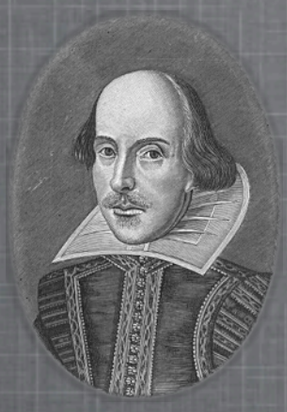
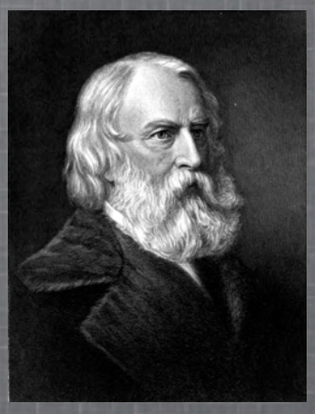
CSS

HTML: Looking Back

- HTML dictates order, structure, and function
- Does very little to specify layout or visual rendering

With HTML, we developed learned about elements in which we could place our content.
We built the items, but we've said nothing of how they should be painted, decorated, or arranged. Or, for an analogy combo, you could say that we've developed a vocabulary--of nouns, verbs, and a couple adjectives, perhaps, but a vocabulary nonetheless!
With enough time we could certainly string together these elements and make a paper--we learned how to make tables and, my gosh, we even learned how to place in some dazzling figures! Well, okay, that might pass the muster for grad students, but well, we're MIT students, and it's IAP--so let's not just make papers, let's be a little creative. Let's write some poetry!





Shakespeare

CSS

Longfellow

Let's Write Some Poetry!

YOU

An example of two accomplished writers--and after today, this will include you!

Images taken from Wikimedia Commons

C
S tyle
S

The Purpose of CSS

- If HTML is the content and meaning

  ➡ CSS helps to convey that meaning

- Allows developers to separate the content from layout and design
  - Content and design inherently different in nature

  ➡ Change in content does not require change in design

CSS Zen Garden

- Site using consistent HTML content
- Differing external CSS files create dramatically different layout
- Support for multiple browsers

link: http://www.csszengarden.com

hint: change the styles on the page

An Example

Consider the **boldface** font in the following examples:

Chunky bacon is delicious.

- Meaning is conveyed by the styling
- Remove the style and meaning is lost

Today I will go outside.
Monday I will run 2 miles.

- Attention is given to the information
- No additional meaning is lost when removed

Say it out loud. The “chunky” is meant to be said strongly. If you lose the intonation and expression, some of the meaning is lost along with it.

On the other hand, the bold font in the second example has no significant meaning. While it does help the viewer to read the data, it can be removed without losing the intrinsic value and meaning of the content.

A Bit of Reasoning

➡ **Hola.**

➡ *Hola!*

- Explicitly conveys meaning

<b>

➡ **Hola.**

<i>

➡ *Hola!*

- Typographical artifacts: any meaning is implied

Thinking about presentation in this manner, is should now be very clear why the tags and are promoted over their predecessors <b> and <i>, respectively. <b> and <i> both refer to layout and typographical preferences--they say nothing about meaning. In fact, any meaning extracted would be implied, at best, from common usage. and are much more explicit--they both mean something very real and just happen to correspond to the same default rendering in browsers.

In Detail

Today I will go outside.
Monday I will run 2 miles.

- Special attention is given to the date
- Assists the user in resolving, parsing the information

➡ Key goal of Design

The boldface font is used to draw attention to the date--to separate the date from the item that will be done. In essence, while the styling does not have intrinsic meaning, its presence enhances the user's ability to discern the meaning of the information. It makes it easier for the user to parse the information, scan the information, and make distinctions between sections of the information. This is the key goal of design: not only to attract the user to the information and assist them in finding use for the information.

