华中师范大学伍伦贡联合研究院
Central China Normal University Wollongong Joint Institute

UNIVERSITY
OF WOLLONGONG
AUSTRALIA

CSIT884
Web Development

Lecture 06 – XML and DTD

Lecture slides are mainly taken from UOW

Objectives

- Use XML to store and transport data over the Internet
- Use DTD to define the structure of an XML document

XML

EXtensible Markup Language

- XML is a markup language much like HTML
- XML is a software- and hardware-independent tool for storing and transporting data.
- XML separates data from presentation.
- File extension is .xml

```
<?xml version="1.0" ?>
<student>
 <firstName>John</firstName>
 <lastName>Smith</lastName>
 <email>jsmith@gmail.com</email>
 <mobile>0211223344</mobile>
</student>
```

XML

- HTML tags are predefined.
- XML tags are defined by user.
- Using XML **D**ocument **T**ype **D**efinition (DTD), or **X**ML **S**chema **D**efinition (XSD), different parties can agree on a standard XML format for interchanging data.
- Another popular format for interchanging data is **J**ava**S**cript **O**bject **N**otation (JSON)

```
{
 "firstName": "John",
 "lastName": "Smith",
 "email": "jsmith@gmail.com",
 "mobile": "0211223344"
}
```

- In most web applications, XML and JSON are used to store or transport data, while HTML and XSLT are used to transform and display the data.

XML

First example of XML:

```
<?xml version="1.0" ?>
<student>
 <firstName>John</firstName>
 <lastName>Smith</lastName>
 <email>jsmith@gmail.com</email>
 <mobile>0211223344</mobile>
</student>
```

XML: XML declaration

```
<?xml version="1.0" ?>
<student>
 <firstName>John</firstName>
 <lastName>Smith</lastName>
 <email>jsmith@gmail.com</email>
 <mobile>0211223344</mobile>
</student>
```

XML declaration

- The **XML declaration** is optional and it **must come first in the document**.
- The XML declaration identifies the document as being XML. Even though it is optional, all XML documents should begin with an XML declaration.
- The XML declaration must be situated at the first position of the first line in the XML document.
- **Do not start an XML file with a blank line!!!**
- Syntax for the XML declaration:

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

XML: root element

```
<?xml version="1.0" ?>
<student>
 <firstName>John</firstName>
 <lastName>Smith</lastName>
 <email>jsmith@gmail.com</email>
 <mobile>0211223344</mobile>
</student>
```

- An XML document **must contain one root element** that is the parent of all other elements

```
<rootElement>
 <child>
  <subchild>.....</subchild>
 </child>
</rootElement>
```

XML: root element

This is NOT a well-formed XML document because it has no root element

```
<?xml version="1.0" encoding="UTF-8"?>
<student>
 <firstName>John</firstName>
 <lastName>Smith</lastName>
 <email>jsmith@gmail.com</email>
</student>
<student>
 <firstName>Mary</firstName>
 <lastName>Jane</lastName>
 <email>mjane@gmail.com</email>
</student>
```

XML: root element

This is a well-formed XML document because it has a root element

```
<?xml version="1.0" encoding="UTF-8"?>
<studentList>
 <student>
  <firstName>John</firstName>
  <lastName>Smith</lastName>
  <email>jsmith@gmail.com</email>
 </student>
 <student>
  <firstName>Mary</firstName>
  <lastName>Jane</lastName>
  <email>mjane@gmail.com</email>
 </student>
</studentList>
```

XML: element

```
<tag attribute1="..." attribute2="...">
</tag>
```

- An **XML element** is everything from (including) the element's start tag to (including) the element's end tag

```
<?xml version="1.0" encoding="UTF-8"?>
<dailyTransaction date="24/02/2015">
 <person staffDbId="103" operation="update">
  <firstName>John</firstName>
  <lastName>Smith</lastName>
  <mobile>0211223344</mobile>
 </person>
 <person staffDbId="-1" operation="add">
  <firstName>Mary</firstName>
  <lastName>Jane</lastName>
  <mobile>0244556677</mobile>
 </person>
</dailyTransaction>
```

Where is the **`dailyTransaction`** element?
Where is a **`person`** element?
Where is a **`mobile`** element?

