Lecture 6: Namespaces in XML 1.1 (Second Edition), SVG 1.1, and XSL (XSL-FO) 1.1

29 November 2007

David J. Malan
malan@post.harvard.edu
Computer Science E-259

Last Time

- XSLT, Continued
Last Time

XSLT 1.0, Continued

- Data Types
- Conditions
- Iteration
- Iteration, Sorted
- Patterns
- Modes
- Built-In Templates, Revisited
- Priorities
- Includes
- Imports
- Variables
- Result Tree Fragments
- Copying Nodes
- Outputting Nodes
- Attribute Value Templates
- Multiple Source Documents
- Recursion
- Extension Elements and Functions
This Time

- Namespaces in XML 1.0 (Second Edition)
- SVG 1.1
- XSL (XSL-FO) 1.1
- Project 2
Namespaces in XML 1.0 (Second Edition)

History

- Recommendation as of 1/99.
  - "XML namespaces provide a simple method for qualifying element and attribute names used in Extensible Markup Language documents by associating them with namespaces identified by URI references."
Namespaces in XML 1.0 (Second Edition)

Declaring

```xml
<cscie259:foo
 xmlns:cscie259="http://www.fas.harvard.edu/~cscie259/namespace">
 <bar baz="qux"/>
 <oogle cscie259:foogle="google"/>
</cscie259:foo>
```

Defaulting

```
<foo
    xmlns="http://www.fas.harvard.edu/~cscie259/namespace">
  <bar baz="qux"/>
  <oogle foogle="google"/>
</foo>
```

SVG 1.1

History

- Recommendation as of 1/03.
  - "SVG is a language for describing two-dimensional graphics in XML. SVG allows for three types of graphic objects: vector graphic shapes (e.g., paths consisting of straight lines and curves), images and text."
SVG 1.1

Bitmaps

SVG 1.1

Bitmaps

Source of screen shot unknown.
SVG 1.1

Vector Graphics

SVG 1.1

Animated Graphics

SVG 1.1

Underground Graphics

SVG 1.1

Russian Graphics

SVG 1.1

Text

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
   "http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">

<svg:svg xml:space="preserve"
   xmlns:svg="http://www.w3.org/2000/svg">
  <svg:text style="fill:red;" y="15">This is SVG.</svg:text>
</svg:svg>
```

Adapted from http://www.adobe.com/svg/basics/getstarted2.html.

SVG 1.1

Shapes

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
"http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">

<svg:svg xml:space="preserve"
   xmlns:svg="http://www.w3.org/2000/svg">
   <svg:rect style="fill:blue;" width="250" height="100"/>
</svg:svg>
```

Adapted from http://www.adobe.com/svg/basics/getstarted2.html.

SVG 1.1

Shapes

```xml
<circle cx="25" cy="25" r="20" fill="red"/>

<ellipse cx="25" cy="25" rx="20" ry="10" fill="red"/>

<line x1="5" y1="5" x2="45" y2="45" stroke="red"/>

<polygon points="5,5 45,45 5,45 45,5" stroke="red" fill="none"/>

<polyline points="5,5 45,45 5,45 45,5" stroke="red" fill="none"/>

<rect x="5" y="5" rx="5" ry="5" width="40" height="40" fill="red"/>
```

Paths

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
 "http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">

<svg:svg xml:space="preserve"
   xmlns:svg="http://www.w3.org/2000/svg">
   <svg:path d="M 50 10 L 350 10 L 200 120 z"/>
</svg:svg>

Paths are described using the following data attributes: "moveto" [M] (set a new current point), "lineto" [L] (draw a straight line), "curveto" [C] (draw a curve using a cubic Bezier), "arc" [A] (elliptical or circular arc), and "closepath" [z] (close the current path by drawing a line to the last "moveto" point).

Adapted from http://www.adobe.com/svg/basics/getstarted2.html.
Paths

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN" "http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">

<svg:svg xml:space="preserve"
   xmlns:svg="http://www.w3.org/2000/svg">
   <svg:path d="M5,45 C5,5 45,5 45,45" stroke="red" fill="none"/>
</svg:svg>

Paths are described using the following data attributes: "moveto" [M] (set a new current point), "lineto" [L] (draw a straight line), "curveto" [C] (draw a curve using a cubic Bezier), "arc" [A] (elliptical or circular arc), and "closepath" [z] (close the current path by drawing a line to the last "moveto" point).

