Lecture 2: CSS
Cascading Style Sheets

Wendy Liu
CSC309F – Fall 2007

Outline

- Motivation
- Introducing Cascaded Style Sheets (CSS)
- Selector Forms
- The Box Model
- Revisiting and Tags
- Conflict Resolution

Motivation

Content vs. Presentation

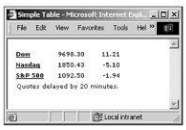
- Original intent for HTML
  - Author: specify document content and structure
  - Browser: present the content and structure using default values
    - Formatting (layout) is based on its environment: desktop, laptop, palmtop, web phone, etc.
- In reality
  - Authors want much more control over layout and presentation
    - Colour and font
    - Alignment and spacing of text
  - Ad hoc tags and attributes are added by browser vendors
  - Browser-specific extensions are a nightmare for portability
  - HTML "degenerated" into a markup language for format
    - Against its designed purpose

Problems

- Reducing browser performance
  - Redundant formatting information swells file size
- Lowering programmer productivity
  - Non-standard way to add features
    - Too many varieties
- Inflexible
  - Difficult to make changes or redesign
  - Difficult to support user customizations
  - Difficult to support heterogeneous devices/interfaces
- Poor accessibility
  - No support for multi-modal interfaces

An Extreme Case

- ~2,400 HTML characters to describe 60 characters of content



Style Sheets

- Separate structure from presentation
- Attach style to structured documents
  - Fonts, colours, spacing, …
- Remove the requirement for further formatting tags
- Advantages
  - Capability of imposing consistency on style
  - Precise control over presentation
  - Simplify site maintenance
  - Faster downloads

7

Introduction to Cascading Style Sheets (CSS)

http://www.w3.org/TR/CSS1.html
http://www.w3.org/TR/CSS2/

8

Cascading Style Sheets (CSS)

- Three different levels of styles, thus *cascading*
  - inline (highest precedence, lowest level)
  - document level
  - external
- Lower level can override higher level style sheets
- May impose a standard style on a whole document, or a collection of documents
- Applies generically to all forms of XML
  - <?xml-stylesheet type="text/css" href="storefront.css"?>

9

Inline Style Sheets

- Being specified for a particular element
- Finest-grain style
- Disadvantages
  - Defeating the purpose of having style sheets – uniformity
  - Embedding style information in XHTML document using a distinct language
- Inline style sheets appear as attribute of a tag

  style = "*property_1: value_1;*
  property_2: value_2;
  …
  *property_n: value_n;***"**
- Deprecated in XHTML 1.1

10

Document-level Style Sheets

- Apply to the entire body of the document
- Defined in the head of the XHMTL document
  - <style type = "text/css">
    <!--
    rule list
    -->
    </style>
  - @import url(*filename*)
    - Only at the beginning of a style element
    - *filename* is not quoted
    - Can contain markup
- Type info is necessary
  - E.g. style sheets can be provided to JavaScript

11

Document-level: Rule List

- Rule list must be placed in a comment
  - It is not XHTML
- A style rule in the rule list looks like

  selector {*property_1: value_1;*
  property_2: value_2;
  …
  property_n: value_n;}
- Comments in the rule list
  - */*…*/*
- Selector represents tag(s) affected by the rule
- Multiple values of a property are separated by spaces or commas
  - font: bold 14pt 'Times New Roman' Courier;
  - font-family: Arial, Helvetica, 'Times New Roman';
- No quotes

12

External Style Sheets

- Can be applied to any document
- Stored in separate files, potentially on any server on the Internet
- Linking to XHMTL
  - <link rel="stylesheet" type="text/css" href="resume.css">
- Consisting a list of style rules

13

Selector Forms

http://www.w3.org/TR/CSS1.html
http://www.w3.org/TR/CSS2/selector.html

14

Simple Selector Forms

- Simplest selector form
  - Single element name
  - Multiple single-elements are separated by comma
    - h1 {font-size: 24pt;}
    - h2, h3 {font-size: 18pt;}

15

Contextual Selectors

- Match a search pattern on a stack of open elements
- Descendant selectors (CSS1)
  - Ancestors, not just parents
  - Separated by space
    - body b em {font-size: 14pt;}
    - ul ul {list-style-type : none;}
- More choices are available in CSS2

16

Class Selectors

- To allow different occurrences of the same tag to use different styles
- Use class to conditionally apply style
  - in XHML