CSS: How to Use It

1. Select the elements you want to modify

   ➡ **Selectors**

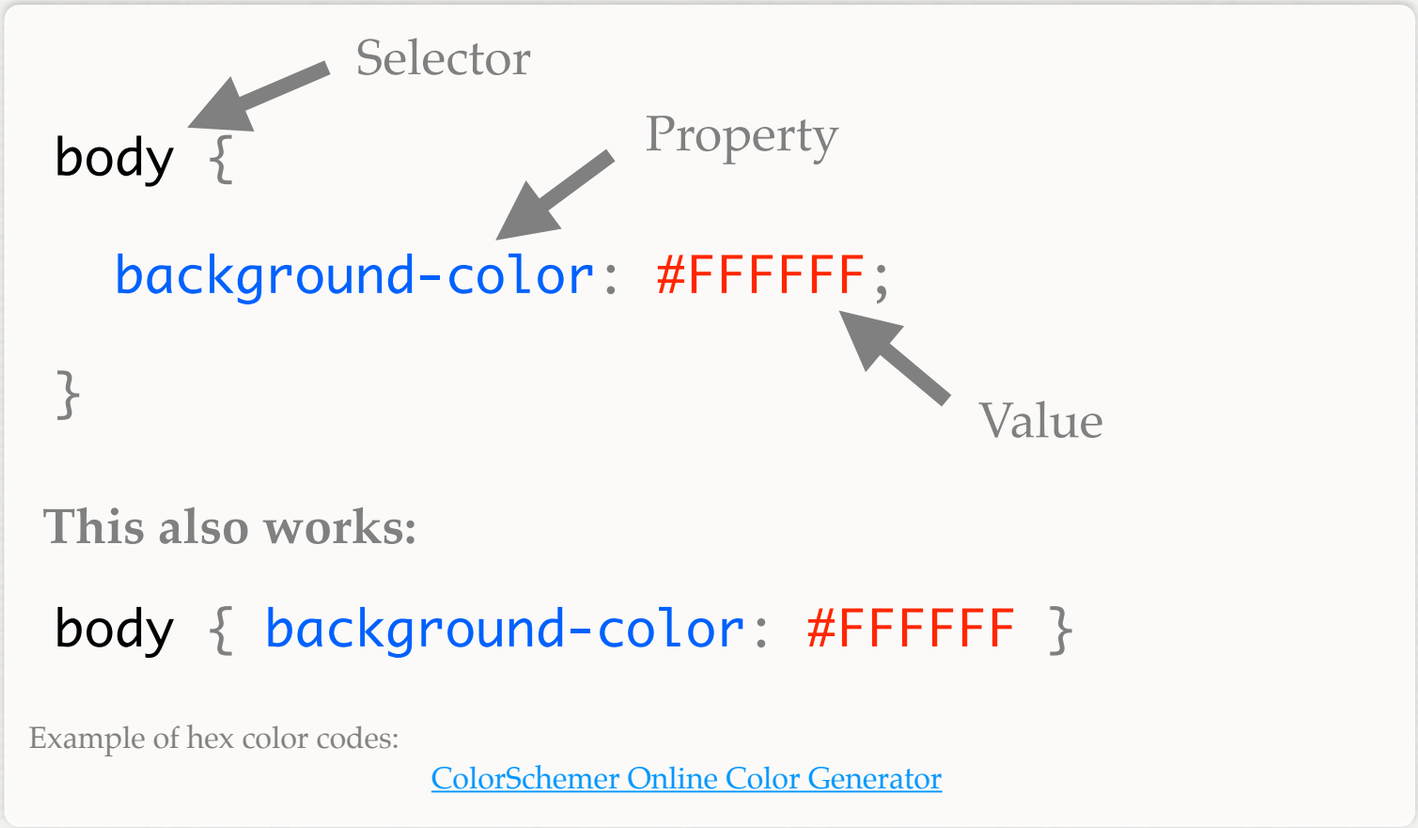
2. Indicate which aspect(s) of the element you want to modify

   ➡ Assign **values** to **properties**

CSS is really very simple. There are only two steps in the process towards adding style to the page.

First, you select the element(s) that you intend to modify, and second, you designate what it is about the elements that you want to change. Intuitively, you use selectors to select the elements. The “things” that you change about the elements are their properties, and you do this by assigning a value to that property.

CSS Anatomy



This also works:

```
body { background-color: #FFFFFF }
```

Example of hex color codes:

ColorSchemer Online Color Generator

This explanation illustrates the code that allows us to do our two necessary tasks--to select an element and update the properties of that element. As you can see, the syntax of CSS is significantly different from HTML--and it should be, it has a different purpose--but it is still very simply in form.

Our selector, along with the property and value to be assigned, are marked above. The selector corresponds to a real HTML element in our document, and the curly braces enable us to begin typing out properties and values. Properties tend to include hyphenated words, and the colon is used to separate properties from values. Values can have different types of values. The property background-color requires a color that can either be a word (blue, green) or a hex (6-digit) color code (the hex color code must be preceded by the # symbol).

CSS is very flexible, and ignores whitespace as demonstrated by the second, equally valid line of code. In all cases, the curly braces and colon are both required, however, the semi-colon is required whenever more than one property will be listed. To list more than one property in the second example we would have to add a semicolon after the hex color code.

C
S tyle
S heet

Stylesheet: A Sheet of Styles

- Place each style block together to make a stylesheet!

You just learned how to make a single style block. Part of CSS's meaning is the listing of multiple style blocks, or rules. Stylesheet: a sheet (or listing) of (many) styles.

Selectors

- Select an **element**

  ➡ body { } p { } strong { } div { }

- Select a **nested element**

  ➡ p strong { } div p { } ul li ul li span { }

- Select **multiple elements**

  ➡ p, div { } strong, em { } p, span { }

These following examples indicate the different ways to indicate selectors. As we have seen, to select an element, you merely write the tag's name.

