Administration -- table of contents

6: Administer applications

Typically, one or more application developers with different areas ofexpertise (such as architects, Java programmers, legacy programmers, and Web programmers) design and create a new application or migrate an existing application to support new Java specifications based on input from business users.

After the files comprising an application have been developed, then the application can be added to the application server for access by users. This section of documentation helps you learn the flow of administrative activities, whether you are a code developer introducing an application into a test environment, or an administrator responsible for theproduction environment.

A feasible end-to-end administrative procedure

The following procedure provides links to more information abouteach step. To practice the procedure using an application providedwith WebSphere Application Server, perform the application configuration and deployment tutorials.

1. **Install the product.** Plan and install a topology comprised of one or more product installations and the necessary prerequisites.
    - Who: Lead architect, network administrator, systems administrator
    - How: See components section, installation sectionand topologies section .
2. **Assemble the application.** Assemble the application modules from the application files. Set values for the deployment descriptor. For applications containing EJB modules, generate code for deployment.
    - Who: Web designer, Java programmer
    - How: Using the Application Assembly Tool, and possibly the Deployment Tool for Enterprise JavaBeans

    For example, for an EJB module, the enterprise bean developer writes and compiles the enterprise bean components. The developer assembles the components and a deployment descriptor into an EJB JAR file and then generates code for deployment.

    For entity beans (BMP or CMP), the developer also generates the database tables the beans will use to store their data.

    The developer transfers the JAR file to the WebSphere administrator, or informs the administrator of its location on a machine in the WebSphere administrative domain.
3. **If migrating from Version 3.5, prepare the application for workload management**.
    - Who: Systems administrator
    - How: As discussed in article 7.2.1 about WLM-enabled EJB JAR files
4. **Secure the application.** Configure security in the application.
    - Who: Systems administrator
    - How: Using the Application Assembly Tool
5. **Configure the runtime and resources.** Create application servers and other resources that will support applications.
    - Who: Systems administrator
    - How: Using the administrative console
6. **Install the application.** Install the application into the test environment.

   - Who: Java programmer, systems administrator
   - How: Using the administrative console

   Installing an application refers to the process of placing the Enterprise Archive (EAR) file in a runtime environment comprised of an application server.

   During this step, the administrator can optionally edit the application's deployment descriptor.

7. **Configure resources for application clients (if applicable).** Create resources for the Java clients that will access your application.
   - Who: Systems administrator
   - How: Using the Application Client Resource Configuration Tool
8. **Test the application.** Test the application prior to enabling runtime securityso that you can tell the difference between security-related problems and other problems.
   - Who: Systems administrator
   - How: Using a Web browser or other application client
9. **Configure runtime security.** Configure and enable security in the application server runtime.
   - Who: Systems administrator
   - How: Using the administrative console

     Note that because security is presented as a scenario, the instructions (such as suggested field values) refer to a particular sample application. Substitute the names of your own application components.
10. **Test application security.** Test the application again, this time with securityenabled.
    - Who: Tester, systems administrator
    - How: Using the administrative console
11. **Perfect the test environment.** Debug and verify the application in the test environment. Perform preliminary tuning.
    - Who: Tester, systems administrator
    - How: Using the administrative console
12. **Port the application to production environment.** Configure the production environment and move the application there.
    - Who: Network administrator, systems administrator
    - How: Using the administrative console
13. **Manage the production environment.** Manage the production application, tuning the application files and configuration as needed.Update the production application as needed.
    - Who: Maintenance -- Network administrator, tester, Updates -- Java programmer, systems administrator
    - How: Using the administrative console
14. **Uninstall applications as needed.**
    - Who: Network administrator, systems administrator
    - How: Using the administrative console
15. **Remove security, as needed.**
    - Who: Network administrator, systems administrator
    - How: Using the administrative console

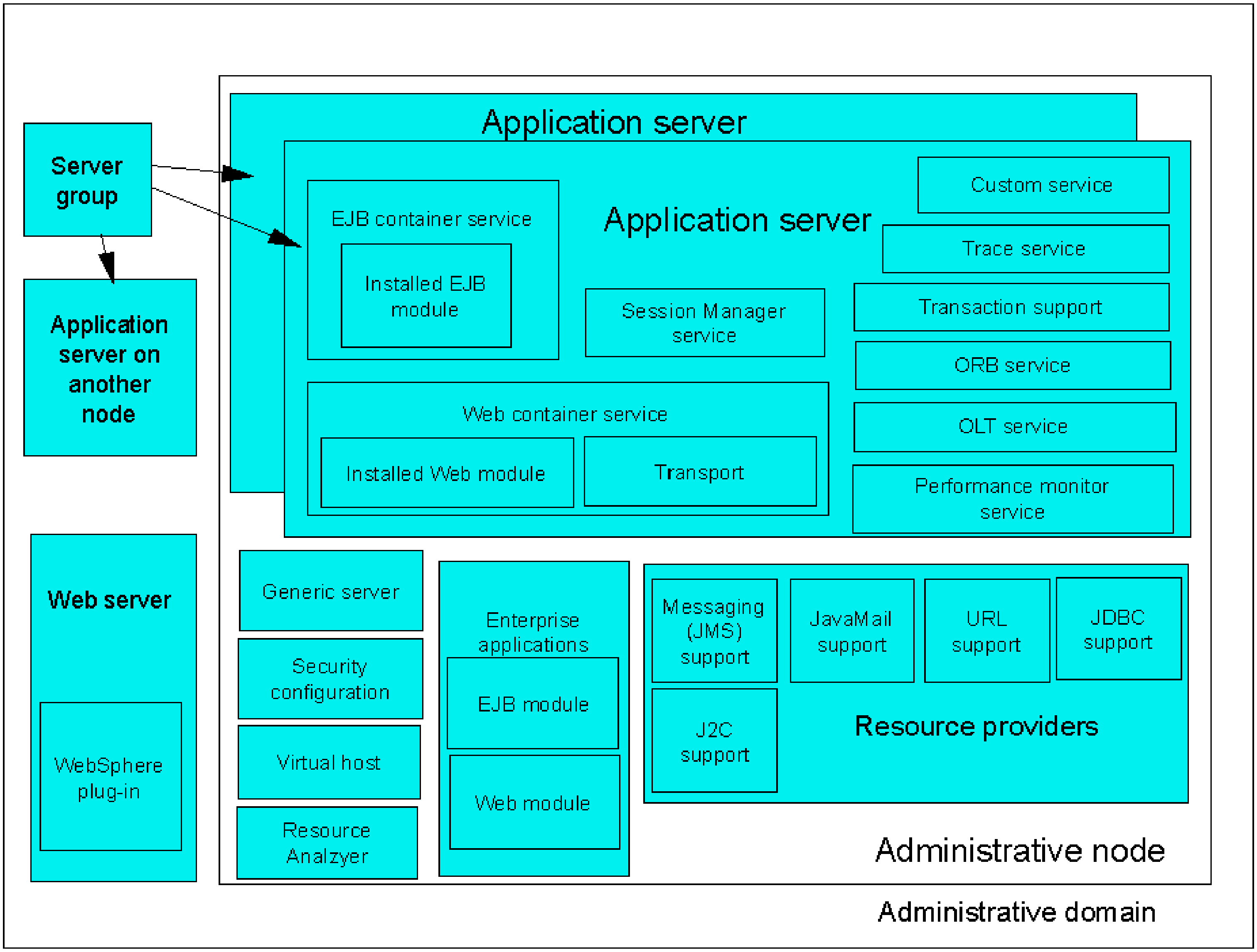
More information

To obtain more information about a step in the above procedure, see the links to sub-topics.

6.1: Quick reference for administration

The administrative model for Advanced Edition (non-Single Server) is summarized graphically below. Click the name of a resource to view the entry point to the documentation for the administering the resource.

See below for further discussion of the administrative model depicted here.



As shown above, and in the tree views of the administrative console and other graphical administrative tools, the WebSphere administrative domain is comprised of several resource types, arranged in a hierarchyof containment relationships.

Discussion

The administrative nodes contain one or more application servers, plus severalresources that transcend application servers (such as those related to JDBC support) or areseparate from application servers (such as generic servers). The WebSphere plug-ins for the supported Web servers also require administration. Finally, logical server groups can be established for distributing workload among multiple, cloned application servers.

One way to classify resources in the administrative model is by whether they canhave a meaningful status, such as running or stopped. Few resource types have a status;most are static entities.

An enterprise application consists EJB modules, Web modules, and application clients. It can reside on any applicationserver. An application installed into multiple application servers can be running in some servers,but stopped in others. Because of the possible mix of states of its instances, it is not meaningful to say whether an application (as the collection of all of its instances) is running or stopped. Therefore, an enterprise application is a *static entity*.

In contrast, the EJB, Web, and application client modules comprising each enterprise application are truly *active objects*, each of which has a valid status.

A server group is a static entity that consists of one or more clones. Usually, some clones are running while others are not.Therefore, it is not meaningful to assign a status to the overall server group.

Want to learn more about a specific resource in the administrative model? ...

Documentation for each resource type

The documentation entry point for each resource type includes:

- An overview of administering the resource type
- Links to instructions for using the relevant administrative tools to administer the resource
- Links to field descriptions for configuring the properties of the resource

- Links to conceptual information answering "What is *this resource type*?"

Again, access the entry points by clicking a text label in the administrative model shown above. Youcan also access the list of available entry points under section 6.6 in the table of contents on the left side of the InfoCenter.

What are "resources"?

In this documentation, the term *resource* is used loosely to describe a logical set of properties that can beadministered, such as settings for session support using the Session Manager service. Resources rangefrom being complex objects that can be started and stopped, such as application servers, to simple groups ofrelated settings, such as the properties for configuring transaction support. The latter "resource" can beadministered using a certain tabbed page in the application server properties dialog. However, to aid retrieval of informationabout transaction support specifically, it has a separate focus in the documentation.

The set of resources outlined in the documentation varies somewhat in granularity from the administrative model portrayed in the administrative tools, geared towards helping you map activities (to administer transaction support, to administer HTTP session support, to administer Java messagingsupport, and so on) onto the specific resources and hierarchy of the WebSphere administrative model.

- For example, as shown above, an application server contains several services. The application server is represented in the tree view of the administrative console, but the individual services are not. Each service has a set of properties that you can configure, with its own entry point in the documentation for quicker, more targeted entry to the information about the service.
- In other cases, resources in theadministrative model are grouped together in the documentation because of their close relationship. Forexample, JMS providers, JMS connection factories, and JMS destinations are grouped under theheading "JMS support" because youmight know that you want to configure Java messaging (JMS), without knowing the specific namesof these JMS resources yet.

6.2: Preparing to host applications

The default application server and a set of default resources are available to helpyou begin quickly (see the related topics links). Suppose you choose instead to configure a new server and set of resources. Here iswhat you need to do in order to set up a runtime environment to support applications.

1. Create an application server.
2. Create other supporting resources.
    - Create a virtual host.
    - Configure a Web container.
    - Configure an EJB container.
3. Create resources for data access.
    - Create a JDBC provider and data source.
4. Create additional resources for J2EE services.
    - Create a URL and URL provider.
    - Create a JMS destination, connection, and provider.
    - Create a JavaMail session.
5. Create resources for personalization support.
    - Configure a Session Manager.

Configuring the application server runtime and resources comprises one step of a recommended end-to-end administrative procedure.

Configuring application client resources

Now is also a good time to prepare the resources needed by any Java clients that will use your application. See the Related information, or returnto the main administrative procedure, which contains a step reminding youto perform this task (if it applies).

6.2.1: The default resources and configurations

IBM WebSphere Application Server provides default configurations, allowingadministrators to begin quickly and easily. The product installs a default application server, a default enterprise application, the set of required resources (such as a JDBC provider for data access), and a default virtual host.

Verifying that the default administrative resources exist

To verify whether the default resources are installed:

1. Start the administrative console.
2. Locate resources in the tree view, such as:
    - Default application server (Default Server)
    - Default enterprise application (sampleApp)

The "default" and "examples" Web applications

The default enterprise application (sampleApp.ear) is installed to provide you with a method to verify the installation of WebSphere Application Server and its functionality.

The sampleApp consists of several Web modules (such as "default_app" and "examples") and EJB JAR files (such as "Increment" and "beenthere").

The Web module "default_app" has a context root of "/". Servlets provided in the default_app are:

- snoop (accessed by entering http:///servlet/snoop in a Web browser)
- hello (accessed by entering http:///servlet/hello in a Web browser)

These servlets can be used to quickly verify that the default application server is up and running.

The Web module named "examples" has a context root of "/webapp/examples". The examples Web module provides a series of tools and examples for use with the application server environment, including:

- The HitCount example, which allows you to verify both session support EJB support
- The BeenThere servlet, which is provided for demonstrating EJB workload management across a cluster of servers.

You can access the servlets in the examples Web application by entering http:///webapp/examples ina browser, to present the welcome page (index.html) for the Web module.

The sampleApp.ear archive is provided in the /installableApps directory in the event that you want to install the application into a different application server. Before installing the sampleApp on another server, you will need to create a data source for use by the HitCount EJB JAR (Inc), along with the requisite INC table.

6.3: Assembling applications and generating deployment code

The Application Assembly Tool is a graphical user interface for assemblingenterprise (J2EE) applications. Use this set of articles to becomefamiliar with the steps needed to assemble an application. Then use therelated tasks to create the application's modules, generate code fordeployment, and verify the application's archive files.

Application assembly is the process of creating an archive file that groupsall files related to an application (the archive file can include otherarchive files as well as individual class files, HTML files, GIF files, and soon). A single archive file can represent an application.Multiple archive files can be further bundled into a higher-level archive ifthe application has many components. Each bundled entity is called a*module*. The assembly process consists of selecting all ofthe files to be included in the module, creating a deployment descriptor (XMLfile) for the module, and then packaging the files into a single archivefile. The archive file lists all files in the module. Thedeployment descriptor lists the characteristics of the module and containsinstructions for how the module is to be deployed.

Assembling an application consists of the following steps:

1. Launch the Application Assembly Tool.
2. Create modules.
   - Specify the location of files that make up the application.
   - Enter or edit properties used to generate the deployment descriptorfile.
   - Optionally, add JNDI binding information for and resolve references to theapplication's external resources.
3. Save the archive file. This automatically generates the XMLdeployment descriptor files for the application.
4. Verify the archive file and, optionally, view the content of the XMLfiles.
5. If the application contains EJB modules, or is a stand-alone EJB module,generate deployment code for the application.

The Application Assembly Tool invokes the Deployment Tool for EnterpriseJavaBeans (ejbdeploy) for generating code and validating archives. ForCMP entity beans, the tool can also generate a map and schema document, andDDL file containing SQL code for creating a database table.

After an application is assembled, it can be installed in an applicationserver by using the administrative console. At installation time, theconsole is used to carry out the security instructions defined in thedeployment descriptor and to locate the required external resources that itwill use, such as enterprise beans and databases. In the console,additional configuration properties can be added and binding propertiesdefined in the Application Assembly Tool can be redefined.

After the application is deployed, you can use the Application AssemblyTool to modify the application by adding or removing modules, editingdeployment descriptor properties, and regenerating code for deployment.

Note: If you are using the Application Assembly Tool to createapplication client modules, you must also use the Application Client ResourceConfiguration Tool. This tool allows you to define references toresources (other than enterprise beans) on the machine where the applicationclient resides.

6.3.1: Assembling modules

A module is a group of related files that represent an application.The files can include class files, HTML files, and GIF files. Assemblyis the process of specifying which files make up the module, creatingdeployment descriptor files for the module, and packaging these files in anarchive. Modules are used to represent any of the following:

- One or more Web components. Web components are servlets orJavaServer Pages (JSP) files.
- One or more enterprise beans.
- Application clients.

A J2EE application consists of any combination of the above.

The standard JAR file format is used to package modules, regardless of thetype of module. However, the archive file is referred to according toits content:

- EJB modules--packaged in JAR files
- Web modules--packaged in Web archive (WAR) files
- Application-client modules--packaged in JAR files
- Enterprise applications--packaged in Enterprise Archive (EAR) files

An EJB module, Web module, or application-client module can be installed asa stand-alone application or can be combined with other modules into anenterprise application. When a stand-alone application isinstalled in WebSphere Application Server, it is automatically packaged in anEAR file.

6.3.2: Setting properties for a module

A module's properties are used to create a deployment descriptor forthe module. The properties represent the deployment descriptor elementsas defined in the J2EE specifications. The Application Assembly toolautomatically creates deployment descriptor files based on values entered inthe property dialog boxes and wizards. The deployment descriptor is anXML document that contains application configuration data that the run-timeuses. In general, a deployment descriptor contains the followinginformation:

- Information about the content of the module being assembled. Forexample, for an EJB module, the deployment descriptor lists each enterprisebean's class, home interface class, remote interface class, whether thebean is an entity or session bean, and the bean's attributes (such aspersistence management type and primary key class for entity beans).
- References to a module's internal and external dependencies (such asenterprise beans, databases, and resource connection factories needed by themodule). Internal dependencies are dependencies on other componentswithin the same module. External dependencies are dependencies oncomponents residing outside of the module. For example, an enterprisebean in an EJB module can require another enterprise bean that is not packagedin the same module. The deployment descriptor can contain references tothe remote bean's home interface. The deployment descriptor canalso include JNDI binding information that the container uses to locate theremote bean at install time.
- Run-time-specific information needed by the application. Forexample, the servlet mappings needed for a Web application or the persistencemanagement (BMP or CMP) to be used by an entity bean.
- References to security roles. Security information is used when themodule is deployed.

In addition to XML files, the Application Assembly Tool automaticallygenerates XMI files to store binding and IBM extension information.Binding information maps a resource (enterprise bean or resource connectionfactory object) from the logical name specified in the deployment descriptorto its actual name in the global JNDI namespace. IBM deploymentdescriptor extensions are additions to the standard descriptors for J2EEapplications, Web applications, and enterprise beans. The extensionsallow you to specify properties that enable Enterprise Edition or legacy(older) systems to work in the WebSphere Application Serverenvironment. For example, an extension property is used to define howto manage transaction scoping for an enterprise bean's methods.

