



Capstone Courseware, LLC

33 Boylston Street  
Jamaica Plain, MA 02130

877-227-2477  
capstonecourseware.com

## 561. Developing Java Web Services

### Version 1.5.3

This one-week course prepares Java programmers to develop interoperable Java Web services and using SOAP, WSDL, and XML Schema. Students get an overview of the interoperable and Java-specific Web services architectures, and then learn the standard APIs for SOAP messaging and WSDL-driven, component-based service development. Both document-style and RPC-style messages and services are covered in depth.

The introductory chapters give overviews of the consensus architecture for interoperable Web services, including the WS-I Basic Profile, and the Java Web services architecture as codified by the J2EE 1.4 specification, including SAAJ and JAX-RPC. These chapters are meant to be equally useful to developers and non-developers -- project managers, analysts, technologists and support staff. There is a great deal of hands-on demonstration of running Web services, inspecting SOAP traffic, WSDL definitions, and a little bit of Java code, but no Java coding. The focus is on the architecture itself, and on the roles that various protocols, APIs, tools, and application components play in a working Web service and/or client.

The course then gets down to the various brass tacks: students learn the Simple Object Access Protocol (SOAP) 1.1, and acquire skills in using the SOAP with Attachments API for Java (SAAJ) and the Java API for XML Messaging (JAXM) to build "low-level" SOAP-based Web services and clients, in which the programmer is responsible for element-by-element content of the SOAP message. Students will learn to read SOAP and to write it by hand, and then will proceed to use the Java APIs to develop servlets that respond to SOAP/HTTP messages.

The course then moves to "high-level" services: component-based development using the Web Services Description Language (WSDL) to define interoperable messaging models and the Java API for XML-Based RPC (JAX-RPC) to automate the SOAP messaging for remote procedure calls between objects. JAX-RPC abstracts almost all the transport-level implementation -- SOAP over HTTP -- and this allows the Java developer to concentrate on application and service specifics. Students get hands-on experience in developing Web services starting either from WSDL descriptors or from existing J2EE applications. Both servlet and EJB endpoint models are studied, as is the management of SOAP headers using JAX-RPC handler chains.



Finally, the course covers advanced techniques including SOAP attachments (using either SAAJ or JAX-RPC), EJBs and JSPs as Web services and clients, and Java Web-service security.

## Prerequisites

- Solid experience in Java Programming, including object-oriented Java and the Java streams model, is essential to learning to build Java Web services. Course 103 is excellent preparation.
- Some experience with J2EE development, especially Web applications using servlets, will be very helpful, but is not strictly required.
- Some understanding of XML and XML Schema is strongly recommended. See Courses 501, "Introduction to XML," and 517, "Introduction to XML Schema."
- Various related technology is discussed in the course: JAXP, SAX, DOM, XSLT, XPath, JSP and JSTL. None of these is a formal prerequisite for the course, and labs are built to allow students without experience in these things to work through successfully. Experience in these areas will be helpful, however.





## Learning Objectives

- Describe the motivation for developing and using Web services in business software.
- Describe the Web services architecture.
- Describe common scenarios for Web-service implementation and client-side use.
- Describe the Java Web services architecture and the requirements of J2EE 1.4.
- Understand the importance of SOAP to the Web services architecture.
- Read, understand and write SOAP messages.
- Understand the role of JAXM and SAAJ in building low-level Java Web services.
- Build a Java Web service as a JAXM/SAAJ servlet.
- Implement simple point-to-point SOAP communications from a client application.
- Mix and match SAAJ, SAX and DOM code in a Web-service implementation.
- Understand the role of WSDL in providing type information for Web services.
- Write WSDL documents to describe messages, interfaces and services.
- Understand the role of the JAX-RPC in the Java Web services architecture.
- Identify the alternatives for development paths through Java code and WSDL artifacts on server and client sides, and describe the advantages of each.
- Understand the standard mappings between WSDL, XML Schema and Java.
- Analyze Java domain models and identify the useful JAX-RPC types.
- Add a SOAP interface to an existing Java Web application by generating SOAP messaging code using JAX-RPC tools.
- Build a Java Web service based on an existing WSDL document.
- Build a Java Web-service client based on a WSDL document.
- Describe the relationship between the EJB 2.1 and JAX-RPC 1.0 specifications, and how EJBs can implement Web-service endpoints.
- Add a SOAP interface to an existing system of EJBs, and build an EJB





implementation of a Web service based on a predefined WSDL descriptor.

