

Extensible Markup Language (XML): Essentials for Climatologists

Alexander V. Besprozvannykh
CCI OPAG 1 Implementation/Coordination Team

The purpose of this material is to give basic knowledge about XML for its use in climatology. An XML language was developed to present data that can be easily understood by humans and computers. Climatological data can be represented as an XML document to simplify data exchange.

XML was designed to describe data and to focus on what data represents. Let us present some of data from a meteorological data set.

```
Station;Date;dd;ff;TTT;PPPP;RRR  
27612;2005-01-18T15:00;200;2;-5.2;1016.4;
```

This example presents coma-delimited data with a header line that defines parameters. The delimiter used is “;”. The same data can be represented by using XML.

```
<?xml version="1.0" encoding="utf-8" ?>  
<Observation>  
  <Station>27612</Station>  
  <Date>2005-01-18T15:00</Date>  
  <dd>200</dd>  
  <ff>2</ff>  
  <TTT>-5.2</TTT>  
  <PPPP>1016.4</PPPP>  
  <RRR/>  
</Observation>
```

What is XML?

- XML stands for EXtensible Markup Language;
- XML is a **markup language**;
- XML was designed to **describe data**;
- XML tags are not predefined. You must **define your own tags**;
- XML uses a **Document Type Definition (DTD)** or an **XML Schema** to define its syntax; and
- XML with a DTD or XML Schema is designed to be **self-descriptive**.

XML was created to structure, store and to send information. XML does not define what is needed to be done with data; it is just pure information wrapped in XML tags and someone

must write software to send, receive, process or display it. XML is free and extensible but **XML tags are not predefined**. Users **must "invent" their own tags**. XML allows the author to define their own tags and also their own document structure. The tags in the example above (like <Date> and <ff>) are not currently defined in any XML standard. The main benefit of XML is this: **XML is a cross-platform, software and hardware independent tool for transmitting information**. XML was not designed to display data.

Data Exchange with XML

With XML, data can be exchanged between incompatible systems. In the real world, computer systems and databases contain data in incompatible formats. One of the most time-consuming challenges for developers has been to exchange data between such systems over the Internet. Converting the data to XML can greatly reduce this complexity and create data that can be read by many different types of applications.

Storing Data in XML

XML can also be used to store data in files or in databases. Relational or native XML databases can be used to store XML. To store XML documents in relational databases you need applications to convert data from tables into XML and back. Generic applications can be used to process data in XML.

Processing Data in XML

Freely distributed applications are available to parse XML documents. International standards for XML Document Object Model (DOM) and document parsers allow for applications from different vendors and therefore simplify user applications.

XML document conversion

Documents in XML can be converted to any text document by using standard applications. A special language for transforming XML documents (XSLT) is available.

XML can be used to Create new Languages

Through defining an XML document structure, you can create your own language. Several of these languages are now available with the better known being:

- Geographic Markup Language (GML);
- Metadata language (ISO 1915) and WMO profile; and
- Meteorological XML (MeteoXml).

XML syntax

The syntax rules of XML are very simple. The rules are very easy to learn, and very easy to use. XML documents use a self-describing syntax.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<station_metadata>
  <station_id>07149</station_id>
  <name>Paris-Orly</name>
  <h>90</h>
  <lat>48.43</lat>
  <lon>2.23</lon>
</station_metadata>
```

The first line in the document - the XML declaration - defines the XML version and the character encoding used in the document. In this case the document conforms to the 1.0 specification of XML and uses the ISO-8859-1 (Latin-1/West European) character set.

The next line describes the root element of the document, i.e. providing information that "this document is a station_metadata":

```
<station_metadata>
```

The next 5 lines describe 5 child elements of the root (station_id, name, h, lat, lon):

```
<station_id>07149</station_id>
<name>Paris-Orly</name>
<h>90</h>
<lat>48.43</lat>
<lon>2.23</lon>
```

And finally the last line defines the end of the root element:

```
</station_metadata>
```

You can detect from this example that the XML document contains metadata for a station.

All XML documents must have opening and closing tags with the same name that are case sensitive. All XML elements must be properly nested. All XML documents must have a root tag. All other elements must be within this root element. All elements can have sub elements (child elements). Sub elements must be correctly nested within their parent element.

```
<root>
  <child>
    <subchild>.....</subchild>
  </child>
</root>
```

XML Attributes

XML elements can have attributes in the start tag that provide additional information about elements. Attributes must be present in name/value pairs. Where name is attribute name and value is value of this attribute.

