Information Management Resource Kit

Module on Digitization and Digital Libraries

UNIT 2. ELECTRONIC DOCUMENTS AND FORMAT

LESSON 5. DESCRIPTIVE MARK-UP: XML

NOTE

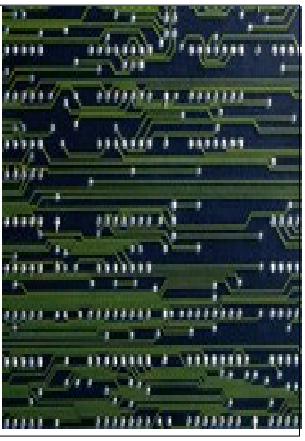
Please note that this PDF version does not have the interactive features offered through the IMARK courseware such as exercises with feedback, pop-ups, animations etc.

We recommend that you take the lesson using the interactive courseware environment, and use the PDF version for printing the lesson and to use as a reference after you have completed the course.

Learning Objectives

At the end of this lesson, you will be able to:

- understand the **features of descriptive mark-up**;
- understand the structure of a **well formed XML document**;
- understand the structure of a **Document Type Definition** (**DTD**) and **XML Schema**;
- distinguish when an XML document is **valid**; and
- know what the main **stylesheets** associated with XML documents are.

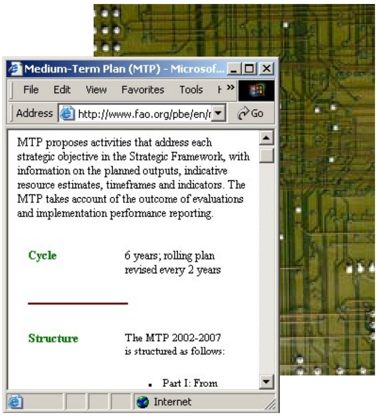


Descriptive Mark-up

Descriptive mark-up consists of codes that describe the **logical structure and semantics of a document**, usually in a way which can be interpreted by many different software applications.

The two main open standards for descriptive mark-up are **SGML** (Standard Generalized Markup Language), published as a Standard by the International Standards Organization (ISO) in 1986, and **XML** (Extensible Markup Language), which was published as a Recommendation of the World Wide Web Consortium (W3C) in 1998.

Descriptive Mark-up


Medium-Term Plan (MTP) - Microsof...
File Edit View Favorites Tools H
Address http://www.fao.org/pbe/en/r
Go
Internet

MTP proposes activities that address each strategic objective in the Strategic Framework, with information on the planned outputs, indicative resource estimates, timeframes and indicators. The MTP takes account of the outcome of evaluations and implementation performance reporting.

Cycle 6 years; rolling plan revised every 2 years

Structure The MTP 2002-2007 is structured as follows:

- Part I: From

The mark-up in an XML or SGML document specifies the structure so that the structure:

- is **separated** from the document content;
- is **logical**, not presentation-oriented;
- can be **processed** (transformed) **easily**;
- can be **verified** against a set of **rules**; and
- is **openly published**, not owned by a vendor.

Why use XML

```
<element A>

   <element B>
         <element C>
         </element C>
   </element B>

</element A>
```

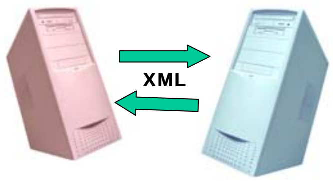
SGML and XML are very similar: when it was originally published, XML was described as a profile of SGML.

Both define the structure of a document as **a set of elements**, nested one inside the other. In both SGML and XML the mark-up consists of **tags** which indicate where each element starts and ends.

However, **XML is simpler and easier to use in web-based applications**.

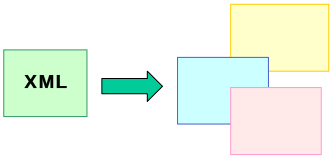
Let's look at some XML's advantages…

Why use XML



With XML, different **systems can communicate with each other**: XML is a cross-platform, software and hardware independent format for exchange of information between applications.

XML is also used as the source format from which to generate other formats (Word, PDF, HTML, etc.), since:



- it is an open, vendor neutral format;
- its mark-up captures the logical meaning of the content;
- it is well defined with public specifications; and
- it is easy to transform to other formats.

Why use XML

XML allows people and organizations to create their own mark-up languages specifically adapted to their needs and to the type of information produced.

Although everyone could create vocabularies for their own applications, in practice we usually prefer to **share our documents with other people** who have a common understanding of the descriptive mark-up In them.

The set of names used to tag the elements in an XML application is often referred to as an **XML Vocabulary**.

Experts have already created specific vocabularies for **applications**, such as mathematics or vector graphics.

They have also created vocabularies for market-specific information types such as equities research or aircraft maintenance.

XML vocabularies

XML vocabularies have been created and agreed upon by organizations that want to **share information** in specific vertical industries (such as publishing, electronics, financial services, aerospace, etc).
Examples include the Docbook standard for technical publishers, the Business Reporting Markup Language (BRML) and the AECMA series of XML standards for the aerospace industry (http://www.aecma.org).

XML standards for business and e-commerce are being developed in the ebXML initiative (www.ebxml.org) and the Universal Business Language (UBL).

