CS 144

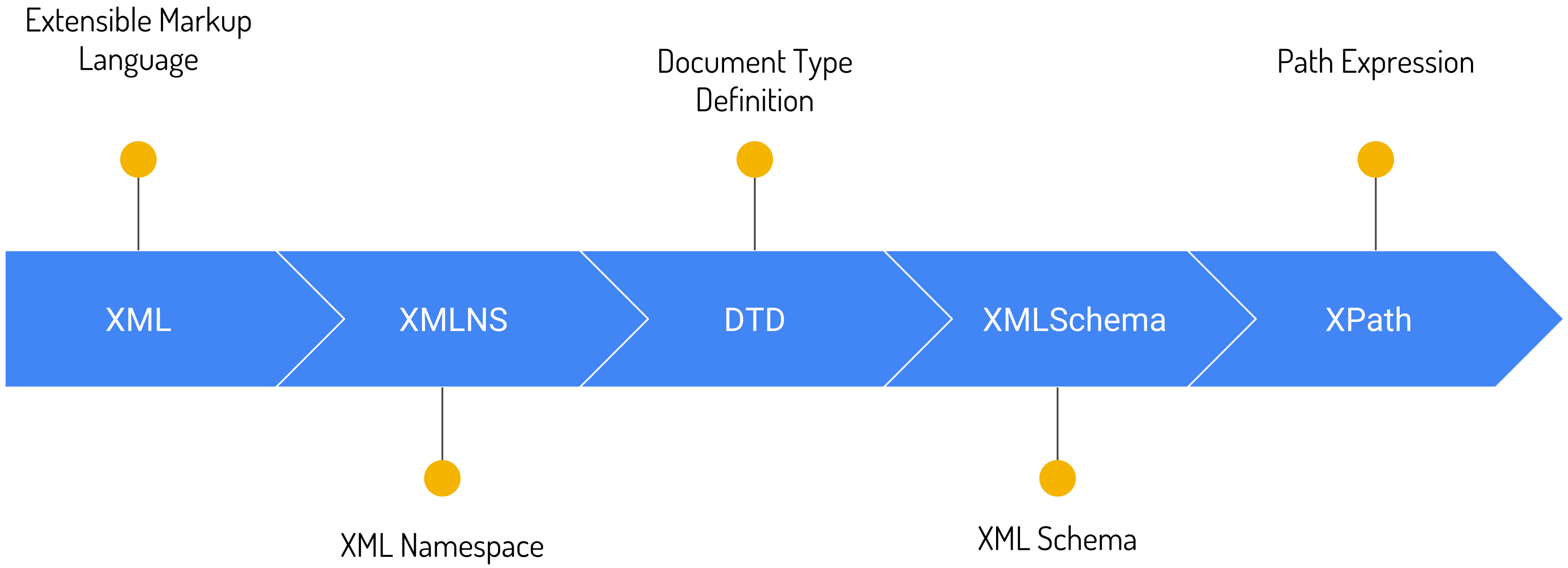
Discussion 2

January 15, 2016

Amogh Param

twitter: @_amogh_
website: amogh.xyz

Lesson Plan


Extensible Markup Language
XML
XMLNS
XML Namespace
Document Type Definition
DTD
XMLSchema
XML Schema
Path Expression
XPath

XML

- Designed to store and transport data.
- Designed to be both human and machine-readable.
- Consists of:
  - Tagged elements | Attributes on elements | Text

```xml
<artist name="Metallica">
        <album title="Master of Puppets">
                <song length="5:12">Battery</song>
                <song length="7:56">Master of Puppets</song>
                <song length="6:28">Welcome Home(Sanitarium)</song>
                <song length="8:21">Disposable Heroes</song>
                <song length="8:26">Orion</song>
        </album>
</artist>
```

XML

- XML Tree
  - Tags become nodes
  - Attributes become child node
  - Text inside XML element creates a separate "text node"

  - Example 1:
    - xmllint
      - Unix command line tool for XML
      - `xmllint --debug music.xml`
    - http://codebeautify.org/xmlviewer

