Cascading Style Sheets (CSS)

Objectives

- How to include different types of Cascading Style Sheets to format HTML documents
- How to use CSS selectors to choose different parts of HTML to be styled
- Get an idea of the types of properties and values that can be used in CSS
- Understand the Box Model of HTML elements
- Use and elements to define sections of HTML to be formatted
- Understand the cascading process of CSS

CSS Basics

Introduction to CSS

Cascading Style Sheets (CSSs) provide the means to **control** and **change** presentation of HTML documents

CSS is not technically HTML, but can be embedded in HTML documents

The CSS1 specification was developed in 1996

CSS2 was released in 1998

CSS3 is the newest version

Introduction to CSS

A ***style sheet*** is a syntactic mechanism for specifying style information

Style sheets allow you to impose a standard style on a whole document, or even a whole collection of documents

Style is specified for a tag by the values of its properties

Levels of Style Sheets

1. **Inline style** - specified for a specific occurrence of a tag and apply only to that tag
   - This is fine-grain style, which defeats the purpose of style sheets - uniform style

2. **Document-level style sheets** - apply to the whole document in which they appear

3. **External style sheets** - can be applied to any number of documents

Levels of Style Sheets (cont.)

Inline style sheets appear in the tag itself

Document-level style sheets appear in the head of the document

External style sheets are in separate files, potentially on any server on the Internet

- Written as text files with the MIME type `text/css`
- External style sheets can be validated at w3c [link]

Links:
http://jigsaw.w3.org/css-validator/

Levels of Style Sheets (cont.)

When more than one style sheet applies to a specific tag in a document, the lowest level style sheet has precedence

In a sense, the browser searches for a style property spec, starting with inline, until it finds one (or there isn't one)

If no style sheet is provided, the browser default property values are used

Style Specification Formats

Inline

- Style sheet appears as the **value** of the **style attribute**
- General form:

```
style = "property_1: value_1;
         property_2: value_2;
         …
         property_n: value_n"
```

EXAMPLE:

```
style = "color: red; text-align: center"
```

Style Specification Formats (cont.)

Document-level

- Style sheet appears as a list of rules that are the **content of a `<style>` tag**
- The `<style>` tag must include the **type attribute**, set to `"text/css"`
- Comments in the rule list must have a different form
  -> use C style comments /*…*/

Style Specification Formats (cont.)

Document-level (cont.)

- General form:

```
<style type = "text/css">
    rule list
</style>
```

- Form of the **rules**:
  ***selector* { *list of property/values* }**
- Each **property/value pair** has the form: ***property: value***
- Pairs are separated by semicolons, just as in the value of a `<style>` tag

Including external style sheets

External style sheets

- Style sheet appears as a list of rules that are contained in a **separate .css file**

- A ***<link> tag*** is used to specify that the browser is to fetch and use an external style sheet file, e.g.:

```
<link rel = "stylesheet"  type = "text/css"
 href = "http://www.wherever.org/termpaper.css">
</link>
```

- An alternative way to reference an external style sheet:

```
@import url(filename);
```

(appears at the beginning of the content of a `style` element)

CSS Selectors

CSS Selectors

"***Selectors*** are patterns used to select the element(s) you want to style." (w3schools)

There are different selector types:

- Simple selectors
- Class selectors
- Generic selectors
- `id` selectors
- Universal selectors
- Pseudo classes

Simple Selector Forms

The ***simple selector*** is a tag name or a list of tag names, separated by commas

Examples:

- `h1, h3`
- `p`

Contextual selectors (nested elements)

- `ol ol li`

Class Selectors

Class selectors are used to allow different occurrences of the same tag to use different style specifications

A style class has a name, which is attached to a tag name, e.g.:

```
p.narrow {property/value list}
p.wide {property/value list}
```

The class you want on a particular occurrence of a tag is specified with the `class` attribute of the tag, e.g.:

```
<p class = "narrow">
  ...
</p>
  ...
<p class = "wide">
  ...
</p>
```

Generic Selectors

A ***generic class*** can be defined if you want a style to apply to more than one kind of tag

A generic class must be named, and the name must begin with a period, e.g.:

```
.sale { … }
```

Use it as if it were a normal style class:

```
<h1 class = "sale"> Weekend Sale </h1>
  ...
<p class = "sale"> … </p>
```

`id` Selectors

An **`id`** ***selector*** allows the application of a style to one specific element

General form:

#specific-id { property-value list }

Example:

```
#section14 {...}
```

Universal Selectors

Universal selectors are denoted by the asterisk

They apply styling to all elements in the document

Example:

```
* {color: red;}
```

Not often useful

Pseudo Classes

Pseudo classes are styles that apply when something happens, rather than because the target element simply exists