XML vocabularies have also been agreed upon for specific types of application.
For example, the next generation of HTML has been defined using an XML vocabulary (xhtml).
Other examples are the Mathematical Markup Language (MathML), the Scalable Vector Graphics language (SVG) and the Chemical Mark-up Language (CML).

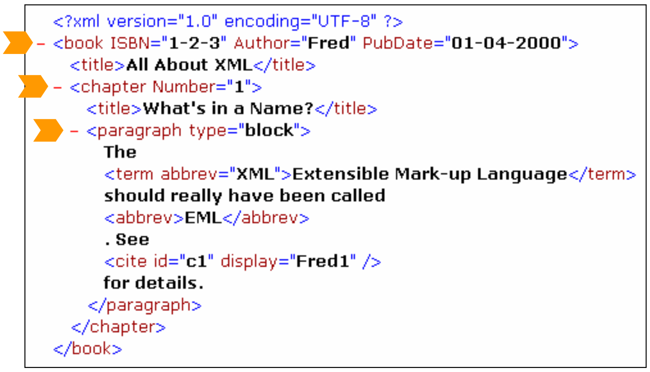
Literally thousands of XML vocabularies have been defined.

Some of the most important application vocabularies come from the World Wide Web Consortium, and an increasing number of vertical market vocabularies are being agreed upon using the standards process of OASIS – the Organisation for the Advancement of Structured Information Standards (www.oasis-open.org).

You can access the list of many of the vocabularies defined since 1998 at:
xml.coverpages.org

XML Documents

Another interesting advantage of XML is the fact that its mark-up is understandable by both humans as computers.
This is an XML document as it is displayed in the Internet Explorer web browser:



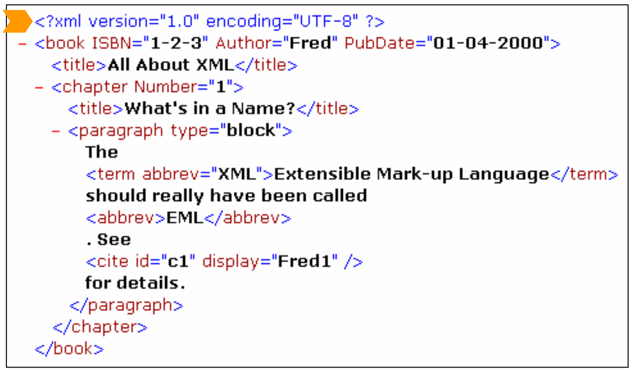
```
  <?xml version="1.0" encoding="UTF-8" ?>
- <book ISBN="1-2-3" Author="Fred" PubDate="01-04-2000">
    <title>All About XML</title>
  - <chapter Number="1">
      <title>What's in a Name?</title>
    - <paragraph type="block">
        The
        <term abbrev="XML">Extensible Mark-up Language</term>
        should really have been called
        <abbrev>EML</abbrev>
        . See
        <cite id="c1" display="Fred1" />
        for details.
      </paragraph>
    </chapter>
  </book>
```

The browser lays out the document showing the nested **tree of** its **elements**.

The **small red dashes** you can see in front of the book, chapter and paragraph elements can be clicked on to **collapse the tree** at that point.

XML Documents

The mark-up at the head of the document, enclosed in the <? … ?> tags, is called a **processing instruction**. These are not part of the document content, but are specific instructions targeted **at applications which process** the document.



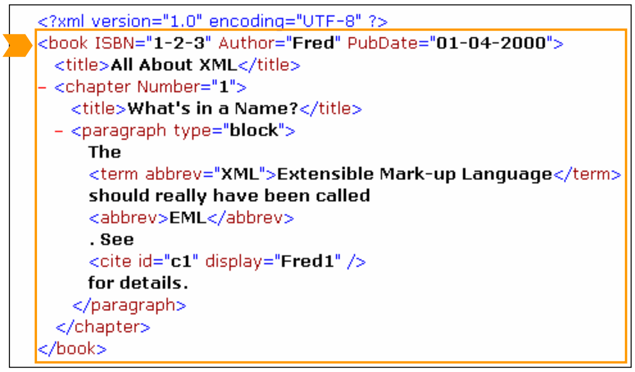
```
 <?xml version="1.0" encoding="UTF-8" ?>
- <book ISBN="1-2-3" Author="Fred" PubDate="01-04-2000">
    <title>All About XML</title>
  - <chapter Number="1">
      <title>What's in a Name?</title>
    - <paragraph type="block">
        The
        <term abbrev="XML">Extensible Mark-up Language</term>
        should really have been called
        <abbrev>EML</abbrev>
        . See
        <cite id="c1" display="Fred1" />
        for details.
      </paragraph>
    </chapter>
  </book>
```

In this case the processing instruction tells the XML processor that we are using version 1.0 of the XML language standard and the UTF-8 character encoding.

Actually, this particular processing instruction, called the **XML Declaration**, is included at the top of most XML documents.

