

# CSC435: Web Programming

## Lecture 6: CSS Positioning and Floating

Bei Xiao

American University

Feb 6, 2018

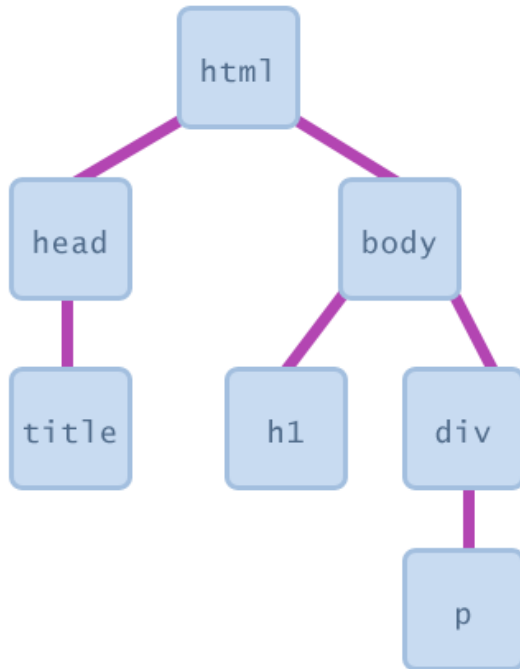
# Activity Outline

- Document object model (DOM) and cascading
- Float, Positioning, Display
- Page Layout Exercise

# CSS Selector Summary

Example	Example described
<b>p</b>	All <b>&lt;p&gt;</b> elements
<b>.abc</b>	All elements with the <b>abc class</b> , i.e. <b>class="abc"</b>
<b>#abc</b>	Element with the <b>abc id</b> , i.e. <b>id="abc"</b>
<b>p.abc</b>	<b>&lt;p&gt;</b> elements with <b>abc class</b>
<b>p#abc</b>	<b>&lt;p&gt;</b> element with <b>abc id</b> ( <b>p</b> is redundant)
<b>div strong</b>	<b>&lt;strong&gt;</b> elements that are descendants of a <b>&lt;div&gt;</b>
<b>h2, div</b>	<b>&lt;h2&gt;</b> elements and <b>&lt;div&gt;</b> s

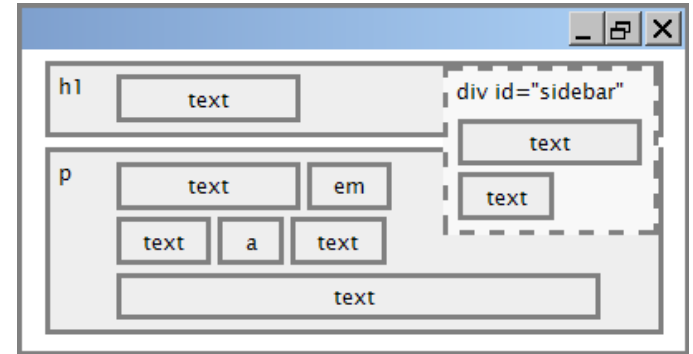
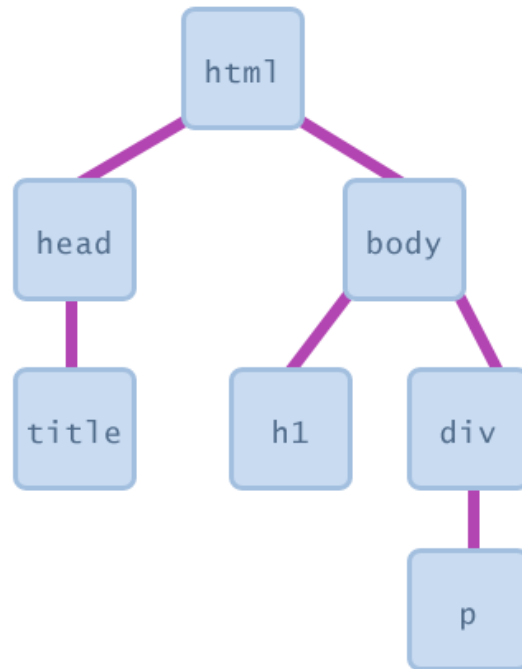
# Document object model (DOM) tree



```
<html>
  <head>
    <title> ... </title>
  </head>
  <body>
    <h1> ... </h1>
    <div>
      <p> ... </p>
    </div>
  </body>
</html>
```