XML

- Elements v/s Attributes
  - Can’t have two attributes with the same name in an element (except?)

```
<musician name="Kirk Lee Hammett" instrument="guitar" stageguitars="ESP KH2">
</musician>"
```

  - Elements can have nested child elements

```
<musician name="Kirk Lee Hammett" instrument="guitar">
        <name>
                <first>Kirk</first>
                <middle>Lee</middle>
                <last>Hammett</last>
        </name>
        <stageguitars>
                <guitar name=”ESP KH2”/>
                <guitar name=”ESP KH602”/>
        </stageguitars>
</musician>"
```

XML Namespaces

xmlns

- Makes it possible for elements of different XML applications to co-exist in the same document.
- Child elements inherit the namespace of the parent. (Example 2.1)
- Namespace declarations look like attribute, but it’s not.
    - Doesn’t show up in the list of attributes
    - Example 2.2 - 2.3

DTD

Document Type Definition

- Purpose:
    - Define a set of elements (tags) and their attributes that can be used to create an XML document
    - Define how elements can be embedded
    - To define the legal building blocks of an XML document (XML validation)

- Can't define element content types:
    - What text can go inside elements.
        - eg: Cannot specify that input should be a number from 0 to 100

DTD

- Internal (.xml)

```
<?xml version="1.0"?>
<!DOCTYPE note [
<!ELEMENT note (to,from,heading,body)>
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT heading (#PCDATA)>
<!ELEMENT body (#PCDATA)>
]>
<note>
<to>Beethoven</to>
<from>Mozart</from>
<heading>Reminder</heading>
<body>Shots tonight</body>
</note>
```

- External (.dtd)

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT Bookstore ( Book* ) >

<!ELEMENT Book ( Title, Author+, Remark? ) >
<!ATTLIST Book  ISBN CDATA #REQUIRED
                Price CDATA #REQUIRED
                Edition CDATA #IMPLIED >

<!ELEMENT Title ( #PCDATA ) >
<!ELEMENT Author ( #PCDATA | ( FirstName,
LastName ) ) >
<!ELEMENT FirstName ( #PCDATA ) >
<!ELEMENT LastName ( #PCDATA ) >
<!ELEMENT Remark ( #PCDATA ) >
```

(In the XML file:)

```
<!DOCTYPE Bookstore SYSTEM “xmlstructure.dtd”>
```

DTD

- <!ELEMENT element-name (element-content)>
- <!ATTLIST element-name attr-name attr-type default-value>
- Entities - &lt; &gt; &amp; &quot; &apos;
- Primitive Types (Example 3.1)
    - PCDATA (Parsed Character Data)
        - PCDATA is text that WILL be parsed by a parser. The text will be examined by the parser for entities and markup.
        - Used with elements
    - CDATA (Character Data)
        - CDATA is text that will NOT be parsed by a parser.
        - Used with attributes

DTD

- PCDATA | CDATA
  - Can have mixed elements as innerText for elements using PCDATA
  - Characters should be escaped for both PCDATA and CDATA
  - Examples 3.2:
- More Primitive Types
  - ID - Value is unique ID (cannot start with a digit)
  - IDREF(S) - The value is the id of another element or list of elements
  - ENTITY - The value is an entity
  - NMTOKEN(S) - The value is a (list of) valid XML name(s)
  - Examples 3.3:

DTD

- Occurrence restrictions on primitive types:
  - ? - zero or one occurrence
  - * - zero or more occurrences
  - + - one or more occurrences
  - | - either types may occur
  - (no modifier) - one occurrence

DTD

- Pros:
  - Compact structure
  - Can be defined inline
  - Wide support among parsers

- Cons:
  - Are not written in XML
  - Don’t support Namespaces
  - Don’t have data – typing
  - Have limited capacity for counters

XML Schema

- XML Schema
  - An XML Schema describes the structure of an XML document.
  - The XML Schema language is also referred to as XML Schema Definition (XSD).

- Purpose:
  - define the legal building blocks of an XML document:
    - the elements and attributes that can appear in a document
    - the number of (and order of) child elements
    - data types for elements and attributes
    - default and fixed values for elements and attributes

XML Schema

Sample XML Schema

