IBM Software Group

Web Tools: Beyond the Basics in IBM WebSphere Development Studio Client for iSeries

iSeries Application Development Team: IBM Toronto

WebSphere. software

business software

Agenda

- J2EE
  - A deeper look into J2EE
  - JDBC
  - J2EE Connector Architecture
  - Java Naming and Directory Interface
- Web Tooling
  - Web projects
  - J2EE Navigator and Hierarch Views
  - Cascading Style Sheets
  - Struts
- Server Tooling
  - Creating new server configurations in the test environment
  - Configuring the test environment

Introduction

- There are many different technologies at your disposal when creating Web applications (and lots of acronyms to go with them!)
  - J2EE, JDBC, JCA, JMS, CSS, HTML, JSP, WAS, …
  - Each has it's own useful purpose
- However, some are more common than others
  - CSS – Cascading Style Sheets for defining a consistent look and feel across all your web pages
  - HTML and JSP – Replace DDS as way to define the user interface
  - Struts – Great architecture to follow for your overall Web application
  - JDBC – Database access and stored procedure call using SQL
  - JCA – Java Connector Architecture for calling iSeries programs and service programs
- Purpose of this presentation is to cover the more common ones in greater detail and provide you with a foundation to explore the others

J2EE – The Technologies

J2EE - Components

- There are 4 main pieces to J2EE application model
  - Components
  - Containers
  - Services
  - Connectors
- Components (Modules)
  - You develop your code as components of a J2EE application
  - Many different types of components
    - Applets
    - Application clients (full graphical client)
    - Enterprise JavaBeans components (business logic)
    - Web components

J2EE - Containers

- Containers
  - Components run inside of a container
  - Containers are typically provided by system vendors like IBM
    - Web and EJB containers are provided with WebSphere Application Server
  - Provide services that can be used by used by the components which run in the container
    - Transaction support
    - Resource pooling
      - database connections
    - Often allow component behavior to be specified at deployment time instead of development time
      - Configuring which database to access
      - Maximum number of database connections

J2EE - Services

- Service Technologies
  - The J2EE specification defines standard APIs to access many common services
    - JDBC
      - Database-independent method for using SQL
      - Database provides provide JDBC drivers
        - IBM, Microsoft, Oracle, …
    - Java Transaction API
    - Naming Service
      - Java Naming and Directory Interface (more on this later)
    - J2EE Connector Architecture
    - Java Message Service (JMS)
    - There are others, but these are the main ones

J2EE – Connector Architecture

- J2EE Connector Architecture
  - Provides a standard / portable API to use in Java components to access Enterprise Information Systems (EIS)
  - Typically provided by the EIS vendor
    - IBM provides connectors for
      - Calling an RPG or COBOL program
      - Accessing CICS

Everything is Components, Containers, Services and Connectors

You develop your components using the help of the services and connectors then deploy to a container!

JDBC
J2EE Connector Architecture (JCA)
Java Naming and Directory Interface (JNDI)

WebSphere. software

e business software

JDBC

- Standard Java interface for running SQL
  - Independent of any single Database vendor
  - Works with DB2 UDB, Cloudscape, Informix, Microsoft SQL Server, Oracle, Sybase, …
  - Lots of JDBC articles, books, web sites, ...
- Development time:
  - You write the code using JDBC and standard SQL
- Deployment time:
  - You specify which Database to use
  - This is configured in the "Web Deployment Descriptor" for you Web project (web.xml)
    - More on this latter!

JDBC Java Interfaces

- Use JDBC to:
  - Directly read, write and update DB2 UDB for iSeries using SQL
  - Call stored procedures written using RPG, COBOL or Java
- JDBC Terms
  - Connection (`java.sql.Connection`)
    - Live connection (session) with a specific database
    - Statements are associated with a Connection
  - Statement (`java.sql.Statement`)
    - Java interface used for executing SQL
  - PreparedStatement (`java.sql.PreparedStatement`)
    - Same as Statement, except it is precompiled for performance
    - Use PreparedStatement if you are running the same statement multiple times
  - CallableStatement (`java.sql.CallableStatement`)
    - Java interface used for calling stored procedures

JDBC Connection Pooling

- Problem:
  - In a typical web application there maybe 1000s of requests coming in every minute
  - There is a lot of overhead to create and close a connection to the database for every request
  - But you need a connection to run SQL queries
- Solution:
  - Use Connection pooling
  - Web App container creates JDBC connections in a pool
  - Instead of creating a Connection in your code you:
    - Ask the pool for a connection
    - Use the connection to run SQL statements
    - Return the connection to the pool so it can be reused

J2EE Connector Architecture (JCA)

- J2EE Connector Architecture provides a standard architecture for accessing various Enterprise Information Systems (EIS) from your Java application
  - RPG and COBOL programs or service programs
  - CICS
  - Enterprise Resource Planning (ERP) systems
- Resource Adapters
  - Provided by each vendor for their EIS system
  - Plugs into the application server and handles things like:
    - Communications
    - Transactions
    - Security
  - WDSC provides resource adapter for calling RPG and COBOL programs
    - D:\WDSC\iseries\eclipse\plugins\com.ibm.etools.iseries.webtools_5.0.1\lib\iseriespgmcall.rar

JNDI – How To Find Things

- Problem:
  - Many of the components to a web application are distributed across multiple servers
  - Components need to be dynamically changed or updated without having to modify the code and recompile
    - Changing a JDBC Database connection from the development database to the production database
- This is the domain of enterprise naming and directory servers
- Many different naming and directory server packages available
  - Need a standard way to interface with them so the code is not directly tied to a specific vendor's implementation

JNDI – How To Find Things

- Solution
  - Java Naming and Directory Interface (JNDI)
  - Allows developer to write programs that can lookup resources dynamically at runtime
    - Can easily change which database is used for JDBC without recompiling
  - Standard extension to the Java platform for connecting to and interfacing with naming and directory servers
- Use JNDI to locate other J2EE resources:
  - Database connections (JDBC)
  - RPG or COBOL program call resource adapters (JCA)
  - Message queue (JMS)
- JNDI is just an interface for locating services / components

Programming With JNDI

- Two steps to working with JNDI
  - Writing you application using JNDI to locate components
    - Or have one of the wizards generate the code
      - iSeries Program Call wizard
      - Database pages wizard
  - Configuring the naming and directory server with the components you need at runtime
    - For example: Define the Database connection and pooling information
    - In a Web application with is done by configuring the properties for the Web application server
      - More on this later…

Code Example: Using JNDI to Lookup JDBC Connection

```
// import JDBC Interfaces
import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.Statement;

// import JNDI classes and interfaces
import javax.naming.Context;
import javax.naming.InitialContext;

...

// Retrieve JNDI context
Context initialContext = new InitialContext();
// Lookup JDBC DataSource using JNDI
DataSource datasource = (DataSource)
  initialContext.lookup("jdbc/customer");

// Use JDBC Data Source to run SQL query
Connection connection = datasource.getConnection();
Statement statement = connection.createStatement();
ResultSet results = statement.executeQuery("SELECT * FROM
  CUSTOMER");

// Do something with the result set
```

J2EE – The Tools

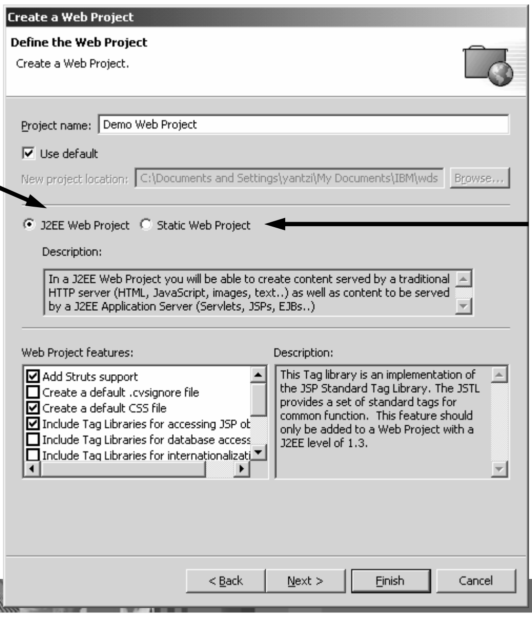
WebSphere. software

e business software

New Web Project Wizard

Use J2EE Web Project for Web applications with **dynamic content** (servlets and JSPs)

Checkboxes to automatically add Struts support and Tag libraries to your new Web Project



Create a Web Project

Define the Web Project

Create a Web Project.

