

# HTML and CSS

CS142 Section 1

# Overview: HTML vs. CSS

HTML = scaffolding / structure

- Applying "meaning/structure" to content, rather than style.
  - e.g. "This is a paragraph, this is a heading."
  - **NOT**: "This is in size-16 Times New Roman."
- (This is why `<i>` and `<b>` tags are now considered bad practice; *semantic* `<em>` and `<strong>` preferred.)



# Overview: HTML vs. CSS

CSS = painting / styling / (some) interactivity



- Styling for the semantic content expressed in HTML.
- CSS Rule = Selector + Declaration(s).
- "Don't Repeat Yourself:"  
Selector lets you apply the same style to multiple elements, change them all at once by modifying declaration.

# HTML: Head and body

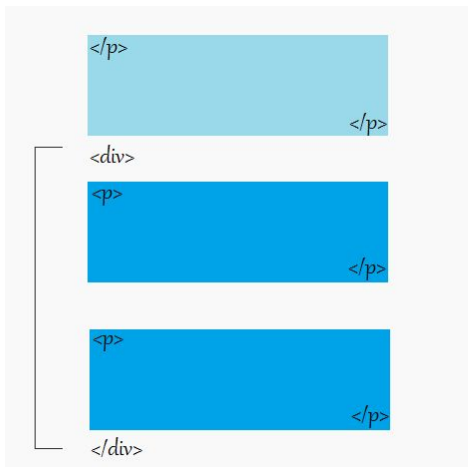
- **Head:** Contains title of page, link to CSS stylesheet, sometimes scripts.
- **Body:** All the stuff that's actually rendered by the browser: text, tables, images, etc.

# HTML: Div and Span

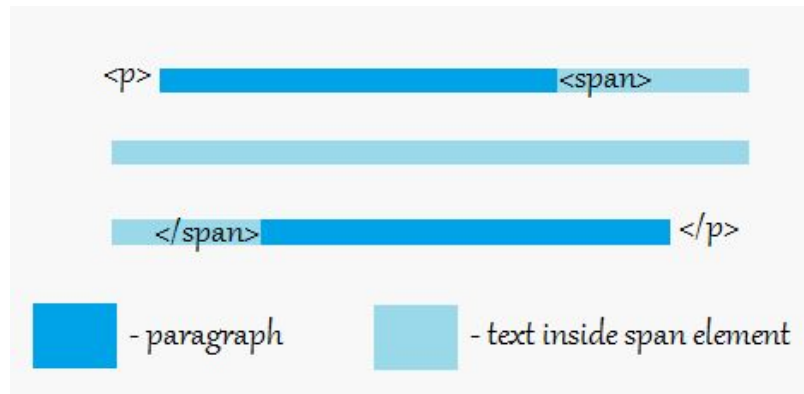
`<p>` = paragraph, `<h1>` = heading...

`<div>` and `<span>` have no semantic meaning, define presentation of document.  
(Can be used to "group" sections of the document.)

*div*: block element (by default)



*span*: inline element (by default)



# HTML: Attributes

Give element a custom "attribute" so that it can be picked out by a CSS selector.

ID: **uniquely** identifies object within a document (**don't re-use!**)

Classes: Think of as a "group" an element belongs to (can be repeated).

## HTML

```
<div id="large" >  
<div class="large">  
  
<div class="large">
```

## CSS

```
#large { // CSS styling by id}  
.large { //CSS styling by class}
```

# CSS Rules

- Selector + Declaration(s)
- **Selector:** Pick out element(s) by tag, class, or id.
  - Examples: `p {}`, `.large {}`, `#container {}`
  - Pseudo-selectors: `:hover {}`
  - Can combine: `p.large {}`, `div:hover{}`
- **Declaration:** Property-Value pair, specifying what the property should be.
  - Examples: `height: 300px;` `font-family: Arial`

# Must-know CSS

## Rule Conflicts / Overriding

- Cascade
- Inheritance

## CSS Display Models

- Box Model, Flexbox

## Useful Features

- Positioning, Shorthand, Pseudo Selectors



# Override Rules

# Cascade

CSS is "cascading" because of fixed priority of rules if more than one conflicting rule refers to the same object. [If two rules don't conflict, both are applied.]

## 1. Importance

!important properties override normal ones `color: red !important;`

(It is strongly discouraged to use !important explicitly)

User style sheets (e.g. browser default) overridden by author stylesheets.