XML Documents

The first element in our example document is the **book** element denoted by the start tag <book> and end tag </book>. Since it contains all the other mark-up and content of our document, it is the **Base Document Element**.



```
 <?xml version="1.0" encoding="UTF-8" ?>
 <book ISBN="1-2-3" Author="Fred" PubDate="01-04-2000">
    <title>All About XML</title>
 - <chapter Number="1">
      <title>What's in a Name?</title>
    - <paragraph type="block">
        The
        <term abbrev="XML">Extensible Mark-up Language</term>
        should really have been called
        <abbrev>EML</abbrev>
        . See
        <cite id="c1" display="Fred1" />
        for details.
      </paragraph>
    </chapter>
 </book>
```

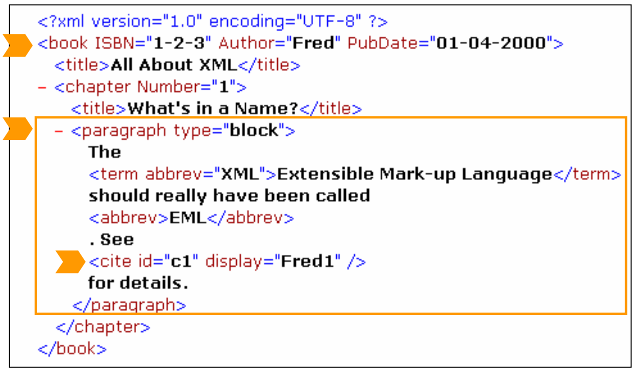
Every XML document must have such a **Base Document Element** (also called **the root**).

The Base Document Element can have any name that you want, except anything beginning with ‘xml’ which is reserved for the use of the xml standards themselves.

There are a few other rules about the characters you can use for names in XML – check the specification for details.

XML Documents

Some of the elements in our example contain attributes in their start tags, which are marked up as **name/value** pairs (e.g., ISBN=attribute name, ‘1-2-3’=attribute value).



```
<?xml version="1.0" encoding="UTF-8" ?>
<book ISBN="1-2-3" Author="Fred" PubDate="01-04-2000">
  <title>All About XML</title>
- <chapter Number="1">
    <title>What's in a Name?</title>
  - <paragraph type="block">
      The
      <term abbrev="XML">Extensible Mark-up Language</term>
      should really have been called
      <abbrev>EML</abbrev>
      . See
      <cite id="c1" display="Fred1" />
      for details.
    </paragraph>
  </chapter>
</book>
```

The **<paragraph>** element is an example of an element with **mixed content**. It contains both text and other elements mixed together.
The **<cite>** element is an example of an **empty element**. It does not have any content or/and end tag. Empty element are marked up, with a forward slash just before the closing > bracket in the start tag.

Well Formed XML Documents

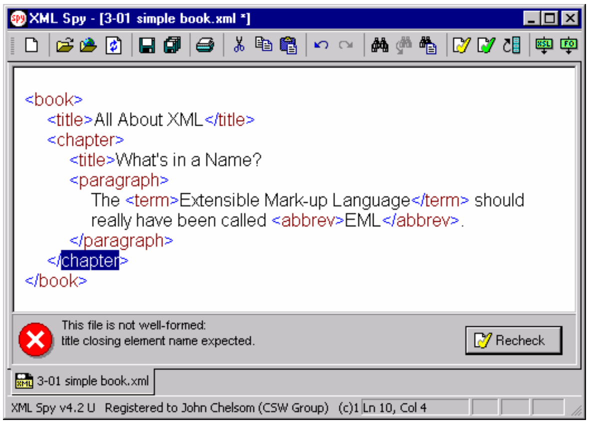
An XML document is said to be **well formed** if it follows the basic rules of **XML syntax**.

Some of the most important constraints are:

<element> **</element>** **attribute value**	Production rules including: **start and end tags** for elements must be properly nested, and **attribute values** must be quoted.
<elementA…> ↕ **</elementA>**	The **name** in an element's end-tag must **match** the **element type** in the start-tag.
<elementA **attributeX=..** **attributeX=..** **attributeY=..>**	No **attribute name** may appear more than once in the same start-tag or empty-element tag.

The ‘well-formedness constraints’ are specified in the W3C XML recommendation of 1998.

Well Formed XML Documents



```
<book>
  <title>All About XML</title>
  <chapter>
    <title>What's in a Name?
    <paragraph>
      The <term>Extensible Mark-up Language</term> should
      really have been called <abbrev>EML</abbrev>.
    </paragraph>
  </chapter>
</book>
```

This file is not well-formed:
title closing element name expected.

XML Spy v4.2 U Registered to John Chelsom (CSW Group) (c)1 Ln 10, Col 4

Software which checks whether an XML document is well formed is called a **non-validating parser**.

On the left, you can see a typical software application (an XML Editor) which has a non-validating parser. In this example, our document is not well formed since the second title element should be closed before the chapter element.

Well Formed XML Documents

Now, can you indicate which of these fragments is part of a well-formed document?