Project name: Demo Web Project

[✓] Use default

New project location: C:\Documents and Settings\yantzi\My Documents\IBM\wds

(●) J2EE Web Project ( ) Static Web Project

Description:

In a J2EE Web Project you will be able to create content served by a traditional HTTP server (HTML, JavaScript, images, text..) as well as content to be served by a J2EE Application Server (Servlets, JSPs, EJBs..)

Web Project features:

- [✓] Add Struts support
- [ ] Create a default .cvsignore file
- [✓] Create a default CSS file
- [✓] Include Tag Libraries for accessing JSP ob
- [ ] Include Tag Libraries for database access
- [ ] Include Tag Libraries for internationalizati

Description:

This Tag library is an implementation of the JSP Standard Tag Library. The JSTL provides a set of standard tags for common function. This feature should only be added to a Web Project with a J2EE level of 1.3.

Use “Static Web Project” for Web sites with **no dynamic content** (just HTML and graphics)

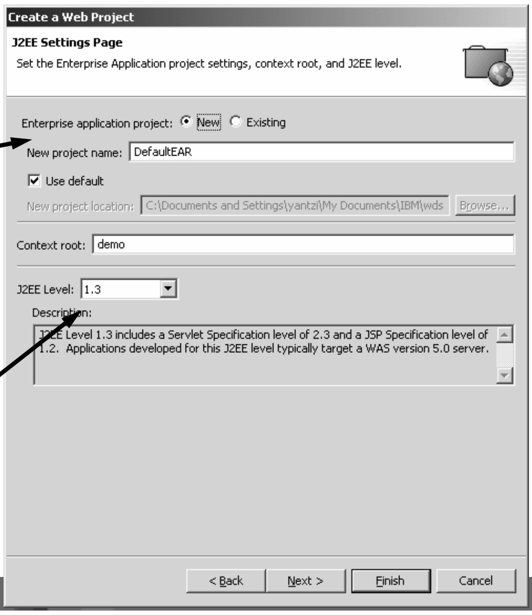
New Web Project Wizard - 2

Each J2EE Web project must be associated with an **Enterprise application project**

Enterprise application projects are associated with test environment server configurations (for testing your Web Application)

Use J2EE 1.3 with WebSphere Application Server 5.0

Use J2EE 1.2 with WebSphere Application Server 4.0



J2EE Settings Page

Set the Enterprise Application project settings, context root, and J2EE level.

Enterprise application project: [✓ New] Existing

New project name: DefaultEAR

[✓] Use default

New project location: C:\Documents and Settings\yantzi\My Documents\IBM\wds

Context root: demo

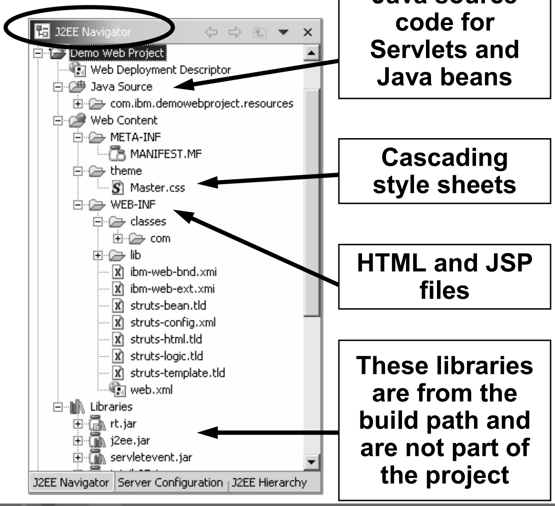
J2EE Level: 1.3

Description:

J2EE Level 1.3 includes a Servlet Specification level of 2.3 and a JSP Specification level of 1.2. Applications developed for this J2EE level typically target a WAS version 5.0 server.

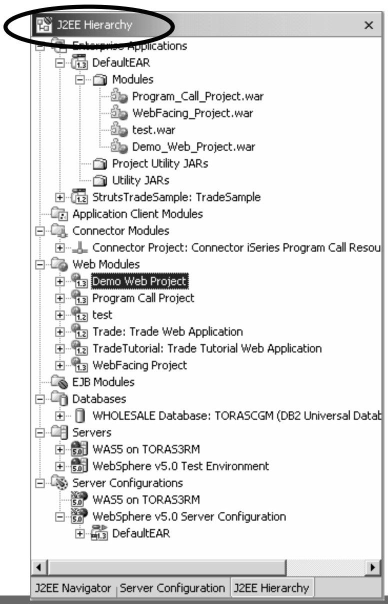
J2EE Navigator View

- The Web project that gets created complies (and enforces) the J2EE standard Web application structure
- J2EE Navigator view
  - Shows all J2EE related projects
    - Web Projects
    - EJB Projects
    - Connector Projects
    - Server Projects
  - Provides an easy way to manage your J2EE projects
    - Edit files
    - Copy, move rename and delete files



J2EE Hierarchy View

- J2EE Hierarchy view
  - Shows all of your J2EE resources in the workspace
    - Enterprise application projects
    - Web Projects (modules)
    - Enterprise JavaBean projects (modules)
    - Databases
    - Connectors
    - Test environment servers and configurations
  - Displays resources in their hierarchy and how they are related
  - Does not show projects



```
J2EE Hierarchy
Enterprise Applications
  DefaultEAR
    Modules
      Program_Call_Project.war
      WebFacing_Project.war
      test.war
      Demo_Web_Project.war
    Project Utility JARs
    Utility JARs
  StrutsTradeSample: TradeSample
Application Client Modules
Connector Modules
  Connector Project: Connector iSeries Program Call Resou
Web Modules
  Demo Web Project
  Program Call Project
  test
  Trade: Trade Web Application
  TradeTutorial: Trade Tutorial Web Application
  WebFacing Project
EJB Modules
Databases
  WHOLESALE Database: TORASCGM (DB2 Universal Datab
Servers
  WAS5 on TORAS3RM
  WebSphere v5.0 Test Environment
Server Configurations
  WAS5 on TORAS3RM
  WebSphere v5.0 Server Configuration
    DefaultEAR
```

J2EE Navigator | Server Configuration | J2EE Hierarchy

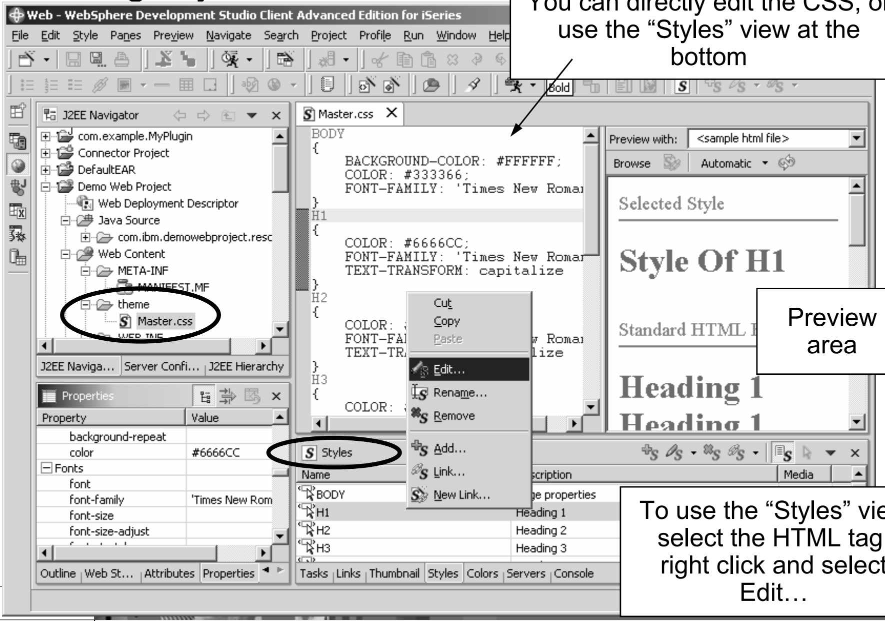
Cascading Style Sheets

- Cascading Style Sheets (CSS) provide a central place to define the appearance of all HTML and JSP pages in your Web app
  - Properties are specified for the various HTML tags like BODY, H1, H2, TABLE
  - Fonts, colors, spacing, margins, positioning, alignment, …
- Stored in a separate .css file
  - Associated with HTML or JSP file using the HTML link tag

```
<LINK href="theme/Master.css" rel="stylesheet" type="text/css">
```

