Full Stack Web development

Lecture 7: Advanced CSS

Today:

- Advanced CSS
  - Selectors Revisited
  - Pseudo-classes and Pseudo-elements
  - CSS Box Model
  - Positioning and Display
  - Size Units
- Lab: Building a dropdown navigation using CSS and HTML

Selectors: Revisited

Recall:

`* {}` – selects all elements

`p {}` – selects all p elements

`.className {}` – selects all elements with class="className"

`#elementID {}` – selects an element with id="elementID"

`p.hidden {}` – selects all p elements with class="hidden"

`p .hidden {}` – selects all elements **inside** p tags with class="hidden"

`p, .hidden {}` – selects all p elements **AND** all elements with class="hidden"

Selectors: Additional Combination Selectors

Recall:

`p > .hidden {}`

– selects all elements with class=”hidden” that are **children** of p

`p + .hidden {}`

– selects all elements with class=“hidden” that are **immediately after** p

`p ~ .hidden {}`

– selects all elements with class=“hidden” that are **siblings** of p (same level)

`a[href=“https://disney.com”] {}`

-selects all a elements with an href=”https://Disney.com”

Selectors: Pseudo-classes

Have you seen links that change color when you hover over them or they become underlined? This is through pseudo-classes.

```
a {
   text-decoration:none;
}
a:hover {
   text-decoration:underline;
}
```

Selectors: Pseudo-classes

Links (works on div too!)

:link

:visited

:hover

:active

Specific Elements

:first-child

:first-of-type

:not(selector)

:nth-child(n)

:nth-last-child(n)

Forms

:checked

:disabled

:enabled

:focus

:required

:optional

:valid

:invalid

:read-only

:read-write

Selectors: Pseudo-elements

Another type of selector (notice the double colon):

```
p.headline::after { content: “Date: “ }
```

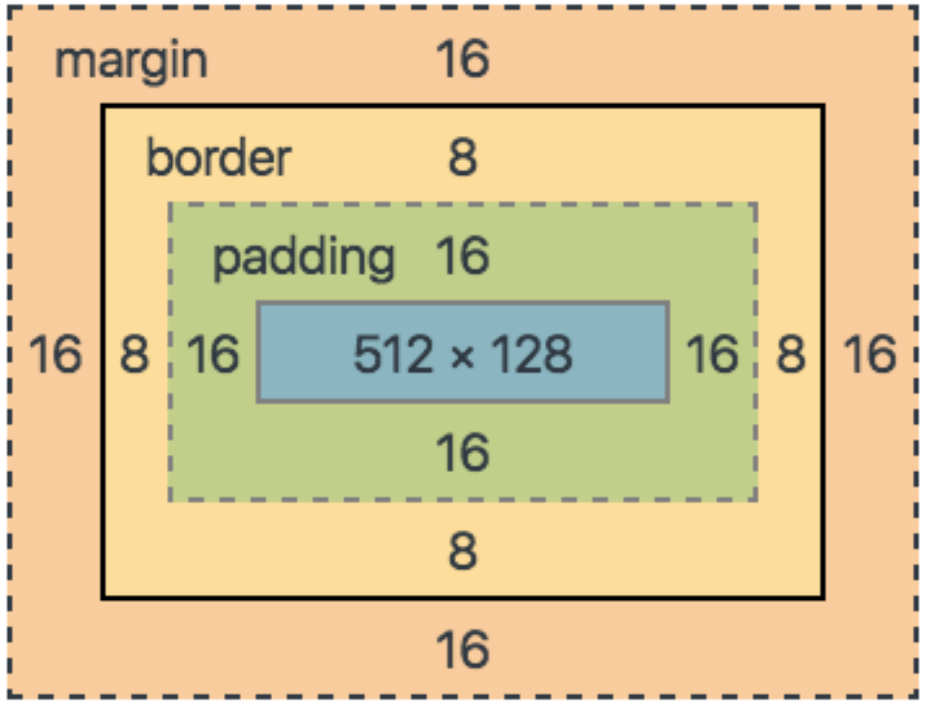
::after – insert content after an element

::before – insert content before an element

::first-letter – select the first letter of an element

::first-line – select the first line of an element

CSS Box Model



- As you've been playing around with placement of items you've probably struggled with the CSS box model.
- Each element of CSS can be considered a box that has margin, border and padding.
- Let's play around with some margin and padding and see why it's important.

Positioning and Display

The `position` property:

`static` – this is the default

`relative` – relative to normal position

`fixed` – relative to the viewport

`absolute` – relative to nearest positioned parent, or the document body

`sticky` – positioned based on scroll positioning.

positioning must be used in conjunction with top, bottom, left, and right CSS properties.

The `display` property:

- `block` – always starts on a new line and takes up the full width available to it.
- `inline` – can start anywhere and only takes up as much space as needed
- `none` – removed from the page

The `visibility` property:

`visible` – the default option

`hidden` – hidden from the browser but the space allocated still displays in the browser.

Size Units

- Many CSS properties require sizing to be defined. Up until now we have used px (pixels). There are other alternatives.
- Fixed Sizes:
  - px (pixels) 96dpi
  - pt (points) 72dpi
  - pc (picas) 1pc = 12pt
- Relative Sizes
  - em (relative to font-size of current element)
  - % (relative to the parent element)
  - vw (relative to the viewport width)
  - vh (relative to the viewport height)

It is a best practice to use relative sizes in production environments as they provide the best support for accessibility and RWD

Responsive Web Design

- One last thought on Responsive Web Design in this segment of the course.
  - You already have all the tools necessary to build it (media queries, positioning, % based sizing), we will add a few more tools as we progress in the course, but it’s worth noting that as you build assignments from here on out, you should think about how it looks across all devices.

- Here’s a great beginners tutorial if you want to learn more.

https://www.w3schools.com/html/html_responsive.asp

Lab – Building a Nav Bar with only CSS

Demo – in class we will build a CSS drop-down nav bar using only the skills we have learned.

Homework – Due 10/10

Going forward all homework will be submitted through GitHub. You must upload your code into your homework folder of your classwork repository, **not** the group project, and then submit a Pull Request to dadkins20.

- Add an interactive navigation to your about me page. Using CSS positioning, have it fixed at the top of the page as the user scrolls down. It should have at least one top level drop-down and one secondary level drop down.
- Continue working on your group project
- Not homework but a reminder: practice writing as much code as you can. It’s the best way to learn.

This concludes our work on Cascading Style Sheets, from here we move into the basics of JavaScript, which will be a major focus for the majority of the class from here on out. Make sure you have an excellent grasp of HTML and CSS.