CSC435: Web Programming
Lecture 4: CSS

Bei Xiao

American University

Jan 30, 2018

Activity Outline

- Review CSS text and list properties
- Cascading Style
- Id and Class
- Grouping content
- Exercise: Resume

- Creative Project 1 is due on Friday.
- Homework 1 is out today (Jan 30th) due next Tuesday).

Exercises & Take-home reading (must read before next class)

- Finish resumu.html, resume.css

- Take-home reading:
  How CSS works:

- https://developer.mozilla.org/en-US/docs/Learn/CSS

- Cascading and inheritance
https://developer.mozilla.org/en-US/docs/Web/Guide/CSS/Getting_Started/Cascading_and_inheritance
  CSS Selectors

https://developer.mozilla.org/en-US/docs/Web/Guide/CSS/Getting_Started/Selectors

Example: Inline-block

```
<img src="http://static.tvtropes.org/
pmwiki/pub/images/
Hello_Kitty_Pink_2981.jpg" />
<img src="http://static.tvtropes.org/
pmwiki/pub/images/
Hello_Kitty_Pink_2981.jpg" />
<img src="http://static.tvtropes.org/
pmwiki/pub/images/
Hello_Kitty_Pink_2981.jpg" />
<img src="http://static.tvtropes.org/
pmwiki/pub/images/
Hello_Kitty_Pink_2981.jpg" />
```

```
img {
 width: 50px;
}
```

What does this look like in the browser?

"http://static.tvtropes.org/pmwiki/pub/images/Hello_Kitty_Pink_2981.jpg”

=

Inline-block

has width and height; flows left to right

Can set **`width`** on inline-block element, so image width is set to 50px. (Codepen)

`inline-block` flows left to right, so images are right next to each other.

https://codepen.io/pen

The display CSS property

You can change an element's default rendering type by changing the **`display`** property. Examples:

```
p {
 display: inline;
}
```

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

Possible values for display:

- Block
- Inline
- Inline-block

Review

1. **block:** flows **top-to-bottom**; **has** **`height`** and **`width`**

   `<p>, <h1>, <blockquote>, <ol>, <ul>, <table>`

2. **inline:** flows **left-to-right**; **does not have** **`height`** and **`width`**

   `<a>, <em>, <strong>,<br>`

   a. **inline block:** flows **left-to-right**; **has** **`height`** and **`width`** equal to size of the content

      `<img>`

Questions?

CSS properties for text: Example

```
<style>
h1 {
   text-align: center;
}
h2 {
   text-align: left;
}
p {
   text-align: justify;
}
</style>
```

```
<body>
<h1>The weather channel</
h1>

<h2>DC Weather and road
condition</h2>

<p>The massive storm left 18
people dead and caused
heavy flooding</p>
</body>
```

Today's news

Near-Record Totals in Parts of East Coast

The massive storm left 18 people dead and caused heavy flooding along the coast. The 26.8 inches of snow that fell in Central Park was the second-highest total ever recorded.

CSS properties for text: Example

```
blockquote { text-align: justify; }
h2 { text-align: center; }
```

The Emperor's Quote

[TO LUKE SKYWALKER] The alliance... will die. As will your friends. Good, I can feel your anger. I am unarmed. Take your weapon. Strike me down with all of your hatred and your journey towards the dark side will be complete.

- can be left, right, center, or justify (which widens all full lines of the element so that they occupy its entire width)

text-align mystery

Summary

The **`text-align`** CSS property describes how inline content like text is aligned in its parent block element. `text-align` does not control the alignment of block elements, only their inline content.

Initial value	`start`, or a nameless value that acts as `left` if `direction` is `ltr`, `right` if `direction` is `rtl` if `start` is not supported by the browser.
Applies to	block containers

Text-decoration

```
p {
    text-decoration: underline;
}
```

This paragraph uses the style above.

- can also be `overline`, ~~`line-through`~~, `blink`, `or none`
- effects can be combined:

```
text-decoration: overline underline;
```

Text-shadow

```
p {
  font-weight: bold;
  text-shadow: 2px 2px gray;
}
```

This paragraph uses the style above.

