ITMD-361
CLASS 5
SEPTEMBER 19, 2017

TONIGHT'S AGENDA

- Review Basic HTML & Layout
- Introduction to CSS
- CSS Cascade & Specificity
- CSS Box Model
- Modular Scale
- CSS Resets

WE HAVE A TEACHING ASSISTANT

Name: **Manish Vishwakarma**

Email: **mvishwakarma@hawk.iit.edu**

Office: **Perlstein Hall - Room 223**

Office Times:

- Wednesday 11:30-12:30
- Friday 11:30-12:30

ITMD-361

HTML REVIEW & BASIC LAYOUT CONSIDERATIONS

ITMD-361

CSS INTRODUCTIONS

CSS INTRO

Advantages of CSS

- Precise control of visual appearance and layout
- Saves work – allows you to change something in one place and have it effect elements in multiple areas
- Allows you to markup your HTML semantically and keep presentation elements separate
- All browsers mostly support CSS 2, and some CSS 3
- CSS 3 support in browsers is growing. Some browsers require prefixes for some properties.

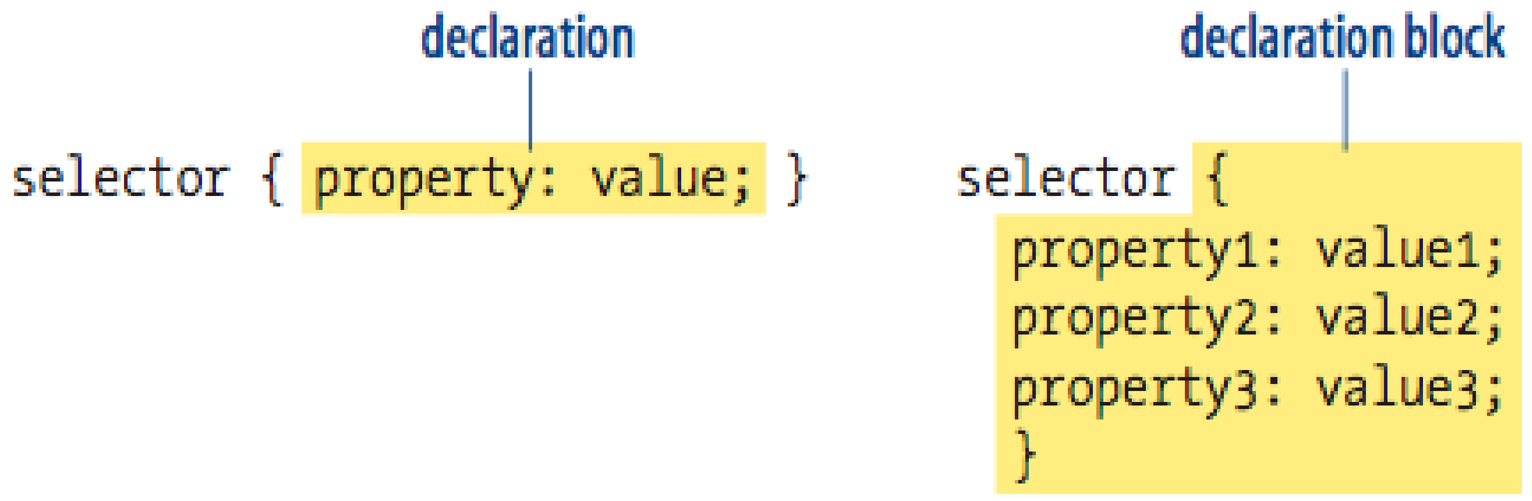
Disadvantages

- None major
- Minor one, some browser inconsistencies, don’t expect things to look exactly the same in all browsers. It will be real close but sometimes things will be slightly different.

CSS INTRO

- **Cascading Style Sheets (CSS)**
- **Made up of selectors and rules that define the visual style of elements**
- **CSS provides the rules to aesthetically style your webpage**
  - Change fonts and the way text looks, colors
  - Width, height, background colors and images
  - Positioning
  - Lines and space between elements
- **Rules cascade**
  - An algorithm defining how to combine properties
  - Ones defined later override or build upon earlier ones
  - More specific ones override earlier ones
  - https://developer.mozilla.org/en-US/docs/Web/CSS/Cascade

ANATOMY OF A CSS RULE



```
selector { property: value; }
```

```
selector {
  property1: value1;
  property2: value2;
  property3: value3;
  }
```

CSS INTRO

- **First Start with well formed HTML markup**
  - After we talk about introductory CSS concepts we will discuss CSS layout and planning your website using mockups to guide you in HTML markup
- **Next determine what style rules need to be written and what elements need to be targeted**
  - CSS rules are targeted with elements, ids, and classes at their most basic form by selectors.
- **Attach your styles to your document in one way**
  - Inline, external, or embedded
- **This in addition to the order you define them will determine some of the specificity**