## 2. Specificity

Declaration in style attribute has highest priority

id > class/pseudo-class > element type (eg. `#first` > `.tab` > `div`)

## 3. Source Code Order

# Inheritance

## CSS

```
div {  
    font-size: 15px;  
}
```

## HTML

```
<div> <p> This font is 15px. </p> </div>
```

Not everything is inherited! (Why?)

# Inheritance

Some properties inherit automatically (font size, font color) while others do not (background color).

Force inheritance by using the `inherit` property:

```
div { background-color: inherit; }
```

Percentages are a computed value, and imply inheritance (parent property needs to be defined)

```
div { font-size: 75%; }
```

More on cascade and inheritance:

[https://developer.mozilla.org/en-US/docs/Learn/CSS/Introduction\\_to\\_CSS/Cascade\\_and\\_inheritance](https://developer.mozilla.org/en-US/docs/Learn/CSS/Introduction_to_CSS/Cascade_and_inheritance)

# Display Models

# Box Model

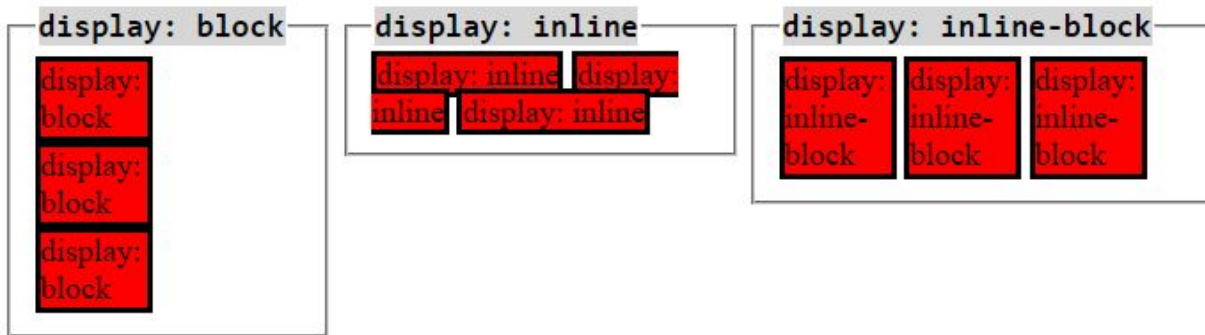


- width/height properties refer to the **content only** by default.
  - (box-sizing: content-box)
  - Have add content width + border + padding to get width of the whole thing.
- If you want width/height to include border and padding:
  - box-sizing: border-box

# Display

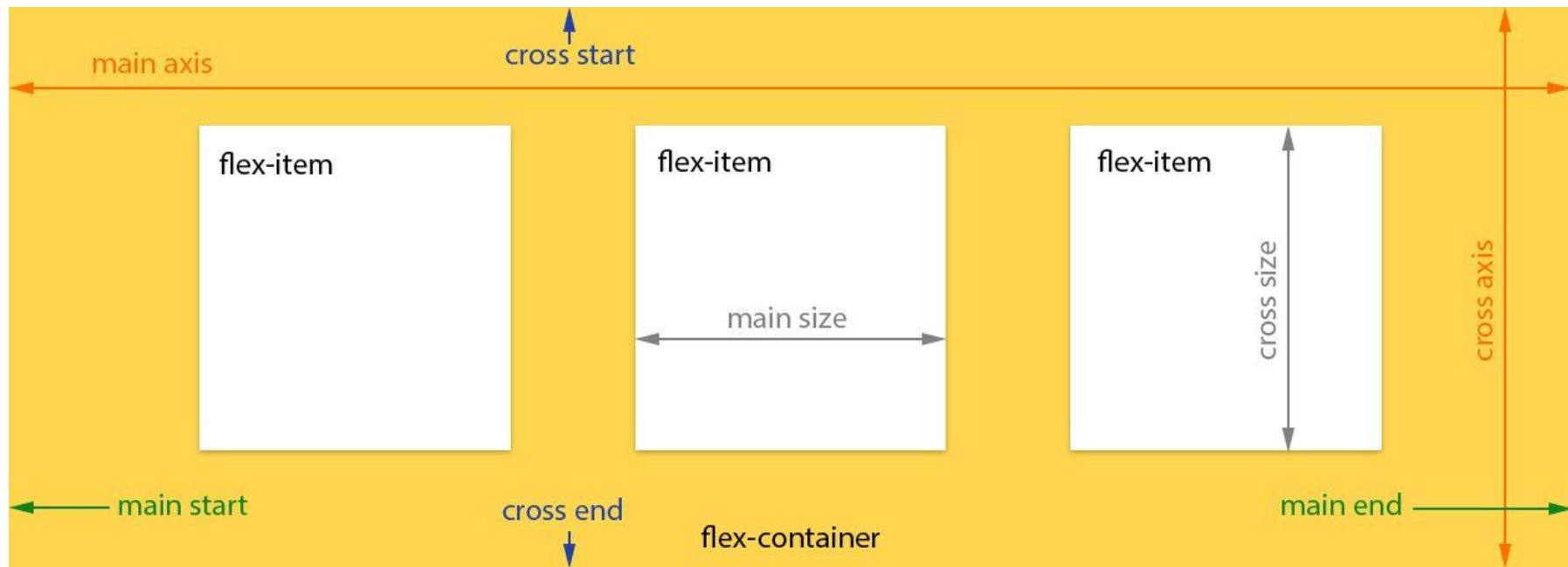
```
#element {  
    display: none;  
}
```

