HTML & CSS: LEVEL 1

Instructor: Beck Johnson

Week 4

SESSION OVERVIEW

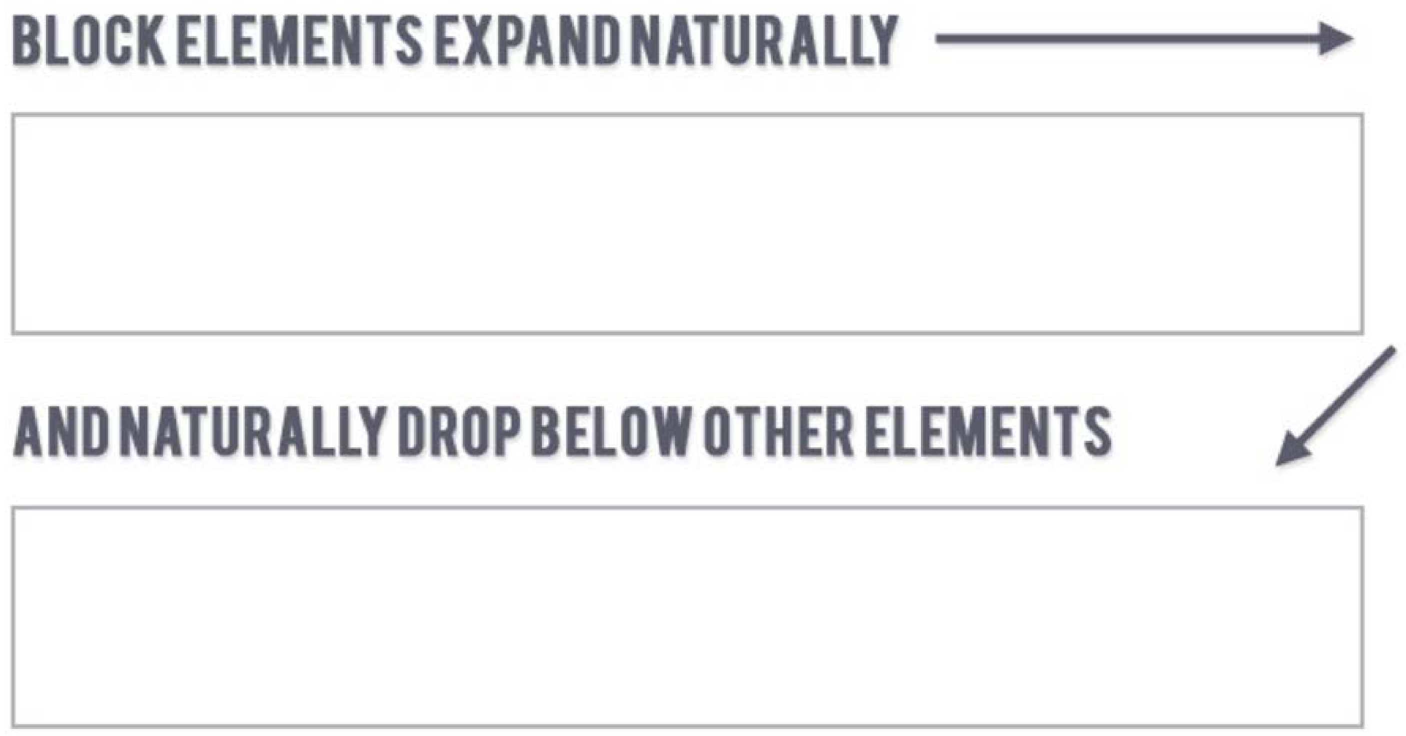
- Review box model of CSS, classes and ids
- Floats and positioning
- Semantic elements and why they matter
- Using web fonts

REVIEW!

