# HACK YALE

## < INTRO TO WEB DEVELOPMENT />

WWW.HACKYALE.COM

HACK YALE

< INTRO TO WEB DEVELOPMENT />

# Week 2

## INTRODUCTION TO CSS

# INTRODUCTION TO CSS

HACK YALE

# Cascading Style Sheets

CSS is the language used to specify the styling of web pages.  This is the “skin” to HTML’s “skeleton.”

Like HTML, CSS has a set of very simple grammar rules.

# Adding CSS to our HTML

## Method 1: Use the <style> tag.

```
<style type="text/css">
   CSS HERE
</style>
```

## Method 2: Use the style attribute.

```
<tag style="css rules for this tag">
   HTML HERE
</tag>
```

# Adding CSS to our HTML

## Method 3: Link to an external CSS file (preferred)

```
<!DOCTYPE html>
<html>
   <head>
      <link rel="stylesheet" type="text/css"
href="MyStyleSheet.css" />
   </head>
   <body>
      HTML BODY HERE
   </body>
</html>
```

# CSS Grammar

CSS documents each consist of a list of rules to apply to the corresponding HTML document.

Each rule takes the following form:

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

# Comments

Comments in CSS are ignored by browsers. To start a comment, use the /* opening. The comment will continue until it is closed by */

```
selector {
    property: value;
    /* inactive properties:
    property: value;
    property: value; */
    property: value; /* active */
}
```

# Selectors

CSS selectors are used to select elements in an HTML document.

Selectors have a lot of flexibility so we can specify exactly when / where we want our styling rules to apply.

We can select elements by type, id, class, attribute, etc.

# Selectors

To apply a rule to all elements of one type, specify the element type as the selector.

Example: Text that falls within an <h1> should be red.

```
h1 {
   color: red;
}
```

# Selectors

To apply a rule to any element with a given id, specify the id as the selector using #.

Example: Any element with the id “foo” should be colored green.

```
#foo {
  color: green;
}
```

# Conflicts

If we have these two rules at the same time and have an h1 with the id “foo,” which should win?

```
h1 {
  color: red;
}

#foo {
  color: green;
}
```

# Specificity

If we have these two rules at the same time and have an h1 with the id “foo,” which should win?

There are many ways for rules to conflict in CSS. There are policies for determining which rules apply in conflicts.

We will discuss CSS’ specificity policies next time.

# Selectors

To apply a rule to any element with a given class, specify the class as the selector using .

Example: Any element with the class “important” should have bold text.

```
.important {
  font-weight: bold;
}
```

# Selectors

You can combine these rules together.

Example: Any paragraph with the class “important” should have bold text.

```
p.important {
   font-weight: bold;
}
```

# Selectors

You can combine these rules together.

Example: Any paragraph OR any element with the class “important” should have bold text.

```
p, .important {
  font-weight: bold;
}
```

# Capitalization

Case-sensitive when matching class & id names.

Case-insensitive when matching HTML tags.

If you want to avoid problems, write all your HTML and CSS in lowercase.

Caution: Make sure id and class names never start with a number!

# Whitespace

CSS ignores whitespace, so you can have as much or as little space between selectors as you want.

Pro Tip: Make your code beautiful.  It should be easy to read and follow a consistent style.


Typical indentation pattern in CSS:

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

# Properties

Different HTML elements have different properties that can apply to them.

Many properties are applicable to all tags.

All available properties for a given tag can be found online.

# Colors & Background Colors

```
body {
   color: navy;
   background-color: beige;
}
```

Colors can be set using color names, but you can get more specific colors using RGB (Red-Green-Blue).

# Why weird RGB value range?

RGB values each range from 0-255.

Why?

The basic unit of memory on a computer is a bit, which is either a 0 or a 1.

8 bits = 1 byte.

# A little bit of comp sci :)

There are $2^8$ possible combinations of 0's and 1's in a byte (8 bits).

$2^8 = 256$, so we can store all the possible color values for a single color (0-255) using only one byte.  We need 3 bytes to store R, G, B.

