Introduction to XML

Definition

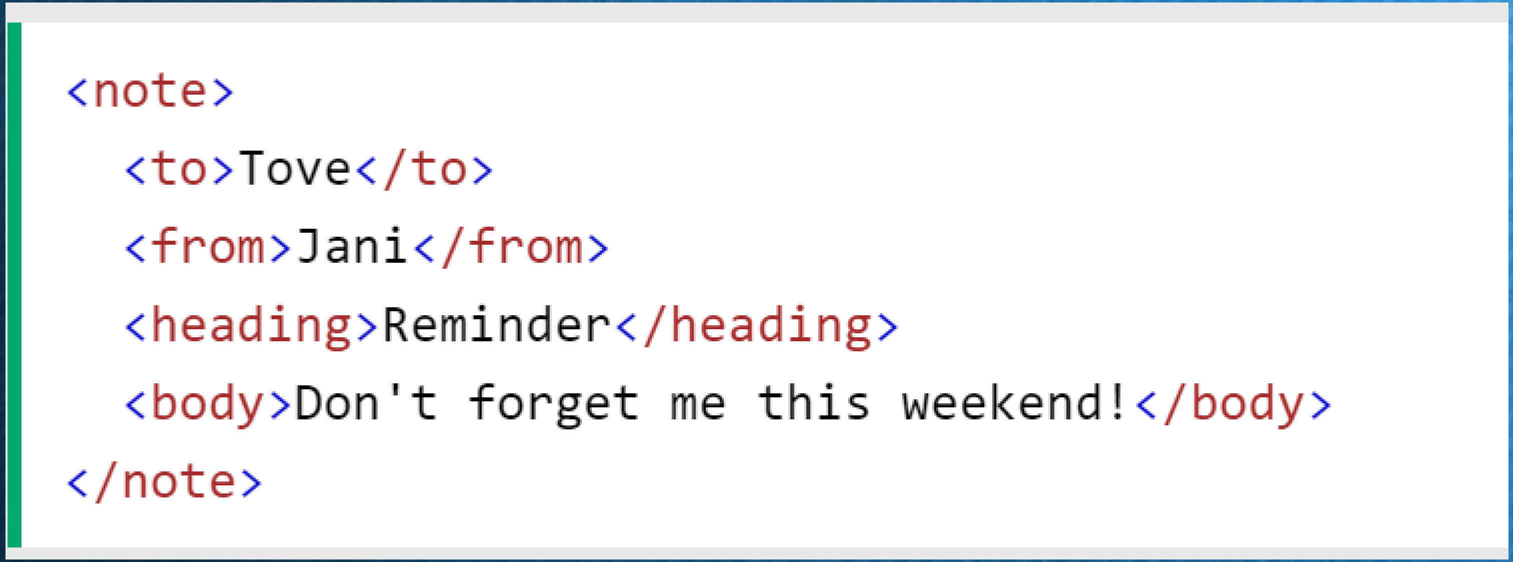
- Extensible Markup Language (XML) is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable. It is a textual data format with strong support via Unicode for different human languages.

--- W3C

Introduction

- XML stands for eXtensible Markup Language
- XML is a markup language much like HTML
- XML was designed to store and transport data
- XML was designed to be self-descriptive
- XML is a W3C Recommendation