- CSS
  - Specialized graphical editor for working with CSSs
  - New HTML and JSP file wizards ask if you want to associate new file with an existing CSS

Cascading Style Sheet Editor



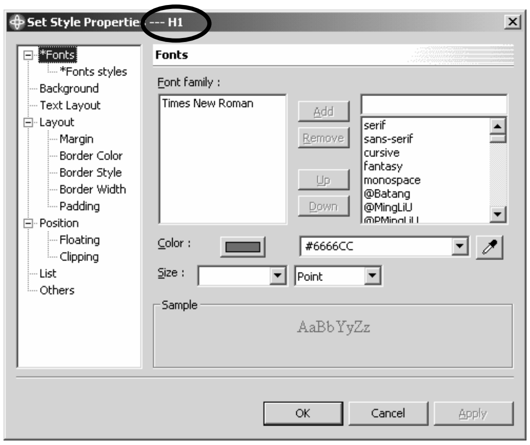
You can directly edit the CSS, or use the “Styles” view at the bottom

```
BODY
{
    BACKGROUND-COLOR: #FFFFFF;
    COLOR: #333366;
    FONT-FAMILY: 'Times New Roma
}
H1
{
    COLOR: #6666CC;
    FONT-FAMILY: 'Times New Roma
    TEXT-TRANSFORM: capitalize
}
H2
{
    COLOR:
    FONT-FA
    TEXT-TR
}
H3
{
    COLOR:
```

Preview area

To use the “Styles” view select the HTML tag, right click and select Edit…

Cascading Style Sheet Editor



Font family : Times New Roman

Color : #6666CC

Size : Point

Properties for the H1 (heading 1) HTML tag.

Changing a property here changes the look of all H1 tags in your Web application.

Dialog for graphically editing the properties for an HTML tag.

No need to know or learn the CSS syntax!

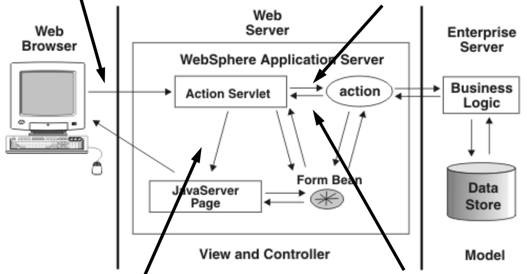
Struts

- What is Struts?
  - Open source framework for developing web applications
  - Sponsored by the Apache Software Foundation
  - Supports developing Web based applications that follow the
    - Model-View-Controller design
- How does it work?
  - Struts provides the Controller
    - You provide the Model and the View
  - Struts also provides:
    - Custom tag libraries for:
      - Internationalization
- Struts is supported by the WebSphere Studio development tools and the WebSphere Application Server runtime

Struts Overview

1. Incoming request from Browser

2. Struts ActionServlet looks up the corresponding action class for the request, populates the a form bean with incoming data and passes the request to the action class



4. ActionServlet forwards the request to the corresponding JSP which sends result to Browser as an HTML page

3. Action class processes the request (using OS/400 *PGMs and *SRVPGMs) places results in form bean and returns to ActionServlet

How Does It Work - Controller

- Struts ActionServlet is the Controller
  - Uses configuration file (struts-config.xml) to determine:
    - ActionFormBeans
      - Uses the <form-bean> tag
    - Global Forwards
      - Uses the <forward> tag
    - ActionMappings
      - Uses the <action> tag
- What do you do?
  - Create an ActionForm to send data between view and model
  - Write an Action class for each request
  - Configure ActionMapping for each request

How Does It Work – Controller

- ActionForm (form bean)
  - Stores and validates data from incoming HTML pages
  - Transfers data between the view and the model
    - Can be stored in either the session or the request
  - Upon receiving a request, the controller populates the associated ActionForm with data from the request and forwards the form bean to the Action class
  - ActionForm can optionally perform validation on input
    - Override the method:

```
validate(ActionMapping mapping, HttpServletRequest
   request)
```

    - Struts handles redisplaying input page with error messages

How Does It Work - Model

- Action class
  - Handle error checking and invokes business logic (model)
  - This is the part you have to code!
  - Implement the method:

```
public ActionForward execute(ActionMapping mapping,
                             ActionForm form,
                              HttpServletRequest request,
                             HttpServletResponse response
```

  - Return ActionForward instance to specify where control goes next
    - Typically a JSP to return results of Action to browser
    - Maps to an **Global Forward** (defined in struts-config.xml)
- ActionMapping
  - This is how the ActionServlet determines which incoming URL requests get mapped to which Action classes
  - ActionMappings are stored in the struts-config.xml file, requires the following info:
    - Incoming URI
    - Name of Action class
    - Name of the form bean used by this Action

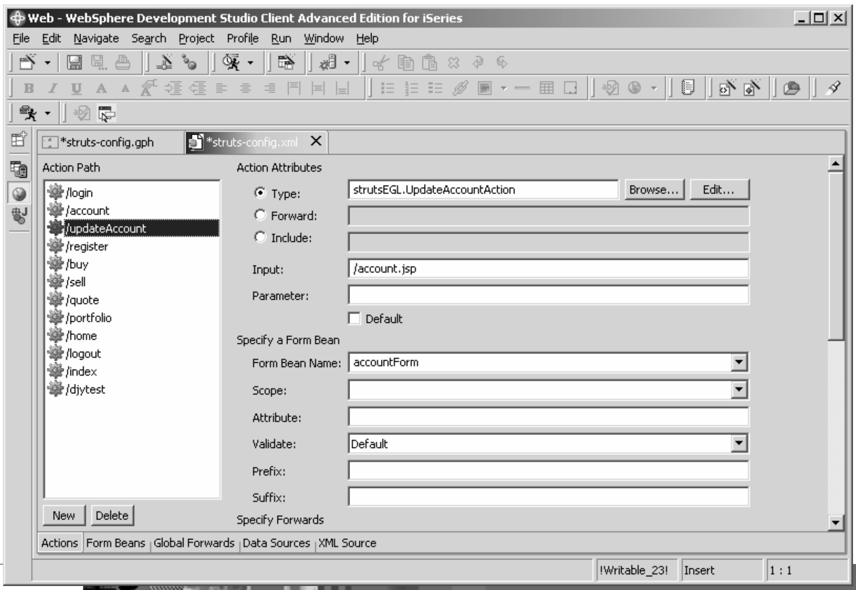
How Does It Work – View

- Struts includes tag libraries to help you
  - Create internationalized applications
    - Load in translated messages
    - Format dates and numbers for different locales
  - Automatically validate user input
    - Automatically redisplay input page with error messages from validation
  - Pre-fill HTML entry fields with data from your application

Struts Tools in Development Studio Client

- Enable Web projects for Struts, this automatically:
  - Creates struts-config.xml
  - Adds Struts tag libraries to the project
- Wizards to create
  - Form beans
  - Action classes
- Special Struts Configuration File Editor
  - You don’t have to know XML or the XML syntax used in the stuts-config.xml

Struts Configuration File Editor


Web - WebSphere Development Studio Client Advanced Edition for iSeries
File Edit Navigate Search Project Profile Run Window Help
*struts-config.gph
*struts-config.xml
New
Delete
Browse...
Edit...
Actions Form Beans Global Forwards Data Sources XML Source
!Writable_23!
Insert
1 : 1

Action Path
- /login
- /account
- /updateAccount
- /register
- /buy
- /sell
- /quote
- /portfolio
- /home
- /logout
- /index
- /djytest

Action Attributes

[✓ Type:]	strutsEGL.UpdateAccountAction
Forward:	
Include:	
Input:	/account.jsp
Parameter:	
☐ Default	

