XML and Rendering -- XSL

Michael B. Spring
Department of Information Science and Telecommunications
University of Pittsburgh
spring@imap.pitt.edu
http://www.sis.pitt.edu/~spring

Context

- XML adds new capabilities that will not be immediately apparent to end users as it will be some time before servers and browsers implement the capability
- The XML Style Language (XSL) provides a capability similar to that provided by Cascading Style Sheets
- XSL can be used like CSS or more appropriately through XSLT to transform an XML document to one specified in terms of formatting objects – the fo namespace
- An XML document using fo tags can be processed to some rendered format by a processor that understands the XSL specification

Overview

- Rendering documents
- Cascading Style Sheets (CSS)
  - CSS Version 1
  - CSS Version 2
- XSL
  - Formatting Objects
  - Master Pages and Page Sequences
  - Static Content and Flow Elements
  - Block, inline, and table elements
  - Properties

The Evolution of Rendering Capability

- Initially, the rendering of HTML elements was built into the browsers
- With the development of XML, the focus returned to the goal of device independent formatting
- Cascading Style Sheets were the first step in this direction. They work for both HTML and XML
  - Cascading Style Sheets worked differently in Internet Explorer and Netscape Navigator
  - Cascading Style Sheets were specified at two levels, 1 and 2.
- The Formatting Object specification takes the rendition of XML documents to new levels.

Rendering Generally

- Historically, rendering to different devices required radical adjustments – i.e., rendering to a teletype was different than rendering to a phototypesetter (n.b. Scribe)
- With the development of “all-points addressable devices” – laser printers and bit-mapped screens – this need decreased.
- Today, with the need to audio-presentation, graphical presentation, and other semantically differentiated output, the need for device independent rendering has returned.

Graphical Rendering

- To understand graphical rendering, one needs to go back to Postscript and Interpress technologies.
- Consider the following page design goals:
  - The design of the first page of a chapter or letter is different from the succeeding pages
  - The design of left and right pages should be different
  - Material in the header or footer of a page should contain both static and dynamic information
  - Some material should be “flowed” differently – text fills from the top while footnotes fill from the bottom
  - Material should inherit some relative values from parents.
- CSS does some of this, XSL finishes the job.

Cascading Style Sheets(CSS)

- A brief review of CSS sets the stage
- XML uses a processing instruction to associate a style sheet with a document:

  <?xml-stylesheet type="text/css" href = "mysheet.css"?>
- The style sheet associates attributes with elements:
  - The nature of the element -- block, inline, list item, none
  - The nature of the text -- font characteristics, text alignment, background, etc.
- Parent node styles are inherited by children.
- Style information is "cascaded" such that multiple author and reader specifications can be resolved

