*:96 Overheads

Part 2ca: Extensible Markup Language (XML)

More about this course about Internet application protocols can be found at URL:

http://www.dsv.su.se/~jpalme/internet-course/Int-app-prot-kurs.html

Last update: 99-09-14 20.01

HTML Example

```
<h2>False Pretences</h2>
<p><b>By: </b>Margaret Yorke<br>
<b>ISBN: </b>0-312-19975-9<br>
<b>Year: </b>1999</p>
```

XML Example

```
<book><author><surname>Yorke</surname>
<given-name>Margaret</given-name></author>
<title>False Pretences</title>
<isbn>0-312-19975-9</isbn>
<year>1999</year></book>
```

The difference between HTML and XML: In XML you can yourself decide which tags to use. In HTML, you can only use the built-in tags specified in HTML. In the example above, I used the tags **<book>**, **<author>**, **<surname>**, **<given-name>**, **<title>**, **<isbn>** and **<year>**. In another application, I could have chosen other tags.

By combining of XML with style sheets, you can still get the documented printed in the same way as if you had been using HTML.

Uses of XML

(1) For transport of information between data bases.

(2) For sending of information to be displayed to a user, just like with HTML.

(3) As a rather readable format in itself (except for encoding of special characters).

(4) For encoding of network operations, as an alternative to ABNF or ASN.1.

Restrictions of XML

(5) Binary data must be either encoded as BASE64 or sent outside of the XML document (like in HTML).

(6) A rather wordy format, but compression can reduce this.

Some acronyms

Standard Generalized Markup Language (SGML)

HTML and XML are both simplifications of SGML.

Document Object Model (DOM)

DOM is an API för XML. Will be supported by version 5 web browsers.

Style sheet languages

eXtensible Style Sheet Language (XSL).

Cascading Style Sheet, level 2 (CSS2).

The same XML document can be shown in different formats, by using different style sheets.

Basics of the XML format

XML facility:	Example:
User-selected tags.	`<book>`, `<songs>`, `<position>` or whatever you need for your data.
Tags can have attributes.	`<book author="Margaret Yorke" title="False Pretences">`
Tags which have no embedded data can be closed in the opening tag.	`<book author="Margaret Yorke" title="False Pretences"/>` instead of `<book author="Margaret Yorke" title="False Pretences"></book>`
Tags can be nested.	`<book><author>Margaret Yorke</author>...<book>`
Tags must be closed.	Not correct: `<book><author>Margaret Yorke</book>`
Certain special character must be encoded.	`<book title="The &quot;queen&quot;of Sheba"/>`

XML is more strict than accepted HTML practice

HTML browsers accept many kinds of formally illegal HTML encodings. This is not allowed in XML. Examples:

```
Legal: <p>First paragraph.</p><p>Second paragraph</p>
Accepted: <p>First paragraph.<p>Second paragraph</p>

Legal: <b><i>Bold and Italics</i></b>
Accepted: <b><i>Bold and Italics</b></i>

Legal: <FONT COLOR="#FFFF66">
Accepted: <FONT COLOR=#FFFF66>
```

Tags are case-sensitive in XML

```
Illegal: <H1>Heading text</h1>
Legal: <H1>Heading text</H1>
```

White space is relevant in PCDATA, but normalized in attributes

```
<CHRISTMAS>
     X
    XXX
   XXXXX
```

```
<CHRISTMAS FATHER="Donald
Duck">
```

is identical to

```
<CHRISTMAS FATHER="Donald Duck"
```

Special Character Encoding in XML

Reserved character	Predefined entity to use instead
<	**`&gt;`**
&	**`&amp;`**
>	**`&lt;`**
'	**`&apos;`**
"	**`&quot;`**

Document Type Definition (DTD)

An XML document may be connected with a document type definition. But this is not mandatory, you can send XML data without a DTD.

The DTD describes the allowed syntax, i.e. the tags and their allowed attributes.