ADDING CSS TO YOUR PAGE

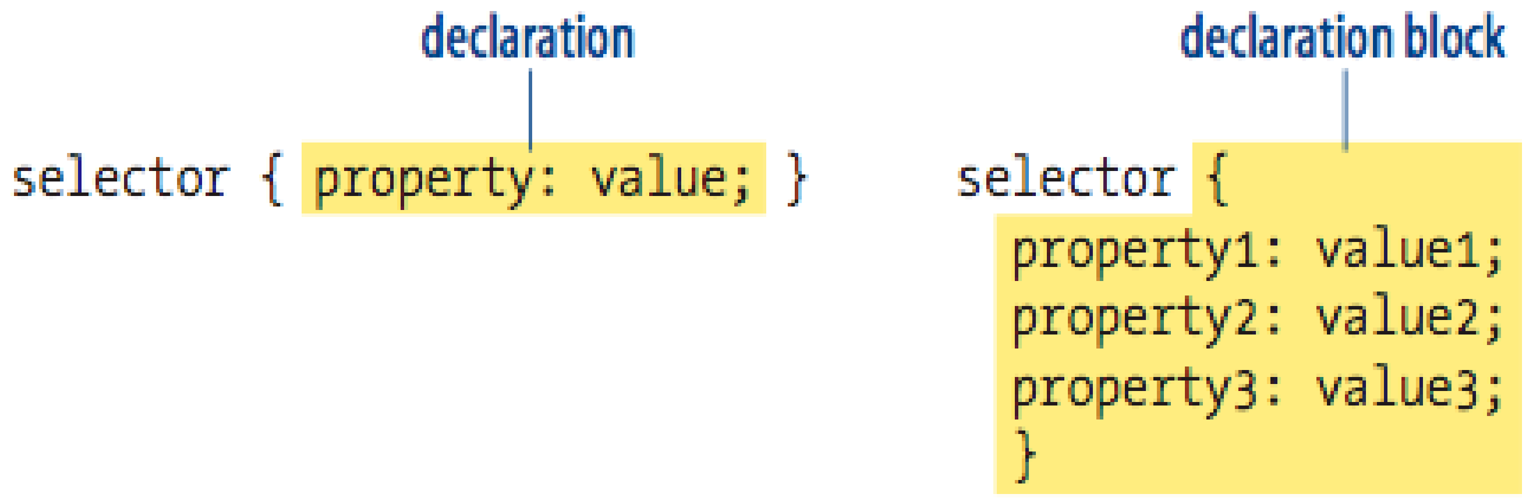
Three main methods

- **External Style Sheet**
  - Text document with a .css extension
  - CSS File is linked to the HTML document in the head section
  - <link rel="stylesheet" href="style.css" />
  - Preferred way, separates presentation in another file
- **Embedded Styles in Page**
  - Styles go in head section between <style></style> tags
- **Inline Styles**
  - Styles go in the element tag in the style attribute
  - <div style="color: #FFF; border: 1px solid #343;"></div>

CSS PROPERTIES

- **Different Properties take different value types**
- **Measurement values should have NO SPACE between number and value, 3px not 3 px**
    - %, px, em are the most common but there are more
    - Pixels are not created equally: See comparisons
- **Properties with color values – Common Methods**
    - Hex RGB Hexadecimal values, #34D2FF, #4D2
    - RGB, rgb(red, green, blue), 8bit 0-255, rgb(100,210,255)
    - Predefined Color Names, 147 named colors
    - Other methods supported in modern browsers
        - RGBA, HSL, HSLA – alpha value is a decimal 0 – 1
- **Properties that take a URL need the value to be wrapped in functional notation url()**
- **Reference the course book and api documentation to see what values a given property will accept.**
- **https://developer.mozilla.org/en-US/docs/Web/CSS/Reference**

ANATOMY OF A CSS RULE AGAIN



```
selector { property: value; }
```

```
selector {
  property1: value1;
  property2: value2;
  property3: value3;
  }
```

CSS CONCEPTS

- **Inheritance**
  - Some properties inherit their settings from their parent element
  - Mostly styles that effect text
- **Parents & Children**
  - Elements nested inside other elements are said to be children of that element.
  - The element that a given element is nested inside is its parent.
- **It is very important that you understand the way your HTML is structured and how each of your elements are nested within each other to be successful writing complex CSS rules.**

CSS INHERITANCE

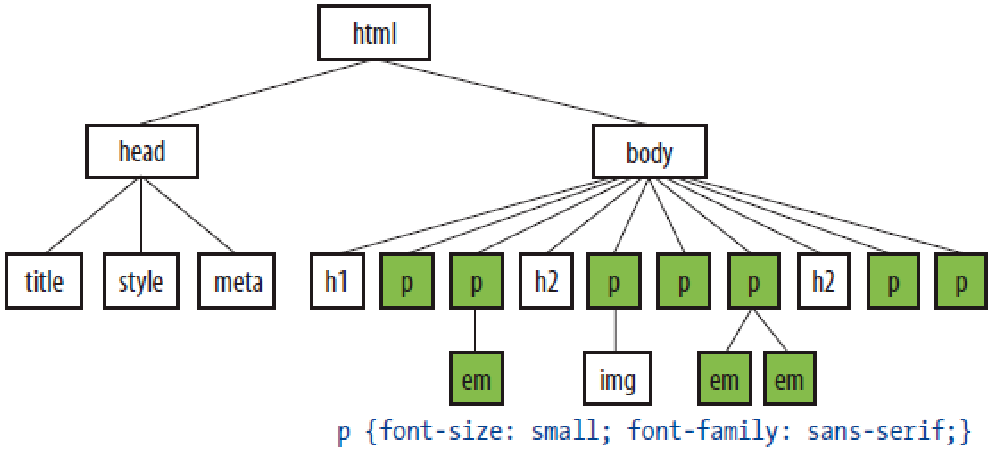


Figure 11-7. Certain properties applied to the **p** *element are inherited by their children.*