- Implement a simple Web service using JSP and JSTL XML tags.
- Implement a JSP Web-service client using custom tags that wrap JAXM.
- Understand the lifecycle and context of JAX-RPC services as J2EE components.
- Describe the use of the JAX-RPC message context in managing SOAP headers.
- Implement a JAX-RPC message handler to adapt an existing Web service.
- Implement a session-aware JAX-RPC Web service that relies on HTTP sessions based on cookies.
- Create, send, receive, and read SOAP attachments using SAAJ or JAX-RPC.
- Describe the various techniques for securing Java Web services available from J2EE and various XML specifications.

**Timeline: 5 days.**





## **Chapter 1. Interoperable Web Services**

- Motivation for Web Services
- Evolution of Web Services
- HTTP and XML
- Interoperability Stacks
- Simple Object Access Protocol (SOAP)
- Web Service Description Language (WSDL)
- Universal Description, Discovery and Integration (UDDI)
- The WS-I Basic Profile
- REST

## **Chapter 2. Building and Hosting Web Services**

- Hosting Web Services: Scenarios
- SOAP Alone
- Service Description
- Building Services and Clients from WSDL
- Publishing and Discovery
- Practical Requirements
- The J2EE Reference Implementation
- Demonstration: A Running Web Service and Client
- Sniffing SOAP Messages
- Development Process

## **Chapter 3. The Java Web Services Architecture**

- Web Services and the J2EE
- The Java API for XML Processing (JAXP)
- The Java API for XML Binding (JAXB)
- The SOAP With Attachments API for Java (SAAJ)
- The Java API for XML Messaging (JAXM)
- Demonstration: A SOAP-Based Web Service Using JAXM and SAAJ
- The Java API for XML-Based RPC (JAX-RPC)
- Demonstration: A WSDL-Enabled Web Service Using JAX-RPC
- WSDL-to-Java vs. Java-to-WSDL
- The Java API for XML Registries (JAXR)

## **Chapter 4. The Simple Object Access Protocol (SOAP)**

- SOAP Messaging Model
- SOAP Namespaces
- SOAP over HTTP





- The SOAP Envelope
- The Message Header
- The Message Body
- SOAP Faults
- Attachments
- XML Schema
- Validating Message Content
- SOAP Encoding

### **Chapter 5. The Java APIs for SOAP Messaging (SAAJ)**

- The SAAJ Object Model
- Parsing a SOAP Message
- Reading Message Content
- Working with Namespaces
- Creating a Message
- Setting Message Content
- Integration with the DOM and JAXP

### **Chapter 6. The Java API for XML Messaging (JAXM)**

- Building Low-Level Web Services
- Messaging Scenarios
- Point-to-Point Messaging
- JAXM Message Providers
- JAXM Servlets
- Creating a SOAP Connection
- Sending a Message

### **Chapter 7. Web Services Description Language (WSDL)**

- Web Services as Component-Based Software
- The Need for an IDL
- Web Services Description Language
- WSDL Information Model
- The Abstract Model - Service Semantics
- Message Description
- Messaging Styles
- The Concrete Model - Ports, Services, Locations
- Extending WSDL - Bindings
- Service Description

### **Chapter 8. The Java API for XML-Based RPC (JAX-RPC)**





- The Java Web Services Architecture
- Two Paths
- How It Works - Build Time and Runtime
- The Web Services for J2EE Specification
- JAX-RPC Deployment
- Mapping Between WSDL/XML and Java
- Generating from WSDL
- Generating from Java