XML: element

- XML tags are **case sensitive**
  - The tag `<student>` is different from the tag `<STUDENT>`
- Common **naming convention** for XML tags:

```
<student_list>
...
</student_list>
```

or

```
<studentList>
...
</studentList>
```

XML: attribute

```
<tag attribute1="..." attribute2="...">

</tag>
```

- **XML attributes** are used to describe XML elements, or to provide additional information about elements

```
<?xml version="1.0" encoding="UTF-8"?>
<dailyTransaction date="24/02/2015">
 <person staffDbId="103" operation="update">
  <firstName>John</firstName>
  <lastName>Smith</lastName>
  <mobile>0211223344</mobile>
 </person>
 <person staffDbId="-1" operation="add">
  <firstName>Mary</firstName>
  <lastName>Jane</lastName>
  <mobile>0244556677</mobile>
 </person>
</dailyTransaction>
```

Does the **`dailyTransaction`** element have attributes?
Does a **`person`** element have attributes?
Does a **`mobile`** element have attributes?

XML: attribute

- In XML, the attribute values must always be quoted (either by single quotes or double quotes):

```
<dailyTransaction date='24/02/2015'>
 <person staffDbId="103" operation="update">
  <firstName>John</firstName>
  <lastName>Smith</lastName>
  <mobile>0211223344</mobile>
 </person>
</dailyTransaction>
```

XML: relationship between elements

- An XML tree starts at a root element and branches from the root to child elements.
- The terms parent, child, and sibling are used to describe the relationships between elements.

```
<parent>
 <child>
  <subchild>.....</subchild>
 </child>
</parent>
```

XML: attribute vs child element

- Any attribute can be defined as a child element.

For example, instead of using **gender** as an attribute

```
<person gender="M">
 <firstName>John</firstName>
 <lastName>Smith</lastName>
 <email>jsmith@gmail.com</email>
</person>
```

we can define **gender** as a child element of **person**

```
<person>
 <firstName>John</firstName>
 <lastName>Smith</lastName>
 <email>jsmith@gmail.com</email>
 <gender>M</gender>
</person>
```

This contains the same information.

XML: attribute vs child element

- Any attribute can be defined as a child element.

For example, attributes **staffDbId** and **operation** of **person** can be defined as its child elements

```
<person staffDbId="103" operation="update">
 <firstName>John</firstName>
 <lastName>Smith</lastName>
 <mobile>0211223344</mobile>
</person>
```

```
<person>
 <firstName>John</firstName>
 <lastName>Smith</lastName>
 <mobile>0211223344</mobile>
 <staffDbId>103</staffDbId>
 <operation>update</operation>
</person>
```

This contains the same information.

XML: attribute vs child element

- Any attribute can be defined as a child element, **so when should we use attribute and when should we use element?**
- **Metadata (data about data) should be stored as attributes, and the data itself should be stored as elements.**

```
<person gender="M">
 <firstName>John</firstName>
 <lastName>Smith</lastName>
 <email>jsmith@gmail.com</email>
</person>
```

```
<person>
 <firstName>John</firstName>
 <lastName>Smith</lastName>
 <email>jsmith@gmail.com</email>
 <gender>M</gender>
</person>
```


This is better

XML: attribute vs child element

- Any attribute can be defined as a child element, **so when should we use attribute and when should we use element?**
- **Metadata (data about data) should be stored as attributes, and the data itself should be stored as elements.**

```
<person staffDbId="103" operation="update">
 <firstName>John</firstName>
 <lastName>Smith</lastName>
 <mobile>0211223344</mobile>
</person>
```


This is better

```
<person>
 <firstName>John</firstName>
 <lastName>Smith</lastName>
 <mobile>0211223344</mobile>
 <staffDbId>103</staffDbId>
 <operation>update</operation>
</person>
```

XML: empty element and self-closing tag

In HTML, some elements might work well, even with a missing closing tag:

```
<br>

<hr>

<p>

<input ...>
```

In XML, all elements **must** have a closing tag:

```
<student>

...

</student>
```

An element with no content is called an **empty element**:

```
<emptyElement></emptyElement>
```

We can use **self-closing tag** for an empty element:

```
<emptyElement />
```

XML: nested rule

In HTML, some elements might not be nested properly:

```
<b><i>This text is bold and italic</b></i>
```

In XML, all elements **must** be properly nested:

```
<student>
 <firstName>John</firstName>
 <lastName>Smith</lastName>
 <email>jsmith@gmail.com</email>
</student>
```

XML: entity reference

If we place a character like **<** inside an XML element, it will generate an error.