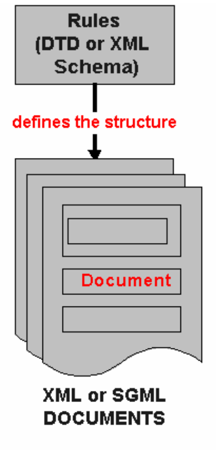
○
```
<book ISBN= “1-2-3”
Author=“Fred Pratt” Pubdate=
“02-01-2001”>
<title>XML</title>
<chapter> <title>My XML</title>
<paragraph type= “block”>This is
my XML document</chapter>
</paragraph>
</book>
```

○
```
<book ISBN= “1-2-3”
Author=“Fred Pratt” Pubdate=
“02-01-2001”>
<title>XML</title>
<chapter> <title>My XML</title>
<paragraph type= “block”>This is
my XML document</paragraph>
</chapter>
</book>
```

○
```
<book ISBN= “1-2-3”
Author=“Fred Pratt”
Author=“James Ricci” Pubdate=
“02-01-2001”>
<title>XML</title>
<chapter> <title>My
XML</title>
<paragraph type= “block”>This
is my XML
document</paragraph>
</chapter>
</book>
```

Please click on the answer of your choice

DTD and XML Schema



XML provides an application independent way of sharing data. It is therefore important to create standardized documents, that can be easily understood by other applications.

Besides following the basic rules of XML syntax, we can also use a set of **rules** which specify the logical structure that is allowable **for a particular type of document** (e.g. a book). With these rules, each of your XML files can carry a description of its own format with it.

Standard for specifying these rules in an XML document are:

- **Document Type Definition (DTD)**
- **W3C XML Schema**

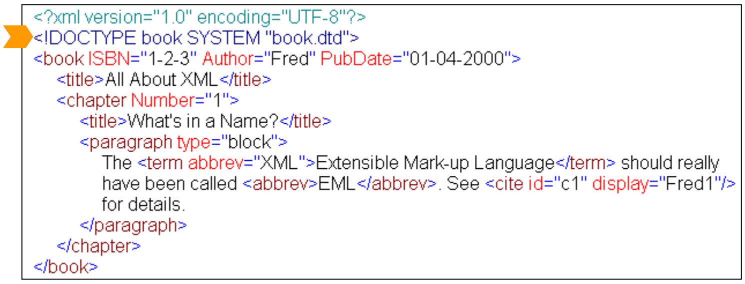
Let's look at each of them…

DTD and XML Schema

The DTD is included in the original XML recommendation published by the W3C in 1998.

It contains declarations for the **elements** and **attributes** that **can be used** to mark up the particular type of document, in our example a **book**.

To associate a DTD with an XML document instance we include a **DOCTYPE declaration** at the head of our document, as shown in our example.

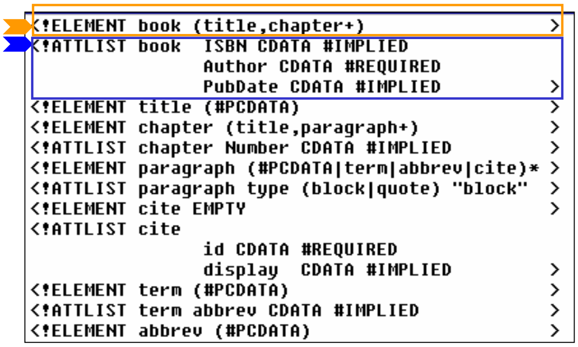


```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE book SYSTEM "book.dtd">
<book ISBN="1-2-3" Author="Fred" PubDate="01-04-2000">
    <title>All About XML</title>
    <chapter Number="1">
        <title>What's in a Name?</title>
        <paragraph type="block">
            The <term abbrev="XML">Extensible Mark-up Language</term> should really
            have been called <abbrev>EML</abbrev>. See <cite id="c1" display="Fred1"/>
            for details.
        </paragraph>
    </chapter>
</book>
```

The **SYSTEM** keyword is followed by a URI which specifies the network location (a file) where the DTD can be found.

DTD and XML Schema

Here, you can see the DTD in its plain text form opened in a text editor.
It defines what tags appear in the XML document, what attributes the tags may have and what a relationship the tags have with each other.



```
<!ELEMENT book (title,chapter+)                               >
<!ATTLIST book   ISBN CDATA #IMPLIED
                 Author CDATA #REQUIRED
                 PubDate CDATA #IMPLIED                       >
<!ELEMENT title (#PCDATA)                                     >
<!ELEMENT chapter (title,paragraph+)                          >
<!ATTLIST chapter Number CDATA #IMPLIED                       >
<!ELEMENT paragraph (#PCDATA|term|abbrev|cite)*               >
<!ATTLIST paragraph type (block|quote) "block"                >
<!ELEMENT cite EMPTY                                          >
<!ATTLIST cite
                 id CDATA #REQUIRED
                 display  CDATA #IMPLIED                      >
<!ELEMENT term (#PCDATA)                                      >
<!ATTLIST term abbrev CDATA #IMPLIED                          >
<!ELEMENT abbrev (#PCDATA)                                    >
```

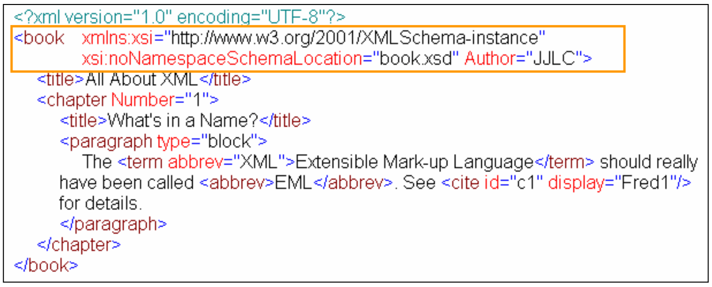
Element declarations are enclosed in the delimiters <! …> and start with the **ELEMENT** keyword, followed by the name of the element being declared and its **content model** in brackets ().