- shadow is specified as an X-offset, a Y-offset, and an optional color

List type property

```
ol { list-style-type: lower-roman; }
```

- Possible values:

  i. `none` : No marker

  ii. `disc` (default), `circle, square`

  iii. `Decimal`: 1, 2, 3, etc.

  iv. `decimal-leading-zero`: 01, 02, 03, etc.

  v. `lower-roman`: i, ii, iii, iv, v, etc.

  vi. `upper-roman`: I, II, III, IV, V, etc.

  vii. `lower-alpha`: a, b, c, d, e, etc.

  viii. `upper-alpha`: A, B, C, D, E, etc.

  x. `lower-greek`: alpha, beta, gamma, etc.

  others: hebrew, armenian, georgian, cjk-ideographic, hiragana…

List type property

```
ul {
List-style: square url(“squarepurple.gif”)
}
```

```
<ul>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Coca Cola</li>
</ul>
```

- Coffee
- Tea
- Coca Cola

The bad way to produce styles

```
<p>
      <font face="Arial">Welcome to Greasy
Joe's.</font>
      You will <b>never</b>, <i>ever</i>,
<u>EVER</u> beat
      <font size="+4" color="red">OUR</font>
prices!                                    CSS
</p>
```

Welcome to Greasy Joe's. You will **never**, *ever*, EVER beat OUR prices!

Cascading Style Sheets (CSS): <link>

```
<head>
      ...
      <link href="filename" type="text/css"
rel="stylesheet" />
      ...
</head>
```

CSS

- CSS describes the appearance and layout of information on a web page (as opposed to HTML, which describes the content of the page)
- can be embedded in HTML or placed into separate .css file (preferred)

Three ways to use style

- **Inline**: add a “style” attribute containing the CSS rule directly to an HTML

- **Internal:** Define CSS Rules in the style tag in the HTML

```
<head>
    <style type="text/css">
    p{
        color:red;
    }
    </style>
</head>
```

- **External**: Add a link to an external stylesheet in the head of your HTML page

```
<head>
    <link rel="stylesheet" type="text/css" href="exercise1.css">
</head>
```

Three ways to use style

- **Inline**: add a “style” attribute containing the CSS rule directly to an HTML

- **Internal:** Define CSS Rules in the style tag in the HTML

```
<head>
    <style type="text/css">
    p{
        color:red;
    }
    </style>
</head>
```

Bad

- **External**: Add a link to an external stylesheet in the head of your HTML page

```
<head>
    <link rel="stylesheet" type="text/css" href="exercise1.css">
</head>
```

Convent vs. Presentation

- HTML is for **content**; *what* is on the page (heading; list; code; etc.)
- CSS is for **presentation**; how to display the page (bold; centered; 20px margin; etc.)
- keeping content separate from presentation is a very important web design principle
- If the HTML contains no styles, its entire appearance can be changed by swapping .css files
- see also: CSS Zen Garden

Cascading Style Sheets

- HTML is for **content**; *what* is on the page (heading; list; code; etc.)
- CSS is for **presentation**; how to display the page (bold; centered; 20px margin; etc.)
- keeping content separate from presentation is a very important web design principle
- If the HTML contains no styles, its entire appearance can be changed by swapping .css files
- see also: CSS Zen Garden

Cascading Style Sheets

- It's called Cascading Style Sheets because the properties of an element *cascade* together in this order:
  1. Browser's default styles (reference)
  2. External style sheet files (in a <link> tag)
  3. Internal style sheets (in a <style> tag in the page header)
  4. Inline style (the style attribute of an HTML element)

Inheriting styles (explanation)

```
body{
    background-color:yellow;
    font-family: georgia ;
}

p{background-color:aqua;color:red;}
a{text-decoration:overline udnerline;}
h2{font-weight:bold;
text-align:center;}
```

CSS

This is a heading.

