Introduction to XML, VRML, CSS, XSL, DOM

Philippe Martin

(philippe.martin@gu.edu.au)

Laboratory for Knowledge Visualisation and Ordering (KVO)

(http://www.int.gu.edu.au/kvo/)

Overview

Content and structure:
- in XML, e.g. XHTML, XLINK, MATHML, VoiceXML, XUL
- or other MLs, e.g. HTML, VRML

Presentation (screen/paper/audio/...): style sheets, e.g. CSS, XSL

Behavior (event handlers): scripts, e.g. Java, Javascript

API (structure, presentation, event): e.g. DOM, SAX

"Dynamic HTML" = HTML plus CSS and/or JavaScript

Virtual Reality Modeling Language (VRML)

```
#VRML V2.0 utf8
#File name: tut1.wrl
WorldInfo { title "Floppy's VRML97 Tutorial Example 1"
            info ["(C) Copyright 1999 Vapour Technology"]
          }
DEF FBOX Shape { appearance Appearance { material Material {
                                           diffuseColor 0 0.5 0
                                           emissiveColor 0 0.8 0
                                           transparency 0.5 } }
                 geometry Box { }
               }
Transform { scale 2 0.5 2    rotation 0 1 0 0.78
            translation 0 -1.5 0  children [USE FBOX]
          }
```

Extensible Markup Language (XML)

SGML less features too complex for authoring or efficient parsing

Specification of syntactic constraints on document elements/attributes

XML Namespace facility -> vocabulary mix-in
-> incremental decentralized developments

Extract of a XHTML DTD

```
<!ELEMENT html (head, body)>
<!ATTLIST html %i18n;
  xmlns  %URI;  #FIXED 'http://www.w3.org/TR/xhtml1/strict'>

<!ELEMENT body %Blocks;>
<!ATTLIST body %attrs;
  onload   %Script;   #IMPLIED
  onunload %Script;   #IMPLIED>

<!ENTITY %attrs "%coreattrs; %i18n; %events;">
<!ENTITY %coreattrs
 "id     ID           #IMPLIED
  class  CDATA        #IMPLIED
  style  %StyleSheet; #IMPLIED
  title  %Text;       #IMPLIED">
```

MathML example: $x^2 + 4x + 4 = 0$

```
<reln><eq/>
   <apply><plus/>
            <apply> <power/> <ci>x</ci> <cn>2</cn> </apply>
            <apply> <times/> <cn>4</cn> <ci>x</ci> </apply>
            <cn>4</cn>
   </apply>
   <cn>0</cn> </reln>
```

```
<mrow><mrow><msup> <mi>x</mi> <mn>2</mn> </msup>
                   <mo>+</mo>
                   <mrow> <mn>4</mn> <mo>&InvisibleTimes;</mo>
                          <mi>x</mi> </mrow>
                   <mo>+</mo>
                   <mn>4</mn> </mrow>
      <mo>=</mo> <mn>0</mn> </mrow>
```

Scalable Vector Graphics (SVG)

- XML for 2D graphics: texts, lines, circles, splines, etc.
- CSS for their style (fill color, stroke width, etc.)
- SVG drawings can be interactive and dynamic.

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG August 1999//EN"
  "http://www.w3.org/Graphics/SVG/SVG-19990812.dtd">
<svg width="4in" height="3in">
  <desc>A sample polyline, and equivalent path</desc>
  <polyline points="20,20 50,100 200,80 70,300"/>
  <path d="M20,20 L50,100 L200,80 L70,300"/>
</svg>
```

Style sheets

Sets of presentation rules (for screens, printers, audio devices).

	CSS	XSL
Can be used with HTML?	yes	no
Can be used with XML?	yes	yes
Transformation language?	no	yes
Syntax	CSS	XML

CSS2 (1998) = CSS1 (1996) + media-specific style sheets
(e.g. printers and aural devices), downloadable fonts,
element positioning and tables.

```
<html>
<head><title>title</title>
  <link rel=stylesheet type="text/css"
        href="http://style.com/aStyleSheet" title="aStyle">
  <STYLE type="text/css">
     @import url(http://style.com/aBasicStyleSheet);
     BODY { color: black;  text-align: justify;
            background: url(http://image.com/aTexture.gif) white;
          }
     H1, H2, H3 { color: blue;  text-align: center; width: 20cm }
     H1 { background: red;  text-transform: uppercase }
     P  { font-size: 10pt; text-indent: 5em }
  </STYLE>
</head>
<body><h1>This sentence is in blue.</h1>
  <p STYLE="color: green">This paragraph is in green.
</body>
</html>
```

```
<html><head><title>title</title>
  <STYLE type="text/css">
    DIV.foo1 { margin-left: 20%; margin-right: 20%;
               font-size: smaller
             }
    .foo2 { color: #FFFF00; font-family: arial;
            background: blue; text-decoration: blink;
          }
  </STYLE>
</head>
<body>
  <DIV CLASS="foo1"><p>Paragraph 1. <p>Paragraph 2. </DIV>
  <p CLASS="foo2">Paragraph 3.</p>
  <strong CLASS="foo2">Paragraph 4.</strong>
</body>
</html>
```

```
<STYLE type="text/css">
  ...
  @media screen, print
  { BODY        { line-height: 1.2 } }
  @media print
  { @page       { margin: 10% }
    UL, OL, DL { page-break-before: avoid;
                 page-break-inside: avoid }
    A:link      { color:black; text-decoration: underline }
    A:visited   { color:gray;  text-decoration: none }
  }
  @media speech
  { A:link      { voice-family: harry, male }
    A:visited   { voice-family: betty, female;
                  pitch-range: 80; pitch: x-high }
  }
```

Extensible Stylesheet Language (XSL)

A language for transforming XML documents, and an XML vocabulary for specifying formats.

```
<xsl:template match="person">
  <p><xsl:value-of select="first-name"/>
     <xsl:text> </xsl:text>
     <xsl:value-of select="surname"/>
  </p>
</xsl:template>
```

Extract of the DOM for HTML

```
interface HTMLDocument : Document
{          attribute DOMString        title;
  readonly attribute DOMString        referrer;
  readonly attribute DOMString        domain;
  readonly attribute DOMString        URL;
           attribute HTMLElement      body;
  readonly attribute HTMLCollection   images;
  readonly attribute HTMLCollection   applets;
  ...
  void open();  void close();
  void write(in DOMString text);
  void writeln(in DOMString text);
  Element  getElementById(in DOMString elementId);
  NodeList getElementsByName(in DOMString elementName);
}
```

Extract of the basic DOM Node class

```
interface Node
{ const unsigned short      ELEMENT_NODE   = 1;
  const unsigned short      ATTRIBUTE_NODE = 2;
  const unsigned short      TEXT_NODE      = 3;
  ...
  readonly attribute DOMString        nodeName;
           attribute DOMString        nodeValue;
  readonly attribute NodeList         childNodes;
  ...
  Node insertBefore (in Node newChild, in Node refChild)
      raises (DOMException);
  Node replaceChild (in Node newChild, in Node oldChild)
      raises (DOMException);
  ...
}
```