XML Does Not DO Anything



```
<note>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

- ➢ It has sender information
- ➢ It has receiver information
- ➢ It has a heading
- ➢ It has a message body

Introduction

- XML uses a Document Type Definition (DTD) or an XML Schema to describe the data
- XMl documents are human readable
- XMl documents end with .xml   e.g. note.xml

Introduction

- The HTML Web Page was created to publish information for people
- “eyes-only” was dominant design perspective
- Hard to search
- Hard to automate processing

The Difference Between XML and HTML

XML and HTML were designed with different goals:

- XML was designed to carry data - with focus on what data is
- HTML was designed to display data - with focus on how data looks
- XML tags are not predefined like HTML tags are

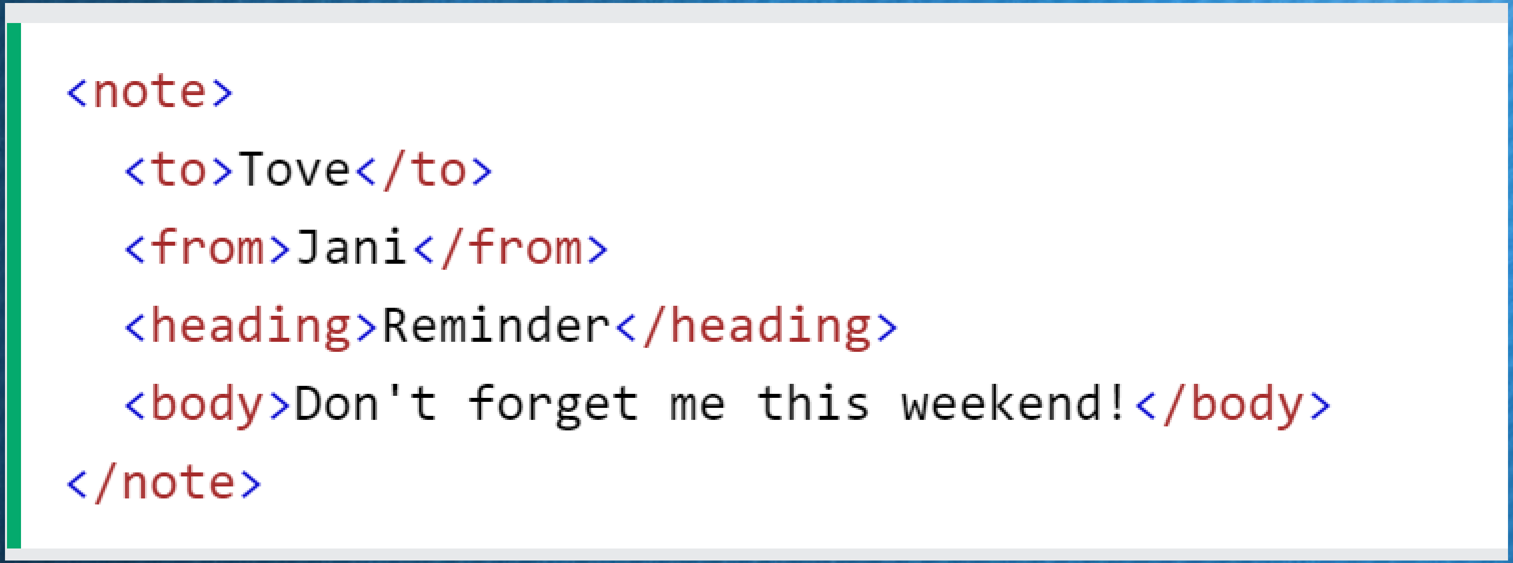
XML Does Not Use Predefined Tags

- The XML language has no predefined tags.
- The tags in the example above (like <to> and <from>) are not defined in any XML standard. These tags are "invented" by the author of the XML document.
- HTML works with predefined tags like <p>, <h1>, <table>, etc.
- With XML, the author must define both the tags and the document structure.

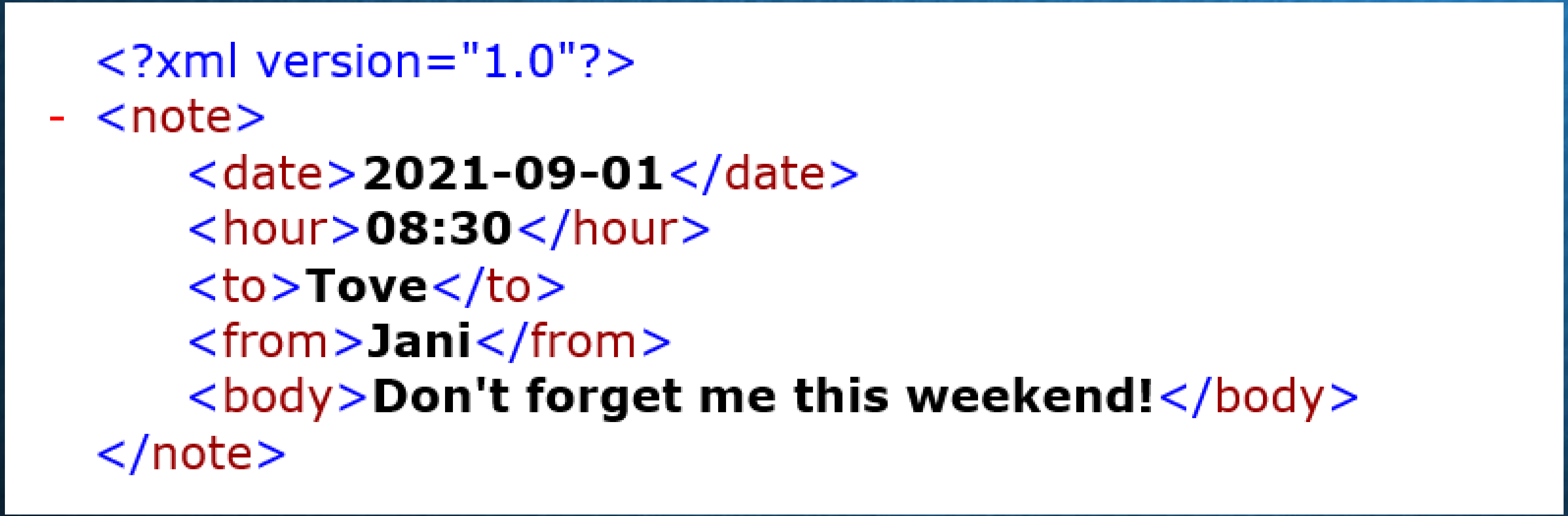
XML is Extensible

- Most XML applications will work as expected even if new data is added (or removed).
- Imagine an application designed to display the original version of note.xml (<to> <from> <heading> <body>).
- Then imagine a newer version of note.xml with added <date> and <hour> elements, and a removed <heading>.
- The way XML is constructed, older version of the application can still work:

XML is Extensible