Example of a DTD

```
<!ELEMENT book (author+)>
<!ATTLIST book
  title CDATA #REQUIRED
  year CDATA #IMPLIED >
<!ELEMENT author (#PCDATA)>
```

Example of XML using this DTD

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE book SYSTEM
"http://www.dsv.su.se/~jpalme/internet-course/xml/book.dtd">
<book title="False Pretences" year="1999" >
<author>Margaret York</author>
</book>
```

DTD ELEMENT with free text content

Example of a DTD

```
<!ELEMENT author (#PCDATA)>
```

Example 1 of XML using this DTD

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE author SYSTEM
"http://www.dsv.su.se/~jpalme/internet-course/xml/author.dtd">
<author>Margaret York</author>
```

Example 2 of XML using this DTD

```
<author>Text containing &gt; special markup &lt;</author>
```

Example 3 of XML using this DTD

```
<author>
<![CDATA[
Text containing < special markup > like & and " and '
]]>
</author>
```

DTD ELEMENT with subelements

`(a,b)` means the element `a` followed by the element `b`.

Example of a DTD

```
<!ELEMENT author (givenname,surname)>
<!ELEMENT givenname (#PCDATA)>
<!ELEMENT surname (#PCDATA)>
```

Example 1 of XML using this DTD

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE author SYSTEM
"http://www.dsv.su.se/~jpalme/internet-course/xml/author.dtd">
<author>
<givenname>Margaret</givenname>
<surname>York</surname>
</author>
```

DTD ELEMENT with subelements

`(a*)` means that `a` is repeated 0, 1 or more times.

Example of a DTD

```
<!ELEMENT family (father,mother,child*)>
<!ELEMENT father (#PCDATA)>
<!ELEMENT mother (#PCDATA)>
<!ELEMENT child (#PCDATA)>
```

Example 1 of XML using this DTD

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE family SYSTEM
"http://www.dsv.su.se/~jpalme/internet-course/xml/family.dtd">
<family>
<father>John</father>
<mother>Margaret</mother>
<child>Eve</child>
<child>Peter</child>
</family>
```

DTD ELEMENT with subelements

`(a+)` means that `a` is repeated 1 or more times.

Example of a DTD

```
<!ELEMENT child-family (father,mother,child+)>
<!ELEMENT father (#PCDATA)>
<!ELEMENT mother (#PCDATA)>
<!ELEMENT child (#PCDATA)>
```

Example 1 of XML using this DTD

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE child-family SYSTEM
"http://www.dsv.su.se/~jpalme/internet-course/xml/child-
family.dtd">
<child-family>
<father>John</father>
<mother>Margaret</mother>
<child>Eve</child>
<child>Peter</child>
</child-family>
```

DTD ELEMENT with subelements

(a?) means that the element **a** is repeated 0 or 1 times.

Example of a DTD

```
<!ELEMENT basic-family (father?,mother?,child*)>
<!ELEMENT father (#PCDATA)>
<!ELEMENT mother (#PCDATA)>
<!ELEMENT child (#PCDATA)>
```

Example 1 of XML using this DTD

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE basic-family SYSTEM
"http://www.dsv.su.se/~jpalme/internet-course/xml/basic-
family.dtd">
<basic-family>
<father>John</father>
<child>Eve</child>
<child>Peter</child>
</basic-family>
```

DTD ELEMENT with subelements

"**|**" means either-or ",**,**" means succession.

EMPTY (without parenthesis) means no contained data.