A J2EE application contains one application-level deployment descriptorfile, governing the application as a whole. It also contains acomponent-level deployment descriptor file, one governing each module in theapplication. For example, the application-level deployment descriptorspecifies security information for the application and defines how often theapplication is to be reloaded. The component-level deploymentdescriptor (for example, for an EJB module) specifies the following:

- General properties of the EJB module, such as the location of class filesneeded for a client program to access the enterprise beans in the module andthe icons to be associated with the module.
- The deployable enterprise beans that the module will contain.
- Security roles used to access resources in the module.
- Transaction attributes for the enterprise bean methods.
- A default datasource to be used by entity beans in the module, if one isnot specified.

6.3.3: Generating deployment code for modules

Deployment code for an application can be generated when the application isinstalled in WebSphere Application Server. Alternatively, it can bemanually generated before installation. During code generation, theApplication Assembly Tool invokes the Deployment Tool for Enterprise JavaBeans(ejbdeploy) to do the following:

1. Generate and compile container implementation classes.
2. Process the implementation classes by using the RMIC compiler, whichgenerates the client-side stub and server-side skeletons needed for remotemethod invocation (RMI).
3. Validate the JAR file and display error messages if there are any J2EEspecification violations within the JAR file. Verifying archive filesis important for successful generation of deployment code for theapplication.
4. For CMP entity beans, generate persistence code.
5. Insert the files back into the JAR file.

For CMP entity beans, if the JAR file contains a map and schema document,that schema is used. If the JAR file does not contain a map and schemadocument, the Application Assembly Tool uses a top-down mapping to generatethe following files:

- META-INF\Table.ddl
- META-INF\map.mapxmi
- META-INF\Schema\schema.rdbxmi

Note:

For Advanced Single Server Edition of this product, only the Table.ddlfile is generated.

The user must create the database table manually. The generatedTable.ddl file can be used as a guide, or the user can create the tableby some other means.

If you do not wish to use the default top-down mapping, the recommendedtool is VisualAge for Java. For information on default mappings, seethe documentation for Deployment Tool for Enterprise JavaBeans.

To migrate CMP entity beans from WebSphere Application Server 3.5for use in version 4.0, use the -35 option of the Deployment Tool forEnterprise JavaBeans. This option is recommended only if you need topreserve use of the mapping rules that were used in the 3.5 version ofthe Deployment Tool.

6.3.4: Verifying archives

Verifying archive files is important for successful generation ofdeployment code for the application. During verification, theApplication Assembly Tool checks that an archive is complete, and thatdeployment descriptor properties and references contain appropriatevalues. Specifically, the verification process checks thefollowing:

- Required deployment properties must contain values.
- Values specified for environment entries must match their associated Javatypes.
- In both EAR and WAR files:
    - For EJB references, the target enterprise bean of the link mustexist.
    - For security role references, the target role must exist.
    - Security roles must be unique.
- For EAR files, each module listed in the deployment descriptor must existin the archive.
- For WAR files, the files for icons, servlets, error, and welcome pageslisted in the deployment descriptor must have corresponding files in thearchive.
- For EJB modules:
    - All class files referenced in the deployment descriptor must exist in theJAR file.
    - Method signatures for enterprise bean home, remote, and implementationclasses must be compliant with the EJB 1.1 specification.

If errors are found, a message is displayed in a scrolling window.

6.4: Installing applications and setting classpaths

This article describes how to install an application into the WebSphere Application Server environment. This includes placing the files comprising an application onto the physical system containing the WebSphere Application Server product and updating the application server configuration with information about the application.

Installing applications (or standalone modules) into the application server runtimeinvolves the following tasks:

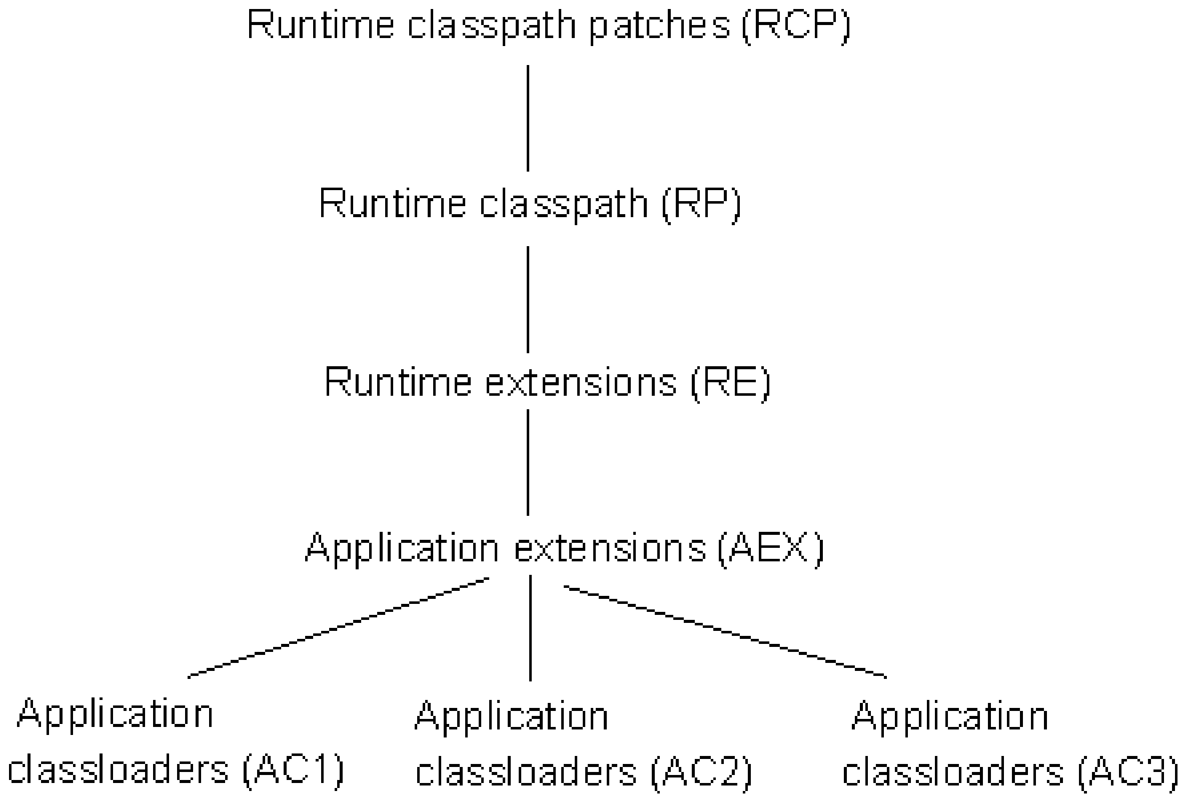
- Configure the application settings in the administrative console,by performing an application installation task (requires theadministrative server to be running)
    - 6.6.1.1.1: Installing enterprise applications and modules with the Java administrative console
- Placing the application files in a directory location, usually relativeto the IBM WebSphere Application Server *product_installation_root*
    - Place the files in the location of your choice. Specify the location when you are performing the application installation task.
- Setting classpaths as needed, in addition to those set during application assembly
    - Setting classpaths

6.4.1: Setting classpaths

- The classpaths and their search orders
- The system classpath
- Module visibility for setting isolation levels
- Guidelines for reloading classes
- Guidelines for packaging classes

The classpaths and their search orders

The relationship among the classloaders represents thefact that classes loaded by a particular classloader canreference other classes as long as these other classes canbe loaded by the same classloader or any of its ancestors (but notits children).


Runtime classpath patches (RCP)
Runtime classpath (RP)
Runtime extensions (RE)
Application extensions (AEX)
Application classloaders (AC1)
Application classloaders (AC2)
Application classloaders (AC3)

Runtime classpath patches (RCP)

- **Description:** Classes and JAR files in the *product_installation_root*/classes directory
- **How to set and view contents:** The location is fixed by the System property ws.ext.dirs. You canview the directory contents to see the contents of this classpath.
- **Search order:** RCP -> RP -> RE. The runtime classpath patches override any copies of the same files lower in the classpath tree
- **Typical contents:** e-Fixes and other APARs to be applied to the application server runtime

Runtime classpath (RP)

- **Description:** Classes and JAR files in the *product_installation_root*/lib directory
- **How to set and view contents:** The location is fixed by the System property ws.ext.dirs. You canview the directory contents to see the contents of this classpath
- **Search order:** RCP -> RP -> RE. The runtime classloader checks the RCP, and then the RP
- **Typical contents:** The core WebSphere Application Server runtime

Runtime extensions (RE)

- **Description:** Classes and JAR files in the *product_installation_root*/lib/ext directory
- **How to set and view contents:** The location is fixed by the System property ws.ext.dirs. You canview the directory contents to see the contents of this classpath
- **Search order:** RCP -> RP -> RE. The runtime classloader checks the RCP, then the RP,then the RE
- **Typical contents:** Extensions to the core WebSphere Application Server runtime

Application extensions (AEX)

- **Description:** Classes and JAR files in the *product_installation_root*/lib/app
- **How to set and view contents:** The location is fixed by the WebSphere Application Server runtime andcannot be configured. You canview the directory contents to see the contents of this classpath
- **Search order:** The runtime classloader checks the AEX, then RCP -> RP -> RE.
- **Typical contents:** Class libraries that need to be shared among all J2EE applications installed on the server. Because these classes are not visible to the WebSphere Application Server runtime classloaders (RPC, RP, and RE), this classpath can contain updated versions of common libraries (such as JAXP)that are present in the runtime.

  Place dependent JAR files in this directory, too.

Application classloaders (AC)

- **Description:** Classpaths as specified for WAR and JAR modules
- **How to set and view contents:** The classpath for WAR and EJB JAR files are set usingMANIFEST Class-Path entries. You can modify these settings using the ApplicationAssembly Tool.
- **Search order:** Depends on the module type.
  - **WAR modules** follow the search order: module classloader -> AEX -> RCP -> RP -> RE

    Within the module classloader for the WAR file, the search order is:

    1. Root of the WAR file
    2. WEB-INF/classes directory contents
    3. JAR files in the WEB-INF/lib directory

    The module classloader includes the JAR or WAR and its pieces, WEB-INF/classes and WEB-INF/lib, and all of the entries specified by the Modules' class-path entry.
  - **EJB JAR modules** follow the search order: AE -> RCP -> RP -> RE -> module classloader

  The **delegation mode** of WAR and EJB JAR files can be changed by defining the following system properties:

    - com.ibm.ws.classloader.warDelegationMode (false by default)
    - com.ibm.ws.classloader.ejbDelegationMode (true by default)

  Set the value of each property to true or false:

    - **true**: AE -> RCP -> RP -> RE -> module classloader
    - **false**: module classloader -> AE -> RCP -> RP -> RE

- **Typical contents:** Application code, including servlets, enterprise beans, JavaServer Pages (JSP)files, JavaBeans, and supporting class libraries

The system classpath

If you use the classpath setting in the administrative console to specify additional entries to be added to the classpath, these entries will be appended to the classpath. If you specify a System property, ws.ext.dirs, these entries will be appended to the ws.ext.dirs.

The WebSphere Application Server runtime is loaded by a bootstrap classloader that loads classesfrom the directories, JAR files in specified directories, and JAR files specified by the systemproperty ws.ext.dirs. Because the system classloader is the parent of the WebSphere bootstrapclassloader, it is important to note that the classes put in the system classpath will notbe able to load classes in the WebSphere runtime (including J2EE APIs).

Module visibility for setting isolation levels

The relationship among the classloaders implies that it is notusually possible to access the classes of one Web application from another Webapplication.

The application server settings include a propertyfor specifying the module visibility, which determines the level of isolationamong the classpaths for various application components.

Note that the default value of the module visibility setting impacts whether you can port applications from previousWebSphere Application Server versions or other editions (from *Advanced SingleServer Edition* Version 4.0 to *Advanced Edition* Version 4.0, for example)without reassembling the applications or resetting the field value.

See the module visibilityfield description in the application server property reference for more information.

Guidelines for reloading classes

To update (reload) the classes being used by an application, it is bestto stop that J2EE application and start it again. This ensures that dependentclasses between cooperating modules will be reloaded in unison, eliminating potential ClassCastExceptions that occur when mismatched versions of classesare being used.

However:

- EJB JAR files can be reloaded by stopping just the EJB module and startingit again. Any dependent EJB or WAR modules will need to be restarted as well.
- You can also reload EJB modules by specifying a reload interval in the EAR extension of the EAR file. Ifthe EJB classes change, then the entire J2EE application (EAR) is restarted, not justthe EJB module.
- WAR modules can be configured to reload automatically when their contents change. Thisis accomplished by setting an Web module assembly property using the Application Assembly Tool.

Guidelines for packaging classes

- The EJB JAR modules and WAR modules comprising an application should be packaged togetherin the same EAR module
- When a Web module (WAR) accesses an EJB module, it should not package theEJB interfaces and stubs in the WAR. Rather, it should express the dependency onthe EJB JAR using a MANIFEST Class-Path entry. When using the ApplicationAssembly Tool, set the classpath property of the Web module.
- Common classes that need to be shared among Web modules and EJB JAR modules shouldbe packaged into a separate JAR file. Add the JAR file to the EAR file. Reference the JARfile in the MANIFEST Class-Paths of both the EJB JAR and Web module (WAR) files.
- Class library JAR files to be used only by WAR modules should be added to the WEB-INF/libdirectory of the Web module

6.5: Maintaining and updating applications

This section includes information about monitoring daily operations and the health of applications, updating applications, and performing necessary tasks after updating an application (such as stopping a process and starting it again).

To update the contents of an application, use the Application Assembly Tool. Typical maintenance tasks include adding or editing assembly properties, addingor importing modules into an application, and adding enterprise beans, Web components,and files.

What should you do if an application changes?

Depending on what you have changed, you can sometimes replacean application with a newer version, or modify configuration settingsor application bindings, without having to disrupt the applicationserver or the installed application.

Article 6.5.1 discusses various changes and the minimum actionsyou must take to update the application in your server runtime.

In the very worst case, you will need to:

1. Reassemble the application, using the Application Assembly Tool.
2. Uninstall the current version of the application.
3. Install the new version of the application.

6.5.1: Hot deployment and dynamic reloading

This article outlines the types of changes that can be made to application servers and their contents with respect to hot deployment and dynamic reloading. In other words, it discusses what you can change,and how, without having to stop the server and start it again.

Hot deployment is the process of adding new components (such as enterprise beans, servlets, and JSP files) to a running server without having to stop the application server process and start it again.

Dynamic reloading is the ability to change an existing component without needing to restart the server in order for the change to take effect. Dynamic reloading involves:

- Changes to the implementation of a component of an application, such as changing the implementation of a servlet
- Changes to the settings of the application, such as changing the deployment descriptor for a Web module

View information about updating and redeploying:

- WAR Files
- EJB Jar Files
- Application
- Server Configuration
- HTTP Plugin Configuration Changes

WAR Files

Change an existing JSP File

- User Action: None
- Comments: The changed JSP can be placed directly in the *product_installation_root*/installedApps/<app name>/<module name> directory, or the appropriate subdirectory. The change will be automatically detected and the JSP will be recompiled and reloaded.
- Hot deployment: Non-applicable
- Dynamic reloading: Yes

Add a new JSP File to an existing application

- User Action: None
- Comments: The new JSP can be placed directly in the *product_installation_root*/installedApps/<app name>/<module name> directory, or the appropriate subdirectory. The new file will be automatically detected and compiled on the first request to the page.
- Hot deployment: Yes
- Dynamic reloading: Yes

Change an existing Servlet class (edit and recompile)

- User Action: None (if reloading is enabled). If reloading is disabled, the server will have to be restarted to detect the change.

- Comments: The new version of the servlet .class file can be placed directly in the *product_installation_root*/installedApps/ <app name>/<module name>/WEB-INF/classes directory. If the .class file is part of a jar file, the new version of the jar file can be placed directly in *product_installation_root*/installedApps/<app name >/<module name>/WEB-INF/lib. In either case, the change will be detected, the web application will be shutdown and reinitialized, picking up the new class.
- Hot deployment: Non-applicable
- Dynamic reloading: Yes

Change a dependent class of an existing Servlet class

- User Action: None (if reloading is enabled). If reloading is disabled, the server will have to be restarted to detect the change.
- Comments: The new version of the dependent .class file can be placed directly in the *product_installation_root*/installedApps /<app name>/<module name>/WEB-INF/classes directory. If the .class file is part of a jar file, the new version of the jar file can be placed directly in *product_installation_root*/installedApps/<app name>/ <module name>/WEB-INF/lib. In either case, the change will be detected, the web application will be shutdown and reinitialized, picking up the new class.
- Hot deployment: Non-applicable
- Dynamic reloading: Yes

Add a new Servlet using the Invoker (Serve Servlets by class name) facility or add a dependent class to an existing application

- User Action: None (if reloading is enabled). If reloading is disabled, the server will have to be restarted to detect the change.
- Comments: The new .class file can be placed directly in the *product_installation_root*/installedApps/<app name>/<module name>/WEB-INF/classes directory. If the .class file is part of a jar file, the new version of the jar file can be placed directly in *product_installation_root*/installedApps/<app name>/<module name>/WEB-INF/lib. This case is treated the same as changing an existing class. The difference is that adding the servlet/class does not immediately cause the web application to reload, since the class has never been loaded before. The class simply becomes available for execution
- Hot deployment: Yes
- Dynamic reloading: Non-applicable

Add a new Servlet, including a new definition of the servlet in the web.xml deployment descriptor for the application

- User Action: None (if reloading is enabled). If reloading is disabled, the server will have to be restarted to detect the change.
- Comments: The new .class file can be placed directly in the *product_installation_root*/installedApps/<app name>/<module name>/WEB-INF/classes directory. If the .class file is part of a jar file, the new version of the jar file can be placed directly in *product_installation_root*/installedApps/<app name>/<module name>/WEB-INF/ lib. The web.xml file can be edited in place or copied into the *product_installation_root*/installedApps/<app name>/<module name>/WEB-INF directory. The new .class file will not trigger a reload of the application. However,

updating the web.xml file will cause the application to reload at the next reload interval. Once the reload is complete, the new servlet will be available for service.