```
<note>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```



```
  <?xml version="1.0"?>
- <note>
      <date>2021-09-01</date>
      <hour>08:30</hour>
      <to>Tove</to>
      <from>Jani</from>
      <body>Don't forget me this weekend!</body>
  </note>
```

Why XML

- XML plays an important role in many different IT systems.
- XML is often used for distributing data over the Internet.
- It is important (for all types of software developers!) to have a good understanding of XML

Why XML

- ➢Sample Catalog Entry in HTML

```
<TITLE> Laptop Computer </TITLE>
  <BODY>
    <UL>
           <LI> 14-inch MacBook Pro
           <LI>10 Core
           <LI> 16 Gb
           <LI>512 Gb
           <LI> 1.6 Kg
           <LI> €2749
      </UL>
 </BODY>
```

- How can I parse the content? E.g. price?
- Need a more flexible mechanism than HTML to interpret content.

Why XML

- Sample Catalog Entry using XML

```
<COMPUTER TYPE=“Laptop”>
    <MANUFACTURER>Apple</MANUFACTURER>
    <LINE> MacBook Pro </LINE>
    <MODEL> 14-inch </MODEL>
    <SPECIFICATIONS>
        <CHIP>10</CHIP>
        <MEMORY UNIT=“GB”>16</MEMORY>
        <STORAGE UNIT=“GB”>512</STORAGE>
        <WEIGHT UNIT=“Kg”>1.6</WEIGHT>
        <PRICE CURRENCY=“Euro”>2749</PRICE>
    </SPECIFICATIONS>
</COMPUTER>
```

Smart processing using XMl

- XML plays an important role in many different IT systems.
- XML is often used for distributing data over the Internet.
- It is important (for all types of software developers!) to have a good understanding of XML

Document exchange

- Use of XML allows companies to exchange information that can be processed automatically without human intervention e.g.
- Purchase orders
- Invoices
- Catalogues etc

Difference between HTML and XML

- XML was designed to carry data.
- XML is not a replacement for HTML.
- XML and HTML were designed with different goals:
- XML was designed to describe data and to focus on what data is.HTML was designed to display data and to focus on how data looks.
- HTML is about displaying information, while XML is about describing information.

Components of an XML Document

- Elements
  - Each element has a beginning and ending tag
    - <TAG_NAME>…</TAG_NAME>
  - Elements can be empty (<TAG_NAME />)
- Attributes
  - Describes an element; e.g. data type, data range, etc.
  - Can only appear on beginning tag
- Processing instructions
  - Encoding specification (Unicode by default)
  - Namespace declaration
  - Schema declaration

Rules For Well-Formed XML

- There must be one, and only one, root element
- Sub-elements must be properly nested
  - A tag must end within the tag in which it was started
- Attributes are optional
  - Defined by an optional schema
- Attribute values must be enclosed in “” or ‘’
- Processing instructions are optional
- XML is case-sensitive
  - <tag> and <TAG> are not the same type of element

Rules For Well-Formed XML

- Entity References
  - Some characters have a special meaning in XML.

&lt;	<	less than
&gt;	>	greater than
&amp;	&	ampersand
&apos;	'	apostrophe
&quot;	"	quotation mark

Well-Formed XML?

- No, CHILD2 and CHILD3 do not nest properly

```
<xml? version="1.0" ?>
<PARENT>
<CHILD1>This is element 1</CHILD1>
<CHILD2><CHILD3>Number
3</CHILD2></CHILD3>
</PARENT>
```

Well-Formed XML?

- No, CHILD2 and CHILD3 do not nest properly

```
<xml? Version="1.0" ?>
<PARENT>
<CHILD1>This is element 1</CHILD1>
</PARENT>
<PARENT>
<CHILD1>This is another element
1</CHILD1>
</PARENT>
```

Well-Formed XML?