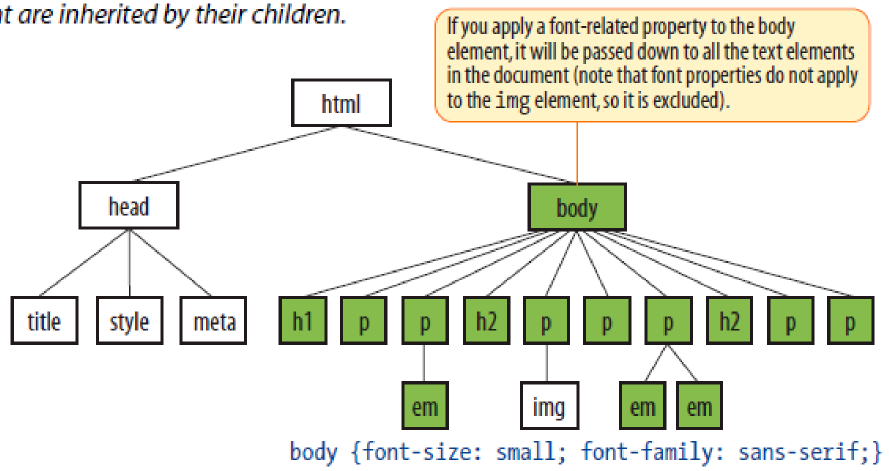

If you apply a font-related property to the body element, it will be passed down to all the text elements in the document (note that font properties do not apply to the img element, so it is excluded).
html
head
body
title
style
meta
h1
p
p
h2
p
p
p
h2
p
p
em
img
em
em
body {font-size: small; font-family: sans-serif;}

Figure 11-8. All the elements in the document inherit certain properties applied to the **body** *element.*

font-family

Values:	*one or more font or generic font family names, separated by commas* | inherit
Default:	*depends on the browser*
Applies to:	*all elements*
Inherits:	*yes*

Use the **font-family** property to specify a font or list of fonts (known as a font stack) by name, as shown in these examples.

```
body { font-family: Arial; }
var { font-family: Courier, monospace; }
p { font-family: "Duru Sans", Verdana, sans-serif; }
```

font-size

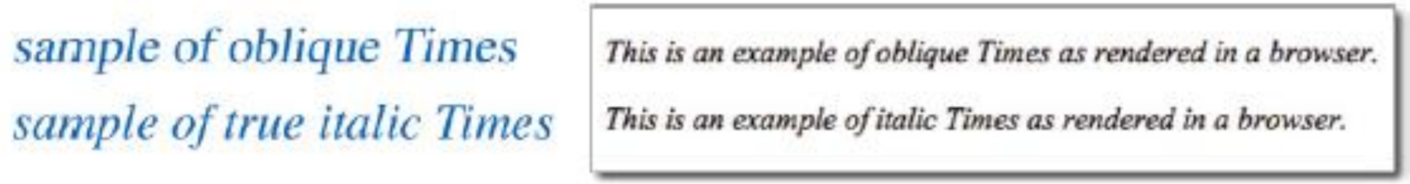
Values:	*length unit* | *percentage* | xx-small | x-small | small | medium | large | x-large | xx-large | smaller | larger | inherit
Default:	medium
Applies to:	*all elements*
Inherits:	*yes*

font-weight

Values:	`normal` | `bold` | `bolder` | `lighter` | `100` | `200` | `300` | `400` | `500` | `600` | `700` | `800` | `900` | `inherit`
Default:	`normal`
Applies to:	*all elements*
Inherits:	*yes*

font-style

Values:	`normal` | `italic` | `oblique` | `inherit`
Default:	`normal`
Applies to:	*all elements*
Inherits:	*yes*


sample of oblique Times

sample of true italic Times

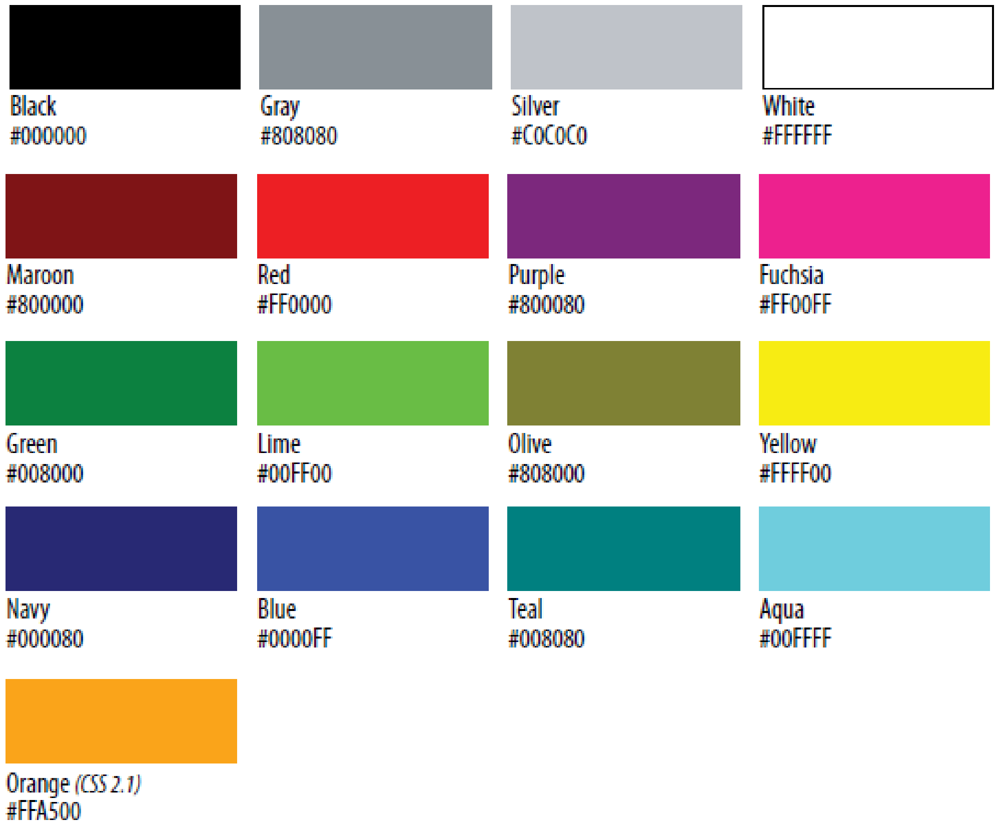
This is an example of oblique Times as rendered in a browser.