Example of a DTD

```
<!ELEMENT operations (((get | put),uri)*)>
<!ELEMENT get EMPTY>
<!ELEMENT put EMPTY>
<!ELEMENT uri (#PCDATA)>
```

Example 1 of XML using this DTD

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE operations SYSTEM
"http://www.dsv.su.se/~jpalme/internet-course/xml/operations.d
td">
<operations>
<get/><uri>http://cmc.dsv.su.se/file1</uri>
<get/><uri>http://cmc.dsv.su.se/file2</uri>
<put/><uri>http://cmc.dsv.su.se/file3</uri>
</operations>
```

Elements versus attributes

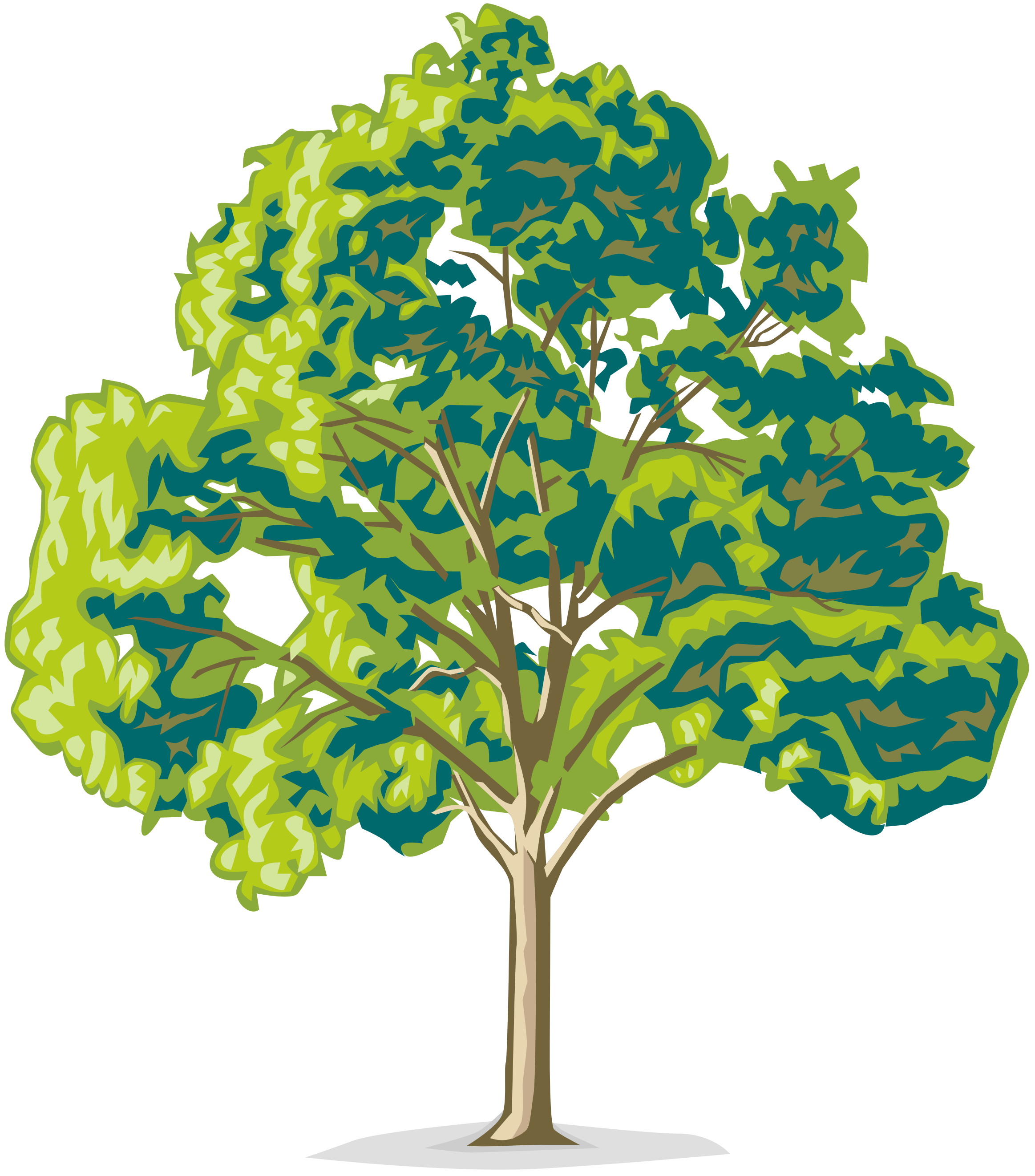
```
<book><author><surname>
Yorke</surname><given-
name>Margaret</given-
name></author></book>
```

versus