- Yes

```
<xml? Version=“1.0” ?>
<PARENT>
<CHILD1>This is element 1</CHILD1>
<CHILD2/>
<CHILD3></CHILD3>
</PARENT>
```

✓

Well-Formed XML?

- Yes

```
<xml? Version=“1.0” ?>
<PARENT>
<CHILD1>This is element 1</CHILD1>
<CHILD2/>
<CHILD3></CHILD3>
</PARENT>
```

An XML Document

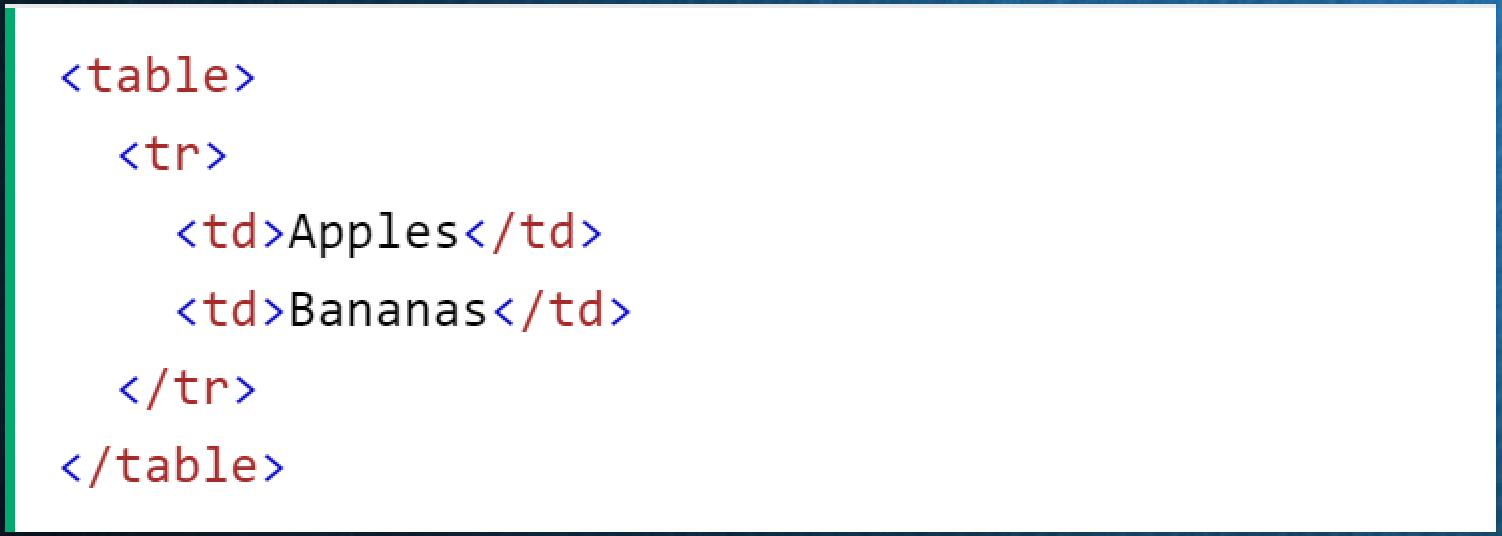
```
<?xml version='1.0'?>
<bookstore>
<book genre='autobiography' publicationdate='1981'
ISBN='1-861003-11-0'>
<title>The Autobiography of Benjamin Franklin</title>
<author>
<first-name>Benjamin</first-name>
<last-name>Franklin</last-name>
</author>
<price>8.99</price>
</book>
<book genre='novel' publicationdate='1967' ISBN='0-201-
63361-2'>
<title>The Confidence Man</title>
<author>
<first-name>Herman</first-name>
<last-name>Melville</last-name>
</author>
<price>11.99</price>
</book>
</bookstore>
```

Namespaces: Overview

- Part of XML’s extensibility
- Allow authors to differentiate between tags of the same name (using a prefix)
  - Frees author to focus on the data and decide how to best describe it
  - Allows multiple XML documents from multiple authors to be merged
- Identified by a URI (Uniform Resource Identifier)
  - When a URI is used, it does NOT have to represent a live server