Examples:

- `hover` classes apply when the mouse cursor is over the element
- `focus` classes apply when an element has focus

Examples

Selector examples can be found on the w3schools site:
http://www.w3schools.com/cssref/trysel.asp

Style Properties

Properties

There are 60 different properties in 7 categories:

- Fonts
- Lists
- Alignment of text
- Margins
- Colors
- Backgrounds
- Borders

Complete list of property values :
http://www.w3schools.com/cssref/default.asp

Property values are inherited by all nested tags, unless overridden

Property Values

Property values can take different forms:

- keywords
- percentage
- URL
- numbers with units

Keywords - `left`, `small`, …
(not case sensitive)

Percentage - number followed immediately by a percent sign

URL values - `url` (protocol **:** / / server / pathname)

Property Values

For length properties: **numbers**, maybe with decimal points, followed by **units**

Units:

- `px` – pixels
- `in` – inches
- `cm` – centimeters
- `mm` – millimeters
- `pt` – points
- `pc` - picas (12 points)
- `em` - height of the letter 'm'
- `ex` - height of the letter 'x'

No space is allowed between the number and the unit specification. E.g., `1.5 in` is illegal!

Property Values

Color values can be specified in 3 different ways:

- Color name
- `rgb(n1, n2, n3)`
  - Numbers can be decimal or percentages
- Hex form: #XXXXXX

Font Properties

Font Properties

Some common font properties:

`font-family`

- Value is a list of font names - browser uses the first in the list it has

```
font-family: Arial, Helvetica, Futura
```

- Generic fonts: `serif`, `sans-serif`, `cursive`, `fantasy`, and `monospace` (defined in CSS) - browser has a specific font for each
- If a font name has more than one word, it should be single-quoted

`font-size`

- Possible values: a length number or a name, such as smaller, xx-large, etc.

`font-variant`

- Default is `normal`, but can be set to `small-caps`

Font Properties (cont.)

`font-style`

- `italic`, `oblique` (useless), `normal`

`font-weight`

- degrees of boldness
- `bolder`, `lighter`, `bold`, `normal`
- Could specify as a multiple of 0f `100` (`100 – 900`)

`font`

- For specifying a list of font properties

```
font: bolder 14pt Arial Helvetica
```

- Order must be: **style, weight, size, name(s)**

Font Properties (cont.)

`text-decoration`

- `line-through`, `overline`, `underline`, `none`

`letter-spacing`

- value is any length property value – ex: `4px`

Examples

Let's see some examples:

fonts.html [link]
fonts2.html [link] with styles.css [link]
decoration.html [link]
text_space.html [link]

Links:
http://www.epogue.info/cpsc-24700/Presentations/examples/w8code3/fonts.html
http://www.epogue.info/cpsc-24700/Presentations/examples/w8code3/fonts2.html
http://www.epogue.info/cpsc-24700/Presentations/examples/w8code3/styles.css
http://www.epogue.info/cpsc-24700/Presentations/examples/w8code3/decoration.html
http://www.epogue.info/cpsc-24700/Presentations/examples/w8code3/text_space.html

List Properties and Color Values

List properties

`list-style-type`

Unordered lists

- Bullet can be a `disc` (default), a `square`, or a `circle`
- Set it on either the `<ul>` or `<li>` tag
- On `<ul>` it applies to all items in the list
- On `<li>`, `list-style-type` applies to just that item
- Could use an image for the bullets:

```
<li style = "list-style-image: url(bird.jpg)">
```

List Properties (cont.)

`list-style-type`

Ordered lists – `list-style-type` can be used to change the sequence values

Property Value	First Four
decimal	1, 2, 3, 4
upper-alpha	A, B, C, D
lower-alpha	a, b, c, d
upper-roman	I, II, III, IV
lower-roman	i, ii, iii, iv

CSS2 has more, like lower-greek, and hebrew, and armenian

Examples

sequence_types.html [link]

Links:
http://www.epogue.info/cpsc-24700/Presentations/examples/w8code3/sequence_types.html

Colors

Color is a problem for the Web for two reasons:

- Old monitors vary widely
- Old browsers vary widely

The `color` property specifies the foreground color of elements

The `background-color` property specifies the background color of elements

Colors (cont.)

