

XML FAQ for AGSML

- HTML and XML: what's the difference?
- XML syntax.
- Using XML.
- Associated XML technologies.

©Dan Cornford 2003,2004.

HTML and XML: what's the difference?

- All familiar with HTML (Hyper-Text Mark-up Language).
- Backbone of the web. Describes **content** and **rendering** (CSS provide some separation) of web pages.
- It has many problems, including the lack of structure / meaning and its static nature (unless external technologies used).
- W3C which oversees public domain development of the web has proposed XML as the successor to HTML to mediate these problems.
- XML: eXtensible Mark-up Language but probably should be called eXtensible Meta Language, since it describes **data** rather than layout.

XML

Key idea in XML is that rather than describing the whole page as in HTML we describe the **data** that makes up the page. The rendering of this (as HTML or whatever) is separate.

- In HTML a page might look like:

```
<p> Message: Date received: 2002-1-7 Time received: 12:32
<p> Please update the GIS web site.
```

- In XML the corresponding **data** might look like:

```
<message dateReceived="2002-1-7" timeReceived="12:32">
  Please update the GIS web site.
</message>
```

- This is just a fragment of XML and there are other ways to describe this data which are equally valid.

XML syntax

- XML syntax is deceptively simple: you have **elements** and **attributes**:

```
<element attribute="value">
  Data goes here.
</element>
```

- It is as simple as that. So what should be attributes and what should be elements:

```
<message dateReceived="2004-7-7" timeReceived="12:32" >
  Please update the AGSML web site.
</message>
or
```

XML syntax

- ...

```
<message>
  <dateReceived>2004-7-7</dateReceived>
  <timeReceived>12:32</timeReceived>
  <details>Please update the GIS web site.</details>
</message>
```

- There are no fixed rules. Elements can have children, but attributes can only take one value – context should dictate what is sensible.
- XML must be **well formed**: properly nested and closed.

Using XML

- This is all very well but XML looks rather abstract. How can I do anything with the data?
- XML is clearly human readable (good XML is!) but what we generally want is to extract information from the data.
- This requires rendering of the data. Most common rendering of XML is using HTML.
- The rules to convert XML to HTML are generally written in XML (thus only one language is learnt) in style sheets: XSL(T) eXtensible Stylesheet Language (Transformations) is the main method. Can convert XML to HTML or XML or other formats.

XSLT

- This is very similar in function to CSS in HTML (CSS can be used to render XML).
- It is an **event-driven, rules-based** programming language written in XML.
- Basically XML + XSL → HTML (or XML, or any other form).
- XSLT is quite complex: a simple example serves to illustrate the application.
- Note that XSLT uses **namespaces**, which prevent naming collisions in XML documents - what does your **<tree>** element mean?

Example: XML storing some data

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl"
  href="http:// url/" cornford/xml/tableperson.xsl"?>
<locationList xmlns="http://seas.aston.ac.uk/namespaces/Location"
  xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"
  xsi:schemaLocation=" http:// url/" cornford/xml/person.xsd">
  <person>
    <forename>Joe</forename>
    <surname>Bloggs</surname>
    <address>Aston University, Birmingham</address>
    <postcode>B4 7ET</postcode>
    <latitude>59.2</latitude>
    <longitude>-3.1</longitude>
  </person>
  <person>
    <forename>Jim</forename>
    <surname>Bloggs</surname>
    <address>Aberdeen University, Aberdeen</address>
    <postcode>AB1 5ZK</postcode>
    <latitude>56.4</latitude>
    <longitude>-1.1</longitude>
  </person>
</locationList>
```

Example: XSLT – to translate XML → HTML

```
<?xml version="1.0"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns="http://www.w3.org/TR/REC-html40"
  version="1.0"
  xmlns:loc="http://seas.aston.ac.uk/namespaces/Location">

<xsl:output method="html"/>

<xsl:template match="/">
  <html>
  <head>
  <title>Where do they all live?</title>
  </head>
  <body>
  <h1> Where do these people live? </h1>
  <xsl:apply-templates select="loc:locationList" />
  </body>
  </html>
</xsl:template>

<xsl:template match="loc:locationList">
  <table border="2px">
  <tr>
  <th colspan="2">Name</th>
  <th>Address</th>
  <th>Postcode</th>
  <th colspan="2">Location</th>
  </tr>
  <xsl:apply-templates select="loc:person" />
  </table>
</xsl:template>
```

