Design of spatial data models

A brief introduction to **XML** and **GML**

XML Basics

https://www.w3.org/XML/

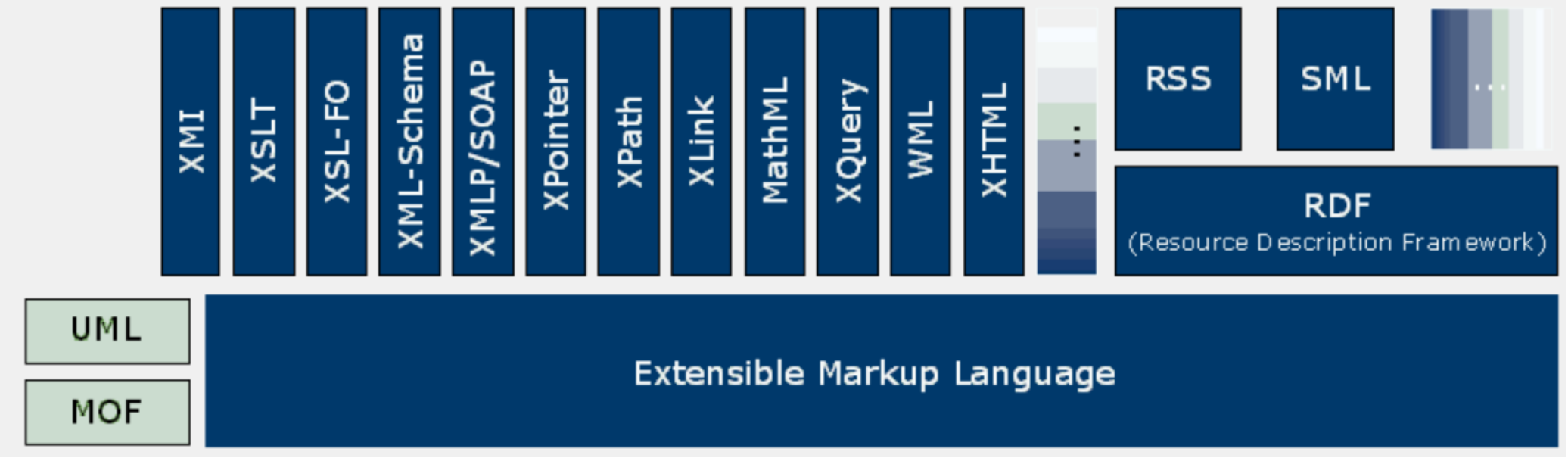
What is XML – key facts

- XML stands for E**X**tensible **M**arkup **L**anguage
- XML is a **markup language** much like HTML
- XML was designed to **describe**, **store** and **transport** data
- XML tags are not predefined. You must **define your own tags**
- XML uses a **Document Type Definition** (DTD) or an **XML Schema** to describe the structure of the data (in recent years, DTD became mostly superseded by the use of XML Schema)
- XML with a DTD or XML schema is designed to be **self-descriptive**
- XML is a W3C recommendation (this means that it is a W3C standard)
  - https://www.w3.org/XML/

An example on an XML document

```
<?xml version="1.0" encoding="UTF-8"?>
<breakfast_menu>
<food>
  <name>Belgian Waffles</name>
  <price>$5.95</price>
  <description>Our famous Belgian Waffles with plenty of real maple syrup</description>
  <calories>650</calories>
</food>
<food>
  <name>French Toast</name>
  <price>$4.50</price>
  <description>Thick slices made from our homemade sourdough bread</description>
  <calories>600</calories>
</food>
<food>
  <name>Homestyle Breakfast</name>
  <price>$6.95</price>
  <description>Two eggs, bacon or sausage, toast, and our ever-popular hash browns</description>
  <calories>950</calories>
</food>
</breakfast_menu>
```

Overview of XML technologies



XML versus HTML

- XML ***is not*** a replacement or enhanced version of HTML
- The purpose of XML is different to the purpose of HTML
  - XML to transport / encode / store / exchange data
  - HTML to display data / information

