CMPT 165

Unit 3 – CSS

May 27th, 2015

SFU SIMON FRASER UNIVERSITY

Today's Agenda

- Recap of relative URL
- More on styling with CSS
  - Styling in groups
  - Color formatting
  - …

Absolute vs. Relative URL

```
http://cmpt165.csil.sfu.ca/~lisat/SFU_logo.png
http://cmpt165.csil.sfu.ca/~lisat/index.html
```

index.html:

Under same directory

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
strict.dtd">
<html>
   <head><title>Title</title></head>
   <body>

     <h1>Heading</h1>
     <p><img src=“http://cmpt165.csil.sfu.ca/~lisat/SFU_logo.png”/></p>

     <p><img src=“SFU_logo.png” /></p>

   </body>
</html>
```

Absolute vs. Relative URL

```
http://cmpt165.csil.sfu.ca/~lisat/d1/SFU_logo.png
http://cmpt165.csil.sfu.ca/~lisat/index.html
```

index.html:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
strict.dtd">
<html>
   <head><title>Title</title></head>
   <body>

     <h1>Heading</h1>
     <p><img src=“http://cmpt165.csil.sfu.ca/~lisat/d1/SFU_logo.png”/></p>

     <p><img src=“d1/SFU_logo.png” /></p>

   </body>
</html>
```

Absolute vs. Relative URL

```
http://cmpt165.csil.sfu.ca/~lisat/d1/SFU_logo.png
http://cmpt165.csil.sfu.ca/~lisat/d2/index.html
```

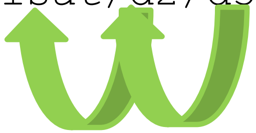
index.html:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"      "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
strict.dtd">
<html>
   <head><title>Title</title></head>
   <body>

     <h1>Heading</h1>
     <p><img src=“http://cmpt165.csil.sfu.ca/~lisat/d1/SFU_logo.png”/></p>

     <p><img src=“../d1/SFU_logo.png” /></p>

   </body>
</html>
```

“parent” directory

Absolute vs. Relative URL

```
http://cmpt165.csil.sfu.ca/~lisat/d1/SFU_logo.png
http://cmpt165.csil.sfu.ca/~lisat/d2/d3/index.html
```



“grandparent” directory

index.html:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"      "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
   <head><title>Title</title></head>
   <body>

     <h1>Heading</h1>
     <p><img src=“http://cmpt165.csil.sfu.ca/~lisat/d1/SFU_logo.png”/></p>

     <p><img src=“../../d1/SFU_logo.png” /></p>

   </body>
</html>
```

Absolute vs. Relative URL

```
http://cmpt165.csil.sfu.ca/~lisat/d1/s/SFU_logo.png
http://cmpt165.csil.sfu.ca/~lisat/d2/d3/index.html
```

index.html:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"       "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
strict.dtd">
<html>
   <head><title>Title</title></head>
   <body>

     <h1>Heading</h1>
     <p><img src="http://cmpt165.csil.sfu.ca/~lisat/d1/s/SFU_logo.png"/></p>

     <p><img src="../../d1/s/SFU_logo.png" /></p>

   </body>
</html>
```

Cascading Style Sheets (CSS)

What?

“Cascading” stylesheet?

- “Cascading”: answered later

Why CSS?

- You should know

How?

```
<style>
<link>
```

<style> tag

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
   <head>
        <title>Title</title>
        <style type="text/css”  media=“all”  >
        p {
         text-align: center;
        }

        h1 {
         text-align: center;
         color: red;
        }
        </style>
   </head>
   <body>
        <h1>Heading</h1>
        <p>This is a paragraph.</p>
        <p>This is another paragraph.</p>
        <h1>Heading 2</h1>
        <p>This is yet another paragraph.</p>
        <p>More paragraphs.</p>
   </body>
</html>
```

Required attribute

Optional attribute

Contents of <style> tag!

CSS syntax

Syntax of content of CSS/style tag

- Selector may refer to any number of elements that you are formatting
- Declarations are separated by semicolons!

