# XML Programming

**Duration:** 5 Days

**US Price:** $2795

**Delivery Options:** Attend face-to-face in the classroom, remote-live or via on-demand training.

## Description

The eXtensible Markup Language (XML) provides a standard, document-based approach to handling, transforming, storing and querying structured data. XML is widely accepted as a file and message format because it preserves the structure of application data in a language-independent way. Standard tools make it possible to merge content from distributed systems with relative ease. XML is a fundamental building block of interactive web applications, enabling service-oriented architectures in which XML is used as the message payload. XML is the basis for web display languages such as XHTML (used in browsers), WML (cell phones), SVG (vector graphics), SMIL multimedia presentations and others.

This hands-on XML programming class is a thorough introduction to using XML in a variety of practical applications using Java, .NET and JavaScript.

The course covers structuring data with XML; validating data with document type definitions (DTDs) and XML Schemas; creating and viewing XML documents; transforming XML documents with the XML Stylesheet Language (XSL, XSLT and XPath); service- oriented architectures using SOAP and Web Services; accessing and editing XML data via the document object model (DOM) and Simple API for XML (SAX) libraries; mapping XML structures to and from databases and object-oriented languages. These techniques are then combined in client or server-based applications to deliver rich AJAX user interfaces with clear and modular code.

Extensive examples in Java, ECMAScript (JavaScript) and .NET environments combined with comprehensive hands-on lab exercises reinforce the concepts being taught and introduce the practical application of XML to business problems.

## Prerequisites

Programming experience in an object-oriented language such as Java, JavaScript (JScript, ECMAScript) or C# is strongly recommended.

## Course Overview

### XML Fundamentals

- Using XML to Represent Structured Data
- XML as an International Standard
- World Wide Web Consortium (W3C) Specifications
- Advantages of XML
- XML Applications and Use Cases
- Essential XML Syntax
    - XML Document Structure
    - Comments
    - Elements, Attributes, Sub-Elements
    - Entity References
    - Special Characters, Entity References and CDATA Sections
    - Processing Instructions
- XML Document Validation
    - Writing Well-Formed Documents
    - Ensuring Validity for a Specific Purpose
    - Document Type Definitions and Schemas
- XML Namespaces
    - Using Namespaces to Identify an XML Application
    - Namespace Identifiers - URLs, URIs, and URNs
    - Namespace Prefixes
    - Using Multiple Namespaces

### Reading, Writing and Modifying XML in Programs

- The W3c Document Object Model (DOM)
    - The DOM Document Tree
    - DOM Node Types and Properties
    - Programming with the Node Interface
    - DOM Language Mappings - Java, ECMAScript, etc.
    - Node Collections - `Nodelist` and `Namednodemap`
- Specialized Interfaces and Text Nodes
- Navigating Through a Document
    - Selecting Elements
    - Handling Whitespace
    - Handling Namespaces
- Using the ECMAScript (JavaScript) DOM Parser
- Using the Java API for XML Processing (JAXP)
- Reading XML Into a DOM Document
- Writing Out XML from a DOM Document
- Modifying and Adding Data to a Document
- Deleting Data from a Document
- Using HTML DOM Extensions

### Designing XML to Model Application Data

- Using XML to Model Real-World Data and Processes
- XML and Object-Oriented Analysis and Design
- XML and Data Modeling
- Modeling Data with Elements and Attributes
- Modeling Relationships
    - Containment, Composition and Subelements
    - IDs and References
    - Collections and Lists

### Defining and Enforcing Correct XML Usage

- Defining An XML Application Dialect
- XML Validation
    - Choosing When to Validate
    - Choosing a Validation Technology
- Validating with a Document Type Definition (DTD)
    - Using the Doctype Declaration
    - Internal DTDs
    - External DTDs and Identifiers
    - DTD Treatment Of Namespaces

- Modeling Activities
    - Commands
    - Loops
    - Sequences
- Modeling Data Schemas
- Refactoring For Reuse - Element Groups and Data Types

- Validating with an XML Schema
    - Using a Schema Instance
    - Schema Treatment of Namespaces
    - Linking to Multiple Schemas
- Using DTDs and Schemas Together
- Using Validating Parsers
- Enabling Validation in ECMAScript, Java, .NET

## Designing Document Type Definitions (DTDs)

- Essential DTD Markup
- Defining Content Models with ELEMENT Declarations
- Defining Attributes with ATTLIST Declarations
- Built-in Attribute Types - CDATA, NMTOKEN, Enumerated Values
- Defining References with ID and IDREF Attributes
- Grouping Elements for Reuse
- Using Entities to Reference External Data
- Using Entities in Attribute Lists
- Using Namespaces with DTDs
- Conditional Sections in DTDs
- Limitations of DTDs

## Viewing and Styling XML in Browsers

- Cascading Style Sheets (CSS)
- Styling XML with CSS
- Using the `<?xml-stylesheet ?>` Processing Instruction
- CSS Essential Syntax - Selectors, Classes, Styles
- CSS Style Attributes
- The CSS Box Layout Model
- Exploiting the Rules For Cascading Styles
- CSS Generated Content
- CSS "At-Rules"
- CSS Limitations

