4.1 CSS - Cascading Style Sheets

- Main ideas and essential features
  - as preliminaries for studying XSL
  - What is it?
  - Use and status
  - Processing model
  - Rules, selectors, generated content
  - Usability in practice

W3C CSS Recommendations

  - basic features, from the point of view of HTML
- Level 2 (CSS2), May 1998
  - different media types
  - paged media (for printing), aural style sheets (for speech synthesis)
  - extended selection mechanism
  - generated content, automatic numbering
  - formatting of tables, ...
- Level 3 (CSS3)
  - ongoing work; split in May 2001 into 26 modules

CSS Style Sheets

- Style sheet is a set of style rules
- Style rule syntax:
  - selector { declarations }
  - selector locates elements affected by the rule
  - declarations syntax:
    - prop 1 : val1; ... ; propN : valN;
      - sets values for style properties
    - CSS1 defines about 50 properties, CSS2 about 120

CSS Style Rules

- Example rules
  - H1 { color: blue } /* blue text for first-level HTML headers */
  - H1, H2, H3 { font-style: bold; } /* Alternate selectors grouped together */
  - CODE { font-family: monospace; color: red }
CSS1 Properties (2/2)

- Box properties:
  - for controlling size, margins, borders etc. of "boxes" (see later)
- Classification properties:
  - display (values: block | inline | list-item | none),
  - white-space (values: normal | pre | nowrap),
  - list-style (type/image/position)

Attaching CSS Style to HTML

- Four ways: 1. with a LINK element;
  2. with a STYLE element in document HEAD;
  3. by an import mechanism; 4. in a STYLE attribute

```html
<HTML><HEAD><TITLE>A sample page</TITLE><LINK REL="stylesheet" TYPE="text/css" HREF="my_style.css" />
<STYLE TYPE="text/css">import url(http://styles.com/basic);
  h1 { color: blue; }<BR>
</STYLE></HEAD><BODY><H1>Headline is blue</H1>
  <P STYLE="color: green">but this is green.</P></BODY></HTML>
```

Attaching CSS Style ...

- W3C Rec for linking styles to XML:
  ```xml
  <?xml-stylesheet href="example.css" type="text/css" ?>
  ```
- Rules from different sources merged together
  - including browser defaults and user preferences
  - Problems of "cascading"
    - author: blue background + user: blue text -> ?
    - Should author have control of style?
      - What if visually disabled user needs large font size?
    - Should user have control of style?
      - What if readability depends on detailed size and placement of text?

CSS Processing Model (simplified)

0. Parse the document into a tree
1. Match style rules to elements of the tree
   - annotate each element with a value assigned for each relevant property
     - inheritance and in case of competing rules, elaborate "cascade" rules applied to select which value is assigned
2. Generate a formatting structure of the annotated document tree
   - consists of nested rectangular boxes
3. Render the formatting structure
   - display, print, speak, ...

Inheritance (1/2)

- Most properties are inherited by subelements
  - can be overridden
  - can be modified (e.g. increase indent, set font size to 150% of current)
- Some non-inherited properties:
  - background properties
    - by default shine through the boxes of descendents
  - padding, margin and border properties
    - but they effect the placement of sub-elements

Inheritance (2/2)

- Consider document fragment
  ```html
  <chap><title>...</title>
  <section>...</section>
  </chap>
  ```
- and rules
  ```css
  chap { font-size: 12pt; font-family: serif; title | font-size: 150% }
  ```
- Now both title and section will be formatted using a serif font, but title with a 50% larger font size
CSS Box Model

- Transcription maps document elements into nested rectangular boxes, which have
  - a core content area
  - optional surrounding margin, border and padding areas
  - controlled by corresponding properties margin, border, padding, width and height
    (last two most useful for scaling images)

Dimensional properties of boxes

CSS1 Properties (2/2)

- Box properties:
  - width, height, float, clear;
  - margin, (top/-right/-bottom/-left);
  - padding, (top/-right/-bottom/-left);
  - border-width, (top/-right/-bottom/-left);
  - border-color, border-style;
  - border, (top/-right/-bottom/-left);

Boxes: Example

Main types of elements

- Inline elements (display:inline)
  - default; can occur on the same line with other elements; e.g., EM in HTML

- Block-level
  - formatted as boxes separated by line breaks
  - default formatting of; e.g., P, H1, H2 in HTML
  - specified by display:block
  - display:list-item
    - block preceded by list-item marker

- Element suppression: display:none

CSS Selectors: Simple

- Application of style rules determined by matching selectors to elements of the document tree
- Element type name, e.g. P or H1
  - matches any instance of the element type

- CSS2 adds
  - a universal selector - matching any element
  - tests for attributes:
    - file="fig1.jpg"
    - with given value for attribute file
    - any element with that value for attribute file
    - fig[fig]: element fig with attribute file
CSS Selectors: Contextual

- Element inclusion by listing simple selectors for ancestors
  - e.g. items in ordered lists in paragraphs:
    - P OL LI [ ... ]
- CSS2 adds
  - direct inclusion: Parent > Child
  - conditions on siblings:
    - E + A: element A preceded by an element
      satisfying selector E
    - B:first-child: B as the first element child

Examples of CSS2 Selectors

- Don't indent a P immediately following MATH:
  - MATH + P { text-indent: 0; }
- Reduce vertical space between an H1 and an H2 immediately following it:
  - H1 + H2 { margin-top: -5mm; }
- Don't indent the first P of a DIV:
  - DIV > P:first-child { text-indent: 0; }

Counters and generated content

- CSS1 restricted to adorning elements with assigned style properties
  - automatic numbering of list items on a single level
- CSS2 adds insertion of generated content before and after selected elements
- Example:
  - Number theorems within each chapter
  - Precede each theorem by "Theorem" and its number
  - Follow the theorem by a little box □

Generated content: Example

- Style rules for the task:
  - chapter { counter-reset: theorCnt }
  - theorembefore {
    content: "Theorem ", " ";
    counter-increment: theorCnt;
    font-weight: bold; }
  - theoremafter {
    content: url("box.gif");
    float: right; }
  - Also possible to include attribute values of the selected element in generated content using
    attr(Title)

Limitations of CSS

- Limited transcription capabilities
  - limited transposition of elements
    (float: left/right)
  - calls of parameterised formatting tasks the major
    transcription type supported
- In CSS1 context specification limited:
  - no sibling or parent/child relationships
  - limited use of attributes (CSS1: only class)
  - CSS2 more powerful, but
    - no access to element’s children or content
    - unable to access targets of cross references

Limitations of CSS

- Non-programmable
  - no decision structures
  - unable to store calculated quantities
  - non-extensible
  - > relatively simple
- Western-language orientation (left-to-right)
- XSL allows unrestricted transformations of the document to precede a CSS-like formatting

Browser Implementations

- CSS support in major browsers is rather discouraging (See, e.g., http://www.webreview.com/style/)
  - full CSS not usable on the Internet
  - (Almost) complete implementations of CSS1
    - Netscape Navigator 6
    - implements also most of CSS2 selectors
      - MS Internet Explorer 5 on Macintosh
  - MS policy of developing own extensions, instead of completing conformance with accepted recommendations strongly criticised