A styled paragraph. Google for questions

- Lecture1

Output

- When multiple styles apply to an element, they are inherited.
- A more tightly-matching rule can override a more general inherited rule.
- Not all properties are inherited (notice link's color above.

CSS pseudo-classes

- A pseudo-class is used to define a special state of an element.
- It can be used:
  - Style an element when a user mouse over it.
  - Style visited and unvisited links differently
  - Style en element when it gets focus

CSS pseudo-classes

```css
a:link{color:#FF0000;}

a:visited{color:#00FF00;}

a:hover{ color:#FF00FF;}
```

CSS

A styled paragraph. Buy early, buy often!

HTML

Pseudo-class can be combined with CSS class.

Example: div:hover{ background-color:blue;}

CSS pseudo-classes

Class	Description
:active	An activated or selected element
:focus	An element that has the keyboard focus
:hover	An element that has the mouse over
:link	A link that has not been visited
:visited	A link that has been already visited
:first-letter	the first letter of text inside an element
:first-line	The first line of text inside an element

More on pseudo-classes

id and class

There are 3 basic types of CSS selectors:

Element selector (this is the one we've been using)	**p**	All **<p>** elements
✧ **ID selector** ✧	**#abc**	element with **id="abc"**
✧ **Class selector** ✧	**.abc**	elements with **class="abc"**

```
<h1 id="title">Homework</h1>
<em class="hw">HW0</em> is due Friday.<br/>
<em class="hw">HW1</em> goes out Monday.<br/>
<em>All homework due at 11:59pm.</em>
```

id and class

Id's are unique

- **Unique identifier for an element**
- **Each element can only have one id**
- Only allowed one id per element per page

Classes are NOT unique

- **Non-unique grouping attribute to share with many elements**
- Many elements (even of different types) can share the same class.
- Each element can have many different classes.

More on difference between id and Class.

id Example

```
<div class="intro">
  <p id="firstname">My name is Donald.</
p>
  <p id="hometown">I live in the
whitehouse.</p>
</div>
```

HTML

```
#firstname {
    background-color: yellow;
}
```

CSS

My name is Donald.

I live in the white house.

Why are these useful?

```
#my-id{
/*    properties*/
}
.my-class{
/*    other properties*/

}
```

CSS

- Gives you another way to talk about your content in CSS (and later in JavaScript) .

- A mnemonic:. Java programs compile into .class files so ….

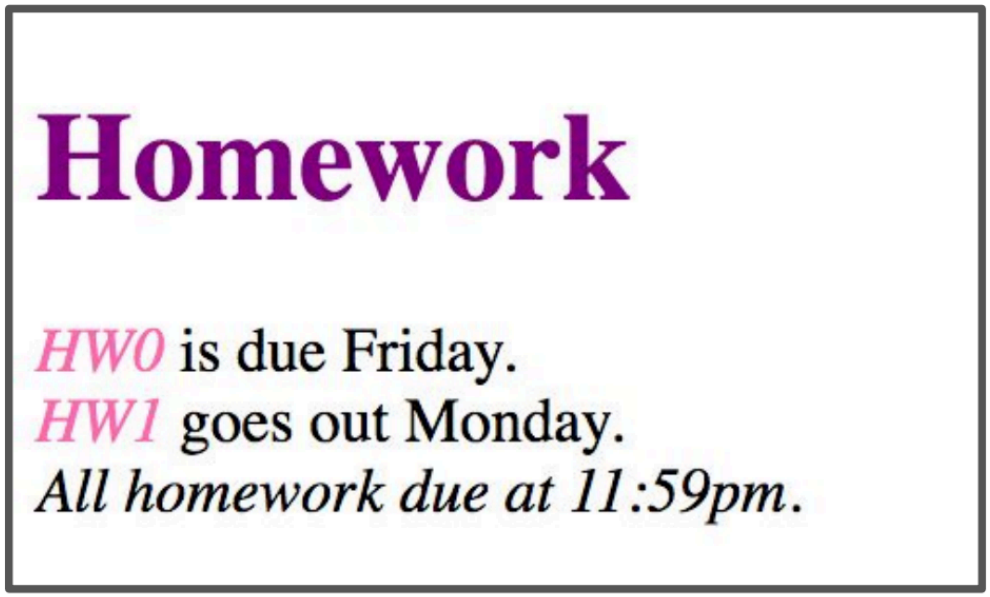
try to remember dot(.) class and hash(#) id.