- The Document Object Model (DOM) is a cross-platform and language-independent convention for representing and interacting with objects in HTML, XHTML, and XML documents.
- JavaScript can add, remove any element and attribute in HTML, CSS, creating events, etc.

# Document object model (DOM) tree



- The Document Object Model (DOM) is a cross-platform and language-independent convention for representing and interacting with objects in HTML, XHTML, and XML documents.
- JavaScript can add, remove any element and attribute in HTML, CSS, creating events, etc.

# CSS inheritance

- The CSS properties set on an element via the cascade can be inherited by that element's child elements.
- But only certain element can be inherited (e.g. margins, width, border will not be inherited).

```
div {  
    font-size: 20px  
}
```

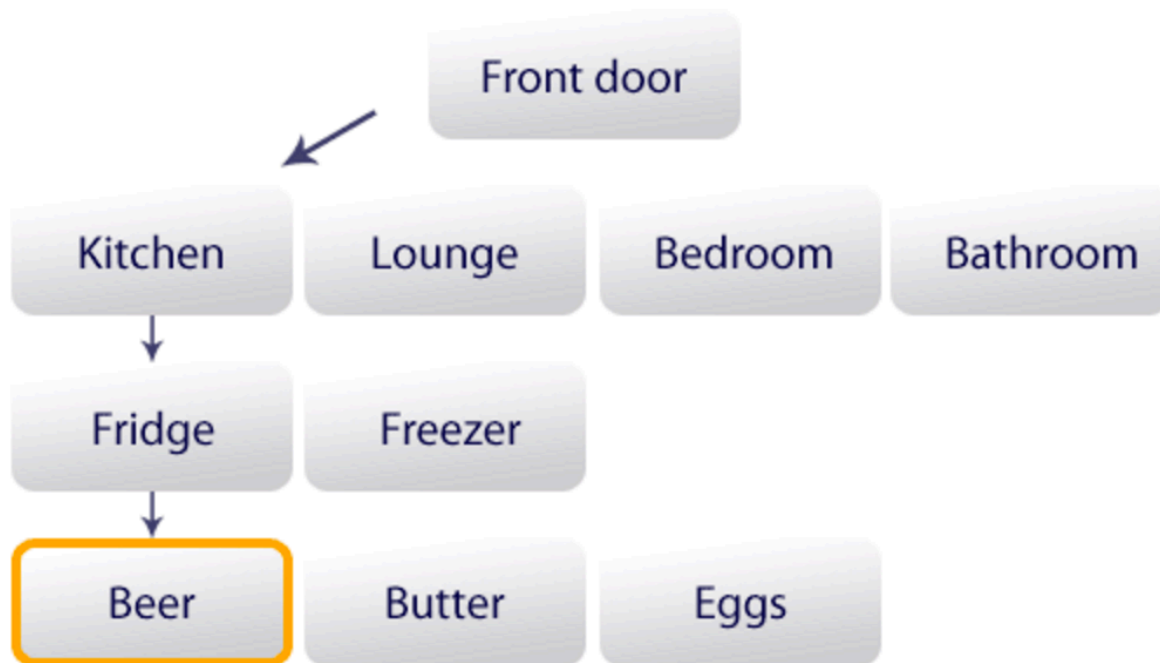
CSS

```
<div>  
    <p> This I <em> sentence</em> will  
have a 20px  
    <a href="#"> font-size </a>  
    </p>  
</div>
```

HTML

Check here for CSS property table: <https://www.w3.org/TR/CSS21/propidx.html>

# Demo: living room



# Demo: living room

Add a style sheets that change the appearance of the page:

- Each div has its own border and box properties.
- All texts are centered.
- All divs are centered on the page. Has a width of 300px and height 100px. Add border accordingly. Add padding of 10px. Border radius is 10px.
- Pizza is in a red container.
- All texts have “**Georgia**” font.
- “**My can of beer**” is **bolded**.
- “Soap” is *italic*.