To select a nested element, that is, an element inside of another, you must simply list the parent element and then element may occur inside the element, separated by a space. Please note: the rule will apply for any element of the same type that occurs at any level within the parent element. For example: div strong { } will select both strong elements as they both occur within the element.

```
<div>
  <strong></strong>
  <p><strong></strong></p>
</div>
```

To select multiple elements, separate the individual selectors by a comma. You can also combine this with the nested element selector, for example: div p, div strong { }

More Selectors

- Select an element with a **class**
  - ➡ p.indent { } span.blue { } div.page { }
- Select an element with an **id**
  - ➡ div#home { } div p#tools { }
- Select an element with an **id** *and* a **class**
  - ➡ ul.square#top { } div.page table.math#pset { }

XHTML defines two properties for nearly all elements, class and id, which can be used to more precisely select elements on a page. This is useful as many pages include more than one or <p> element, and these each may need to be separately styled.

To give an element a class, use the class attribute on the element in your HTML document:

You can now access the element as displayed using a period. Placing a period before a phrase in a CSS selector causes it to refer to a class-- it may be used alone or in conjunction with the element's tag name.

Just as periods specify classes, the # symbol specifies the selection of an id, which is set for an element in the HTML source using the id attribute:
<p id="home">

Please note: XHTML requires that no more than one element in any valid document have the same id. Because of this, in a given XHTML document, the id corresponds to a specific element within the document.

Each type of selector can be concatenated to one another so that we can get chains that combine the element's tag name, its class, and its id.

About Properties

- Properties can be specific to a group of elements
  - ex: `border-spacing` is useless for elements
- *Some* property values can be inherited by children
  - ex: `font-size` set on <body> will be inherited by a child element
- Properties can have default values
  - ex: `background-color`'s default value is `transparent`

Some properties simply do not make sense for certain elements. The property border-spacing specifies the space between cells in a table. This property would be useless for element which does not even have cells. Therefore, browsers effectively ignore this property for non-tables.

Certain property values will be inherited from the parent element, and font-size is a prime example of this idea. Any element inside body, unless it has its own default values, will take on the font-size specified on the body. This is why it is a good first step to specify such properties on the body element in your CSS file.

As stated, many properties have a default value. background-color is not a property that is typically inherited from the parent element. Rather, it was decided that the default property of elements would take on the "transparent" background-color which effectively means that you see the background color of the object behind it.

How do you know?

... Or find out?

- Luckily, the W3C produces a specification for CSS, too
  - Lists all properties, defaults, possible values
  - Also suggests how property values affect rendering
- Many commercial sites also provide similar references
  - ex: SitePoint, w3schools (both have HTML & CSS)

Like HTML, XHTML, XML, MathML, and several other technologies, the W3C, the World-Wide-Web Consortium (actually hosted in the US by MIT-- it's in the Stata Center) produces a specification for how CSS should be formed by developed and rendered by web browsers. This is great because it lists all of the properties, default values, and possible values in CSS.

However, the specification is rather abstract and will include phrases like "viewport" and "user-agent" that do not mean much to most developers. Many commercial sites, such as SitePoint (linked on our site) and w3schools, provide rather complete references for HTML and CSS and much more (ex: Javascript and ASP) along with tools for previewing the effects of certain properties and their values.

Key Properties

- background-color
- background-image
- color
- width
- height
- font-family
- font-size
- font-weight
- text-decoration
- text-align

These are a number of key properties that will get you started with CSS. For complete information on the properties see one of the commercial sites.

Values & Units

- Values are typically keywords
  - ex: colors: `red, blue`
  - ex: text alignment: `left, right, center`
- Values, especially for layout
  - Pixels: `15px`
  - Points: `12pt`
  - Percentages- relative to size of parent: `50%`

Property values are typically keywords such as those shown, corresponding to the “color” and “text-align” properties.

However, these values may also be numbers such as those that would be used in layout--to specify the spacing between elements or the font-size for text. It is generally recommended to begin working with fixed values such as pixels and points, and then move on to percentages, if at all. Percentages are useful in some circumstances, but can lead to an inconsistent user experience on multiple platforms (differences between your web browser and a mobile device).

An Example

Consider the **boldface** font in the following examples:

Chunky bacon is delicious.

- Use the tag

Today I will go outside.
Monday I will run 2 miles.

- use `font-weight: bold;`

Returning to our example with the two uses of boldface font, we now have the tools to write generate the document style without altering much of the document itself. For example, to fix the second example, we would enclose “Today” and “Monday” in tags and perhaps give them a class of “date”. Finally, in the CSS we would select all span elements with the class “date” and set their font-weight property to bold.

Firebug: Your New Best Friend

Get Firebug

- Firefox extension for “inspecting” page elements
  - Useful tool when “something doesn’t look right”
- Provides interactive view of HTML source
- Allows for real-time changes in HTML and CSS

➡ The Web is inherently Open Source!