

Judd Solutions

Development, Mentoring and Training



Consuming and Producing Web Services with WST and JST

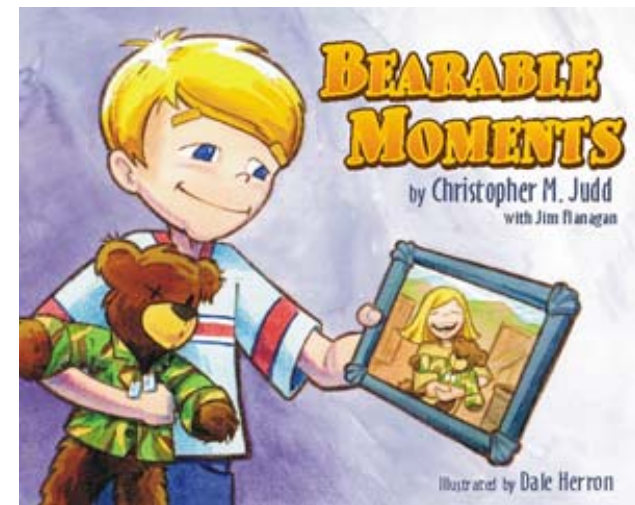
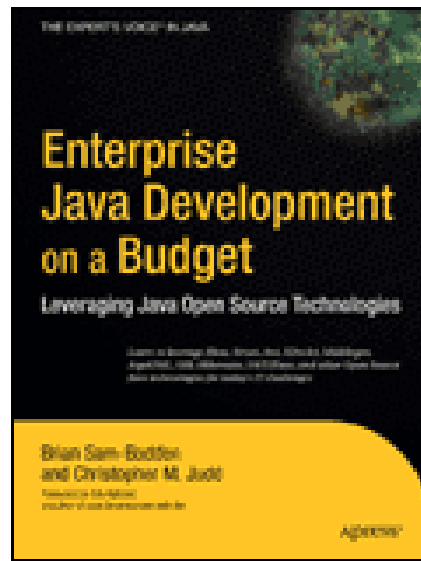
Christopher M. Judd
President/Consultant
Judd Solutions, LLC



Christopher M. Judd



- President/Consultant of Judd Solutions
- Central Ohio Java User Group (COJUG) coordinator





Other WTP Events



- Long Talks

- Developing Data Tools with the DTP SQL Models and Parsers (Tues. 10:45)
- EJB 3.0 Persistence and the Dali EJB ORM Project (Tues. 2:15)
- What's New in Web Tools 1.0 and 1.5 (Thurs. 9:45)
- Using and Extending the Eclipse Web Tools Platform (WTP) (Thurs. 10:45)
- Build JavaServer™ Faces Applications with the JSF Tools Project (Thurs. 2:15)

- Short Talks

- Managing APIs with the Eclipse API Scanner (Wed. 4:33)
- The AJAX Toolkit Framework (Wed. 4:51)
- Authoring in Eclipse (Thurs. 1:27)
- Eclipse and Apache Derby (Thurs. 1:36)

What's new in WTP? Find out at the WTP project sprint!
Mon. 6:30-8:30pm, room 203



Agenda



- WTP Overview
- Web Services Overview
- Consuming
- Producing Web Services (bottom up)
- Testing
- Deployment
- Conclusion



Web Tools Platform Project (WTP)



- Top Level Eclipse Project
- www.eclipse.com/webtools/
- Development tools for web development
 - No runtime dependencies
 - Vendor extensible
- Subprojects
 - Web Standard Tools (WST)
 - J2EE Standard Tools (JST)
 - JavaServer Faces Tools (JSF)
- Version 1.0 was released Dec 20, 2005.
- Dependencies
 - Eclipse 3.1.2
 - Eclipse Modeling Framework (EMF) 2.1.2
 - Graphic Editor Framework (GEF) 3.1.1
 - Java EMF Model (JEM) 1.1.0.1



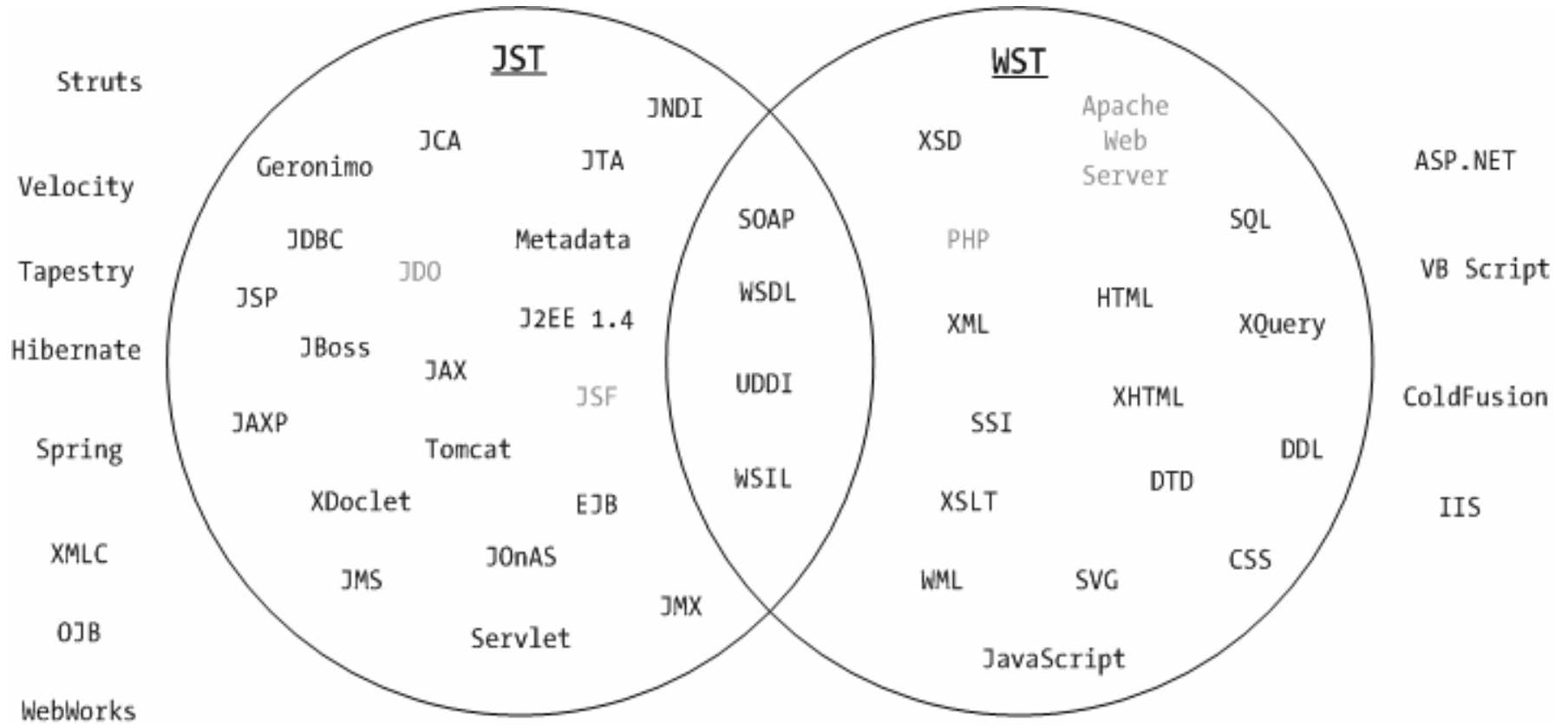
WTP Subprojects



- Web Standard Tools (WST)
 - Web artifacts
 - Defined by open standards bodies
- J2EE Standard Tools (JST)
 - J2EE components
 - Java Community Process (JCP)
 - Depends on WST
- JavaServer Faces Tools (JSF)