# Normal flow of the page

- Statically-positioned elements are positioned according to the “normal flow of the page”.
- Block element: always take up the full width available, with **newlines** before and after.  
div, h1,p, ul, table, etc.
- In-line element: only takes as much width as needed and do not force new lines  
span, a, img, etc

# The CSS Float

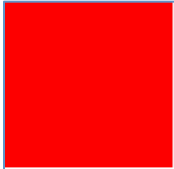
Informally:

- A float element is removed from normal document flow.
- Two properties: “float:left” and “float:right” pushing it to the left or right of the containing element .
- Block elements will ignore it.
- Inline elements will wrap around it.
- One use for float is for text to wrap around an image since text is inline.

# Floating Example

The red block is a non-floating (regular) block element (a div). It's only 100px wide, but since it is a block element, the browser positions it on its own line (spanning the width of the page).

```
<div> (white background)
<div> (red block)
<p> (text)
```




A buncha really long text yaad yo yodyo aoydoyo dyo adya dyo yaad yo yodyo  
aoydoyo dyo adya dyo yaad yo yodyo aoydoyo dyo adya dyo yaad yo yodyo aoydoyo  
dyo adya dyo yaad yo yodyo aoydoyo dyo adya dyo yaad yo yodyo aoydoyo dyo adya  
dyo

# Floating Example

Now the red block has the float CSS property set to left. This tells the browser to give the elements as much spaces as it needs, and then start brining the next content up from below and fill the cracks.

```
<div> (white background)  
<div> (red block, floating left)  
<p> (text)
```

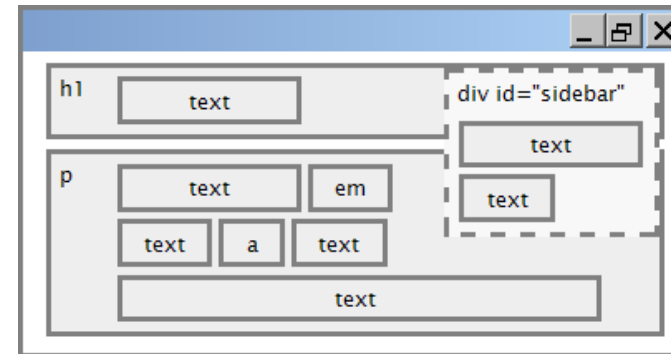


A buncha really long text yaad yo yodyo aoydoyo dyo adya dyo yaad yo yodyo  
aoydoyo dyo adya dyo yaad yo yodyo aoydoyo dyo adya dyo yaad yo yodyo  
aoydoyo dyo adya dyo yaad yo yodyo aoydoyo dyo adya dyo yaad yo yodyo  
aoydoyo dyo adya dyo

# The CSS float property

property	description
<u>float</u>	side to hover on; can be left, right, or none (default)

- a ***floating*** element is removed from normal document flow
- Underlying text wraps around it as necessary



# Float example

- ``
- Lorem ipsum dolor sit amet, consectetur adipiscing elit....

```
img.headericon {  
    float: left;  
}
```



Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aliquam sc  
rus ut dui mollis, sed malesuada leo pretium. Morbi bibendum mi  
rum convallis. Duis id eros dolor. In id eros blandit lectus viverra  
ommodo velit. Cras pretium nunc id nisl elementum, at interdum

odio blandit. Donec luctus rutrum iaculis. Praesent luctus ante et cursus suscipit.