“Curly brace”

Declaration

```
selector1 {
        property_1: value_1;
        property_2: value_2;
        …
        property_N: value_N;
}
selector2 {
        property_1: value_1;
        property_2: value_2;
        …
        property_M: value_M;
}
```

Example declarations:

```
ul {
list-style-type: circle;
}
ol {
start: c;
}
h1 {
text-align: center;
}
h2 {
text-align: center;
}
```

Ways to define styles

1. In the header

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">

<html>
   <head>
         <title>Title</title>
         <style type="text/css”>
          p{
              text-align: center;
              color: blue;
          }
         </style>
   </head>
   <body>
         <h1>Heading</h1>
         <p>This is a paragraph.</p>


   </body>
</html>
```

2. “Inline” (FYI)

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">

<html>
   <head>
         <title>Title</title>






   </head>
   <body>
     <h1>Heading</h1>
     <p style=“text-align: center;
     color: blue;”>This is a
     paragraph.</p>

   </body>
</html>
```

Ways to define styles

3. Linking to an external stylesheet

second_page.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
   <head>
         <title>Page 2</title>
         <link href=“my_style.css" type="text/css" />
   </head>
   <body>
        <h1>Heading</h1>
        <p>This is yet another page.</p>
   </body>
</html>
```

inde

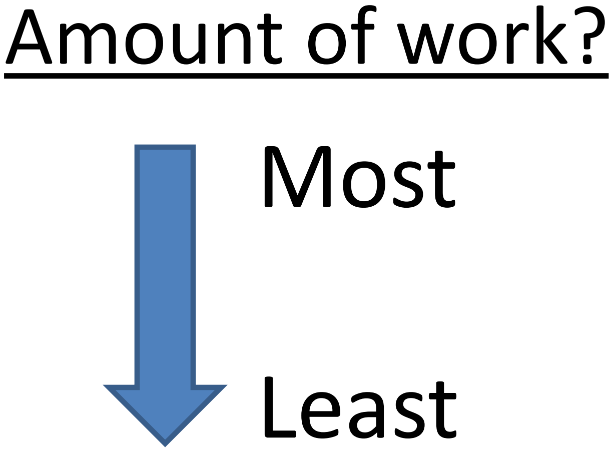
```
<!DOC
<htm
   <
   <
   <
     <p>This is another paragraph.</p>
   </body>
</html>
```

my_style.css

```
p {
  text-align: center;
 }
```

3 ways to styling markup

1. Inline
2. In the header
3. In a separate CSS file



Amount of work?

Most

Least

CSS: multiple selectors

Grouping declarations by selector:

my_style.css

```
h1, h2, h3 {
   text-align: center;
 }
```

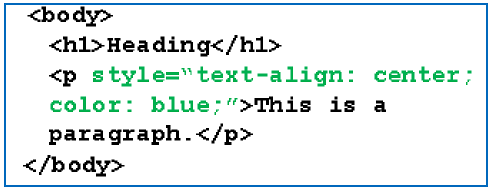
```
p, h1, h2, h3, h4, h5, h6 {
    color: red;
}
```

```
p, h1, h2, h3 {
   text-align: center;
   color: red;
 }
```

More reasons for using CSS

- Link to an external CSS file produces smaller files
  - Smaller files ➔ faster transfer times
- Define style once
  - Consistency
  - Easily update appearance of entire site
    - E.g. Apply different color schemes for different seasons
- Easier to read markup/stylesheet
- Easier for search engine to parse them
  - Increase chances of user finding your webpages

Minimize clutter:



```
 <body>
   <h1>Heading</h1>
   <p style="text-align: center;
   color: blue;">This is a
   paragraph.</p>
</body>
```

CSS: background

- Versions=levels:
  - Level 1: released in Dec 1996
  - Level 2: released in May 1989
  - Level 2.1: released in June 2011
  - Level 3 is being developed
  - Level 4 is planned for the future

  - CMPT 165 uses CSS Level 1

CSS Level 1

Style support for:

- **Color** of element text, backgrounds, etc.
- **Alignment** of elements (text, images, etc.)
- **Text** formatting: e.g. spacing of words, letters, lines
- **Font** properties: typeface and emphasis
- **Boxing**: margin, border, padding, and positioning
- **Unique identification**: explained later
- **Generic classification**: explained later

Colors

17 standard color presets

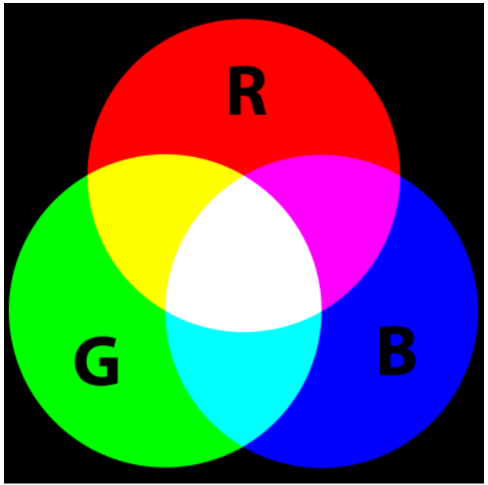
- White
- Silver
- Gray
- Black
- Red
- Maroon
- Yellow
- Olive
- Orange
- Lime
- Green
- Aqua
- Teal
- Blue
- Navy
- Fuchsia
- Purple

RGB model

- “primary colors”: Red + Green + Blue
- “Additive” model:

  i.e. Red + Green = Yellow

  Red + Blue = Purple



http://en.wikipedia.org/wiki/File:AdditiveColor.svg

RGB coding:

- System for specifying web colors
- A numerical value reflecting *intensity* (“strength”) of a color channel: e.g.
  - 5 is brighter than 0
  - 8 is brighter than 2
- 3 channels → 3 sets of values, e.g. Bright red given by 900

RGB model

Red Green Blue

Examples:	Possible RGB encoding:
• Bright green?	0 9 0
• Dark green?	0 5 0
• Bright blue?	0 0 9
• Dark red?	5 0 0
• Dark blue?	0 0 5
• Purple?	5 0 5
• Black?	0 0 0
• Gray?	5 5 5

Color specification via hex triplet

Decimal system (base=10):

0 1 2 3 4 5 6 7 8 9

Darkest → Brightest

Hexadecimal system (base=16):

0 1 2 3 4 5 6 7 8 9 A B C D E F

Darkest → Brightest

Example:

Brightest red? RGB = X X X?

= F 0 0

RGB model

Red Green Blue

↓

0 1 2 3 4 5 6 7 8 9 A B C D E F

Examples:

	Decimal	Hexadecimal
• Bright green?	0 9 0	0 F 0
• Dark green?	0 5 0	0 8 0
• Bright blue?	0 0 9	0 0 F
• Dark red?	5 0 0	8 0 0
• Dark blue?	0 0 5	0 0 8
• Purple?	5 0 5	8 0 8
• Black?	0 0 0	0 0 0
• Gray?	5 5 5	8 8 8

RGB model

“Truecolor”: 6-digit hexadecimal code

Hexadecimal system (base=16):

0 1 2 3 4 5 6 7 8 9 A B C D E F

Dark → Bright

00 01 02 …. 88 … FE FF

Dark → Bright

Example:

Bright red? RGB= XX XX XX?

= FF 00 00

Side notes: 24-bit true color

- Use 1 byte/channel
- 1 byte = X bits?
  - X = 8
  - 1 “binary” bit gives 2 variations: {0 1} ← (see Week#1)
  - 2^8 = ?
  - 2^8 = 256 variations per channel
    - How many hexadecimal numbers do you need for each channel?
    - 16 x 16 = 256 ➔ 2 hexadecimal numbers
      - Need 2 hexadecimal numbers to represent 1 byte
- 3 bytes = Y bits?
  - 3 bytes = 3 bytes x 8 bit/byte = 24 bits ➔ “24-bit true color”

RGB model

Red Green Blue

↓

0 1 2 3 4 5 6 7 8 9 A B C D E F

Examples:

	Decimal	Hexadecimal
• Bright green?	090	0F0 = 00FF00
• Dark green?	050	080 = 008800
• Bright blue?	009	00F = 0000FF
• Dark red?	500	800 = 880000
• Dark blue?	005	008 = 000088
• Purple?	505	808 = 880088
• Black?	000	000 = 000000
• Gray?	555	888 = 888888

Color specification in CSS

Start with hash key (#):

