

CSC435: Web Programming

Lecture 5: Page layout and CSS box model

Bei Xiao

American University

Feb 2, 2018

Activity Outline

- More on Class and IDs
- Precedence
- CSS Box model
- Quiz
- Exercise

HTML id attribute

- `<p>Spatula City! Spatula City!</p>`
- `<p id="mission">Our mission is to provide the most`
- `spectacular spatulas and splurge on our specials`
- `until our customers <q>esplode</q> with splendor!</p>`

Spatula City! Spatula City!

Our mission is to provide the most spectacular spatulas and splurge on our specials until our customers “esplode” with splendor!

- allows you to give a unique ID to any element on a page
- each ID must be unique; can only be used once in the page

CSS ID selectors

```
#mission {  
    font-style: italic;  
    font-family: "Garamond", "Century Gothic", serif;  
}  
CSS
```

Spatula City! [Spatula City!](#)

Our mission is to provide the most spectacular spatulas and splurge on our specials until our customers "explode" with splendor!

output

- applies style only to the paragraph that has the ID of `mission`
- element can be specified explicitly: `p#mission {`

The HTML class attribute

- `<p class="shout">Spatula City! Spatula City!</p>`
- `<p class="special">See our spectacular spatula specials!</p>`
- `<p class="special">Today only: satisfaction guaranteed.</p>`

Spatula City! Spatula City!

See our spectacular spatula specials!

Today only: satisfaction guaranteed.

- classes are a way to group some elements and give a style to only that group
("I don't want ALL paragraphs to be yellow, just these three...")
- unlike an `id`, a `class` can be reused as much as you like on the page

CSS class selectors

```
• .special {                               /* any element with
  class="special" */
•   font-weight: bold;
• }
• p.shout {                               /* only p elements with
  class="shout" */
•   color: red;
•   font-family: cursive;
• }
CSS
```

Spatula City! Spatula City!

See our spectacular spatula specials!

Today only: satisfaction guaranteed.

output

- applies rule to any element with class `special`, or a `p` with class `shout`

Multiple classes

- `<h2 class="shout">Spatula City! Spatula City!</h2>`
- `<p class="special">See our spectacular spatula specials!</p>`
- `<p class="special shout">Satisfaction guaranteed.</p>`
- `<p class="shout">We'll beat any advertised price!</p>`

Spatula City! Spatula City!

We'll beat any advertised price!

- an element can be a member of multiple classes (separated by spaces)

CSS for following examples

```
.special {  
  background-color: yellow;  
  font-weight: bold;  
}  
.shout {  
  color: red;  
  font-family: cursive;  
}
```

CSS

Spatula City! Spatula City!

We'll beat any advertised price!

Precedence

- Browser default [lowest]
- External style sheet
- Internal style sheet
- Inline style [Highest]
 - *If a link to an external style sheet is placed after an internal style sheet, the external will take precedence!*

Precedence

- If two rules otherwise have equal precedence, the last one declared wins.
- The more specific rule has precedence.
 - “p#paragarph_1” has precedence over “p”
 - Sometimes this can be confusing:
 - Div.p.bio{ font-size: 14px}
 - #siebar p {font-size:12px}
 - Details here:

<http://www.vanseodesign.com/css/css-specificity-inheritance-cascaade/>

Precedence summary

- The more specific rule wins (according to a well-defined set of specificity rules).
- If two rules have equal specificity, the one that comes later wins.

CSS page layout

- Look at www.xkcd.com
- CSS can manage page layout

Generic HTML tags: `span` and `div`

These tags are for when no other more specific tag applies (like the English words “things” and “stuff”).

Span

A generic inline tag (like `` or ``)

Div

A generic block tag (like `<header>`, `<article>`, `<section>`, `<p>`)

CSS Attributes: width and height

- Block and inline elements normally have the height of their content.
- Inline elements have the width of their content
- Block elements have a width that stretches across the whole page.