- Hot deployment: Yes
- Dynamic reloading: Non-applicable

Modification to the web.xml file of an application

- User Action: None (if reloading is enabled). If reloading is disabled, the server will have to be restarted to detect the change.
- Comments: You can change most of the parameters of the web.xml file without restarting the server. However, there are exceptions, listed here:
    - <security-constraints> A security constraint of the application
    - <security-role> Change, add, or delete security roles
    - <login-config> The login config values
    - To change the listed properties, you will need to restart the module using the DrAdmin command.
- Hot deployment: Yes
- Dynamic reloading: Yes

Modification to the ibm-web-ext.xmi of an application

- User Action: None (if reloading is enabled). If reloading is disabled, the server will have to be restarted to detect the change.
- Comments: You can change all of the extension settings. The only warning is if you set the reloadInterval property to zero (0) or you set reloadEnabled property to false, the application will reload once at the time of the change. However, after this final reload, the web module will no longer detect any changes of any kind. Both of these changes disable the reloading function. The only way to re-enable reloading and the Dynamic functions is to change the appropriate property and restart the module using the DrAdmin command.
- Hot deployment: Non-applicable
- Dynamic reloading: Yes

Modification to the ibm-web-bnd.xmi of an application

- User Action: Restart the Module
- Comments: You can use the DrAdmin command to issue a restart of module of the application. The bindings will be reloaded at that time.
- Hot deployment: Non-applicable
- Dynamic reloading: Yes

EJB Jar Files

Modification to ejb-jar.xml of an application

- User Action: Restart the Module
- Comments: You can use the DrAdmin command to issue a restart of module of the application. The changes will be reloaded at that time.

- Hot deployment: Non-applicable
- Dynamic reloading: Yes

Modification to ibm-ejb-jar-ext.xmi

- User Action: Restart the Module
- Comments: You can use the DrAdmin command to issue a restart of module of the application. The changes will be reloaded at that time.
- Hot deployment: Non-applicable
- Dynamic reloading: Yes

Modification to ibm-ejb-jar-bnd.xmi

- User Action: Restart the Module
- Comments: You can use the DrAdmin command to issue a restart of module of the application. The changes will be reloaded at that time.
- Hot deployment: Non-applicable
- Dynamic reloading: Yes

Modification to Table.ddl for an EJB Jar

- User Action: Rerun DDL file on user database server
- Comments: This change really has no effect on the application server and is a change to the database table schema for the EJBs
- Hot deployment: Non-applicable
- Dynamic reloading: Non-applicable

Modification to the Map.mapxmi file for an EJB Jar

- User Action: Redeploy the EJB and restart the module
- Comments: A change to this file requires you to regenerate the deployed code artifacts for the EJB, apply the new EJB jar to the server and restart the module using the DrAdmin program.
- Hot deployment: Non-applicable
- Dynamic reloading: Yes

Modification to the Schema.dbxmi file for an EJB Jar

- User Action: Redeploy the EJB and restart the module
- Comments: A change to this file requires you to regenerate the deployed code artifacts for the EJB, apply the new EJB jar to the server and restart the module using the DrAdmin program.
- Hot deployment: Non-applicable
- Dynamic reloading: Yes

Update the Implementation class for an EJB or a dependent class of the implementation class for an EJB

- User Action: Restart the Module and/or Application (This is not working on the current driver)
- Comments: After changing the Jar/Class File in the installedApps folder, you must restart the module or application that the EJB is part of. You should restart the application is other modules in the application use the changed module. If the changed module is used by modules in other applications, you should restart those applications/modules.
- Hot deployment: Non-applicable
- Dynamic reloading: Yes

Update the Home/Remote interface class for an EJB

- User Action: Redeploy the EJB and restart the application
- Comments: A changing the interfaces of the EJB requires you to regenerate the deployed code artifacts for the EJB, apply the new EJB jar to the server and restart the application that the EJB jar is a part of. This will always work as long as you are using Module or Application visibility for class loading on the server.
- Hot deployment: Non-applicable
- Dynamic reloading: Yes

Add a new EJB to a running Server

- User Action: Apply the new Jar to the installedApps folder and restart the application using the DrAdmin program.
- Comments: For this scenario, I am assuming that the EJB is part of an existing EJB Jar file.
- Hot deployment: Yes
- Dynamic reloading: Yes

Application

Modification to the application.xml for an application

- User Action: Restart the Application
- Comments: The application must be restarted with the DrAdmin program.
- Hot deployment: Non-applicable
- Dynamic reloading: Yes

Modification to the ibm-app-ext.xmi file for an application

- User Action: Restart the Application
- Comments: The application must be restarted with the DrAdmin program.
- Hot deployment: Non-applicable
- Dynamic reloading: Yes

Modification to the ibm-app-bnd.xmi file for an application

- User Action: Restart the Application
- Comments: The application must be restarted with the DrAdmin program.

- Hot deployment: Non-applicable
- Dynamic reloading: Yes

Modification to a non-module Jar contained in the EAR

- User Action: Restart the Application
- Comments: Use the DrAdmin tool to restart the application containing the Jar that has changed
- Hot deployment: Yes
- Dynamic reloading: Yes

Add a new EJB module to an existing, running application

- User Action: Restart the Server
- Comments: Because adding a new module requires a change to server-cfg.xml, the server must be restarted to pick up the change.
- Hot deployment: No
- Dynamic reloading: No

Add a new web module to an existing, running application

- User Action: Restart the Server
- Comments: Because adding a new module requires a change to server-cfg.xml, the server must be restarted to pick up the change.
- Hot deployment: No
- Dynamic reloading: No

Addition of new application to server-cfg.xml of running server

- User Action: Restart the Server
- Comments: Because adding a new application requires a change to server-cfg.xml, the server must be restarted to pick up the change.
- Hot deployment: No
- Dynamic reloading: No

Server Configuration

Modification to server-cfg.xml file of running server

- User Action: Restart the Server
- Comments: Any change to server-cfg.xml requires the server to be restarted to pick up the change.
- Hot deployment: No
- Dynamic reloading: No

HTTP Plugin Configuration Changes

Modifications to application.xml to change the context root of a war file

- User Action: Generate the Plugin config file through the Web-browser admin or through the GenPluginCfg.bat/sh script.
- Comments:
- Hot deployment: Yes
- Dynamic reloading: Yes

Modifications to web.xml to add, remove, or modify a servlet mapping

- User Action: Possibly have to re-generate the http plugin configuration file through the Web-browser admin or through the GenPluginCfg.bat/sh script.
- Comments: If the web application has file serving enabled or has a servlet mapping of '/', you do not have to regenerate the plugin configuration. In all other cases the regeneration is required.
- Hot deployment: Yes
- Dynamic reloading: Yes

Modification to server-cfg.xml to add, remove, or modify a HTTP transport, to add or remove a Virtual Host, or to add, remove, or modify a Virtual Host alias

- User Action: Generate the Plugin config file through the Web-browser admin or through the GenPluginCfg.bat/sh file.
- Comments: Any of these changes affect the plugin-cfg.xml file and require you to regenerate that file.
- Hot deployment: Yes
- Dynamic reloading: Yes

6.5.3: Operational dependencies and best practices

Some configurationsrequire you to start your servers in a certain order or the startwill fail. This article documents best practices and "gotchas," as theyare discovered through use of the product in realistic environments.

See also the IBM Redbooks about this product edition and version.

Configurations involving EJB modules and Web modules in separate servers

If WebSphere Application Server is configured such that you have separate servers dedicatedto running your EJB modules and Web modules, you will have trouble if youstart the servers in the wrong sequence. The following scenario illustrates the problem andone solution.

Configuration:

- Application (Deployed_WLMApp.ear) with both EJB modules and Web modules
- Two application servers:
    - WLMServletServer (for the Web modules)
    - WLMEJBServer (for the EJB modules)

Error situation:

1. Create WLMEJBServer.
2. Create WLMServletServer.
3. Install application:
    - Select that EJB beans run on WLMEJBServer.
    - Select that Web module runs on WLMServletServer.
4. Start WLMServletServer:
    - Web module loads and HTTP transport on port 9080
5. Start WLMEJBServer.
6. Obtain ERROR because the port 9080 is already in use.

The WLMEJBServer cannot be started, even though no Web modules are installedto it to require the use of port 9080.

Solution:
If the WLMEJBServer had been started first, everything would havebeen fine, because the WLMEJBServer would have loaded without locking port 9080. The port would have been available when the WLMServletServer started.

6.6: Tools and resources quick reference

This article provides various ways to access information about administeringthe product and your applications being hosted by the product. Find informationbased on:

- The tasks that you want to perform
- The resources that you want to administer
- The tools that you want to use

Tasks quick reference

Click a check mark in the table to go to the instructions for thetask.

	Application Assembly Tool	AE adminstrative console	Application client resource configuration tool
Configuring client resources			✓
Creating an application client	✓		
Inspecting the runtime values of properties		✓	
Finding resources		✓	
Starting and stopping resources		✓	
Pinging resources		✓	
Removing objects from EAR files	✓		✓
Configuring enterprise applications	✓	✓	
Creating the EAR.WAR and JAR files in an application	✓		
Installing enterprise applications		✓	
Starting, stopping, and restarting applications and modules		✓	
Uninstalling applications			
Updating application configurations	✓	✓	
Exporting application configurations		✓	
Viewing deployment descriptor information for enterprise applications (read-only)		✓	

Showing the status of enterprise applications		✓	
Exporting enterprise applications		✓	
Updating nodes		✓	
Reconnecting administrative nodes		✓	
Regenerating WebSphere plug-in configurations		✓	
Configuring tracing on administrative nodes		✓	
Starting and stopping application servers		✓	
Updating application servers		✓	
Configuring new EJB containers		✓	
Updating EJB containers		✓	
Creating an EJB module	✓		
Installing EJB modules		✓	
Viewing deployment descriptor information for EJB modules		✓	
Updating EJB module configurations		✓	
Showing the status of EJB modules		✓	
Moving EJB modules to other application servers		✓	
Exporting table DDLs of EJB modules		✓	
Updating Web container configurations		✓	
Creating a Web module	✓		
Installing Web modules		✓	
Viewing deployment descriptor information for Web modules		✓	
Updating Web module configurations		✓	
Showing the status of Web modules		✓	
Moving Web modules to other application servers		✓	

Updating session management settings		✓	
Configuring new HTTP transports		✓	
Removing HTTP transport configurations		✓	
Updating transport configurations		✓	
Configuring new data source providers		✓	✓
Configuring new data sources		✓	✓
Removing data source providers (JDBC providers) and data sources		✓	✓
Updating data source and data source provider configurations		✓	✓
Configuring new custom services		✓	
Removing custom services		✓	
Updating custom service configurations		✓	
Administering virtual hosts		✓	
Configuring security	✓	✓	
Administering server traces		✓	
Administering the transaction service		✓	
Analyzing resources		✓	
Specifying server groups		✓	
Editing services of server groups		✓	
Configuring OLT and Debugger		✓	
Configuring Object Request Brokers (ORBs)		✓	
Monitoring and tuning performance		✓	
Using the Performance Tuner Wizard		✓	
Configuring JVM settings		✓	
Configuring new mail sessions		✓	✓

Updating mail session configurations		✓	✓
Removing mail session configurations		✓	✓
Configuring new URL providers		✓	✓
Updating URL provider configurations		✓	✓
Removing URL provider configurations		✓	✓
Removing URL configurations		✓	✓
Installing and uninstalling URL providers on nodes		✓	
Configuring new JMS providers		✓	✓
Configuring new JMS connection factories		✓	✓
Configuring new JMS destinations		✓	✓
Updating JMS provider, connection factory, and destination configurations		✓	✓
Installing and uninstalling JMS providers on nodes		✓	
Removing JMS providers, connection factories, and destinations		✓	✓
Updating JMS provider, connection factory, and destination configurations		✓	✓
Administering WebSphere administrative domains		✓	
Configuring J2C connection factories		✓	
Installing and uninstalling J2C adapters on nodes		✓	
Administering generic servers		✓	
Adding generic servers		✓	
Regenerating the Web server plug-in configuration		✓	

Resources quick reference

The following tables provide quick entry points into descriptions,tasks, and settings of various objects that you can administer.

ℹ See also the administrative model, which enables you to click a resource in a topology diagram to go straight to the information about the resource.

Assembling applications and generating code for deployment

Description	Administrative overview	Settings
Application assembly	See Application Assembly Tool	See properties for applications, EJB modules, Web modules, and application client modules.
Generating code for deployment	See Application Assembly Tool	Assembly properties for generating deployment code

Configuring the overall administrative environment

Description	Administrative overview	Settings
Administrative domains	Administering WebSphere administrative domains	Administrative domain properties
Nodes	Administering nodes	Node properties
Administrative servers	Administering WebSphere administrative servers	Administrative server properties
Virtual hosts	Administering virtual hosts	Virtual host properties

Configuring specific application servers

Description	Administrative overview	Settings
Application servers	Administering application servers	Application server properties
JVMs	Administering JVMs	JVM properties
EJB containers	Administering EJB containers	EJB container properties
Web containers	Administering Web containers	Web container properties

Configuring application components installed in a runtime

Description	Administrative overview	Settings
Applications	Administering applications	Application properties
EJB modules	Administering EJB modules	EJB module properties
Web modules	Administering Web modules	Web module properties
Application client modules	Administering application client modules	Application client module properties

Configuring resource providers for data access and other functions

Description	Administrative overview	Settings

JavaMail resources	Administering JavaMail providers and sessions	JavaMail provider properties and JavaMail session properties
URL resources	Administering URL providers and URLs	URL provider properties and URL properties
JMS resources	Administering JMS providers	JMS provider properties and JMS connection factory properties and JMS destination properties
J2C resources	Administering J2C	J2C properties
Data access	Administering database connections	JDBC provider properties Data source properties

Configuring various services to support applications in the runtime

Description	Administrative overview	Settings
Security	Administering security	Security properties
Sessions and Session Managers	Administering Session Managers	Session Manager properties
User Profile Managers	Administering User Profile Managers	Non-applicable -- no settings in console
Tracing	Administering the product trace	Trace properties
Transactions	Administering transactions	Transaction properties
Custom services	Administering custom services	Custom service properties
Resource Analyzer	Administering Resource Analyzer	Resource Analyzer properties
Performance Monitor	Administering Performance Monitors	Performance Monitor properties
OLT and Distributed Debugger	Administering OLT and Distributed Debugger	OLT and Distributed Debugger properties

Configuring resources related to routing and distributing application requests

Description	Administrative overview	Settings
Transports	Administering transports	Transport properties
Server groups	Administering server groups	Server group properties
Plug-ins for Web servers	Administering WebSphere plug-ins for Web servers	Plug-in properties

Tools quick reference

Use the following information to compare, select among, and learn how to use the features of the various product interfaces.

- Summary of available product interfaces (consoles and tools)
- Scope of administrative capabilities

Summary of available product interfaces (consoles and tools)

The table summarizes the available product interfaces (with theexception of the installation program and the problem determination tools).

Interface	Purpose
Administrative console (Java based)	Graphical interface for configuring and operating application servers and other resources that support applications in a runtime environment
Application assembly tool	Graphical interface for assembling applications from application components and modifying theirJ2EE deployment descriptors
wscp	Command line interface for operational tasks, such as starting and stoppingapplication servers
XMLConfig	Command line interface for configuration tasks, including the import and export of human readable XML files describing resource configurations
Configuration using property files	Ability to manually edit various property files in the product directory structurefor miscellaneous configuration purposes
Client launcher (launchClient)	Command line interface for starting a client application.
EAR expander	Command line interface for expanding .ear files into the directory structure typicalfor use in the runtime of this product. Should not need to use this tool on an ordinarybasis.
Application client resource configuration tool	Graphical interface for configuring resources that support client applications in a runtime environment. (See also information on developinga Java client.)
SOAP EAR Enabler	Command line interface for enabling a set of SOAP services within an Enterprise Application Archive (EAR) file.
EJBCommandTarget	Enables administrators to execute commands in a designated server without providing their own CommandTarget implementation.
Resource Analyzer	Enables administrators to gather and analyze performance data of servlets,enterprise beans, and related resources.
XML-SOAP Admin tool	A modified version of the Apache SOAP XML-Admin interface for managing each SOAP-enabled EAR file.
Deployment tool	Interface for generating code for deployment. Useful for more complicated or customscenarios. For simple scenarios, the use of this tool is hidden from the administrator. It is invokedby other consoles and tools. (See also the article about generating codefor deployment).

Scope of administrative capabilities

Administrators that have access to an administrative client canuse the full capabilities of the client. At this time, it is **not**possible to allow administrators to perform some operations (such as starting andstopping servers) but deny them the ability to perform other operations(such as creating and configuring resources).

6.6.0: About user assistance

The InfoCenter provides a copy of the help files (or user assistance files) that are installed with the product. In some cases, the InfoCenter versions of the help filesare more up to date.

Articles named 6.6.0.* approach administration by describing how to use theavailable tools. Each set of articles about a particular administrative toolprovides entry points into tasks that you can perform with the tool.

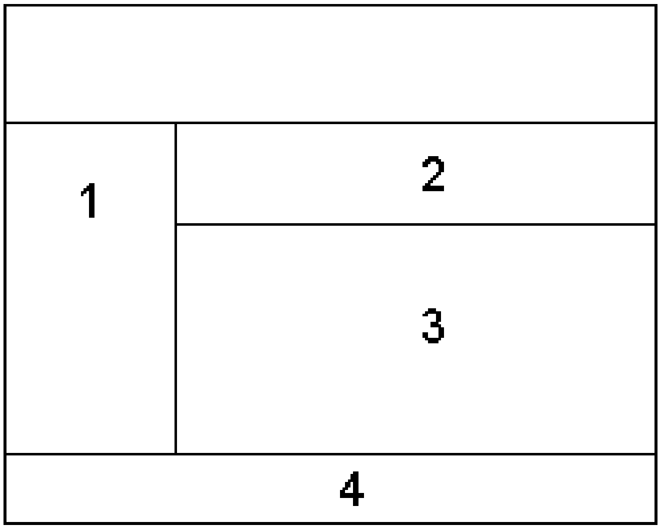
Articles named 6.6.1 through 6.6.*N* complement the tool-based documentationby providing entry points to administering various object types, such as nodesand application servers. Each entry point provides links to tasks and tools relatedto administering the object type.

