

The Extensible Markup Language (XML) defines a way of marking up text to describe the structure of data. XML allows you to create your own markup language: you define the tags that give meaning to your data. The World-Wide Web Consortium (W3C) creates and maintains the definition of XML, making it a standard for creating markup languages. Industries and organizations use XML to write rules defining their own markup languages.

In this course, students will learn how to create well-formed XML documents. By adding a third day to the course, we've enhanced *Introduction to XML* by extending our coverage of XML Schema and XSLT. A new chapter covering XML Schema in conjunction with Namespaces supplements a trio of existing chapters on XML, Namespaces, and XML Schema. The additional three chapters on XSLT give students more than just a jump start on XSLT, but rather full, in-depth coverage of this complex topic.

Course Objectives:

- Explain what XML is, and how it is used in application and document development.
- Write well-formed documents that conform to XML's basic rules of syntax.
- Validate XML documents with both DTDs and XML Schemas.
- Identify the key differences between DTDs and XML Schemas.
- Use XML Namespaces to distinguish between XML tags.
- Transform an XML document into an HTML document using XSLT.
- Use XPath to navigate a document tree.
- Explain how programs can use DOM and SAX to parse XML documents.

Audience: Application developers, web developers and administrators, and XML authors.

Prerequisites: *HTML*. Familiarity with web and data processing concepts. Programming experience is helpful, but not necessary.

Number of Days: 3 days

<p>1. Course Introduction Course Objectives Overview Suggested References</p> <p>2. Getting Started with XML Data and Document Structure XML Well-Formed XML Valid vs. Well-Formed XML Enforcing Valid Documents: DTDs Enforcing Valid Documents: XML Schema Presentation Style XSL and XSLT</p>	<p>3. Writing Well-Formed XML Using XML XML Fundamentals Tag Attributes Naming Rules Empty and Non-Empty Elements Nesting and Hierarchy of Tags Processing Instructions and the XML Declaration Other XML Constructs Entity and Character References</p> <p>4. Namespaces Why Namespaces? Namespace Prefixes and Declaration</p>
---	---

- Multiple Namespace Declarations
- Declaring Namespaces in the Root Element
- Default Namespaces
- 5. Validating XML with DTDs**
- XML DTDs
- DOCTYPE
- Element Conditions and Quantifiers
- Attributes
- Attribute Types
- REQUIRED, IMPLIED, and FIXED
- Parsed General Entities
- Parsed Parameterized Entities
- DTDs and Namespaces
- 6. Validating XML with XML Schemas**
- Schema Overview
- A Minimal Schema
- Associating XML With a Schema
- Simple and Built-in Types
- Complex Types
- Element Declarations
- Attribute Declarations
- Choices
- Named Type and Anonymous Types
- 7. Using XML Schema with Namespaces**
- Qualified and Unqualified XML
- Associating Qualified XML with a Schema
- Associating a Schema with a Namespace
- Controlling Element and Attribute Qualification
- Merging Schema with the Same Namespace
- Merging Schema with Different Namespaces
- 8. Intro to XSLT**
- Stylesheet, Source, and Result
- XSLT Processors
- Processor Implementations
- XPath Basics
- xsl:stylesheet
- xsl:template
- xsl:value-of
- xsl:apply-templates
- xsl:output
- 9. XPath Nodetypes**
- XPath Expressions
- XPath Context
- XPath Location Steps
- Element and Root Nodes
- Text and Attribute Nodes
- Comment and Processing Instruction Nodes
- Namespace Nodes
- Wildcards
- Whitespace
- Default Template Rules
- 10. XPath Axes and Predicates**
- Location Paths and Location Steps
- Peer Axis Types
- More Peer Axis Types
- Descendant Axis Types
- Ancestor Axis Types
- Node Tests
- Predicates
- Functions
- 11. XSLT Flow Control**
- Introduction
- xsl:if
- xsl:choose
- xsl:for-each
- xsl:sort
- Named Templates
- Mode
- 12. XML in Applications**
- Reasons and Places for Using XML
- DOM Parsers
- SAX Parsers
- Web Services
- 13. Appendix A – Effective Document Design**
- Design Goals
- Intended Audience
- Document Types
- Choosing a Validation Method
- Incorporating Namespaces
- Modular Document Design
- Planning for Extensibility