XML

Simon Law

eXtensible Markup Language

- Based on IBM's SGML
- SGML is really big
- XML is a W3C standard
- XML is a markup language

Why XML?

- XML allows you to define languages
- XML is a standardised format
- XML parsers are cheap and available
- XML can be human-readible
- XML is data descriptive
  - Document driven
  - Data driven

XML Technologies

- XML
  - Document Type Definition
  - XPath
  - XLinks
  - XPointers
- XSL Transformations
- Cascading Style Sheets
- XSL Formatting Objects

First Look

- Elements
  - Tags

<name>Simon Law</name>

<menu type="french"/>

  - Entity references

< >

&

" '

Element Names

- Elements have names
  - Begin with:
    - Alphanumeric characters
    - Underscores
  - After that:
    - Hyphens
    - Periods

Misc.

- CDATA sections
  - Reserved for raw character data

```
<![CDATA[
        Hello world
        . . .
]]>
```

- Comments

```
<!-- This is a comment -->
```

Processing Instructions

- Processing instructions are for parsers

```
<?robots index="yes" follow="no"?>
```

- XML has them too

```
<?xml version="1.0" encoding="UTF-8"?>
<name>
   Simon Law
</name>
```

Document Type Definition

- Element definition

```
<!ELEMENT person      (name, profession*)>
<!ELEMENT name        (first, middle?, last)>
<!ELEMENT first       (#PCDATA)>
<!ELEMENT middle      (#PCDATA)>
<!ELEMENT last        (#PCDATA)>
<!ELEMENT profession (#PCDATA)>
```

Element Attributes

- Element Attributes

```
<!ELEMENT image EMPTY>
<!ATTLIST image source CDATA #REQUIRED
                    width   CDATA #IMPLIED
>
```

- Entity references

```
<!ENTITY csc "computer science club">

&csc;
```

XSL Transformations

- Sablotron and XSLTproc
- Simple XSLT

```
<?xml version="1.0">
<xsl:stylesheet version="1.0"
xlmns:xsl="http://www.w3.org/1999/XML/Transform">
</xsl:stylesheet>
```

- By default, XSLT echos

Templates

- Templates match tags
- XML input file

```
<?xml version="1.0"?>
<people>
    <person>Alan Turing</person>
    <person>John von Neumann</person>
</people>
```

Templates

- XSLT

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
    <xsl:template match="person">
        <p>A Person</p>
    </xsl:template>
</xsl:stylesheet>
```

- Output

```
<?xml version="1.0"?>
<p>A Person</p>
<p>A Person</p>
```

Value of an Element

- You can select the values of elements

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
    <xsl:template match="person">
        <p><xsl:value-of select="."></p>
    </xsl:template>
</xsl:stylesheet>
```

- Output

```
<?xml version="1.0"?>
<p>Alan Turing</p>
<p>John von Neumann</p>
```

XPath

- XPath selects elements
- Root path
  <xsl:template match="/" />
- Child elements
  <xsl:value-of select="name" />
- Attributes
  <xsl:value-of select="@version" />

XPath

- comment()
- text()
- processing-instruction()

```
<xsl:template match="comment()">
    <i>Comment deleted</i>
</xsl:template>
```

XPath

- You can also select relative paths
  - Current
    <xsl:value-of select="." />
  - Parent
    <xsl:value-of select=".." />
  - All descendents
    <xsl:value-of select="//name" />
- Predicates

```
<xsl:template match="//name[.='Alan Turing']" />
<xsl:template match="//html[@version&lt;2] />
```

XLinks

- XLinks defines a one-way connexion
- Example:

```
<novel xmlns:xlink="http://www.w3.org/1999/xlink"
    xlink:type="simple"
    xlink:href="ftp://archive.org/pub/etext/etext93/wizoz10.txt"
    xlink:show="new"
    xlink:actuate="onRequest"
    xlink:title="The complete text"
    xlink:role="http://promo.net/pg/"
>
    <title>The Wonderful Wizard of Oz</title>
</novel>
```

XLinks

- XLinks also have other types
  - xlink:type="extended"
  - xlink:type="locator"
  - xlink:type="arc"
  - xlink:type="title"
  - xlink:type="resource"

XLinks

- <series xlink:type="extended"
  xmlns:xlink="http://www.w3.org/1999/xlink">
  <novel xlink:type="locator" xlink:label="oz1"
  xlink:href="urn:isbn:0688069444"/>
  <novel xlink:type="locator" xlink:label="oz2"
  xlink:href="urn:isbn:0192839306"/>
  <next xlink:type="arc" xlink:from="oz1" xlink:to="oz2"/>
  <previous xlink:type="arc" xlink:from="oz2" xlink:to="oz1"/>
  <author xlink:type="resource" xlink:label="baum">L. Frank Baum</author>
  <book xlink:type="arc" xlink:from="baum" xlink:to="oz1"/>
  <book xlink:type="arc" xlink:from="baum" xlink:to="oz2"/>
  <publisher xlink:type="title">
  The Kansas Centential Edition
  1999
  </publisher>
  </series>

XPointer

- XPointers are XPaths
- XPointers are used to locate points in XML documents
- Examples:
  xpointer(/)
  xpointer(//first-name)
  xpointer(//first-name/comment())
  xpointer(//name[.="Alan Turing"])

XPointer

- Use it in URLs

```
<a href=
    "http://www.ibiblio.org/xml/people.xml#xpointer
(//name[position()=1])"
>
    The first person listed
</a>
```