```
<remark lang="en">Synop observation</remark >
```

In this example element remark has attribute “lang”. The “lang” attribute provides additional information about the remark element, i.e. the language used. Attribute values must always be quoted.

Comments in XML

The syntax for writing comments in XML:

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

Element Naming

XML elements must follow these naming rules:

- Names can contain letters, numbers, and other characters;
- Names must not start with a number or punctuation character;
- Names must not start with the letters xml (or XML or Xml ..); and
- Names cannot contain spaces.

Non-English letters like éòá are perfectly legal for XML element names, but watch out for problems if your software vendor doesn't support them. The ":" should not be used in element names because it is reserved to be used for something called namespaces.

XML document extension

XML documents can be easily extended to carry more information. Looking again at our earlier ‘Observation’ example, this can be extended with the addition of Td, i.e. dew point.

```
<?xml version="1.0" encoding="utf-8"?>
<Observation>
  <Station>27612</Station>
  <Date>2005-01-18T15:00</Date>
  <dd>200</dd>
  <ff>2</ff>
  <TTT>-5.2</TTT>
  <Td>-6.1</Td>
  <PPPP>1016.4</PPPP>
  <RRR/>
</Observation>
```

XML document encoding

An XML document can contain not only English characters but also, for example, Norwegian or French characters at the same time. To let your XML parser understand these characters, you should save your XML documents as Unicode and identify corresponding encoding.

```
<?xml version="1.0" encoding="utf-16"?>
```

It is possible to use encoding for specific language. For instance, encoding "windows-1251" support Russian Cyrillic characters.

XML DTD

A DTD defines the legal elements of an XML document. The purpose of a DTD is to define the legal building blocks of an XML document. It defines the document structure with a list of legal elements.

An example of DTD that corresponds to the observation XML document follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT Observation (Station, Date, dd?, ff?, TTT?, Td?, PPPP?, RRR?)>
<!ELEMENT Station (#PCDATA)>
<!ELEMENT Date (#PCDATA)>
<!ELEMENT dd (#PCDATA)>
<!ELEMENT ff (#PCDATA)>
<!ELEMENT TTT (#PCDATA)>
<!ELEMENT Td (#PCDATA)>
<!ELEMENT PPPP (#PCDATA)>
<!ELEMENT RRR (#PCDATA)>
```

This DTD document define all tags that can be included in a document. Station and Date tags are permanent but others are optional.

XML Schema

XML Schema is an XML based alternative to DTD. W3C supports an alternative to DTD called XML Schema.

XML document editors

XML is a text based markup language and XML files can be created and edited using a simple text editor like Notepad. However, when you start working with XML, you will soon find that it is better to edit XML documents using a professional XML editor. Notepad can be used for quick editing of simple HTML, CSS, and XML files. But, Notepad does not know that you are writing XML, so it will not be able to assist you. In using a simple text editor, you are more likely to create many errors, and as your XML documents grow larger you could lose control. To be able to write XML documents for all your new development projects, you will need an intelligent editor to help you write error free XML documents. Appropriate XML editors will also validate your text against a DTD or a schema, and force you to stick to a valid XML structure.

A good XML editor should be able to:

- Add closing tags to your opening tags automatically;
- Force you to write valid XML;
- Verify your XML against a DTD;
- Verify your XML against a Schema; and
- Colour code your XML syntax.

XMLSPY is a very popular XML editor and these are some of the features:

- Easy to use;
- Syntax colouring;
- Automatic tag completion;
- Automatic well-formed check;
- Easy switching between text view and grid view;
- Built in DTD and / or Schema validation;
- Built in graphical XML Schema designer;
- Powerful conversion utilities;
- Database import and export;
- Built in templates for most XML document types;
- Built in XPath analyzer; and
- Powerful project management.

Conclusions

An XML document is a very powerful facility to define a flexible data exchange format that is very important for climate data, where we often do not have predefined formats and have a very wide list of parameters.

Bibliography

1. The World Wide Web Consortium (W3C) - <http://www.w3.org>
2. Extensible Markup Language (XML) - <http://www.w3.org/XML/>
3. XML Schema - <http://www.w3.org/XML/Schema>
4. MeteoXml - http://cliware.meteo.ru/meteo/MeteoXml_en.html
5. Open source software - <http://jakarta.apache.org>