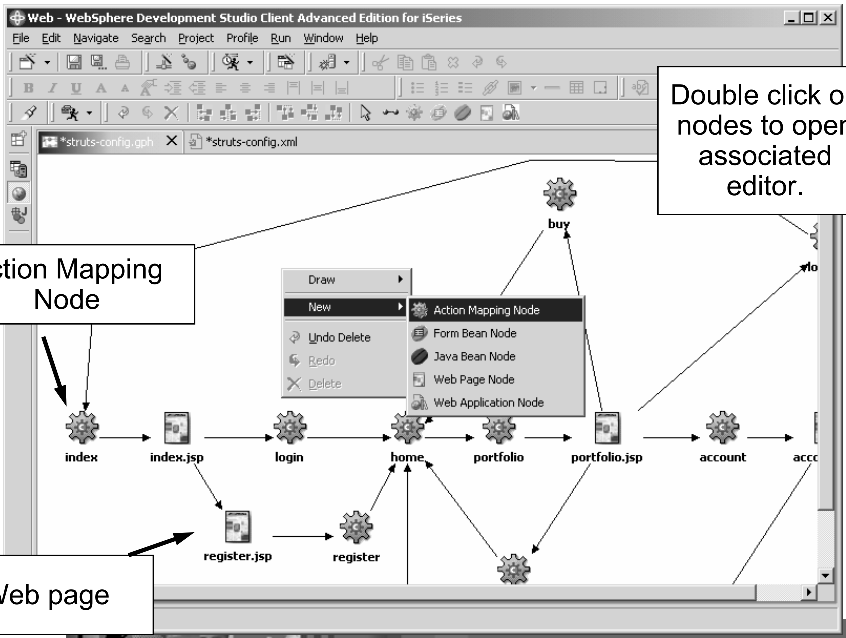
Specify a Form Bean

Form Bean Name:	accountForm
Scope:	
Attribute:	
Validate:	Default
Prefix:	
Suffix:	

Specify Forwards

Struts Tools in Development Studio Client

- Web Diagram Editor
  - Shows a graph view of your Struts based Web application
  - Can be used as a central point for working with the Web app
  - Useful for:
    - Adding new actions / form beans / JSPs
    - Editing existing actions / form beans / JSPs
    - Documenting overall architecture of the Web application
  - As parts are added, deleted or updated in the Web diagram editor the struts-config.xml file is updated with changes



Double click on nodes to open associated editor.

Action Mapping Node

Web page

Server Tool
Configuring the Test Environment

Servers and Server Configurations

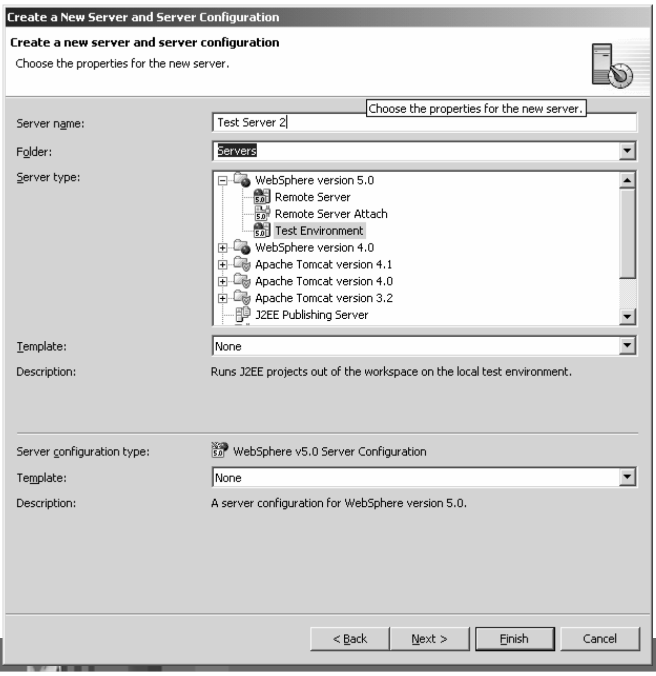
- The test environment uses servers and server configurations to run and debug your web projects
  - The first time you test a Web project a server and server configuration is created for you
  - You can create your own, customize them and associate web projects with different servers

- Server Configuration
  - Setup and configuration information for a Server

- Server
  - Instance of a server configuration where you can test your Web applications

- Types of Servers and Server Configurations
  - WebSphere Application Server V5.0 and V4.0
  - Apache Tomcat V4.1, V4.0 and V3.2

Creating a New Test Environment Configuration



Create a New Server and Server Configuration

Create a new server and server configuration

Choose the properties for the new server.

Choose the properties for the new server.

Server name: Test Server 2

Folder: Servers

Server type:

- WebSphere version 5.0
  - Remote Server
  - Remote Server Attach
  - Test Environment
- WebSphere version 4.0
- Apache Tomcat version 4.1
- Apache Tomcat version 4.0
- Apache Tomcat version 3.2
- J2EE Publishing Server

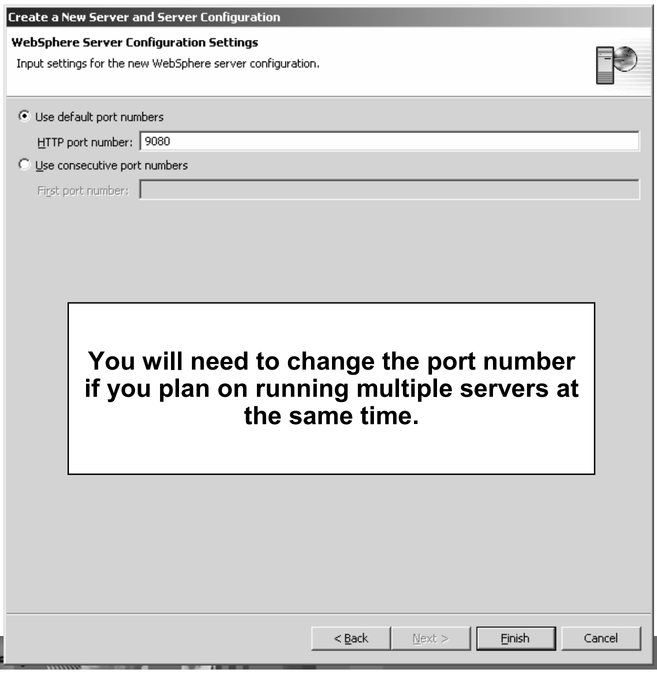
Template: None

Description: Runs J2EE projects out of the workspace on the local test environment.

Server configuration type: WebSphere v5.0 Server Configuration

Template: None

Description: A server configuration for WebSphere version 5.0.


Create a New Server and Server Configuration

WebSphere Server Configuration Settings

Input settings for the new WebSphere server configuration.

[✓ Use default port numbers]

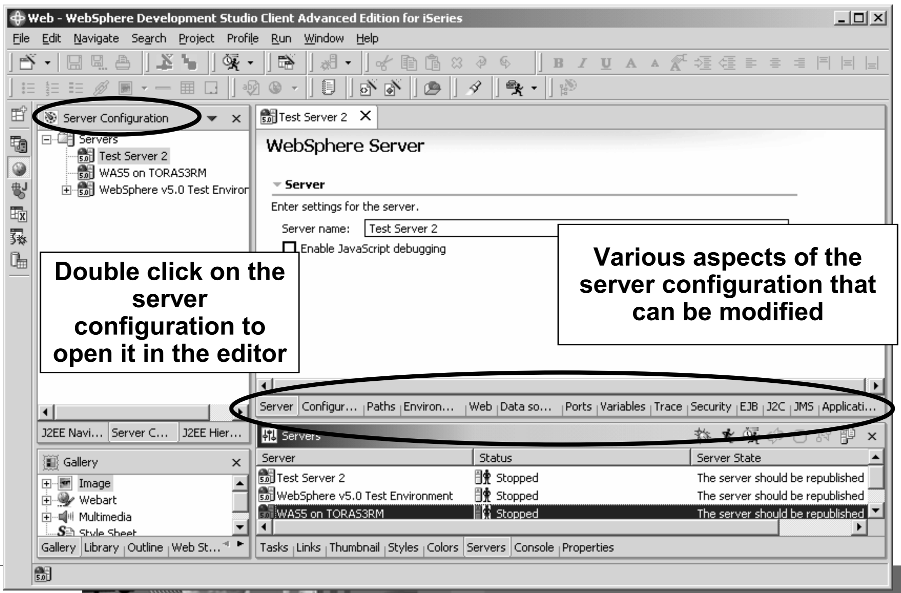
HTTP port number: 9080

Use consecutive port numbers

First port number:

You will need to change the port number if you plan on running multiple servers at the same time.