```
<xsl:template match="loc:person">
  <tr>
  <td><xsl:value-of select="loc:forename" /></td>
  <td><xsl:value-of select="loc:surname" /></td>
  <td><xsl:value-of select="loc:address" /></td>
  <td><xsl:value-of select="loc:postcode" /></td>
  <td><xsl:value-of select="loc:latitude" /></td>
  <td><xsl:value-of select="loc:longitude" /></td>
  </tr>
</xsl:template>
</xsl:stylesheet>
```

Example: Resulting HTML

```
<html>
  <head>
  <META http-equiv="Content-Type" content="text/html">
  <title>Where do they all live?</title>
  </head>
  <body>
  <h1> Where do these people live? </h1>
  <table border="2px">
  <tr>
  <th colspan="2">Name</th>
  <th>Address</th>
  <th>Postcode</th>
  <th colspan="2">Location</th>
  </tr>
  <tr>
  <td>Joe</td>
  <td>Bliggs</td>
  <td>Aston University, Birmingham</td>
  <td>B4 7ET</td>
  <td>49.2</td>
  <td>-3.1</td>
  </tr>
  <tr>
  <td>Jim</td>
  <td>Bliggs</td>
  <td>Aberdeen University, Aberdeen</td>
  <td>AB1 5ZK</td>
  <td>56.4</td>
  <td>-1.1</td>
  </tr>
  </table>
</body>
</html>
```

Schema

- **Schema** define what can and cannot appear in an XML document.
- They enable **automatic validation** of XML.
- XSD (XML Schema Definition) language is itself written in XML.
- Can define the **elements** and **attributes** and **restrictions** on these such as their type, whether they are required and their default values.
- Similar to DTD's (which XSD will replace) – best illustrated by an example.

Example: the associated XSD file

```
<?xml version="1.0"?>
<schema xmlns="http://www.w3.org/2000/10/XMLSchema"
  targetNamespace="http://seas.aston.ac.uk/namespaces/Location"
  xmlns:loc="http://seas.aston.ac.uk/namespaces/Location">

<complexType name="locationListType">
  <element name="person" minOccurs="0" maxOccurs="unbounded" type="loc:personType"/>
</complexType>

<element name="locationList" type="loc:locationListType" />

<complexType name="fullNameType">
  <sequence>
  <element name="forename" minOccurs="1" maxOccurs="1" type="string"/>
  <element name="surname" minOccurs="1" maxOccurs="1" type="string"/>
  </sequence>
</complexType>

<complexType name="locationType">
  <sequence>
  <element name="latitude" minOccurs="1" maxOccurs="1" type="decimal"/>
  <element name="longitude" minOccurs="1" maxOccurs="1" type="decimal"/>
  </sequence>
</complexType>

<complexType name="personType">
  <sequence>
  <element name="fullname" minOccurs="1" maxOccurs="1" type="loc:fullNameType" />
  <element name="address" minOccurs="1" maxOccurs="1" type="string"/>
  <element name="postcode" minOccurs="1" maxOccurs="1" type="string"/>
  <element name="location" minOccurs="1" maxOccurs="1" type="loc:locationType" />
  </sequence>
</complexType>
</schema>
```

Putting it all together

- **XML** - data description language (meta-language).
- **XSL(T)** - transformation language, generally to HTML but also other forms.
- **XSD** - schema definition language - define valid XML - important for extensibility.
- XML is an open standard which can be extended to particular application domains: if this is agreed across a community it provides an open framework for data exchange, and facilitates distributed applications.

A word of caution

- A lot has **not** been covered.
- XML is a developing technology – however the core is stable.
- We have not discussed Xpath and Xpointer for locating parts within and across XML documents, and the associated Xlink for linking XML documents.
- Also we have not discussed parsing. Can be done client side (IE5.0+, some Netscape) or server side e.g. SAX.

Summary

- The key points were:
 - XML is a meta language for **describing data**.
 - Makes the web **dynamic** and increases the ability to extract information from data.
 - Can be **transformed** to HTML, Scalable Vector Graphics etc, **depending on user requirements**.
 - Schema are important in defining what is permissible in XML documents: AGSML will be an XML schema.
- XML will be very important in the development of the web, and provides exciting opportunities for distributed data and application development.