An Example

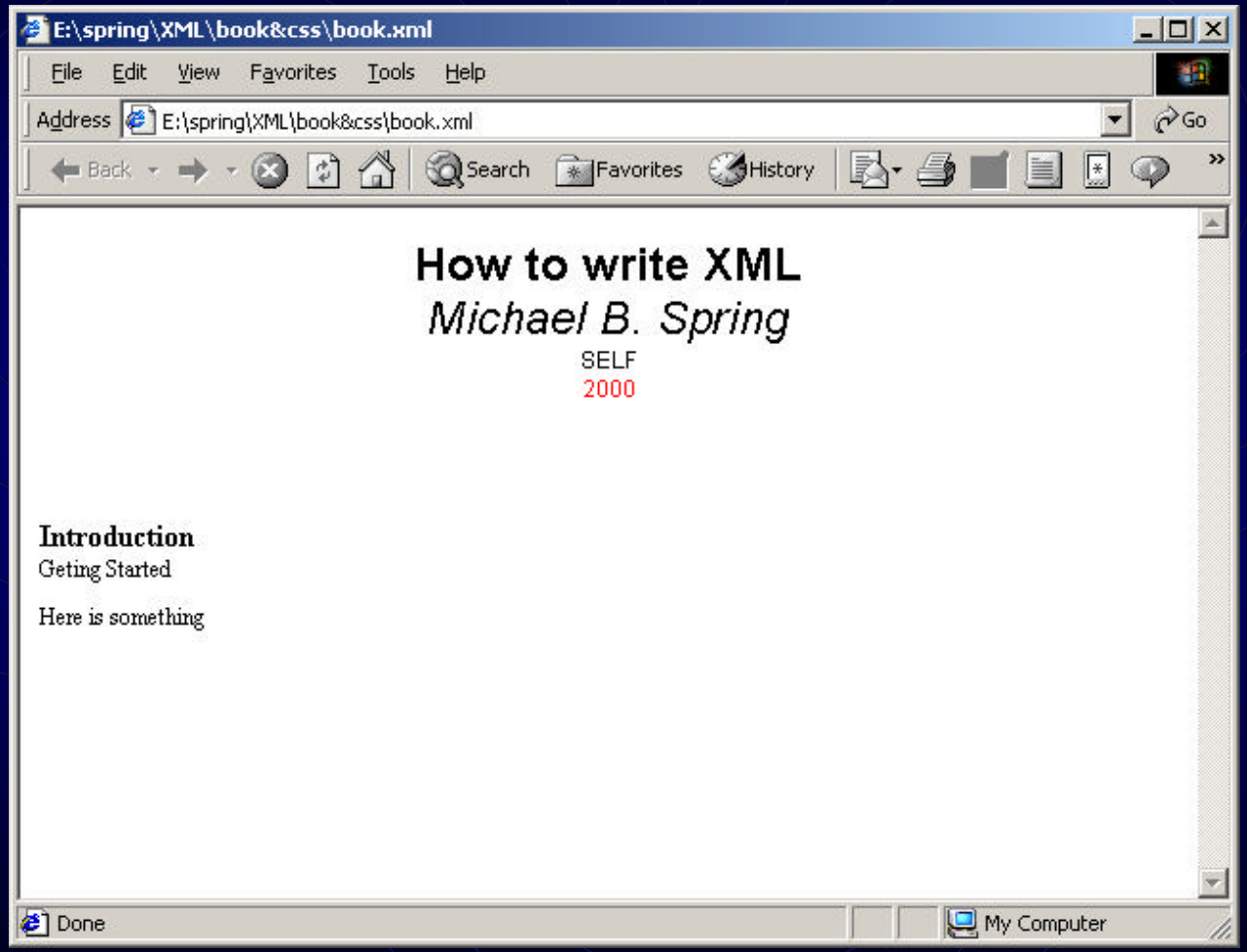
- **Given the following Document**

```
<?xml version='1.0' standalone='no'?>
<?xml-stylesheet type="text/css"
    href="book.css"?>
<book>
<titlepage>
<btitle>How to write XML</btitle>
<author>Michael B. Spring</author>
<publisher>SELF</publisher>
<date>2000</date>
</titlepage>
<chapter>
<ctitle>Introduction</ctitle>
<para>Geting Started</para>
<para>Here is something</para>
</chapter>
</book>
```

- **And the following stylesheet**

```
book { display: block }
titlepage { display: block; font-family: Arial;
    text-align: center; font-size: 24}
btitle { display: block; font-weight: bold }
author { display: block; font-style: italic }
publisher { display: block; font-size: 12 }
date { display: block; font-size: 12;
    color: red }
chapter { display: block; font-family: Times;
    margin-top: 60px; font-size: 12}
ctitle { display: block; font-size: 16;
    font-weight: bold }
para { display: block;
    margin-bottom: 10px }
```

Internet Explorer Output



Overview of Cascading Style Sheets (CSS)

- All of Cascading Style Sheets is beyond the scope of this presentation. A few of the capabilities are:
  - CSS Version 1
    - Differentiating elements
    - Specifying values
    - “display” types
    - Font and text properties
    - Box properties
  - CSS Version 2
    - New differentiation
    - Pages and visual formatting
    - Tables
    - Generated content

CSS Version 1: Differentiating Elements

- CLASS and ID attributes may be used to differentiate elements
  ELEMENT.CLASS {style info}
- Multiple Elements may use the same style
  E1, E2, E3 {style info}
- The first letter and first line may be handled separately
  ELEMENT:first-letter {style info}
  ELEMENT:first-line {style info}
- Elements may inherit from multiple overlapping styles
  E1, E2, E3, E4 {some style info}
  E3 {some style info}
  E1, E2 {some style info}
- Elements may be differentiated by context(parent)
  ELEMENT1 {style info}
  PARENTB ELEMENT1 {style info}
- Elements may be differentiated individually
  <ELEMENT1 STYLE="style info">

CSS Version 1: Specifying Values

- The various style elements have appropriate units – numbers, colors, keywords.
- The various style elements may be specified, as appropriate, in absolute, percentage, or relative values
  - When the values are absolute, they may be specified in inches, centimeters, points, or picas
  - The relative values relate to proportional typefaces and include em – the with of the letter “m” in the current font, ex – the height of the letter x, and px – the square size of the pixel
  - Percentage units pertain to the size in relation to the size of the parent.