Name Conflicts

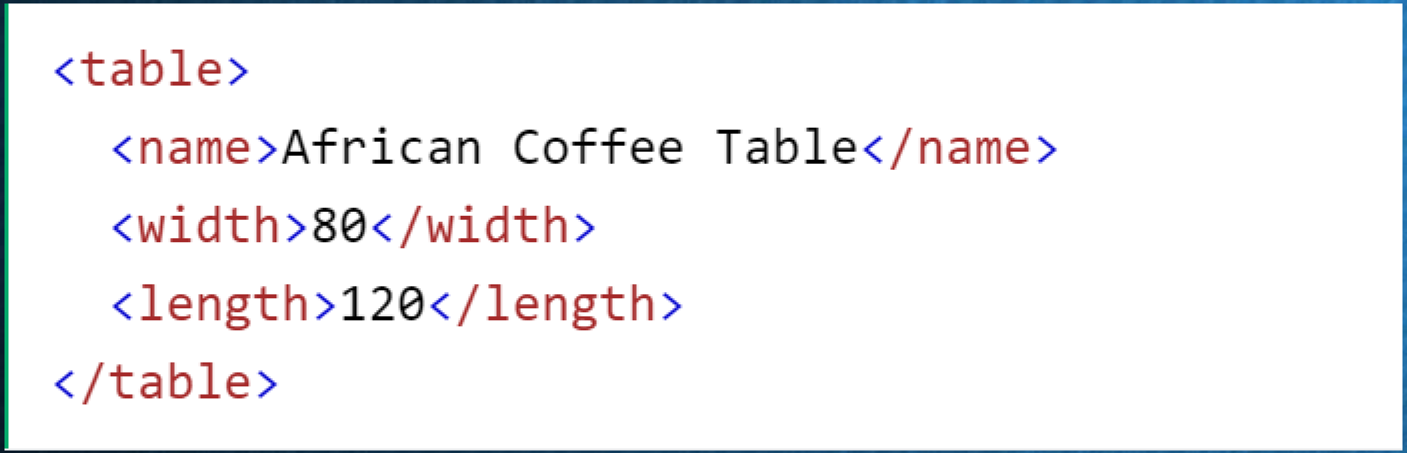
- In XML, element names are defined by the developer. This often results in a conflict when trying to mix XML documents from different XML applications.
- This XML carries HTML table information:



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

Name Conflicts

- This XML carries information about a table (a piece of furniture):

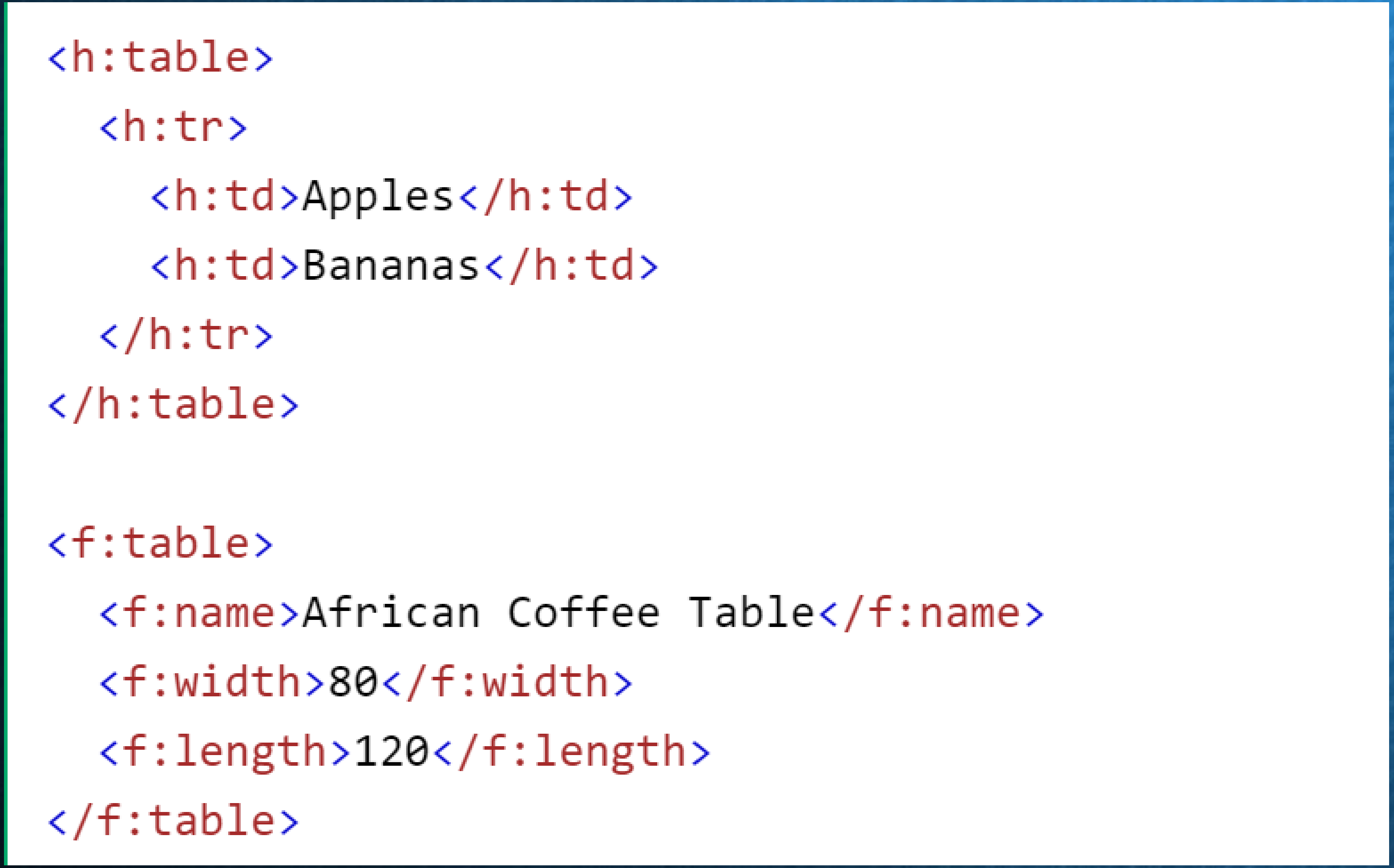


```
<table>
  <name>African Coffee Table</name>
  <width>80</width>
  <length>120</length>
</table>
```