WTP Scope





WTP Web Services Scope



- World Wide Web Consortium (W3C)
 - <http://www.w3.org>
 - HTML, XHTML, CSS, XML, XSLT, XML Schema, XML Query
- Organizations for Advancement of Structured Information Standards (OASIS)
 - <http://www.oasis-open.org>
 - e-Business standards for web services
- Web Services Interoperability Organizations (WS-I)
 - <http://www.ws-i.org>
 - Interoperable message exchange between web services
- Java Community Process (JCP)
 - <http://www.jcp.org>
 - Java Web Services APIs



WTP Installation Options



- All-in-one
 - Eclipse 3.1.2
 - Eclipse Modeling Framework (EMF) 2.1.2
 - Graphic Editor Framework (GEF) 3.1.1
 - Java EMF Model (JEM) 1.1.0.1
- Update Manager
 - Eclipse.org update site
- Piecemeal



Agenda



- WTP Overview
- Web Services Overview
- Consuming
- Producing Web Services (bottom up)
- Testing
- Deployment
- Conclusion



Simple Object Access Protocol



From the draft W3C specification:

SOAP is a lightweight protocol for exchange of information in a decentralized, distributed environment. It is an XML based protocol that consists of three parts: an envelope that defines a framework for describing what is in a message and how to process it, a set of encoding rules for expressing instances of application-defined datatypes, and a convention for representing remote procedure calls and responses.

<http://www.w3.org/TR/soap/>



WSDL



- Web Service Definition Language
- Describes
 - What the service can do
 - Where it resides
 - How to invoke it
- Elements
 - Types – data type definition
 - Message – definition of data being communicated
 - Port Type – abstract set of operations
 - Binding – concrete protocol and data format
 - Service – collection of related endpoints
 - Port – binding and a network address



Apache Axis



- Apache Web Services Project
- Open Source
- SOAP implementation
- Version 1.2.1
- <http://ws.apache.org/axis/>



Agenda



- WTP Overview
- Web Services Overview
- Consuming
- Producing Web Services (bottom up)
- Testing
- Deployment
- Conclusion



Google Web Service Client



- Web Service for searching
- <http://www.google.com/apis/index.html>
 - Download developer's kit
 - WSDL file
 - Java wrapper library
 - API documentation
 - Create a Google account
 - License key



Consuming steps



1. Create Java Project
2. Generate Web Service Client from WSDL



Create Project



- Create a standard Java Project
- File > New > Project > Java Project

New Java Project

Create a Java project
Create a Java project in the workspace or in an external location.

Project name: GoogleClient

Contents

Create new project in workspace
 Create project from existing source

Directory: C:\dev\workspaces\eclipse2006\GoogleClient [Browse...](#)

JDK Compliance

Use default compiler compliance (Currently 1.4) [Configure default...](#)
 Use a project specific compliance: 1.4

Project layout

Use project folder as root for sources and class files
 Create separate source and output folders [Configure default...](#)

< Back Next > Finish Cancel



Generate Web Service Client



- File > New > Other > Web Services > Web Service Client
- Specify Java Proxy
- Test application
- Monitor traffic

A screenshot of the 'Web Service Client' dialog box. The title bar reads 'Web Service Client'. The main heading is 'Web Services'. Below the heading is a sub-heading 'Web Services' and a paragraph: 'Review your Web service options and make any necessary changes before proceeding to the next page.' There is a small icon of a bell and a document. The dialog contains several options:

- 'Client proxy' section with a dropdown menu set to 'Java Proxy'.
- An unchecked checkbox: 'Install Web service client on server (managed clients only)'.
- Five other checkboxes: 'Test the Web service', 'Monitor the Web service', 'Overwrite files without warning', 'Create folders when necessary' (checked), and 'Check out files without warning'.

At the bottom are four buttons: '< Back', 'Next >', 'Finish', and 'Cancel'.



Generate Web Service Client



- Enter location of WSDL locally or via URL

Web Service Client

Web Service Selection Page

Enter a web service URI.

Enter an URI to a WSDL, WSIL or HTML document:

Select a WSDL

file:/C:/dev/workspaces/eclipse2006/GoogleClient/GoogleSearch.wsdl

WSDL validation messages:

seve...	line	colu...	message

< Back Next > Finish Cancel



Generate Web Service Client



- Specify project to generate the code in

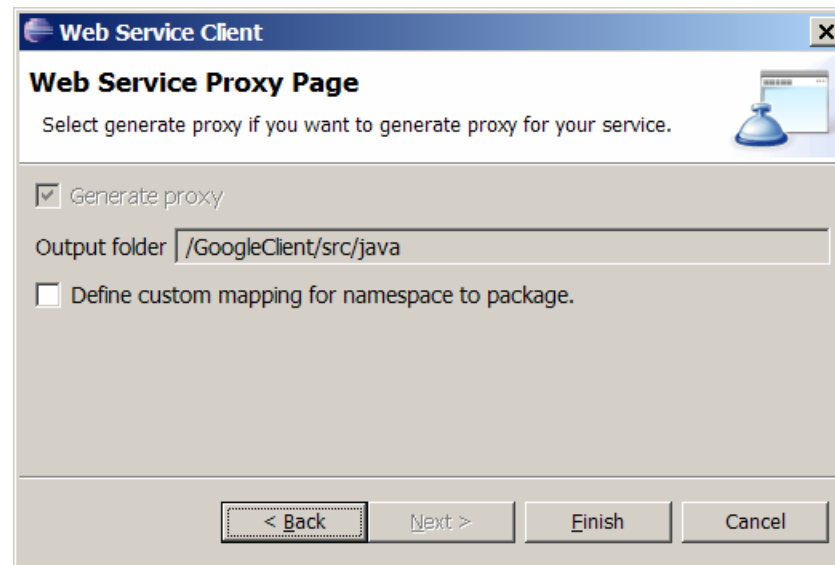
A screenshot of a 'Web Service Client' configuration dialog box. The title bar reads 'Web Service Client'. The main heading is 'Client Environment Configuration'. Below this, there is a sub-heading 'Client-Side Environment Selection:' followed by 'Web service runtime: Apache Axis' and 'Server: Apache Tomcat v5.0'. There is an 'Edit...' button next to the server information. Below these are three dropdown menus: 'Client Project type:', 'Client project: GoogleClient', and 'Client EAR project:'. At the bottom, there are four buttons: '< Back', 'Next >', 'Finish', and 'Cancel'. The 'Next >' button is highlighted with a dotted border.