CSS Version 1: Display Values

- The basis of layout under CSS1 involves separating all display into four categories:
  - block elements are those that involve a line break between elements
  - inline elements are those that do not involve a line break
  - list-item elements are those that have some marker associate with the block element
  - none elements are those that are not displayed
- Spacing and margin information is only associated with block elements.
- Block elements may also be nested in various ways

CSS Version 1: Font Properties

- The font properties that can be specified include:
  - font-family allows either specific “Palatino” or generic “serif” families to be specified – a coma separated list provides fallback
  - font-style allows normal, italic, or oblique
  - font-variant allows normal or small-caps
  - font-weight allows normal or bold or weights from 100 to 900 (only hundreds)
  - font-size allows point sizes or keyword sizes – x-small, small, medium, large, xx-large
    - Font-size can also be specified as “smaller” or as a percentage of the parent

CSS Version 1: Text Properties

- The following text properties may be set:
  - text-alignment allows center, left, right, justify
  - text-indent allows first line indents
  - line-height allows for line spacing
  - vertical-alignment allows sub, super, top, etc.
  - text-transform allows uppercase, lowercase, capitalize
  - text-decoration allows underline, overline, blink, etc.
  - word-spacing takes a positive or negative value to increase or decrease interword spacing
  - letter-spacing takes a positive or negative value to increase or decrease interletter spacing

CSS Version 1: Box Properties

- Box properties may be set related to blocks:
  - Basic features: The following three styles can be specified generally of for the top, left, etc. individually
    - margins specifies whitespace around the box
    - borders specifies the style, width, color of borders
    - padding specifies the space between the border and the text
  - Special Features
    - size is generally used for images, but any box can be absolutely specified – with unpredictable results
    - positioning is normally one box after another with no two at the same vertical position, but “float” allows two elements side by side and clear demands non shared space

CSS Version 2: Differentiating Elements

- New classes:
  - :first-child – the first child of a parent
  - :hover and :focus – the designated and selected input object
  - :first, :left and :right – the first, right, and left hand pages of a printed document
- New pseudo elements
  - :after and :before – allows objects to be inserted before of after elements
- Pattern matching
  - CSS2 allows all sorts of relationships to be specified such as sibling relationships, elements with an attribute, elements in a given language, all elements, etc.

CSS Version 2: Pages and Visual Formatting

- Page formatting includes:
  - Specification of left, right, and first pages
  - Setting of sizes, margins, rotation, and marks and page breaks
  - Attachment of page properties to elements

    @page mypage {size: landscape; margin: 1.0in }
    ELEMENT {page: mypage}
- Visual formatting (beyond block and inline)
  - compact and run-in allow conditional treatment of blocks as inline elements and vice-versa
  - marker allows generated output from the style sheet used by the :before and :after classes
  - table values are incorporated as shown on the next slide

CSS Version 2: Tables

- A whole set of values for tables is provided including:
  - table
  - inline table
  - table-row-group
  - table-header-group
  - table-footer-group
  - table-row
  - table-column group
  - table-column
  - table-cell
  - table-caption

CSS Version 2: Miscellaneous

- Size can now be specified as min, max, and absolute
- When size is specified, what do do with extra text can be specified as scroll, hidden, visible, etc.
- Clipping areas can be specified
- Cursor shape in an area can be controlled
- Finer font adjustments can be made
- Positioning allows for the z-axis
- Counters are provided for automatic numbering
- Various aural capabilities are provided for

The XML Style Language (XSL)

- XSL is a two part standard and is highly dependent on the XPath specification
  - Part one specifies the transformation language (XSLT) that allows one document to be transformed into another
  - Path two specifies the XSL formatting objects (XSLFO) that define the semantics and the specification of rendered objects.
- XSLFO provide a vast and precise set of tools for rendering documents
  - XSLFO is an alternative to the CSS standards
  - XSLFO owes its heritage to the Document Style Semantics and Specification Language (DSSSL)