width and height Example

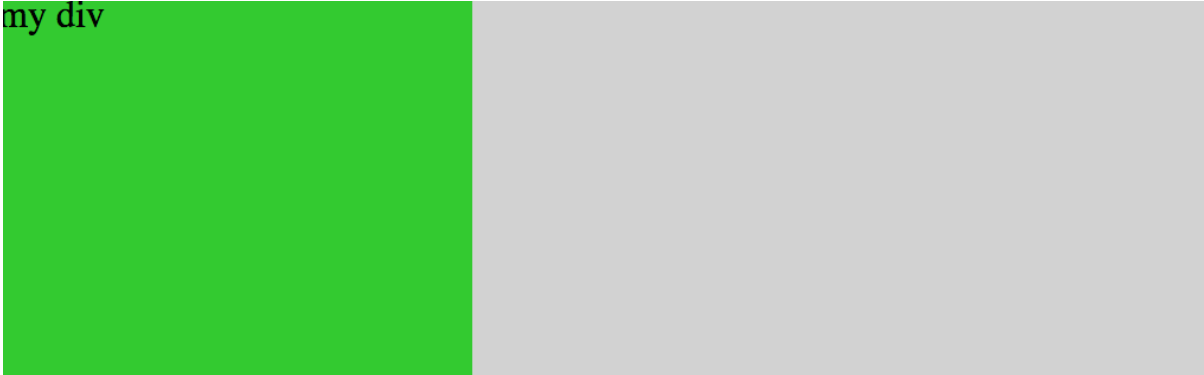
```
<div class='block'>my div</div>
```

HTML

```
div.block{  
  height:200px;  
  width:200px;  
  background-color: limegreen;  
}
```

CSS

my div



output

Box Model

Box Model



What?

Margin

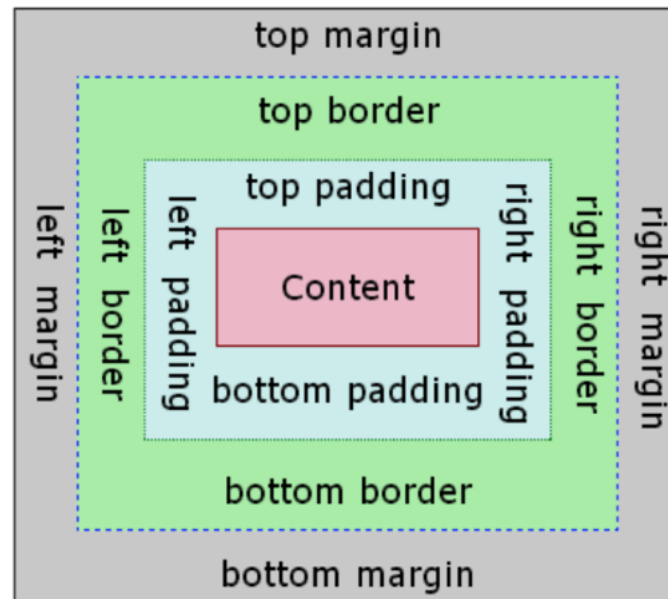
(outside) space between different elements

Border

(optionally visible) line that separates elements

Padding

(inside) space between elements content and border



Box Model

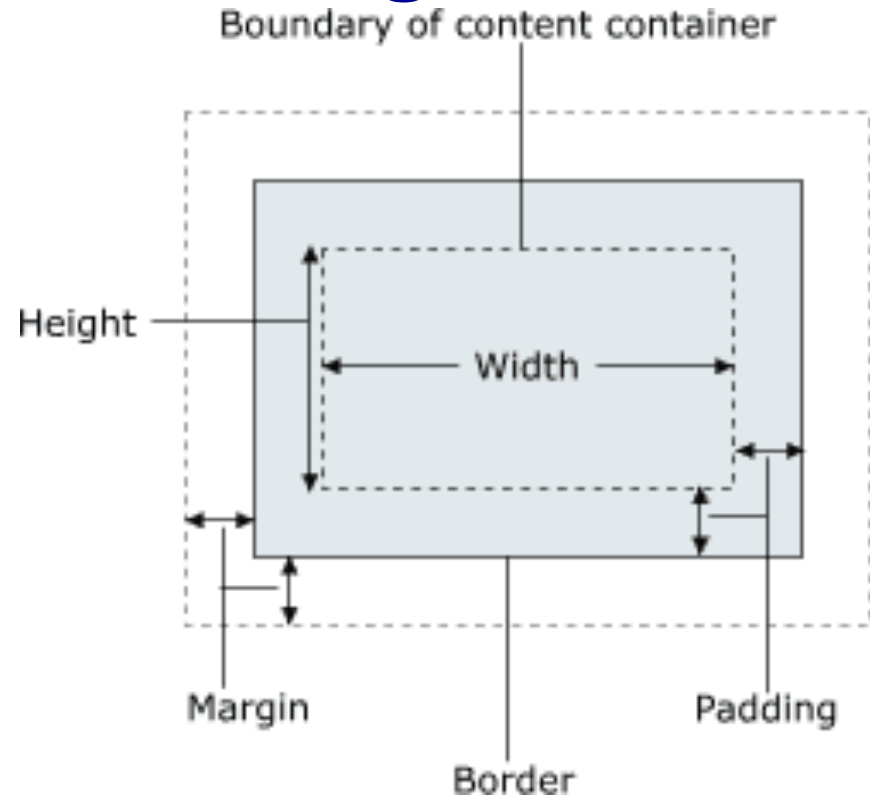
```
div {  
  width: 300px;  
  padding: 25px;  
  border: 25px solid navy;  
  margin: 25px;  
}
```