Adapted from http://www.adobe.com/svg/basics/getstarted2.html.
Anchors

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
    "http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">

<svg:svg xml:space="preserve"
    xmlns:svg="http://www.w3.org/2000/svg"
    xmlns:xlink="http://www.w3.org/1999/xlink">
    <svg:a xlink:href="http://www.fas.harvard.edu/~cscie259/" target="_blank">
        <svg:text style="fill:red;" y="15">This is CSCI E-259.</svg:text>
    </svg:a>
</svg:svg>

Using Definitions

<defs>
  <rect id="rect" width="15" height="15" fill="red"/>
</defs>
<use x="5" y="5" xlink:href="#rect"/>
<use x="30" y="30" xlink:href="#rect"/>
**XSL (XSL-FO) 1.1**

**History**

- Extensible Stylesheet Language (XSL) Version 1.0 is a recommendation since 10/01.
  - “Given a class of arbitrarily structured XML documents or data files, designers use an XSL stylesheet to express their intentions about how that structured content should be presented; that is, how the source content should be styled, laid out, and paginated onto some presentation medium, such as a window in a Web browser or a hand-held device, or a set of physical pages in a catalog, report, pamphlet, or book.”
- Builds on CSS2 and Document Style Semantics and Specification Language (DSSSL).
XSL (XSL-FO) 1.1

Rendering

- Framemaker's Maker Interchange Format (MIF)
- HP's Printer Control Language (PCL)
- Java's Abstract Window Toolkit (AWT)
- Portable Document Format (PDF)
- Postscript
- Rich Text Format (RTF)
- Scalable Vector Graphics (SVG)
- Text
- XML Area Tree
- ...
XSL (XSL-FO) 1.1

hello, world

<?xml version="1.0" encoding="iso-8859-1"?>

<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
   <fo:layout-master-set>
      <fo:simple-page-master master-name="mylayout">
         <fo:region-body margin="1in"/>
      </fo:simple-page-master>
   </fo:layout-master-set>

   <fo:page-sequence master-reference="mylayout">
      <fo:flow flow-name="xsl-region-body">
         <fo:block>hello, world</fo:block>
      </fo:flow>
   </fo:page-sequence>

</fo:root>

XSL (XSL-FO) 1.1

Terminology

- master
  - "pagination and layout specifications"
- flow
  - "the content that is distributed into the pages"
- block
  - "The fo:block formatting object is commonly used for formatting paragraphs, titles, headlines, figure and table captions, etc."
- inline
  - "The fo:inline formatting object is commonly used for formatting a portion of text with a background or enclosing it in a border."
XSL (XSL-FO) 1.1

hello, world

<?xml version="1.0" encoding="iso-8859-1"?>
"This is the top node of the formatting object tree. It holds an fo:layout-master-set formatting object (which holds all masters used in the document), an optional fo:declarations, and one or more fo:page-sequence objects. Each fo:page-sequence represents a sequence of pages that result from formatting the content children of the fo:page-sequence."
hello, world

"The fo:layout-master-set is a wrapper around all masters used in the document. This includes page-sequence-masters, page-masters, and region-masters."

<fo:layout-master-set>

</fo:layout-master-set>
"The fo:simple-page-master is used in the generation of pages and specifies the geometry of the page. The page may be subdivided into up to five regions: region-body, region-before, region-after, region-start, and region-end."

```xml
<fo:simple-page-master master-name="myfirst">
  ...
</fo:simple-page-master>
```
XSL (XSL-FO) 1.1

hello, world

"Used in constructing a simple-page-master. This region specifies a viewport/reference pair that is located in the 'center' of the fo:simple-page-master."

<fo:region-body margin="1in"/>

"The fo:page-sequence formatting object is used to specify how to create a (sub-)sequence of pages within a document; for example, a chapter of a report. The content of these pages comes from flow children (consisting of the single fo:flow and any fo:static-content flow objects) of the fo:page-sequence. The layout of these pages comes from the fo:page-sequence-master or page-master referenced by the master-reference trait on the fo:page-sequence."

<fo:page-sequence master-reference="myfirst">

</fo:page-sequence>
"The content of the fo:flow formatting object is a sequence of flow objects that provides the flowing text content that is distributed into pages. . . . The flow-names reserved in XSL are: xsl-region-body, xsl-region-before, xsl-region-after, xsl-region-start, xsl-region-end, xsl-before-float-separator, xsl-footnote-separator."

```xml
<fo:flow flow-name="xsl-region-body">

</fo:flow>
```
"The fo:block formatting object is commonly used for formatting paragraphs, titles, headlines, figure and table captions, etc."