## Designing XML Schemas I – Document Structure

- W3C XML Schemas
- Overcoming DTD Limitations with XML Schemas
- Essential Structural Elements
    - `schema`
    - `element`
    - `attribute`
    - `simpleType`
    - `complexType`
- Built-in Data Types
    - String, Numeric, Data/time, etc.
- Defining Simple and Complex Types
- Anonymous, Local and Global Types
- Factoring For Reuse
    - Named Complex Types
    - Groups
    - Attribute Groups

## Designing XML Schemas II - Data Types

- Deriving Types with Extension and Restriction
- Deriving Complex Types with Inheritance
- Using Facets to Restrict Derived Types
- Validating with Patterns and Regular Expressions
- Substitution Groups
- Validating Key and Keyref Elements
- Validating Uniqueness
- Validating Required Fields
- Union and List Types

Combining Schemas with Include, Import and Redefine
- Handling Target Documents with No Declared Namespace
- Mixing DTDs and Schemas

## Transforming XML with XSL and XPath

- The eXtensible Stylesheet Language (XSL)
  - XSL Components and Processors
- XSL Transforms (XSLT)
  - XSLT Use Cases
  - Transforming Business XML to Alternative Dialects
  - Transforming XML to Display Languages
- Linking a Stylesheet to an XML Document
- Template Text and XSLT Output
- Rendering HTML Using XSLT and CSS
- Applying a Transform in a Web Browser
- Applying a Transform Programmatically
- Transforms in Java, ECMAscript, .NET
- Selecting Output Options (Text, Output Stream, File)
- Choosing Server-Side vs. Client-Side Transformation
- XPath Expressions
  - The XPath Data XPath
  - Essential XPath Syntax
  - Paths, Axes, Node Tests, Functions and Operators
  - XPath Predicates
- Using XPath Expressions to Select Data
- XPath in Programs and Stylesheets

## Using XML As Web Content

- XML For Display Technologies
- XHTML - An Improved HTML For Browsers
- XHTML Mobile Profile (XHTML MP)
- Wireless Markup Language (WML) a Display Language For Wireless Devices
- Special-Purpose Display XML - SVG, SMIL, MathML and Others
- Browser Support For Rendering XML
- Including Raw XML Data in HTML
- Converting XML Data with the JavaScript DOM

## Defining XSL Transforms

- Defining An XSL Stylesheet
- Essential Stylesheet Elements
  - `template`
  - `apply-templates`
  - `value-of`
- Built-in Templates
- Applying a Template to Selected

## Formatting XML Documents with XSL-FO

- XSL Formatting Objects (XSL-FO)
- Overcoming XSLT Limitations with XSL-FO
- Designing XSL-FO Stylesheets
- Inserting XSL-FO Tags with XSLT
- Using XSL-FO Processors to Produce

Content
- Using XPath Expressions in `"select"` and `"match"` Attributes
- Sorting Selected Content
- Conditional Output with `if, choose, when, otherwise`
- Looping Through Content with `for-each`
- Creating Well-Formed Output Elements
- Copying Content from Source to Destination
- Modular Design Using Named Templates
- Calling Named Templates
- Exploiting Parameters, Variables and Generated Content
- Combining Stylesheets

## Using XML to Build Service-Oriented Architectures (SOA)

- Architecture of Web Services
- Web Service Use Cases
- Protocols and Message Payloads
- SOAP's Role
    - Soap Namespaces and Schemas
    - Elements Of a SOAP Message
    - Sending and Receiving Soap Messages (SOAP Clients and Receivers)
    - Handling SOAP Faults
- Web Services Description Language (WSDL)
- Deploying and Consuming Web Services
- WS-I Profile for Web Services
- Writing Web Services in .NET
- Writing Web Services in Java EE
- Deploying a Service
- Generating Code from WSDL
- Discovery Mechanisms For Web Services
- OASIS Universal Description, Discovery, and Integration (UDDI)
- RESTful Web Services

PDF and Text Files

## Storing and Mapping XML Databases

- Mapping XML to Database DDL
- Building XML from Query Results
- Storing XML in Databases
- XQuery
    - Searching XML Data
    - Querying XML Data
- XQuery Essential Syntax
    - Expressions
    - Queries
    - Output

## Creating Rich Web Interfaces with AJAX

- Purpose and Architecture of Asynchronous JavaScript and XML (AJAX)
- AJAX Use Cases
- Designing Interactive Web Applications
- Using JavaScript Event Triggers
- Sending HTTP Requests with the `XMLHttpRequest` Object
- Processing Asynchronous Responses
- Incorporating Results into the Current Page
- AJAX with JSON
- Supporting Bookmarks and History Lists
- Fallback Support For Limited Browsers

## Simple API For XML Parsing (SAX)

- SAX Purpose and Fundamental Architecture
- Event-Driven Parsing
- Building a SAX Handler
- Writing Namespace-Aware Code
- Choosing Between SAX and DOM Parsing
- Loading and Processing an XML File
- Using the SAX Parser in Java and .NET Applications