{} BLOCK VS INLINE ELEMENTS

BLOCK ELEMENTS

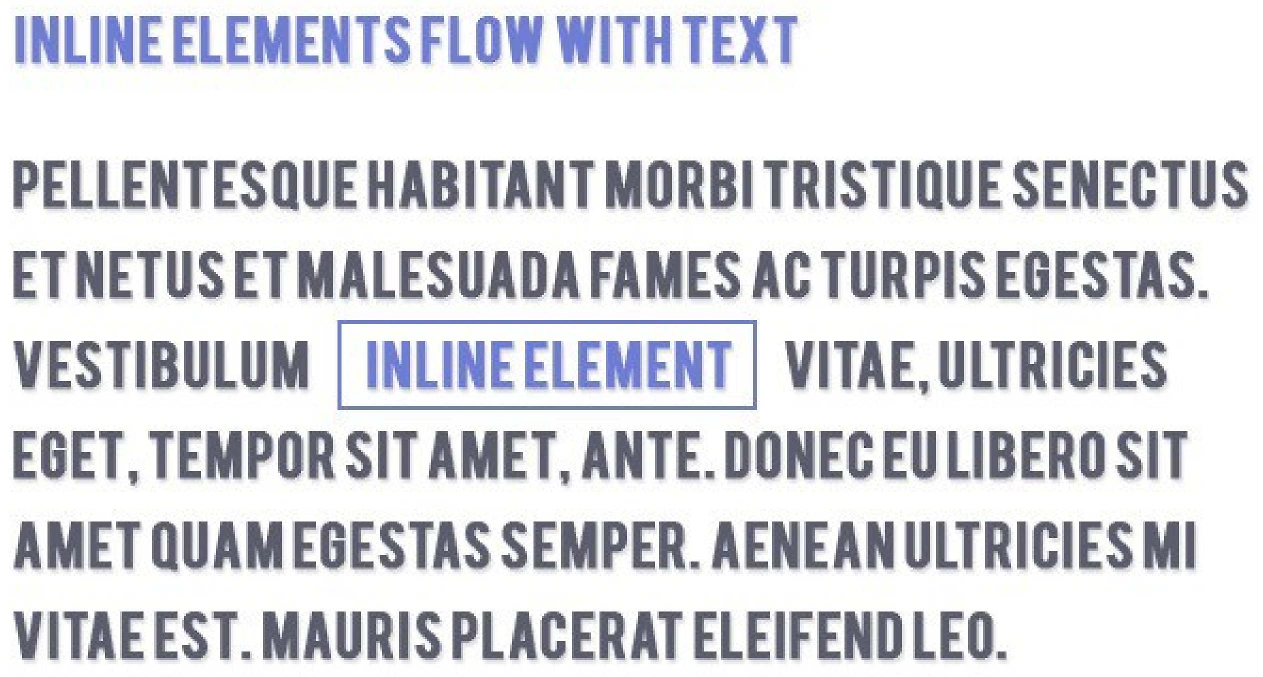
- Expand naturally to fill their parent container
- Can have margin and/or padding


BLOCK ELEMENTS EXPAND NATURALLY
AND NATURALLY DROP BELOW OTHER ELEMENTS

{} BLOCK VS INLINE ELEMENTS

INLINE ELEMENTS

- Flow along with text content
- Ignores height, width, top margin, and bottom margin
- Honors left and right margins (and any padding)


INLINE ELEMENTS FLOW WITH TEXT

PELLENTESQUE HABITANT MORBI TRISTIQUE SENECTUS ET NETUS ET MALESUADA FAMES AC TURPIS EGESTAS. VESTIBULUM INLINE ELEMENT VITAE, ULTRICIES EGET, TEMPOR SIT AMET, ANTE. DONEC EU LIBERO SIT AMET QUAM EGESTAS SEMPER. AENEAN ULTRICIES MI VITAE EST. MAURIS PLACERAT ELEIFEND LEO.