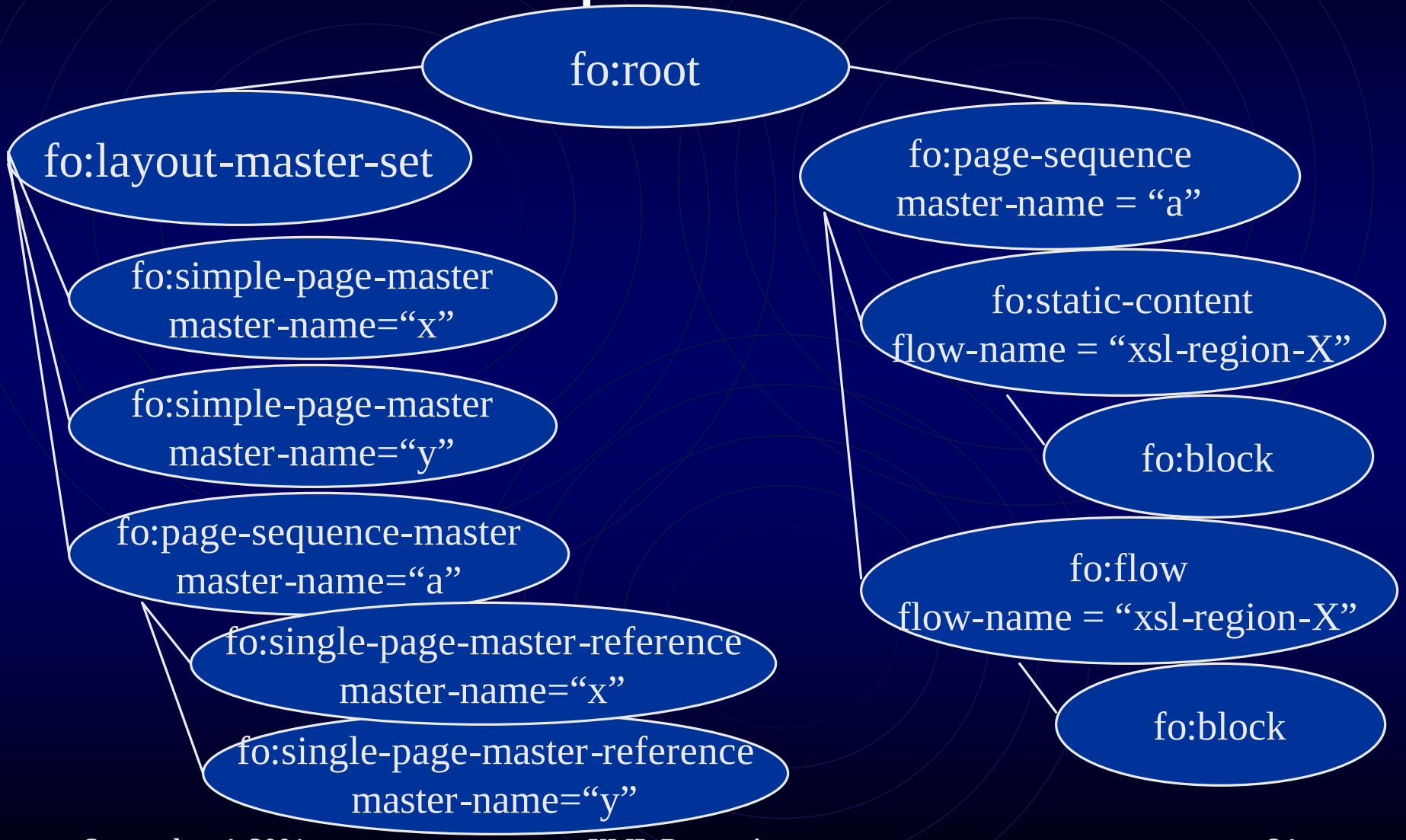
Formatting Objects

- There are 56 formatting object elements
- There are more than 250 properties defined that can be specified related to these elements
- The elements define a hierarchy similar to the layout hierarchy specified for interpress and ODA
- The formatting objects are specified as an XML document where the root is fo:root
  - fo:root has two children
    - fo:layout-master-set, which defines one or more page masters
    - fo:page-sequence, which maps content to page masters