### **Chapter 9. Generating Web Services from Java Code**

- The Java-to-XML Mapping
- Primitive Types and Standard Classes
- Value Types and JavaBeans
- The Java-to-WSDL Mapping
- Simple and Complex Types
- Arrays and Enumerations
- Service Endpoint Interface
- Scope of Code Generation
- Inheritance Support
- Multi-Tier Application Design
- Analyzing the Domain
- When Things Don't Fit

### **Chapter 10. Generating Java Web Services from WSDL**

- The XML-to-Java Mapping
- Simple and Complex Types
- Enumerations
- Arrays
- Miscellaneous, Optionally-Supported Constructs
- The WSDL-to-Java Mapping
- Mapping Operation Inputs and Outputs
- Building a Service Client
- Locating a Service
- Client-Side Validation
- Creating a Web Service
- Deploying the Service

### **Chapter 11. Best Practices and Techniques**

- Which Way to Go?
- Interoperability Impact





- Controlling Names and URIs
- Polymorphism in JAX-RPC
- The Dynamic Invocation Interface
- Extensible Type Mapping
- Passing Objects
- Performance Patterns
- Another CORBA?

### **Chapter 12. EJB, JSP and Web Services**

- Enterprise JavaBeans
- Three Tiers for J2EE
- EJB 2.1 and JAX-RPC
- Session Beans as Web Service Endpoints
- The Bean's Service Endpoint Interface
- SOAP as an RMI Transport
- Adding a SOAP Interface to a Session Bean
- Generating From WSDL
- "Gotchas"
- JSP and XML
- The JSTL: Core and XML Actions
- JSP, JSTL and SOAP
- Reading SOAP Using XPath
- Performing XSLT Transformations
- JSPs as Web-Service Clients
- Custom Tags for SAAJ and JAXM

### **Chapter 13. Service Lifecycle and Message Handlers**

- Web Services as J2EE Components
- Service Lifecycle
- Component Environment and JNDI
- Handling SOAP Headers
- Servlet Endpoint Context
- EJB Endpoint Context
- MessageContext and SOAPMessageContext
- Message Handlers and Handler Chains
- Processing Model and Patterns
- Session Management in JAX-RPC

### **Chapter 14. SOAP Attachments**

- SAAJ Object Model, Revisited





- The SOAPMessage Class
- MIME
- The Java Activation Framework
- The MimeHeaders Class
- The AttachmentPart Class
- Adding SOAP Attachments
- Identifying Attachments
- Reading Attachments
- JAX-RPC and Attachments
- Generic Mapping for MIME Types
- Using Images and Binary Types in Interfaces and Structs

### **Chapter 15. Security**

- Web Services and Security
- Threats
- Technology and Techniques
- Public Key Encryption
- Digital Signature
- J2EE Techniques
- Securing Web-Service URIs
- HTTPS
- XML and SOAP Solutions
- XML Encryption and Signature
- WS-Security
- SAML
- XACML

### **Appendix A. Learning Resources**

### **Appendix B. Quick Reference: W3C and Web-Services Namespaces**

### **Appendix C. Basics of XML and XML Schema**

### **System Requirements**

<b>Hardware Requirements (Minimum)</b>	1.0 GHz, 256 meg RAM, 500 meg disk space.
<b>Hardware Requirements (Recommended)</b>	1.5 GHz, 512 meg RAM, 1 gig disk space.
<b>Operating System</b>	Tested on Windows XP Professional. Course software should be viable on all systems which support the J2EE 1.4 reference implementation.
<b>Network and Security</b>	Limited privileges required -- please see our standard security requirements at





## 561. Developing Java Web Services

Outline

### Software Requirements

<http://capcourse.com/Guides/Security.html>.

All free downloadable tools.