    <tr class="header">…</tr>
    ...
    ...
  - in CSS

    tr.header { color: blue;}
    p.minor {font-size: 10pt;}
    p.major {font-size: 14pt;}

17

Generic Class Selectors

- Apply a style to more than one kind of tags
- Use the name of the generic class
  - Must begin with a period
  - in CSS

    .really-big { … }
  - in XHML

    <h1 class = "really-big"> … </h1>
     …

18

id Selectors

- Apply to the element with the specific id
  - *#specific-id* {*property-value list*}
  - XHML

    <li id="bookname">Programming the WWW
  - CSS

    #bookname { color: blue;}
- Ids are unique in XHTML

19

Universal Selector

- Applies to all elements in the document
  - * {color: green;}

20

Pseudo-elements and Pseudo-classes

- To permit formatting based on information that lies outside the document tree
- Name begins with a colon
- Pseudo-elements
  - Create abstractions about the document tree beyond those specified by the document language
- Pseudo-classes
  - Classify elements on characteristics other than their name, attributes or content
    - Those cannot be deduced from the document tree
  - Pseudo-classes may be dynamic
    - I.e. an element may acquire or lose a pseudo-class while a user interacts with the document
    - The exception is ':first-child', which can be deduced from the document tree

21

Typographical Pseudo-Elements

- First-line formatting

```
p:first-line { font-weight:bold }
```

- First-letter formatting

```
p:first-letter { font-size: 200%; float:
left }
```

"This", quote I, is a paragraph that could be split anywhere in this sentence.

22

Pseudo Classes

- Dynamic pseudo classes
  - Apply when something happens
    - Rather than simply existing
  - **`:hover`** class
    - Apply when the mouse cursor is over the element

      input:hover {color: red;}
  - **`:focus`** class
    - Apply when an element has focus

      input:focus {color: green;}

      <input type = "text" />
- Link pseudo-classes

```
a:link {color:green}  /* unvisited link */
:visited {color:purple} /* visited link */
```

  - No effect on elements other than <a>, so ***a*** may be omitted

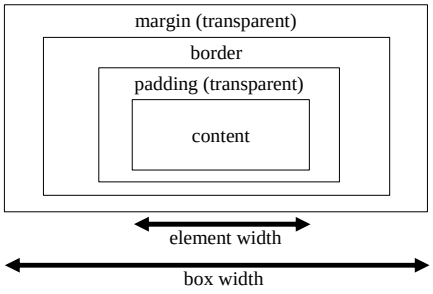
23

The Box Model

For Formatting

24

The Box Model



25

Box Properties

- Borders
  - border-style
    - solid, dotted, dashed, double, none (default)
  - border-width
    - thin, medium (default), thick, *number in pixel*
  - border-color
- Padding
  - padding
    - Specify length
- Margin
  - margin
    - Specify length
- May set for individual sides
  - E.g., border-top-style, border-bottom-width, padding-left, margin-right
- The background of an element extends into the padding, but not into the margin

26

and

- Use these tags to group text and form fragments or sections for specific styling needs
  - Useful when there is no other convenient means
-
  - Inline, fragment
-
  - Block-level, section

27

Conflict Resolution

The Cascade

Determining the property values for each element

28

Sources of Conflict

- Style-sheet level
  - Resolve by precedence
- Within the same sheet
  - Declared twice
  - By inheritance
- Between sheets of the same level
- From different sources
  - author, user, browser
- By the specificity of the selector
  - p em versus p
- Author-designated marker (weight)
  - !important

29

Conflict Resolution Overview

- It is a multistage sorting process
  1. Gather the style specifications from style sheets and sort w.r.t. precedence
  2. Add those from user and browser, then sort by origin and weight in the following order (high to low)
     - Important declarations with user origin
     - Important declarations with author origin
     - Normal declarations with author origin
     - Normal declarations with user origin
     - Any declarations with browser (or other user agent) origin
- User-origin has the highest precedence because of potential special needs (e.g. visual impairment)

30

Tie-Breakers

- Specificity
  1. id selectors
  2. Class and pseudo-class selectors
  3. Contextual selectors
  4. General selectors
- Position
  - Later has precedence over earlier

31

Summary

- Why Use CSS
- Three Types of CSS
- Specifying Style Rule
- Selector Forms
- The Box Model
- Revisiting and
- Conflict Resolution – The Cascade

32

Property Value Forms, Font Properties, List Properties, Color, and Text Alignment (Sections 3.5-3.9)

Review on your own

33

Assignment Discussion

What kind of cool sites do you want to build in your assignments?

34