6.6.0.1: Using the Java administrative console

The WebSphere Administrative Console is a graphical, Java-basedadministrative client to the IBM WebSphere Application Server administrativeserver. This administrative console supports the full range of product administrative activities.

The console has these main areas:

1. The tree view
2. The details view
3. The properties view
4. The messages area



The console features a tree view for surveying andmanipulating resources in the administrative domain. In addition, taskwizards are provided to lead administrators through the creation and protection of new resources, such as applications and application servers.

The console also provides a runtime object inspector.

You can resize these areas as desired.

The tree view

You use the tree view on the left side of the console to survey, select, and manage components in the WebSphere administrative domain.

Clicking on a "+" beside a tree folder or item expands the tree for the folder or item. Similarly, clicking on a "-" collapses the tree for the folder or item.Double-clicking on an item in the tree view toggles its state betweenexpanded and collapsed.

Clicking on (selecting) a folder or item in the tree view controls the contentdisplayed on the right side of the console. Selecting a folder or item alsoenables menu choice supporting actions that an administrator can take in regardto that folder or item.

Selecting a non-folder item from the tree view populates the properties view with the properties of the selected item, if the properties view is visible.

The details view

The details view on the upper-right side of the console displays informationon the selection in the tree view. When you click a folder in the tree view, the details view lists information on instances of that folder type. When a non-folder resource in the details view is selected, the properties for the resourceare shown below the details view, provided the properties view is visible.

Double-clicking on an item in the details view expands a collapsed tree viewto show the item. Double-clicking on an item in the details view also selects that item in the tree view and populates the details view with its children.

The properties view

The properties view on the lower-right of the console displays the properties sheet for a resource selected in the details view or in the tree view. You use a properties sheet to view and edit property values.

Note that the properties view is shown only when the **Show Property Pane**toggle on the console **View** menu is enabled (the default).

Also note that a properties sheet is like a properties dialog. To access a properties dialog, you right-click a resource in the tree viewand selects **Properties** from the popup menu.

Further, a properties sheet does not display runtime (in use) property values. To see runtime property values you must use the runtime inspector, which is available from the **View -> Show Runtime Inspector** menu.

The messages area

The messages area at the bottom of the console lists messages returned bythe WebSphere Administrative Server as well as messages about events such as successful completions and fatal errors.

You can customize the contents of the message list and limit themessage log size by clicking **Options** and setting message preferences in the dialog that opens.

Note that the messages area is shown only when the **Show Console Messages**toggle on the console **View** menu is enabled (the default).

6.6.0.1.1: Starting and stopping Java administrative consoles

Preconditions

The WebSphere administrative server must be running before youstart the Java administrative console. By default, the server does not startautomatically when you start the machine.

The Java administrative console (also known as the WebSphere AdministrativeConsole) is a Java process. As such, it must be able to locate a JDK supported by IBM WebSphere Application Server, asspecified on the prerequisites Web page.

Ensure that the Java console and WebSphere administrative serverwith which it is communicating are using the same supported JDK (and ORB) level.

Instructions for AIX, HP-UX, Linux, and Solaris

1. Start the administrative server, or ensure it is already running.
2. Start the administrative console:
    1. Change directory to the bin directory under the product installation root.

       ```
       cd product_installation_root/bin
       ```
    2. Ensure that the DISPLAY variable is set in the shell from which the administrative client script will be run:

       ```
       $ export DISPLAY=localMachineName:0.0
       ```
    3. Run the administrative client script:

       ```
       ./adminclient.sh
       ```

To stop the console, use the Console -> Exit option on the consolemenu bar.

Instructions for Windows NT and 2000

From the Start menu, select Programs -> IBM WebSphere -> Application Server 4.0 -> Administrator's Console.

Another way to start the console is to run the administrative client script:

```
product_installation_root/bin/adminclient.bat
```

To stop the console, use the Console -> Exit option on the consolemenu bar.

6.6.0.1.2: Inspecting the runtime values of properties with the Java administrative console

To inspect the runtime (in use) values of properties, you can use the runtime inspector. The runtime inspector lists attributes of a property selected in the administrative console and their runtime values. Examples ofruntime values include a process ID or classpath for an application server.

The inspector is a separate dialog that stays on top of other Application Server windows.

The contents of the inspector update when a different item in the console tree view or details view is selected, provided the selection has runtime attributes.Clicking on a "+" beside an attribute expands the list of attributes. Similarly, clicking on a "-" collapses the list.

The runtime inspector is shown when the **Show Runtime Inspector** toggle on the console **View** menu is enabled. View Runtime Inspector" is disabled by default.

6.6.0.1.3: Updating resources with the Java administrative console

To edit the properties of an existing resource, do one of the following:

- Select a resource (such as an instance in the tree view or details view), right-click it,and select Properties.
- Select a resource in the tree view, then select**Console -> Properties**.
- Select a resource (such as an instance in the tree view or details view) to display its properties in theproperties view. (The **View -> Show PropertyPane** toggle on the console menu bar must be enabled).

6.6.0.1.4: Removing resources with the Java administrative console

To remove a resource:

1. Select the resource in the tree view or details view.
2. Do one of the following:
    - Click **Remove** on its right-click menu
    - Use the **Console -> Remove** menu option

If a resource does not have a **Remove** menu option, it cannot be removed.

6.6.0.1.5: Finding resources with the Java administrative console

To locate resources of the same resource type within the administrative domain:

1. Access the Find Objects dialog by selecting **Console -> Find**.
2. Select a resource type from the drop-down list.
3. Select whether to retrieve all objects of the given type, or to retrieve objects havinga designated string in the object name.

   When specifying the object name, a wildcard character (*) can assist in the search.
4. Click **OK** or **Apply** to start the search.

Results will be displayed in a search results dialog.

6.6.0.1.6: Starting and stopping resources

To start or stop a resource:

1. Locate the resource in the tree view or details view.
2. Do one of the following:
   - Select an action from the **Console** menu on the menu bar
   - Select an action from the right-click menu of the resource

Actions pertaining to starting and stopping resources can include the following. See below for restrictions.

Start

Changes the resource to running state.

Restart

Stops a resource, then returns the resource to the state it was in before it (or its parent resource) was stopped.

For example, suppose you performed **Stop for Restart** (or just **Restart**) on a running administrative server node containing application server "A" in the stopped state and application server "B" in the running state. The **Restart** command would cause application server A to remain stopped. It would return application server B to the running state, along with the administrative server.

Stop

Stops the resource.

For an administrative server node, this action stops the administrative server running on the node. It does not shut down the operating system.

Stop for Restart

Stops a resource and any resources contained by the resource, in preparation for the **Restart** command.

It can be useful for stopping and restarting an administrative server that contains multiple application servers, some of which are running and others of which are stopped. Using a combination of **Stop for Restart** and then **Restart** will return the application servers to their former states (stopped or running) when the administrative server is restarted.

Force Stop

Stops the resource when the regular **Stop** action is ineffective. Use the option when the administrative server or other resource is in a strange state and cannot be stopped using the ordinary **Stop** action.

Notes and restrictions

Not all resources can be started or stopped. Depending on the resource type, all, some, or none of the actions might be available. Options could be unavailable (grayed out oromitted entirely) for resources that cannot be started or stopped, or are presently in the wrong state to be started or stopped. For example, trying to stop an alreadystopped resource is not a valid action.

6.6.0.1.7: Pinging resources with the Java administrative console

Ping a resource to see whether the resource is responding. The ping action provides a basic check of the "health" of the resource.

To ping a resource:

1. Select a resource from the tree view or details view.
2. Do one of the following:
   - Right-click the resource and select **Ping**
   - Select **Console -> Ping** from the console menu bar

For information about the ping operation, consult the documentation for the operating system in use.

6.6.0.1.8: Actions on the Java administrative console menu

The Java administrative console menu bar provides the following options:

Console

- **New -> *resource_name***

  Displays a Properties dialog for *resource_name* in which you can create an instance of the resource. You can later use a Properties dialog to update a resource instance.
- **Wizards ->**
  - **Install Enterprise Application**
  - **Create Application Server**
  - **Create Server Group**
  - **Create Data Source**
  - **Create JMS Resource**
  - **Create J2C Connection Factory**
  - **Create URL Resource**
  - **Performance Tuner**

  Displays a task wizard to assist you in creating and updating a resource. A wizard enables you to view and update *some* properties of a resource. To view and update *all* properties, use a Properties dialog.
- **Trace ->**
  - **Enabled** -- Toggles between enabling and disabling tracing of WebSphere Application Server events.
  - **Trace Settings** -- Displays a window in which you can set tracing globally for a particular component.

- **Security Center**
- **Import from XML**

  Displays a file selection dialog in which you specify the XML file to import into WebSphere Application Server.
- **Export to XML**

  Displays a file selection dialog in which you specify the target directory for an XML file containing the complete administrative domain configuration.
- **Start**
- **Stop**
- **Force Stop**
- **Ping**
- **Remove**

- **Properties**
- **Find**
- **Command History**

  Displays a window that lists WebSphere Application Server commands as well as each command's status, completion time, and result.
- **Exit**

  Ends processing of the WebSphere Application Server product and closes the administrative console window.

View

- **Show Property Pane**
- **Show Console Messages**
- **Show Runtime Inspector**

Tools

- **Application Assembly Tool**
- **Log Analyzer**
- **Resource Analyzer**

Help

- **Concept Help**

  Displays a brief and general description of the resource selected in the console.
- **Task Help**

  Displays instructions for creating, updating and performing other tasks pertinent to the resource selected in the console.
- **Field Help (F1)**

  Displays information on properties of the resource selected in the console.
- **Information Center**
- **About**

  Displays information about the WebSphere Application Server product.

6.6.0.1.9: Actions on the right-click menus of the Java administrative console

The table summarizes choices available on the right-click menus offolders or items in the tree view.

Some actions are not available for certain folders or items, or can becometemporarily unavailable due to the state of the selection. For example,if a server is already stopped, its **Stop** action will be unavailable.

Action	Result
Clone	Displays a Create Clone dialog for defining a new clone for a server group.This option is available for a server group as part of the right-click menuchoice **New -> Clone**.
Create Server Group	Displays a Create Server Group task wizard for defining a new server group or for defining one based on an existing server. The wizard can be used to set values for *some* server group properties.To view or change *all* properties, the server group properties dialog can be used.
Export Application	Displays an Export Application dialog for specifying a node and directory towhich an enterprise application should be exported. The application is saved asan EAR file.
Export Binding	Displays a dialog for specifying the exporting destination of binding files. Binding files contain location information needed to resolve external references for an application or module. For example, binding information is needed to map an external resource from its logical name (in the deployment descriptor) to its actual JNDI name.
Export Table DDL	Displays an Export Table DDL dialog for specifying a node and directoryto which table DDL files should be exported. This action is only enabled for EJB modules containing container managedpersistence (CMP) beans and for Web modules that use EJB modules containing CMP beans.
Find	Displays a Find dialog for compiling a list of all objects thathave the same object type as the selection.
Force Stop	Stops the resource when the regular **Stop** action is ineffective. Usethe option when the administrative server or other resource is in astrange state and cannot be stopped using the ordinary **Stop** action.
Install Enterprise Application	Displays an Application Installation task wizard for installing an application (EAR file) or module (JAR or WAR file).
Move	Displays a dialog for moving an EAR file to a node that contains thedestination server or server group. If a server group is named as thedestination, then the EAR file is copied to all nodes that contain theclones of the server group. Manual steps are required. Please see "Moving EJB modules to other application servers." This option is only available to EJB modules or Web modules. The EAR file containing the EJB or Web module is the EAR file moved.
New	Displays a Properties dialog for configuring the resource.

Ping	Pings the resource to see whether the resource is responding. Theadministrator can use the ping action as a basic check of the "health"of the resource. For information about the ping operation, consult the documentationfor the operating system in use.
Properties	Displays a properties dialog for configuring the resource.
Reconnect	Connects a node to the administrative server.
Regen WebServer Plugin	Regenerates the configuration for a WebSphere plugin for Web servers. This option will overwrite the existing plugin configuration so the information in "6.6.45: Administering WebSphere plug-ins for Web servers" should be reviewed before using this option.
Remove	Permanently removes the selection and its children from the administrative domain.
Restart	Stops a resource, then returns the resource to the state it was in before it (or its parent resource) was stopped. For example, suppose you performed **Stop for Restart** (or just **Restart**) on a running administrativeserver node containing application server "A" in the stopped state and applicationserver "B" in the running state. The **Restart** command would cause application server A to remain stopped. It would return application server B to the running state, along with the administrative server.
Show Status	Displays a Module Status window that provides information about the modules, the servers that the modules are installed on and their corresponding status values.
Start	Changes the resource to running state.
Stop	Stops the resource. For an administrative server node, this action stops the administrative serverrunning on the node. It *does not* shut down the operating system.
Stop for Restart	Stops a resource and any resources contained by the resource, in preparation for the **Restart** command. It can be useful for stopping and restarting an administrative server that contains multiple application servers, some of which are running and others of which are stopped. Using a combination of **Stop for Restart** and then **Restart** will return the application servers to their former states (stopped or running) when the administrative server is restarted.
Transactions	Displays the Transactions dialog for viewing and controllingtransactions occurring on the resource. See the related informationto learn more about administering transactions.
Trace	Displays the Trace Administration dialog. See the related informationto learn more about tracing.
View Deployment Descriptor	Displays the deployment descriptor of the selected enterprise application, EJB module or Web Module.

6.6.0.2: Command line administration

The following command line administrative tools are available for interacting with theWebSphere administrative server.

- Use **wscp** for operational and configuration tasks such as starting, stopping, and configuring application servers and other object types.
- Use **XMLConfig** for configuration tasks. This tool provides a way to work with the administrative domain as represented in WebSphere XML.

6.6.0.2.1: XMLConfig command line interface for XML configuration

Use the *XMLConfig* tool to import and export configuration data to and from the WebSphere Application Server administration repository. This XML-based approach complements the administration you can perform through the administrative console.

You can use this tool to perform multiple changes to the WebSphere Application Server administration repository at a single time without having to go through the repetitive routines on the administration console. You may also use this tool to extract the repository information from one server and import it onto a cloned server.

The XMLConfig tool provides three fundamental features:

- **full export**

  Generates an XML document describing the configuration of the entire administrative domain. In effect, the full export takes a "snapshot" of the administrative repository contents. Click here for an example of a full export, including a discussion of the various parts of the resulting XML file.
- **partial export**

  Provides an XML document specifying administrative objects to export from the administrative domain. The objects and their children are exported to an XML document that you specify. Click here for an example of a partial export.
- **import**

  Imports a newly created XML document or a document you previously exported and modified. In the XML document, you can add, modify, or remove administrative objects such as servlets and virtual hosts. Your XML document can replace the administrative repository contents partially or entirely.

6.6.0.2.1.1: XMLConfig - Command syntax

This section describes the command line syntax for the XMLConfig tool.

Because setting the class path appropriately is vital to the tool's success, the product contains an XMLConfig.bat (Windows NT) or XMLConfig.sh (*IX) file for starting the tool. The file is located in the bin directory of the product installation root, uses the com.ibm.websphere.xmlconfig.XMLConfig class, and has the following command-line syntax:

```
{ ( -import xml_data_file ) ||     [ ( -export xml_output_file [-partial xml_data_file] )
-adminNodeName primary_node_name       [-nameServiceHost host_name [ -nameServicePort port_number ]]
[-traceString trace_spec [-traceFile file_name]]       [-generatePluginCfg true | false]
[-substitute "key1=value1[;key2=value2;[...]]"]}
```

Supported arguments include:

-adminNodeName

Required argument that specifies the node containing the administrative server to which you are connecting. The value of this argument must match the node name given in the topology tree on the Topology tab of the WebSphere Administrative Console.

-import || -export || -export -partial

Required argument that specifies the operation to perform: an import or export. Unless you also specify the parameter -partial, the export is treated as a full export.

-nameServiceHost, -nameServicePort

Optional arguments that specify the host name of the machine that contains the naming service, and the port through which to communicate with the naming service. The default value of -nameServicePort is 900.

-traceString

Optional argument that specifies the internal code to trace. For more information, see the traceString section of the trace help.

-generatePluginCfg

Generate the plug-in configuration if necessary.

-substitute

Optional argument that specifies the variables to be substituted (for example, `-substitute "NODE_NAME=admin_node;APP_SERVER=default_server"`).

In the input XML file, each key should appear as key for substitution. This argument substitutes any occurrence of `$NODE_NAME$` with admin_node and `$APP_SERVER$` with default_server in the input XML file.

If the substitution string contains semicolons,use `$semiColon$` to separate it from the ";" delimiter.

The following examples demonstrate correct syntax. Node1 is the name by which the node that contains the administrative server is administered.

Import operation:

```
XMLConfig -adminNodeName Node1 -import import.xml
```

Full export operation:

```
XMLConfig -adminNodeName Node1 -export export.xml
```

Partial export operation:

```
XMLConfig -adminNodeName Node1 -export export.xml -partial imput.xml
```

6.6.0.2.1.1.1: XMLConfig - Example of a full export

The following example shows the XML elements for each object type in the WebSphere administrative domain. It is a full export of the administrative repository, featuring the default administrative configuration.

To produce a similar export, you would issue the command:

```
XMLConfig -adminNodeName host_name -export export.xml
```

where host_name is the host name of the machine that contains the administrative serverand export.xml is the name of the output file.

When you perform an export, the XML output file does not contain blank lines or gaps. In contrast, the following export has been broken into segments, each of which is briefly discussed.

Also, this example was obtained from a system that used the default configuration, and might vary slightly from actual output due to changes on your particular system. It is recommended that you try the export command yourself to see exactly what output is produced.

```
<?xml version="1.0" encoding="UTF-8"?><!DOCTYPE websphere-sa-config   SYSTEM
"file:///$XMLConfigDTDLocation$$dsep$xmlconfig.dtd"><websphere-sa-config>
```

These tags mark the beginning of the export. The following part of the export contains a tag for the default virtual host, *default_host*, in the administrative domain.

