Cascading Style Sheets (CSS)

Overview

- Levels of style sheets
- Style specification formats
- Selector forms
- Property value forms
- Examples of properties – font, list, color, text alignment, background images

CSS

- CSS provides the means to control and change presentation of HTML documents
- Style sheets allow you to impose a standard style on a whole document, or even a whole collection of documents
- CSS1 specification – 1996
- CSS2 specification - 1998
- CSS level 2 revision 1 (“CSS 2.1”) is a Candidate W3C Recommendation
- CSS3 is under development

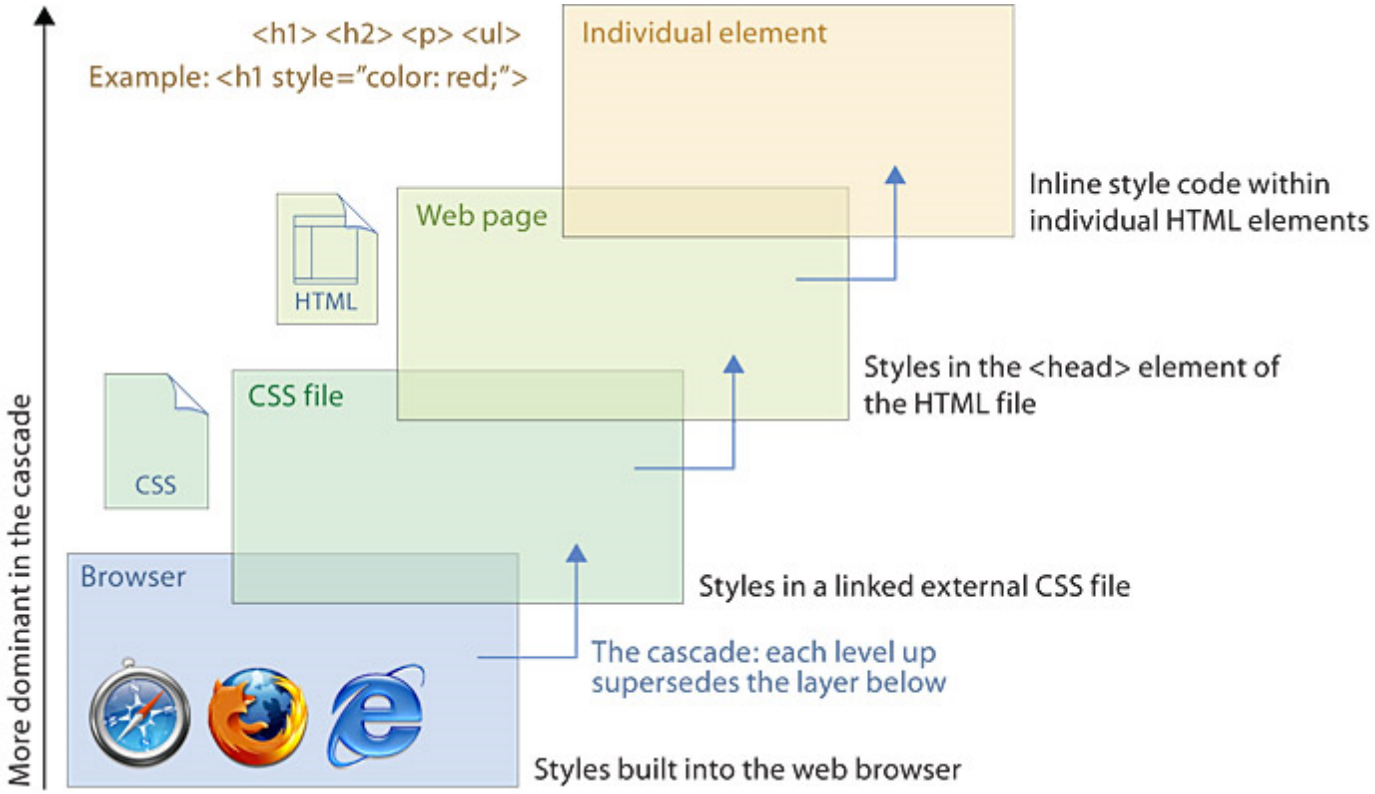
Levels of Style Sheets

There are three levels of style sheets

1. **Inline** - 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

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

CSS cascade hierarchy



Levels of Style Sheets

- 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`**
  - A **`<link>`** tag is used to specify that the browser is to fetch and use an external style sheet file