Generate Web Service Client



- Optionally, can specify namespace to package mapping



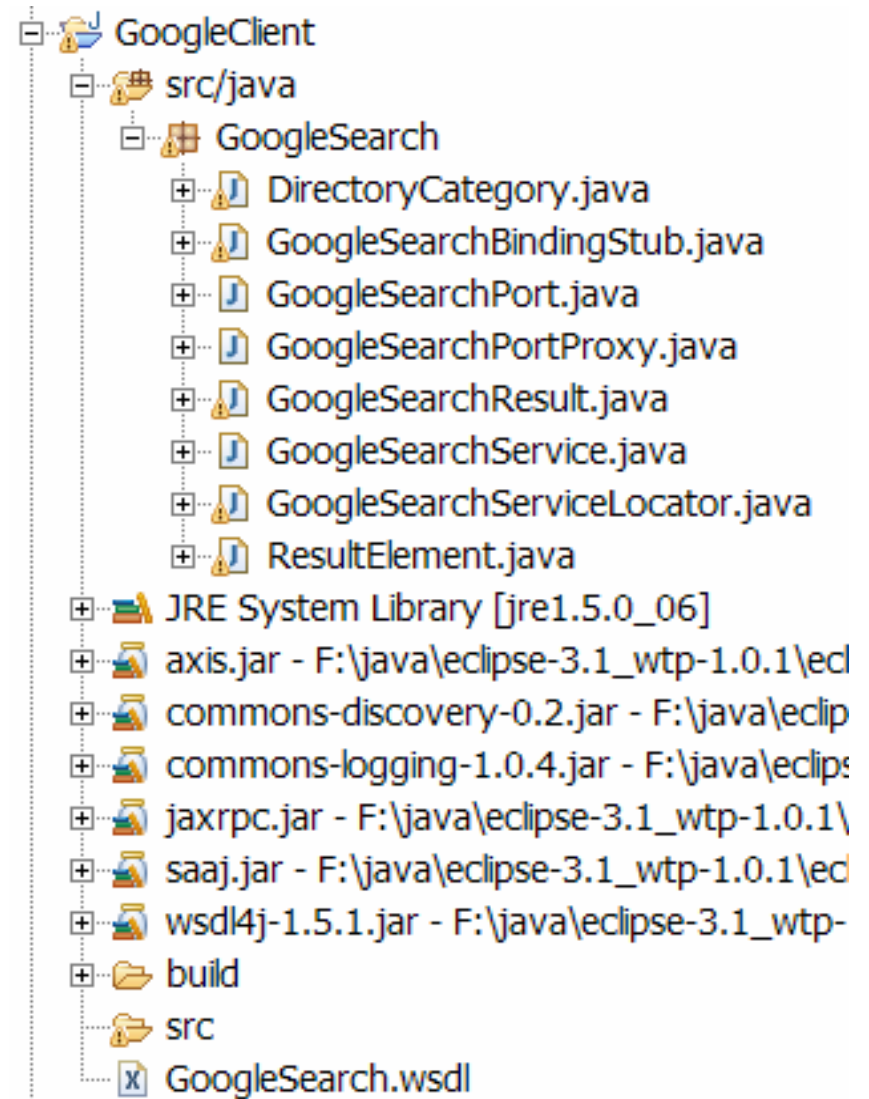
The image shows a screenshot of a 'Web Service Client' dialog box. The title bar reads 'Web Service Client'. The main heading is 'Web Service Proxy Page'. Below the heading, there is a text prompt: 'Select generate proxy if you want to generate proxy for your service.' There are two checkboxes: the first is checked and labeled 'Generate proxy'; the second is unchecked and labeled 'Define custom mapping for namespace to package.'. Below the checkboxes is a text input field for 'Output folder' containing the path '/GoogleClient/src/java'. At the bottom of the dialog, there are four buttons: '< Back', 'Next >', 'Finish', and 'Cancel'.



Output



- Classes
 - DictionaryCategory
 - GoogleSearchBindingStub
 - GoogleSearchPort
 - GoogleSearchPortProxy
 - GoogleSearchResult
 - GoogleSearchService
 - GoogleSearchServiceLocator
 - ResultElement
- Axis Jars
 - axis.jar
 - jaxrpc.jar
 - saaj.jar
 - wsdl4j-1.5.1.jar
 - commons-discovery-0.2.jar
 - commons-logging-1.0.4.jar





Using Generate Classes



```
GoogleSearchPortProxy proxy = new GoogleSearchPortProxy();
GoogleSearchResult result = proxy.doGoogleSearch(
    "Ocmw4/RQFHIX511w/c4Ww5fu*****", // key
    "Bearable Moments", // query terms
    0, // start index
    10, // max results (<= 10)
    true, // filter similar results
    "", // subset restrict
    true, // safe search
    "", // language restrict
    "UTF-8", // input encoding
    "UTF-8"); // output encoding

ResultElement[] resultElements = result.getResultElements();
for (int i = 0; i < resultElements.length; i++) {
    ResultElement element = resultElements[i];
    System.out.println(element.getTitle());
    System.out.println("  " + element.getURL());
}
```



Agenda



- WTP Overview
- Web Services Overview
- Consuming
- Producing Web Services (bottom up)
- Testing
- Deployment
- Conclusion



Producing Web Services



- Bottom up
 - Java code -> Generate WSDL
- Top down
 - WSDL -> Generate Java code



Bottom Up Steps



1. Create Dynamic Web Project
2. Create Service and DTOs
3. Generate Web Service



Create Dynamic Web Project



- File > New > Other > Web > Dynamic Web Project
- Project Name
- Target server

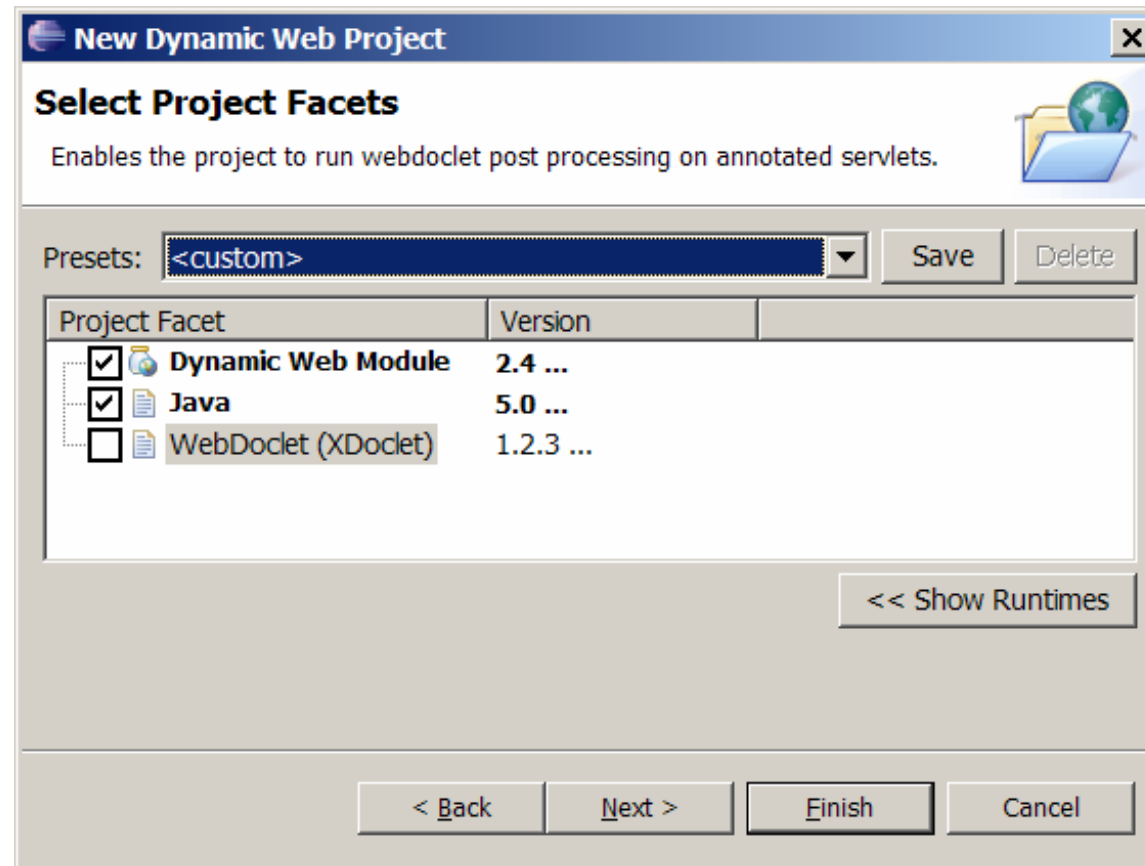
A screenshot of the 'New Dynamic Web Project' dialog box in the Eclipse IDE. The dialog has a title bar with the text 'New Dynamic Web Project' and a close button. Below the title bar, there is a section titled 'Dynamic Web Project' with a small globe icon and the text 'Create a standalone Dynamic Web project or add it to a new or existing Enterprise Application.' The main area contains several fields and options: 'Project Name:' with the text 'Order' entered; 'Project contents' section with a checked checkbox 'Use default' and a 'Directory:' field containing 'C:\dev\workspaces\eclipse2006\Order' and a 'Browse...' button; 'Target runtime:' dropdown menu showing 'Apache Tomcat v5.0' and a 'New...' button; an unchecked checkbox 'Add project to an EAR'; and an 'EAR Project Name:' dropdown menu showing 'EAR' and a 'New...' button. At the bottom, there are four buttons: '< Back', 'Next >', 'Finish', and 'Cancel'.