- If these XML fragments were added together, there would be a name conflict. Both contain a <table> element, but the elements have different content and meaning.
- A user or an XML application will not know how to handle these differences.

Solving the Name Conflict Using a Prefix

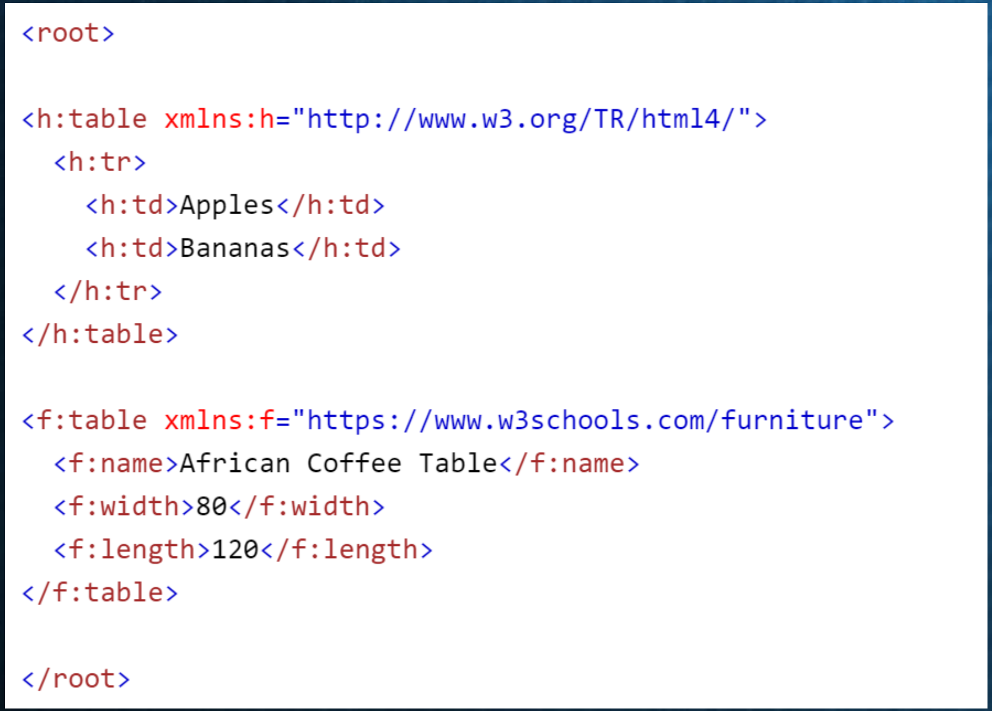
- Name conflicts in XML can easily be avoided using a name prefix.



