4. Introduction to Style Sheets

- Discusses recently:
  - Programmatic manipulation of documents
- Now a more human-oriented point of view:
  - How to specify the external representation of structured documents?
- Concepts, properties and requirements of style systems on a general level
  - later examples of concrete style languages (CSS, XSL)

Introduction to Style Sheets


- Declarative markup of structured documents indicates purely syntactic structure
  - no semantics (processing, formatting, ...)
- Electronic style sheets
  - specify layout and appearance of document content
  - e.g. FrameMaker templates, Word styles, or LaTeX style files (macros);
  - CSS and XSL style sheets

Why Style Sheets?

- Separation of content and presentation is a basic "dogma" in structured documenting
  - supports maintenance and multi-use of documents
  - (unnecessary complication for single-purpose and single-use documents)
- Relationship between documents and style sheets is many-to-many
  - single style for many documents
    - supports manageability (of, say, consistent look and feel of a corporate Web site)
  - many styles for a single document
    - supports multiple output media (print, different Web clients, handheld devices, ...)

Tasks of a Style Sheet

- Style sheet guides the transformation of
  - descriptive markup into formatter input (stream of formatting commands and text)
    - called transcription by Brüggemann-Klein & Wood
- "Styling = transforming + formatting"
  - matches Extensible Stylesheet Language:
    - XSL = XSLT (transformation language)
    - XSL FO (formatting language)
Process of Transcription/Transformation

Structured document

Transformation

Style sheet
- LaTeX style file, CSS, XSLT

TeX
FOT
(ASCII formatting object tree)

Formatter input

Process of Formatting

■ Creates a detailed description of presentation

Formatter

TeX
FOT...

Style sheet may not have complete control of the final formatted presentation

Process of Rendering

■ Display/play the document on output medium

Printer driver

Display driver

Style Rules

■ Style sheet is a set of style rules
  - Attach transformations to document elements
  - Most style rules in practice structure-based
  - Associated to instances of element types in the document structure tree
  - Also grammar-based style rules
  - Associated to occurrences of element types in the document grammar

Transcription types

■ Fundamental operations of style rules fall into following transcription types:
  1. Calls of parameterised formatting tasks
  2. Generation of text
  3. Automatic numbering
  4. Rearrangement of elements

■ Application of transcriptions can depend on element context

Context Specification (1)

■ General rule:
  - Semantically equivalent structures (instances of the same element type) should be formatted identically

■ Exceptions (due to conventions, aesthetics, etc)
  - All paragraphs indented, except for the first one
  - Heading numbers 1, 2, ... in body, but
    A, B, ... in appendix
  - Author lists in references: "Abo and Ullman" (just a few) vs "Abo et al." (if several authors)
  - Indication of target element type (Table, Figure, Section, ...) for cross references
Context Specification (2)

- Need access to ancestors, siblings, descendants, targets of cross references
- Context conditions by a context predicate
  - Boolean expression built of
    - a reference to the current element
    - functions like parent, leftSibling, leftMostSibling, children, and ref
  - Rule applied if the context condition is true
- Context specification orthogonal to transcription types
  (in XSL through the XPath expression language)

Parameterised Formatting Tasks

- The most common transcription type:
  - set formatting characteristics for sub-elements
    - typeface attribution (for strings or inline-elements)
    - line breaking (for paragraphs or blocks)
    - page breaking (for documents)
  - parameterised by type size, line length, indentation, page height...
- hierarchy of elements
  - hierarchy of nested formatting tasks
  - hierarchy of nested presentation areas

Hierarchy in Formatting

Derived or Inserted Content

- boilerplate text
  - text not present in the source document
    - e.g., letter headings, © marks, bullets, ...
  - textual content inserted at the beginning
    or at the end of the current element
- table of contents, indexes
  - need to specify the source of included material

Numbering (1)

- Different schemes
  - consecutively through document
    - same numbering sequence, possibly common to a
      set of element types (e.g., for Theorems and Examples)
    - nested numbering for, say, nested lists
      - relative to occurrences of another element type
        higher in document hierarchy

Numbering (2)

- Example of different numbering schemes:
  - Section 1
    - Theorem 1, or 1.1
    - Figure 2, or 1, or 1.2, or 1.1
  - Section 2
    - Theorem 3, or 2, or 2.1
- often via named counters with a start value
  and a scope (e.g., in CSS2)
  - in XSLT: special expressions for generating numbers
**Sub-element Rearrangement**

- **Examples**
  - reverse the order of titles and authors in references
  - sorting of the reference list
  - requires functions operating on textual contents
- **Sub-element suppression** a special case
- Not supported by the weakest style systems

**Viewpoint on Style Languages**

- **Style systems should support traditional stylistic conventions discussed above**
- **Next a look at CSS, and later at XSL**
  - How do the discussed concepts appear in the languages?
  - How do the languages support these general requirements?