Attribute declarations are enclosed in <! …> and start with the **ATTLIST** keyword, followed by the name of the element for which attributes are being defined and sets of triples that specify an attribute **name**, its **data type** and a possible **default value**.

DTD and XML Schema

The **W3C XML Schema** fulfills the same function as DTDs did in the original specification, but **extends the capabilities of DTDs**, particularly in the areas of data typing and specification of constraints on the values of attributes and element content.

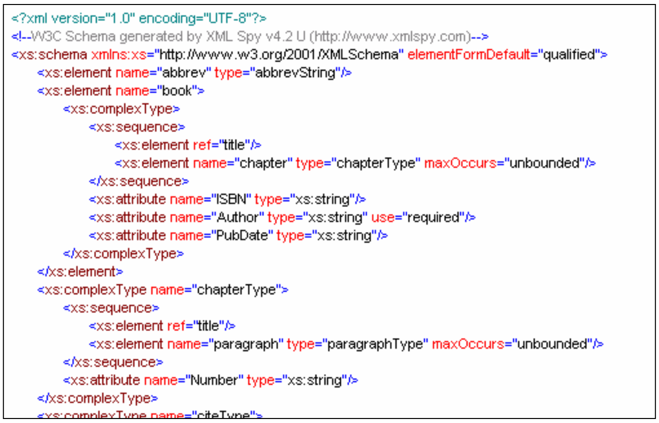
Our XML document shows how a schema can be associated with an XML document by including **two additional attributes in the start tag** of the base document element:



```
<?xml version="1.0" encoding="UTF-8"?>
<book   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:noNamespaceSchemaLocation="book.xsd" Author="JJLC">
    <title>All About XML</title>
    <chapter Number="1">
        <title>What's in a Name?</title>
        <paragraph type="block">
            The <term abbrev="XML">Extensible Mark-up Language</term> should really
        have been called <abbrev>EML</abbrev>. See <cite id="c1" display="Fred1"/>
        for details.
        </paragraph>
    </chapter>
</book>
```

DTD and XML Schema

Here's a fragment (about a quarter) of the **XML schema** that defines the structure of our simple <book> document. As you can see, it is very different from an XML DTD!



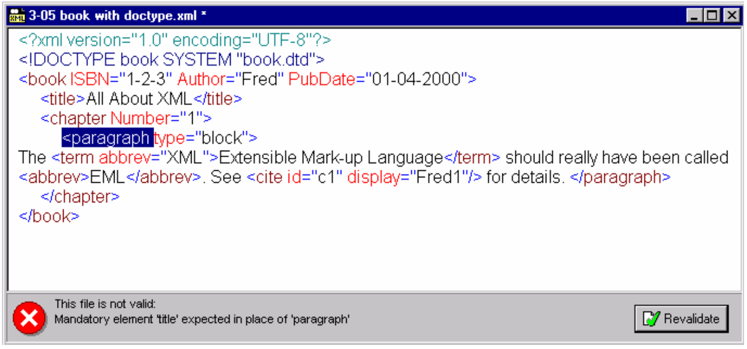
```
<?xml version="1.0" encoding="UTF-8"?>
<!--W3C Schema generated by XML Spy v4.2 U (http://www.xmlspy.com)-->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
    <xs:element name="abbrev" type="abbrevString"/>
    <xs:element name="book">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="title"/>
                <xs:element name="chapter" type="chapterType" maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:attribute name="ISBN" type="xs:string"/>
            <xs:attribute name="Author" type="xs:string" use="required"/>
            <xs:attribute name="PubDate" type="xs:string"/>
        </xs:complexType>
    </xs:element>
    <xs:complexType name="chapterType">
        <xs:sequence>
            <xs:element ref="title"/>
            <xs:element name="paragraph" type="paragraphType" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="Number" type="xs:string"/>
    </xs:complexType>
    <xs:complexType name="citeType">
```

The XML schema **is itself an XML document**, and it contains a lot of mark-up.

In fact, it can be created **by tools** such as XML Spy.

Valid XML Documents

When an XML document is processed, it is compared with the DTD to be sure it is structured correctly and all tags are used in the proper manner.
This comparison process is called **validation** and it is performed by a tool called a **validating parser**.
In the following example, the validating parser has detected that the document is **not conform** to the specified DTD (since in a book document the chapter element must be followed by the title element).



```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE book SYSTEM "book.dtd">
<book ISBN="1-2-3" Author="Fred" PubDate="01-04-2000">
    <title>All About XML</title>
    <chapter Number="1">
        <paragraph type="block">
The <term abbrev="XML">Extensible Mark-up Language</term> should really have been called
<abbrev>EML</abbrev>. See <cite id="c1" display="Fred1"/> for details. </paragraph>
    </chapter>
</book>
```

This file is not valid:
Mandatory element 'title' expected in place of 'paragraph'

Valid XML Documents

To summarize, the DTD and XML schema are...