The default host recognizes several MIME types, which are listed as part of a MIME table in the virtual-host tag. In fact, there are so many MIME types that many are omitted from the example. These are followed by the aliases for the virtual host.

Skip ahead

```
<virtual-host action="update" name="default_host">    <mime-table>        <mime
type="application/vnd.lotus-wordpro">            <ext>lwp</ext>            <ext>sam</ext>
</mime>        <mime type="text/tab-separated-values">            <ext>tsv</ext>        </mime>
<mime type="application/x-troff-me">            <ext>me</ext>        </mime>        <mime
type="image/x-portable-anymap">            <ext>pnm</ext>        </mime>        <mime
type="text/x-ssi-html">            <ext>htmls</ext>            <ext>shtml</ext>        </mime>
<mime type="application/vnd.lotus-screencam">            <ext>scm</ext>        </mime>        <mime
type="application/xml">            <ext>xsl</ext>        </mime>        ...    </mime-table>
<alias-list>        <alias>*:80</alias>        <alias>*:9080</alias>    </alias-list></virtual-host>
```

Next is the Java Messaging Service (JMS) configuration:

Skip ahead

```
<jms-provider action="update" name="IBM MQSeries">    <description>(OPTIONAL) Description of JMS
provider.</description>    <external-initial-context-factory>
com.ibm.websphere.naming.WsnInitialContextFactory    </external-initial-context-factory>
<external-provider-url>iiop://localhost</external-provider-url>    <jndi-binding-mechanism/>
<jms-connection-factory action="update" name="Test JMSConnFactory">        <description>Test
factory</description>        <jndi-name>jms/Test JMSConnFactory</jndi-name>
<external-jndi-name>TestJMSFactory</external-jndi-name>    </jms-connection-factory>
<jms-destination action="update" name="JMSTestDest">        <description>JMS destination
</description>        <external-jndi-name>JMSDest</external-jndi-name>
<jndi-name>jms/JMSTestDest</jndi-name>    </jms-destination></jms-provider>
```

The following configuration describesa resource adaptor for CICS:

Skip ahead

```
<j2c-resource-adapter action="update" name="TestJ2cAdapter">
<description>TestJ2cAdapter</description>
<archive-file>D:\WebSphere\AppServer\bin\cicseci.rar</archive-file>    <j2c-adapter-install-info>
<node-name>subodh</node-name>        <resource-archive-file>
D:\WebSphere\AppServer\installedConnectors\cicseci.rar        </resource-archive-file>
</j2c-adapter-install-info>    <j2c-connection-factory action="update" name="TestFactory">
<jndi-name>TestJ2cFactory</jndi-name>        <description>Test factory</description>
<connection-timeout>0</connection-timeout>        <maximum-connections>0</maximum-connections>
<minimum-connections>0</minimum-connections>        <reap-time>0</reap-time>
<unused-timeout>0</unused-timeout>        <pool-name>POOLNAME</pool-name>
<subpool-name>SUBPOOLNAME</subpool-name>        <config-property name="TraceLevel" value="1">
<description>(OPTIONAL)The level of trace to be output                to the Server Trace Log. Range
0-3. 0=off,                1=exceptions, 2=1+entry/exit, 3=2+debug</description>
<type>java.lang.Integer</type>        </config-property>        <config-property
name="ServerSecurity" value="">            <description>(OPTIONAL)Fully Qualified Class
implementing ServerSecurity for connections to the                Gateway (use on conjunction with
ClientSecurity            </description>            <type>java.lang.String</type>
</config-property>        <config-property name="PortNumber" value="2006">
<description>The port number the gateway is listening on</description>
<type>java.lang.String</type>        </config-property>        <config-property
```

```
name="KeyRingPassword" value="">            <description>The Password for the KeyRing
Class</description>            <type>java.lang.String</type>        </config-property>
<config-property name="ConnectionURL" value="">            <description>The URL of the CICS
Transaction Gateway</description>            <type>java.lang.String</type>        </config-property>
<config-property name="ClientSecurity" value="">            <description>(OPTIONAL)Fully Qualified
Class                implementing ClientSecurity for connections to the                Gateway (use
on conjunction with ServerSecurity</description>            <type>java.lang.String</type>
</config-property>        <config-property name="TranName" value="">            <description>The
Transaction name for programs to run under.            </description>
<type>java.lang.String</type>        </config-property>        <config-property name="TPNName"
value="">            <description>The TPN id for programs to run under. This                takes
precedence over TranName.</description>            <type>java.lang.String</type>
</config-property>        <config-property name="Password" value="">            <description>A
Password for the UserName</description>            <type>java.lang.String</type>
</config-property>        <config-property name="ServerName" value="">            <description>The
CICS Server as defined in the CICS                Transaction Gateway</description>
<type>java.lang.String</type>        </config-property>        <config-property name="UserName"
value="">            <description>A user Name to access CICS Resources</description>
<type>java.lang.String</type>        </config-property>        <config-property name="KeyRingClass"
value="">            <description>Fully Qualified Class containing the SSL                Keyrings.
Required only for SSL protocol</description>            <type>java.lang.String</type>
</config-property>    </j2c-connection-factory></j2c-resource-adapter>
```

Next is the configuration for the default URL provider:

Skip ahead

```
<url-provider action="update" name="Default URL Provider">    <description>This is the URL Provider
for the protocols        supported by the JDK (e.g., http, file, ftp, etc.).  The        protocol
and stream handler class name properties are not        used for this provider.</description>
<protocol>unused</protocol>    <stream-handler-class-name>unused</stream-handler-class-name>    <url
action="create" name="TestURL">        <description>test ur</description>
<specification>testurl.com</specification>        <jndi-name>testurl</jndi-name>    </url>
<install-info>        <node-name>subodh</node-name>
<jar-file-location>unused</jar-file-location>    </install-info></url-provider>
```

The following configuration is for JavaMail:

Skip ahead

```
<mail-session action="update" name="TestMailSession">    <description>Test mail sesson</description>
<jndi-name>TestMail</jndi-name>    <mail-transport-protocol>smtp</mail-transport-protocol>
<mail-transport-host>testServer</mail-transport-host>
<mail-transport-user>testuser</mail-transport-user>
<mail-transport-password>test</mail-transport-password>    <mail-from>testoriginator</mail-from>
<mail-store-protocol>imap</mail-store-protocol>    <mail-store-host>testhost</mail-store-host>
<mail-store-user>testuser</mail-store-user>
<mail-store-password>test</mail-store-password></mail-session>
```

The following section contains information aboutthe JDBC database driver and datasources. The configuration contains multiple data-source elements,only one of which is shown:

Skip ahead

```
<jdbc-driver action="update" name="Sample DB Driver">    <implementation-class>
COM.ibm.db2.jdbc.DB2ConnectionPoolDataSource    </implementation-class>    <description/>
<data-source action="update" name="SampleDataSource">        <database-name>was</database-name>
<minimum-pool-size>1</minimum-pool-size>        <maximum-pool-size>10</maximum-pool-size>
<connection-timeout>180</connection-timeout>        <idle-timeout>1800</idle-timeout>
<orphan-timeout>1800</orphan-timeout>        <statement-cache-size>100</statement-cache-size>
<default-user>subodhv</default-user>        <default-password>{xor}MzIsZg==</default-password>
<disable-auto-connectioncleanup>false</disable-auto-connectioncleanup>        <description>Sample
Data Source</description>        <jndi-name>jdbc/SampleDataSource</jndi-name>
<config-properties>            <property name="serverName" value=""/>            <property
name="portNumber" value=""/>            <property name="URL" value=""/>        </config-properties>
</data-source>    ...    <install-info>        <node-name>subodh</node-name>
<jdbc-zipfile-location>            K:/PROGRA~1/SQLLIB/java/db2java.zip
</jdbc-zipfile-location>    </install-info></jdbc-driver>
```

The following section contains the node, or physical machine, in the administrative domain. Its host name is subodh:

```
<node action="update" name="subodh">
```

Next is an application server. In this case, it is the default application server, Default Server:

Skip ahead

```
    <application-server action="update" name="Default Server">        <executable>java</executable>
```

```
<command-line-arguments/>        <environment/>        <user-id/>        <group-id/>
<working-directory/>        <umask>18</umask>        <stdin/>
<stdout>D:\WebSphere\AppServer/logs/Default_Server_stdout.log</stdout>
<stderr>D:\WebSphere\AppServer/logs/Default_Server_stderr.log</stderr>
<process-priority>20</process-priority>
<maximum-startup-attempts>2</maximum-startup-attempts>        <ping-interval>60</ping-interval>
<ping-timeout>200</ping-timeout>        <ping-initial-timeout>300</ping-initial-timeout>
<selection-policy>roundrobinpreferlocal</selection-policy>        <trace-specification/>
<trace-output/>        <transaction-log-file/>        <olt-enabled>false</olt-enabled>
<system-properties/>        <debug-enabled>false</debug-enabled>
<transaction-timeout>120</transaction-timeout>
<transaction-inactivity-timeout>60000</transaction-inactivity-timeout>
<thread-pool-size>20</thread-pool-size>        <security-enabled>false</security-enabled>
<admin-agent-ior/>        <cache-config>            <cache-size>2047</cache-size>
<cache-sweep-interval>1000</cache-sweep-interval>            <passivation-directory/>
</cache-config>        <log-file-spec/>        <performance-monitor-spec>
pmi=none:beanModule=maximum:connectionPoolModule=medium:
j2cModule=medium:jvmRuntimeModule=none:jvmpiModule=low:
servletSessionsModule=none:threadPoolModule=high:
transactionModule=none:webAppModule=maximum        </performance-monitor-spec>
<olt-server-host>localhost</olt-server-host>        <olt-server-port>2102</olt-server-port>
<selection-policy>roundrobinpreferlocal</selection-policy>        <source-path/>
<wlm-template-ior/>        <thread-pool-size>20</thread-pool-size>        <jvm-config>
<initial-heap-size>64</initial-heap-size>            <max-heap-size>64</max-heap-size>
<generated-command-line-arguments>                -Xms64m -Xmx64m
</generated-command-line-arguments>            <system-properties/>
<additional-command-line-arguments/>            <debug-mode>false</debug-mode>
<debug-string/>            <run-hprof>false</run-hprof>            <hprof-args/>
<disable-jit>false</disable-jit>            <verbose-class-loading>false</verbose-class-loading>
<verbose-jni>false</verbose-jni>            <verbose-gc>false</verbose-gc>
<boot-classpath-replace/>            <boot-classpath-append/>            <boot-classpath-prepend/>
</jvm-config>
```

Within the application-server configurationare the attributes of the Web container, as follows:

Skip ahead

```
        <web-container>            <dynamic-cache-config>                <enabled>false</enabled>
<cache-size>0</cache-size>                <default-priority>0</default-priority>
</dynamic-cache-config>            <transport name="http">
<transport-host>*</transport-host>                <transport-port>9080</transport-port>
<http-transport>                    <connection-timeout>5</connection-timeout>
<backlog-connections>50</backlog-connections>
<keep-alive-timeout>5</keep-alive-timeout>
<maximum-keep-alive>25</maximum-keep-alive>
<maximum-req-keep-alive>100</maximum-req-keep-alive>
<ssl-enabled>false</ssl-enabled>                </http-transport>            </transport>
<thread-maximum-size>50</thread-maximum-size>
<thread-minimum-size>25</thread-minimum-size>
<thread-inactivity-timeout>10</thread-inactivity-timeout>
<thread-is-growable>true</thread-is-growable>
```

A session manager configuration follows.The tag for the Web container is still open.

Skip ahead

```
            <session-manager>
<enable-security-integration>false</enable-security-integration>
<enable-ssl-tracking>false</enable-ssl-tracking>
<invalidation-schedule-first-hour>0</invalidation-schedule-first-hour>
<invalidation-schedule-second-hour>0                </invalidation-schedule-second-hour>
<persistent-db2-row-size>0</persistent-db2-row-size>
<maximum-inmemory-session-count>1000</maximum-inmemory-session-count>
<write-contents>0</write-contents>                <write-frequency>0</write-frequency>
<write-interval>0</write-interval>                <enable-sessions>false</enable-sessions>
<enable-url-rewriting>false</enable-url-rewriting>
<enable-cookies>true</enable-cookies>                <enable-protocol-switch-rewriting>false
</enable-protocol-switch-rewriting>                <cookie name="JSESSIONID">
<domain/>                    <maximum>-1</maximum>                    <path>/</path>
<secure>false</secure>                </cookie>
<tuning-invalidation-time>30</tuning-invalidation-time>
<persistent-sessions>false</persistent-sessions>                <data-source name="">
<default-user/>                    <default-password/>                </data-source>
<using-multi-row>false</using-multi-row>                <allow-overflow>true</allow-overflow>
</session-manager>
```

Configuration for the Web container ends:

```
        </web-container>
```

Next is an ORB configuration:

Skip ahead

```
        <orb-config>            <bootstrap-host-name/>
<bootstrap-port>900</bootstrap-port>            <comm-trace-enabled>false</comm-trace-enabled>
<connection-cache-maximum>240</connection-cache-maximum>
<connection-cache-minimum>100</connection-cache-minimum>            <external-config-url/>
<force-tunnel>whenrequired</force-tunnel>            <listener-port>0</listener-port>
<locate-request-timeout>180</locate-request-timeout>            <local-host-name/>
<lsd-host-name/>            <no-local-copies>false</no-local-copies>
<request-retries-count>1</request-retries-count>
<request-retries-delay>0</request-retries-delay>            <request-timeout>180</request-timeout>
<thread-pool-size>20</thread-pool-size>            <tunnel-agent-url/>
<rmi-remote-code-base/>            <ssl-listener-port>0</ssl-listener-port>        </orb-config>
```

The WebSphere plug-in configuration comes next:

Skip ahead

```
        <custom-service-config-list>            <custom-service-config
name="Automatic Generation of Plugin Configuration">                <description>If enabled, the
plugin configuration                    files will be regenerated when the application
server is started</description>                <classname>
com.ibm.websphere.plugincfg.initializers.AEPluginCfgService                </classname>
<classpath/>                <external-config-url/>                <enabled>false</enabled>
</custom-service-config>        </custom-service-config-list>
```

Configuration for the application server ends:

```
    </application-server>
```

Then the end tag for the node is placed, indicating that all the elements pertaining to the node have been addressed:

```
</node>
```

The security configuration comes next:

Skip ahead

```
<security-config security-cache-timeout="600" security-enabled="false">    <app-security-defaults>
<realm-name>Default</realm-name>        <challenge-type ssl-enabled="false">
<basic-challenge/>        </challenge-type>    </app-security-defaults>    <auth-mechanism>
<localos>            <user-id/>            <password>{xor}</password>        </localos>
</auth-mechanism></security-config>
```

Then comes the server group configuration:

Skip ahead

```
<server-group action="update" name="Test Server group">    <server-group-attributes>
<environment/>        <user-id/>        <group-id/>        <working-directory/>
<umask>18</umask>        <stdin/>        <stdout>stdout.txt</stdout>
<stderr>stderr.txt</stderr>        <process-priority>20</process-priority>
<maximum-startup-attempts>2</maximum-startup-attempts>        <ping-interval>60</ping-interval>
<ping-timeout>200</ping-timeout>        <ping-initial-timeout>300</ping-initial-timeout>
<selection-policy>roundrobinpreferlocal</selection-policy>
<debug-enabled>false</debug-enabled>        <transaction-timeout>120</transaction-timeout>
<transaction-inactivity-timeout>60000</transaction-inactivity-timeout>
<selection-policy>roundrobinpreferlocal</selection-policy>        <source-path/>        <jvm-config>
<initial-heap-size>64</initial-heap-size>            <max-heap-size>64</max-heap-size>
<generated-command-line-arguments>                -Xms64m -Xmx256m
</generated-command-line-arguments>            <system-properties/>
<additional-command-line-arguments/>            <debug-mode>false</debug-mode>
<debug-string/>            <run-hprof>false</run-hprof>            <hprof-args/>
<disable-jit>false</disable-jit>            <verbose-class-loading>false</verbose-class-loading>
<verbose-jni>false</verbose-jni>            <verbose-gc>false</verbose-gc>
<boot-classpath-replace/>            <boot-classpath-append/>            <boot-classpath-prepend/>
</jvm-config>        <custom-service-config-list/>    </server-group-attributes>    <clone
name="TestCLone">        <parent-node>subodh</parent-node>    </clone></server-group>
```

Several enterprise application entries come next;only one is shown here, with one EJB module and one Web module:

Skip ahead

```
<enterprise-application action="create" name="subodh/sampleApp">
<source-node>subodh</source-node>     <ear-file-name>
```

```
D:\WebSphere\AppServer\installableApps\sampleApp.ear     </ear-file-name>
<enterprise-app-install-info>         <node-name>subodh</node-name>         <ear-install-directory>
D:\WebSphere\AppServer/installedApps/sampleApp.ear         </ear-install-directory>
</enterprise-app-install-info>     <application-binding>         <authorization-table>
<role name="All Role">                <description>                    All Authenticated users in
the enterprise.                 </description>                 <all-authenticated-users
name="AllAuthenticatedUsers"/>             </role>             <role name="Everyone Role">
<description>Everyone in the enterprise.</description>                 <everyone name="Everyone"/>
</role>         </authorization-table>         <run-as-map/>     </application-binding>
<ejb-module name="Increment">         <jar-file>Increment.jar</jar-file>
<module-install-info>             <application-server-full-name>
/NodeHome:subodh/EJBServerHome:Default Server/             </application-server-full-name>
</module-install-info>         <ejb-module-binding>             <data-source>
<jndi-name>jdbc/SampleDataSource</jndi-name>                 <default-user/>
<default-password/>             </data-source>             <enterprise-bean-binding
name="EnterpriseBeanBinding_1">                <jndi-name>IncBean</jndi-name>
</enterprise-bean-binding>         </ejb-module-binding>     </ejb-module>     ...     <web-module
name="default_app">         <war-file>default_app.war</war-file>
<context-root>/</context-root>         <module-install-info>
<application-server-full-name>                /NodeHome:subodh/EJBServerHome:Default Server/
</application-server-full-name>         </module-install-info>         <web-module-binding>
<virtual-host-name>default_host</virtual-host-name>         </web-module-binding>     </web-module>
... </enterprise-application>
```

The last item is the end tag for the export itself:

```
</websphere-sa-config>
```

6.6.0.2.1.1.2: XMLConfig - Example of a partial export

To do a partial export of your Websphere Administrative Domain configuration into an XML file, you need to create an XML file specifying the resources you would like to export. This partial file is then used as an input parameter in the XMLConfig export command line.