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

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

XML Namespaces - The xmlns Attribute

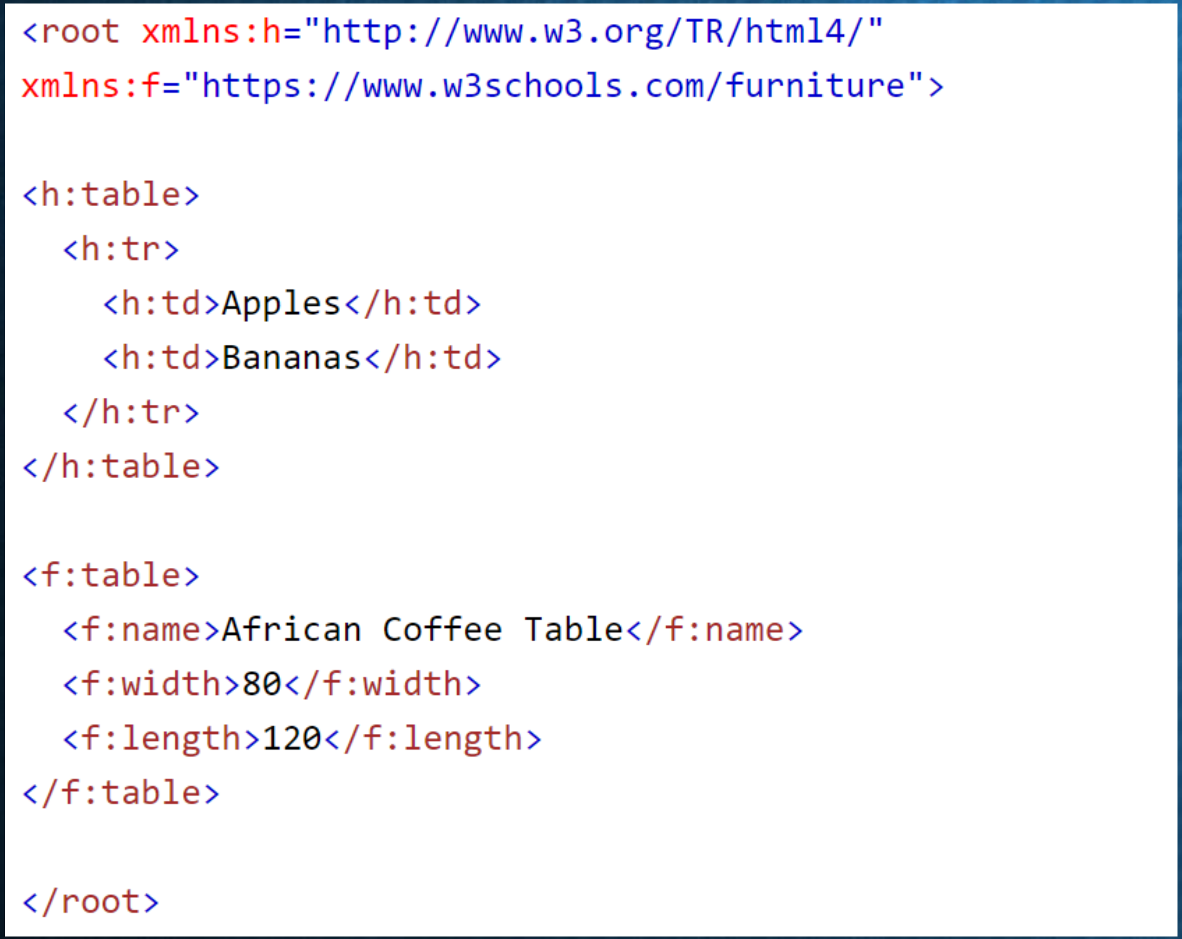
- When using prefixes in XML, a namespace for the prefix must be defined.
- The namespace can be defined by an xmlns attribute in the start tag of an element.
- The namespace declaration has the following syntax. xmlns:prefix="URI".



```xml
<root>

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

<f:table xmlns:f="https://www.w3schools.com/furniture">
  <f:name>African Coffee Table</f:name>
  <f:width>80</f:width>
  <f:length>120</f:length>
</f:table>

</root>
```

XML Namespaces - The xmlns Attribute

- Namespaces can also be declared in the XML root element:



```
<root xmlns:h="http://www.w3.org/TR/html4/"
xmlns:f="https://www.w3schools.com/furniture">

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

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

</root>
```

To manually create or write an XML document:

- Figure out what the overall document is (-> root tag)
- Figure out what the key fields of information.. (-> tag names)
- Figure out which information is data (-> tag contents)

To manually create or write an XML document:

Example: Write an XML document for the following

Address book

Name	Telephone	Address
Jeremy Cannon	0098837	22, Marlboro Ct
Naomi Murphy	992887	39, Alma Road
Sheila Zheng	999287	Apt 4, Hyde Road

<u>To manually create or write an XML document:</u>

Figure out what the overall document is (-> root tag)

--- Address book

Figure out what the key fields of information.. (-> tag names)

--- Information is Name (which can be broken into first name, surname), Telephone, Address (which can be broken down into house number, street name)

Figure out which information is data (= tag contents)

--- number, street name)

To manually create or write an XML document:

```
<addressbook>
   <person>
       <name>
              <firstname> Jeremy </first name>
              <surname> Cannon</surname>
        </name>
        <telephone> 0098837</telephone>
         <address>
             <housenumber>22</house number>
             <street> Marlboro Ct</street>
          </address>
     </person>
     <person>
         <name>
             <firstname> Jeremy </first name>
        … etc

     </person
</addressbook>
```

