WEB TECHNOLOGIES
ASSIGNMENT 2

Done By

Harikrishnan V

CS6A

Roll No : 27

QUESTION

What is the purpose of XSLT style sheets? Suppose that an XML document contains a root element named students which has a child element student. Each student element has the child elements name, roll_no & branch. Suppose there are 5 instances of the student element. Write an XSLT style sheet to display the student data as an HTML document.

ANSWER

XSLT, Extensible Stylesheet Language Transformations, provides the ability to transform XML data from one format to another automatically.

An XSLT stylesheet is used to define the transformation rules to be applied on the target XML document. XSLT stylesheet is written in XML format. XSLT processor takes the XSLT stylesheet & applies the transformation rules on the target XML document & then it generates a formatted document in the form of XML, HTML, or text format. This formatted document is then utilized by XSLT formatter to generate the actual output which is to be displayed to the end-user.

XML Document

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="student.xsl"?>
<students>
  <student>
     <roll_no>1</roll_no>
     <name>Abhijith G Anil </name>
     <branch> CS </branch>
  </student>
```

```
<student>
    <roll_no>27 </roll_no>
    <name> Harikrishnan V </name>
    <branch> CS </branch>
</student>
<student>
    <roll_no> 38 </roll_no>
    <name> Megha K C </name>
    <branch> CS </branch>
</student>
<student>
    <roll_no> 41 </roll_no>
    <name> Priyanga P Kini </name>
    <branch> CS </branch>
</student>
<student>
    <roll_no> 55 </roll_no>
    <name> Srividya Krishnakumar </name>
    <branch> CS </branch>
</student>
</students>
```

<u>XSLT Stylesheet</u>

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
<html>
  <body>
    <h2> Students Information </h2>
```

```
      <table style="border: 1px solid black;">
         <tr style="background-color: springgreen;">
            <th style="text-align: left;">Roll No</th>
            <th style="text-align: left;">Name</th>
            <th style="text-align: left;">Branch</th>
         </tr>
         <xsl:for-each select="students/student">
         <tr>
            <td><xsl:value-of select="roll_no"/></td>
            <td><xsl:value-of select="name"/></td>
            <td><xsl:value-of select="branch"/></td>
         </tr>
         </xsl:for-each>
      </table>
   </body>
</html>
</xsl:template>
</xsl:stylesheet>
```

QUESTION

What are the benefits of XML schema over DTDs? Discuss with examples, the various user defined data types permissible for XML schemas.

ANSWER

The benefits of XML schemas are as follows:

(i) XML schema use basic XML syntax: XML schemas are created by using XML syntax whereas DTD's use separate syntax.

(ii) XML schema support namespace: XML schemas support namespace functionality, but DTD's don't support this functionality completely.

(iii) XML schema allow the validation of text elements based on datatypes: XML schemas specify the type of textual data that can be used within attributes & elements. This is done by the simple type declarations.

(iv) XML schema allow the creation of complex & reusable content models easily: In a DTD, content model can be reused only when the utilization of parameter entities is allowed. But, XML schemas provide a wide variety of mechanisms to reuse the content models & also model some complex programming concepts easily.

Data Types

XML schema data types can be generally categorized as "simple type" (including embedded simple type) & "complex type". The "embedded simple type" is already defined, but can be used to create a new type through restriction or extension.

Simple Type

A simple type is a type that only contains text data when expressed according to XML 1.0. This type can be used with element declarations & attribute declarations The embedded simple type is provided for in XML Schema Part 2. A restriction may be placed on an embedded simple type to create a new, unique simple type.

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

Here, the section described together with "xs:string" is an embedded simple type according to XML Schema. In this example, we have established the definition that the data type for the element called "Department" is a text string.

Complex Type

A complex data type is a type that has a child element or attribute structure when expressed according to XML 1.0. An element declaration may be used with this type. There are no predefined complex type data types, so the user will always define their own.

eg:
```
<xs:complexType name="EmployeeType">
  <xs:sequence maxOccurs="unbounded">
    <xs:element ref="Name"/>
    <xs:element ref="Department"/>
  </xs:sequence>
</xs:complexType>
<xs:element name="Name" type="xs:string"/>
<xs:element name="Department" type="xs:string"/>
```

In this case the type name "EmployeeType" is designated by the name attribute of the complexType element. A model group is designated in the child element.