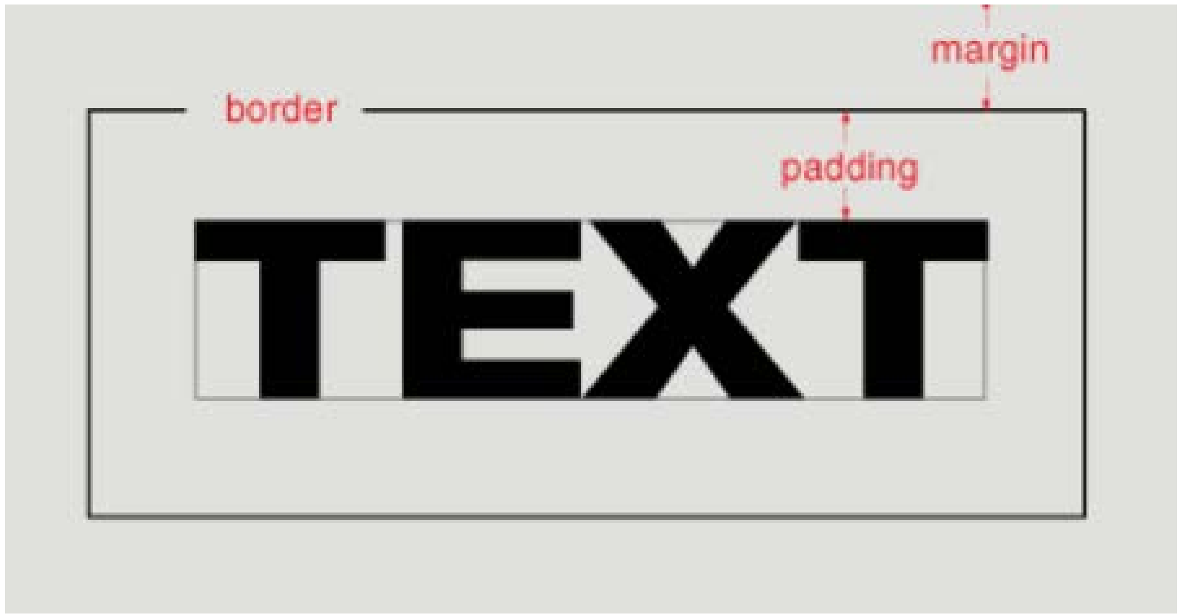
{} CSS BOX MODEL

CONTENT: stuff in the box

PADDING: bubble wrap and packing peanuts

BORDER: sides of the box

MARGIN: space between multiple boxes



{} CLASSES

{} IDS

QUESTIONS?

WEB LAYOUTS

WEB LAYOUTS

- Before CSS, we used **<table>** elements to make layouts (bad!!)
- But now, with the advancements of CSS, we can use a variety of properties to arrange elements on the screen by adjusting the flow of the page.
- Basically, you can put elements anywhere.
  - (this can be a good and a bad thing!)

3 WEB LAYOUT PROPERTIES

- **display**: for dictating how elements behave within the box model. **(block, inline, inline-block)**
- **float:** for moving elements around within the page flow.
- **position:** for moving elements in and out of the page flow altogether.

THE DISPLAY PROPERTY

- Remember **block, inline, and inline-block** elements?
- You can tell elements to display differently using the CSS **display** property.
- Example:
  - display: block;
  - display: inline;
  - display: inline-block;

WHY USE DISPLAY?

- Make a **link** look more like a button.
- Add padding and margins to an inline element like a span.
- Make navigation links display horizontally.
- Make any text elements display inline.
- Make divs behave like images.

FLOATS!!!!!!

CSS FLOATS

CSS **floats** can be tricky to grasp, but are foundational in creating complex web layouts.

The **float** CSS property specifies that an element should be taken from the normal flow and **placed along the left or right side of its container**, where text and inline elements will wrap around it. (MDN)