- none ( as opposed to visibility: hidden )
  - Question: What's the difference between these two?
- block, inline, inline-block, float
- flex



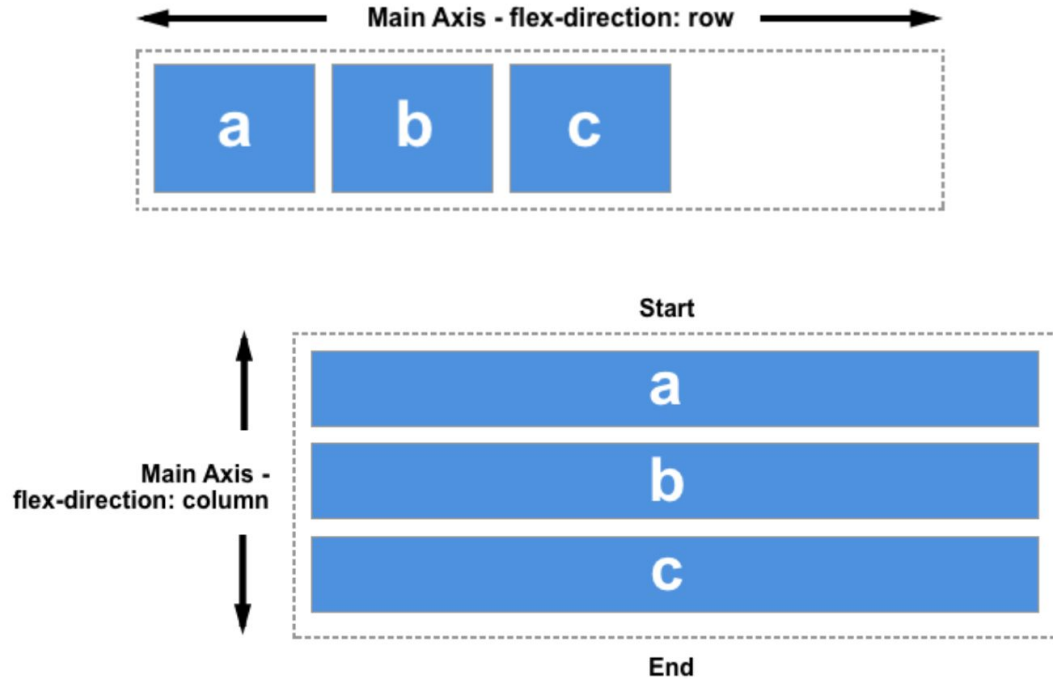
# Flexbox

cross and main instead of width and height





# Flex Container: Direction



# Flex Container: Alignment

- `justify-content`: determines how items spaced out along main axis
  - `left/right/center, space-between, space-around`
- `align-items`: determines how items aligned on cross axis