Why are these useful?

```
<h1 id="title">Homework</h1>
<em class="hw">HW0</em> is due Friday.<br/>
<em class="hw">HW1</em> goes out Monday.<br/>
<em>All homework due at 11:59pm.</em>
```

```
.hw {
  color: hotpink;
}

#title {
  color: purple;
}
```



`id` or `class`?

- How do you decide whether to use an `id` or `class`?
- Probably prefer class. You can use an id per page, so it is good to be a little stingy with them. `Class`es are free.
- On the other hand, if you know you are making a unique section or page element (form, submit button), id is the way to go.

A caveat:

- It’s easy to just make classes for everything, but don’t forget the that HTML is made to describe your content. So, prefer a <p> tag over a `class` named *paragraph.*

Demo: id selector (#nameofid)

```
<style>
#para1 {
   text-align: center;
   color: red;
}
</style>
```

CSS

```
<body>

<p id="para1">Hello World!</p>
<p>This paragraph is not affected by the
style.</p>

</body>
```

Hello World!

This paragraph is not affected by the style.

Demo: class selector (.nameofclass)

```
p.blue_paragraphs{
        color:blue;
}
```

```
 <p>A styled paragraph. <a
href="http:www.google.com">Buy early, buy often!</a></
p>

<p class="blue_paragraphs">This is a blue paragraph. <
```

A styled paragraph. Buy early, buy often!

This is a blue paragraph.

Demo: more than one class

```
p.center {
    text-align: center;
    color: red;
}
p.large {
    font-size: 300%;
}
```


```
<h1 class="center">This heading will not be
affected</h1>
<p class="center">This paragraph will be red and
center-aligned.</p>
<p class="center large">This paragraph will be red,
center-aligned, and in a large font-size.</p>
```

HTML

CSS Selector Combinators

- A combinator is something that explains the relationship between the selectors.

- A CSS selector can contain more than one simple selector. There are mainly two different combinators in CSS:

  - descendant selector (space)
  - Child selector (>)

CSS descendant selector Example (without >)

```
<p>Shop at <strong>Hardwick's Hardware</
strong>...</p>
<ul>
  <li>The <strong>best</strong> prices in town!
</li>
  <li><em><strong>Act</strong></em> while
supplies last!</li>
</ul>
```

HTML

```
li strong { text-decoration: underline; }
```

CSS

Produces:

Shop at **Hardwick's Hardware**...

- The **best** prices in town!
- ***Act*** while supplies last!

CSS child selector Example (with >)

The child selector selects all elements that are the immediate children of a specified element.

```
<p>Shop at <strong>Hardwick's Hardware</
strong>...</p>
<ul>
  <li>The <strong>best</strong> prices in town!
</li>
  <li><em><strong>Act</strong></em> while
supplies last!</li>
</ul>
```

HTML

```
Li >strong { text-decoration: underline; }
```

CSS

Produces:

Shop at **Hardwick's Hardware**...

- The **best** prices in town!
- ***Act*** while supplies last!

Exercise: Resume

- Download the resume.html from blackboard.

- Please fix the typos and inconsistency in the .html

- Can you create a style.css so that this page looks better?

- For the body text: use one of the Google font:
- https://www.google.com/fonts/specimen/Open+Sans
- For the headers, choose another font.
- Make a nice background color for the page
- Make the header have different font from the paragraph
- Experiment with font size and font spacing.
- Experiment with Class and Ids.

Next Class:

- CSS Box model

Read here:
https://www.w3schools.com/css/css_boxmodel.asp

- Page layout
- Quiz!!
- Homework 2 out today, due in a week (Feb 6).