The partial XML file always begins with the following header lines:

```
<?xml version="1.0" encoding="UTF-8"?><!DOCTYPE websphere-sa-config   SYSTEM
"file:///$XMLConfigDTDLocation$$dsep$xmlconfig.dtd">
```

The contents of the Websphere administrative domain that you wish to extract into an XML file start with <websphere-sa-config> and end with </websphere-sa-config>. What goes in between these tags depends on what you want to export. This is an example of a partial XML file you would create to export the entire contents of an application server on a node named mynode:

```
<?xml version="1.0" encoding="UTF-8"?><!DOCTYPE websphere-sa-config   SYSTEM
"file:///$XMLConfigDTDLocation$$dsep$xmlconfig.dtd"><websphere-sa-config>    <node name="mynode"
action="locate">        <application-server name="Default Server" action="export">
</application-server>    </node></websphere-sa-config>
```

As an example of how you would do a partial configuration export into an XML file, we can name this file *PartialFile.xml*, place it in the appserver/bin directory, and run the following command from that directory:

```
XMLConfig -export NewExport.xml -adminNodeName mynode -partial PartialFile.xml
```

An XML file named *NewExport.xml* is then created in the appserver/bin directory with the following output:

```
<?xml version="1.0" encoding="UTF-8"?><!DOCTYPE websphere-sa-config   SYSTEM
"file:///$XMLConfigDTDLocation$$dsep$xmlconfig.dtd"><websphere-sa-config>    <node name="mynode"
action="locate">        <application-server action="update" name="Default Server">
<executable>java</executable>            <command-line-arguments/>            <environment/>
<user-id/>            <group-id/>            <working-directory/>            <umask>18</umask>
<stdin/>            <stdout>D:\WebSphere\AppServer/logs/Default_Server_stdout.log</stdout>
<stderr>D:\WebSphere\AppServer/logs/Default_Server_stderr.log</stderr>
<process-priority>20</process-priority>
<maximum-startup-attempts>2</maximum-startup-attempts>            <ping-interval>60</ping-interval>
<ping-timeout>200</ping-timeout>            <ping-initial-timeout>300</ping-initial-timeout>
<selection-policy>roundrobinpreferlocal</selection-policy>            <trace-specification/>
<trace-output/>            <transaction-log-file/>            <olt-enabled>false</olt-enabled>
<system-properties/>            <debug-enabled>false</debug-enabled>
<transaction-timeout>120</transaction-timeout>
<transaction-inactivity-timeout>60000</transaction-inactivity-timeout>
<thread-pool-size>20</thread-pool-size>            <security-enabled>false</security-enabled>
<admin-agent-ior/>            <cache-config>                <cache-size>2047</cache-size>
<cache-sweep-interval>1000</cache-sweep-interval>                <passivation-directory/>
</cache-config>            <log-file-spec/>            <performance-monitor-spec>
pmi=none:beanModule=maximum:connectionPoolModule=medium:
j2cModule=medium:jvmRuntimeModule=none:jvmpiModule=low:
servletSessionsModule=none:threadPoolModule=high:
transactionModule=none:webAppModule=maximum            </performance-monitor-spec>
<olt-server-host>localhost</olt-server-host>            <olt-server-port>2102</olt-server-port>
<selection-policy>roundrobinpreferlocal</selection-policy>            <source-path/>
<wlm-template-ior/>            <thread-pool-size>20</thread-pool-size>            <jvm-config>
<initial-heap-size>64</initial-heap-size>                <max-heap-size>64</max-heap-size>
<generated-command-line-arguments>-Xms64m -Xmx64m </generated-command-line-arguments>
<system-properties/>                <additional-command-line-arguments/>
<debug-mode>false</debug-mode>                <debug-string/>
<run-hprof>false</run-hprof>                <hprof-args/>
<disable-jit>false</disable-jit>                <verbose-class-loading>false</verbose-class-loading>
<verbose-jni>false</verbose-jni>                <verbose-gc>false</verbose-gc>
<boot-classpath-replace/>                <boot-classpath-append/>
<boot-classpath-prepend/>            </jvm-config>            <web-container>
<dynamic-cache-config>                    <enabled>false</enabled>
<cache-size>0</cache-size>                    <default-priority>0</default-priority>
</dynamic-cache-config>                <transport name="http">
<transport-host>*</transport-host>                    <transport-port>9080</transport-port>
<http-transport>                        <connection-timeout>5</connection-timeout>
<backlog-connections>50</backlog-connections>
<keep-alive-timeout>5</keep-alive-timeout>
<maximum-keep-alive>25</maximum-keep-alive>
<maximum-req-keep-alive>100</maximum-req-keep-alive>
<ssl-enabled>false</ssl-enabled>                    </http-transport>                </transport>
<thread-maximum-size>50</thread-maximum-size>
<thread-minimum-size>25</thread-minimum-size>
<thread-inactivity-timeout>10</thread-inactivity-timeout>
<thread-is-growable>true</thread-is-growable>                <session-manager>
```

```
<enable-security-integration>false</enable-security-integration>
<enable-ssl-tracking>false</enable-ssl-tracking>
<invalidation-schedule-first-hour>0</invalidation-schedule-first-hour>
<invalidation-schedule-second-hour>0</invalidation-schedule-second-hour>
<persistent-db2-row-size>0</persistent-db2-row-size>
<maximum-inmemory-session-count>1000</maximum-inmemory-session-count>
<write-contents>0</write-contents>                    <write-frequency>0</write-frequency>
<write-interval>0</write-interval>                    <enable-sessions>false</enable-sessions>
<enable-url-rewriting>false</enable-url-rewriting>
<enable-cookies>true</enable-cookies>
<enable-protocol-switch-rewriting>false</enable-protocol-switch-rewriting>
<cookie name="JSESSIONID">                        <domain/>
<maximum>-1</maximum>                        <path>/</path>
<secure>false</secure>                    </cookie>
<tuning-invalidation-time>30</tuning-invalidation-time>
<persistent-sessions>false</persistent-sessions>                    <data-source name="">
<default-user/>                        <default-password/>                    </data-source>
<using-multi-row>false</using-multi-row>                    <allow-overflow>true</allow-overflow>
</session-manager>            </web-container>            <orb-config>
<bootstrap-host-name/>                <bootstrap-port>900</bootstrap-port>
<comm-trace-enabled>false</comm-trace-enabled>
<connection-cache-maximum>240</connection-cache-maximum>
<connection-cache-minimum>100</connection-cache-minimum>                <external-config-url/>
<force-tunnel>whenrequired</force-tunnel>                <listener-port>0</listener-port>
<locate-request-timeout>180</locate-request-timeout>                <local-host-name/>
<lsd-host-name/>                <no-local-copies>false</no-local-copies>
<request-retries-count>1</request-retries-count>
<request-retries-delay>0</request-retries-delay>
<request-timeout>180</request-timeout>                <thread-pool-size>20</thread-pool-size>
<tunnel-agent-url/>                <rmi-remote-code-base/>
<ssl-listener-port>0</ssl-listener-port>            </orb-config>
<custom-service-config-list>                <custom-service-config name="Automatic Generation of
Plugin Configuration">                    <description>If enabled, the plugin configuration
files will be regenerated when the application                        server is
started</description>
<classname>com.ibm.websphere.plugincfg.initializers.AEPluginCfgService</classname>
<classpath/>                    <external-config-url/>                    <enabled>false</enabled>
</custom-service-config>            </custom-service-config-list>        </application-server>
</node></websphere-sa-config>
```

6.6.0.2.1.2: XMLConfig grammar

This section discusses the general structure of an XML element. For detailed information about each object, see the package summary file.See also the full export example topic.

Each object tag in the XML document contains:

- An object type, such as `virtual-host`
- A name attribute that identifies the particular resource on which to perform the action
- An action attribute that controls the behavior of the import or partial export operation

For example, an application server named MyAppServer has the following object tag:

```
<application-server name="MyAppServer" action="update">
```

In this case, the action is update. The following list describes all of the available actions.Unless otherwise stated, actions apply only to the import operation.

create

Adds the specified resource to the administrative domain. If the resource already exists, the create action is treated as an update action.When using this action, you must specify all required attributes for the object type.

update

Updates the properties of a specified resource. You need only specify the properties that you want to update.

However, if the resource does not exist, the update action becomes a create. In such a case, if some of the properties are not specified, an error occurs.

delete

Removes the specified resource and its children (recursive delete).

locate

Locates the specified resource. Use this action to provide the containment path to specific child resources. Applies to the partial export operation as well as the import operation.

export

Exports the configuration of the specified resource and its children to the output XML document. Applies only to the partial export operation.

start

Starts the specified resource (if applicable).

stop

Stops the specified resource, (if applicable).

stopforrestart

Stops the specified resource, and restarts it.(if applicable). Node only.

restart

Stops the specified resource and starts it again (if applicable).

enable

Makes the specified resource available for user requests (currently applies only to servlets and Web applications).

disable

Makes the specified resource unavailable for user requests (currently applies only to servlets and Web applications).

ℹ Some operational actions, such as start and stop, do not apply to all object types. In general, if the operation is supported in the WebSphere Administrative Console, it is supported in the XML import operation.

6.6.0.2.1.3: XMLConfig - Using the tool programmatically

The XMLConfig class is structured so that you can use it programmatically to retrieve information as Document or Element. The import/export facility can thus be included in a Java program, as well as being operated from a command line.

Creating platform-neutral configurations

For import and partial export operations, a variable substitution operation is performed on the input XML document, allowing you to create platform-neutral XML documents. The following variables are available:

$server_root$

Replace with the product installation directory, such as `C:\WebSphere\AppServer` on Windows NT. (This variable is not available for use on iSeries.)

$psep$

Replace with the path separator as specified in the operating system JDK.

- On Windows NT, it is **;** (semicolon)
- On the UNIX platforms and Linux, it is **:** (colon)

$dsep$

Replace with the directory separator as specified in the operating system JDK.

- On Windows NT, it is \ (backward slash)
- On the UNIX platforms and Linux, it is / (forward slash)

Security XML configurations

In Version 4.0, you can enter your own*custom user registry* entries (key/value pairs).All user-defined keysmust be delimited with a special prefix, `Custom_`.When you use the product GUI to add custom entries,the product adds the prefix for you.However, if you want to configure custom entries programmatically,you must add the prefix yourself.

The following example sets these properties:

```
groupsFile = c:\temp\groups.props    db_URL = jdbc:db2:customDB
```

A complete stanza for security configuration follows.The markup that corresponds to the needed property settings is shown in bold.

```
<security-config security-cache-timeout="150" security-enabled="true">   <app-security-defaults>
<realm-name>Default</realm-name>      <challenge-type ssl-enabled="false">
<basic-challenge/>      </challenge-type>   </app-security-defaults>   <auth-mechanism>
<ltpa-config>         <ltpa-password>{xor}B7rj9Lrj77rj7Q==</ltpa-password>
<ltpa-timeout>1200000</ltpa-timeout>         <custom-ur-config>           <user-id>bob</user-id>
<password>{xor}PTA9bg==</password>           <attribute name="Custom_groupsFile"
value="c:/temp/groups.props"/>           <attribute name="Custom_db_URL"
value="jdbc:db2:customDB"/>         </custom-ur-config>      </ltpa-config>
</auth-mechanism></security-config>
```

Javadoc for the tool

It is recommended that you refer to the Javadoc for the latest programmatic use of XMLConfig, and refer to the exported XML for the sample markup for repository objects.

Javadoc for the com.ibm.websphere.xmlconfig class and all of the related object classes resides in the apidocs directory:

```
installation_root\web\apidocs\package_and_class_name
```

See the package summary file for a list of class names, such as ApplicationServerConfig. The Javadoc is labeled by the class name preceded by the package name, com.ibm.websphere.

6.6.0.2.1.5: Troubleshooting XMLConfig

This article describes what to do if XMLConfig fails and displays the following message at its command line:

```
MLC0137E Make sure that adminserver is up and running.Additionally check -nameServiceHost and
-nameServicePort pair, if using remote admin server.
```

One or more of the following conditions is likely to have caused the problem:

- The admin server did not start properly.
- XMLConfig is being run remotely from the machine on which the product administrative serveris installed, and the -nameServiceHost parameter has not been set to the host name of the machine that contains the administrative server.
- The naming service port was changed from the default value of 900 on the application server and the -nameServicePort parameter was not set to the changed port value.
- The -nameServicePort parameter was defined, but the -nameServiceHost parameter was not.
- If the fully qualified host name (host name and domain) is specified for the -adminNodeName parameter, instead of the node name (short host name), expect this error:

  ```
  001.553 22f45b XMLConfig     X Unabled to export Virtual Host Data: {0}
  javax.naming.NameNotFoundException:
  ```

To resolve the problem:

1. Check whether the administrative server started successfully, as describedin the documentation for starting the administrative server.
2. If the XMLConfig tool is being run remotely with respect to the machine on whichthe administrative server is running, ensure that the -nameServiceHost and -adminNodeName parameters are set to the host name of the remote machine.
3. If the naming service port has been changed from the default value on the machineon which the administrative server is running, look in the administrative server configuration file for the parameter:

   ```
   com.ibm.ejs.sm.adminServer.bootstrapPort
   ```

   Set the XMLConfig -nameServicePort parameter to this port value.
4. If you use the -nameServicePort parameter, you must also use the -nameServiceHost parameter, even if you are running XMLConfig on the same machine asthe administrative server.
5. Modify the -adminNodeName parameter to use the node name, which can be found as your host name in your TCP/IP networking configuration and also in the WebSphere Administrative Console under **WebSphere Administrative Domain**.

6.6.0.2.2: WebSphere Control Program (wscp)

The WebSphere Control Program (wscp) is a command line client for the administrativeserver. It can be used to administer application servers, enterprise applications, andother types of WebSphere Application Server objects.

- WebSphere Control Program (wscp) overview introduces **wscp**, describes the relationship between the administrative console and **wscp**, and discusses the benefits and drawbacks of using **wscp**. It also lists the object types and services that **wscp** supports.
- Command syntax and usage covers basic and detailed syntax for the **wscp** interface, describes how to use **wscp** in command line and interactive modes and run Tcl scripts, and gives examples of how to use the **wscp** online help facility.
- Advanced usage of wscp goes into depth on how to use various **wscp** features, including abbreviations of **wscp** commands, lists, Tcl commands, operating system commands, error information, and tracing. It also describes how to use **wscp** to manipulate the Java Naming and Directory Interface (JNDI) context of objects, monitor performance, set security defaults, administer role-based security, and connect to remote servers.
- Example wscp commands, procedures and scripts contains examples of how to use various **wscp** commands, Tcl procedures, and Tcl scripts. Note that not all **wscp** operations are described in this section.
- Sample Tcl procedures and scripts lists the example Tcl scripts that are used in the **wscp** documentation.
- Migrating wscp scripts from version 3.5.x to version 4.0 summarizes the **wscp** changes that are most likely to affect existing **wscp** scripts.
- Using the WscpCommand interface describes how to use the WscpCommand class, which enables you to execute **wscp** operations from within Java applications.

6.6.0.2.2.1: WebSphere Control Program (wscp) overview

The WebSphere Control Program (**wscp**) is a command-line andscripting interface for administering resources in WebSphere ApplicationServer Advanced Edition. It is based on Tcl (tool commandlanguage). Tcl is a portable command language that provides programmingfacilities, such as variables, procedures, conditionals, list-processingfunctions, and looping constructs. The **wscp** interfaceextends Tcl by providing a set of commands for manipulating WebSphereobjects.

This section contains the following topics:

- The Administrative Console and wscp
- Benefits of using wscp
- Limitations of using wscp
- Supported object types
- Supported services and commands

The Administrative Console and wscp

The administrative server manages the contents and activities of a domainby maintaining a repository. A repository is the database ofinformation about all resources in a domain. The repository allowsadministration of a domain from any machine--all information is stored ina central location. The repository contains descriptive informationabout the applications that are configured to run in the domain. Forexample, it contains the names of all application servers, nodes, servergroups, and J2EE resources (such as JDBC providers), and their current state(running, defined, or stopped).

All administration takes place through the manipulation of objects in therepository. Each resource in a domain corresponds to an object in therepository. For example, when you create an application server, acorresponding application server object is created in the repository.

Both the administrative console and the **wscp** interface can beused to administer the resources in a domain. They both modify therepository in response to user commands, and they both reflect any changes tothe configuration and status of the domain. Console users access andmodify the repository via a graphical user interface; **wscp**users manipulate objects in the repository by executing commands orscripts. The console and the **wscp** interface arecompatible. The results of actions performed with **wscp** arereflected in the console interface, and vice versa.

Both the console and **wscp** can be used to do the following:

- Define, configure, and manage application servers, cloned applicationservers, and other application server resources from any node in thenetwork.
- Install applications (by using wizards in the console and install commandsand scripts in **wscp**).
- Perform daily administrative operations, such as starting and stoppingapplication servers and making changes to their configuration.
- Replicate application servers to improve performance or availability or tosimplify administration tasks (by defining and managing server groups).
- Track the occurrence of specific events by setting and enablingtracing.

Benefits of using wscp

The **wscp** interface provides a high-level command-lineadministrative tool with programming capabilities. With**wscp**, you can do the following:

- Use Tcl (tool command language) to extend and automate administrativeoperations through the use of scripts. For example, you can write ascript to start and stop servers or applications as a group. You canalso use scripts to create a basic configuration or (by invoking the XMLConfiguration Management Tool, XMLConfig) to store an existingconfiguration. The stored configuration can be used as a backup or tore-create a configuration on another machine.
- Use Tcl to create custom procedures for specialized administrative tasks,such as making identical modifications to a particular attribute throughoutthe domain, or displaying the values of select attributes that need to befrequently monitored.
- Issue UNIX or Windows NT commands from within a **wscp** sessionand incorporate them into scripts.