In this case, we need to use the entity reference **`<`**

Entity references

`<`	<	less than
`>`	>	greater than
`&`	&	ampersand
`&apos;`	‘	apostrophe
`&quot;`	"	quotation mark

XML: comments

- Comments in XML:

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

DTD

- XML **D**ocument **T**ype **D**efinition, commonly known as DTD, is a way to define the legal building blocks of an XML document. It defines the document structure with a list of legal elements and attributes.
- Using a DTD, different parties can agree on a standard XML format for interchanging data.
- We can check whether an XML document conforms to a DTD or not.
- File extension is .dtd

DTD

- The DTD can be declared inside the XML file, or it can be defined in a separate file:
  - Internal DTD
  - External DTD

DTD: internal DTD

The following DTD is declared inside the XML file:

```
<?xml version="1.0" standalone="yes" ?>
<!DOCTYPE student [
 <!ELEMENT student (firstName,lastName,email,mobile)>
 <!ELEMENT firstName (#PCDATA)>
 <!ELEMENT lastName (#PCDATA)>
 <!ELEMENT email (#PCDATA)>
 <!ELEMENT mobile (#PCDATA)>
]>
<student>
 <firstName>John</firstName>
 <lastName>Smith</lastName>
 <email>jsmith@gmail.com</email>
 <mobile>0211223344</mobile>
</student>
```

DTD: external DTD

DTD is declared outside the XML file:

```
<?xml version="1.0" standalone=”no" ?>
<!DOCTYPE student SYSTEM "student.dtd">
<student>
 <firstName>John</firstName>
 <lastName>Smith</lastName>
 <email>jsmith@gmail.com</email>
 <mobile>0211223344</mobile>
</student>
```

The content of **`student.dtd`**

```
<!ELEMENT student (firstName,lastName,email,mobile)>
<!ELEMENT firstName (#PCDATA)>
<!ELEMENT lastName (#PCDATA)>
<!ELEMENT email (#PCDATA)>
<!ELEMENT mobile (#PCDATA)>
```

DTD: internal DTD

The following DTD is declared inside the XML file:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
<!DOCTYPE studentList [
 <!ELEMENT studentList (student*)>
 <!ELEMENT student (firstName,lastName,email)>
 <!ELEMENT firstName (#PCDATA)>
 <!ELEMENT lastName (#PCDATA)>
 <!ELEMENT email (#PCDATA)>
]>
```

```
<studentList>
 <student>
  <firstName>John</firstName>
  <lastName>Smith</lastName>
  <email>jsmith@gmail.com</email>
 </student>
 <student>
  <firstName>Mary</firstName>
  <lastName>Jane</lastName>
  <email>mjane@gmail.com</email>
 </student>
</studentList>
```

DTD: external DTD

DTD is declared outside the XML file:

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<!DOCTYPE studentList SYSTEM "studentList.dtd">
<studentList>
 <student>
  <firstName>John</firstName>
  <lastName>Smith</lastName>
  <email>jsmith@gmail.com</email>
 </student>
 <student>
  <firstName>Mary</firstName>
  <lastName>Jane</lastName>
  <email>mjane@gmail.com</email>
 </student>
</studentList>
```

The content of **studentList.dtd**

```
<!ELEMENT studentList (student*)>
<!ELEMENT student (firstName,lastName,email)>
<!ELEMENT firstName (#PCDATA)>
<!ELEMENT lastName (#PCDATA)>
<!ELEMENT email (#PCDATA)>
```

DTD: external DTD

DTD is declared outside the XML file:

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<!DOCTYPE studentList SYSTEM "studentList.dtd">
<studentList>
 <student>
  <firstName>John</firstName>
  <lastName>Smith</lastName>
  <email>jsmith@gmail.com</email>
 </student>
 <student>
  <firstName>Mary</firstName>
  <lastName>Jane</lastName>
  <email>mjane@gmail.com</email>
 </student>
</studentList>
```

To reference it as external DTD, **standalone** attribute in the XML declaration must be set as **no**. This means, declaration includes information from the external source.

DTD: Element declaration

- XML elements are building blocks of an XML document.
- An element is everything from the element's start tag to the element's end tag:

```
<firstName>John</firstName>

<lastName>Smith</lastName>
```

- In DTD, we declare element as follows:

```
<!ELEMENT firstName (#PCDATA)>

<!ELEMENT lastName (#PCDATA)>
```

Here `#PCDATA` stands for Parseable Character Data.

DTD: Element declaration

An element can contain other elements

```
<student>
 <firstName>John</firstName>
 <lastName>Smith</lastName>
 <email>jsmith@gmail.com</email>
</student>
```

In DTD, we declare element containing other elements as follows:

```
<!ELEMENT student (firstName,lastName,email)>
```

It means that the element **`student`** contains elements **`firstName`**, **`lastName`** and **`email`**.

DTD: Element declaration

An element can contain other elements

```
<studentList>
 <student>
  <firstName>John</firstName>
  <lastName>Smith</lastName>
  <email>jsmith@gmail.com</email>
 </student>
 <student>
  <firstName>Mary</firstName>
  <lastName>Jane</lastName>
  <email>mjane@gmail.com</email>
 </student>
</studentList>
```

In DTD, we declare as follows:

```
<!ELEMENT studentList (student*)>
```

It means, the element **`studentList`** contains zero or more elements **`student`**.

DTD: Element declaration

This is the general form of element declaration:

```
<!ELEMENT elementName (content)>
```

- **`elementName`** is the element name that you are defining.
- **`content`** defines what content (if any) can go within the element

DTD: Element declaration

Element content:

```
<!ELEMENT elementName (child1, child2,...)>
```

Example:

```
<!ELEMENT studentList (student*)>

<!ELEMENT student (firstName,lastName,email)>
```

DTD: Element declaration

`<!ELEMENT elementName (child+)>`	child element can occur **one or more** times inside parent element
`<!ELEMENT elementName (child*)>`	child element can occur **zero or more** times inside parent element
`<!ELEMENT elementName (child?)>`	child element can occur **zero or one** time inside parent element
`<!ELEMENT elementName (child1|child2)>`	either of **`child1`** or **`child2`** must occur inside parent element
`<!ELEMENT elementName (child1,child2,child3,...)>`	Child elements **`child1,child2,child3,...`** must occur inside the parent element and **must appear in this order**

DTD: Attribute declaration

This is the general form of attribute declaration:

```
<!ATTLIST elementName attributeName attributeType attributeValue>
```

- `elementName` specifies the name of the element to which the attribute applies
- `attributeName` specifies the name of the attribute
- `attributeType` defines the type of attributes
- `attributeValue` defines the attribute value

DTD: Attribute declaration

```
<!ATTLIST elementName attributeName attributeType attributeValue>
```

`attributeValue`

- can have a default value

```
<!ATTLIST elementName attributeName attributeType "default-value">
```

- can have a fixed value

```
<!ATTLIST elementName attributeName attributeType #FIXED "value">
```

- is required

```
<!ATTLIST elementName attributeName attributeType #REQUIRED>
```

- is implied: if the attribute has no default value, has no fixed value, and is not required, then it must be declared as implied

```
<!ATTLIST elementName attributeName attributeType #IMPLIED>
```

DTD: Attribute declaration

```
<?xml version="1.0" ?>
<dailyTransaction date="24/02/2015">
 <person staffDbId="103" operation="update">
  <firstName>John</firstName>
  <lastName>Smith</lastName>
  <mobile>0211223344</mobile>
 </person>
 <person staffDbId="-1" operation="add">
  <firstName>Mary</firstName>
  <lastName>Jane</lastName>
  <mobile>0244556677</mobile>
 </person>
</dailyTransaction>
```

```
<!ELEMENT dailyTransaction (person*)>
<!ATTLIST dailyTransaction date CDATA #REQUIRED>
<!ELEMENT person (firstName,lastName,mobile)>
<!ATTLIST person staffDbId CDATA #REQUIRED>
<!ATTLIST person operation CDATA #REQUIRED>
<!ELEMENT firstName (#PCDATA)>
<!ELEMENT lastName (#PCDATA)>
<!ELEMENT mobile (#PCDATA)>
```

References

- XML: https://developer.mozilla.org/en-US/docs/Web/XML
- DTD: https://msdn.microsoft.com/en-us/library/ms256469.aspx

  https://www.w3schools.com/xml/xml_dtd.asp