Editing a Server Configuration


Web - WebSphere Development Studio Client Advanced Edition for iSeries
File Edit Navigate Search Project Profile Run Window Help
Server Configuration
Servers
Test Server 2
WAS5 on TORAS3RM
WebSphere v5.0 Test Enviror
Test Server 2
WebSphere Server
Server
Enter settings for the server.
Server name: Test Server 2
Enable JavaScript debugging
Server Configur... Paths Environ... Web Data so... Ports Variables Trace Security EJB J2C JMS Applicati...
J2EE Navi... Server C... J2EE Hier...
Servers
Gallery
Image
Webart
Multimedia
Style Sheet
Gallery Library Outline Web St...
Tasks Links Thumbnail Styles Colors Servers Console Properties

Server	Status	Server State
Test Server 2	Stopped	The server should be republished
WebSphere v5.0 Test Environment	Stopped	The server should be republished
WAS5 on TORAS3RM	Stopped	The server should be republished

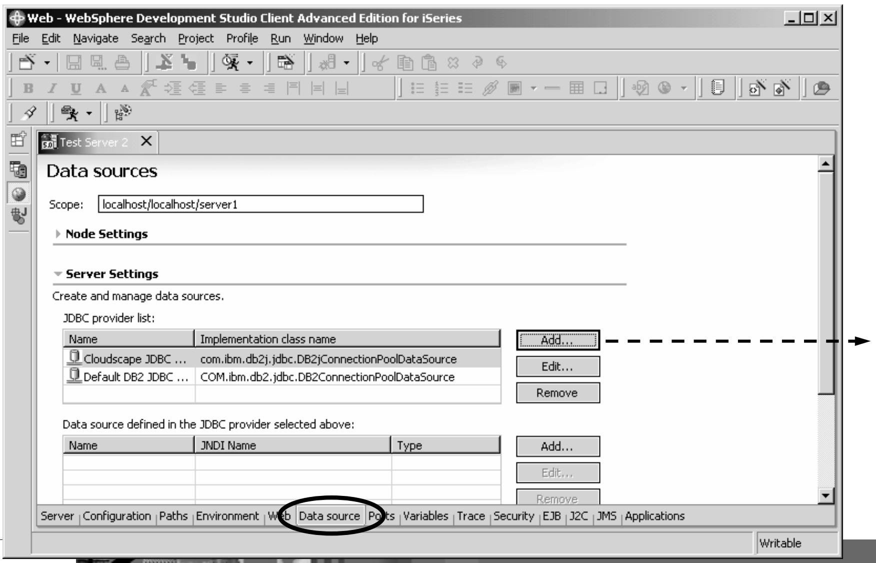
Double click on the server configuration to open it in the editor

Various aspects of the server configuration that can be modified

JDBC: Defining Connection Pools in the Test Environment

- Earlier we looked at JDBC, what it is and why you would use it
- Now we will look at how to configure JDBC connection pools in the WebSphere test environment
  - First you need to add the required JDBC driver to the **providers list**
  - Then you can define a connection pool for the JDBC driver

JDBC: Adding a Driver to the Provider List


Web - WebSphere Development Studio Client Advanced Edition for iSeries
File Edit Navigate Search Project Profile Run Window Help
Test Server 2

Data sources

Scope: localhost/localhost/server1

Node Settings

Server Settings

Create and manage data sources.

JDBC provider list:

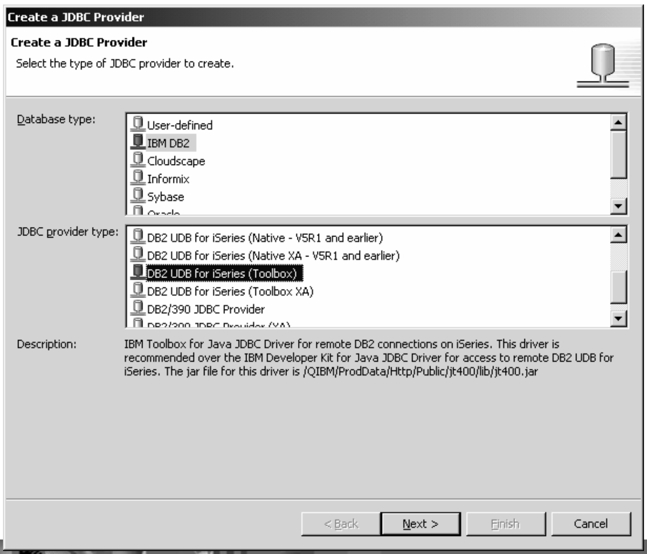
Name	Implementation class name
Cloudscape JDBC ...	com.ibm.db2j.jdbc.DB2jConnectionPoolDataSource
Default DB2 JDBC ...	COM.ibm.db2.jdbc.DB2ConnectionPoolDataSource

Data source defined in the JDBC provider selected above:

Name	JNDI Name	Type

Add...
Edit...
Remove
Server | Configuration | Paths | Environment | Web | Data source | Ports | Variables | Trace | Security | EJB | J2C | JMS | Applications
Writable

JDBC: Adding a Driver to the Provider List - 2


Create a JDBC Provider

Create a JDBC Provider

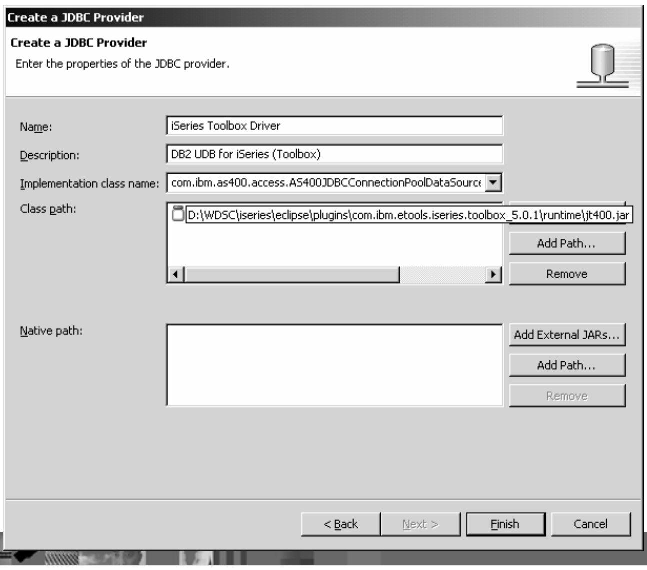
Select the type of JDBC provider to create.

Database type:
- User-defined
- IBM DB2
- Cloudscape
- Informix
- Sybase
- Oracle

JDBC provider type:
- DB2 UDB for iSeries (Native - V5R1 and earlier)
- DB2 UDB for iSeries (Native XA - V5R1 and earlier)
- DB2 UDB for iSeries (Toolbox)
- DB2 UDB for iSeries (Toolbox XA)
- DB2/390 JDBC Provider
- DB2/390 JDBC Provider (XA)

Description: IBM Toolbox for Java JDBC Driver for remote DB2 connections on iSeries. This driver is recommended over the IBM Developer Kit for Java JDBC Driver for access to remote DB2 UDB for iSeries. The jar file for this driver is /QIBM/ProdData/Http/Public/jt400/lib/jt400.jar

JDBC: Adding a Driver to the Provider List - 3



Create a JDBC Provider

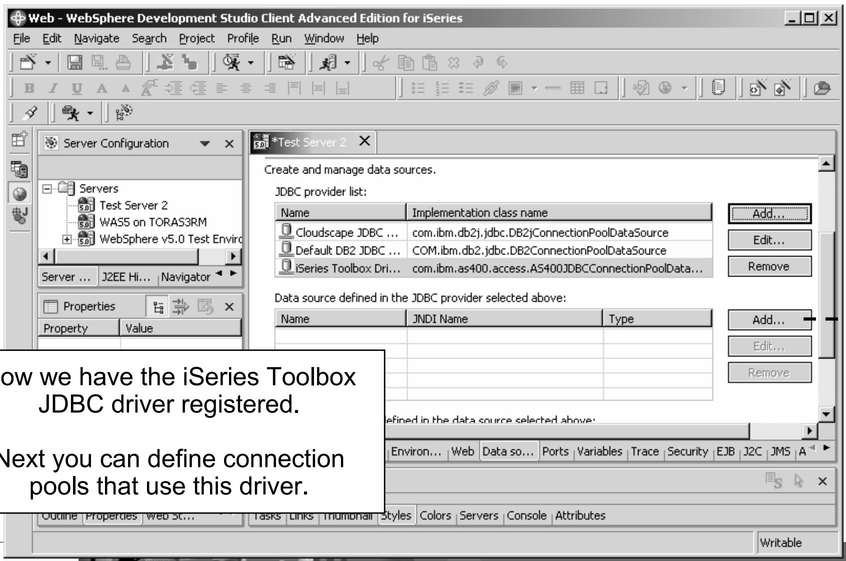
Create a JDBC Provider

Enter the properties of the JDBC provider.