Create Dynamic Web Project



- Facets and versions





Create Dynamic Web Project



- Context Root
- Directories

The image shows a 'New Dynamic Web Project' dialog box with the following fields and controls:

- Web Module**: Configure web module settings. (Icon: folder with globe)
- Context Root:**
- Content Directory:**
- Java Source Directory:**
- Buttons: < Back, Next >, Finish, Cancel



Create Service



```
public class Order {  
  
    private static final float DISCOUNT = 0.9f;  
  
    public float quote(float price, int quantity) {  
        if (quantity > 100) {  
            price = price * DISCOUNT;  
        }  
        return price * quantity;  
    }  
}
```



Generate Web Service



- New > File > Other > Web Services > Web Service
- Right click on Class > Web Services > Create Web Services

Web Service

Web Services

Review your Web service options and make any necessary changes before proceeding to the next page.

Service

Web service type: Bottom up Java bean Web Service

Install Web service on server

Start Web service in Web project

Launch the Web Services Explorer to publish this Web service to a UDDI Registry

Generate a proxy

Client proxy

Client proxy type: Java Proxy

Install Web service client on server (managed clients only)

Test the Web service

Monitor the Web service

Overwrite files without warning

Create folders when necessary

Check out files without warning

Do not show me this dialog box again.

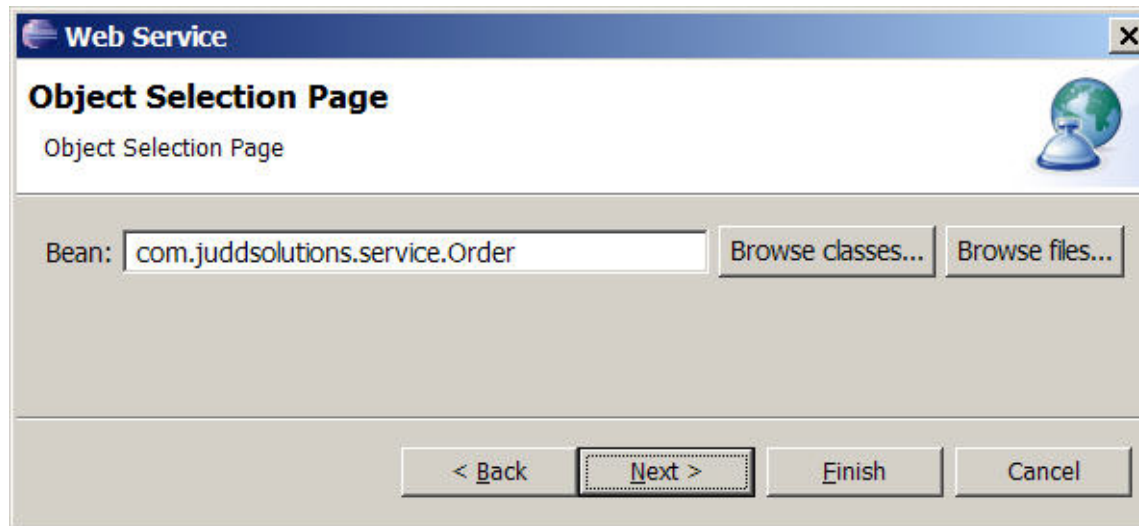
< Back Next > Finish Cancel



Generate Web Service



- Select class containing services to expose if not selected already





Generate Web Service



- Select project for deployment descriptors
- Create new client project

Web Service

Service Deployment Configuration

Choose from the list of runtimes and deployment servers, or use the default settings.

Select the service project and the EAR project with which you want it to be associated. If an EAR or project does not exist or is currently unassociated, it will be created and associated as required when you click Next.

Server-Side Deployment Selection:

Web service runtime: Apache Axis
Server: Tomcat v5.0 Server

Service Project type:
Service project: Order
Service EAR project:

Client-Side Environment Selection:

Web service runtime: Apache Axis
Server: Tomcat v5.0 Server

Client Project type: Dynamic Web Project
Client project: OrderClient
Client EAR project:

< Back **Next >** Finish Cancel



Generate Web Service



- Select methods to expose
- Select type
 - RPC
 - Performance
 - Simple types
 - Document
 - Interoperability
 - XML

Web Service

Web Service Java Bean Identity

Configure the Java bean as a Web service.

Web service URI:

WSDL Folder:

WSDL File:

Methods

quote(float,int)

Select All Deselect All

Style and Use

Document/ Literal (Wrapped)

Document/ Literal

RPC/ Encoded

Define custom mapping for package to namespace.