□ rules to produce valid XML documents.

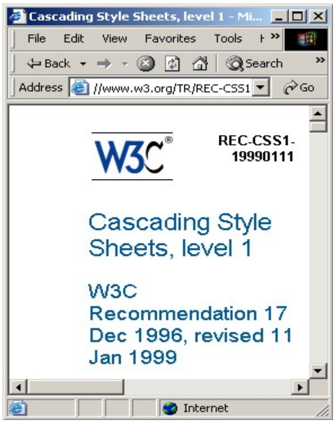
□ rules to produce well-formed XML documents.

□ verified by a non-validating parser.

□ verified by a validating parser.

Please select the options of your choice (2 or more) and press Check Answer

Cascading Style Sheets



REC-CSS1-19990111

Cascading Style Sheets, level 1

W3C Recommendation 17 Dec 1996, revised 11 Jan 1999

As you already know, descriptive mark-up describes the logical structure: it says nothing about **how a document should be displayed** in a web browser or on the printed page.

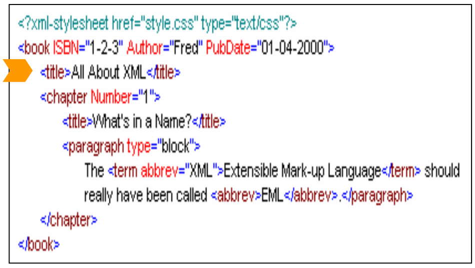
The information required to do that can be stored in a **separate stylesheet** which contains the rendering instructions.

One of the simplest ways to render an XML document directly in a **web browser** is to create a **Cascading Style Sheet (CSS)**.

Originally developed for use with HTML, CSS can be used directly with XML as well.

Some other XML applications such as editing packages may also support CSS. The first version of Cascading Style Sheets, CSS 1.0, was published as a Recommendation by the W3C in 1996 (see **www.w3.org/TR/REC-CSS1**). A subsequent version, CSS 2, was released in 1998, but it is not universally supported by software vendors. Although it contains some useful features not in CSS 1, it should be used with caution.

Cascading Style Sheets

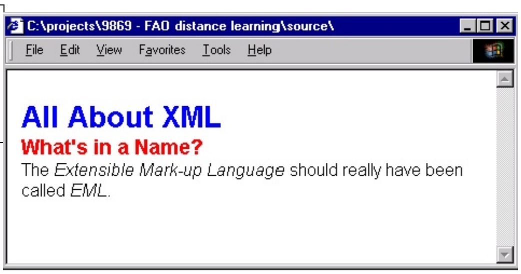


```
<?xml-stylesheet href="style.css" type="text/css"?>
<book ISBN="1-2-3" Author="Fred" PubDate="01-04-2000">
    <title>All About XML</title>
    <chapter Number="1">
        <title>What's in a Name?</title>
        <paragraph type="block">
            The <term abbrev="XML">Extensible Mark-up Language</term> should
            really have been called <abbrev>EML</abbrev>.</paragraph>
    </chapter>
</book>
```

A Cascading Style Sheet contains formatting instructions for the elements in the document.
It can be associated with an XML document by including the xml-stylesheet processing instruction in the document.

Here you have an example of an XML document, its associated style sheet and the result when the document is loaded in the IE5 web browser.

```
book {color:black ; font-family:Arial;}
book title {font-size:20pt ; font-weight:bold ; color:blue}
chapter {display:block}
chapter title {font-size:14pt ; font-weight:bold ; color:red}
paragraph {display:block}
paragraph {font-size:12pt ;  font-weight:normal}
term, abbrev {font-style:italic}
```

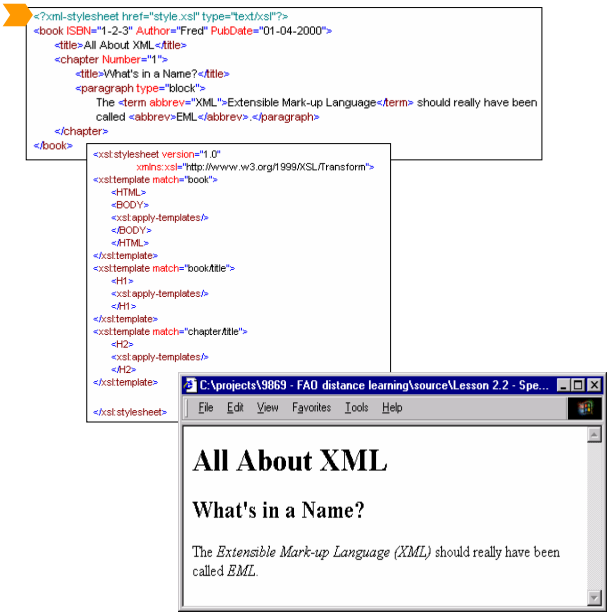

C:\projects\9869 - FAO distance learning\source\
File Edit View Favorites Tools Help

All About XML

What's in a Name?

The *Extensible Mark-up Language* should really have been called *EML*.