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Computing Box Dimensions

- “width” and “height” of an element refers to the **content**. If you have margins, padding, etc. Those also add to the dimensions of the box.
 - To compute the box dimension, make sure to account for the margins, etc. on both sides (e.g. both left and right margin)
- This is important because it is the **entire box**, and not merely the content, that takes space on the page.

Width and Height



width = content width + L/R padding + L/R border + L/R margin
height = content height + T/B padding + T/B border + T/B margin

Ways to specify size in CSS

- Pixels (px)
- Pt (1/72 inch) i.e. 12 pt font
- mm, cm, in
- Em, ex (for font), 12 pt font
- Percent % (usually of the browser window dimension)

CSS properties for borders

```
• h2 { border: 5px solid red; }
```

CSS

This is a heading.

output

property	description
border	thickness/style/color of border on all 4 sides

Thickness (specified in `px`, `pt`, `em`, or `thin`, `medium`, `thick`)

Style (`none`, `hidden`, `dotted`, `dashed`, `double`, `groove`, `inset`, `outset`, `ridge`, `solid`)

Color (specified as seen previously for text and background colors)

More Border Properties

- Specific properties of borders of all sides:
 - [border-color](#), [border-width](#), [border-style](#)
- All Properties for specific side:
 - [border-bottom](#), [border-left](#), [border-right](#), [border-top](#)

Properties of border on a particular side.

- [border-top-color](#)

CSS border example

```
h1 {  
  border-left: thick dotted #CC0088;  
  border-bottom-color: rgb(0, 128, 128);  
  border-bottom-style: double;
```

}
CSS

Heading 1

Output

- each side's border properties can be set individually
- if you omit some properties, they receive default values (e.g. border-bottom-width above)
- More on default:
- <https://css-tricks.com/the-css-box-model/>

Rounded corners with border-radius

```
p {  
  border: 3px solid blue;  
  border-radius: 12px;  
  padding: 0.5em;  
}
```

This is a paragraph.

This is another paragraph.
It spans multiple lines.

- each side's border radius can be set individually, separated by spaces

Margin example 1

```
p {  
  margin: 50px;  
  background-color: fuchsia;  
}
```

This is the first paragraph

This is the second paragraph

- notice that margins are always transparent (they don't contain the element's background color, etc.)

CSS properties for Margins

Specified in px, pt, em, rem, %, ex, cm (don't) (see more on [CSS units](#))

Property	
margin	Margin on all 4 sides
margin-bottom	Margin on bottom only
margin-left	Margin on left only
margin-right	Margin on right only
Margin-top	Margin on top side only
Complete list of margin properties	

```
p {  
  margin: 25px 50px 75px 100px;  
}
```

- top margin is 25px
- right margin is 50px
- bottom margin is 75px
- left margin is 100px

CSS properties of margins

- `p { width: 350px; background-color: yellow; }`
- `h2 { width: 50%; background-color: aqua; }`

This paragraph uses the first style above

An h2 heading

property	description
<u>width</u> , <u>height</u>	how wide or tall to make this element (block elements only)
<u>max-width</u> , <u>max-height</u> , <u>min-width</u> , <u>min-height</u>	max/min size of this element in given dimension

Center a block: auto margins

```
p {  
  margin-left: auto;  
  margin-right: auto;  
  width: 750px;  
}
```

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

- You can set the margin property to auto to horizontally center the element **within its container** (e.g. <p>).
- The element will then take up the specified width, and the remaining space will be split equally between the left and right margins