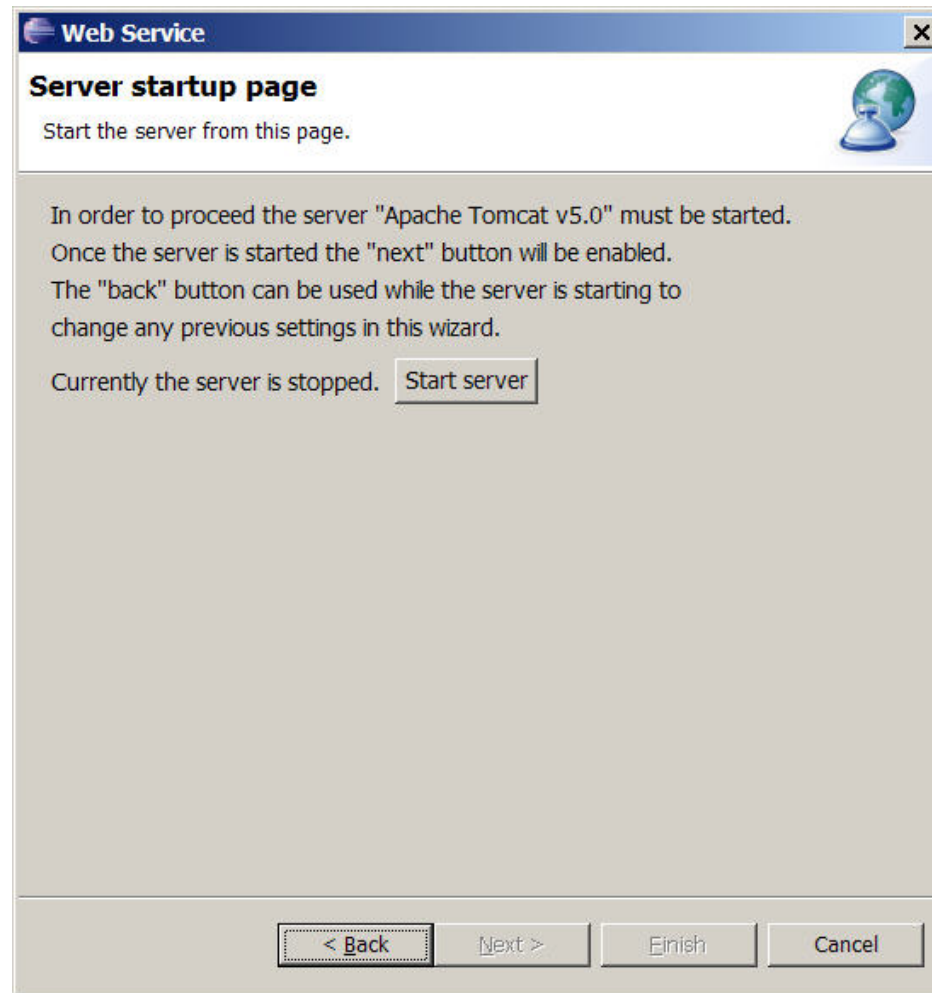
< Back Next > Finish Cancel



Generate Web Service



- Start server to host web service and test client

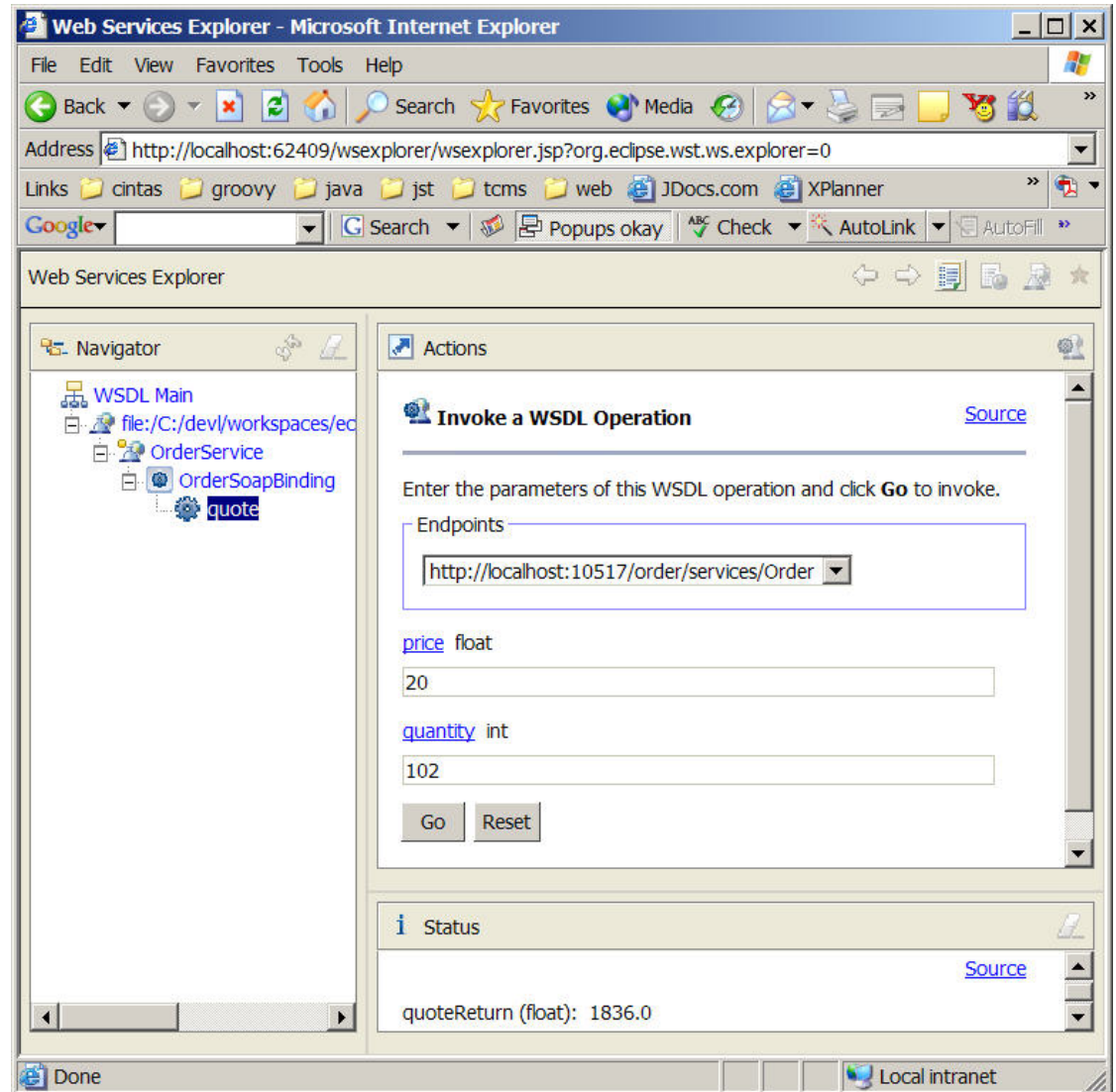
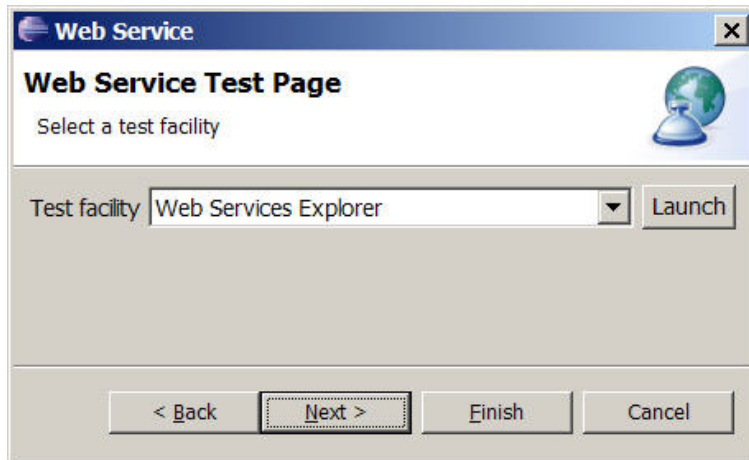




Generate Web Service



- Test early





Generate Web Service



- Directory for client code
- Optionally, provide namespace to package mapping

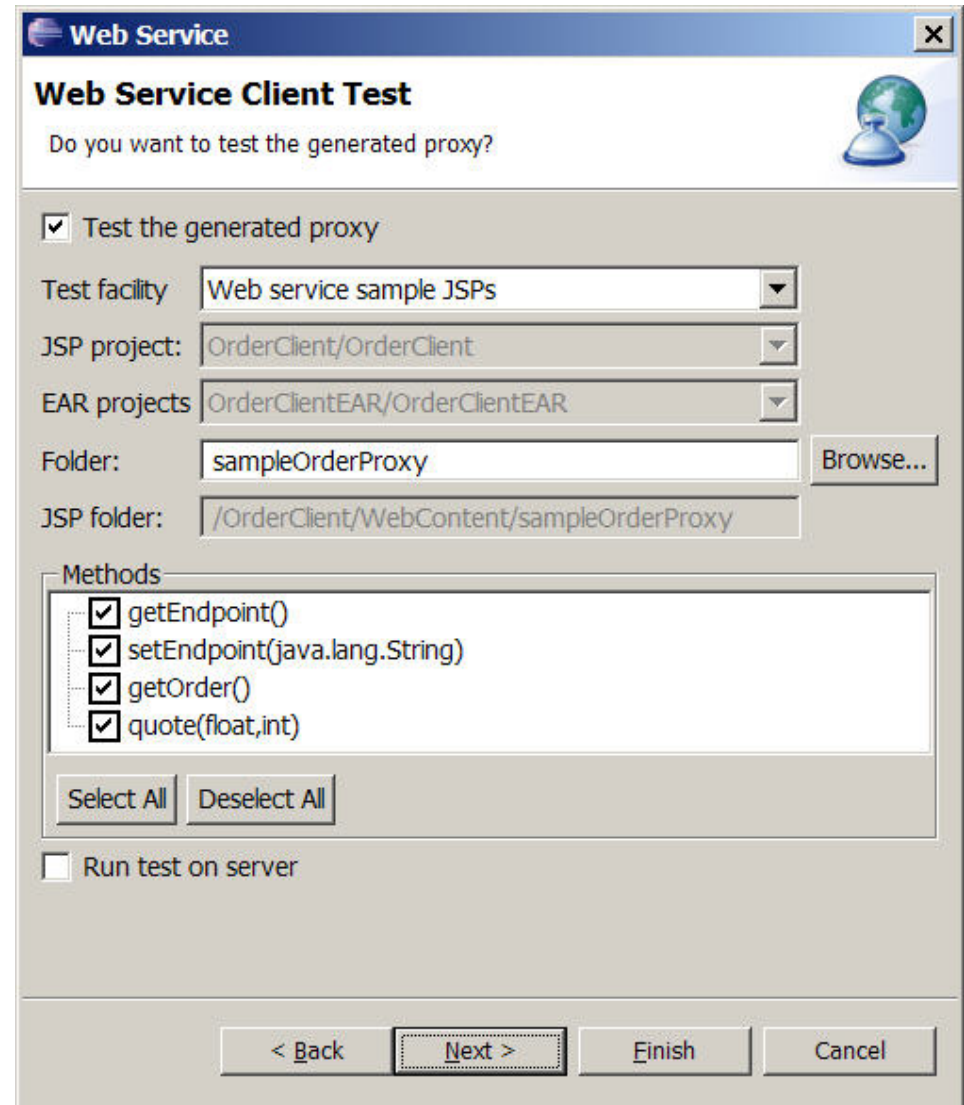
A screenshot of a 'Web Service Proxy Page' dialog box. The title bar reads 'Web Service'. The main heading is 'Web Service Proxy Page' with a globe icon. Below the heading, it says 'Select generate proxy if you want to generate proxy for your service.' There are two checkboxes: 'Generate proxy' (checked) and 'Define custom mapping for namespace to package.' (unchecked). An 'Output folder' text box contains '/OrderClient/src'. At the bottom, there are four buttons: '< Back', 'Next >', 'Finish', and 'Cancel'. The 'Next >' button is highlighted with a dotted border.