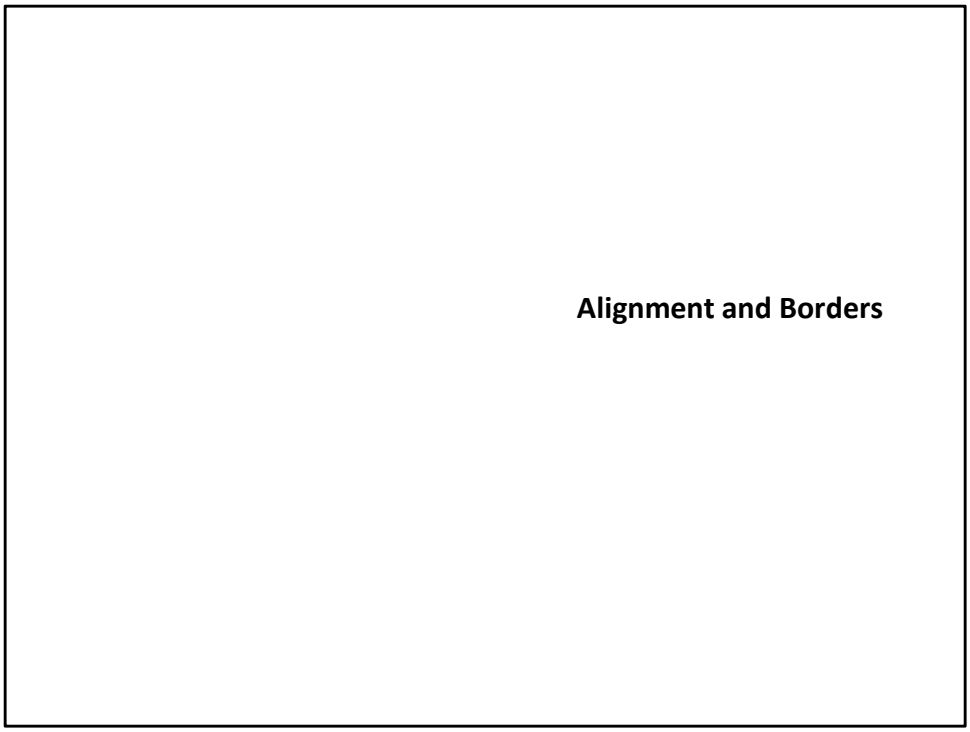
There are three color collections:

1. The set of 17 colors that are guaranteed to be displayable by all graphical browsers on all color monitors:

```
black 000000 purple 800080 navy   000080
olive 808000 blue   0000FF gray   808080
green 008000 silver C0C0C0 teal   008080
red   FF0000 lime   00FF00 fuchia FF00FF
aqua  00FFFF yellow FFFF00 maroon 800000
white FFFFFF orange FFA500
```

2. The Web Palette
   - 216 colors
   - Use hex color values of `00`, `33`, `66`, `99`, `CC`, and `FF`
3. Any one of 16 million different colors
   - Use 6 hexadecimal digit color values

http://www.w3schools.com/html/html_colors.asp


Alignment and Borders

Alignment of Text

There are several ways of aligning text and other elements:

The `text-indent` property allows indentation

- Takes either a length or a % value

The `text-align` property has the possible values:

- `left` (the default)
- `center`
- `right`
- `justify`

Alignment of Text

Sometimes we want text to flow around another element

We can use the `float` property

- The `float` property has the possible values: `left, right, none` (the default)

Example: we want an element to be on the right and text flowing on its left

- We use the default `text-align` value (`left`) for the text and
- the `right` value for `float` on the element we want on the right

Borders

Every element has a `border-style` property

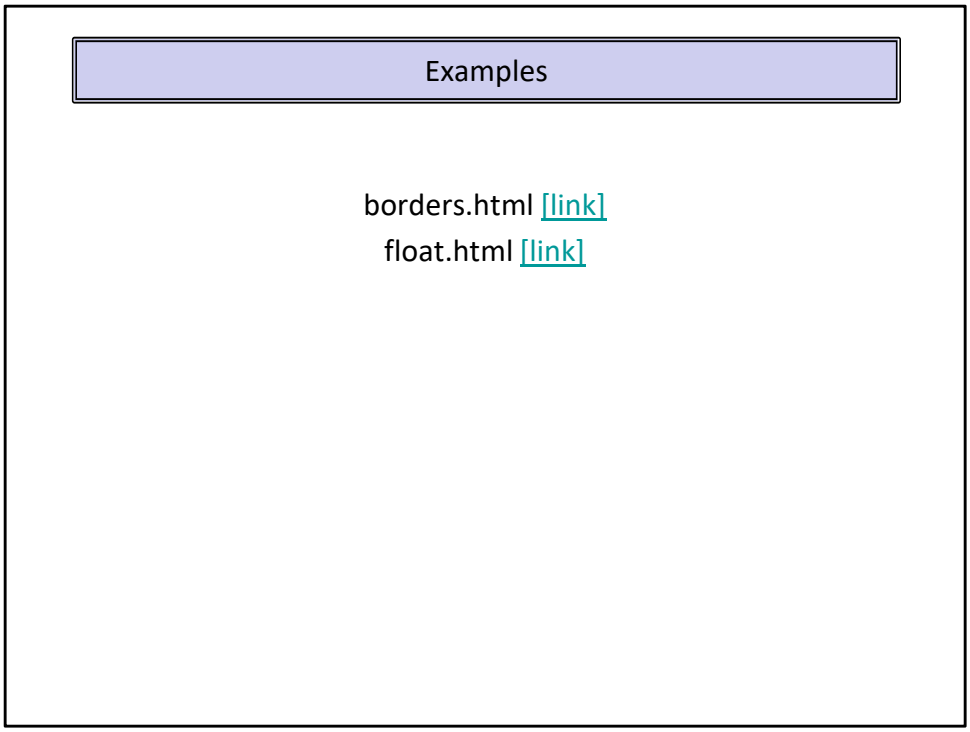
- Controls whether the element has a border and if so, the style of the border