This is an example of italic Times as rendered in a browser.

color

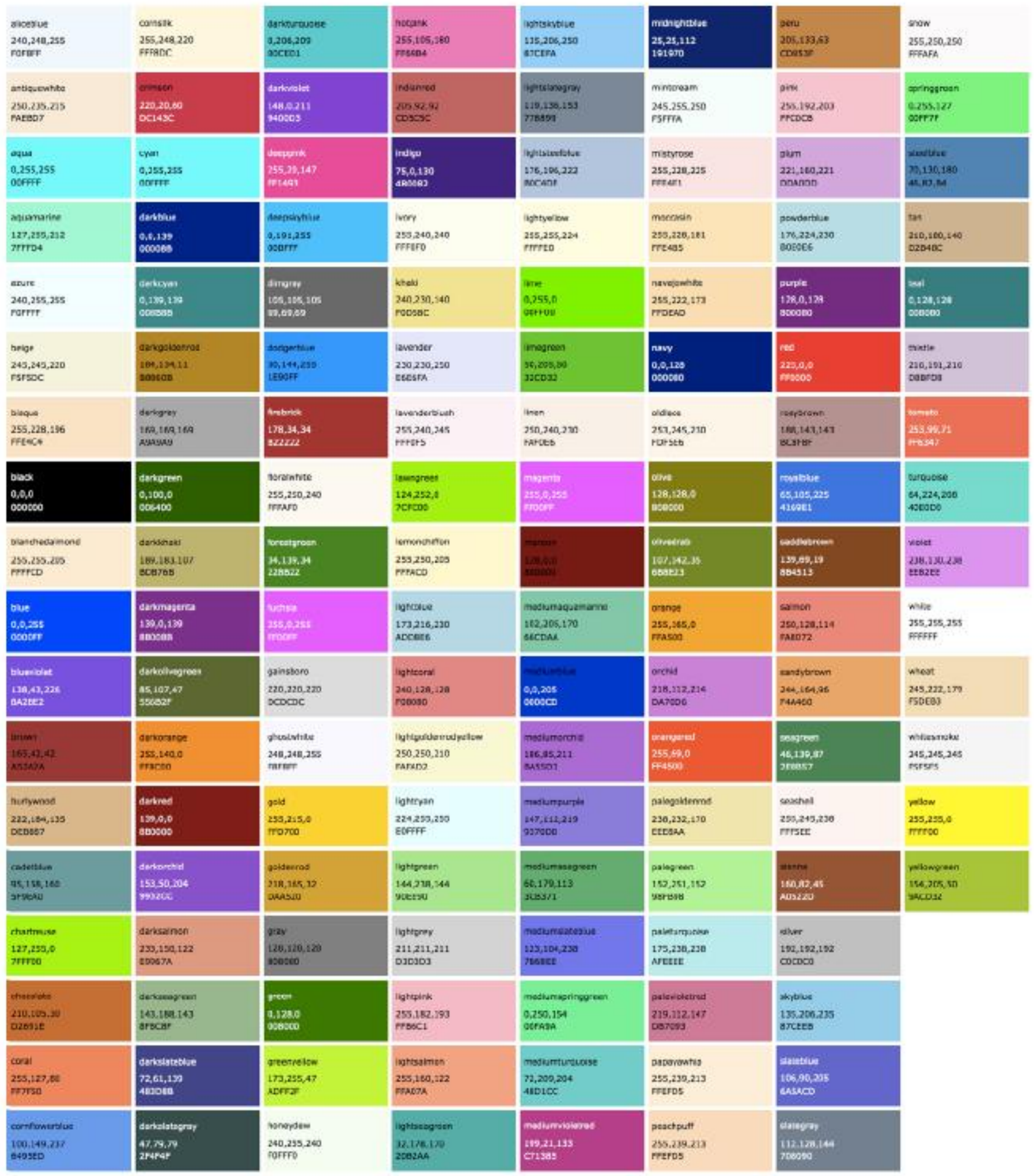
Values:	*color value (name or numeric)* | `inherit`
Default:	*depends on the browser and user's preferences*
Applies to:	*all elements*
Inherits:	*yes*

```
h1 { color: gray; }
h1 { color: #666666; }
h1 { color: #666; }
h1 { color: rgb(102,102,102); }
```



Black #000000	Gray #808080	Silver #C0C0C0	White #FFFFFF
Maroon #800000	Red #FF0000	Purple #800080	Fuchsia #FF00FF
Green #008000	Lime #00FF00	Olive #808000	Yellow #FFFF00
Navy #000080	Blue #0000FF	Teal #008080	Aqua #00FFFF
Orange *(CSS 2.1)* #FFA500			

