XML Basics

XML ? What is it ?

The EXtensible Markup Language (XML) is a general-purpose specification for creating custom markup languages

(Wikipedia http://en.wikipedia.org/wiki/XML)

XML ? What is it ?

XML is a framework for defining other languages:

- **There is no predefined collection of markup tags; it is possible to define new tags, tailored for our kind of information**
- **Each XML language targets its own application domain**
- **There is a common set of generic tools for processing XML documents**

We shall use XML to implement hierarchical data containers

When comparing with the relational data model XML data containers are organized in a hierarchical way rather than in a tabular way

Example of XML document

```
<?xml version="1.0" encoding ="UTF-8"?>
<!DOCTYPE recipe SYSTEM "/xml-resources/dtd/recipe.dtd">
<recipe id="117" category="dessert">
  <title>Rhubarb Cobble</title>
  <author>
    <email>Maggie.Herrick@bbs.mhv.net</email>
  </author>
  <date>Wed, 14 Jun 95</date>
  <description>Rhubarb Cobbler made with bananas</description>
  <ingredients>
    <item>
      <amount unit="cup">2 1/2</amount>
      <type>diced rhubarb</type>
    </item>
    <item>
      <amount unit="tablespoon">2</amount>
      <type>sugar</type>
    </item>
  </ingredients>
</recipe>
```

XML declaration
Document Type Declaration
Element
Attribute name
Attribute value
Start tag
End tag
Contents of element

Tag names

A name can start with any letter `a–z` (or `A–Z`), or an underscore (_)

```
<_This_is_a_very_long_element_name-but_Perfectly_acceptable>
```

The initial letter or underscore can be followed by one or more letters, digits, hyphens, underscores, full stops, combining characters, extender characters, and ignorable characters

```
<Element_12567-2>
```

Spaces and tabs are not allowed in element names; the only punctuation signs allowed are the hyphen (-) and full stop (.)

```
Not Acceptable:   <It is not allowed>
Acceptable:       <this-is-OK> <so.is.this>
```


Elements

An XML document is a text with markup tags and other meta-information

Markup tags denote *elements*

The *contents* of element

```
<email>Maggie.Herrick@bbs.mhv.net</email>
```

An *element start tag* with name `email`

A matching *element end tag*

Attributes

An *attribute* provides more information about the semantics of an element

An attribute consists of a *name* and a *value*

An element can have many attributes with unique names

An *attribute* with a name `base`

```
<amount unit="cup" base="decimal">2 1/2</amount>
```

An *attribute* with a name `unit`

An *value of attribute* enclosed by ' or "

Empty elements

An empty element

```
<bottle unit="litre"></bottle>
```

A shorthand notation for empty elements

```
<bottle unit="litre"/>
```


Special characters

Special characters can be escaped using Unicode *character references*:

`&#60;` and `<` both yield `<`

`&#38;` and `&` both yield `&`

`CDATA` *section* is an alternative way of escaping many characters

```
<![CDATA[<start> this is an incorrect element</end>]]>
```


Special elements

Instruction for a processor

Processor for which it is directed

```
<?xml-stylesheet href="book.css" type="text/css?>
```

String containing the instructions

Comment

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

Document Type Declaration (DTD, discussed later)

```
<!DOCTYPE name [markup declarations]>
```

Starting instruction

```
<?xml version="1.0"?>
```


Well formed XML

XML document must be well-formed:

start and end tags must match

element tags must be properly nested

attributes must be unique

```
<person language="EN">This is my best friend
      <first_name>Alan</first_name>
      <last_name>Smith</last_name>
            I am not sure about email
      <email>al@abc.com</email>
</person>
```

Note, that mixed contents elements are well formed

Well formed XML

A well-formed XML document contains one or more elements, and it has just one root (or document) element that contains all other elements

Well formed XML

Is it enough to be "well-formed" ?

```
<person>
      <name> Alan </name>
      <age> 42 </age>
      <email> al@abc.com </email>
</person>
```

```
<person name="Alan" age="42"
        email="al@abc.com"/>
```

```
<person name="Alan">
      <age> 42 </age>
      <email> al@abc.com </email>
</person>
```


