

HTML



CSS



HTML & CSS: LEVEL 1

Instructor: Beck Johnson

Week 5



SESSION OVERVIEW

- Review fonts, floats and positioning.
- Tables
- Creating a form
- Next steps: Javascript and jQuery
- Evaluations



REVIEW!

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.

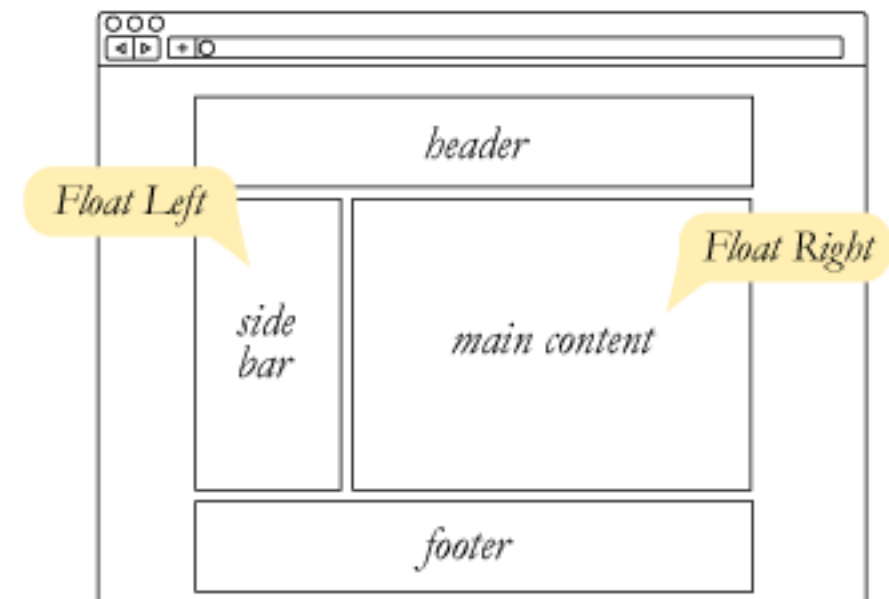
CSS FLOATS

- 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

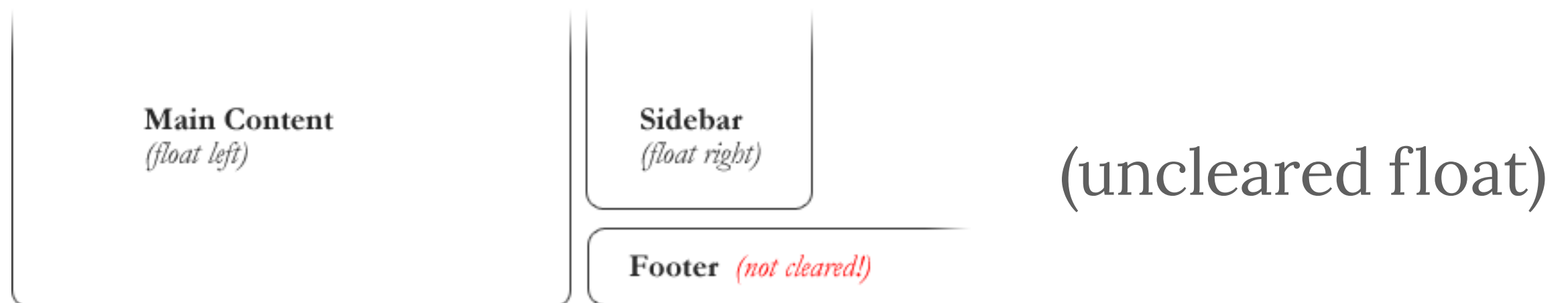
- Easiest way to offset content like divs, images, pullquotes, or other elements within the flow of a document.
- Requires that an element have **display: block;**
- Three possible values: **left, right, none;**
- **float: none;** is browser default.

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



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**.



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;  
}
```

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;  
}
```

QUESTIONS?



TABLES

WHY TABLES?

We use tables to present data in a tabular format.

- Listings of people, addresses, etc.
- Financial data
- Sports stats
- Product features

TABLE ELEMENTS

- `<table>` wraps the whole table.
- `<thead>` wraps the header cells.
- `<tbody>` wraps the main data cells
- `<tr>` creates a **row** of table cells.
- `<th>` creates a **table header** cell for a column or a row. (in `<thead>`)
- `<td>` creates a regular **table data** cell within a row.

A BASIC TABLE

```
<table>
  <thead>
    <tr>
      <th>Column 1 Header</th>
      <th>Column 2 Header</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>Column 1 Data Cell</td>
      <td>Column 2 Data Cell</td>
    </tr>
  </tbody>
</table>
```

<> THEAD AND TBODY

- Separate body from header of table.
- By default, do not affect layout of the table.
 - But can provide more scope for styling in CSS.
 - Semantically better as well.
- Provide semantic information for browsers, printers, and displays.