How does the computer translate 0's and 1's to a number between 0 and 255?

# Intro to Binary

```
0  = 00000000        8  = 00001000
1  = 00000001        9  = 00001001
2  = 00000010       10  = 00001010
3  = 00000011       11  = 00001011
4  = 00000100       12  = 00001100
5  = 00000101       13  = 00001101
6  = 00000110       14  = 00001110
7  = 00000111       15  = 00001111
```

# Intro to Binary

Binary is “base 2,” which means it’s like counting if we had 2 fingers.  Instead of moving to the next column after 9, we move to the next column after 1.

The columns are 1’s, 2’s, 4’s, 8’s, 16’s, 32’s, …

The number 11111111, which is the biggest number you can have with 8 bits, is 255.

128+64+32+16+8+4+2+1 = 255

# Intro to Binary

```
0  = 00000000        8  = 00001000
1  = 00000001        9  = 00001001
2  = 00000010       10  = 00001010
3  = 00000011       11  = 00001011
4  = 00000100       12  = 00001100
5  = 00000101       13  = 00001101
6  = 00000110       14  = 00001110
7  = 00000111       15  = 00001111
```

# Color with RGB

```
body {
   color: rgb(250, 250, 170);
   background-color: rgb(80, 150, 180);
}
```

We can specify colors using any RGB values we want.

There are tools available for helping you figure out the RGB values of any color you see on the web.

# Hexadecimal notation

We can also specify RGB values using hexadecimal notation. This practice is common in the design world.

Idea: instead of using base 2 to represent binary, use base 16 so we can represent 4 bits at once.

# Intro to Hexadecimal

```
0 = 0000
1 = 0001
2 = 0010
3 = 0011
4 = 0100
5 = 0101
6 = 0110
7 = 0111
8 = 1000
9 = 1001
A = 1010
B = 1011
C = 1100
D = 1101
E = 1110
F = 1111
```

# Intro to Hexadecimal

Hexadecimal is “base 16,” which means it’s like counting if we had 16 fingers. Instead of moving to the next column after 9, we move to the next column after F.

The columns are 1’s, 16’s, 256’s, …

255, which is 11111111 in binary, is FF in hexadecimal. F is 15, so we have 15-16’s and 15-1’s. 15(16) + 15(1) = 255.

# Color with RGB

```
body {
   /* same as RGB values, now in HEX */
   color: #fafaaa;
   background-color: #5096b4;
}
```

The notation is much shorter with hexadecimal, but might be harder to read at first.

You can get to the point where reading hexadecimal is as easy as reading decimal.

# Alignment

```
body {
   text-align: left;
}
```

Alignment options: left, right, center, justify

# Text Styling

```
body {
  text-align: left;
  text-decoration: underline;
  font-weight: bold;
  font-style: italic;
}
```

font-weight: 100-900, normal=400, bold=700
text-decoration: none or underline (usually)
font-style: normal or italic

# Text Styling

```
body {
   font-size: 16px;
   font-family: Helvetica, Arial;
}
```

We will choose the first font from the font-family that is available on the client's computer.

# Link Styling

You can style <a> tags differently based on the state they are in:

```
a:link      An unvisited link
a:visited   A visited link
a:hover     Mouse is hovering
a:active    On click
```

# Link Styling

Example: underline links only when hovering.

```
a { /* general rule for links */
   text-decoration: none;
}

a:hover { /* more specific rule */
   text-decoration: underline;
}
```

# Span Tags

HTML has a special tag called  that does nothing.

This tag gives you a hook in your HTML document so you can apply special styling.

```
<p class="story">
The girl in the <span style="color:
red;">red hood</span> was walking
through the <span style="color:
green;">thick forest</span>.
</p>
```

# Homework

Homework has been posted on the course website: https://github.com/hackyale/Web-Development-101

The third assignment can be found under assignments/week_2.md

# HACK YALE

## < INTRO TO WEB DEVELOPMENT />

WWW.HACKYALE.COM