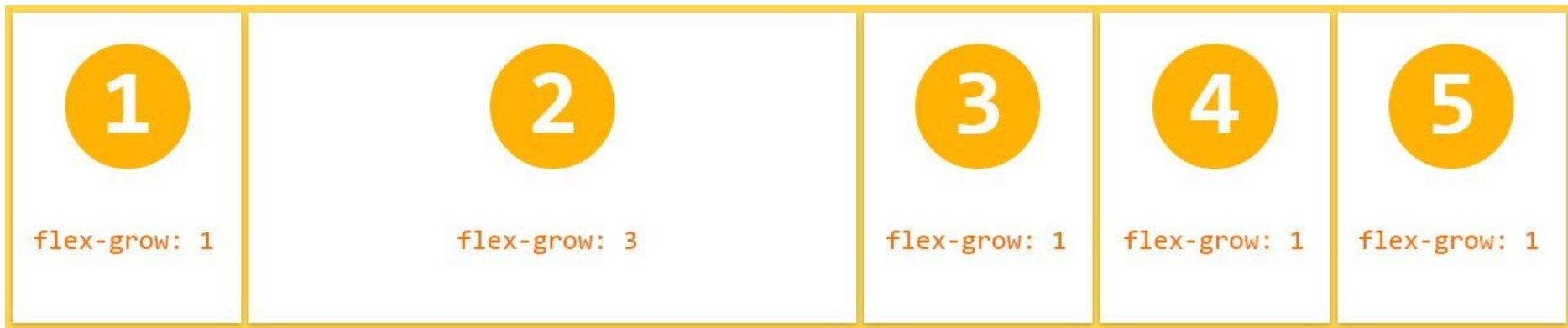


<https://scotch.io/tutorials/a-visual-guide-to-css3-flexbox-properties>

Justify Content Demo: <https://developer.mozilla.org/en-US/docs/Web/CSS/justify-content>

# Flex Items: Sizing

Elements can expand (or shrink) to fill available space



# Useful Features

# CSS Positioning

`position: static;`

- Item is positioned as it falls in the flow of the document.

`position: relative;`

- Item is *offset* relative to where it would be with static, using top, left, right, and/or bottom properties.

`position: fixed;`

- Item is positioned relative to browser window (viewport) with top, left, right, and/or bottom properties. This means if you scroll, item stays put.

`position: absolute;`

- Item is positioned relative to closest "positioned" (non-static) ancestor, offset by top, left, right, and/or bottom properties.

# CSS Shorthand

```
background-color: #000;
```

```
background-image: url(images/bg.gif);
```

```
background-repeat: no-repeat;
```

```
background-position: top right;
```

```
background: #000 url(images/bg.gif) no-repeat top right;
```

# Useful Pseudo-Class Selectors

`:nth-child(odd, even, 8)`

`:first/last-child`

`:hover :focus`

# Validation

XHTML 1.0 validated at: <https://validator.w3.org/>



# Debugging CSS (very useful!)

## Chrome Inspector (right-click > Inspect)

The screenshot displays the Chrome DevTools interface with the CSS Inspector open. The top section shows a preview of a web page with a header titled "Upcoming Projects" in blue text on an orange background. A tooltip indicates the element's dimensions: 555px x 26px. Below the preview, the "Elements" panel shows the DOM tree with the selected element highlighted: `<h3 class="left-border">Upcoming Projects</h3>`. The "Styles" panel on the right shows the computed styles for the selected element, including a border with a left color of `rgb(41, 128, 185)` and a width of `3px`. A diagram illustrates the box model with margin, border, and padding values.

Upcoming Lectures  
Cascading Style Sheets (CSS)  
Friday, April 1, 2016  
URLs and Links

Upcoming Projects  
h3.left-border 555px x 26px CSS  
Due: Thursday, April 7, 2016 at 11:59PM

Elements Console Sources Network Timeline Profiles Resources Security Audits Adblock Plus

```
<!DOCTYPE html>
<html>
  <#shadow-root (open)>
  <head>...</head>
  <body>
    <header>...</header>
    <main>
      <div class="container">
        ::before
        <h1>CS142: Web Applications (Spring 2016)</h1>
        <section>...</section>
        <section>...</section>
        <section>
          <!-- 604800 seconds is one week -->
          <div class="row">
            ::before
            <div class="col-md-6">...</div>
            <div class="col-md-6">
              <h3 class="left-border">Upcoming Projects</h3>
              <div>...</div>
            </div>
            ::after
          </div>
        </section>
        ::after
      </div>
    </main>
  </body>
</html>
```

html body main div.container section div.row div.col-md-6 h3.left-border

Styles Computed Event Listeners DOM Breakpoints Properties

margin 20  
border -  
padding 10  
542 x 26  
10

Filter Show all

- border-left-color `rgb(41, 128, 185)`
- border-left-style `solid`
- border-left-width `3px`
- box-sizing `border-box`
- color `rgb(51, 51, 51)`
- display `block`
- font-family `'Open Sans', Helvetica, sans-serif`
- font-size `24px`
- font-weight `500`
- height `26px`

# Use references frequently!

<https://developer.mozilla.org/en-US/docs/Web/CSS/Reference>

<https://www.w3schools.com/cssref/>