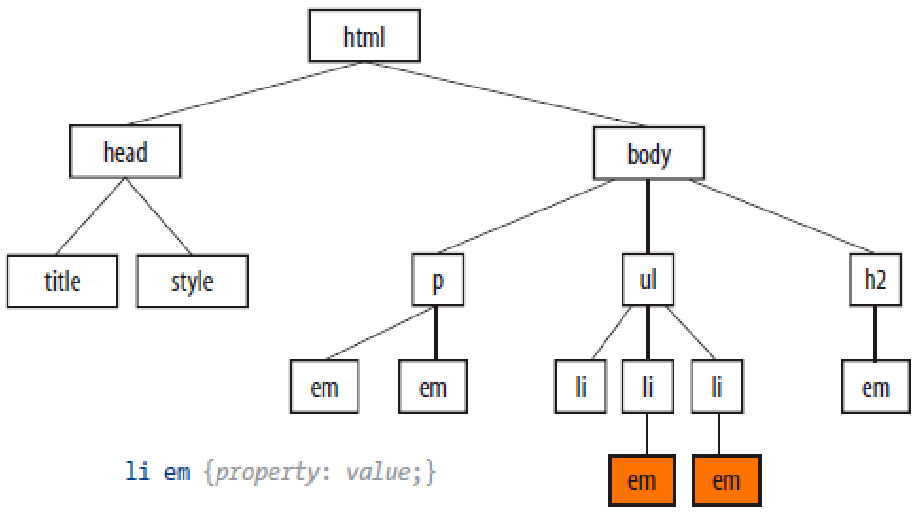
Figure 13-1. The 17 standard color names in CSS2.1.



aliceblue 240,248,255 F0F8FF	cornsilk 255,248,220 FFF8DC	darkturquoise 0,206,209 00CED1	hotpink 255,105,180 FF69B4	lightskyblue 135,206,250 87CEFA	midnightblue 25,25,112 191970	peru 205,133,63 CD853F	snow 255,250,250 FFFAFA
antiquewhite 250,235,215 FAEBD7	crimson 220,20,60 DC143C	darkviolet 148,0,211 9400D3	indianred 205,92,92 CD5C5C	lightslategray 119,136,153 778899	mintcream 245,255,250 F5FFFA	pink 255,192,203 FFC0CB	springgreen 0,255,127 00FF7F
aqua 0,255,255 00FFFF	cyan 0,255,255 00FFFF	deeppink 255,20,147 FF1493	indigo 75,0,130 4B0082	lightsteelblue 176,196,222 B0C4DE	mistyrose 255,228,225 FFE4E1	plum 221,160,221 DDA0DD	steelblue 70,130,180 4682B4
aquamarine 127,255,212 7FFFD4	darkblue 0,0,139 00008B	deepskyblue 0,191,255 00BFFF	ivory 255,240,240 FFF0F0	lightyellow 255,255,224 FFFFE0	moccasin 255,228,181 FFE4B5	powderblue 176,224,230 B0E0E6	tan 210,180,140 D2B48C
azure 240,255,255 F0FFFF	darkcyan 0,139,139 008B8B	dimgray 105,105,105 69,69,69	khaki 240,230,140 F0E68C	lime 0,255,0 00FF00	navajowhite 255,222,173 FFDEAD	purple 128,0,128 800080	teal 0,128,128 008080
beige 245,245,220 F5F5DC	darkgoldenrod 184,134,11 B8860B	dodgerblue 30,144,255 1E90FF	lavender 230,230,250 E6E6FA	limegreen 50,205,50 32CD32	navy 0,0,128 000080	red 255,0,0 FF0000	thistle 216,191,216 D8BFD8
bisque 255,228,196 FFE4C4	darkgray 169,169,169 A9A9A9	firebrick 178,34,34 B22222	lavenderblush 255,240,245 FFF0F5	linen 250,240,230 FAF0E6	oldlace 253,245,230 FDF5E6	rosybrown 188,143,143 BC8F8F	tomato 253,99,71 FF6347
black 0,0,0 000000	darkgreen 0,100,0 006400	floralwhite 255,250,240 FFFAF0	lawngreen 124,252,0 7CFC00	magenta 255,0,255 FF00FF	olive 128,128,0 808000	royalblue 65,105,225 4169E1	turquoise 64,224,208 40E0D0
blanchedalmond 255,255,205 FFFFCD	darkkhaki 189,183,107 BDB76B	forestgreen 34,139,34 228B22	lemonchiffon 255,250,205 FFFACD	maroon 128,0,0 800000	olivedrab 107,142,35 6B8E23	saddlebrown 139,69,19 8B4513	violet 238,130,238 EE82EE
blue 0,0,255 0000FF	darkmagenta 139,0,139 8B008B	fuchsia 255,0,255 FF00FF	lightblue 173,216,230 ADD8E6	mediumaquamarine 102,205,170 66CDAA	orange 255,165,0 FFA500	salmon 250,128,114 FA8072	white 255,255,255 FFFFFF
blueviolet 138,43,226 8A2BE2	darkolivegreen 85,107,47 556B2F	gainsboro 220,220,220 DCDCDC	lightcoral 240,128,128 F08080	mediumblue 0,0,205 0000CD	orchid 218,112,214 DA70D6	sandybrown 244,164,96 F4A460	wheat 245,222,179 F5DEB3
brown 165,42,42 A52A2A	darkorange 255,140,0 FF8C00	ghostwhite 248,248,255 F8F8FF	lightgoldenrodyellow 250,250,210 FAFAD2	mediumorchid 186,85,211 BA55D3	orangered 255,69,0 FF4500	seagreen 46,139,87 2E8B57	whitesmoke 245,245,245 F5F5F5
burlywood 222,184,135 DEB887	darkred 139,0,0 8B0000	gold 255,215,0 FFD700	lightcyan 224,255,255 E0FFFF	mediumpurple 147,112,219 9370DB	palegoldenrod 238,232,170 EEE8AA	seashell 255,245,238 FFF5EE	yellow 255,255,0 FFFF00
cadetblue 95,158,160 5F9EA0	darkorchid 153,50,204 9932CC	goldenrod 218,165,32 DAA520	lightgreen 144,238,144 90EE90	mediumseagreen 60,179,113 3CB371	palegreen 152,251,152 98FB98	sienna 160,82,45 A0522D	yellowgreen 154,205,50 9ACD32
chartreuse 127,255,0 7FFF00	darksalmon 233,150,122 E9967A	gray 128,128,128 808080	lightgrey 211,211,211 D3D3D3	mediumslateblue 123,104,238 7B68EE	paleturquoise 175,238,238 AFEEEE	silver 192,192,192 C0C0C0	
chocolate 210,105,30 D2691E	darkseagreen 143,188,143 8FBC8F	green 0,128,0 008000	lightpink 255,182,193 FFB6C1	mediumspringgreen 0,250,154 00FA9A	palevioletred 219,112,147 DB7093	skyblue 135,206,235 87CEEB	
coral 255,127,80 FF7F50	darkslateblue 72,61,139 483D8B	greenyellow 173,255,47 ADFF2F	lightsalmon 255,160,122 FFA07A	mediumturquoise 72,209,204 48D1CC	papayawhip 255,239,213 FFEFD5	slateblue 106,90,205 6A5ACD	
cornflowerblue 100,149,237 6495ED	darkslategray 47,79,79 2F4F4F	honeydew 240,255,240 F0FFF0	lightseagreen 32,178,170 20B2AA	mediumvioletred 199,21,133 C71585	peachpuff 255,239,213 FFEFD5	slategray 112,128,144 708090	