Name:	iSeries Toolbox Driver
Description:	DB2 UDB for iSeries (Toolbox)
Implementation class name:	com.ibm.as400.access.AS400JDBCConnectionPoolDataSource
Class path:	D:\WDSC\iseries\eclipse\plugins\com.ibm.etools.iseries.toolbox_5.0.1\runtime\jt400.jar
Native path:	

JDBC: Configuring a Connection Pool - 4

JDBC: Configuring a Connection Pool


Web - WebSphere Development Studio Client Advanced Edition for iSeries
File Edit Navigate Search Project Profile Run Window Help
Server Configuration
Servers
Test Server 2
WAS5 on TORAS3RM
WebSphere v5.0 Test Envirc
Server ... J2EE Hi... Navigator
Properties
Property Value
*Test Server 2
Add...
Edit...
Remove
Add...
Edit...
Remove
Environ... Web Data so... Ports Variables Trace Security EJB J2C JMS A
Outline Properties Web St...
Tasks Links Thumbnail Styles Colors Servers Console Attributes
Writable

Create and manage data sources.

JDBC provider list:

Name	Implementation class name
Cloudscape JDBC ...	com.ibm.db2j.jdbc.DB2jConnectionPoolDataSource
Default DB2 JDBC ...	COM.ibm.db2.jdbc.DB2ConnectionPoolDataSource
iSeries Toolbox Dri...	com.ibm.as400.access.AS400JDBCConnectionPoolData...

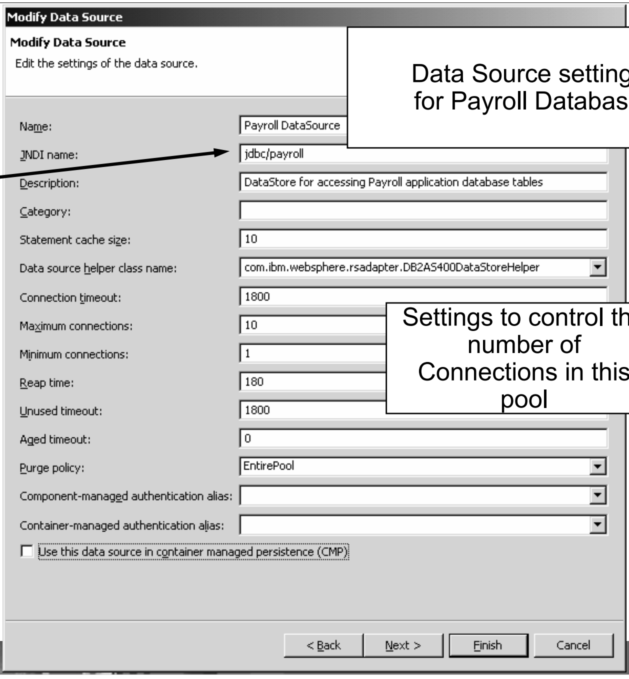
Data source defined in the JDBC provider selected above:

Name	JNDI Name	Type

efined in the data source selected above:

Now we have the iSeries Toolbox JDBC driver registered.

Next you can define connection pools that use this driver.



Modify Data Source

Modify Data Source

Edit the settings of the data source.

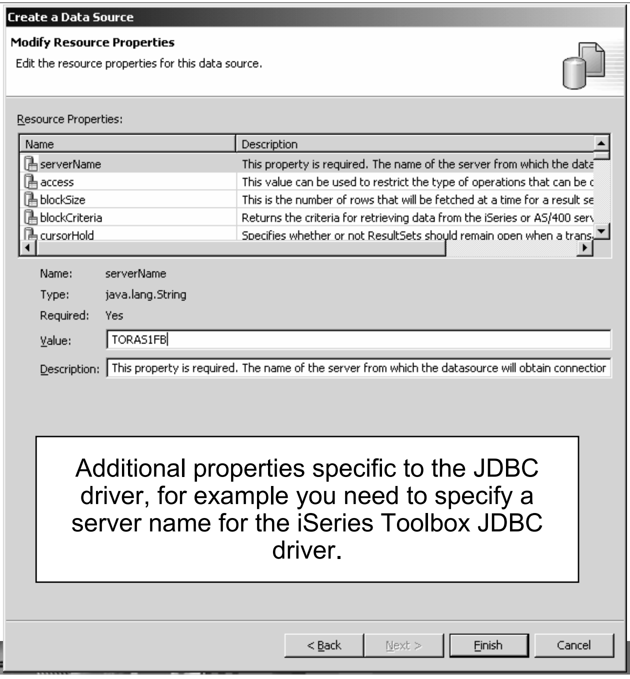
Name:	Payroll DataSource
JNDI name:	jdbc/payroll
Description:	DataStore for accessing Payroll application database tables
Category:	
Statement cache size:	10
Data source helper class name:	com.ibm.websphere.rsadapter.DB2AS400DataStoreHelper
Connection timeout:	1800
Maximum connections:	10
Minimum connections:	1
Reap time:	180
Unused timeout:	1800
Aged timeout:	0
Purge policy:	EntirePool
Component-managed authentication alias:	
Container-managed authentication alias:	

☐ Use this data source in container managed persistence (CMP)

Use this JNDI name in your Code to lookup JDBC connections From this Data Source

Data Source settings for Payroll Database

Settings to control the number of Connections in this pool



Create a Data Source

Modify Resource Properties

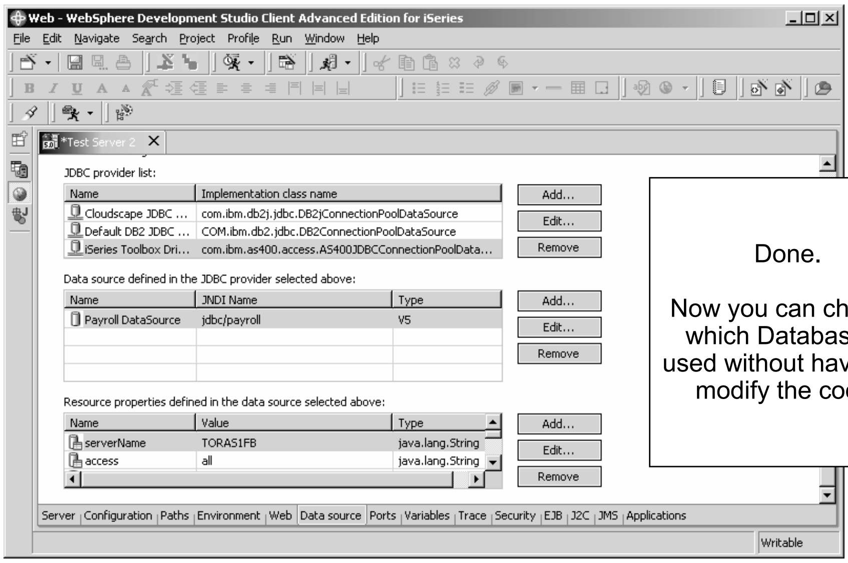
Edit the resource properties for this data source.

Resource Properties:

Name	Description
serverName	This property is required. The name of the server from which the data
access	This value can be used to restrict the type of operations that can be c
blockSize	This is the number of rows that will be fetched at a time for a result se
blockCriteria	Returns the criteria for retrieving data from the iSeries or AS/400 serv
cursorHold	Specifies whether or not ResultSets should remain open when a trans

Name:	serverName
Type:	java.lang.String
Required:	Yes
Value:	TORAS1FB
Description:	This property is required. The name of the server from which the datasource will obtain connectior

Additional properties specific to the JDBC driver, for example you need to specify a server name for the iSeries Toolbox JDBC driver.