Using XSL to Render Documents

- There are four basic ways to render documents with XSL:
  1. Use the transform language and output the text of the XML document
  2. Use the transform language to replace XML tags with HTML tags for display in conventional browsers
  3. Hand write a document using formatting object elements to allow XSLFO compliant applications to display the result.
  4. Finally, and most desirably, transform an XML document using XSLT to replace the original XML elements with XMLFO tags that cane be rendered by XSLFO compliant applications

Conceptual Model



A Simple Example Using XSLFO Explicitly

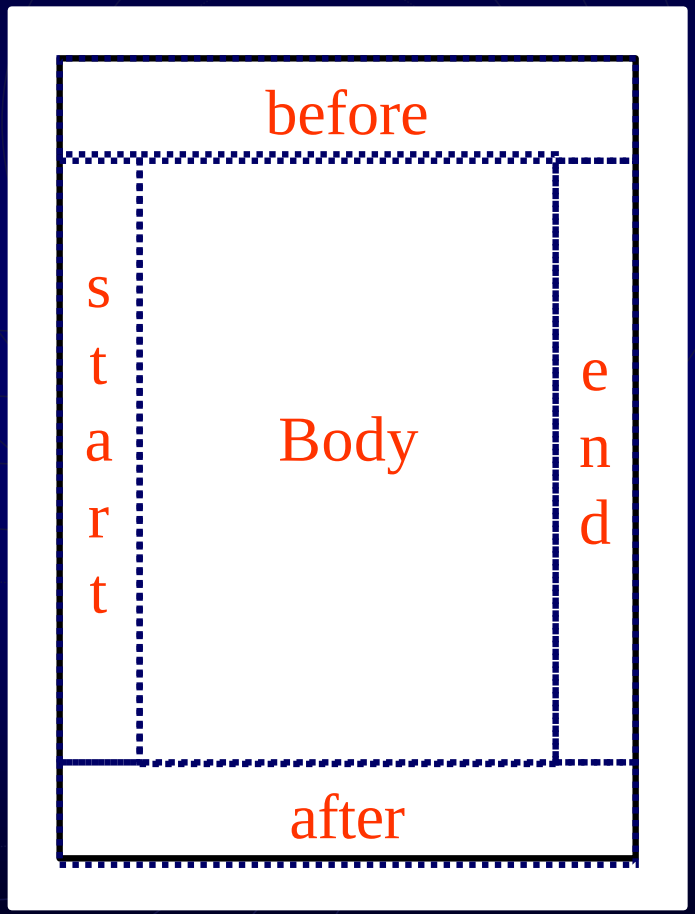
```
<?xml version="1.0"?>
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
 <fo:layout-master-set>
  <fo:simple-page-master master-name=“mypage">
   <fo:region-body/>  </fo:simple-page-master>
 </fo:layout-master-set>
 <fo:page-sequence master-name=“mypage">
  <fo:flow flow-name="xsl-region-body">
   <fo:block font-size="20pt" font-family=“sanserif" line-height=“24pt">
    Michael B. Spring   </fo:block>
   <fo:block font-size=“10pt" font-family="serif"   line-height=“12pt" >
    M.B.Spring is a faculty member in the Department of Information Science and
   Telecommunications  </fo:block>
  </fo:flow>
 </fo:page-sequence>
</fo:root>
```

Master Pages and Regions

- The layout master set allows one or more page templates to be defined including:
  - Overall page size and margins
  - Writing mode and orientation
  - The size of five regions of the page named before, after, start, end, and body
    - The body region is not sized but extends from the left page margin to the right
    - The other regions have “extent” which is a logical distance from the appropriate margin
    - The body region overlaps the other regions and margins must be used to prevent overwriting

A Page Master with Regions

```
<fo:layout-master-set>
  <fo:simple-page-master
     master-name="only"
     page-width="8.5in“ page-height="11in"
     margin-top="0.5in"  margin-bottom="0.5in"
     margin-left="0.5in"  margin-right="0.5in">
    <fo:region-start  extent="1.0in"/>
    <fo:region-before extent="1.0in"/>
    <fo:region-body   margin="1.0in"/>
    <fo:region-end    extent="1.0in"/>
    <fo:region-after  extent="1.0in"/>
  </fo:simple-page-master>
</fo:layout-master-set>
```



