XML Data Management

2. XML Syntax

Werner Nutt

HTML

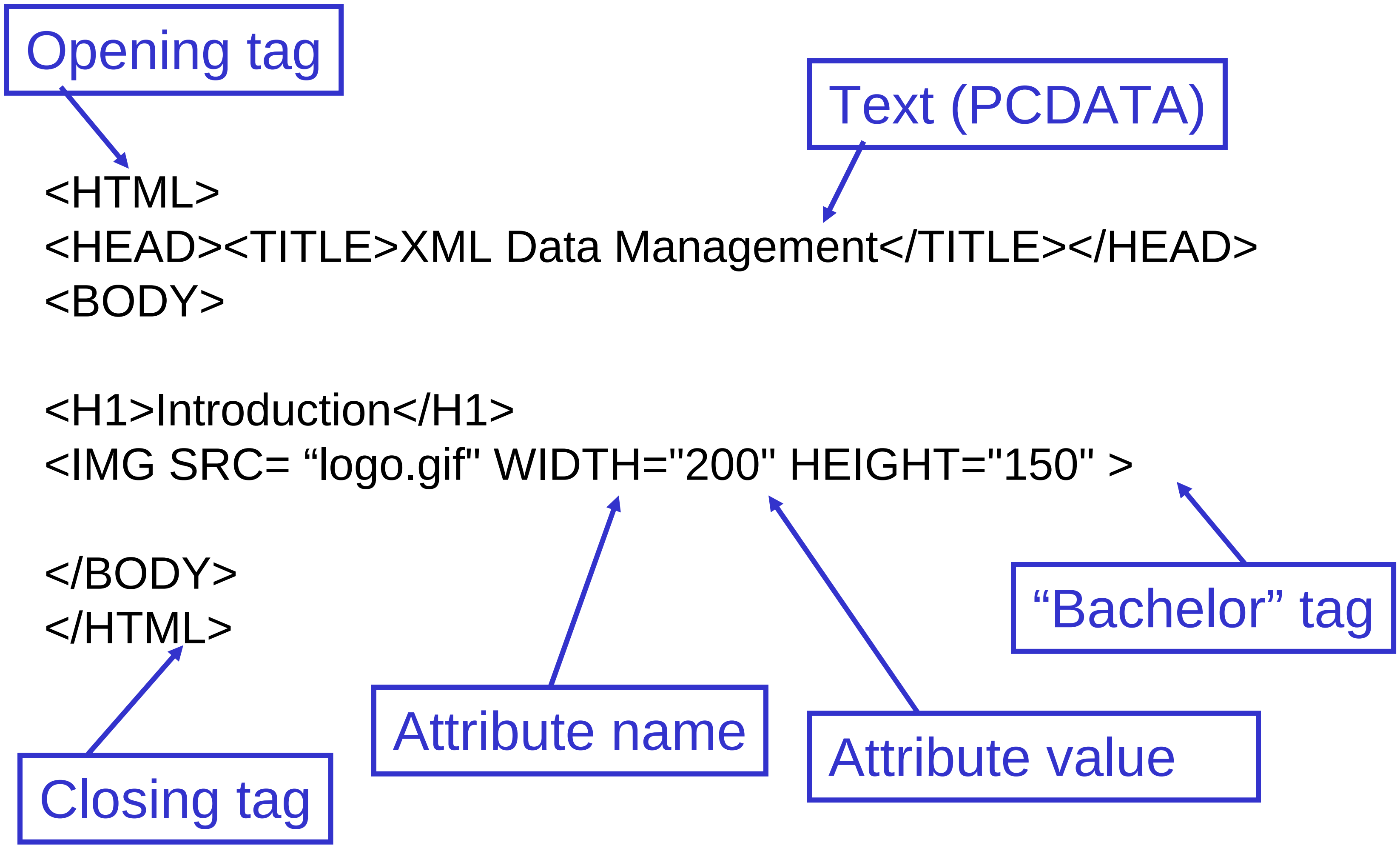
Designed for publishing hypertext on the Web