```
<?xml version="1.0"?>
<xs:schema targetNamespace="http://oak.cs.ucla.edu/cs144" xmlns:xs="http://www.w3.org/2001/XMLSchema">
      <xs:element name="Book">
            <xs:complexType>
                  <xs:sequence>
                        <xs:element name="Title" type="xs:string"/>
                        <xs:element maxOccurs="unbounded" minOccurs="1"name="Author" type="xs:string"/>
                        <xs:element maxOccurs="1" minOccurs="0"name="Remark" type="xs:string"/>
                  </xs:sequence>
                  <xs:attribute name="ISBN" type="xs:string"use="required"/>
                  <xs:attribute name="Edition" type="xs:string"/>
            </xs:complexType>
      </xs:element>
</xs:schema>
```

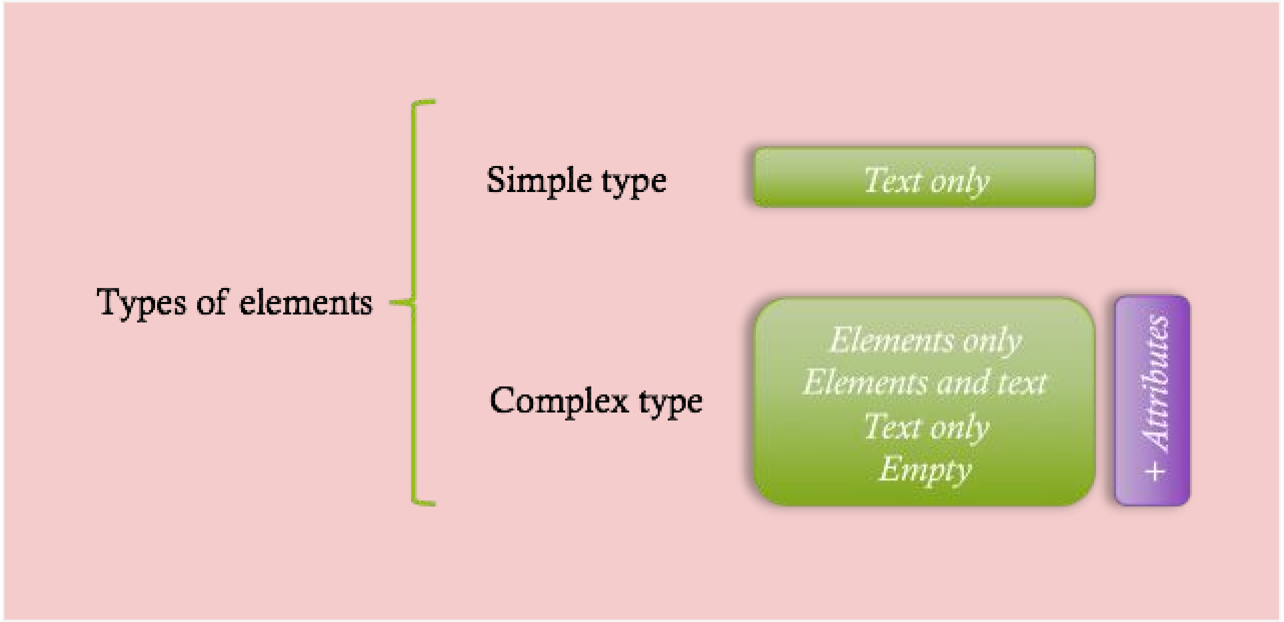
XML Schema

Sample XML Schema

```
<?xml version="1.0"?>
<Bookstore>
      <Book Authors="JU" Ed="2nd" ISBN="0130353000" Price="$65">
            <Title>A First Course in Database Systems</Title>
      </Book>
      <Book Authors="HGM JU" ISBN="0130319953" Price="$75">
            <Title>Database Systems: Complete Book</Title>
            <Remark>It's a great deal!</Remark>
      </Book>
      <Author Ident="HGM">Hector Garcia-Molina</Author>
      <Author Ident="JU">
            <First_Name>Jeffrey</First_Name>
            <Last_Name>Ullman</Last_Name>
      </Author>
</Bookstore>
```

XML Schema

Validation in XML



XML Schema

Simple Type