Limitations of using wscp

Although **wscp** offers many of the same administration tasks asthe WebSphere Advanced Administrative Console, the following differencesapply:

- New configurations or changes to existing configurations made with**wscp** are not immediately reflected in the console. You mustactively poll the administrative server repository to see the changes byselecting all or a portion of the Topology view in the console and then usingthe **Refresh** button.

  Similarly, if a resource is created or deleted in the console, the changewill not be immediately reflected in **wscp**. You mustexplicitly refresh all object references in the repository cache by issuing a**wscp** list operation for the instance's object type.(Alternatively, exiting and reinvoking **wscp** also refreshes thecache.)

  To avoid problems due to inconsistent cached information, it is best toavoid issuing concurrent operations on the same object from within**wscp** and the console.

- All administrative tasks that can be done with the console can be donewith **wscp**, with the exception of higher-level aggregate tasks suchas those in the wizards--for example, creating an application.Instead, **wscp** scripts can be created to provide the samefunctionality as provided by the wizards.

Supported object types

In **wscp**, a resource is represented as an object type. Forexample, the object type ApplicationServer represents an application server(enterprise bean server) and the object type DataSource represents adatabase. Each object type has attributes (called *properties*in the console) that describe the characteristics of the object. Forexample, application server object attributes include Executable,CurrentState, and WorkingDirectory. Data source object attributesinclude DatabaseName, MinPoolSize, and MaxPoolSize. An object type canbe thought of as a template object that defines the characteristics of allobjects of that type. Instances of the object type represent specificobjects in the domain.

The **wscp** interface manipulates objects in the repository byperforming operations on them. Examples of operations are create,start, show, modify, and stop. The following is a list of object typessupported by **wscp**.

Note:

> Any object that represents a live entity outside of the repository (forexample, application servers), can be started and stopped with**wscp**. The attributes of several object types cannot bemodified. The message `EditorNotDefinedForThisProperty` isdisplayed when you attempt to access these attributes.

- ApplicationServer
- DataSource
- EnterpriseApp
- GenericServer
- J2CConnectionFactory
- J2CResourceAdapter
- JDBCDriver
- JMSConnectionFactory
- JMSDestination
- JMSProvider
- MailSession
- Module
- Node
- ServerGroup
- URL
- URLProvider
- VirtualHost

As the console does, **wscp** automatically provides defaultattributes and values for an object when the object is created. Youneed to specify values only for those attributes that are required and lackdefaults. If needed, you can override the default value for anyattribute when creating an object by specifying a value for thatattribute.

In the repository, objects are represented as attribute lists. Anattribute list is a collection of attribute-value pairs. The followingoutput shows how **wscp** displays the attributes of a DataSourceobject named ds1:

```
{Name ds1} {FullName /JDBCDriver:connDrv/DataSource:ds1/}{Description null} {ConfigProperties
{{URLjdbc:oracle:thin:@wssol2:1521:oraejs}}} {ConnTimeout 180}{DatabaseName WAS} {DefaultPassword
null} {DefaultUser null}{DisableAutoConnectionCleanup False} {IdleTimeout 1800}
{JNDINamejdbc/carddb} {MaxPoolSize 10} {MinPoolSize 1} {OrphanTimeout 1800}{StatementCacheSize 100}
```

See 6.6.0.2.2.3.2: Specifying lists in wscp commands for details on attribute lists.

In general, objects are created by specifying the object type, theoperation to be performed, the name of the object instance, and one or moreoptions. For example, to create a DataSource object, you specify theobject type (DataSource), the operation (create), the name of the objectinstance to create, and if attributes are required, the -attributeoption. The argument to the -attribute option is a list ofattributes and their values. For example, the following **wscp**command (in interactive mode) creates a DataSource object named ds1 that is aresource for the WAS database. The database driver must also bespecified:

```
wscp> DataSource create /JDBCDriver:DB2Driver/DataSource:ds1/-attribute {{DatabaseName WAS}}
```

In the following example, the ApplicationServer start command is used tostart an application server named myServer:

```
wscp> ApplicationServer start /Node:dev-pc/ApplicationServer:myServer/
```

See 6.6.0.2.2.2: Command syntax and usage for more example commands, detailed syntax, and anexplanation of the convention for object names.

The following example uses the Tcl **foreach** command to iteratethrough all application servers in a domain and stop them:

```
wscp> foreach ejbserver [ApplicationServer  list] {ApplicationServer stop $ejbserver}
```

See 6.6.0.2.2.3.3: Example use of wscp and Tcl and 6.6.0.2.2.4: Example wscp commands, procedures, and scripts.

Supported services and commands

In addition to objects, **wscp** supports several services andcommands that perform various administration tasks, such as configuringWebSphere Application Server, enabling global security, enabling tracing anddata collection, manipulating Java Naming and Directory Interface (JNDI)contexts, and displaying help on **wscp** commands. Theseservices are as follows:

- Context
- DrAdmin
- Help
- PmiService
- Remote
- SecurityConfig
- SecurityRoleAssignment
- XMLConfig

6.6.0.2.2.2: Command syntax and usage

This section contains basic and detailed syntax for the **wscp**interface, and describes how to use the **wscp** online helpfacility. The topics include:

- 6.6.0.2.2.2.1: Basic syntax
- 6.6.0.2.2.2.2: Invoking and terminating wscp
- 6.6.0.2.2.2.3: Authenticating to the administrative server
- 6.6.0.2.2.2.4: Connecting to local and remote nodes
- 6.6.0.2.2.2.5: Using command-line mode
- 6.6.0.2.2.2.6: Using interactive mode
- 6.6.0.2.2.2.7: Running scripts
- 6.6.0.2.2.2.8: Detailed syntax
- 6.6.0.2.2.2.9: Specifying object names
- 6.6.0.2.2.2.10: Using the wscp help facility

6.6.0.2.2.2.1: Basic syntax

The syntax for the **wscp** command is as follows:

```
wscp [ -h  ]  [ -c command ] [ -f  Tcl_file_name] [ -p  properties_file_name] [ -x  extension_class]
[ [ -- ] options ] [-node node_name]
```

The command options and arguments are as follows:

- The -h option displays help for the command.
- The -c option indicates command-line mode. The commandargument specifies a single command to be executed by **wscp**.This option can be repeated multiple times on the command line. See 6.6.0.2.2.2.8: Detailed syntax for more information on the commandargument.
- The -f option evaluates the specified file (script) of Tclcommands. Scripts can have arguments, which are specified following thedouble hyphen (- -). This option can be repeated multiple times on thecommand line. See 6.6.0.2.2.2.7: Running scripts.
- The -p option loads the specified properties file. This option canbe repeated multiple times on the command line.
- The -x option loads the specified Tcl extension class. This optioncan be repeated multiple times on the command line.
- The options following the double hyphen (- -) are used to setTcl argc and argv variables as specified. The double hyphen isnecessary only if an option can otherwise be mistaken for a **wscp**shell option.

For example, to invoke **wscp** and load theinit.tcl script, issue the following command:

```
wscp -f init.tcl
```

If no command-line options or files are specified, an interactive shell(Tcl interpreter) is invoked, which is terminated by the **exit**command. Command-line options not supported by **wscp**, orspecified after the double hyphen (- -) on the command line, are used to setthe Tcl argc or argv variables. These variables can be interpreted byTcl extensions or other commands.

The **wscp** shell evaluates Tcl commands in the order specified onthe command line, so any extensions must be loaded prior to invoking commandsdependent on those extensions. The following extensions areautomatically loaded:

- com.ibm.ejs.sm.ejscp.EjscpExtension,the class for the **wscp** commands
- com.ibm. ejs.sm.ejscp.ContextExtension,the class for manipulating Java Naming and Directory Interface (JNDI) contexts
- com.ibm.ejs.sm.ejscp.DrAdminExtension,the class for tracing the **wscp** client, WebSphere applicationservers, and the WebSphere administrative server
- com.ibm.ejs.sm.ejscp.RemoteExtension,the class for the **wscp** Remote extension.
- com.ibm.ejs.sm.ejscp.PmiServiceExtension,the class for monitoring application server performance.
- com.ibm.ejs.sm.ejscp.SecurityConfigExtension,the class for setting basic security defaults.
- com.ibm.ejs.sm.ejscp.SecurityRoleAssignmentExtension,the class for manipulating J2EE security roles.

After a command or script is executed, control is returned to theshell.

The **wscp** commands can be run as individual **wscp**invocations from the operating system prompt (command-line mode), as scripts,or interactively in a **wscp** session (interactive mode).

6.6.0.2.2.2.2: Invoking and terminating wscp

To invoke **wscp**, use one of the following procedures.

- To run **wscp** with its default extensions loaded, enter one ofthe following at the command prompt:
    - On Unix systems:

      ```
      > wscp.sh
      ```
    - On Window systems:

      ```
      C:\ wscp.bat
      ```
- To run **wscp** with different extensions loaded, do thefollowing:
    1. Set up the CLASSPATH environment variable, including the appropriate JARfiles.
    2. Run Java, invoking the following:
        - The **wscp** Tcl shell (thecom.ibm.ejs.sm.ejscp.WscpShell class)
        - Desired extensions (the -x option). Extensions can also be loadedinteractively.
    3. Optionally, set properties to enable tracing.

When **wscp** is invoked without specifying the -p option, theproperty file homeDirectory/.wscprc is loaded if it exists,where homeDirectory is the value of the Java propertyuser.home. The .wscprc file can be used to automaticallyload settings for various properties, amending or replacing system propertiesalready defined. For example, you can modify the values of thewscp.hostName and wscp.hostPort properties to connect to aremote node. See 6.6.0.2.2.2.4: Connecting to local and remote nodes for details.

Note that the value of the Java user.home property can differdepending on the SDK version level. On Windows NT systems, the SDK1.2.2 used by WebSphere Application Server sets theuser.home property based on the value of the USERPROFILE environmentvariable.

The following example .wscprc file specifies values for variousproperties.

```
## primaryNode is used with qualifyHomeNames (default=hostName property)## wscp.primaryNode=pc-dev1## hostName is used for
com.ibm.CORBA.BootstrapHost#wscp.hostName=pc-dev1## hostPort is used for com.ibm.CORBA.BootstrapPort (default=900)##
wscp.hostPort=900## wscp.remotePasswordFile (default : uses random number for password)# wscp.remoteConnectionTimeout (default =
300 seconds)# wscp.remoteConnectionsAllowed (default = false)# wscp.remoteHostListAccept (default = null, colon-separated
string)# wscp.remoteHostListReject (default = null, colon-separated
string)#wscp.remotePasswordFile=C:/TEMP/passwordwscp.remoteConnectionTimeout=300wscp.remoteConnectionsAllowed=truewscp.remoteHostListAccept=abc.def.com:abc.comwscp.remoteHostListReject=dhcp-198-7.abc.def.com##
set traceString to enable tracing## wscp.traceString=com.ibm.ejs.sm.client.ui.desc.*=all=enabled#
wscp.traceString=com.ibm.ejs.sm.ejscp.*=all=enabled### Typically, these need to be set if only ContextExtension is loaded#
(otherwise, EjscpExtension sets them as noted above)## com.ibm.CORBA.BootstrapHost=pc-dev1# com.ibm.CORBA.BootstrapPort=900#
java.naming.factory.initial=com.ibm.ejs.ns.jndi.CNInitialContextFactory
```

The **exit** command terminates an **wscp** interactivesession.

6.6.0.2.2.2.3: Authenticating to the administrative server

If security is enabled for your administrative server, you must prepare the**wscp** client so that it can authenticate to the server. To dothis, perform the following steps:

- Make a copy of the properties file sas.client. props.This file is located in the subdirectory named properties, in the directorywhere you installed the product--for example,C:\WebSphere\AppServer\properties. Rename the file--forexample, you can name the file sas.wscp.props--and edit oradd the following property-value pairs:

  ```
  com.ibm.CORBA.loginSource=propertiescom.ibm.CORBA.loginUserid=your_user_idcom.ibm.CORBA.principalName=NT_domain/user_idcom.ibm.CORBA.loginPassword=your_password
  ```

  The value `properties` is the only option for thecom.ibm.CORBA.loginSource property. The value`prompt` (prompting for a user ID) is not supported for**wscp**.
- If you are using digital certificates, you must also enableaccess to the server key stores. A dummy key store file is provided as aWebSphere Application Server installation option. The dummy key storeis not intended for use in a production environment. See 5.5: Certificate-based authentication, for moreinformation on using certificates in WebSphere Application Server.
- Add the following line to your .wscprc file:

  ```
  com.ibm.CORBA.ConfigURL=URL of properties file
  ```

  An example URL isfile:///C:/WebSphere/AppServer/properties/sas.wscp.props.Alternatively, you can add the following option to the Java command line(preceding the application class name):

  ```
  -Dcom.ibm.CORBA.ConfigURL=URL of properties file
  ```

6.6.0.2.2.2.4: Connecting to local and remote nodes

When you invoke **wscp**, you are automatically connected to theadministrative server running on the local machine. (The host name isdefined in the default .wscprc property file.) To connect to aremote node (that is, to specify the host name of a remote administrativeserver), set the property wscp.hostName to the name that a domain namesystem (DNS) server can resolve. For example, the following propertyspecifies that the administrative server running on myHost is to beused:

```
wscp.hostName=myHost
```

This property can be set in the default .wscprc file. You can alsouse the -p option of the **wscp** command to specify a differentproperty file to use. See 6.6.0.2.2.2.2: Invoking and terminating wscp for information on the .wscprc property file.

6.6.0.2.2.2.5: Using command-line mode

To execute an individual **wscp** command in command-line mode, usethe -c option followed by the command. The following examples execute a**wscp** command on the local node. On Windows NT systems,enclose the command in double quotation marks ("). On UNIX systems,enclose the command in single quotation marks (').

```
C:\> wscp -c "ApplicationServer list"
```

```
% wscp.sh -c 'ApplicationServer list' -c 'Node list'
```

The -c option can be repeated multiple times on the command line.

6.6.0.2.2.2.6: Using interactive mode

To invoke an interactive **wscp** session, type **wscp** andpress Return. The following prompt signals that you have enteredinteractive mode:

```
wscp>
```

At the interactive prompt, enter a **wscp** command;**wscp** executes the command, displays the result, and is ready toaccept another command. Use the **exit** command to terminate a**wscp** interactive session.

In an interactive **wscp** session, you can break the line of a**wscp** command after a left brace ({) or after typing a backslash (\ ). The **wscp** session displays a question mark (?) prompt,and you can continue typing the command.

Note:

> The WscpCommand class allows you to embed interactive **wscp**operations in a Java application. See 6.6.0.2.2.7: Using the wscpCommand interface for more information on using this class.

6.6.0.2.2.2.7: Running scripts

To execute a **wscp** script from the command line, specify the -foption and the name of the executable file that contains the script.For example, the following command executes the script in themodEnv.tcl file.

```
C:\> wscp -f  modEnv.tcl
```

To execute a script from within an interactive **wscp** session, usethe Tcl **source** command to source the script. The followingcommand runs the script in the file myScript.tcl:

```
wscp> source myScript.tcl
```

6.6.0.2.2.2.8: Detailed syntax

The detailed syntax for the **wscp** command argument is asfollows:

```
object_type operation [object_name] [ -option  [value] | -attribute  attribute_list]
```

A command argument can be issued immediately following the -coption of the **wscp** command, or at the **wscp** interactiveprompt, as shown in the following examples.

```
C:\> wscp -c  "ApplicationServer list"wscp> ApplicationServer list
```

When using the -c option, enclose the command in double quotation marks (")(Windows NT systems) or in single quotation marks (') (UNIXsystems).

Note:

All object type names, object instance names, and attributes are casesensitive.

The arguments are as follows:

- The object_type argument specifies the name of an objecttype. Examples are application servers (the object typeApplicationServer) and JDBC providers (the object type JDBCDriver).Object type names can be abbreviated to the shortest unique string (ininteractive mode only).
- The operation argument specifies the action to be performed onthe object. Examples of **wscp** operations are create, modify,show, and start.
- The object_name argument specifies the name of an object on whichthe action is to be performed. An object_name indicates aparticular instance of an object type--for example, ejbserver1.You must use fully qualified object names. See 6.6.0.2.2.2.9: Specifying object names for details on object naming.
- The option argument specifies a qualifier that controls theprecise behavior of an **wscp** operation. For example, usingthe -force option with a stop operation performs a risky operation orcompletes an operation without confirmation. Some options take values,which further define the operation's behavior. For example, the-constraint option of list operations takes an attribute list as itsvalue. The use of this option causes **wscp** to display onlythose object instances that contain the attribute-value pair specified.
- The -attribute option specifies one or more properties of anobject. Examples of attributes are the DatabaseName and IdleTimeoutattributes of a DataSource object. Properties are specified in a listof attribute-value pairs. For example, the following is an attributelist for a DataSource instance:

  ```
  {Name ds1} {FullName /JDBCDriver:j1/DataSource:ds1/}{ConfigProperties {}} {Conn Timeout 180}
  {DatabaseName WAS}{DefaultPassword {}} {DefaultUser {}} {Description
  {}}{DisableAutoConnectionCleanup False} {IdleTimeout 1800} {JNDINamejdbc/ds 1} {MaxPoolSize 10}
  {MinPoolSize 1} {OrphanTimeout 1800}{StatementCacheSize 100 }
  ```

  See 6.6.0.2.2.3.2: Specifying lists in wscp commands for details.

6.6.0.2.2.2.9: Specifying object names

In the console, simple names such as myAppServer or dataSource1 are used tospecify object names. However, in the repository, object names arestored as fully qualified names. Fully qualified names are requiredwhen using **wscp**. Fully qualified names reflect anobject's containment hierarchy. Containment determines howresources are related to one another.