```
p {
    background-color: #0F0;
    color: #A1AF00;
}

body {
    background: #00FF00;
    color: #0F0;
}

hr {
    border-color: #0a0;
}
```

Q: Why might this be a poor choice?

Side notes:

- Use American spelling “color”
- British spelling “colour” won’t work

Color specification in CSS

1. Presets: “red”, “blue”, …
2. Hexadecimal code

   a) 3-byte hexadecimal → 6 digits, e.g. #0000FF

   b) Abbreviated form: 3 (hexadecimal)-digit, e.g. #00F
3. Function notation: rgb(value1,value2,value3)

Side note on function notation:
function(input1,input2,input3)=output

Recall: 256 intensity values per channel, e.g.

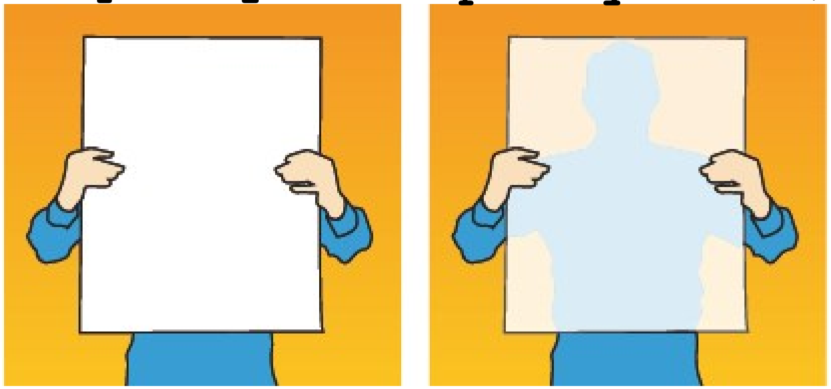
Brightest red:	`rgb(255,0,0);`
	`rgb(300,0,0); /*FYI, same as above*/`
Grey:	`rgb(128,128,128);`
Brightest red:	`rgb(100%,0%,0%);`

Opacity property

- May add transparency to any element
- "Opacity"
  - Amount of light absorption
    - Strength of 0 to 1
    - 1 means 100% absorbed
- Examples:

```
hr {
        border-color: red;
        opacity: 0.5;
}
img {
        opacity: 0.25;
}
```

`opacity: 1;` `opacity: 0.25;`



http://www.archive.europapier.com/service/knowhow/testingpaper/visual-characteristics/img/opacity.jpg

Today’s Highlights

- More motivations for using CSS
- CSS
  - Formatting colors
  - Hexadecimal color coding
  - Color specification
  - Opacity (transparency)
- Commenting in markup and stylesheets

Commenting in markup

Sometimes you need to add comments (not to be displayed by browser)…

```
<!-- this is a comment -->
```

index.html:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
strict.dtd">
<html>
   <head><title>Title</title></head>
   <body>

     <h1>Heading</h1> <!-- this will be ignored -->
     <p><img src=“an_image.png” alt=“A random image” /></p>
     <!-- this image was downloaded from www.abc.ca -->

   </body>
</html>
```

Commenting

In stylesheets:

```
/* this is a comment */
```

```
hr {
     border-color: red; /* tested on May2 */
     border-style: solid;
}

p { /* color found at http://abc.ca */
     background-color: #0F0;
     color: #A0F00;
}
```

Questions?

Lab exercises

- Lab exercise 2:
  - XHTML + CSS
  - Due tomorrow 23:59pm
  - TA office hours today
- Lab exercise 1: grades released