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

Optional Attributes

“rel”=Specifies the relationship between the current document and the linked document

“target”=Specifies where the linked document is to be loaded

“type”=Specifies the MIME type of the linked document

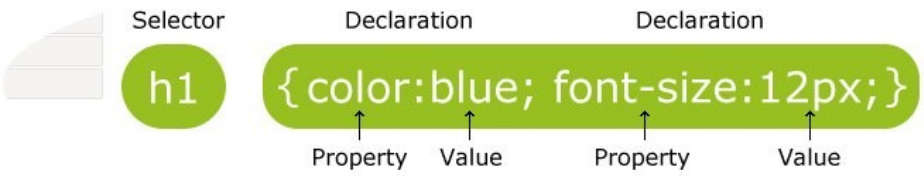
“media”=Specifies on what device the linked document will be displayed

“charset”=Specifies the character encoding of the linked document

value(“char_encoding”)

href=” Specifies the location of the linked document”(URL)

...s main parts: a selector, and one or more declarations:



Inline Style Specification

- 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;"
```

Document Style Specification

- 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`**"

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

- Form of the rules:

```
selector {property_1:value_1;
  property_2:value_2; …;
  property_n:value_n;}
```

External Style Sheet Specification

- Form is a list of style rules
  - Like the content of a `<style>` tag for document-level style sheets

Types of Selectors forms

Simple Selector Forms

Class Selectors

Generic Selectors

Id Selectors

Universal Selectors

Pseudo Classes

Simple Selector

- The selector is a tag name or a list of tag names
- Examples:

```
h1 {font_size: 24pt;}
h2, h3 {font_size: 20pt;}
```

- Contextual selectors
  - Apply style only to elements in specified position in body of document
  - List element hierarchy

```
body p b {font_size: 30pt}
```

CLASS

A class is best described as a self-contained set of formatting instructions which can be applied to one or more elements in markup

Types of class

A Generic class- class which can be applied to any element in the markup.

A Specific class-Which can only be applied to a specific element in the markup.

A Sub-class-allowing you a very fine level of control while keeping structure simple -

Generic classes

This is the simplest class to start off with because it is not tied to any particular elemen

```
.LargeBlueText {
    color: blue;
    background-color: transparent;
    font-size: 12pt;
  }
```

Specific classes

This class is constructed the same way as a generic class with the obvious exception that when we define it we associate it with an elment

```
span.LargeBlueText {
      color: blue;
      background-color: transparent;
      font-size: 12pt;
  }
```

Classes and Sub-Classes

These are the most complex type of class available, they are best thought of as optional instructions which apply only within the parent class

```
.GreyText {
    color: gray;
    background-color: transparent;
}
.GreyText code {
    color: white;
    font-weight: bold;
}
.GreyText img.WhiteBorder {
    border-width: 5px;
    border-color: white;
    border-style: solid;
}
```

Class Selector

- Used to allow different occurrences of the same tag to have different style specifications
- A style class has a name, which is attached to a tag name

```
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

```
<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

```
.really-big { … }
```

- Use in body of doc like normal style class

```
<h1 class = "really-big"> … </h1>
 ...
<p class = "really-big"> … </p>
```

id Selectors

- An id selector allow the application of a style to one specific element
- General form:

```
#specific-id {property-value list}
```

e.g. `#section3 {font-size: 20}`

- In XHMTL doc:

```
<h2 id = “section3”>
  3. Properties for sale
</h2>
```

Universal Selectors

An universal selector is denoted by * ,applies to all elements in the document

- General form:

```
*{Colour:red;}
```

Pseudo Classes

- Pseudo classes are styles that apply when something happens, rather than because the target element simply exists
- Names begin with colons
- hover class applies when the mouse cursor is over the element
- focus class applies when an element has focus

Properties

There are different properties in 12 categories:

- Background
- Border and outline
- Dimension
- Font
- Generated content
- List
- Margin
- Padding
- Positioning
- Print
- Table
- Text

Property Value Forms

- **Keywords** - `left`, `small`, …
- **Length** - numbers, maybe with decimal points
  - Units:
    - px – pixels
    - in – inches
    - cm – centimeters

mm – 1em is equal to the current font size. 2em means 2 times the size of the current font. E.g., if an element is displayed with a font of 12 pt, then '2em' is 24 pt. The 'em' is a very useful unit in CSS, since it can adapt automatically to the font that the reader uses

    - pt – points
    - pc - picas (12 points)
    - em - height of the letter ‘m’
    - ex-height - height of the letter ‘x’
  - No space is allowed between the number and

Property Value Forms

- **Percentage** - just a number followed immediately by a percent sign
- **URL values**
  - url(protocol://server/pathname)
- **Colors**
  - Color name, e.g. white
  - Hex form: #XXXXXX, e.g. #FFFFFF
  - rgb(n1, n2, n3), e.g. rgb(255, 255, 255)
    - Numbers can be decimal (0-255) or percentages
- Property values are inherited by all nested tags, unless overridden

Font Properties

- **`font-family`**
  - Value is a list of font names - browser uses the first in the list it has
- **`font-size`**
  - Possible values: a length number or a name, such as smaller, xx-large, etc.
- **`font-style`**
  - italic, oblique (useless), normal
- **`font-weight`** - degrees of boldness
  - bolder, lighter, bold, normal
- **`font`** - for specifying a list of font properties
  - font: bolder 14pt Arial Helvetica
  - Order must be: style, weight, size, font name(s)

List Properties
`list-style-type`

- On unordered lists `list-style-type` can be used to specify the shape of the bullets
  - disc (default), square, or circle
  - Set it on either the `<ul>` or `<li>` tag

```
<h3> Fruit </h3>
 <ul>
     <li style = "list-style-type: disc"> Apple </li>
     <li style = "list-style-type: square"> Orange </li>
     <li style = "list-style-type: circle"> Pear </li>
 </ul>
```

- On ordered lists `list-style-type` can be used to change the sequence values

http://www.cs.nott.ac.uk/~bnk/WPS/sequence_types.html

Text and Background Colour

- The **`color`** property specifies colour of text

```
<style type = “text/css”>
    th.red {color: red}
    th.orange {color: orange}
</style>
```

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

http://www.cs.nott.ac.uk/~bnk/WPS/back_color.html

Background Images

- The **`background-image`** property

- **`background-repeat`** property
  - Possible values: `repeat` (default), `no-repeat`, `repeat-x`, or `repeat-y`
- **`background-position`** property
  - Possible values: `top`, `center`, `bottom`, `left`, or `right`

Text Alignment

- 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`, or `justify`
- Sometimes we want text to flow around another element - the **`float`** property
  - values of `left`, `right`, and `none` (the default)

http://www.cs.nott.ac.uk/~bnk/WPS/float.html

The and tags

- 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 is to leave the content as it is

```
<style type = "text/css">
  .bigred {font-size: 24pt; font-family: Ariel; color: red}
</style>
…
 <p> Now is the <span class = "bigred"> best time </span>
ever! </p>
```

The `<span>` and `<div>` tags

- The `<span>` tag is similar to other HTML tags, they can be nested and they have `id` and `class` attributes
- Another tag that is useful for style specifications: **`<div>`**
  - Used to create document sections (or divisions) for which style can be specified
  - e.g., a section of five paragraphs for which you want some particular style

BOX MODEL


margin-top
border-top
padding-top
width
background-color
margin-left
border-left
padding-left
height
padding-right
border-right
margin-right
padding-bottom
border-bottom
margin-bottom


Margin
Border
Padding
Content

Margin - Clears an area around the border. The margin does not have a background color, it is completely transparent

Border - A border that goes around the padding and content. The border is affected by the background color of the box

Padding - Clears an area around the content. The padding is affected by the background color of the box

Content - The content of the box, where text and images appear

Summary

- Motivation
- Levels of style sheets
  - Inline, document, external
- Style specification formats
- Selector forms
  - Simple, class, generic, id, pseudo classes
- Property value forms
- Examples of properties
  - font, list, color, text alignment, background images
- `<span>` and `<div>` tags