Generate Web Service



- Methods to test from test client



Web Service

Web Service Client Test

Do you want to test the generated proxy?

Test the generated proxy

Test facility: Web service sample JSPs

JSP project: OrderClient/OrderClient

EAR projects: OrderClientEAR/OrderClientEAR

Folder: sampleOrderProxy

JSP folder: /OrderClient/WebContent/sampleOrderProxy

Methods

- getEndpoint()
- setEndpoint(java.lang.String)
- getOrder()
- quote(float,int)

Run test on server



Generate Web Service



- Publish web service to UDDI registry

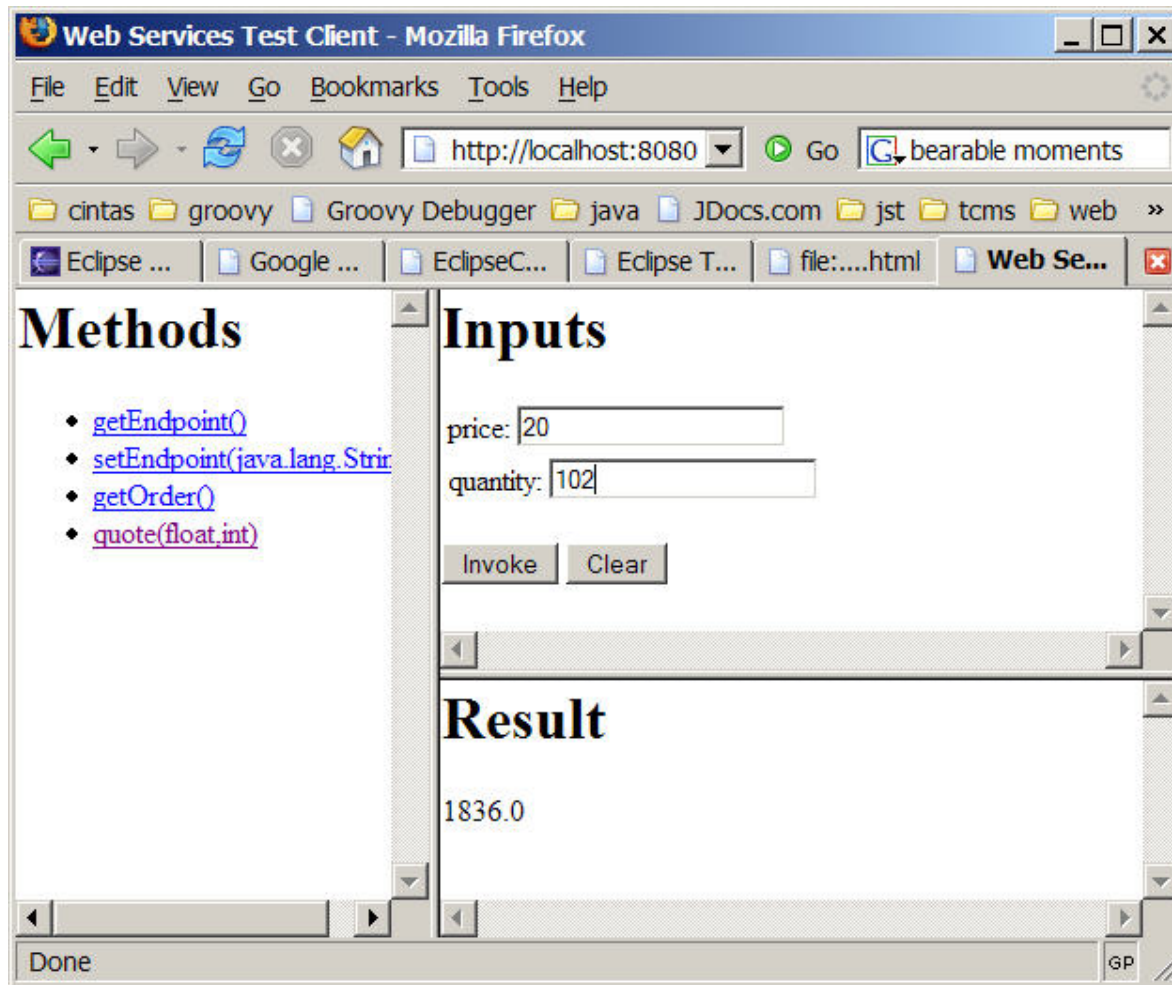
The image shows a Windows-style dialog box titled "Web Service". The main heading is "Web Service Publication" with a globe icon. Below the heading, it asks "Do you want to publish your Web service?". There are two unchecked checkboxes: "Launch the Web Services Explorer to publish this Web service to the Unit Test UDDI Registry" and "Launch the Web Services Explorer to publish this Web service to a UDDI Registry". Below these is a dropdown menu labeled "Public UDDI Registry" with "IBM UDDI Test Registry" selected. At the bottom, there are four buttons: "< Back", "Next >", "Finish", and "Cancel".