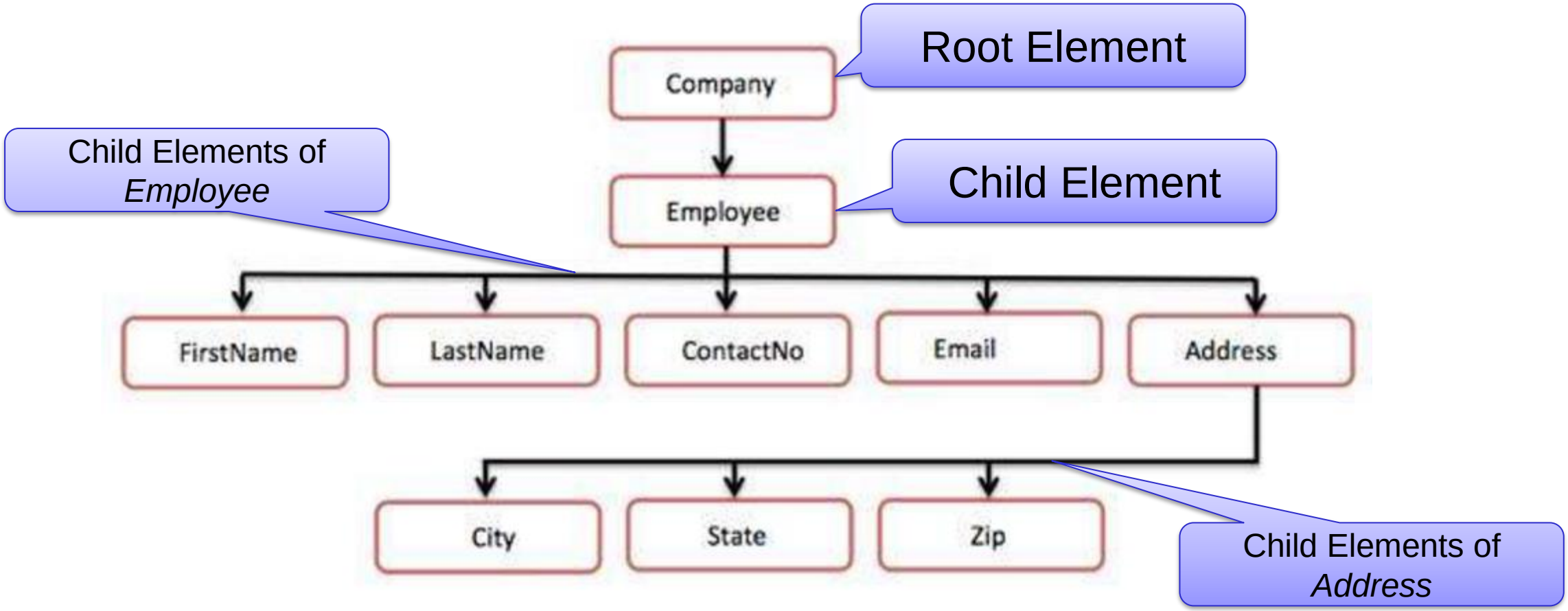
Why XML?

- Facilitates data exchange
  - Easier data transport
  - independent of software and hardware
- Structured data exchange
  - Data is structured
  - Structure can be easily extended
- Standardized data exchange
  - XML is a W3C standard
  - XML allows for the implementation of domain standards (e.g. for spatial data exchange -> Geography Markup Language)