Well formed XML

Is it enough to be "well-formed" ?

```
<hello_message>
  <language>English</language>
  <message_body>Hello world !</message_body>
</hello_message>
```

```
<hello_message language="English">
                  Hello world !</hello_message>
```

```
<address format="XLB56" language="French">
      <country>France</country>
      <city>Paris</city>
      <street>Rue Lamande</street>
      <house>25</house>
      <flat>1</flat>
</address>
```


Order of XML elements

Does order matter ?

```
<person>
      <first_name>James</first_name>
      <last_name>Bond</last_name>
</person>
```

```
<person>
      <last_name>Bond</last_name>
      <first_name>james</first_name>
</person>
```

Yes, It does !

XML references

A simple example

```
<state ID="s1">
      <scode> NSW </scode>
      <sname> New South Wales </sname>
</state>
```

```
<city ID="c1">
      <ccode> SYD </ccode>
      <cname> Sydney </cname>
      <state-of IDREF="s1"/>
</city>
```


Name conflicts

HTML table, HTML component

```
<table>
      <tr>
            <td>Apples</td>
            <td>Bananas</td>
      </tr>
</table>
```

Coffee table, a piece of furniture

```
<table>
   <category>Coffee</category>
   <width>80</width>
   <length>120</length>
</table>
```

Relational table, a database object

```
<table>
      <tuple>
                  <attribute>Name</attribute>
                  <value>John Doe</value>
                  <attribute>Date of birth</attribute>
                  <value>29 February 1980</value>
      </tuple>
</table>
```


Solving name conflicts

HTML table, HTML component

```
<html:table>
 <html:tr>
   <html:td>Apples</html:td>
   <html:td>Bananas</html:td>
 </html:tr>
</html:table>
```

Coffee table, a piece of furniture

```
<fur:table>
  <fur:category>Coffee
              </fur:category>
  <fur:width>80</fur:width>
  <fur:length>120</fur:length>
</fur:table>
```

Relational table, a database object

```
<db:table>
      <db:tuple>
                  <db:attribute>Name</db:attribute>
                  <db:value>John Doe</db:value>
                  <db:attribute>Date of birth</db:attribute>
                  <db:value>29 February 1980</db:value>
      </db:tuple>
</db:table>
```

How do we find a unique prefix ?

Creating unique prefixes

HTML table, HTML component

```
<html:table xmlns:html="http://www.w3.org/TR/html4/">
 <html:tr>
   <html:td>Apples</html:td>
   <html:td>Bananas</html:td>
 </html:tr>
</html:table>
```

Coffee table, a piece of furniture

```
<fur:table xmlns:fur="http://www.ikea.com.au/furniture">
  <fur:category>Coffee</fur:category>
  <fur:width>80</fur:width>
  <fur:length>120</fur:length>
</fur:table>
```


Creating unique prefixes

Relational table, a database object

```
<db:table xmlns:db="http://www.ansi.com/sql">
       <db:tuple>
                     <db:attribute>Name</db:attribute>
                     <db:value>John Doe</db:value>
                     <db:attribute>Date of birth</db:attribute>
                     <db:value>29 February 1980</db:value>
       </db:tuple>
</db:table>
```


Namespaces

Namespace attribute is always located in the beginning tag of an element

```
<xmltag xmlns:namespace-prefix="namespace-spec">
```

Namespace specification is an **Uniform Resource Identifier** (URI)

```
<lib:book xmlns:lib="http://www.uow.au/lib">
```

Uniform Resource Identifier identifies an Internet resource

Uniform Resource Identifier is either **Uniform Resource Locator** (URL)(see above) or **Universal Resource Name** (URN) (see below)

```
<proc:semaphore urn:dijkstra:p007,692>
```


Default namespace

Default namespace for an element becomes a namespace for all the child elements of that element

