03: Web Design

Advanced Layouts

Session Outline

- Recap
- CSS Box Model
- CSS Display Property

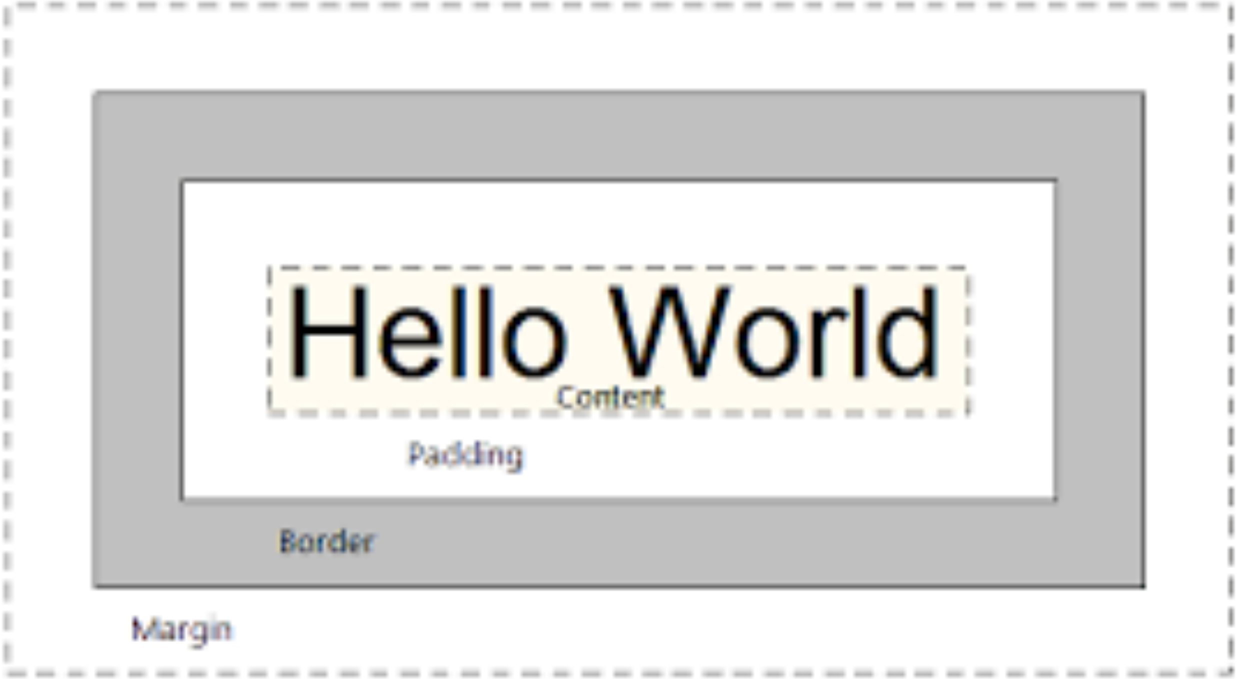
Review

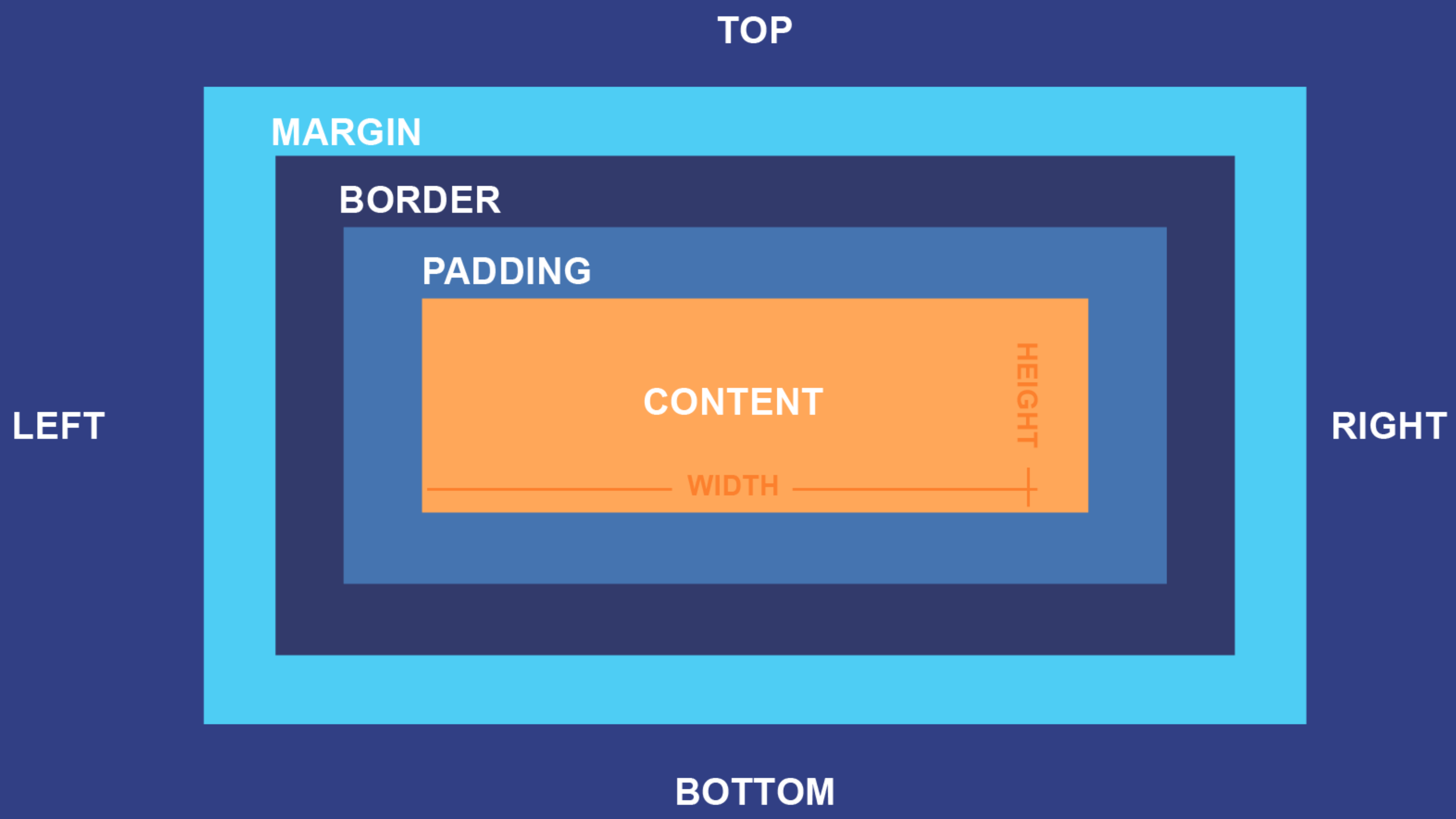
- Semantic HTML
- CSS Syntax
- HTML ID and Class attributes for styling (# for IDs; . for classes)