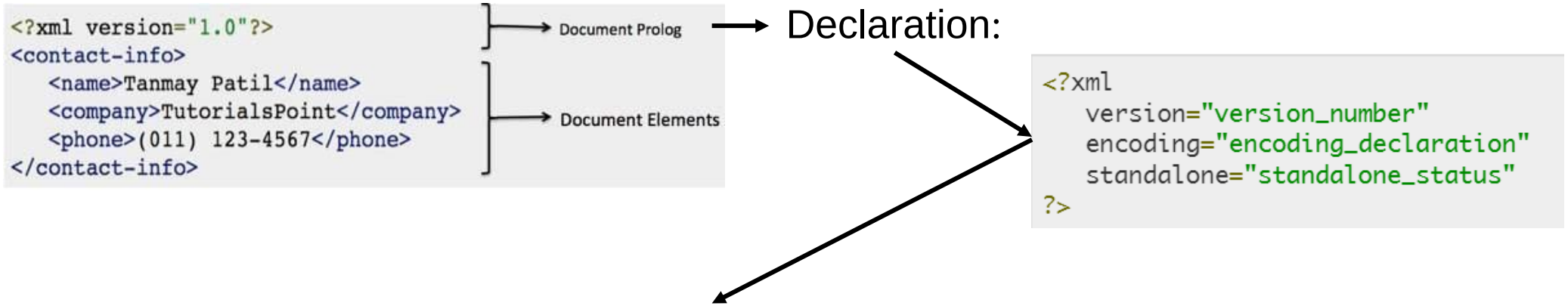
XML – hierarchical structure

```
<?xml version="1.0"?>
<Company>
  <Employee>
      <FirstName>Tanmay</FirstName>
      <LastName>Patil</LastName>
      <ContactNo>1234567890</ContactNo>
      <Email>tanmaypatil@xyz.com</Email>
      <Address>
          <City>Bangalore</City>
          <State>Karnataka</State>
          <Zip>560212</Zip>
      </Address>
  </Employee>
</Company>
```

XML – hierarchical structure



XML –structure




```
<?xml version="1.0"?>
<contact-info>
   <name>Tanmay Patil</name>
   <company>TutorialsPoint</company>
   <phone>(011) 123-4567</phone>
</contact-info>
```

```
<?xml
   version="version_number"
   encoding="encoding_declaration"
   standalone="standalone_status"
?>
```

Parameter	Parameter_value	Parameter_description
Version	1.0	Specifies the version of the XML standard used.
Encoding	UTF-8, UTF-16, ISO-10646-UCS-2, ISO-10646-UCS-4, ISO-8859-1 to ISO-8859-9, ISO-2022-JP, Shift_JIS, EUC-JP	It defines the character encoding used in the document. UTF-8 is the default encoding used.
Standalone	*yes* or *no*.	It informs the parser whether the document relies on the information from an external source, such as external document type definition (DTD), for its content. The default value is set to *no*. Setting it to *yes* tells the processor there are no external declarations required for parsing the document.

XML – Basics structure / definitions

- Tags
  - < FirstName> … opening tag
  - </FirstName> … closing tag
  - Tags mark the start and the end of XML Elements
- Elements (also called XML Nodes)
  - <FirstName>Gerhard</FirstName>
  - Can contain Text (e.g. Gerhard) and attributes
  - can inlcude other XML elements -> nested XML Elements
  - Note:
    - ***Each XML document must have exactly one (and only one) root element. It encloses all the other elements and is therefore the sole parent element to all the other elements***
- Declaration
  - <?xml version="1.0" encoding="ISO-8859-1"?>
  - Defines the xml-version used in the document and the character encoding used (in the example case special characters used in german are supported)
- Attribute
  - <Book ISBN=“12345“>
  - ISBN is an attribute of the „Book“ Element

XML Basic syntax rules

- XML is Case Sensitive
  - <Name> is different to <NAME> or <name>
- Reserved characters

  <Value> precipitation > 1000 mm </Value> … this is a wrong XML syntax

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

  <Value> precipitation &gt 1000 mm </Value> …. this is a correct XML syntax
- Name of the XML elements
  - **must begin with a character (or underscore)** -> <Salzburg123> is valid, <123Salzburg> is not valid
  - can include digits, hyphens, underscores, and periods
  - Cannot include spaces
  - Cannot start with the letters XML (regardless of the case)
  - Apply naming conventions similar to those for UML classes or entities (short, simple and meaningful)
- XML Attributes
  - **must always be quoted** e.g. <Book ISBN=“12345“> or <Book ISBN=‘12345‘>