- Describes how a browser should arrange
  text, images, push-buttons, etc. on a page

- Does not convey structure

- Fixed tag set

HTML Example



```
<HTML>
<HEAD><TITLE>XML Data Management</TITLE></HEAD>
<BODY>

<H1>Introduction</H1>
<IMG SRC= “logo.gif" WIDTH="200" HEIGHT="150" >

</BODY>
</HTML>
```

XML vs. HTML

- XML and HTML are both “descendants” of SGML
  - HTML is an instance
  - XML is a subset
- HTML has specific tag and attribute names, with a specific meaning
- XML can have any tag and attribute name. These are not associated with any meaning
- HTML is used to specify visual style
- XML is used to specify meaning

XML Terminology

The segment of an XML document
between an opening and a corresponding closing tag
is called an *element*

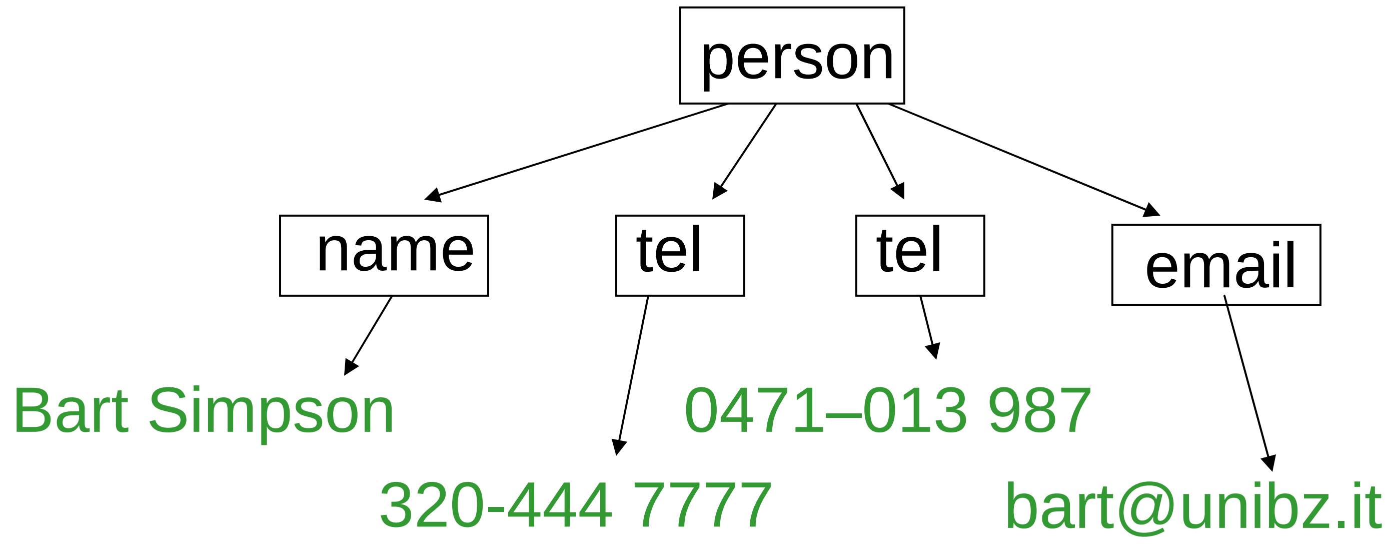
element

element,
a sub-element of the first

```
 <person>
     <name> Bart Simpson </name>
     <tel> 320-444 7777 </tel>
     <tel> 0471–013 987 </tel>
     <email> bart@unibz.it </email>
</person>
```

not an element

XML Documents are Trees



- XML documents represent trees,
  the parent-child relation is reflected by element nesting
- Some XML documents represent graphs (DAGs)
  if they contain ID and IDREF attributes