# Float demo

```
img
{
float:left;
margin:5px 10px 10px 5px;
height:300px;
width:370px;
}
```

## My Favourite Pumpkin Pie



This is the traditional holiday pumpkin pie. This classic recipe has been on Pumpkin labels since 1950. This pie is easy to prepare and even easier to enjoy. Just mix, pour, bake for a delicious homemade tradition. Pumpkin pie is my favourite dessert.

- 3/4 cup granulated sugar
- 1 teaspoon ground cinnamon
- 1/2 teaspoon salt
- 1/2 teaspoon ground ginger
- 1/4 teaspoon ground cloves
- 2 large eggs
- 1 can pureed 100% pumpkin
- NESTLE Evaporated Milk
- 1 unbaked 9-inch (4-cup volume) deep-dish pie shell

### Directions

Directions MIX sugar, cinnamon, salt, ginger and cloves in small bowl. Beat eggs in large bowl. Stir in pumpkin and sugar-spice mixture. Gradually stir in evaporated milk. POUR into pie shell. BAKE in preheated 425Â° F oven for 15 minutes. Reduce temperature to 350Â° F; bake for 40 to 50 minutes or until knife inserted near center comes out clean. Cool on wire rack for 2 hours. Serve immediately or refrigerate. Top with whipped cream before serving.

# Float demo







# Float content with width

- I am not floating, no width set

I am floating right, no width set

I am floating right, no width set, but my text is very long so this paragraph doesn't really seem like it's floating at all, darn

I am not floating, 45% width

I am floating right, 45% width

- often floating elements should have a `width` property value
  - if no `width` is specified, other content may be unable to wrap around the floating element

# Float

- It is the common way to get block elements to sit side-by-side.
- Specifics:
  - A left-floated box will float to the left until it touches either the edge of the parent element, or the edge of another floated box
  - If the size of the floated box exceeds the available horizontal space, the floated box will be shifted down

# Clear property

- `p { background-color: fuchsia; }`
- `h2 { clear: right; background-color: cyan; }`

XKCD a webcomic of romance, sarcasm, math, and language...



## My XKCD Fan Site

property	description
<u><a href="#">clear</a></u>	disallows floating elements from overlapping this element; can be left, right, both, or none (default)

# Common error: container too short

```
<p>  
  XKCD a webcomic of romance, sarcasm,  
  math, and language...</p>
```

HTML

```
p { border: 2px dashed black; }  
img { float: right; }
```

CSS

XKCD a webcomic of romance, sarcasm, math, and  
language...



- We want the `p` containing the image to extend downward so that its border encloses the entire image

# Positioning Elements

## position: fixed

Puts an element at a position within the browser window.  
Always stays the same place even if the page is scrolled.

## position: absolute

Puts an element at an absolute position with respect to  
whole webpage or a parent container.

# Effects on Parent Containers

position: relative

Makes children positioned relative to the container.

position: absolute

Makes children positioned relative to the container.

# The position property

```
div#ad {  
  position: fixed;  
  right: 10%;  
  top: 45%;  
}
```

Here I am!

property	value	description
position	static	default position
	relative	offset from its normal static position
	absolute	a fixed position within its containing element
	fixed	a fixed position within the browser window
<u>top</u> , <u>bottom</u> , <u>left</u> , <u>right</u>	positions of box's corners	



# Positioning: static

- Default position for all element. You don't have to specify this unless you want to override a position previously set.

```
#div-1 {  
position:static;}
```

id = div-before

id = div-1

id = div-1a

Lorem ipsum dolor sit amet, consectetur adipiscing elit.  
Integer pretium dui sit amet felis. Integer sit amet diam.  
Phasellus ultrices viverra velit.

id = div-1b

Lorem ipsum dolor sit amet, consectetur adipiscing elit.  
Integer pretium dui sit amet felis. Integer sit amet diam.  
Phasellus ultrices viverra velit. Nam mattis, arcu ut  
bibendum commodo, magna nisi tincidunt tortor, quis  
accumsan augue ipsum id lorem.