`border-style` values: `none`, `dotted`, `dashed`, and `double`

You can control the border properties with:

`border-width` – `thin`, `medium` (default), `thick`, or a length value in pixels

`border-color` – any color



Links:
http://www.epogue.info/cpsc-24700/Presentations/examples/w8code3/borders.html
http://www.epogue.info/cpsc-24700/Presentations/examples/w8code3/float.html

The Box Model and Backgrounds

The Box Model

Margin – the space between the border of an element and its neighbor element

- The margins around an element can be set with `margin-left`, etc. - just assign them a length value

Padding – the distance between the content of an element and its border

- Controlled by `padding`, `padding-left`, etc.

Background Images

You can put a **background image** on your page using the `background-image` property

Repetition can be controlled using the `background-repeat` property

- Possible values: `repeat` (default), `no-repeat`, `repeat-x`, or `repeat-y`

You can set the position of the background using the `background-position` property

- Possible values: `top`, `center`, `bottom`, `left`, or `right`

Examples

back_image.html [link]
marpads.html [link]

Links:
http://www.epogue.info/cpsc-24700/Presentations/examples/w8code3/back_image.html
http://www.epogue.info/cpsc-24700/Presentations/examples/w8code3/marpads.html

**Structuring Documents with and **

The tag

One problem with the font properties is that they apply to whole elements, which are often too large

Solution: a new tag to define an element in the content of a larger element - ****

The default meaning of `<span>` is to leave the content as it is

The `<span>` tag can be nested and have `id` and `class` attributes

The tag

Another tag that is useful for style specifications: **`<div>`**

Used to create document sections (or **divisions**) for which style can be specified

Conflict Resolution and the Cascade

Conflict Resolution

A conflict occurs when there are two or more values for the same property on the same element

Sources of conflict:

- Conflicting values between levels of style sheets
- Within one style sheet
- Inheritance can cause conflicts
- Property values can come from style sheets written by the document author, the browser user, and the browser defaults

Conflict Resolution (cont.)

Resolution mechanisms:

- Precedence rules for the different levels of style sheets
- Source of the property value
- The specificity of the selector used to set the property value
- Property value specifications can be marked to indicate their weight (importance)

Weight is assigned to a property value by attaching `!important` to the value

Cascade

Conflict resolution is a multistage process, called the ***cascade***:

- First, all of the style specs from the different levels of style sheets are gathered

- All available specs, from all sources, are then sorted by origin and weight, using the following rules, which are given in **precedence order**:
  1. Important declarations with user origin
  2. Important declarations with author origin
  3. Normal declarations with author origin
  4. Normal declarations with user origin
  5. Any declarations with browser (or other user agent) origin

Cascade

- If any conflicts remain, sort them by **specificity**:
  1. `id` selectors
  2. Class and pseudo-class selectors
  3. Contextual selectors
  4. Universal selectors

If there are still conflicts, resolve them by precedence to the most recently seen specification

Summary

- Cascading Styles Sheets (CSSs) provide a means to control and change the presentation of HTML documents
- CSSs can be inline, document-level, or external
- CSS selectors are patterns used to select the HTML elements you want to style
- There are over 60 different properties in various categories that can be used for styling
- The Box Model places padding, a border, and a margin around the content of HTML elements
- The and tags can be used to specify parts of HTML to be styled
- Styling of CSS is a multistage cascade process in which the styling is applied in order of the level of style sheet, the importance and origin of the declaration, and the specificity of the selector