Page Sequences(1)

- The layout-master-set may include one or more named page-sequence-masters that specify the order and occurrence of page-masters
- The simplest page-sequence-master would specify a single page – what happens to content that doesn't fit is unclear

```
<fo:page-sequence-master master-name="mysequence">
  <fo:single-page-master-reference master-name="A"/>
</fo:page-sequence-master>
```

- A more normal page-sequence-master could specify a first page master and a subsequent page master.

```
<fo:page-sequence-master master-name="mysequence">
    <fo:single-page-master-reference master-name="A"/>
    <fo:repeatable-page-master-reference master-name="B"
                maximum-repeats="100"/>
</fo:page-sequence-master>
```

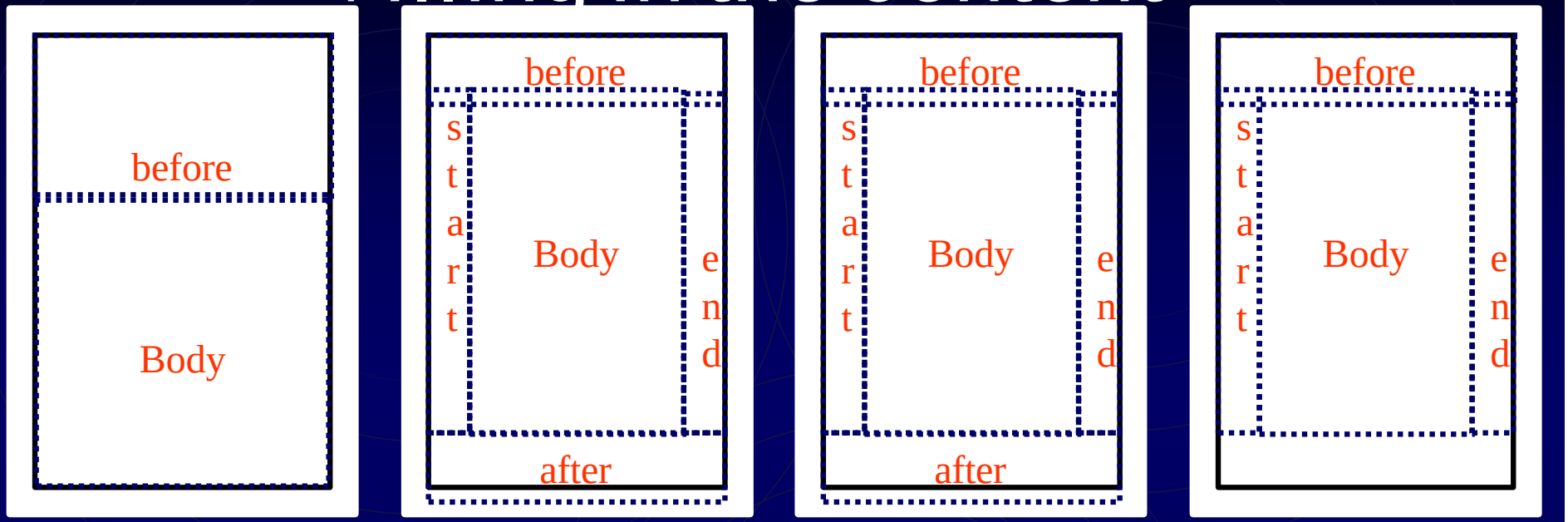
Page Sequences(2)

- For complex documents the subelement, repeatable-page-master-alternatives is used.
- The children of this element, which have three attributes are conditional-page-master-references.
  - The “page-position” attribute can be first, last, rest, or any
  - The “odd-or-even” attribute can be odd, even, or any
  - The “blank-or-not-blank” attribute can be blank, not-blank, or any
- Combinations of these allow the kind of complex formatting that might be used in a book as shown on the following slide.