id = div-1c

# Positioning: relative

- Relative position means relative to the element where it would normally occur. You can specify top, bottom, left or right.

```
#div-1 {  
  position: relative;  
  top: 20px;  
  right: -40px; }
```

id = div-before

id = div-1

id = div-1a

Lorem ipsum dolor sit amet, consectetur adipiscing elit.  
Integer pretium dui sit amet felis. Integer sit amet diam.  
Phasellus ultrices viverra velit.

id = div-1b

Lorem ipsum dolor sit amet, consectetur adipiscing elit.  
Integer pretium dui sit amet felis. Integer sit amet diam.  
Phasellus ultrices viverra velit. Nam mattis, arcu ut  
bibendum commodo, magna nisi tincidunt tortor, quis  
accumsan augue ipsum id lorem.

id = div-1c

id = div-after

# Positioning: absolute

- the element is removed from
- the document and placed exactly

where you tell it to go

```
#div-1a {  
  position: absolute;  
  top: 5px;  
  left: 5px;  
  width: 200px;  
}
```

id = div-1a

Lorem ipsum dolor sit amet,  
consectetuer adipiscing elit.  
Integer pretium dui sit amet  
felis. Integer sit amet diam.  
Phasellus ultrices viverra velit.

consectetuer adipiscing elit.  
Integer pretium dui sit amet felis. Integer sit amet diam.  
Phasellus ultrices viverra velit. Nam mattis, arcu ut  
bibendum commodo, magna nisi tincidunt tortor, quis  
accumsan augue ipsum id lorem.

id = div-1c

id = div-after

# Positioning: relative + position: absolute

If we set relative positioning on div-1,  
Any elements within div-1 will be positioned  
relative to div-1. Here  
of div-1:

```
#div-1{  
  position:relative;}  
  
#div_1a{  
  position:absolute;  
  top:0  
  right:0  
  width:200px;  
}
```

id = div-before

id = div-1

id = div-1a

id = div-1b

Lorem ipsum dolor sit amet, Integer pretium dui sit amet  
Integer pretium dui sit amet felis. Integer sit amet diam.  
Phasellus ultrices viverra velit. Phasellus ultrices viverra velit.  
bibendum commodo, magna i  
accumsan augue ipsum id lorem.

id = div-1c

id = div-after

# Positioning: two columns layout

How do we make the page has two columns using position relative and absolute?

Please modify div-1, div-1a and div-1b to make

id = div-before

id = div-1b

Lorem ipsum dolor sit amet,  
consectetuer adipiscing elit.  
Integer pretium dui sit amet  
felis. Integer sit amet diam.  
Phasellus ultrices viverra velit.  
Nam mattis, arcu ut bibendum  
commodo, magna nisi  
tincidunt tortor, quis accumsan  
augue ipsum id lorem.

id = div-1a

Lorem ipsum dolor sit amet,  
consectetuer adipiscing elit.  
Integer pretium dui sit amet  
felis. Integer sit amet diam.  
Phasellus ultrices viverra velit.





# This can also be easily achieved by float but notice text will wrap around

id = div-before

id = div-1

id = div-1a

Lorem ipsum dolor sit amet,  
consectetuer adipiscing elit.  
Integer pretium dui sit amet  
felis. Integer sit amet diam.  
Phasellus ultrices viverra velit.

id = div-1b

Lorem ipsum dolor sit  
amet, consectetuer  
adipiscing elit. Integer  
pretium dui sit amet felis.  
Integer sit amet diam.  
Phasellus ultrices viverra  
velit. Nam mattis, arcu ut

bibendum commodo, magna nisi tincidunt tortor, quis  
accumsan augue ipsum id lorem.

id = div-1c

```
#div-1{  
position:relative;  
#div_1a{  
float:left  
}  
#div_1b{  
}
```

id = div-after



# How to achieve this? Use clear

id = div-before

id = div-1

id = div-1a

Lorem ipsum dolor sit  
amet, consectetur  
adipiscing elit. Integer  
pretium dui sit amet  
felis. Integer sit amet  
diam. Phasellus  
ultrices viverra velit.