<fo:block>hello, world</fo:block>
Font Properties (plus fo:inline)

<fo:block color="#ff0000"
  font-family="Times"
  font-size="24pt"
  font-weight="bold">
  <fo:inline font-size="300%">h</fo:inline>ello, world
</fo:block>
XSL (XSL-FO) 1.1

Text Properties (plus fo:inline)

<fo:block letter-spacing="3pt"
    text-align="justify"
    text-decoration="underline"
    text-indent="3cm"
    word-spacing="10pt">
<fo:inline text-transform="uppercase">Intended for students with previous Java programming and web development experience, this course introduces XML as a key enabling technology in today's e-business applications.</fo:inline> Students will learn the fundamentals of XML: schemas, XSL stylesheets, and programmatic access using standard APIs. Building on these foundations, the course will explore in detail a number of case studies that utilize XML in e-business: e-commerce, web personalization, portals, and web services. Data modeling techniques in XML will be introduced in the context of the case studies.
</fo:block>
XSL (XSL-FO) 1.1

Tables

XSL (XSL-FO) 1.1

Multiple Blocks

<?xml version="1.0" encoding="iso-8859-1"?>
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
  <fo:layout-master-set>
    <fo:simple-page-master master-name="mylayout">
      <fo:region-body margin="1in"/>
    </fo:simple-page-master>
  </fo:layout-master-set>
  <fo:page-sequence master-reference="mylayout">
    <fo:flow flow-name="xsl-region-body" font="14pt Times">
      <fo:block font-size="24pt" text-align="center" space-before="30pt" space-before.conditionality="retain" space-after="12pt">The Jabberwocky</fo:block>
      <fo:block font-style="italic" text-align="end" space-before="9pt">Lewis Carroll</fo:block>
      <fo:block start-indent="1.5in" space-after="9pt">Twas brillig, and the slithy toves</fo:block>
      <fo:block>Did gyre and gimble in the wabe:</fo:block>
      <fo:block>All mimsy were the borogoves,</fo:block>
      <fo:block>And the mome raths outgrabe.</fo:block>
      <fo:block start-indent="1.5in">Beware the Jabberwock, my son!</fo:block>
      <fo:block>The jaws that bite, the claws that catch!</fo:block>
      <fo:block>Beware the Jubjub bird, and shun</fo:block>
      <fo:block>The frumious Bandersnatch!</fo:block>
    </fo:flow>
  </fo:page-sequence>
</fo:root>


XSL (XSL-FO) 1.1

With XSLT

```xml
<?xml version="1.0" encoding="iso-8859-1"?>
<poem>
<title>Jabberwocky</title>
<poet>Lewis Carroll</poet>
<stanzas>
  <stanza>
    <verse>'Twas brillig, and the slithy toves</verse>
    <verse>Did gyre and gimble in the wabe;</verse>
    <verse>All mimsy were the borogoves,</verse>
    <verse>And the mome raths outgrabe.</verse>
  </stanza>
  <stanza>
    <verse>"Beware the Jabberwock, my son!"
    <verse>The jaws that bite, the claws that catch!
    <verse>Beware the Jubjub bird, and shun
    <verse>The frumious Bandersnatch!"
  </stanza>
  <stanza>
    <verse>He took his vorpal sword in hand:
    <verse>Long time the manxome foe he sought--
    <verse>So rested he by the Tumtum tree,
    <verse>And stood awhile in thought.
  </stanza>
  <stanza>
    <verse>And, as in uffish thought he stood,
    <verse>The Jabberwock, with eyes of flame,
    <verse>Came whiffling through the tulgey wood,
    <verse>And burbled as it came!
  </stanza>
  <stanza>
    <verse>One two! One two! And through and through
    <verse>The vorpal blade went snicker-snack!
    <verse>He left it dead, and with its head
    <verse>He went galumphing back.
  </stanza>
  <stanza>
    <verse>"And hast thou slain the Jabberwock?
    <verse>Come to my arms, my beamish boy!
    <verse>O frabjous day! Callooh! Callay!"
    <verse>He chortled in his joy.
  </stanza>
</stanzas>
</poem>
```
XSL (XSL-FO) 1.1

The Big Picture

Result XML tree is the result of XSLT processing.

Next Time

HTTP 1.1, JavaServer Pages 2.1, and Java Servlet 2.5

- HTTP 1.1
- n-Tier Enterprise Applications
- JavaServer Pages 2.1
- Java Servlet 2.5
- Project 3
Lecture 6:
Namespaces in XML 1.1 (Second Edition), SVG 1.1, and XSL (XSL-FO) 1.1

29 November 2007

David J. Malan
malan@post.harvard.edu