Building Fluid Layouts with CSS

Understanding the Box Model

- Every element can be understood as a box in CSS
- Which means, every box is a rectangle of varying widths and heights




TOP
MARGIN
BORDER
PADDING
LEFT
CONTENT
HEIGHT
WIDTH
RIGHT
BOTTOM

Key types of boxes

- Block types
- Inline types
- Inline-block types

display: block

- The element generates a block box, generating line breaks both before and after the element when in the normal flow.
- Takes up the full width of its parent container
- Allows **width and height** to be set
- Elements with default display:block
    - div, h1, p, section

```
div {
       display: block;
}
```

display: inline

- The element generates one or more inline boxes that do not generate line breaks before or after themselves. In normal flow, the next element will be on the same line if there is space.
- Only takes up as much width as necessary (no full-width behavior)
- **Width and height properties have no effect**
- Common inline elements:
    - <span>, <a>, <strong>, <em>

```
span {
        display: inline;
}
```

display: inline-block

- The element generates a block box that will be flowed with surrounding content as if it were a single inline box (behaving much like a replaced element would).
- Works both like an inline and a block property type

```
.my-class {
        display: inline-block;
}
```

Comparision

Property	Block	Inline	Inline-block
Starts on new line	Yes	No	No
Width/Height	Can be set	CanNOT be set	Can be set
Takes full width	Yes	No	No

Exercise 1

- Create an HTML file with the following elements: A header with a title
  - A navigation menu with 4-5 items
  - Three content sections, each with a heading and a paragraph
  - A sidebar with a list of 3-4 items
  - A footer with copyright information
- Style the elements using CSS to achieve the following layout: The header should span the full width of the page (block)
  - Navigation menu items should be horizontally aligned (inline or inline-block)
  - Content sections should be arranged vertically (block)
  - The sidebar should be positioned to the right of the content sections (inline-block)
  - Footer should span the full width of the page (block)
- Experiment with different display properties for various elements and observe how they affect the layout.

CSS Positions

- The position property specifies how an element is positioned in a document
- It works with top, right, bottom, and left properties to determine the final location

position: static

- Default positioning for all elements
- Elements are positioned according to the normal flow of the document
- top, right, bottom, and left properties have no effect

position: relative

- Element is positioned relative to its normal position
- Can use top, right, bottom, and left to offset from normal position
- Other elements are not affected

position: absolute

- Element is removed from the normal document flow
- Positioned relative to its nearest positioned ancestor or initial containing block
- Can use top, right, bottom, and left to specify the position

position: fixed

- Element is removed from the normal document flow
- Positioned relative to the browser window
- Stays in the same place even when the page is scrolled

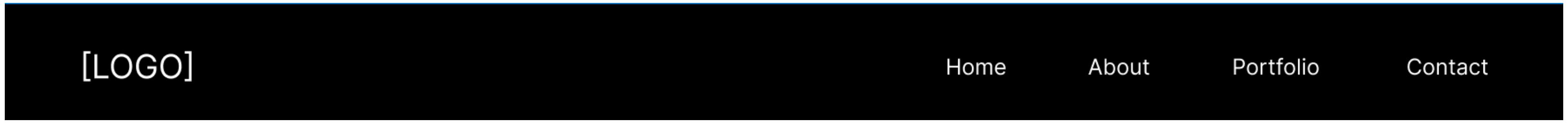
position: sticky

- Element is positioned based on the user's scroll position
- Toggles between relative and fixed
- Stays in place when a specified threshold is met in the viewport

Walk-through

Exercise 2

- Setup a week-3 project on your computer
- Create an index.html to contain all your HTML code
  - Using `<header>` and `<nav>` elements as parent containers, create a navigation menu
- Create a style.css file
  - Link it with your HTML file
- Create a navigation bar that stays at the top of the page when scrolling.


[LOGO]
Home
About
Portfolio
Contact