id = div-1b

Lorem ipsum dolor sit  
amet, consectetur  
adipiscing elit. Integer  
pretium dui sit amet  
felis. Integer sit amet  
diam. Phasellus  
ultrices viverra velit.  
Nam mattis, arcu ut  
bibendum commodo,  
magna nisi tincidunt  
tortor, quis accumsan  
augue ipsum id lorem.

id = div-1c

id = div-after

# How to achieve this? Use clear

id = div-before

id = div-1

id = div-1a

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer pretium dui sit amet felis. Integer sit amet diam. Phasellus ultrices viverra velit.

id = div-1b

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer pretium dui sit amet felis. Integer sit amet diam. Phasellus ultrices viverra velit. Nam mattis, arcu ut bibendum commodo, magna nisi tincidunt tortor, quis accumsan augue ipsum id lorem.

id = div-1c

id = div-after

```
#div-1{
position:relative;}
#div_1a{
float:left;
width:190px
}
#div_1b{
float:left
width:190px
}
#div_1c{
clear:both:
}
```

# Alignment vs. float vs. position

1. if possible, lay out an element by *aligning* its content
  - horizontal alignment: `text-align`
    - set this on a block element; it aligns the content within it (not the block element itself)
  - vertical alignment: `vertical-align`
    - set this on an inline element, and it aligns it vertically within its containing element
2. if alignment won't work, try *floating* the element
3. if floating won't work, try *positioning* the element
  - **absolute/fixed positioning are a last resort and should not be overused**

# display property

- Affects how an element is displayed.
- **none**- removes element from page. Other elements as positioned as if element doesn't exist.
- **block-element** is treated as a block element.
- **inline-element** is treated as inline element. You can't specify width and height.
- **Inline-block**-element only takes up available width and doesn't force newlines, but you can specify width, height, and margins, etc.
  - Another way to get block elements to go side by side.

# Display block elements as inline

```
• <ul id="topmenu">
•   <li>Item 1</li>
•   <li>Item 2</li>
•   <li>Item 3</li>
• </ul>
```

```
#topmenu li {
  display: inline;
  border: 2px solid gray;
  margin-right: 1em;
}
```

Item 1

Item 2

Item 3

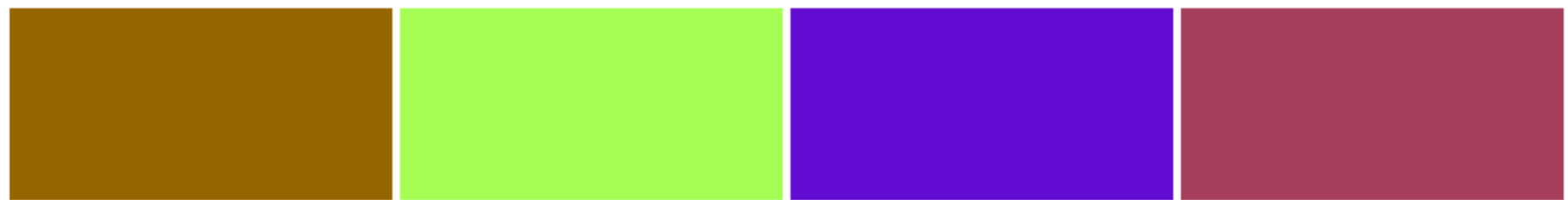
- lists and other block elements can be displayed inline
  - flow left-to-right on same line
  - width is determined by content (block elements are 100% of page width)

# Flexbox

- A layout library built into CSS-more on this next week.
- Useful for nestling block elements next to each other in rows or columns and specifying how much space each of the elements take up.
- When you set a parent to `display:flex`, all items inside becomes “flex-items”.
- <https://css-tricks.com/snippets/css/a-guide-to-flexbox/>

# Exercise 1:

Using Display Property, create the following page:



# Exercise 2: Page layout

## Document Heading

- [News](#)
- [Blog](#)
- [Weather](#)

...