Figure 13-2. *The 140 extended color names in CSS3. Bear in mind that these will look quite different on a screen.*

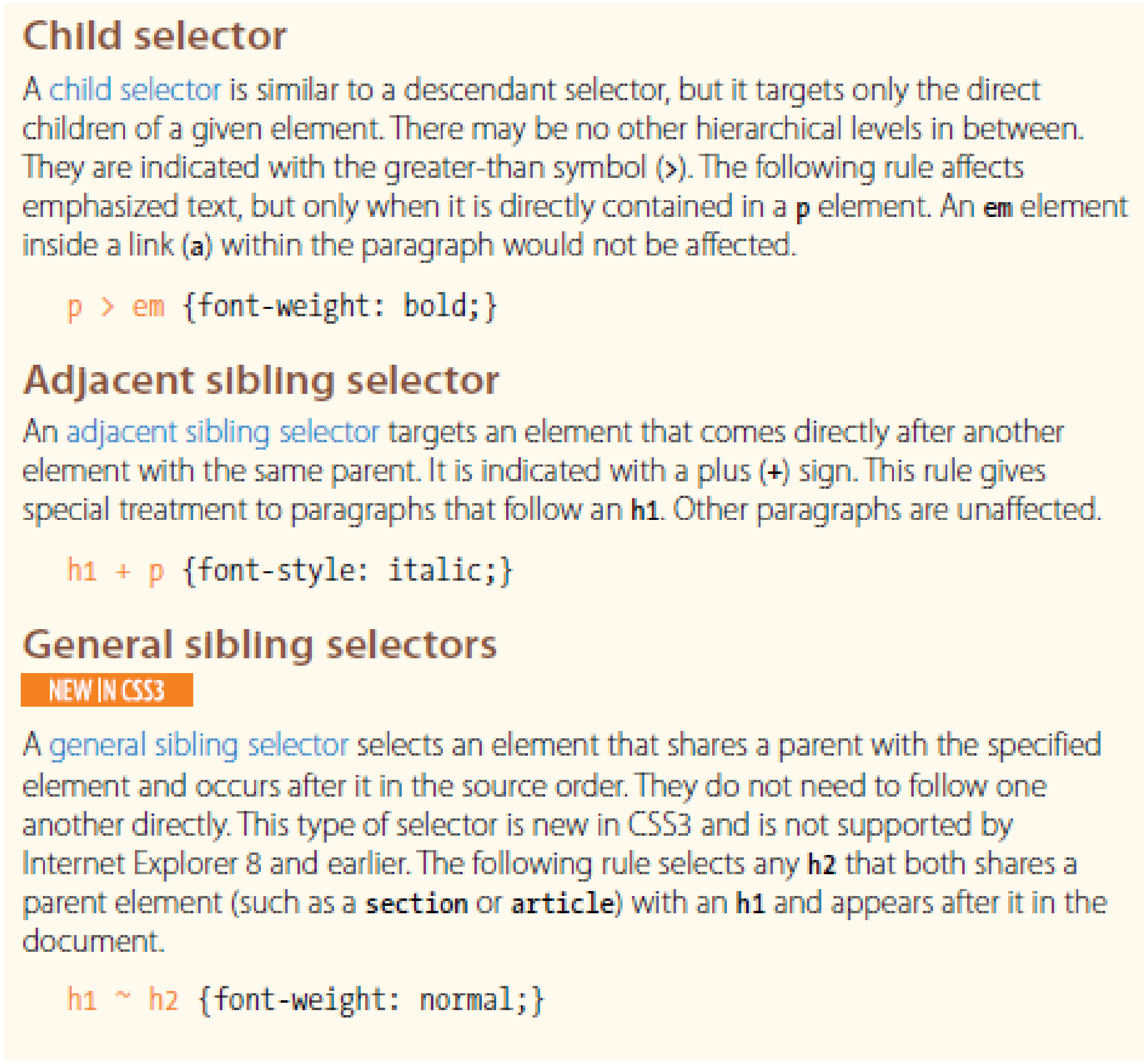
Element selector	`p { color: navy; }`
Grouped selectors	`p, ul, td, th { color: navy; }`