New types are created by placing restrictions on or extending simple or complex types.

QUESTION

Can CSS be used to display an XML document? Illustrate with an example.

ANSWER

CSS can be used to display the contents of an XML document in a clear & precise manner. It gives the design & style to the whole XML document.

Basic steps in defining a CSS style sheet for XML:

1. Define the style rules for the text elements such as font-size, color, font-weight, etc.
2. Define each element either as a block, inline or list element, using the display property of CSS.
3. Identify the titles & bold them.

Linking XML with CSS:

In order to display the XML file using CSS, link XML file with CSS.

Syntax:

```
<?xml-stylesheet type="text/css" href="name-of-css-file.css"?>
```

eg: XML File

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/css" href="Rule.css"?>
<books>
    <heading> Welcome </heading>
    <book>
        <title> Title -: Web Programming </title>
        <author> Author -: Chrisbates </author>
        <publisher> Publisher -: Wiley </publisher>
        <edition> Edition -: 3 </edition>
        <price> Price -: 300 </price>
    </book>
    <book>
      <title> Title -: Internet world-wide-web </title>
      <author>Author -: Ditel </author>
      <publisher>Publisher -: Pearson </publisher>
      <edition> Edition -: 3 </edition>
      <price> Price - 400 </price>
    </book>
</books>
```

CSS File (Rule.css)

```
books {
    color: black;
    background-color: white;
    width: 100%;
}
heading {
    color: red;
    font-size: 40px;
    background-color: white;
}
heading, title, author, publisher, edition, price {
    display: block;
}
title {
    font-size: 25px;
    font-weight: bold;
}
```

Output

```
    Welcome
Title -: Web Programming
 Author -: Chrisbates
 Publisher -: Wiley
 Edition -: 3
 Price -: 300
Title -: Internet world-wide-web
 Author -: Ditel
 Publisher -: Pearson
 Edition -: 3
 Price -: 400
```

QUESTION

Give an overview & basics of Responsive CSS Frameworks & also give description about Bootstrap.

ANSWER

Responsive Web Design

Responsive web design makes our web page look good on all devices. It uses only HTML & CSS. It is not a program or a JavaScript.
Web pages can be viewed using many different devices. Our web page should look good, & be easy to use, regardless of the device.
It is called responsive web design when we use CSS & HTML to resize, hide, shrink, enlarge or move the content to make it look good on any screen.

Responsive CSS Frameworks

There are many existing CSS frameworks that offer Responsive Design. They are free & easy to use.
A great way to create a responsive design, is to use a responsive style sheet, like W3.CSS. W3.CSS makes it easy to develop sites that look nice at any size; desktop, laptop, tablet or phone.

eg:
```
<!DOCTYPE html>
  <html>
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
  <body>
  <div class="w3-container w3-green">
    <h1> Demo </h1>
    <p> Resize this responsive page!</p>
  </div>
```

Scanned with CamScanner

```
<div class="w3-row-padding">
 <div class="w3-third">
  <h2> London </h2>
  <p> London is the capital of England </p>
 </div>
 <div class="w3-third">
  <h2> Paris </h2>
  <p> Paris is the capital of France </p>
 </div>
 <div class="w3-third">
  <h2> Tokyo </h2>
  <p> Tokyo is the capital of Japan </p>
 </div>
</div>
</body>
</html>
```

Bootstrap

Bootstrap is a free & open-source CSS framework directed at responsive, mobile-first front-end web development. It contains CSS & (optionally) JavaScript based design templates for typography, forms, buttons, navigation & other interface components.

Bootstrap 4 is the newest version of Bootstrap, which is the most popular HTML, CSS & JavaScript framework for developing responsive, mobile-first websites.

Bootstrap 4 is completely free to download & use.