## Column 1

There is what happened today. On the neighborhood of Adams Morgan, a man with....

## Side Bar

Lorem ipsum dolor sit amet, consectetur adipiscing elit...

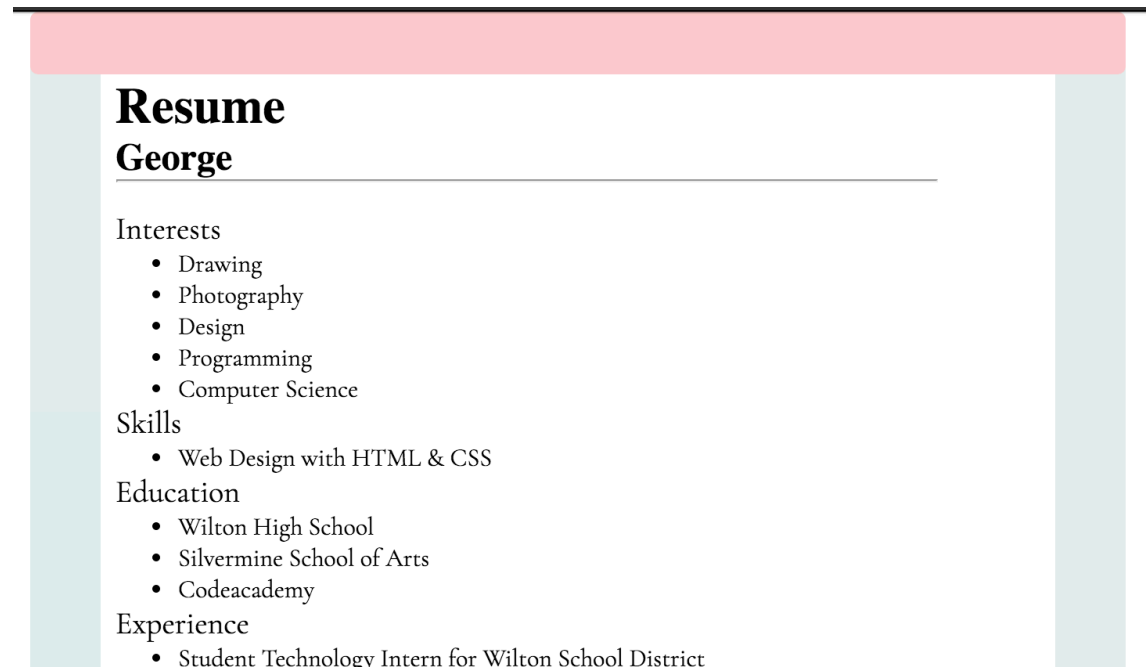
- [Link 1](#)
- [Link 2](#)

...



# Now, let's go back to Resume

- Download SimpeResume.html or you your own resume.html
- Add style sheets.



# Hyperlinks

- Link states
- a: link- a normal unvisited link
- a: visited- a link the user has visited
- a: hover\_ a link when the user mousess over it.
- A: active- a link the moment it is clicked
- Example a: link{...}
- A#my\_id: hover{...}

# Next Week:

- Wrap up HTML/CSS, buttons, links, etc.
- Finish take-home exercise: Journal.html.
- Homework 2 is due on Monday, end of the day.
- Onto JavaScript.

# Exercises & Take-home reading

- Take-home tutorials (this will be so helpful for homework)

CSS positioning:

[http://www.w3schools.com/css/css\\_positioning.asp](http://www.w3schools.com/css/css_positioning.asp)

CSS box model:

This is a great read:

<https://css-tricks.com/the-css-box-model/>

Simple tutorials to review:

[http://www.w3schools.com/css/css\\_boxmodel.asp](http://www.w3schools.com/css/css_boxmodel.asp)

[https://developer.mozilla.org/en-US/docs/Web/CSS/CSS\\_Box\\_Model/Introduction\\_to\\_the\\_CSS\\_box\\_model](https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Box_Model/Introduction_to_the_CSS_box_model)