CSS Properties for padding

Again, specified in px, pt, em, rem, %, ex, cm (don't) (see more on [CSS units](#))

Property	
padding	padding on all 4 sides
padding-bottom	padding on bottom only
padding-left	padding on left only
padding-right	padding on right only
padding-top	Padding on top side only
Complete list of padding properties	

Alignment

- text-align
 - Align inline elements horizontally inside a parent container
- Vertical-align
 - Align inline elements vertically relative to the text

Example: Alignment

```
div {  
  text-align: right;  
}  
  
img {  
  vertical-align: middle;  
}
```

CSS

```
<div>  
  <p>  
    This is some text!  
      
  </p>  
</div>
```

HTML

Output

This is some text!



How to Center Page Elements (output next slide0

```
<body>
  <div class="block">
</body>
```

HTML

```
body {
  text-align: center; /** Doesn't work! */
}

.block {
  width: 100px;
  height: 100px;
  background-color: blue

  /** Try this instead: */
  /** margin-right: auto; */
  /** margin-left: auto; */
}
```

CSS



How do we center the blue box?

Confusing Stuff:

- `text-align` apply to a parent container to align the inline content within
- `Vertical-align` apply to inline items (usually those with a height, like an image) to vertically align them relative to other inline elements.
- `margin-left: auto; margin-right: auto` --- use auto margins and a width to center a block element in its parent.

Exercise 1: box model

- Download Exercise1.html from blackboard.
- Finish the declarations in the `<style>` tag that create the effects so that the paragraph box has the same width as the heading box.

Exercise 1 : box model

- Download Exercise1.html from blackboard.
- Finish the declarations in the <style> tag that create the effects so that the paragraph box has the same width as the heading box.
- Answer: $225(\text{width}) + 5(\text{left margin}) + 5(\text{left Border}) + 25(\text{left padding}) + 25(\text{right padding}) + 5(\text{right border}) + 10(\text{right margin}) = 300$

Exercise 2

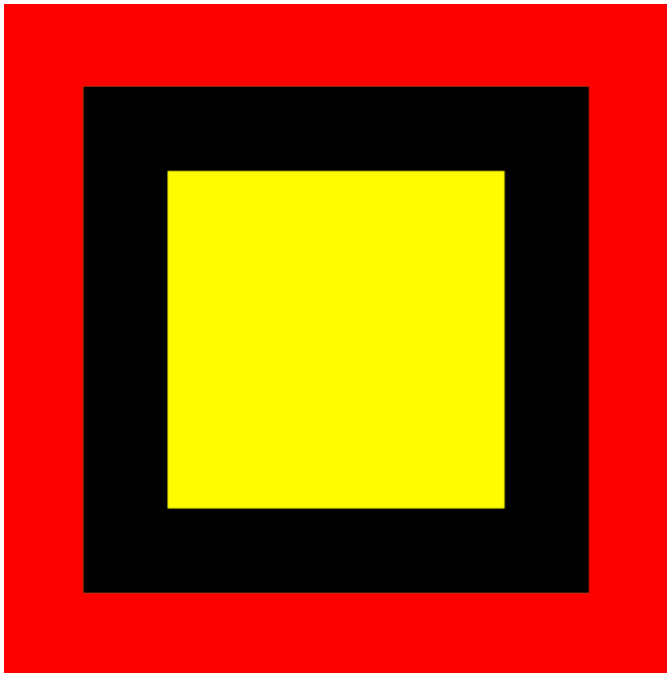
- Download Widthheightstarter.html and add a `<style>` section to make the webpage appear



The picture above is 300px wide.
The total width of this element is
also 300px.

Exercise 3

- Give boxes.html (download from blackboard), write boxes.css to make the appearance on the slide below.



The outer border of the box is red, the inner border of the box is black, and the inner background color of the box is yellow.

Both the outer and inner borders have a width of 50 pixels. The yellow portion of the box has a width and height of 200 pixels. The overall box has a width and height of 400 pixels.

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.

Next Class:

- CSS positioning
- Flow
- Review & Exercise: Journal page lay out
- Homework 1 due on Tuesday
- Next week: Guest lecture, Mike Ternor
- Onto JavaScript

Exercises & Take-home reading

- Finish resumu.html, resume.css
- Take-home tutorials (this will be so helpful for homework)

CSS positioning:

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

https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Box_Model/Introduction_to_the_CSS_box_model