XML Elements or XML Attributes

XML Attributes do have some disadvantages against XML Elements

- Attributes cannot have "multiple" attribute values (elements can have multiple values)
- Attributes cannot contain hierarchical structures
- Attributes cannot be easily extended

XML Attributes are most commonly used to uniquely identify XML elements (i.e. for attributes that constitute the primary key)

Well formed XML documents / valid XML documents

- **Well formed** XML documents
  - XML documents must have a root element
  - XML elements must have a closing tag
  - XML tags are case sensitive
  - XML elements must be properly nested
  - XML attribute values must be quoted

- **Schema valid** XML documents
  - Are well formed
  - in full adherence to the data structure as defined by a referenced DTD or XML Schema

XML Benefits

- XML is platform independent and programming language independent
- XML supports unicode. Unicode is an international encoding standard for use with different languages and scripts, by which each letter, digit, or symbol is assigned a unique numeric value that applies across different platforms and programs. This feature allows XML to transmit any information written in any human language.
- The data stored and transported using XML can be changed at any point of time without affecting the data presentation. HTML gets the data from XML and display it on the GUI (graphical user interface), once data is updated in XML, it does reflect in HTML without making any change in HTML GUI.
- XML allows validation using DTD and Schema. This validation ensures that the XML document is free from any syntax error and validates the structural integrity of information.
- XML simplifies data sharing between various systems because of its platform independent nature. XML data doesn't require any conversion when transferred between different systems.
- XML can be used to create new XML based derivates for specific information domains – like GML (Geographic Markup Language) for the geographic information domain.

XML considerations

- XML is verbose in comparison to other text based data transmission formats – like for instance JSON
  - causes higher storage and transportation costs, since XML documents can become very large
- Does not support arrays (i.e. multiple values that are stored in one variable)

XML Schema

https://www.w3.org/XML/Schema

XML Schemas

- Def.: „*The purpose of a schema is to define a class of XML documents, and so the term "instance document" is often used to describe an XML document that conforms to a particular schema*„ (http://www.w3.org/TR/xmlschema-0/#PO)
- The schema defines the structure (allowable structure) of a XML document. If a document is structured according to the schema, it is said to be **“valid”** (or “schema valid”) and therefore can be addressed of being a **XML Instance document** of the schema.
- **XML parsers** can parse an XML document in order to check, if it is valid according to the XML schemas.
- XML schemas itself can be defined by the means of XML (XML Syntax)
- There are certain base schemas defined in function of the XML standards (see the W3C standard)
- Typically XML schemas are stored in files with the ending .XSD

XML Schemas - usage

- to provide a list of elements and attributes in a vocabulary
- to associate data types, such as integer, string, etc., for XML elements and XML attributes
- to constrain where elements and attributes can appear, and what can appear inside those elements
- to provide documentation that is both human-readable and machine-processable
- to give a formal description of one or more documents.

- Checking a document against a XML Schema is known as validating against that schema **Validating against a schema is an important component of quality assurance**

XML Schemas – example (including instance document)

XML Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:element name="contact">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="name" type="xs:string" />
            <xs:element name="company" type="xs:string" />
            <xs:element name="phone" type="xs:int" />
        </xs:sequence>
    </xs:complexType>
</xs:element>
</xs:schema>
```

```
<?xml version = "1.0"?>
<contact>
<name>Max Mustermann</name>
  <company>TutorialsPoint</company>
  <phone>(011) 123-4567</phone>
</contact>
```

XML instance document

XML schema - fragment

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
```

- indicates that the elements and data types used in the schema come from the "http://www.w3.org/2001/XMLSchema" namespace
- It also specifies that the elements and data types that come from the "http://www.w3.org/2001/XMLSchema" namespace should be prefixed with xs:

Namespaces

Usage of the same term with a different meaning

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

HTML Table

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

A coffee table

Namespaces – declaration in the root element

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

Namespaces

- Used to distinguish between different XML Vocabularies (think about different dictionaries or glossaries) – **avoiding naming conflicts**
- Namespaces **do provide the normative references for the vocabularies**. A definition of the vocabulary itself is not required to be in place – physically. However, frequently the referenced URI contains the vocabulary definition of name space information (i.e. namespaces provided by standardization organizations).
- The same XML term may have different meanings, dependent on the namespace they stem from.
- Examples for namespaces:
  - http://www.opengis.net/wfs - WFS interface vocabularies
  - http://www.opengis.net/gml - GML vocabularies
  - http://www.opengis.net/ogc - OGC filter vocabularies
- Namespace – Syntax:

  xmlns:*namespace-prefix*="*namespaceURI*„

  e.g. xmlns:wfs=http://www.opengis.net/wfs

  Or:

  xmlns:xsl=http://www.w3.org/1999/XSL/Transform

  <xsl:template match="/">

Namespaces

- A namespace can be declared within an element

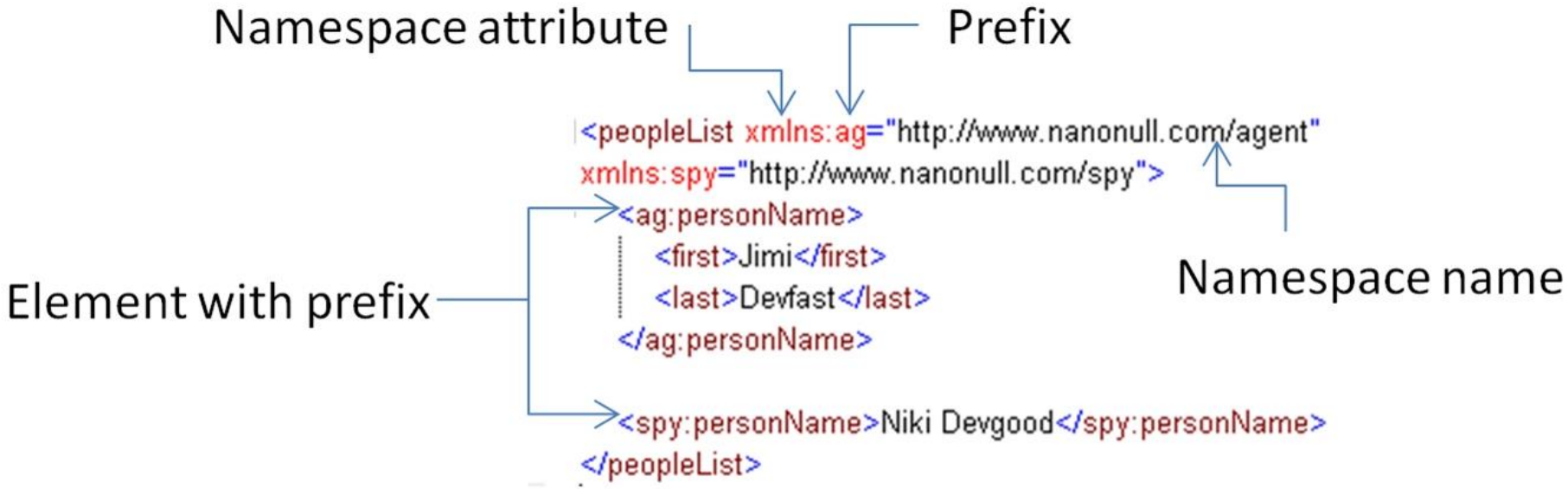
```
<root>

<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 Schema – namespaces declaration



- In xml-schema at least the XML-Schema namespace **must** be defined
  e.g. `xs:schema xmlns:xs=http://www.w3.org/2001/XMLSchema`
- Optionally a target namespace can be defined
  e.g `xmlns:mc=http://my-company.com/namespace targetNamespace="http://my-company.com/namespace"`
- The `targetNamespace` attribute allows us to specify the namespace for which this particular schema is defining components (i.e. all enclosed definition are part of the target namespace)

XML Schema – declaration example

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

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="https://www.w3schools.com"
xmlns="https://www.w3schools.com"
elementFormDefault="qualified">
...
...
</xs:schema>
```

```
xmlns:xs="http://www.w3.org/2001/XMLSchema"
```