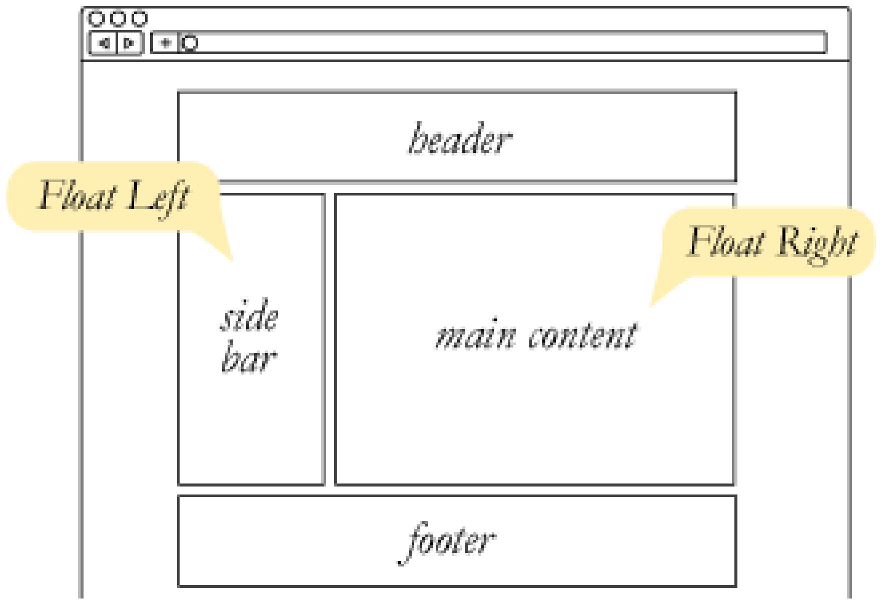
CSS FLOATS

Easiest way to offset content like divs, images, pullquotes, or other elements within the flow of a document.

Requires that an element have **display: block;**

Three possible values: **left, right, none;**

float: none; is browser default.

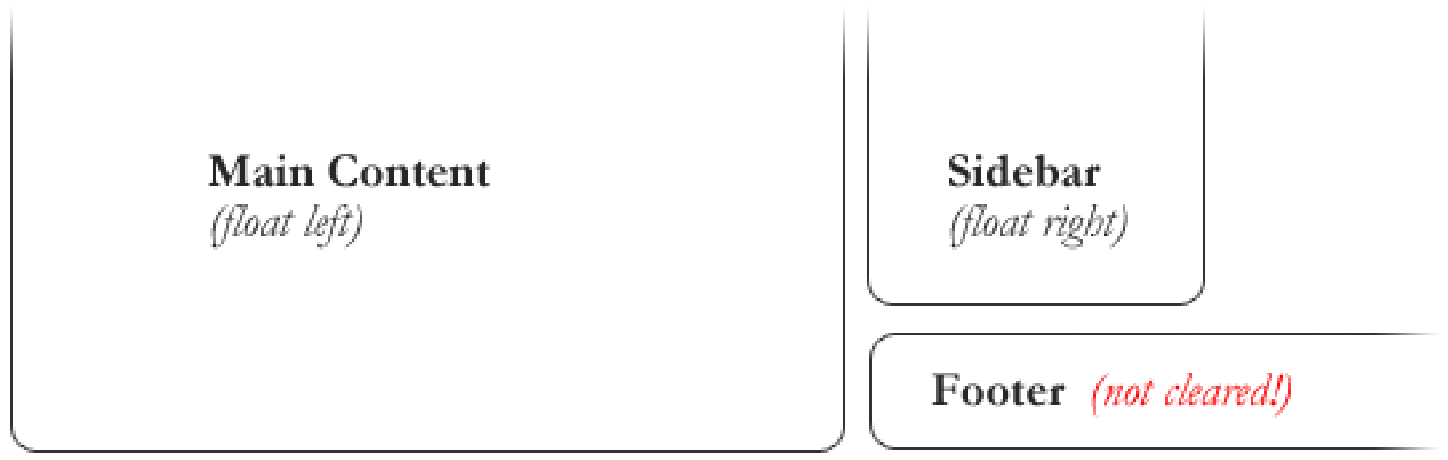


```
#sidebar {
   float: left;
}
```

DEMO

THE CLEAR PROPERTY

- CSS float's sister is the **clear** property.
- An element that has the **clear** property set on it will not move up adjacent to the float like the float desires, but **will move itself down past the float**.



