size is 600px or less (which is common for **mobile devices**), the media query is activated and the width of the .container class becomes 100%.
  + This makes the content take up the full width of the screen on smaller devices, making it **more readable and user-friendly.**



**Not on QC But I recommend looking into---------------------------**

**Bootstrap**

* Why write your own CSS when you can **STEAL** someone else’s? Bootstrap basically lets you do this. It’s an open-source framework with a “mobile first” design approach. Bootstrap and its competitors are used pretty much everywhere nowadays
* It uses HTML 5 and CSS3 to **give predefined styling to web pages**. This styling can be resized and displayed on most any sized screen.
* Bootstrap can be added to your webpage in two main ways:
  + You can download it directly from getbootstrap.com
  + You can simply **use a link tag to the bootstrap CDN in your header**
  + **Linked here for your convenience:**
    - [**https://getbootstrap.com/**](https://getbootstrap.com/)
    - **<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.0-beta1/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-0evHe/X+R7YkIZDRvuzKMRqM+OrBnVFBL6DOitfPri4tjfHxaWutUpFmBp4vmVor" crossorigin="anonymous">**

**Bootstrap Grid System**

* Bootstrap uses a layout called the grid system. It consists of containers which contain rows, which themselves contain columns. **This layout will be responsive to window size changes,** which means it will change to fit the window size.
  + **This is one of the most common ways to achieve responsive web design.** It’s RESPONSIVE to window size changes!
* **Containers** are the basic layout element used to wrap the content of a web page. There are two container classes:
  + Container - responsive fixed-width container
  + Container-fluid - full width container that spans the entire viewport.
    - A viewport is **a region of the screen used to display a portion of the total image to be shown**. It sizes the screen to match the screen size of the user
* **Rows** are horizontal slices of a container. They are created with the “row” class. Rows have a base size of 12 (the typical width of a webpage)
* **Columns** have a dynamic width determined by their size relative to the size of the row in the container’s view.
* Columns can be assigned a size. If the size of the columns exceeds 12, the column exceeding the size will be moved below the columns that do not.
  + So we can have a 3, a 3, and a 6, and they’d all print inline
  + class=”col-sm-3” col is for column, the sm (small) represents the default size of the column, and the 3 is the relative size compared to the row’s size of 12.

**Common Bootstrap Styled Elements**

* Bootstrap has many classes to style all sorts of HTML elements
* **Bootstrap Tables** are predefined styling for tablesthat can be declared again by class.
  + class=”table table-dark” creates a table with a dark background
  + table-bordered = add a border around the table
  + table-striped = adds zebra striping to the table
* Bootstrap has a bunch of **button classes** that let you color your buttons.
  + E.g. btn-danger will give you a red button.
* Bootstrap has a **navbar** class for styling navbars.
* MANY MANY more.... This is the entire list of them: <https://getbootstrap.com/2.3.2/components.html>