indicates that the elements and data types used in the schema come from the "http://www.w3.org/2001/XMLSchema" namespace. It also specifies that the elements and data types that come from the "http://www.w3.org/2001/XMLSchema" namespace should be prefixed with xs:

```
targetNamespace="https://www.w3schools.com"
```

indicates that the elements defined by this schema come from the "https://www.w3schools.com" namespace (the intended target namespace when using this schema)

```
xmlns="https://www.w3schools.com"
```

indicates that the default namespace is https://www.w3schools.com (i.e. the namespace for all non-prefixed elements)

```
elementFormDefault="qualified"
```

indicates that any elements used by the XML instance document which were declared in this schema must be namespace qualified (default is unqualified)

XML Schema – Data Types

```
<xs:element name="AddressType">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="name" type="xs:string" />
            <xs:element name="company" type="xs:string" />
        </xs:sequence>
    </xs:complexType>
</xs:element>
```

Named complex data type

```
<xs:element name="Address1">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="address" type="AddressType" />
            <xs:element name="phone1" type="xs:int" />
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="Address2">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="address" type="AddressType" />
            <xs:element name="phone2" type="xs:int" />
        </xs:sequence>
    </xs:complexType>
</xs:element>
```

Reuse of data types (Global type)

XML Schema Anatomy

Referencing a schema in a XML document

XML Schema

```xml
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="https://www.w3schools.com"
xmlns="https://www.w3schools.com"
elementFormDefault="qualified">

<xs:element name="note">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="to" type="xs:string"/>
      <xs:element name="from" type="xs:string"/>
      <xs:element name="heading" type="xs:string"/>
      <xs:element name="body" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

</xs:schema>
```

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

<note
xmlns="https://www.w3schools.com"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="https://www.w3schools.com/xml note.xsd">
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

XML instance document

Referencing a schema in a XML document

- Reference the XSD schema in the XML document using XML schema instance attributes such as
  - schemaLocation
    - This attribute has two values, separated by a space. The first value is the namespace to use. The second value is the location of the XML schema to use for that namespace:
  - noNamespaceSchemaLocation
    - references an XML Schema document that does not have a target namespace

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

<note
xmlns="https://www.w3schools.com"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="https://www.w3schools.com/xml note.xsd">
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

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

<shiporder orderid="889923"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="shiporder.xsd">
  <orderperson>John Smith</orderperson>
  <shipto>
    <name>Ola Nordmann</name>
    <address>Langgt 23</address>
    <city>4000 Stavanger</city>
    <country>Norway</country>
  </shipto>
  <item>
    <title>Empire Burlesque</title>
    <note>Special Edition</note>
    <quantity>1</quantity>
    <price>10.90</price>
  </item>
  <item>
    <title>Hide your heart</title>
    <quantity>1</quantity>
    <price>9.90</price>
  </item>
</shiporder>
```

XML Online Resources and Mandatory Exercise

- www.w3c.org
  - To study the XML standards and tutorials and presentations
- http://www.w3schools.com/default.asp
  - To get an quick and comprehensive overview / tutorial to XML techniques very useful for beginners.
- **Mandatory exercise:**
  - Conduct the tutorials available under the link
  - http://www.w3schools.com/xml/default.asp - for the **basic XML tutorial**
    - **Note**: for the content of the class it is enough if you cover the topics shown in the figure to the right
  - http://www.w3schools.com/xml/schema_intro.asp - for the **XML schema tutorial**
    - **Note**: for the content of the class it is enough if you cover the topics shown in the figure to the right

The content of the tutorial will be subject of the exam!!!