Generate Web Service



- Test Client

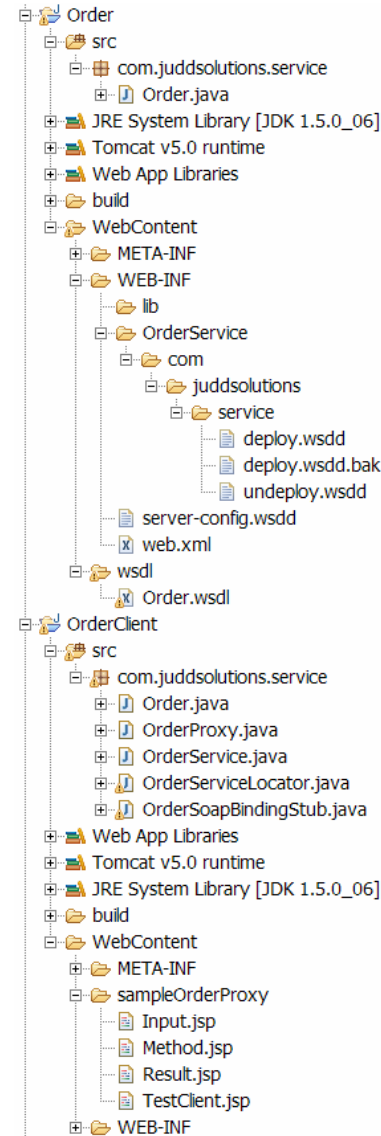




Output



- Web Service
 - Axis deployment descriptors
 - WSDL
- Test Client
 - Dynamic web app
 - Proxy
 - JSPs





Data Types



- Simple Java data types
- JavaBeans
 - No parameter constructor
- Arrays



Agenda



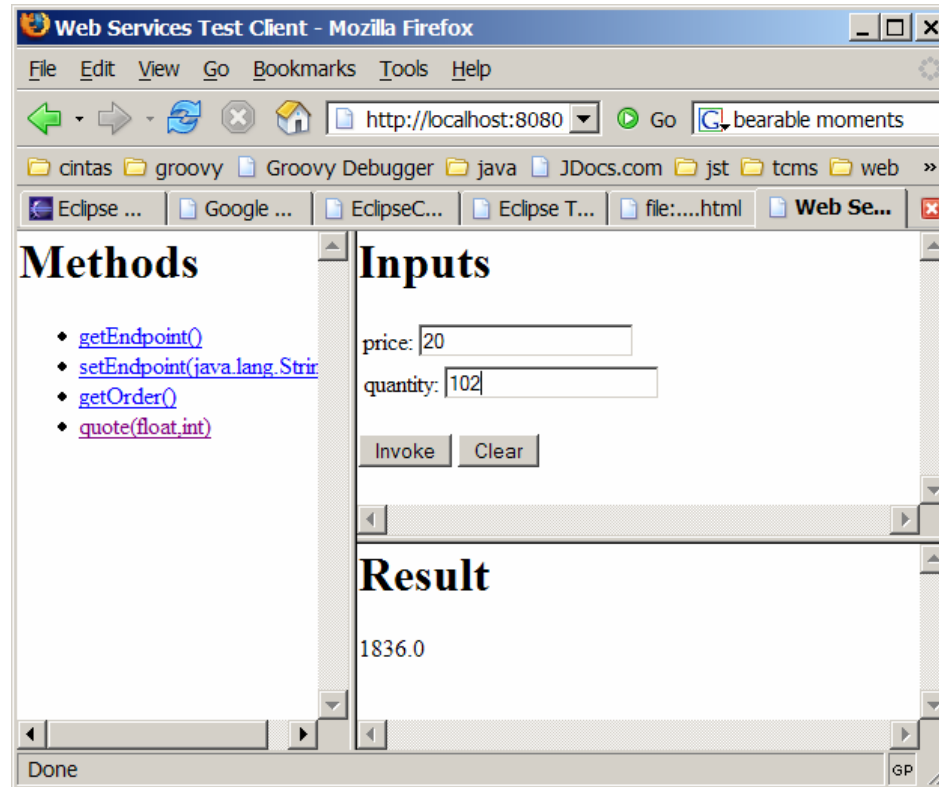
- WTP Overview
- Web Services Overview
- Consuming
- Producing Web Services (bottom up)
- Testing
- Deployment
- Conclusion



Test Client



- <http://localhost:8080/OrderClient/sampleOrderProxy/TestClient.jsp?endpoint=http://localhost:10517/order/services/Order>





TCP/IP Monitoring



- View SOAP request and response

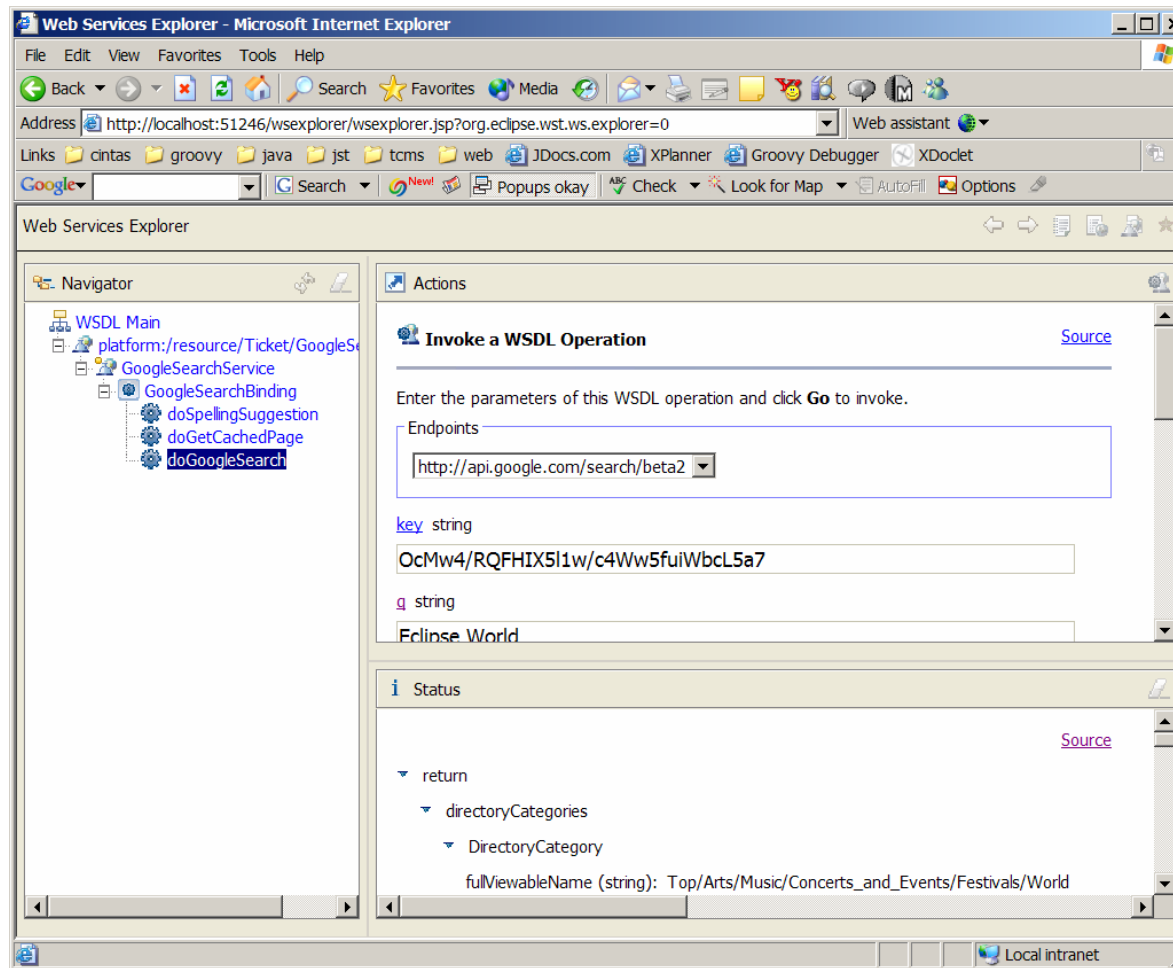
The screenshot shows the TCP/IP Monitor tool interface. The top menu bar includes Problems, Javadoc, Declaration, Console, Snippets, Properties, Servers, and TCP/IP Monitor. The main window displays a tree view of the network path: localhost:8209 > /orderWS/services/Order. The right-hand pane shows request details: Time of request: 5:22.10.996 AM, Response Time: 831 ms, and Type: HTTP. Below this, the request and response are displayed in two side-by-side text areas. The request area shows: Request: localhost:8209, Size: 823 (1104) bytes, Header: POST /orderWS/services/Order HTTP/1.0, and the XML body: <?xml version="1.0" encoding="UTF-8"?>. The response area shows: Response: localhost:8080, Size: 463 (603) bytes, Header: HTTP/1.1 200 OK, and the XML body: <?xml version="1.0" encoding="utf-8"?>. Both text areas have a 'Byte' dropdown menu above them.