(uncleared float)

THE CLEAR PROPERTY

- One of the trickiest things about floats is when to stop "floating"

- You can give any element the **clear:both;** style to prevent it from floating.

```
#sidebar {
    clear: both;
}
```

THE MAGIC FLOAT FIX

- The most common fix today though is the self-clearing float.
- You can use a **pseudo-element** on the parent of the floated elements to create a **“self-clearing”** float.

```
.clearfix:after {
   content: “”;
   display: block;
   height: 0;
   clear: both;
}
```

DEMO

CSS POSITIONING

CSS POSITIONING

- The **position** property lets us arrange elements:
    - In relation to the normal flow (**relative**)
    - In a very specific place outside of the flow or within a **relative** element. (**absolute**)
    - In relation to the browser window (**fixed**)
- How **position** is applied depends on where the element is in the flow by default.

CSS POSITIONING

We can dictate where elements go on the page down to the pixel!

left, right, top, bottom

Can tweak positively or negatively

```
nav {
   position: absolute;
   right: -10px;
   top: 30px;
}
```

POSITION: FIXED

- **position: fixed;** is a way to make content "stick" to the browser window, regardless of where the user scrolls.
- Commonly used to make headers, nav, or footers that follow the page as it scrolls.

```
nav {
   position: fixed;
   width: 100%;
   left: 0;
   top: 0;
}
```

SEMANTIC ELEMENTS

SEMANTIC ELEMENTS

Semantics is the study of the meanings of words and phrases in a language.

Semantic elements = elements with a meaning.

A semantic element clearly describes its meaning to both the **browser** and the **developer**.

Non-semantic: <div>, <span>
Semantic: <form>, <article>, <section>

SEMANTIC ELEMENTS (HTML5)

<article>

<aside>

<footer>

<header>

<main>

<mark>

<nav>

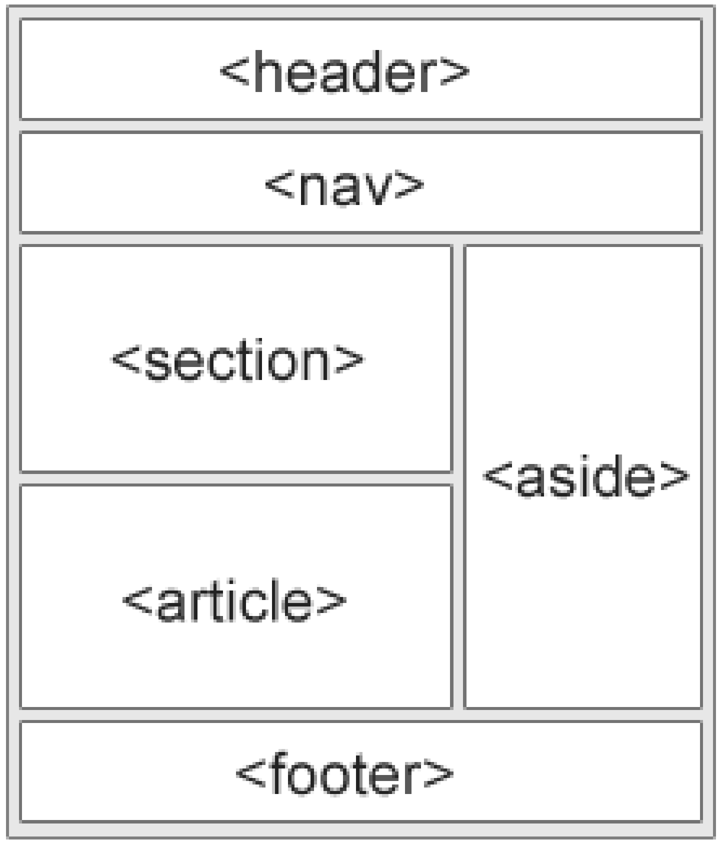
<section>

<progress>

<time>

FURTHER READING

- http://www.w3schools.com/html/html5_semantic_elements.asp



PRACTICE TIME!

ASSIGNMENT

- Find a free, open source font and use it on your site.
- Update your website to use semantic elements.
- Float at least one element – remember to clear it afterwards!

“HOMEWORK”

- Practice!

- Optional: read chapters 15 and 17 of *HTML and CSS: Design and Build Websites*