```
<book author="Margaret
Yorke">
```

Elements are like a tree with branches, each branch can split into new branches.

Attributes are like leaves or fruits, they are the end point, cannot be split further.



DTD ELEMENT with XML attributes

Example of a DTD

```
<!ELEMENT book EMPTY>
<!ATTLIST book
  title CDATA #REQUIRED
  author CDATA 'anonymous'
  weight CDATA #IMPLIED
  format (paper-back | hard-back) 'paper-back'
>
```

Example 1 of XML using this DTD

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE book SYSTEM
"http://www.dsv.su.se/~jpalme/internet-course/xml/book.dtd">
<book
  title="False Pretences"
  author="Margaret Yorke"
  format="hard-back"
/>
```

ENTITIES

Built-in character entities

Example: `&quot; &amp;`

Internal entities

You can add your own additional entity declarations to represent characters or sequences of characters. For example:

```
<!ENTITY KTH "Kungliga Tekniska Högskolan">
<DESCRIPTION>&KTH; is a technical university.</DESCRIPTION>
```

is identical to

```
<DESCRIPTION>Kungliga Tekniska Högskolan is a technical
university.</DESCRIPTION>
```

External entities

```
<!ENTITY polisvåld SYSTEM
"http://www.palme.nu/free/pv.html">

<!ENTITY comic SYSTEM
"http://www.palme.nu/comics/a-11.gif" NDATA GIF87A>
```

Use of entities to reference external DTD files

Example of the DTD book.dtd

```
<!ELEMENT book EMPTY>
<!ATTLIST book
  title CDATA #REQUIRED author CDATA 'anonymous'
  weight CDATA #IMPLIED
  format (paper-back | hard-back) 'paper-back' >
```

Example of the DTD collection.dtd

```
<!ENTITY % book SYSTEM "book.dtd">
 %book;
 <!ELEMENT collection (book+)>
 <!ATTLIST collection owner CDATA #REQUIRED >
```

Example of XML using these DTDs

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE collection SYSTEM
"http://www.dsv.su.se/~jpalme/internet-course/xml/collection.d
td">
<collection
  owner="Kungliga Biblioteket"
```

```
>
<book
  title="False Pretences"
  author="Margaret Yorke"
  format="hard-back"
/>
<book
  title="Act of Violence"
  author="Margaret Yorke"
  format="paper-back"
/>
</collection>
```

IDs in XML

Unique names can be used to refer between different places in a document.

XML example:

```
<author ref="myorke">Margaret Yorke</author>
...
<book author="myorke">False Pretences</book>
```

Based on the DTD:

```
<!ELEMENT author (#PCDATA)>
<!ATTLIST author
  ref ID #REQUIRED>
<!ELEMENT book (#PCDATA)>
<!ATTLIST book
  author IDREF #IMPLIED>
```

Attribute types:

ID = Name of this object

IDREF = One single ID reference

IDREFS = List of names separated by white space

NMTOKEN, **NMTOKENS** = Single words or lists of words separated by white space

More information about XML

The official XML standards specification (rather difficult to read):

`http://www.w3.org/TR/REC-xml`

Norman Walsh's XML tutorial:

`http://www.xml.com/xml/pub/98/10/guide1.html`

Rolf Pfeiffer's XML tutorial:

`http://www.software.ibm.com/developer/education/tutorial-prog/abstract.html`

Doug Tidwell's XML tutorial:

`http://www.software.ibm.com/developer/education/xmlintro/`

Validator of DTD/XML encodings:

`http://www.stg.brown.edu/service/xmlvalid/`