Elements Can Be Nested

```
<addresses>
        <person>
                <name> Donald Duck</name>
                <tel> 0471-828 1345 </tel>
                <tel> 0471-828 1374 </tel>
                <email> donald@eurac.edu </email>
        </person>
        <person>
                <name> Mickey Mouse </name>
                <tel> 0473-426 1142 </tel>
        </person>
</addresses>
```

Another Example

An element may contain a mixture of sub-elements and PCDATA *(= parsed character data)*

```
<airline>
    <name> British Airways </name>
    <motto>
        The world’s <dubious> favourite </dubious>
        airline
    </motto>
</airline>
```

A Complete XML Document

```
<?XML version ="1.0"  encoding="UTF-8">
<!DOCTYPE addresses SYSTEM
          "http://www.addbook.com/addresses.dtd">
<addresses>
          <person>
                    <name> Lisa Simpson</name>
                    <tel> 0471-828 1234 </tel>
                    <tel> 329-473 17 775 </tel>
                    <email> lisa@provinz.bz.it </email>
          </person>
</addresses>
```

XML Declaration, optional

Doctype Declaration, optional

Element, mandatory

An opening tag may contain *attributes*

Attributes are typically used
to describe the contents of an element

```
<entry>
  <word language = “en”> cheese </word>
  <word language = “fr”> fromage </word>
  <word language = “it”> formaggio </word>
  <meaning> A food made … </meaning>
</entry>
```

Attribute or Element?

It’s not always clear when to use attributes

Guideline:

- Use an element to nest data
- Use an attribute for “IDs”, i.e., to identify data

```
  </email>
  ...
</person>
```

```
    lisa@inf.unibz.it
  </email>
  ...
</person>
```

XML Rules

- XML is order sensitive for elements,

  i.e. the following are different:

```
<entry>
   <word language = “en”> cheese</word>
   <word language = “fr”> fromage</word>
</entry>
```

```
<entry>
   <word language = “fr”> fromage</word>
   <word language = “en”> cheese</word>
</entry>
```

- XML is case-sensitive, i.e., the following are different: <person>, <Person>, <PERSON>

XML Rules (cntd)

- Tags come in pairs <date> ...</date>
- They must be properly nested
  - Good: <date> ... <day> ... </day> ... </date>
  - Bad: <date> ... <day> ... </date> ... </day>
  - Bad: <date> ... </Date>
- There is a special shortcut for tags that have no text in between them (empty elements)
  - <person fname = “Sam” lname = “Tam” />
  - <person fname = “Sam” lname = “Tam” > </person>

XML Rules (cntd)

- There must be exactly one top-level element
  - ⇒ This element is also called the root element

```
<?xml version="1.0"?>
<Question> This is legal </Question>

<?xml version="1.0"?>
<Question> Is this legal? </Question>
<Answer> No </Answer>
```

Otherwise, the document does not represent a tree

XML Rules

- An element
  - can have several children elements with the same tag
  - Can have only one attribute with the same name

- Order of attributes does not matter, that is, the following are equivalent:

  <person fname = “Sam” lname = “Tam” />

  <person lname = “Tam” fname = “Sam” />

Well Formed Documents

- A document that satisfies all the above rules is well-formed

Relational Tables vs XML

How can one represent the contents of a table in XML?

Example:

```
Projects(title, budget, managedBy)
Employees(name, age, ssn)
```

How can one represent the contents of an XML document in a table?

Example: The address book documents

Syntax Wrap Up

So far, we have seen four constituents of an XML document

- elements
- attributes
- text
- document type definitions

… but there are a few more

More on XML Syntax

In addition, an XML document can contain

- explicit document type definitions
- comments
- processing instructions
- entity references
- CDATA sections
- namespaces

Referenced and Explicit DTDs

We have seen references to DTDs

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE math PUBLIC "-//W3C//DTD MathML 2.0//EN"
   "http://www.w3.org/Math/DTD/mathml2/mathml2.dtd">
<math> ... </math>
```

DTDs can also be referenced without a key