To check your document

- ➢Save it as .xml file
- ➢Open it in a browser – and see if any errors are produced

OR

- ➢go to a specialist XML validator for better error diagnosis

e.g. http://www.w3schools.com/dom/dom_validate.asp

Valid XML documents

- XML itself is fairly simple
- Most of the learning curve is knowing about all of the related technologies

Valid XML documents

- DTD (Document Type Definitions)
  - Not written in XML
  - No support for data types or namespaces
- XSD (XML Schema Definition)
  - Written in XML
  - Supports data types
  - Current standard recommended by W3C

Valid XML documents

- Define the “rules” (grammar) of the document
  - Data types
  - Value bounds
- A XML document that conforms to a schema is said to be valid
  - More restrictive than well-formed XML
- Define which elements are present and in what order
- Define the structural relationships of elements

Valid XML documents

- XML document:

```
<BOOK>
<TITLE>All About XML</TITLE>
<AUTHOR>Joe Developer</AUTHOR>
</BOOK>
```

- DTD schema:

```
<!DOCTYPE BOOK     [
<!ELEMENT BOOK     (TITLE+, AUTHOR) >
<!ELEMENT TITLE    (#PCDATA) >
<!ELEMENT AUTHOR   (#PCDATA) >
]>
```

Document Type Definitions

The DTD (note.DTD) for XMl document Note: is

```
<!DOCTYPE note [
 <!ELEMENT note (to,from,heading,body)>
 <!ELEMENT to (#PCDATA)>
 <!ELEMENT from (#PCDATA)>
 <!ELEMENT heading (#PCDATA)>
 <!ELEMENT body (#PCDATA)> ]>
```

Lists the document type (note) and the valid elements, and the type of content they can accept

#PCDATA means parse-able text data.

XML Schema

XML Schema is an XML based alternative to DTD.

An XML schema describes the structure of an XML document.

XML Schemas will probably be used in most Web applications as a replacement for DTDs because:

XML Schemas are supported by the W3C

XML Schemas are richer and more useful than DTDs

XML Schemas are written in XML

XML Schemas support data types

XML Schemas are extensible to future additions

XML Schemas support namespaces

Schemas: XSD Example

- XML document:

```
<CATALOG>
   <BOOK>
      <TITLE>All About XML</TITLE>
      <AUTHOR>Joe Developer</AUTHOR>
   </BOOK>
   …
</CATALOG>
```

Schemas: XSD Example

```
<xsd:schema id="NewDataSet“ targetNamespace="http://tempuri.org/schema1.xsd"
   xmlns="http://tempuri.org/schema1.xsd"
   xmlns:xsd="http://www.w3.org/1999/XMLSchema"
   xmlns:msdata="urn:schemas-microsoft-com:xml-msdata">
  <xsd:element name="book">
    <xsd:complexType content="elementOnly">
      <xsd:all>
        <xsd:element name="title" minOccurs="0" type="xsd:string"/>
        <xsd:element name="author" minOccurs="0" type="xsd:string"/>
      </xsd:all>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name=“Catalog" msdata:IsDataSet="True">
    <xsd:complexType>
      <xsd:choice maxOccurs="unbounded">
        <xsd:element ref="book"/>
      </xsd:choice>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

XML Parser

```
<!DOCTYPE html>
<html>
<body>

<h1>DT228/2 Internal Note</h1>
<div>
<b>To:</b> <span id="to"></span><br>
<b>From:</b> <span id="from"></span><br>
<b>Message:</b> <span id="message"></span>
</div>
```

XML Parser

```
<script>
var txt, parser, xmlDoc;
txt = "<note>" +
"<to>Tove</to>" +
"<from>Jani</from>" +
"<heading>Reminder</heading>" +
"<body>Don't forget me this weekend!</body>" +
"</note>";
```

XML Parser

```
parser = new DOMParser();
xmlDoc = parser.parseFromString(txt,"text/xml");

document.getElementById("to").innerHTML =
xmlDoc.getElementsByTagName("to")[0].childNodes[0].nod
eValue;
document.getElementById("from").innerHTML =
xmlDoc.getElementsByTagName("from")[0].childNodes[0].n
odeValue;
document.getElementById("message").innerHTML =
xmlDoc.getElementsByTagName("body")[0].childNodes[0].n
odeValue;
```

XML-Based Applications

- Microsoft SQL Server
  - Retrieve relational data as XML
  - Query XML data
  - Join XML data with existing database tables
  - Update the database via XML Updategrams
  - New XML data type in SQL 2005
- Microsoft Exchange Server
  - XML is native representation of many types of data
  - Used to enhance performance of UI scenarios (for example, Outlook Web Access (OWA))