`li em {property: value;}`

It is also possible to nest descendant selectors several layers deep. This example targets `em` elements that appear in anchors (`a`) in ordered lists (`ol`).

```
ol a em { font-variant: small-caps; }
```



Child selector

A child selector is similar to a descendant selector, but it targets only the direct children of a given element. There may be no other hierarchical levels in between. They are indicated with the greater-than symbol (>). The following rule affects emphasized text, but only when it is directly contained in a **p** element. An **em** element inside a link (**a**) within the paragraph would not be affected.

```
p > em {font-weight: bold;}
```

Adjacent sibling selector

An adjacent sibling selector targets an element that comes directly after another element with the same parent. It is indicated with a plus (+) sign. This rule gives special treatment to paragraphs that follow an **h1**. Other paragraphs are unaffected.

```
h1 + p {font-style: italic;}
```

General sibling selectors

NEW IN CSS3

A general sibling selector selects an element that shares a parent with the specified element and occurs after it in the source order. They do not need to follow one another directly. This type of selector is new in CSS3 and is not supported by Internet Explorer 8 and earlier. The following rule selects any **h2** that both shares a parent element (such as a **section** or **article**) with an **h1** and appears after it in the document.

```
h1 ~ h2 {font-weight: normal;}
```

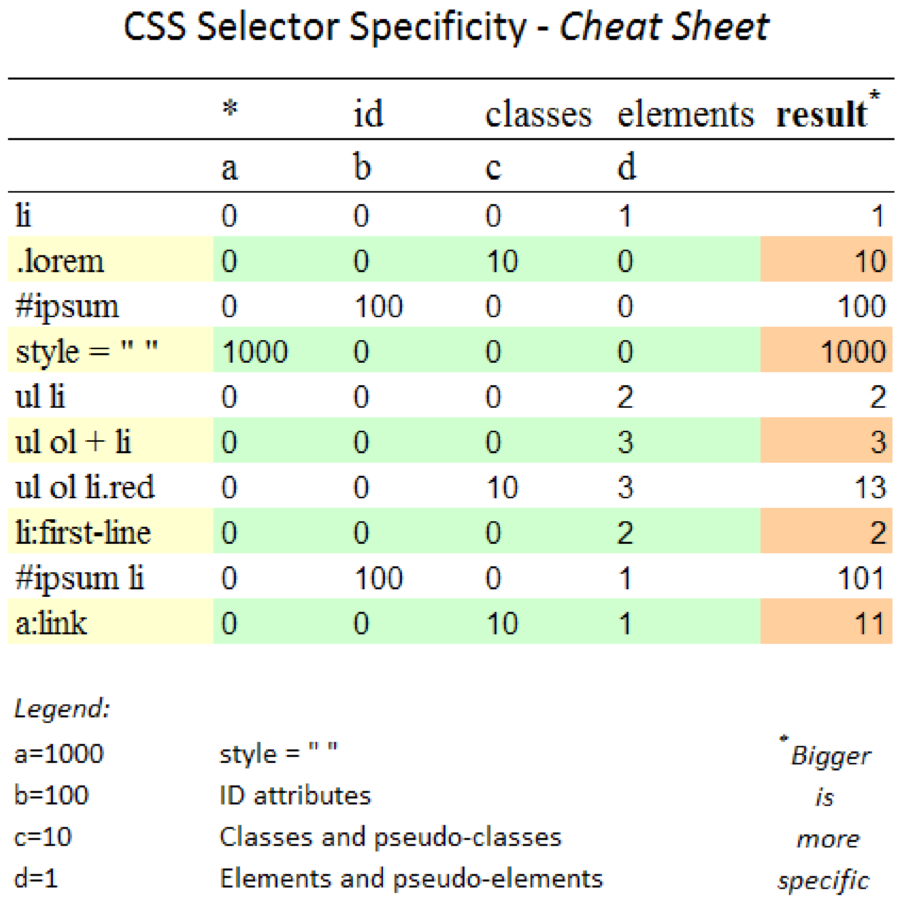
ITMD-361

CSS SPECIFICITY & BOX MODEL

CSS CASCADE & SPECIFICITY

- **Style passes down (cascades) until a rule with more weight overrides a previous style**
- **First goes by style sheet hierarchy**
- **Then goes by order defined in the style sheets**
- **If there is a conflict it resolves with a point system**
  - Creators developed a point system
  - inline is 1000, id is 100, class is 10, element is 1
- **See Charts on next two slides**
- **https://developer.mozilla.org/en-US/docs/Web/CSS/Cascade**
- **https://developer.mozilla.org/en-US/docs/Web/CSS/Specificity**
- **http://css-tricks.com/specifics-on-css-specificity/**
- **http://www.smashingmagazine.com/2007/07/27/css-specificity-things-you-should-know/**
- **http://code.tutsplus.com/tutorials/quick-tip-understanding-css-specificity--net-10963**

