week::nine

Working with Class & ID Selectors, and CSS Layout

Class Selectors

Overview

- Allow you to *selectively* apply formatting to an HTML page element.
- Classes can be **applied to multiple elements** within a page.
- Classes are identified by a single period (.) in front of the name.

When to Use

- To target specific areas of a page and style the HTML elements with (or within) that class with styles different than the rest of the page.
- Use when the hierarchy of HTML elements is not specific enough to style with CSS.
- Classes should be used before IDs—creating re-usable code is always better than one-off code.

Code

```
<style>
   .photogallery   { border: 1px solid gray; }
   .photogallery p { font-size: .9em; }
</style>
<div class="photogallery">
   <p>Some content goes here</p>
</div>
<div class="photogallery">
   <p>Yet some more content goes here</p>
</div>
<div>
   <p>Some other content is here. It also looks different than the text above.</p>
</div>
```

ID Selectors

ID Styles

- IDs can be applied to **only one element** within a page.
- They provide access to all of the same options as Classes.
- ID styles are identified by the pound sign (#) in front of the name.

When to Use

- When an element is the only one of its kind on a page and will always be the only one.
- In most circumstances, you will use these the *least* of the four selectors discussed in class.

Code

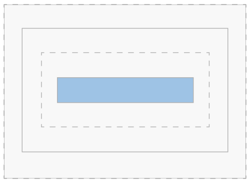
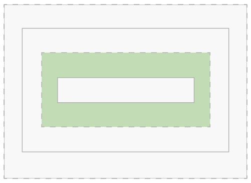
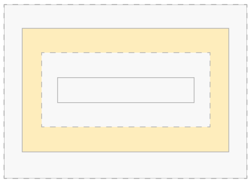
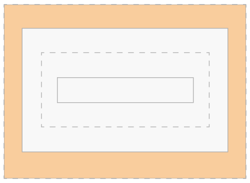
```
<style>
   nav ul li a#home { font-weight: bold; }
</style>
<nav><ul>
   <li><a href="index.html" id="home">Home</a></li>
   <li><a href="about.html" id="about">About Us</a></li>
</ul></nav>
```

Introducing the CSS Box Model

In Web layout, everything is a box

Components

- All **block-level** elements include these four controllable components:

			
Content Area The area in which HTML content may be displayed.	**Padding** Space between the content area and the border. Can be styled with background images and background colors; No HTML content.	**Border** A border surrounding all sides of a box.	**Margin** Empty white space between the border and other elements. Cannot be styled other than the amount of space. No HTML content

Key CSS Layout Properties

`width`

- Best set in percentages; can also be ems and pixels.
- `width: 65%;`

`height`

- *Avoid* setting height unless necessary
- Set in px, em or vh
- `height: 20em;`

`max-width` *and* `min-width`

- Set in ems, pixels, or percentages
- `max-width: 61.5em;`

`max-height` *and* `min-height`

- *Avoid* setting height unless necessary
- Set in px, em or vh
- `min-height: 10em;`

`display`

- Controls the display of and role that an element plays in a page.
- Values include *block*, *inline*, and *none*
- `display: block;`
`display: none;`

`border`

- Set width in px; Use solid as style

-
  ```
  border: 1px solid red;
  border-bottom: 2px solid #ccc;
  border-width: 2px 0;
  ```

`margin`

- Shorthand options:
  - `top right bottom left`
  - `top/bottom left/right`
  - `all_sides`

-
  ```
  margin: 0 0 1em .5em;
  margin-right: 8%;
  ```

`padding`

- Shorthand options:
  - `top right bottom left`
  - `top/bottom left/right`
  - `all_sides`

-
  ```
  padding: 1em 0;
  padding-top: 2.25em;
  ```

`float`

- left, right, or none
- Use as few of these as possible
- `float: right;`

`clear`

- Clears floats
- both, left, right
- `clear: both;`

CSS Units of Measurement (not on exam)

Overview

- When entering numbers in CSS, always specify the units of measurement.
- The only time you can skip the units is with zero.

Percentages (`%`)

- Used mainly for width.
- Calculates a value based on the available space using the available space within the current element.
- Modern layout techniques (responsive web design) are moving towards the use of percentages for defining widths and heights, using pixels less and less.

Ems (`em`)

- Most common unit of measurement for type, including size and leading (`line-height`).
- Allow for flexible typography and layouts whose size is based on the current default font size of the device being used to view the page.
- Based on the relative size of type in the containing element.
- Also used for width and height in some circumstances.

Root ems (`rem`)

- Same as ems, but calculated from the *root* element of the page.
- Eliminates variations of size based on their parent element's type size.
- Can be a more predicable method of setting type sizes, but comes with its own set of issues.

Viewport Width & Viewport Height (`vw & vh`)

- Similar to percentage, but are always sized relative to the root element.
- 1vw equals 1% of the root element; 100vw equals 100%
- Very helpful for elements that need to take up a specific part or all of a device's screen.
  - Viewport width and height are better units than percentages for this because they are based solely on the Viewport, not the parent element's size.

Pixels (`px`)

- Common for widths, heights, margin, padding, and borders.
- Is technically a relative unit of measurement, though most desktop browsers treat it as absolute (monitor pixels).
- On HiDPI screens, the pixel is abstracted based on the PPI of the device's screen.

Recommendations:

- Use **ems** for all typography and margin/padding rules where the exact spacing is not critical, and where it's best that sizes scale with the page typography.
- Use **percentages** (%) as much as possible for widths, as they are inherently flexible, adapting to different screen sizes as needed. Percentages help create flexible (and ultimately, responsive) layouts—ones that are *not* exactly the same across every browser and device.
- Avoid percentages for heights as they are unpredictable.
- Use **pixels** (px) or **percentages** (%) when exact spacing and layout is required and setting widths of some container elements.
  - For percentages, divide the pixel values by the pixel width of the container elements to get exact percentage values.

Additional Reading

- `https://css-tricks.com/the-lengths-of-css/`

Common CSS Layout Properties

This page presents just an overview of a few common CSS properties. We will explore all of these much more in-depth in future classes.

Name	Property	Description	Example CSS Code
Background Color	`background-color:`	Sets the background color of an element	`background-color: #dfdfdf;`
Element Border	`border:`	Creates a border around an element	`border: 1px solid #ccc;`
Rounded Corners	`border-radius:`	Rounds the corners of elements	`border-radius: 20px;`

Padding	`padding:`	Space between content and the edge of the box	`padding-left: 20px;`
			`padding-top: 13px;`

Margin	`margin:`	Transparent space around a content box	`margin-right: 45px;`
			`margin-bottom: 100px;`

Element Display Type	`display:`	Controls the display type of an element. In this example, the CSS code hides the element.	`display: none; /*Hides element*/`
Element Width	`width:`	Width of a block element	`width: 300px;`
Element Height	`height:`	Height of a block element	`height: 400px;`
Maximum Width	`max-width`	Sets the maximum width of an element	`max-width: 68.75em;`
Minimum Width	`min-width`	Sets the minimum width of an element	`min-width: 50%;`
Maximum Height	`max-height`	Sets the maximum height of an element	`max-height: 10em;`
Minimum-Height	`min-height`	Sets the minimum height of an element	`min-height: 2em;`
Bulleted List Style	`list-style-type:`	Controls the appearance of bullets in <ul><li> lists.	`list-style-type: circle;`
Element Float	`float:`	Removes an element from the document flow.	`float: right;`