Week 7:
CSS Part 2

ATLS 2200 (Web)
Spring 2022

roadmap

TODAY...

1. Wrapping Up Last Week – Inheritance
2. Directory Structure
3. CSS Padding, Margins, and the Box Model
4. CSS Layouts
5. Wrap-up + Next Steps

WHILE YOU'RE GETTING SETTLED

Make sure to check in via Canvas ("Week 7 Lecture Check-in").

wrapping up week 6

this topic is *inherited* from last week

CSS

INHERITANCE

Inheritance refers to the property for child elements to inherit property values from their parent elements.

This is where paying attention to **nesting** is important!

- An element is a **parent element** to any element nested within it
- An element is a **child element** to any element it is nested within

Not all property values are subject to inheritance.

CSS

INHERITANCE

In the following, which elements are parent elements? Which are child elements? **Note, this is heavily simplified HTML.*

```
<html>
  <body>
    <p>This is my page</p>
    <ul>
        <li>List item 1</li>
        <li>List item 2</li>
    </ul>
  </body>
</html>
```

directory structure

a quick refresher.

directory structure

DIRECTORY STRUCTURE

I’m seeing a lot of weird, wacky directory structures and naming conventions.

Let’s get on the same page.

The **root** folder is the folder that your website exists in. It should be named in the following syntax:

<username>.githib.io

Where <username> is your GitHub username.

directory structure

DIRECTORY STRUCTURE

Within your root folder, you can have files and subdirectories.

What file is the homepage of your website?

Answer here.

Where should that file be in your folder?

Answer here.

directory structure

DIRECTORY STRUCTURE

Where should images go?

Answer here.

Where should CSS go?

Answer here.

directory structure

DIRECTORY STRUCTURE

Let’s say you have a CSS stylesheet – named “styles.css” in a folder called “css”.

You are working in your index.html, but want to use the styles.css in index.html.

What is the file path you would type in the href attribute of the link tag to do this?

Answer here.

Why?

directory structure

DIRECTORY STRUCTURE

Let's say you have a CSS stylesheet – named "styles.css" in a folder called "css".

You are working on assignment7.html, which is in a folder called "assignments", but want to use styles.css.

What is the file path you would type in the href attribute of the link tag to do this?

Answer here.

Why?

directory structure

DIRECTORY STRUCTURE

You are working in your index.html, but want to link to a file “test.html” in a folder called “test-code”.

What is the file path you would type in the href attribute of an <a> tag to do this?

Answer here.

Why?

the box model

nesting boxes

CSS

THE BOX MODEL

Everything in CSS has a box around it. There are two broad types of boxes – **block boxes** and **inline boxes.**

Boxes also have an **inner display type** and **outer display type.**

The **inner display type** dictates how the content within the box is laid out.

The **outer display type** dictates if it is a **block** or **inline** box.

CSS

THE BOX MODEL

The box model applies to **block boxes**, and some parts apply to **inline boxes.**

A box is made up of 4 parts:

- **Content box** – the area where content is displayed
- **Padding box** – padding sits around the content, appearing as white space
- **Border box** – the border wraps the content and padding
- **Margin box** – the margin wraps content, padding, and border as white space outside of the box.

CSS

THE BOX MODEL

Block boxes:

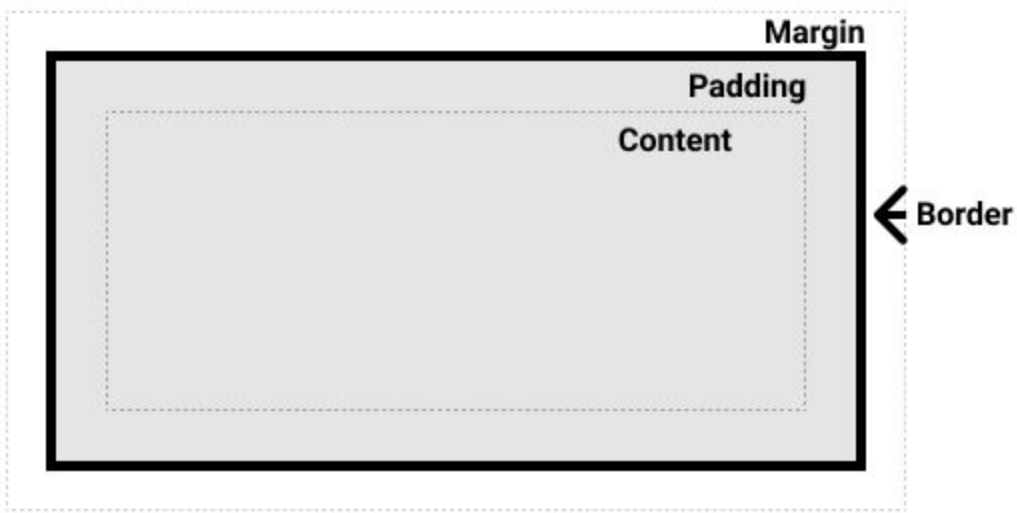
- The box will break onto a new line
- The box will extend in the inline direction to fill the space available in its container. In most cases this means that the box will become as wide as its container, filling up 100% of the space available.
- The width and height properties are respected.
- Padding, margin and border will cause other elements to be pushed away from the box

CSS

THE BOX MODEL

Inline boxes:

- The box will not break onto a new line.
- The width and height properties will not apply.
- Vertical padding, margins, and borders will apply but will not cause other inline boxes to move away from the box.
- Horizontal padding, margins, and borders will apply and will cause other inline boxes to move away from the box.


Margin
Padding
Content
Border

CSS

THE (STANDARD) BOX MODEL

When we make a new box (be it a , a <p>, or some other element, we are given 5 basic attributes to work with:

- **Width** - takes 1 value; similar to font-size… many options!
- **Height** - takes 1 value; similar to font-size… many options!
- **Margin** - takes 1 or 4 values. 1 value: the margin is that value on all sides; 4 values: sets the top, right, bottom, and left
- **Padding** - takes 1 or 4 values. 1 value: the margin is that value on all sides; 4 values: sets the top, right, bottom, and left
- **Border** - takes 1-3 values; border-width, border-style (required), border-color

Width and height control the **content box**. Any padding and border is added to get the total space taken up by the entire box.

```
.box {
  width: 350px;
  height: 150px;
  margin: 10px;
  padding: 25px;
  border: 5px solid black;
}
```


height: 150px
width: 350px

css layouts

moving boxes around

CSS

CSS LAYOUTS

Layouts let us control where elements on a web page are.

There are four types of layout techniques we're concerned with:

- Normal flow
- The display property
- Flexbox
- Grid

CSS

CSS LAYOUTS

Normal flow is how browsers will lay out your pages by default.

(this should be familiar, and consequently boring).

CSS

CSS LAYOUTS

The **display** property lets us change how something displays.

For example, we could change an anchor element <a> (which is normally an inline element) to a block element by specifying the following in our CSS:

```
a {
    display: block;
}
```

CSS

CSS LAYOUTS

Flexbox lets us lay things out in one dimension; either vertically or horizontally. To set up flexbox, you set the display attribute for the parent element to be **flex.** (display: flex;)

CSS

CSS LAYOUTS

Grid layout lets us set up elements in rows and columns (2 dimensions).

To use grid layout, we set the display attribute of the parent to be **grid** (display:grid;).

We might also use **grid-template-columns, grid-template-rows,** and **grid-gap** to set how many columns and rows there should be, and what the gap should be between cells of the grid.

CSS

CSS LAYOUTS

A not-subtle look ahead to next week – grids and flexboxes are one way of designing basic responsive sites.

We'll get practice with all of these things over the next two weeks.

roadmap

WHAT'S NEXT?

In recitation this week – practice CSS!

Quiz 5 opens at 10:45 AM; due before recitation.
Assignment 7 opened at 10:30 AM; continue on it in recitation; due Sunday by 11:59 PM.

DAILY NOTE

We'll post the Daily Note in Slack in **#atls-2200-web-spring-2022.**

Make sure to do it now.