XML: an overview

(camille@mandrakesoft.com) MandrakeSoft

This short presentation will make you familiar with the format XML.

1. What does XML stand for?

eXtensible Markup Language: some more precision, in reverse order:

Language

XML is a language to define other languages: it is in fact a meta-language.

Mark-up

This informs us that the structuring elements of that language will be tags, like used in HTML: a tag name enclosed in brackets: <ATag>.

eXtensible

XML is not a language (which is by definition fixed) but a meta-language. XML is a set of rules, that a language must respect in order to be XML conformant. Therefore you can define your own mark-up language, and make it to evolve at your envy.

2. Where does XML come from?

In a nutshell, XML was first developed to make SGML usable for Web purposes. SGML is a very large and complicated norm (thousands of pages). XML is kind of a subset of XML, but at the same time has more restrictive rules, so that parsers and transformers working on XML are more easy to implement.

- XML is more easy than SGML
- XML was aimed to Web applications but proved useful for plenty of other applications

XML is a W3C recommendation to which are attached other recommendations as XPath, XPointer, XLink, DOM, XSL, SVG, etc.

3. So, what is it used for?

- XML solves your problems
- XML stores your data
- XML makes the coffee
- XML gives meaning to your data

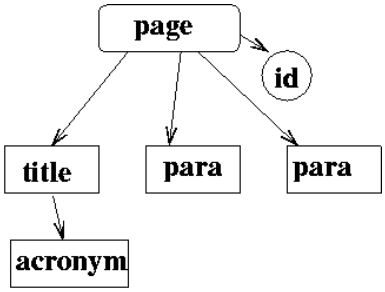
The key point is this last one. Meta-Data is your friend. That's the door open to true inter-operability and re-usability at low costs.

Another key concept is the clear separation between the data and the way it is rendered or displayed. In XML you put the data and any meta-data to identify this data. Then on a stylesheet you will define the way a particular type of data is rendered.

4. What does XML look like?

XML looks like HTML but is NOT HTML.

```
<page id="acronym">
  <title>
    What does <acronym>XML</acronym> stands for?
  </title>
  <para>
    eXtensible Mark-up Language. These are some more precision, in reverse order:
  </para>
  <para>
    This is another piece of text for the needs of that presentation.
  </para>
</page>
```



What we generally see of XML is its flat version. Indeed XML data is structured like a tree. The root element being here <page>...</page>. One must always keep in mind the tree structure of an XML document. It is another key of XML data. Here, the element <title>...</title> is not simply a <title>, but a children of <page> and the first preceding sibling of <para>...

5. Why XML is so ugly and verbose?

First of all it is important to note that XML is not meant to be edited as is by the masses. Just as a database is not meant to be updated from a command line or a report to be written editing a .doc file in a text editor.

Then verbosity is precisely the interesting point in XML. If the tags are smartly defined, anyone with a little knowledge in XML will be able to open a source file, and immediately tell what it deals about. He will be able to quickly write a transformer to translate it to any SGML-like language.

Finally, the fact that it takes a lot of place is no more a problem given the storage capacity currently available, as well as effective compressing programs.

6. Then, what do I need to write XML?

• First of all it is necessary to define your language or the tags that will be allowed in your XML. This information is stored in a DTD (Document Type Definition). You can create your own or choose an existing DTD. There exist already plenty of them from music to technical documentation.
• If data as to be stored in your format by a lot of people it is a good idea to write an interface.
• Finally you need your data to be rendered. That implies the creation of a stylesheet, that will define the transformation of your data into a common display format like HTML or PDF. You can also decide to export the data to another system like a database.

7. What does a DTD look like?

A DTD simply defines the nodes of the XML tree, Which parents accept which children, and the attributes associated to each elements.

```
<!-- Main DTD defining the structure of a presentation -->

<!-- Content elements -->
<!ENTITY % content "image|items|listing">

<!-- The root element -->
<!ELEMENT presentation (title,abstract,page*)>
    <!ELEMENT title (#PCDATA)>
    <!ELEMENT abstract (para*)>
    <!ELEMENT page ((title,abstract?),(para|section|%content;)*)>
        <!ATTLIST page id ID #REQUIRED>
        <!ELEMENT section (title,(para|%content;)*)>
        <!ELEMENT para (#PCDATA|%content;)*>
        <!ELEMENT image EMPTY>
                <!ATTLIST image name CDATA #REQUIRED>
        <!ELEMENT items (item+)>
                <!ELEMENT item (para+)>
        <!ELEMENT listing (#PCDATA)>
```

8. What are attributes?

Basically, the data in an XML document can be stored in two ways: between the start tag and the end tag, or as the value of an attribute defined inside the start tag. Here is a simple example, where we put the same information but in three different ways:

```
<message>
  <type>
    email
  </type>
  <text>
    I'll be back!
```

```
  </text>
</message>



<message type="email">
    I'll be back!
</message>



<message type="email" message="I'll be back!"/>
```

Traditionally, the second method is preferred. The element contains the data aimed to the end-user, while the attributes are used for processing the element content, adding to it meta-information.

9. What about entities?

The concept of entity is valid in writing either XML, DTD, or XSL. There are two possible entities:

• Internal: The piece of code referred is directly written in the entity declaration.
• External: allows the definition of a variable referencing an external piece of code. This entity will be used later and replace by the referred piece of code when the document is processed.

```
<!ENTITY mdk-co "<company>MandrakeSoft</company>">
<!ENTITY LM "<application>Linux-Mandrake</application>">

<para>
&LM; is made by &mdk-co;.
</para>



<!ENTITY entity-example SYSTEM "xml-entity-example.xml">

<para>
  We already saw an example of entity in the previous DTD,
  this is an example in XML:
</para>
<listing>
&entity-example;
</listing>
```

10. What are namespaces?

In order to allow data owned by two different DTDs to live together in a single XML document, it is possible to prefix the tags by a name referencing the DTD involved.

You need first to declare the namespaces that will be used in parent element beginning tag:

```
<para xmlns:xhtml='http://www.w3.org/TR/xhtml1'>
```

Then, inside this element, you will be allowed to use the "xhtml:" prefix for tags which do not belong to the presentation DTD but to the XHTML namespace:

```
<para xmlns:xhtml='http://www.w3.org/TR/xhtml1'>
This point is <xhtml:b>very</xhtml:b> important!
</para>
```

11. Resources

• World Wide Web Consortium (http://www.w3.org/) : absolutely everything you need to know on XML, and more. notably contains all official recommendations.
• ZVON (http://www.zvon.org/) : basic XML tutorials.
• StartKabel (http://www.startkabel.nl/k/xml/) : a huge bookmark for XML related sites.