HEXAWARE

XML

Course Objective

- To introduces the web application development using HTML 5, CSS3, JavaScript , XML, Type script and Angular
- To understand tags and API's Available in HTML 5 and new properties available in the CSS.
- To understand the implementation of XML tags.
- To develop UI component using Angular

Session Objective

- Introduction to XML
- Syntax, Rules and Structure of XML
- Attributes of XML
- Schema and its types

Learning material references

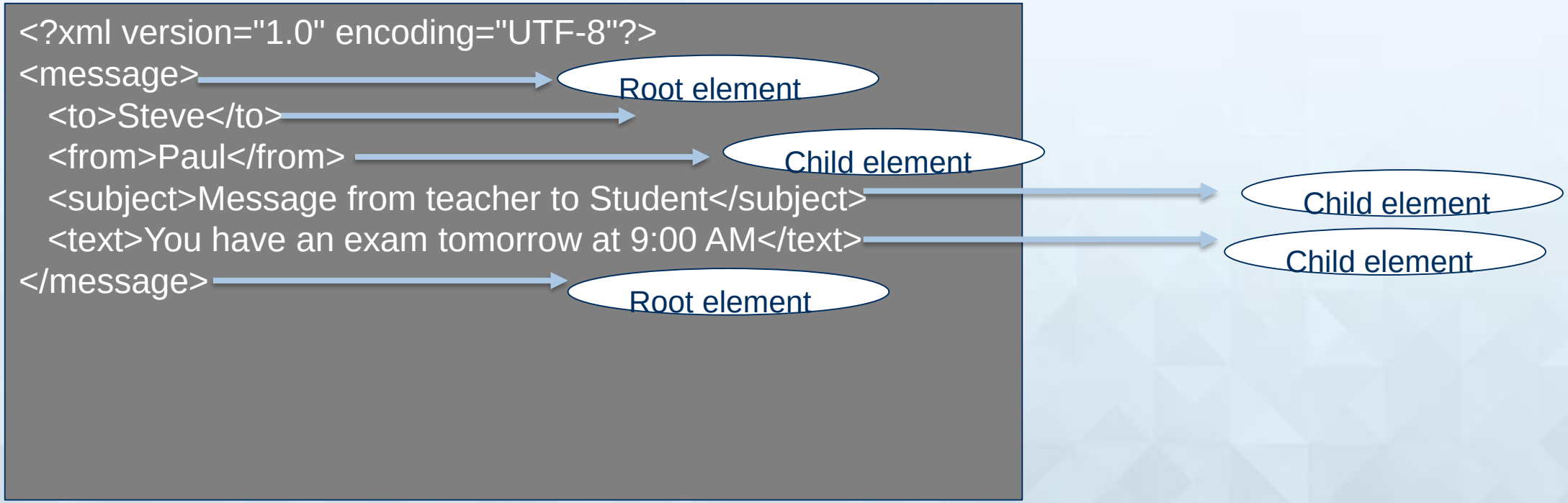
- HexaGuru+
  - Course detail
- Books
  - Henry F Korth, Abraham Silberschatz, “Database system concepts”, McGraw-Hill International editions, Computer Science Series…
  - …
  - ..
- Web
  - URLs..

Introduction to XML

- XML stands for eXtensible Markup Language.
- A markup language is used to provide information about a document.
- Tags are added to the document to provide the extra information.
- XML documents are used to transfer data from one place to another often over the Internet.
- XML subsets are designed for particular applications
- XML is common for all operating system and software.

XML Syntax

- XML document must have a root element. A root element can have child elements and sub-child elements.



XML Rules

- Tags are enclosed in angle brackets.
- Tags come in pairs with start-tags and end-tags.
- Tags must be properly nested.
  - **<name><email>…</name></email> is not allowed.**
  - **<name><email>…</email><name> is.**
- Tags that do not have end-tags must be terminated by a ‘/’.
  -
 is an html example.
- Tags are case sensitive.
- <address> is not the same as <Address>
- XML in any combination of cases is not allowed as part of a tag.
- Tags may not contain ‘<‘ or ‘&’.
- Documents must have a single root tag that begins the document.

Structure of XML Data

- **Tag**: label for a section of data
- **Element**: section of data beginning with <*tagname*> and ending with matching </*tagname*>
- Elements must be properly nested
  - Proper nesting
    - <account> … <balance> …. </balance> </account>
  - Improper nesting
    - <account> … <balance> …. </account> </balance>
  - Formally: every start tag must have a unique matching end tag, that is in the context of the same parent element.
- Every document must have a single top-level element

Example of an XML Document

```
<?xml version="1.0"/>
<address>
    <name>Vinoth</name>
    <email>vinoth@hexaware.com</email>
    <phone>044-45471234</phone>
    <birthday>1984-03-22</birthday>
</address>
```

XML Attributes

- Located in the start tag of elements
- Provide additional information about elements
- Often provide information that is not a part of data
- Must be enclosed in quotes
- Metadata (data about data) should be stored as attributes, and that data itself should be stored as elements

Attributes

- Elements can have **attributes**

```
<account acct-type = “checking” >
    <account_number> A-102 </account_number>
    <branch_name> Perryridge </branch_name>
    <balance> 400 </balance>
</account>
```

- Attributes are specified by *name=value* pairs inside the starting tag of an element
- An element may have several attributes, but each attribute name can only occur once

```
<account  acct-type = “checking”  monthly-fee=“5”>
```


XML Schemas

Schemas

- Schemas are themselves XML documents.
- They were standardized after DTDs and provide more information about the document.
- They have a number of data types including string, decimal, integer, boolean, date, and time.
- They divide elements into simple and complex types.
- They also determine the tree structure and how many children a node may have.

Schema determine

- The type of elements can appear in the document.
- Mandatory elements of schema.
- Structure of parent, child and leaf element of schema.
- Mandatory elements and optional elements of schema.
- Mandatory and optional values can/must be in an attribute of schema.

Schema for First address Example

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:element name="address">
    <xs:complexType>
        <xs:sequence>
                <xs:element name="name" type="xs:string"/>
                <xs:element name="email" type="xs:string"/>
                <xs:element name="phone" type="xs:string"/>
                <xs:element name="birthday" type="xs:date"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
</xs:schema>
```

Explanation of Example Schema

<?xml version="1.0" encoding="ISO-8859-1" ?>

- ISO-8859-1, Latin-1, is the same as UTF-8 in the first 128 characters.

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

- www.w3.org/2001/XMLSchema contains the schema standards.

<xs:element name="address">

<xs:complexType>

- This states that address is a complex type element.

<xs:sequence>

- This states that the following elements form a sequence and must come in the order shown.

<xs:element name="name" type="xs:string"/>

- This says that the element, name, must be a string.

<xs:element name="birthday" type="xs:date"/>

- This states that the element, birthday, is a date. Dates are always of the form yyyy-mm-dd.

Schema: types of content

- Simple type – A simple type element can contain only the text.
- Complex type – A complex type element can contain attributes, other elements, and text.`

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

<xs:element name="beginnersbook">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="to" type="xs:string"/>
      <xs:element name="from" type="xs:string"/>
      <xs:element name="subject" type="xs:string"/>
      <xs:element name="message" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

</xs:schema>
```

Complex type

Simple type

HEXAWARE

Innovative Services

Passionate Employees

Delighted Customers

Thank you

www.hexaware.com