Complex Page Sequence Example

```
<fo:page-sequence-master master-name="mysequence">
  <fo:repeatable-page-master-alternatives>
    <fo:conditional-page-master-reference
          page-position="first" odd-or-even="odd"
          master-name="ChapterStart"/>
    <fo:conditional-page-master-reference
          page-position="rest" odd-or-even="odd"
          master-name="ChapterOdd"/>
    <fo:conditional-page-master-reference
          page-position="rest" odd-or-even="even"
          master-name="ChapterEven"/>
    <fo:conditional-page-master-reference
          page-position="last" odd-or-even="even"
          master-name="ChapterLastEven"/>
  </fo:repeatable-page-master-alternatives>
</fo:page-sequence-master>
```

Filling in the Content



- The “page sequence” element defines the master set of pages.
- To fill the various regions of the pages with content, “static-content” and “flow” elements specify the regions to be filled with content

An Example of the Associations

Causes a page set to be built

Fills the body region of those pages

```
<fo:page-sequence master-name="mypage">
  <fo:flow flow-name="xsl-region-body">
    <fo:block font-size="20pt" font-family="sanserif" line-height="24pt">
      Michael B. Spring   </fo:block>
    <fo:block font-size="10pt" font-family="serif"   line-height="12pt" >
      M.B.Spring is a faculty member in the Department of Information
    Science and Telecommunications  </fo:block>
  </fo:flow>
</fo:page-sequence>
```

Fills in one block after another

Static-Content and Flow Elements

- A page-sequence element has “static-content” and “flow” element children
- If static-content elements are used, they must appear before any flow elements
- The static-content information appears on every page, while flow information appears only once.
- The information in a static-content element can be developed dynamically – e.g. page number.
- These elements have many possible children which do the actual job of specifying appearance

The Formatting Objects Proper

- Block objects are the primary means by which properties are assigned. They include
  - <block> and <block-container>
  - <table> and <list-block>
- Line objects are generated by the system from the content of block and inline area objects
- Inline objects allow local overriding of selected block attributes. They include among others:
  - <inline> and <inline-container>
  - <character> and <bidi-override>
  - <external-graphic> <leader>
- Out-of-line objects, which float on pages, include:
  - <float> <footnote>, and <footnote-body>

Block Elements

- Basically there are only two block elements
  - <block>
  - <block-container>
- Both of these may be nested
- The <block-container> element is used to apply a set of properties to the blocks nested within it
- <table>s and <list-block>s are also considered block elements and serve the expected special functions
- <fo:block> is the workhorse for associating properties through its attributes

Tables and Lists

- Tables match up with CSS tables and include:
  - <table> and <table-and-caption>
  - <table-caption>
  - <table-header> and <table-footer>
  - <table-body> and <table-column>
  - <table-row> and <table-cell>
- List elements include:
  - <list-block>,
  - <list-item-label> and <list-item-body>
- These elements function as they do under html and CSS and can have all of the relevant properties associated with them.

Selected In-line Elements

- In-line elements allow overrides within the scope of a block.
  - <inline> is the most basic element and line the block does the majority of work of associating attributes with a set of characters
  - <inline-container> sets a superset of attributes with the enclosed elements
  - <character> can be used to substitute a specialized glyph with a character
  - <bidi-override> can set text directionality
  - <page-number> and <page-number-reference> can be used to set the current page number or the page number on which an element with an id exists

Out-of-line Elements

- The out-of-line elements are primarily intended for objects that float on a page and include:
  - <float> is the generic block container for objects that are to float on the page.
  - <footnote> and <footnote-body> -- where footnote body is a child of footnote. The footnote appears in line and the body floats.

Properties

- How to most economically organize the properties that can be associated with blocks and inline elements is problematic.
- Here, they are presented in 5 categories:
  - Block spacing properties
  - Font properties
  - Line spacing properties
  - Alignment properties
  - Break, widow and orphan properties
  - Others