Web Service Explorer Testing



- Test any Web Services

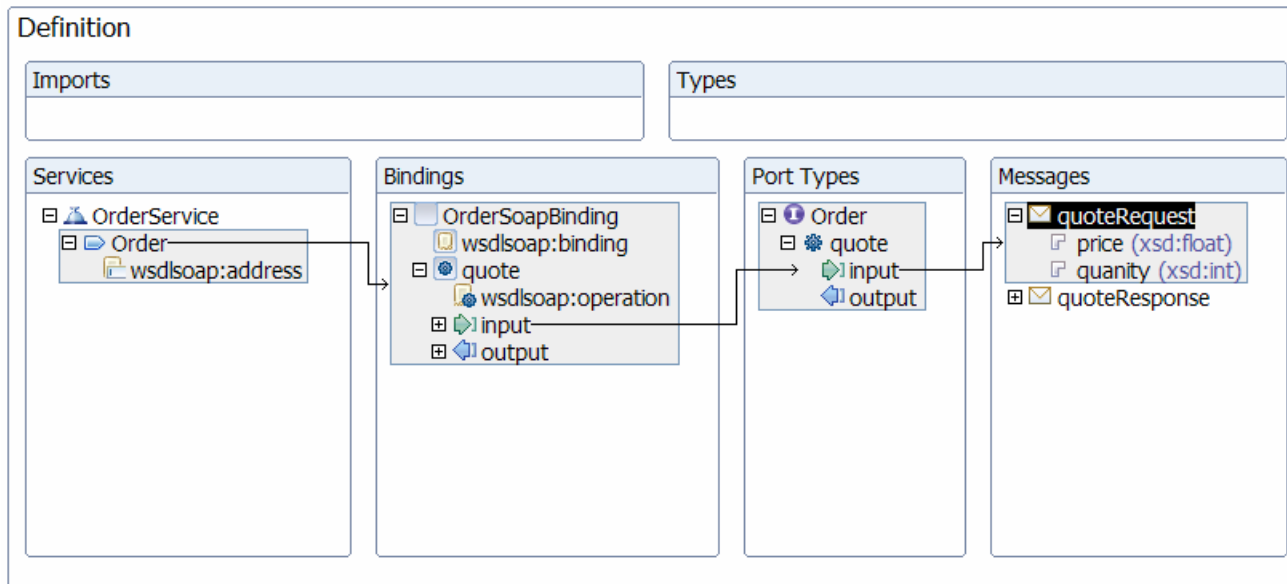




WSDL Editor



- Graphically edit WSDL





Agenda



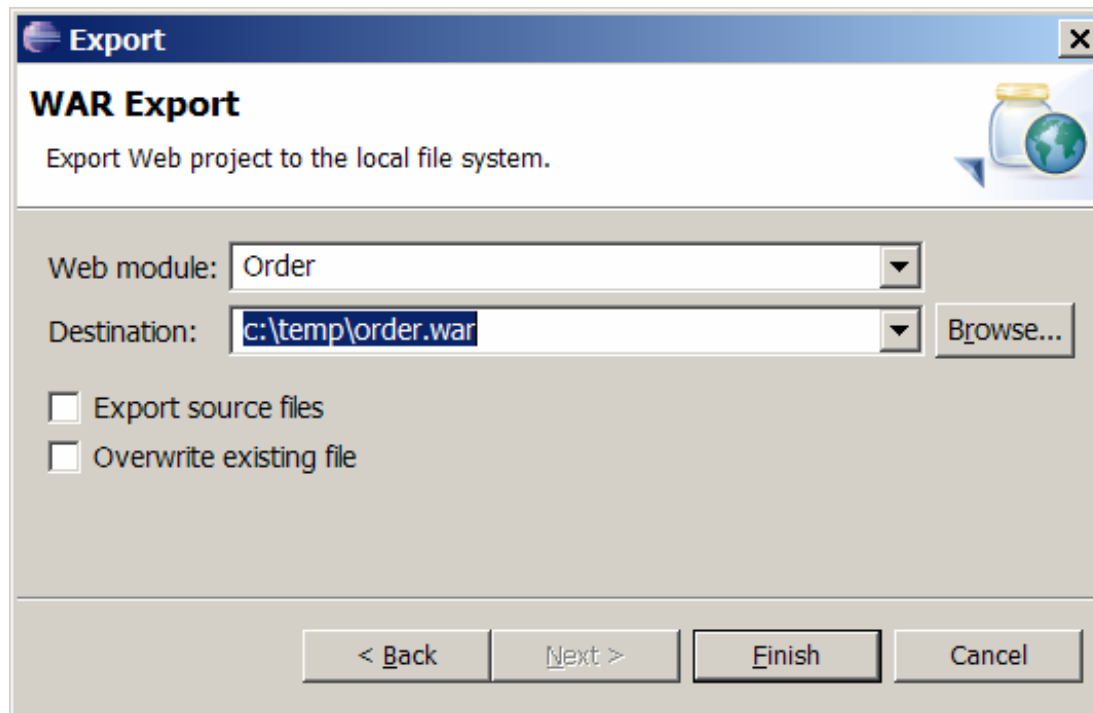
- WTP Overview
- Web Services Overview
- Consuming
- Producing Web Services (bottom up)
- Testing
- Deployment
- Conclusion



Package



- File > Export > WAR file



The image shows a screenshot of an 'Export' dialog box titled 'WAR Export'. The dialog box has a blue header bar with the title 'Export' and a close button. Below the header, the title 'WAR Export' is displayed in bold, followed by the instruction 'Export Web project to the local file system.' and a small icon of a jar and a globe. The main area of the dialog contains two dropdown menus: 'Web module:' with 'Order' selected, and 'Destination:' with 'c:\temp\order.war' selected. To the right of the 'Destination:' dropdown is a 'Browse...' button. Below these fields are two checkboxes: 'Export source files' and 'Overwrite existing file', both of which are currently unchecked. At the bottom of the dialog, there are four buttons: '< Back', 'Next >', 'Finish', and 'Cancel'.



Deployment



- Container specific
- Admin console
- Deployment directory
- Ant task
- Server View



Agenda



- WTP Overview
- Web Services Overview
- Consuming
- Producing Web Services (bottom up)
- Testing
- Deployment
- Conclusion



WTP Resources



- www.eclipse.org/webtools/
- www.projst.com
- Tutorials
 - <http://www.eclipse.org/webtools/community/communityresources.html#tutorials>
- Articles
 - <http://www.eclipse.org/webtools/community/communityresources.html#articles>
 - Build rich Internet applications - <http://www-128.ibm.com/developerworks/edu/os-dw-os-laszlo-i.html>
- New Group
 - <news://news.eclipse.org/eclipse.webtools>



Project Antoine



Help improve the Eclipse WTP user experience!

- What is this study about?
 - UBC and IBM are conducting a usage study to improve the organization of commands and functions in Eclipse based projects.
- How can you take part?
 - Simply install the study plug-in and use Eclipse WTP for your daily work.

For more information, or to sign up, please visit:
<http://www.cs.ubc.ca/~lkf/antoine>



Contact Information



- <http://www.juddsolutions.com>
- cjudd@juddsolutions.com
- Blog
 - <http://blogs.apress.com/authors.php?author=Christopher%20Judd>
- Pro Eclipse JST
 - <http://www.projst.com>
 - <http://www.apress.com/book/bookDisplay.html?bID=447>
- Enterprise Java Development on a Budget
 - <http://www.apress.com/book/bookDisplay.html?bID=197>





Questions ?

Please fill out your evaluations.