```
<location>Egypt</location>
```

```
<xs:element name=”location" type="xs:string" />
```

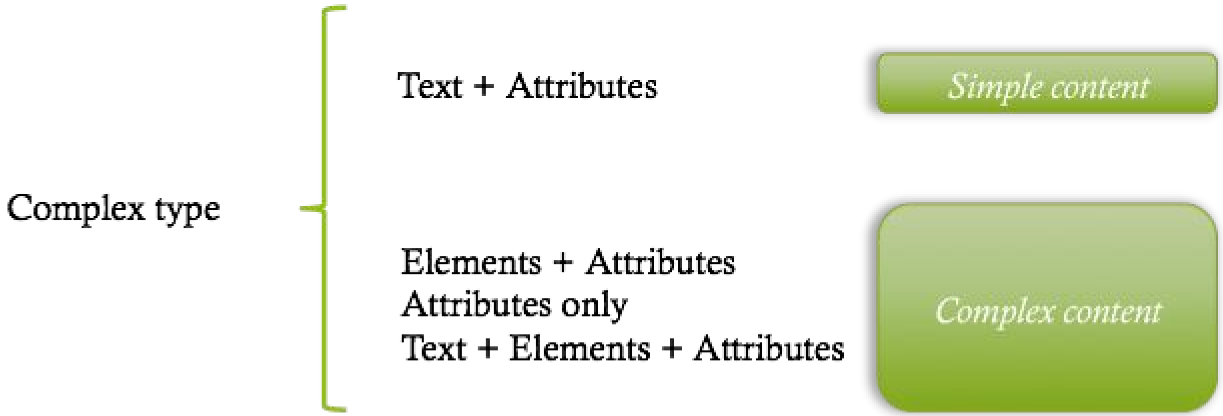
```
<xs:element name="location" type="myStringType" />
<xs:simpleType name="myStringType">
    <xs:restriction base="xs:string">
        <xs:maxLength value="256" />
    </xs:restriction>
</xs:simpleType>
```

XML Schema

http://www.w3schools.com/xml/schema_facets.asp

XML Schema

Complex Type



XML Schema

Complex Type - Simple Content

```
<yearBuilt era="BC">100</yearBuilt>
```

```
<xs:element name="yearBuilt">
   <xs:complexType>
      <xs:simpleContent>
         <!-- Inheritance -->
         <xs:extension base="xs:positiveInteger">
            <xs:attribute name="era">
               <xs:simpleType>
                  <xs:restriction base="xs:string">
                     <!-- Regular expression -->
                     <xs:pattern value="(BC)|(AD)” />
                  </xs:restriction>
               </xs:simpleType>
            </xs:attribute>
         </xs:extension>
      </xs:simpleContent>
   </xs:complexType>
</xs:element>
```

XML Schema

Complex Type - Complex Content

```
<xs:element name="employee" type="fullpersoninfo"/>

<xs:complexType name="personinfo">
  <xs:sequence>
    <xs:element name="firstname" type="xs:string"/>
    <xs:element name="lastname" type="xs:string"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="fullpersoninfo">
  <xs:complexContent>
    <xs:extension base="personinfo">
      <xs:sequence>
        <xs:element name="address" type="xs:string"/>
        <xs:element name="city" type="xs:string"/>
        <xs:element name="country" type="xs:string"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

XPath

Path Expression

- XPath is used to navigate through elements and attributes in an XML document.
- XPath is a syntax for defining parts of an XML document
- XPath contains a library of standard functions
- simple "path expression" that matches XML data by navigating down (and occasionally up or across) the tree and possibly evaluating conditions over data in the tree.

XPath

XPath Lab Examples

http://oak.cs.ucla.edu/cs144/examples/xpath.html