Web - WebSphere Development Studio Client Advanced Edition for iSeries
File Edit Navigate Search Project Profile Run Window Help
*Test Server 2
Add...
Edit...
Remove
Server Configuration Paths Environment Web Data source Ports Variables Trace Security EJB J2C JMS Applications
Writable

JDBC provider list:

Name	Implementation class name
Cloudscape JDBC ...	com.ibm.db2j.jdbc.DB2jConnectionPoolDataSource
Default DB2 JDBC ...	COM.ibm.db2.jdbc.DB2ConnectionPoolDataSource
iSeries Toolbox Dri...	com.ibm.as400.access.AS400JDBCConnectionPoolData...

Data source defined in the JDBC provider selected above:

Name	JNDI Name	Type
Payroll DataSource	jdbc/payroll	V5

Resource properties defined in the data source selected above:

Name	Value	Type
serverName	TORAS1FB	java.lang.String
access	all	java.lang.String

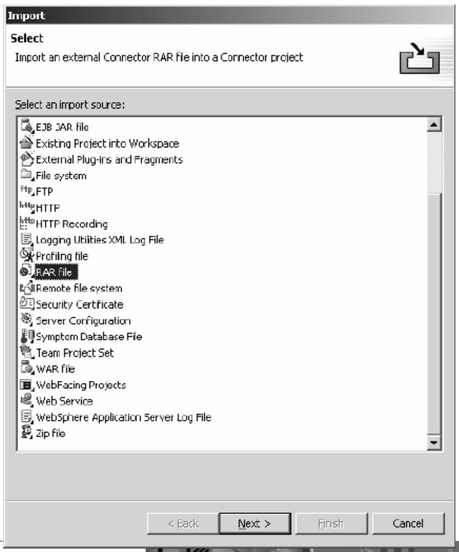
Done.

Now you can change which Database is used without having to modify the code.

J2EE Connector Architecture

- Earlier we looked at the J2EE Connector Architecture, what it is and why you would use it
- Now we will look at how to configure JCA in the WebSphere test environment
  - First you need to import the required **Resource Adapter** into the Workbench
    - Resource adapters get imported into a special type of project called a Connector Project
  - Then you can define a Connection in the Server Configuration

JCA: Importing Resource Adapters into WDSc



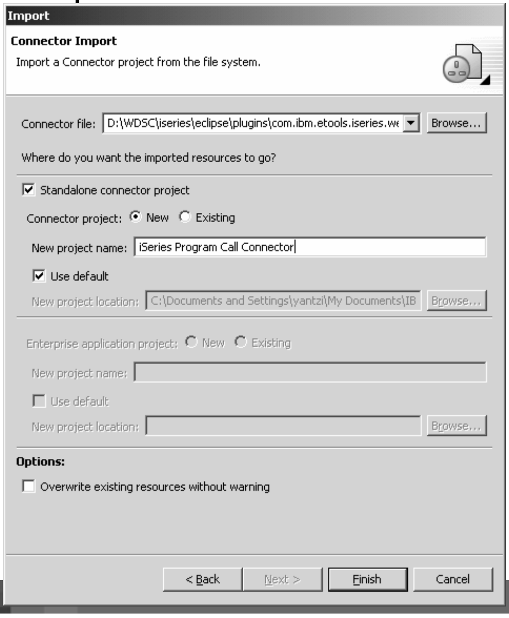
Import

Select

Import an external Connector RAR file into a Connector project

Select an import source:

- EJB JAR file
- Existing Project into Workspace
- External Plug-ins and Fragments
- File system
- FTP
- HTTP
- HTTP Recording
- Logging Utilities XML Log File
- Profiling file
- RAR file
- Remote file system
- Security Certificate
- Server Configuration
- Symptom Database File
- Team Project Set
- WAR file
- WebFacing Projects
- Web Service
- WebSphere Application Server Log File
- Zip file



Import

Connector Import

Import a Connector project from the file system.

Connector file: D:\WDSC\iseries\eclipse\plugins\com.ibm.etools.iseries.we

Where do you want the imported resources to go?

[✓] Standalone connector project

Connector project: [✓ New] Existing

New project name: iSeries Program Call Connector

[✓] Use default

New project location: C:\Documents and Settings\yantzi\My Documents\IB

Enterprise application project: New Existing

New project name:

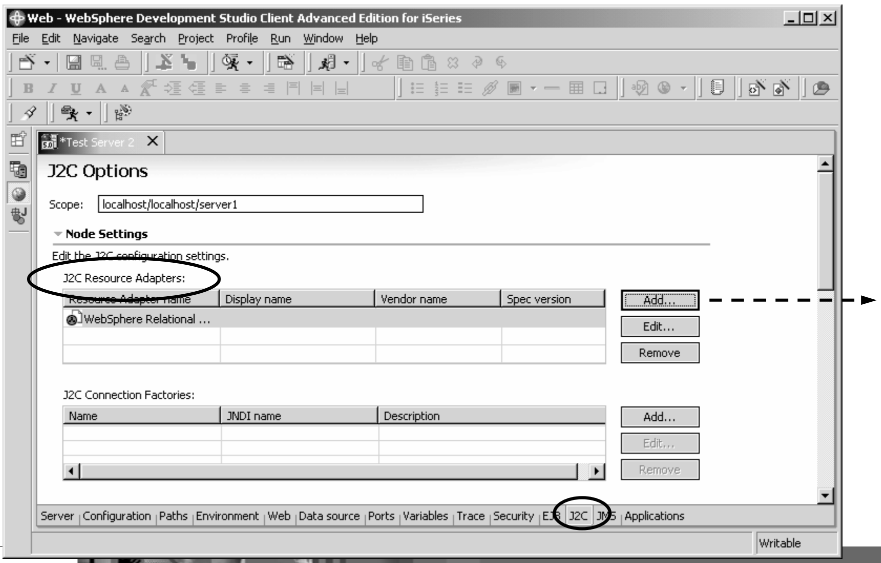
[ ] Use default

New project location:

Options:

[ ] Overwrite existing resources without warning

Adding Resource Adapters to the Test Environment


Web - WebSphere Development Studio Client Advanced Edition for iSeries
File Edit Navigate Search Project Profile Run Window Help
*Test Server 2
Add...
Edit...
Remove
Add...
Edit...
Remove
Server Configuration Paths Environment Web Data source Ports Variables Trace Security EJB J2C JMS Applications
Writable

J2C Options

Scope: localhost/localhost/server1

Node Settings

Edit the J2C configuration settings.

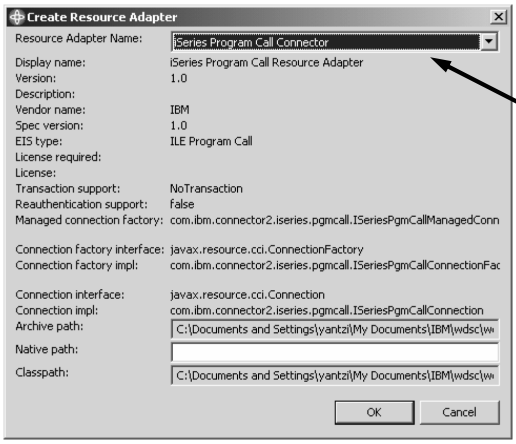
J2C Resource Adapters:

Resource Adapter name	Display name	Vendor name	Spec version
WebSphere Relational ...			

