ONLINE LEARNING PLATFORM VALIDATION

Introduction

This is a case study whereby we are required to convert an XML data of courses offered in an online learning platform into a more user-friendly HTML format for a website. The available details are course title, instructor, duration, and availability. In this respect, we shall develop an Extensible Stylesheet Language (XSL) stylesheet in order to structure the XML data into an organized table in HTML. Moreover, such XML data will be validated against the XML Schema Definition (XSD) to ensure that the data is correct and it composes the proper structure of the information.

XSL Stylesheet: courses.xsl

Purpose

The XSLT stylesheet formats XML data into an HTML format, which easily enables the user to analyze course offerings. The stylesheet formats the XML data into a structured HTML table with proper headers and footers and includes an "Enrol Now" button next to each course, which links to the course enrolment page.

- Navigation: Next to every course is an "Enrol Now" button that navigates the user to that course's enrolment page.

Key Sections

- <xsl:stylesheet> tag: This tag defines the version for XSLT and declares the required namespaces.

- <xsl:template match=\"/"> tag: This is the root template that generates the overall HTML structure.

- <xsl:for-each select="cs:courses/cs:course"> tag: This loop goes through each course entry and formats the table with the corresponding course information.

XML Schema Definition: courses.xsd

Purpose

The main purpose of XSD is to define the structure, constraints, and rules of the XML data. It ensures that the XML data follows a predefined structure and validates the integrity of the data.

Key Components

- Element Declarations: Defines elements such as course title, instructor, duration, availability.

- Data Validation: Validates whether the XML data corresponds to the required structure and adheres to the constraints.

- Facets: It involves facets like ensuring that the duration is numeric; the availability field is properly formatted.

Base Building Constructs

- <xs:schema> tag :** It states the namespace and the basic structure of the XSD.

- <xs:element name="courses">` : Here is where root element of the XML is defined.

- `<xs:complexType>` tag : It describes the overall structure for elements inside the `courses` element.

XML Data File: `courses.xml`

Usage

This XML file contains information about the various courses that the platform offers. Details include the course title, instructor, duration, and if a course is available or not. The display of this XML data in HTML is attained using an XSL stylesheet.

Salient Features

Structured Data: Data in the form of XML is well-formatted, and each entry is comprehensive for a course.

It contains information regarding all courses being run.

Crucial Sections

- `<courses>`: The root element that includes all course entries.

- `<course>`: Each individual entry for a course, with information like title for the course, instructor, and the number of hours.

Transformation and Validation Process

Tools to be Used

- XSLT Processor: Any tool like Saxon or the XSLT transformation feature built in modern web browsers to transform XML data using XSL stylesheet.

- XML Validator: XML Validation Against XSD Supported by Tools or IDEs.

Process

1. Open XML data file, `courses.xml`.

2. Apply the XSL stylesheet, `courses.xsl`, to transform the XML data into an HTML table format.

3. The transformation transulates the XML into a nicely formatted HTML table for each of the courses, styled with CSS, and adds an "Enroll Now" button for each course.

Testing

Testing Tool Implemented

- XML Validator: Either an XML checker or an IDE that has the feature of XSD Validation.

Steps

1. Test the XML data against the provided XSD schema. The data must adhere to the structure and rules specified.

2. Structure, content, and constraint testing, and all structure, content and constraint problems fixed.

Problems and Issues Encountered

XSD Assertion Failures

- Mandatory Elements Missing: Some mandatory elements, such as course title or instructor were missing initially. These were all corrected by ensuring that all fields declared as mandatory were provided.

- Bad Data Types: The wrong data types were used such as the non-numeric values which were used for duration. These were corrected by applying the right data types in the XML document.

- Constraint Violations: _duplicates and other constraint violations were noted and corrected.

XSL Transformation Problems

- Poorly Formed URL: It was checked that all "Enrol Now" links had properly formatted and valid URLs.

- Entity Reference Problems: Any problems with the poorly defined entity references were fixed.

Test Cases

Test for Valid data

A well formatted XML file, which was structured according to the XSD schema, was used to test if the transformation is generating the expected output for HTML format with details of courses in tabular form.

Test for Invalid data

Mandatory Elements Missing Tested XML files with missing elements such as course title missing. The XML validation correctly detected these errors.

Data Type Mismatch: Tested XML files with data type mismatches, including non-numeric duration. These were captured during the validation of the XSD.

Scripts/Programs Used

XSLT Processor: This was used to apply the XSL stylesheet against the XML data.

XML Validator: These are tools or IDEs used to validate the XML data against the XSD schema.

Conclusion

This study case has proven that course data, extracted from the online learning platform, can be dynamically transformed into HTML and compliant with an XSD schema. Testing activities ensured structuring of data as well as correctness and user-friendly presentation; all the errors found were fixed, so the solution is safe and efficient in use to visualize course information to the end user.