<> TD ELEMENT

- Used to provide a cell in a table row.
- Two important attributes:
 - **colspan:** defines how columns the cell should cover.
 - **rowspan:** defines how rows the cell should cover.

STYLING TABLES

- Way back when, we styled tables inline (no!)
- We now use CSS for all styles for tables.
- All of our box model styles can be applied.



DEMO!



FORMS

FORMS

Forms are used to collect input and/or information from a user.

Typical HTML form tags:

form, label, fieldset, input, select

```
<form>
```

```
  <input type="text">
```

```
</form>
```

LABELS

A label describes what should go inside the input on a form, like “Name” or “Address”

```
<label>Name:</label>
```

```
<input type="text" name="name">
```

INPUT

- We use form inputs to allow input from users in various ways.
- Lots to choose from:
 - `text`
 - `textarea`
 - `checkbox`
 - `radio`
 - `submit`

INPUT

- Inputs have **THREE** required attributes.
- **type**: the type of input to display
- **name**: associated with groups of inputs, gives identifiers when submitting/processing forms.
 - radio button groups get same name.
- **value**: what is the value of input?

```
<label><input type="radio" name="answer"
```

```
value="yes">Yes</label>
```

```
<label><input type="radio" name="answer"
```

```
value="no">No</label>
```

DROPDOWN

- **select:** name of the dropdown
- **option:** value that gets processed by form.

```
<select name="mydropdown">  
  <option value="cats">Cats</option>  
  <option value="dogs">Dogs</option>  
  <option value="other">Other</option>  
</select>
```

FIELDSET AND LEGENDS

- **Fieldset** groups forms into sections.
- **Legend** gives your fields an overall label.
- *Not required, and in my opinion not used very much anymore.

```
<fieldset id="address">  
    <legend>Please fill out the following:</legend>  
</fieldset>
```

SUBMIT

- **submit** button to send the form
- Often, long forms will have a **reset** button to clear inputs and start over. (not required)
- These should go **last** in the form.

```
<input type="submit" name="submit" value="Submit">
```

```
<input type="reset" name="reset" value="Clear form">
```

DEMO TIME

SUPPLEMENTAL TECHNOLOGIES

- JAVASCRIPT
- Javascript/DOM - Javascript is a scripting language that developers use to manipulate the Document Object Model in conjunction with HTML and CSS.

JQUERY



- Makes creating complex JS functions very easy.
- Great library for easily creating animations and interactions with very little code.
- <http://jqueryfordesignersbook.com/>

CONTENT MANAGEMENT SYSTEMS

- Wordpress: wordpress.com or WP Engine
- Squarespace: Templated portfolio-like sites.
- github.io: Hosted, version-controlled pages. (Class site is hosted on github.io)

WEB HOSTING

- First thing you have to do is choose a **hosting provider**.
- I personally use bluehost.com for all my hosting.
- Other great hosts:
 - Media Temple (mediatemple.com)
 - Dreamhost (dreamhost.com)
- **Note:** Buy your domain and web hosting from the same company. Makes it so much easier.

REGISTER DOMAIN

- Most hosting providers also allow you to purchase domains from them (though they might not have access to very “unique” domain signatures (.io, .rocks, .photography))
- If possible though, purchase your domain through your hosting provider, as it makes managing your domain and hosting simpler and seamless.


CPANEL

- Most hosting providers have a dashboard called the “cpanel” for managing all aspects of your account:
 - email, hosting, domain managements, databases, etc.
- Most hosting providers also now have 1-click installs for many applications, including Wordpress.

FTP (FILE TRANSFER PROTOCOL)

- To upload files to your site, you use what's called **FTP**.
- Stands for **FILE TRANSFER PROTOCOL**
- To get started, you create an **FTP account**, in your cpanel or hosting dashboard.

FTP (FILE TRANSFER PROTOCOL)

seanthompsonphoto.com (shared)

[hosting](#) [WordPress tools](#) [domains](#) [addons](#) [account](#) [cart](#) [help](#) [logout](#)

[home](#) [cpanel](#) [server](#) [email](#) [website](#) [ftp](#) [databases](#) [manage ips](#) [install scripts](#)

 @ [Feeling stuck? Have us pick one!](#)

A lowercase letter (e.g. abcde)

An uppercase letter (e.g. ABCDE)

A number (e.g. 123456)

A symbol or a space (e.g. ~!@#\$%^)

FTP (FILE TRANSFER PROTOCOL)

- To upload files to your site, you need to use an **FTP client**.
- Many FTP applications are out there, good ones are:
 - Fetch (for Mac)
 - Transmit (Mac)
 - FileZilla (All platforms)
- Cyberduck (PC and Mac)
- Your site will be in a folder called **public_html** (that is your root directory of your site)

THATS ALL FOLKS!

Please provide feedback of the class!

<http://svcseattle.com/evaluation>