CSS SPECIFICITY



CSS Selector Specificity - *Cheat Sheet*

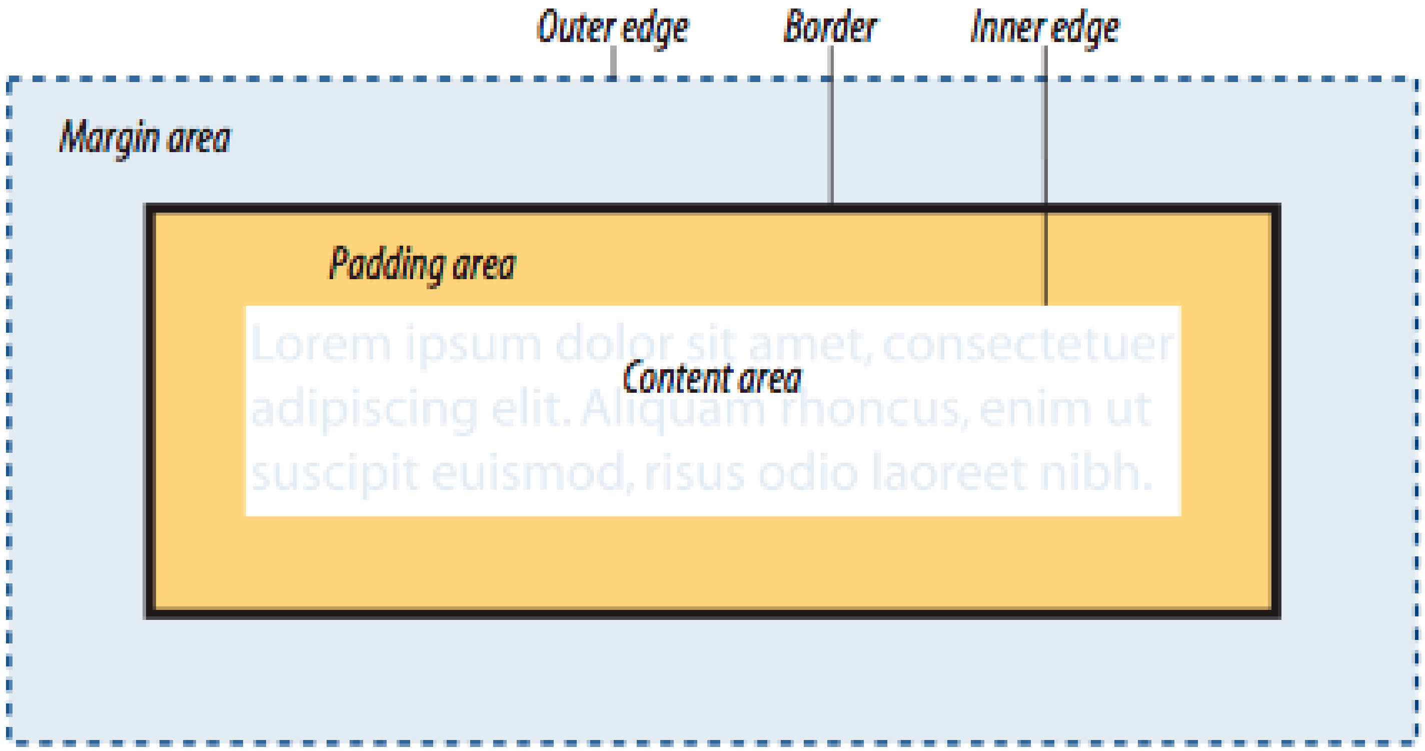
	*	id	classes	elements	**result**[*]
	a	b	c	d	
li	0	0	0	1	1
.lorem	0	0	10	0	10
#ipsum	0	100	0	0	100
style = " "	1000	0	0	0	1000
ul li	0	0	0	2	2
ul ol + li	0	0	0	3	3
ul ol li.red	0	0	10	3	13
li:first-line	0	0	0	2	2
#ipsum li	0	100	0	1	101
a:link	0	0	10	1	11

Legend:

a=1000	style = " "	[*] *Bigger*
b=100	ID attributes	*is*
c=10	Classes and pseudo-classes	*more*
d=1	Elements and pseudo-elements	*specific*

CSS BOX MODEL

The Browser sees every element, block or inline, as a little rectangular box


Outer edge
Border
Inner edge
Margin area
Padding area
Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aliquam rhoncus, enim ut suscipit euismod, risus odio laoreet nibh.
Content area

CSS BOX MODEL

Standard CSS Box Model

- **This is the standard way the box model is calculated.**
- **CSS *box-sizing* property is:**
  - Default: Set to content-box
- **Width of an element on screen is:**
  - *width + padding + border*
  - *width* property sets content area width
  - You must account for padding and borders when setting the width to get the exact size you want.
- **Supported in CSS 1**
- **https://developer.mozilla.org/en-US/docs/Web/CSS/box_model**
- **http://css-tricks.com/the-css-box-model/**

CSS BOX MODEL

New Border Box CSS Box Model

- **This is the new way the box model can be calculated.**
- **CSS *box-sizing* property is:**
  - Set to border-box
- **Width of an element on screen is:**
  - Whatever you set the width property to.
  - The browser will take the width of the padding and border out of the content area so your final box is the size you set.
- **CSS 3 Property that has fairly good browser support**
  - http://caniuse.com/#feat=css3-boxsizing
  - Even back to IE8
- **http://www.paulirish.com/2012/box-sizing-border-box-ftw/**
- **https://developer.mozilla.org/en-US/docs/Web/CSS/box-sizing**
- **http://css-tricks.com/box-sizing/**

MODULA SCALE

See My Link!

CSS RESETS

CSS RESETS

- Used to reset all css properties so they will be consistent in all browsers.
- Eric Meyer's css reset
  - Complete reset to no styles
  - http://meyerweb.com/eric/tools/css/reset/
- Normalize
  - Targets only styles that need to be reset
  - http://necolas.github.com/normalize.css/
- Yahoo YUI css reset
  - Mostly complete reset
  - http://yuilibrary.com/yui/docs/cssreset/
- http://www.cssreset.com/