```
<!DOCTYPE mydoc SYSTEM "mydoc.dtd">
```

```
<!DOCTYPE mydoc SYSTEM
                   "http://www.unibz.it/mydoc.dtd">
```

Referenced and Explicit DTDs (cntd)

DTDs can also be included explicitly

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE gifts [
 <!ELEMENT gifts (Model+)>
 <!ELEMENT Model (#PCDATA)>
 <!ATTLIST Model
               group (acc | toy)  "acc"
               loc   CDATA    #IMPLIED
               id    ID    #IMPLIED  >
]>

<gifts>
  <Model group="acc" loc="Shop">P6204</Model>
  <Model group="toy">P6205</Model>
  <Model group="acc">1103</Model>
</gifts>
```

Comments, Processing Instructions, Entity References

- Comments

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

- Processing Instructions

```
<?xml-stylesheet type="text/xsl" href="books.xsl"?>
```

- CDATA Sections

Blocks of character data

```
 <![CDATA[
      Tags in XML are delimited by "<" and ">".
    ]]>
```

The content of a CDATA section is not interpreted as markup

Entity References

We cannot write

```
<math> ... <mop> < </mop> ... <math>
```

Instead we write

```
<math> ... <mop> &lt; </mop> ... <math>
```

Here, `&lt;` refers to entity `lt` , which stands for the char "<"

Other built-in entities in XML

quot	→	""""
apos	→	"'"
amp	→	"&“
gt	→	">"

Arbitrary entities can be defined in DTDs

Name Spaces: The Problem

Names become ambiguous if we start to mix data

XHTML Table

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

Table entry from a furniture catalogue

```
<table>
  <name>Coffee Table</name>
  <width>80</width>
  <length>120</length>
</table>
```

If both occur in a document, applications need to distinguish between the two kinds of tables

Naming Conflicts: Solution 1

Use prefixes, say xh for XHTML and f for furniture

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

Table entry from a furniture catalogue

```
<f:table>
  <f:name>Coffee Table</f:name>
  <f:width>80</f:width>
  <f:length>120</f:length>
</f:table>
```

How can tags with prefixes be processed by an application
that is written for the original tags?

Namespaces: Dynamic Prefixing

A namespace is a collection of element and attribute names
  XHTML namespace: **`table, tr, td, head, body, color`**, ...
  Furniture namespace: **`table, name, width, length, color`**, ...

Namespaces are identified by a URI
  XHTML: **`http://www.w3.org/1999/xhtml`**
  Furniture: **`http://www.inf.unibz.it/furniture`**

Prefixes are associated to namespaces and defined in the XML document

```
<xh:table xmlns:hx="http://www.w3.org/1999/xhtml">
  <xh:tr>
    <xh:td>Apples</xh:td><xh:td>Bananas</xh:td>
  </xh:tr>
</xh:table>
```

and similarly for the furniture

Namespace Declaration in Top Level Element

```
<root
  xmlns:xh="http://www.w3.org/1999/xhtml"
  xmlns:f="http://www.inf.unibz.it/furniture">

<xh:table>
  <xh:tr>
    <xh:td>Apples</xh:td><xh:td>Bananas</xh:td>
  </xh:tr>
</xh:table>

<f:table>
  <f:name>Coffee Table</f:name>
  <f:width>80</f:width>
  <f:length>120</f:length>
</f:table>

</root>
```

Default Namespaces

No need to place prefixes in front of names from default namespace

```
<root
  xmlns="http://www.w3.org/1999/xhtml"
  xmlns:f="http://www.inf.unibz.it/furniture">

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

  <f:table>
    <f:name>Coffee Table</f:name>
    <f:width>80</f:width>
    <f:length>120</f:length>
  </f:table>

</root>
```

Namespaces in XSLT

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0"
     xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
     xmlns:fo="http://www.w3.org/1999/XSL/Format">

  <xsl:template match="/">
    <html>
      <head>
        <title>Books</title>
      </head>
      <xsl:apply-templates select="books"/>
    </html>
  </xsl:template>
```