```
eg: <div class="jumbotron text-center">
      <h1> My first Bootstrap page </h1>
      <p> Resize this responsive page to see the effect </p>
    </div>
    <div class="container">
      <div class="row">
        <div class="col-sm-4">
          <h3> Column 1 </h3>
          <p> Lorem ipsum dolor.. </p>
        </div>
        <div class="col-sm-4">
          <h3> Column 2 </h3>
          <p> Lorem ipsum dolor.. </p>
        </div>
        <div class="col-sm-4">
          <h3> Column 3 </h3>
          <p> Lorem ipsum dolor.. </p>
        </div>
      </div>
    </div>
```

QUESTION

Write the different data types used in JSON. Explain about schemas & objects. Give examples.

ANSWER

JSON Data Types

In JSON, values must be one of the following data types:

- String

  Strings in JSON must be written in double quotes.

  eg: { "name": "John" }

- Number

  Numbers in JSON must be an integer or a floating point.

  eg: {"age": 30}

- Object

  Values in JSON can be objects.

  eg:
  ```
  {
      "employee": {"name": "John", "age": 30, "city": "New York"}
  }
  ```

- Array

  Values in JSON can be arrays.

  eg:
  ```
  {
      "employees": ["John", "Anna", "Peter"]
  }
  ```

- Boolean

  Values in JSON can be true/ false.

  eg: { "sale": true }

• Null

Values in JSON can be null.

eg: {"middlename": null}

JSON Schemas

JSON Schema is a specification for JSON based format for defining the structure of JSON data. JSON Schema-

- Describes your existing data format.
- Clear, human & machine-readable documentation.
- Complete structural validation, useful for automated testing.
- Complete structural validation, validating client-submitted data.

There are several validators currently available. Currently the most complete & compliant JSON Schema validator available is JSV.

eg:
```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "title": "Product",
    "description": "A product from Acme's catalog",
    "type": "object",
    "properties": {
        "id": {
            "description": "The unique identifier for a product",
            "type": "integer"
        },
        "name": {
            "description": "Name of the product",
            "type": "string"
        },
        "price": {
            "type": "number",
            "minimum": 0,
            "exclusiveMinimum": true
        }
      },
    "required": ["id", "name", "price"]
}
```

JSON Objects

JSON objects are surrounded by curly braces ({}). JSON objects are written in key/value pairs. Keys must be strings, & values must be a valid JSON data type (string, number, object, array, boolean or null). Keys & values are separated by a colon. Each key/value pair is separated by a comma.

Accessing Object Values:

We can access the object values by using dot (.) notation.

```
eg  myObj = {"name": "John", "age": 30, "car": null};
    x = myObj.name;
```

We can also access the object values by using bracket ([]) notation.

```
eg: myObj = {"name": "John", "age": 30, "car": null};
    x = myObj["name"];
```

Looping an Object:

We can loop through object properties by using the for-in loop.

```
eg: myObj = {"name": "John", "age": 30, "car": null};
    for(x in myObj) {
        document.getElementById("demo").innerHTML += x;
    }
```

In a for-in loop, use the bracket notation to access the property values.

eg:
```
myObj = {"name": "John", "age": 30, "car": null};
  for (x in myObj) {
      document.getElementById("demo").innerHTML += myObj[x];
  }
```

Nested JSON Objects:

Values in a JSON object can be another JSON object.

eg:
```
myObj = {
      "name": "John",
      "age": 30,
      "cars": {
          "car1": "Ford",
          "car2": "BMW",
          "car3": "Fiat"
      }
  }
```

We can access nested JSON objects by using the dot notation or bracket notation.

eg:
```
x = myObj.cars.car2;
  //or:
  x = myObj.cars["car2"];
```

Modify Values:

We can use the dot notation to modify any value in a JSON object.

eg:
```
myObj.cars.car2 = "Mercedes";
```

We can also use the bracket notation to modify a value in a JSON object.

eg:
```
myObj.cars["car2"] = "Mercedes";
```

Delete Object Properties

Use the delete keyword to delete properties from a JSON object.

eg: delete myObj. cars. car2;