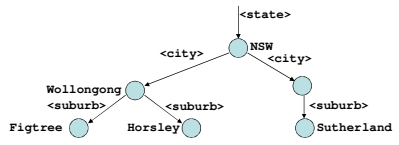
```
<table xmlns="http://www.ansi.com/sql">
       <tuple>
                     <attribute>Name</attribute>
                     <value>John Doe</value>
       </tuple>
</table>
```

It is equivalent to:

```
<db:table xmlns:db="http://www.ansi.com/sql">
       <db:tuple>
                     <db:attribute>Name</db:attribute>
                     <db:value>John Doe</db:value>
       </db:tuple>
</db:table>
```

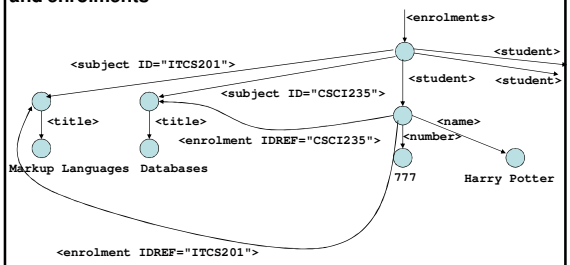

Visualization of XML documents

```
<state> NSW
       <city>Wollongong
              <suburb>Figtree</suburb>
              <suburb>Horsley</suburb>
       </city>
       <city>Sydney
       <suburb>Sutherland</suburb>
       </city>
</state>
```



Visualization of XML documents

We would like to record information about students, subjects, and enrolments



XML references

Are the hierarchical structures expressive enough ?
We would like to record information about students, subjects, and enrolments

```
<enrolments>
  <subject>
    <code>ITCS206</code>
    <title>Markup languages</title>
    <students>
        <student>
          <number>007</number>
          <name>James Bond</name>
        </student>
        <student>
          <number>777</number>
          <name>Harry Potter</name>
        </student>
    </students>
  </subject>
...
```


XML References

Are the hierarchical structures expressive enough ?
We would like to record information about students, subjects, and enrolments

```
...
  <subject>
    <code>CSCI235</code>
    <title>Databases</title>
    <students>
        <student>
          <number>001</number>
          <name>Janusz Getta</name>
        </student>
        <student>
          <number>777</number>
          <name>Harry Potter</name>
        </student>
    </students>
  </subject>
</enrolments>
```

Harry Potter's record is repeated twice !

XML References

Are the hierarchical structures expressive enough ?
We would like to record information about students, subjects, and enrolments

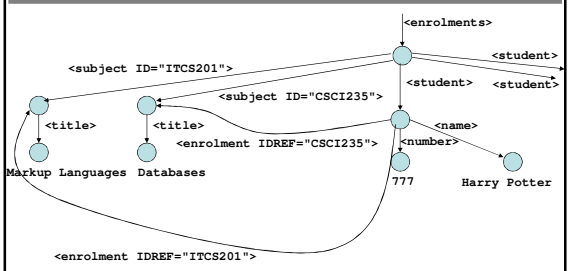
```
<enrolments>
  <subject ID="ITCS206">
    <title>Markup languages</title>
  </subject>
  <subject ID="CSCI253">
    <title>Databases</title>
  </subject>
  <students>
    <student>
      <number>007</number>
      <name>James Bond</name>
      <enrolment IDREF="ITCS206"/>
    </student>
...
```


XML References

Are the hierarchical structures expressive enough ?
We would like to record information about students, subjects, and enrolments

```
...
    <student>
      <number>777</number>
      <name>Harry Potter</name>
      <enrolment IDREF="ITCS206"/>
      <enrolment IDREF="CSCI253"/>
    </student>
    <student>
      <number>001</number>
      <name>Janusz Getta</name>
      <enrolment IDREF="CSCI253"/>
    </student>
  </students>
</enrolments>
```


XML References



How to design XML documents ?

References

Elmasri R., Navathe S., Fundamentals of Database Systems, 6th edition, chapter 12 XML: Extensible Markup Language, pp. 420-448

`http://www.uow.edu.au/~jrg/235/HOMEWORK`

8.1 How to create well formed XML documents ?