J2C Connection Factories:

Name	JNDI name	Description

JCA: Configuring Resource Adapters in Servers

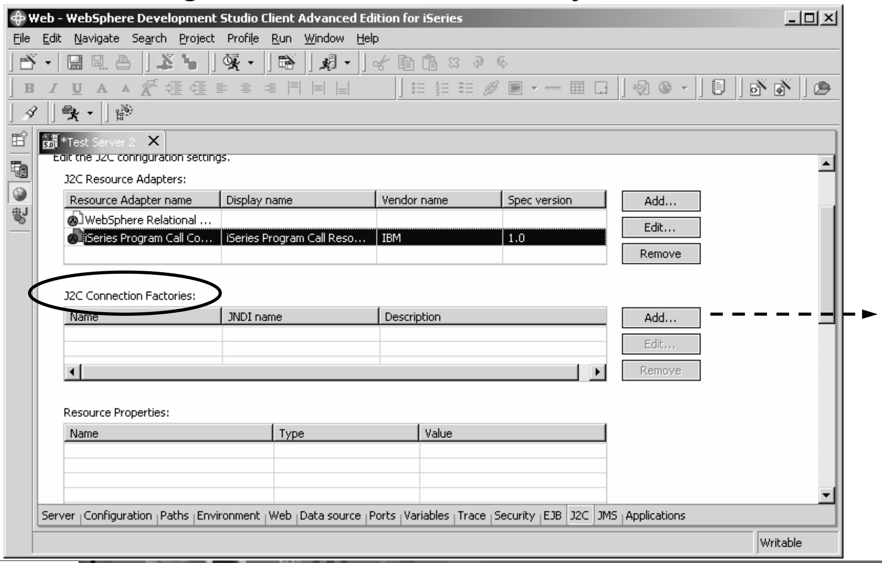


Create Resource Adapter

Resource Adapter Name:	iSeries Program Call Connector
Display name:	iSeries Program Call Resource Adapter
Version:	1.0
Description:	
Vendor name:	IBM
Spec version:	1.0
EIS type:	ILE Program Call
License required:	
License:	
Transaction support:	NoTransaction
Reauthentication support:	false
Managed connection factory:	com.ibm.connector2.iseries.pgmcall.ISeriesPgmCallManagedConn
Connection factory interface:	javax.resource.cci.ConnectionFactory
Connection factory impl:	com.ibm.connector2.iseries.pgmcall.ISeriesPgmCallConnectionFac
Connection interface:	javax.resource.cci.Connection
Connection impl:	com.ibm.connector2.iseries.pgmcall.ISeriesPgmCallConnection
Archive path:	C:\Documents and Settings\yantzi\My Documents\IBM\wdsc\w
Native path:	
Classpath:	C:\Documents and Settings\yantzi\My Documents\IBM\wdsc\w

Project where you imported the resource adapter

JCA: Creating A Connection Factory



Web - WebSphere Development Studio Client Advanced Edition for iSeries

File Edit Navigate Search Project Profile Run Window Help

*Test Server 2

Edit the J2C configuration settings.

J2C Resource Adapters:

Resource Adapter name	Display name	Vendor name	Spec version
WebSphere Relational ...			
iSeries Program Call Co...	iSeries Program Call Reso...	IBM	1.0

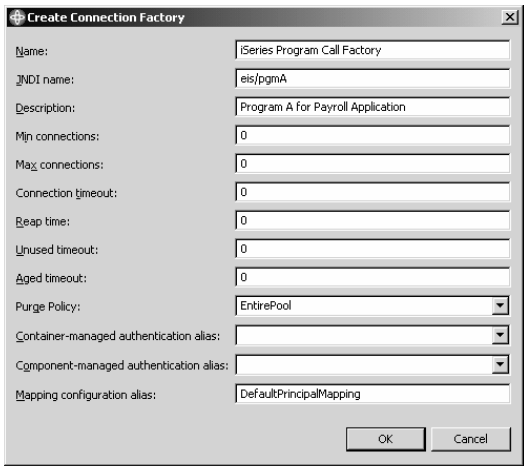
J2C Connection Factories:

Name	JNDI name	Description

Resource Properties:

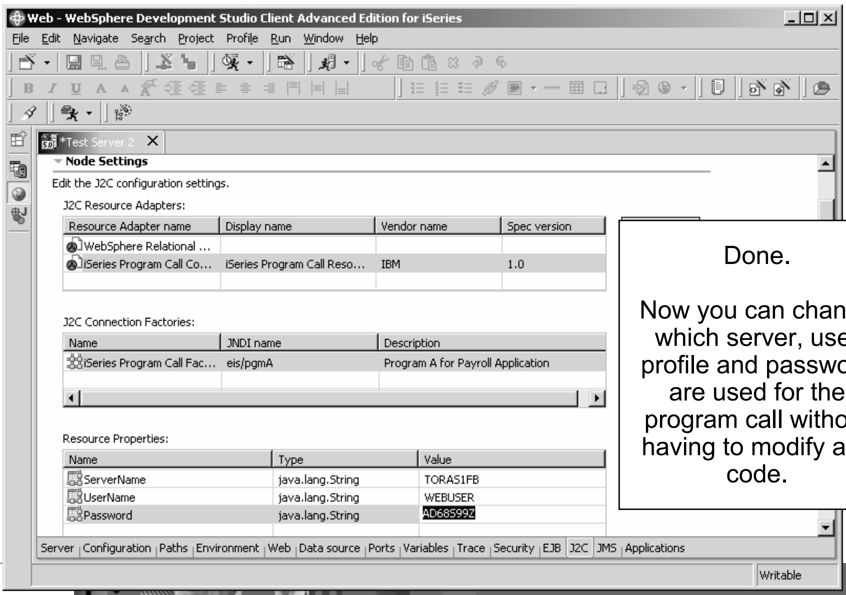
Name	Type	Value

JCA: Creating A Connection Factory



Name:	iSeries Program Call Factory
JNDI name:	eis/pgmA
Description:	Program A for Payroll Application
Min connections:	0
Max connections:	0
Connection timeout:	0
Reap time:	0
Unused timeout:	0
Aged timeout:	0
Purge Policy:	EntirePool
Container-managed authentication alias:	
Component-managed authentication alias:	
Mapping configuration alias:	DefaultPrincipalMapping

JCA: Creating A Connection Factory


Web - WebSphere Development Studio Client Advanced Edition for iSeries
File Edit Navigate Search Project Profile Run Window Help
*Test Server 2
Server Configuration Paths Environment Web Data source Ports Variables Trace Security EJB J2C JMS Applications
Writable

Node Settings

Edit the J2C configuration settings.

J2C Resource Adapters:

Resource Adapter name	Display name	Vendor name	Spec version
WebSphere Relational ...			
iSeries Program Call Co...	iSeries Program Call Reso...	IBM	1.0

J2C Connection Factories:

Name	JNDI name	Description
iSeries Program Call Fac...	eis/pgmA	Program A for Payroll Application

Resource Properties:

Name	Type	Value
ServerName	java.lang.String	TORAS1FB
UserName	java.lang.String	WEBUSER
Password	java.lang.String	AD68599Z

Done.

Now you can change which server, user profile and password are used for the program call without having to modify any code.

Summary

WebSphere. software

e business software

Summary

- Java 2 Enterprise Edition
  - Standards based model for developing applications in Java
    - Web applications
    - Enterprise applications
    - Client / Server applications
  - Supported by major Web Application Server vendors

- WebSphere Development Studio Client
  - Great Web tools to make developing J2EE applications productive and easy
  - Views of your J2EE resources
  - Customized editors for all the various technologies
    - Cascading style sheets, JSPs, HTML, animations, server configurations, Web deployment descriptor, …

Additional Resources

- J2EE Technologies
  http://java.sun.com/j2ee
- Cascading Style Sheets
  http://www.w3c.org/Style/CSS/
- Struts
  http://jakarta.apache.org/struts/
- WebSphere Workbench Tools
  http://www.software.ibm.com/wsdd/zones/studio/
  http://www.ibm.com/developer

Trademarks & Disclaimers

Disclaimer

- **Acknowledgment:**

  This presentation is a collaborative effort of the IBM Toronto AS/400 Application Development presentation team, including work done by:

  Don Yantzi, Phil Coulthard, George Farr, Claus Weiss, David Slater, Alison Butteril, Linda Cole

- **Disclaimer:**

  The information contained in this document has not been submitted to any formal IBM test and is distributed on an as is basis without any warranty either express or implied. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customers' ability to evaluate and integrate them into the customers' operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will result elsewhere. Customers attempting to adapt these techniques to their own environment do so at their own risk.

- **Reproduction:**

  The base presentation is the property of IBM Corporation. Permission must be obtained PRIOR to making copies of this material for any reason.

Web Tools: Beyond the Basics in IBM WebSphere Development Studio Client for iSeries

iSeries Application Development Team: IBM Toronto