XSLT



```
<?xml-stylesheet href="style.xsl" type="text/xsl"?>
<book ISBN="1-2-3" Author="Fred" PubDate="01-04-2000">
    <title>All About XML</title>
    <chapter Number="1">
        <title>What's in a Name?</title>
        <paragraph type="block">
            The <term abbrev="XML">Extensible Mark-up Language</term> should really have been
            called <abbrev>EML</abbrev>.</paragraph>
    </chapter>
</book>
```

```
<xsl:stylesheet version="1.0"
        xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="book">
    <HTML>
    <BODY>
    <xsl:apply-templates/>
    </BODY>
    </HTML>
</xsl:template>
<xsl:template match="book/title">
    <H1>
    <xsl:apply-templates/>
    </H1>
</xsl:template>
<xsl:template match="chapter/title">
    <H2>
    <xsl:apply-templates/>
    </H2>
</xsl:template>

</xsl:stylesheet>
```

C:\projects\9869 - FAO distance learning\source\Lesson 2.2 - Spe...
File Edit View Favorites Tools Help

All About XML

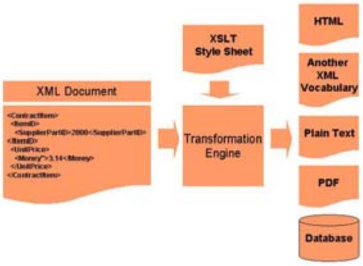
What's in a Name?

The *Extensible Mark-up Language (XML)* should really have been called *EML*.

The **Extensible Stylesheet Language for Transformations (XSLT)** is a Stylesheet language for XML.

An XSLT stylesheet is **itself an XML document**, containing templates that match against elements or attributes in the source document. Each template contains a set of rules which specify the output to be generated when the template is matched. The figure shows a simple XML document and part of its associated XSLT stylesheet.

XSLT



An XSLT processor takes as its input an XML source document and its associated stylesheet and generates the output as specified in the stylesheet.

The most common transformation is from arbitrary XML mark-up into HTML for display in a web browser, but in fact, **any output format** can be generated.

Most web browsers now have XSLT processors built-in, and so can display an XML document rendered directly with its stylesheet.

The Extensible Stylesheet Language for Transformations (XSLT) was published as a Recommendation of the W3C in 1999.
Implementations of XSLT processors have been written in many languages (Java, C++, Perl, etc) and are freely available as open source software. Two of the most widely used are called Saxon (http://saxon.sourceforge.net) and Xalan (http://xml.apache.org).

Summary

- XML, born as a profile of SGML, is an open standard for descriptive mark-up, used as exchange format between applications.

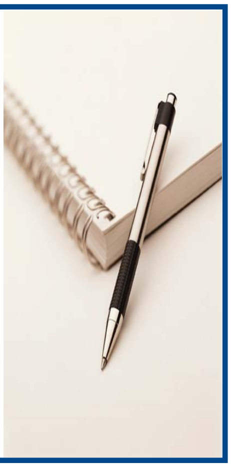
- An XML document is **well formed** if it follows the basic rules of **XML syntax**.

- **Document Type Definition** (**DTD**) and **XML Schema** are sets of **rules** which specify the logical structure that is allowable **for a particular type of document**.

- An XML document is **valid** if it complies with the rules set out in a DTD or XML Schema with which it is associated.

- **A Cascading Style Sheet (CSS)** is a **separate stylesheet** which contains simple rendering instructions for a XML document.

- **Extensible Stylesheet Language for Transformations (XSLT)** is used to create stylesheets which define transformations from XML to other XML or non-XML formats.



Exercises

The following four exercises will help you test your understanding of the concepts covered in the lesson and will provide you with feedback.

Good luck!

Exercise 1

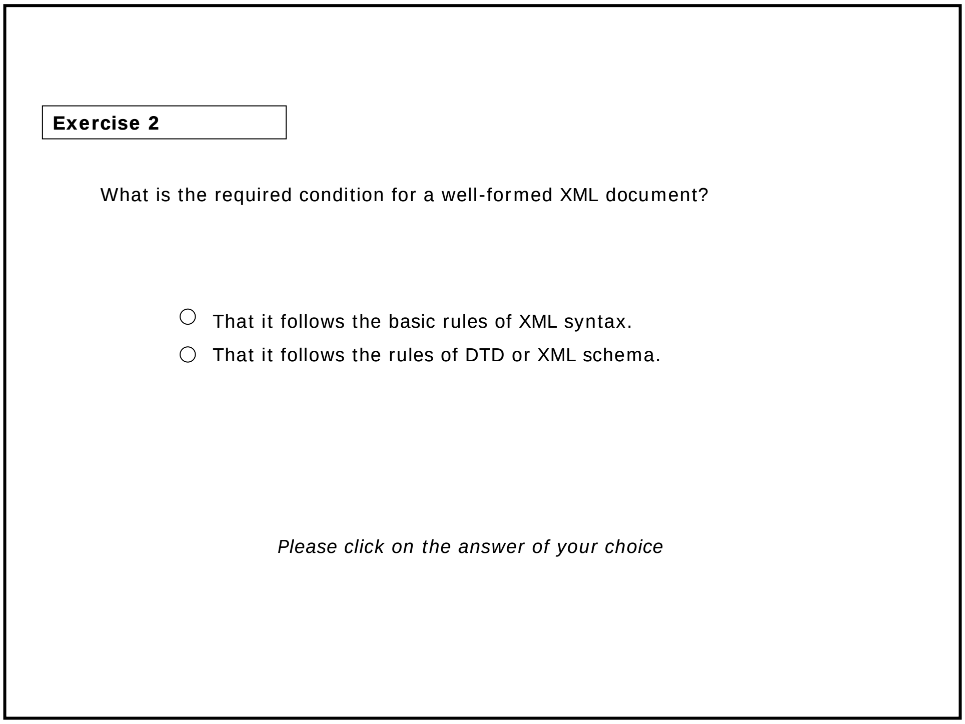
What differentiates XML from SGML ?