How the XSL Standard Organizes Properties

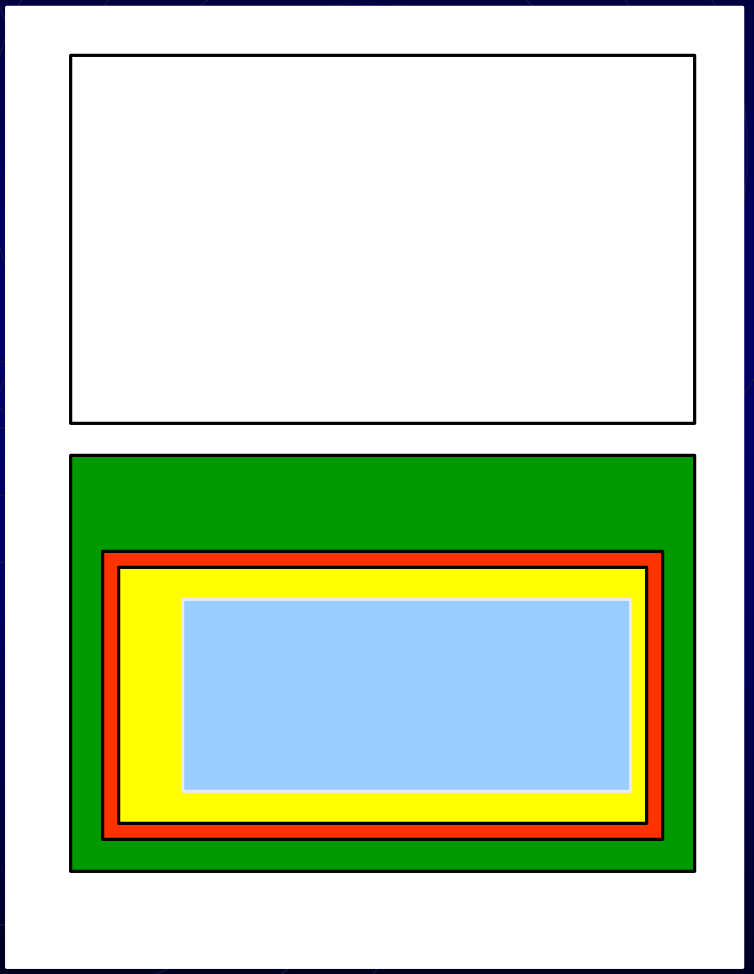
- Common Accessibility
- Common Absolute Position
- Common Aural
- Common Border, Padding, and Background
- Common Font
- Common Hyphenation
- Common Margin -Block
- Common Margin -Inline
- Common Relative Position
- Area Alignment
- Area Dimension
- Block and Line-related
- Character
- Color-related
- Float-related
- Keeps and Breaks
- Layout-related
- Leader and Rule
- Props for Dynamic Effects Formatting Objects
- Props for Markers
- Props for Number to String Conversion
- Pagination and Layout
- Table
- Writing-mode-related
- Miscellaneous
- Shorthand

Font Properties

- The basic font properties include:
  - font-family: A ordered list of font names (in order of preference)
  - font-size: A signed length
  - font-weight: The thickness of the font – bold, normal, light, or as a century value between 100 and 900
  - font-style: The slant of the font – italic, normal, reverse-normal, etc.
  - font-stretch: The width of the font – condensed, expanded, etc.
  - font-variant: Either normal or small-caps
- There are additional text (e.g. capitalization) and character (e.g. color) properties as well

“Spacing” Properties

- Spacing include dozens of properties and various ways to specify them.
- They are normally applied to blocks, but can be applied to inline areas.
- It is easiest to begin by viewing the properties graphically where each object (blue) has space (e.g. –before) (green), a border (red), and padding (yellow).
- The spacing may be homogeneous or not.



Spacing Properties

- The spacing properties are similar to those in CSS, and have alternative wordings for consistency with CSS
  - The preferred reference to the directions is before and after and top and bottom.
  - The border has a width, color, and style
  - The padding attributes are a distance metric
  - The space attributes are a little unusual:
    - space-before and space after is an amount exclusive of the border and padding
    - start-indent and end-indent is an amount that includes the border and padding

Alignment Properties

- Text alignment properties allow a rich set including:
  - left, right, centered justified
  - inside and outside – alignment in folio style layouts
- Vertical alignment properties allow the alignment of an inline element related to the line and include:
  - super and sub
  - baseline, top, middle, bottom, text-top, and text-bottom

Break, Widow and Orphan Properties

- The break properties allow the contiguity of objects to be specified:
  - Keep-with-next, keep-with-previous and Keep-together properties take an always or auto value or an integer value where larger is a stronger directive
  - Break-before and break-after property specify the nature of the break – column, page, even-page, odd-page
- The widows and orphans properties
  - The number of lines from a block that must be left at the bottom of a page together (to prevent orphans)
  - The number of lines from a block that must be left at the top of a page together (to prevent widows)

Other Properties

- Indent
- Sentence
- Hypenation
- Write properties
- Scroll hide clip
- Leaders

The Big Picture