○ It describes a logical structure of a document.

○ It is openly published.

○ It is easy to use in web-based applications.

Please click on the answer of your choice

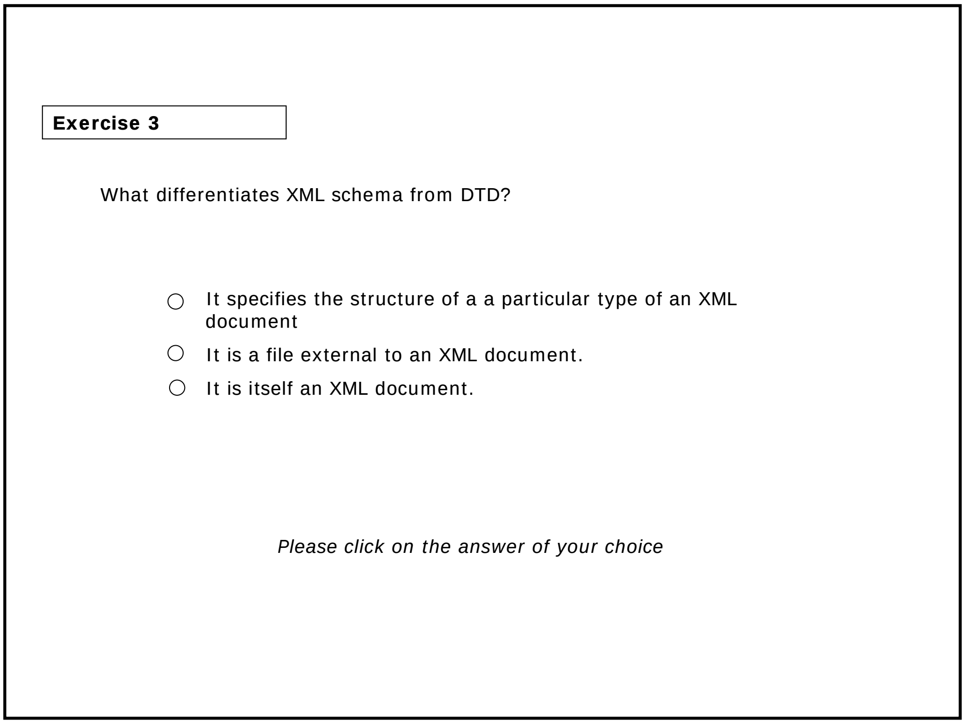


Exercise 2

What is the required condition for a well-formed XML document?

- ○ That it follows the basic rules of XML syntax.
- ○ That it follows the rules of DTD or XML schema.

Please click on the answer of your choice



Exercise 3

What differentiates XML schema from DTD?

- ○ It specifies the structure of a a particular type of an XML document
- ○ It is a file external to an XML document.
- ○ It is itself an XML document.

Please click on the answer of your choice

Exercise 4

Can you indicate the features corresponding to each kind of stylesheet?

1

Cascading Style Sheet (CSS).

Extensible Stylesheet Language for Transformations (XSLT)

a

It was originally developed for use with HTML

It was originally developed for use with XML

It is itself an XML document

It is not itself an XML document

Click each option, drag it and drop it in the corresponding box.
When you have finished, click on the confirm button.

If you want to know more...

Online Resources:

Information Processing -Text and Office Systems - Standard Generalized Markup Language (SGML), (ISO 8879:1986): (http://www.iso.org/iso/en/ISOOnline.frontpage)

World Wide Web Consortium Open information standards for the Web: (http://www.w3.org)

XML.com is an online magazine and portal to XML information: (http://www.xml.com,)

The Organization for the Advancement of Structured Information Standards (OASIS): (http://www.oasis-open.org)

An online magazine, similar to xml.com but tending to be more controversial in its views: (http://www.xmlhack.com)

ebXML - an open XML-based infrastructure enabling the interchange of electronic business information globally: (http://www.ebxml.org)

Apache Software Foundation XML project provides open source software tools for XML: (http://xml.apache.org)

Saxon and Xalan, two of the most widely used implementations of XSLT, freely available as open source software: (http://saxon.sourceforge.net/)

Saxon and Xalan, two of the most widely used implementations of XSLT, freely available as open source software: (http://xml.apache.org/#xalan)

A list of many of the vocabularies defined since 1998: http://xml.coverpages.org

Additional Reading:

Bradley, N. 2001. The XML Companion (3rd Edition). Published by Addison Wesley Professional. ISBN: 0201770598.

Ducharme, B. 2001. XSLT Quickly. Manning Publications Company. ISBN: 1930110111.