For example, a node can contain application servers. An enterpriseapplication contains one or more modules.

Containment represents a hierarchical naming structure. Thisstructure prevents name clashes between objects that have the same name butbelong to different containment hierarchies. A resource's positionin the containment structure is used to generate the resource's full,unique name within the repository. Examples of containment hierarchiesare:

- Node/ApplicationServer
- JDBCDriver/DataSource
- URLProvider/URL

Containment is also used for efficient operations: starting acontained resource starts the resources above it in the hierarchy.Containment is also used to impose restrictions on relationships.

Fully qualified names have the form/objectType**:**objectInstance/.. . /. For example, if the application servernamed AppServ1 runs on the node named Node1, the repository name of theapplication server reflects this hierarchy. The name of the applicationserver, /Node:Node1/ApplicationServer:AppServ1/, is a fullyqualified name.

Some objects, for example Node objects, do not belong to a containmenthierarchy. These objects are known as *root types*. Anexample of a fully qualified name for the Node object named myNode is/Node:myNode/.

Note:

> You must use fully qualified names in **wscp** commands andscripts. Due to the syntax of fully qualified names, an object namecannot contain a colon (:). Fully qualified names can containspaces; if spaces are used, the name must be enclosed in braces or doublequotation marks (") so that Tcl parses the name as a single argument.

To reduce typing and for readability of scripts, use variables to store afull or partial name of an object. For example, the following commandsset variables for the name of a node, server, container, and bean.These variables can then be used in building other object names:

```
set node  "Node:my-Pc"set serv "/${node}/ApplicationServer:myServer/"
```

The braces are required so that the remaining part of the object name isnot considered part of the variable name.

Elsewhere, the variable names can be used as shown in the followingexample:

```
wscp> ApplicationServer create $serv
```

The containment operation lists the containment hierarchy for a specifiedobject type. This operation is useful for determining the requiredelements in a fully qualified name. The syntax is as follows:

```
object_type containment
```

The following example command displays the containment hierarchy for theModule object type:

```
wscp> Module containmentEntepriseApp Module
```

6.6.0.2.2.2.10: Using the wscp help facility

You can access **wscp** online help in varying levels of detail.Either of the following commands displays a summary of syntax for the helpfacility.

```
C:\> wscp -c "Help"
wscp> Help
```

The following topics contain examples of how to obtain help:

- Help on object types
- Help on operations
- Help on attributes

Help on object types

To obtain help on an object type, use the following syntax:

```
object_type help  [operation [ -verbose ] ]
```

When no arguments are specified, this command displays a brief descriptionof all valid operations for the object type. Depending on the argumentspecified, you can access help on the following:

- The operation argument displays valid options for the specifiedoperation.
- The -verbose option provides details on the options for anoperation.

For example, the following command requests help on the ApplicationServerobject. The command displays the names and descriptions of all validoperations for that object type.

```
wscp> ApplicationServer helpThe following actions are available for ApplicationServerattributes
Display the attributes of the objectcontainment        Display the containment hierarchy for the
objectcreate                 Create the specified objectdefaults     Display or set attribute
defaultshelp                  Display this help messagelist                   Display all the
instances of this typemodify                    Modify the attributes of the specified
objectoperations         List all the actions available on the object typeremove
Remove the specified objectshow                 Display the attributes of specified objectstart
Start the specified objectstop                  Stop the specified object
```

The following command displays help for the list operation ofApplicationServer object types:

```
wscp> ApplicationServer help listApplicationServer list [-constraint <attribute list>] [-recursive]
```

The following command displays verbose help for the list operation:

```
wscp> ApplicationServer help list -verboseApplicationServer listThe following options are available
for list[-constraint <attribute list>]      Display only objects which satisfy the
constraints[-recursive]                   Display subtype objects
```

The following command displays the valid options for the clone operation ofServerGroup objects:

```
wscp> ServerGroup help cloneServerGroup clone <name> -cloneAttrs <attribute list> -node <name>
```

The following command displays verbose help for the clone operation:

```
wscp> ServerGroup help clone -verboseServerGroup clone <name> -cloneAttrs <attribute list> -node
<name>The following options are available for clone -cloneAttrs <attribute list>    Attribute list
for the clone-node <name>                    Node name
```

Help on operations

To obtain help on an object type's operations, use the followingsyntax:

```
object_type operations
```

This command displays a list of all valid operations for the given objecttype.

The following examples display a list of all valid operations for theServerGroup and EnterpriseApp object types:

```
wscp> ServerGroup operationsattributes clone containment create help list listClones modify
operations remove removeClone show showAttrs start stopwscp> EnterpriseApp operationsattributes
containment create defaults help list modify operations remove show start stop install listmodules
listnodes showdeploymentinfo
```

Help on attributes

To obtain help on an object type's attributes, use the followingsyntax:

```
object_type attributes [ -cloneOnly ]   [ -modelOnly ]  [ -optional ][ -readOnly ]  [ -readWrite ]
[ -repository ]  [ -required ]  [ -runtime ]   [ -server ]  [ -startup ]
```

When no options are specified, this command displays a list of all validattributes for the given object type. Various options allow you todisplay a subset of attributes. The options are as follows:

- -cloneOnly. Displays only those attributes associated withclones of the object type.
- -optional. Displays only those attributes that do not require avalue.
- -readOnly. Displays only those attributes that can be read but notchanged directly. An example of a read-only attribute isCurrentState.
- -readWrite. Displays only those attributes that can be viewed andmodified. A read/write attribute can be changed at any time, and thechanges take effect immediately.
- -repository. Displays only those attributes that are stored in therepository. (The values of some attributes are short-lived andtherefore are not stored permanently, for example, a process ID.)
- -required. Displays only those attributes for which valuesmust be specified in order for an object to be created.
- -runtime. Displays only those attributes that come intoexistence after an object has been started. Run-time attributes areread-only attributes not stored in the repository. Examples of run-timeattributes are ProcessID and StartTime.
- -server. Displays only those attributes that reside in a server(run-time attributes of a server, for example, the ProcessIDattribute).
- -startup. Displays only those attributes that can be changedwhile a server is running but do not take effect until the next time theobject is started. Examples of startup attributes are Executable andEnvironment.

The following commands display all valid attributes of the Node andGenericServer object types:

```
wscp> Node attributesName FullName CurrentState DesiredState StartTime HostName HostSystemType
ProcessId InstallRoot PathMapwscp> GenericServer attributesName FullName CurrentState DesiredState
StartTime ProcessId Environment SelectionPolicy Executable ExecutableActive CommandLineArgs
CommandLineArgsActive EnvironmentActive UserId UserIdActive GroupId GroupIdActive WorkingDirectory
WorkingDirectoryActive Umask UmaskActive Stdin StdinActive Stdout StdoutActive Stderr StderrActive
MaxStartupAttempts ProcessPriority ProcessPriorityActive PingInterval PingTimeout PingInitialTimeout
```

The following command displays only required attributes of the DataSourceand JDBCDriver object types:

```
wscp> DataSource attributes -requiredName wscp> JDBCDriver attributes -requiredName ImplClass
```

The following command displays only those attributes of ApplicationServerobjects that apply to server groups and clones.

```
wscp> ApplicationServer attributes -modelOnlySelectionPolicywscp> ApplicationServer attributes
-cloneOnlyName FullName
```

See Printing an object's attributes for a custom Tcl procedure that provides a shortcut methodof displaying various combinations of attributes for one or moreobjects.

6.6.0.2.2.3: Advanced usage of wscp

This section contains the following topics on using **wscp** andTcl:

- 6.6.0.2.2.3.1: Using abbreviations in wscp commands
- 6.6.0.2.2.3.2: Specifying lists in wscp commands
- 6.6.0.2.2.3.3: Example use of wscp and Tcl
- 6.6.0.2.2.3.4: Using wscp and operating system commands
- 6.6.0.2.2.3.5: Obtaining status and error information
- 6.6.0.2.2.3.6: Use of qualified home names in the administrative server
- 6.6.0.2.2.3.7: Tracing the administrative server, application servers, and the wscp client
- 6.6.0.2.2.3.8: Enabling tracing with DrAdmin
- 6.6.0.2.2.3.9: Manipulating the JNDI context of objects
- 6.6.0.2.2.3.10: Monitoring performance
- 6.6.0.2.2.3.11: Setting global security defaults
- 6.6.0.2.2.3.12: Managing security roles
- 6.6.0.2.2.3.13: Connecting to remote servers
- 6.6.0.2.2.3.14: Where to find more information about Tcl

6.6.0.2.2.3.1: Using abbreviations in wscp commands

In interactive mode, the **wscp** interface allows abbreviations forthe first word of a command only (that is, for object types). All otheroptions must be fully specified. If an abbreviation is ambiguous,**wscp** returns an error that includes a list of all matches.Entering a question mark (?) at the `wscp>` prompt lists allsupported commands and object types (including Tcl commands).

In the following example, the word JMS is ambiguous:

```
wscp> JMS listAmbiguous command name  "JMS":  JMSConnectionFactory, JMSDestination, JMSProvider
```

To resolve the ambiguity, you must provide as many characters as needed touniquely identify the desired object type--for example, JMSD.

6.6.0.2.2.3.2: Specifying lists in wscp commands

Some **wscp** options take Tcl lists as values (notably the -attributeoption of create and modify operations). A Tcl list is an orderedcollection of elements, where each element can be a string, a number, oranother list. Lists can be delimited by either double quotation marks(" ") or braces ({ }). The following example sets the variable x to athree-element list whose first element is the string "account", whose secondelement is "term", and whose third element is itself a list containing the twostrings "maturity" and "date".

```
set x {account term {maturity date}}
```

In **wscp**, an attribute list is a list of attribute-value pairs (aTcl list of lists). The following command modifies the PingInterval andPingTimeout attributes of an application server. The argument to the-attribute option is a list containing two elements. Bothelements are themselves lists.

```
wscp> ApplicationServer modify /Node:dev-pc/ApplicationServer:myServer/  -attribute {{PingInterval
120}  {PingTimeout 240}}
```

Note that Tcl parsers sometimes require a space between list items.

An attribute list containing only one attribute-value pair must also beenclosed in a list. For example, in the following command, theattribute-value pair {PingTimeout 240} must be nested within braces:

```
wscp> ApplicationServer  modify /Node:dev-pc/ApplicationServer:myServer/  \-attribute {{PingTimeout
240}}
```

In addition to the modify operation, the show operation has an -attributeoption that expects a Tcl list as its argument. If you specify only oneattribute to be displayed, you can use either a string or a Tcl list.Tcl interprets both the string and the Tcl list of one element asequivalent--for example, the arguments JarFile and {JarFile} areequivalent.

As in **wscp**, Tcl commands consist of one or more words separatedby spaces. A Tcl word that contains spaces must be enclosed in eitherbraces ({ }) or double quotation marks (" "). The use of braces anddouble quotation marks is similar. Both braces and double quotationmarks can be placed around a word that contains embedded spaces.However, using braces and double quotation marks differs in tworespects. First, unlike double quotation marks, braces can nest.Also, no substitutions occur inside braces, as they do inside double quotationmarks. All characters between braces are passed as an argument to acommand or procedure, without any special processing. Substitutions canoccur later if the argument is evaluated again.

Object names in **wscp** can contain spaces and therefore must beenclosed in either braces or double quotation marks. If object namesare enclosed in braces, no evaluation takes place and everything in braces ispassed to the command as an argument. If substitution must take place,for example, when the object name being passed as an argument contains avariable that must be expanded to form the name, then the object name must beenclosed in double quotation marks. The following example commandsdemonstrate the use of braces and double quotation marks in each case:

```
# Braces used because object name contains spaces# Object name is passed to the command as an
argument# and no substitution takes place wscp> ApplicationServer create
{/Node:dev-pc/ApplicationServer:My Server/} wscp> ApplicationServer
list{/Node:dev-pc/ApplicationServer:Default Server/} {/Node:dev-pc/ApplicationServer:Appl EJB
Server/} {/Node:dev-pc/ApplicationServer:Model EJB Server/} {/Node:dev-pc/ApplicationServer:Model
EJB Server2/} {/Node:dev-pc/ApplicationServer:Bean EJBServer/} {/Node:dev-pc/ApplicationServer:My
Server/}  # Double quotation marks used because substitution must take place# Assumes the NODE
constant is defined as /Node:dev-pc/ wscp> ApplicationServer create "${NODE}ApplicationServer:Test
Server/" wscp> ApplicationServer list{/Node:dev-pc/ApplicationServer:Default Server/}
{/Node:dev-pc/ApplicationServer:Appl EJB Server/} {/Node:dev-pc/ApplicationServer:Model EJB Server/}
{/Node:dev-pc/ApplicationServer:Model EJB Server2/} {/Node:dev-pc/ApplicationServer:Bean EJBServer/}
{/Node:dev-pc/ApplicationServer:My Server/} {/Node:dev-pc/ApplicationServer:Test Server/}
```

Lists are a special case of quoted strings, and Tcl quoting conventions canbe nonintuitive. Be particularly careful when specifying strings thatcontain spaces. For example, the Environment attribute of applicationservers is a list of strings of the formname=value. If the value contains spaces,braces are required. The following command modifies the existing valueof an Environment attribute:

```
wscp> ApplicationServer modify /Node:dev-pc/ApplicationServer:Server1/ -attribute \{{Environment
{{VARIABLE=word1 word2}}}}
```

The following command creates an application server and specifies twovalues for the Environment attribute. Note that the PATH environmentvariable does not require braces because it contains no spaces.

```
wscp> ApplicationServer create /Node:dev-pc/ApplicationServer:Server2/ -attribute \{{Environment
{PATH=/myPath {OTHERVARIABLE=word1 word2}}}}
```

The following commands display the Environment attributes of both serversin the above example:

```
wscp> ApplicationServer show /Node:dev-pc/ApplicationServer:Server1/ -attribute
Environment{Environment {{VARIABLE=word1 word2}}}wscp> ApplicationServer show
/Node:dev-pc/ApplicationServer:Server2/ -attribute Environment{Environment {PATH=/myPath
{OTHERVARIABLE=word1 word2}}}
```

Note that when you modify an attribute, you are replacing the existingvalue with a new value. If you want to add to or replace part of anexisting value (for example, change a path name), you must use a customprocedure. See Modifying an Environment attribute (modEnv procedure) for an example procedure named modEnv. The procedurecan be used to modify the Environment attribute of one or more servers in adomain. It modifies one element of the list and retains the values ofthe other elements.

Other attributes that take a list of strings as their arguments are theCommandLineArgs and SystemProperties attributes of application serverobjects.

6.6.0.2.2.3.3: Example use of wscp and Tcl

The **wscp** interface consists of **wscp** operations andbuilt-in Tcl commands. Tcl provides a portable method of controllingand extending **wscp** administrative operations. You can usenative Tcl commands for creating and executing new commands (the**proc** and **eval** commands), conditionalizing andcontrolling the flow of execution (if statements and loops), and handlingerrors and exceptions (the **catch** command).

The Tcl **foreach** looping command iterates over all elements in alist. In the following example, the **foreach** command is usedto iterate over all instances of ApplicationServer objects in a domain andthen stop each server object. The square brackets ([ ]) invoke commandsubstitution--the result of the ApplicationServer list operation (a listof server names) is used as the list argument to the **foreach**command. In turn, each ApplicationServer server name is substituted forthe variable $ejbserver.

```
wscp> foreach ejbserver [ApplicationServer  list] \ {puts "stopping $ejbserver...";
ApplicationServer stop $ejbserver}
```

As part of many server administration tasks, you often need to monitor thevalues of one or more server attributes. To do so, you can create andrun a procedure that displays the attributes of interest. The followingis an example procedure called showServerStatus, which displays the currentstate for each application server in a domain. As written, theshowServerStatus procedure displays the Name and CurrentState attributes ofall application servers. You can modify the procedure to displayadditional or different attributes.

```
proc showServerStatus {} {      puts "\nStatus of servers in the domain:\n"          foreach
ejbserver [ApplicationServer list] {          set serverInfo($ejbserver) [ApplicationServer show
$ejbserver -attribute \          {Name CurrentState}]          puts $serverInfo($ejbserver)
}}
```

The following example demonstrates output of the showServerStatusprocedure:

```
wscp> showServerStatusStatus of servers in the domain:{Name {Default Server}} {CurrentState
{Initialization Failed}}{Name {Appl EJB Server}} {CurrentState Running}{Name {Model EJB Server}}
{CurrentState Stopped}{Name {Model EJB Server2}} {CurrentState Stopped}{Name {Bean EJB Server}}
{CurrentState Running}{Name {My Server}} {CurrentState Stopped}{Name {Test Server}} {CurrentState
Stopped}
```

The following procedure displays the attributes of a specified objectinstance, formatting them so that the attributes are displayed one per line,without enclosing braces.

```
proc display {type name} {      set attrs [$type show $name]    foreach attr $attrs {           puts
$attr      }}
```

The following example demonstrates output from the displayprocedure. The procedure is used display the attributes of theDataSource object named testDataSource:

```
wscp> display DataSource /JDBCDriver:OracleJDBC/DataSource:testDataSource/Name
testDataSourceFullName /JDBCDriver:OracleJDBC/DataSource:testDataSource/ConfigProperties {URL
jdbc:oracle:thin:@wssol:1521:oraejs}ConnTimeout 300DatabaseName WASDisableAutoConnectionCleanup
FalseIdleTimeout 1800JNDIname testDataSourceMaxPoolSize 30MinPoolSize 3OrphanTimeout
1800StatementCacheSize 100
```

6.6.0.2.2.3.4: Using wscp and operating system commands

All UNIX and Windows NT commands can be issued from within a **wscp**session. In interactive mode, Tcl executes operating system commandswith an explicit **exec** command. The **exec** commandcreates one or more subprocesses and waits until they complete beforereturning. If you wish to disable this behavior and instead have Tclsearch your UNIX or Windows NT PATH environment variable for command namesbefore checking whether they are abbreviations, enter the following command ininteractive mode:

```
wscp> unset auto_noexec
```

To check the definition of the variable and to reset it to true, enter thefollowing in interactive mode:

```
wscp> set auto_noexec wscp> set auto_noexec true
```

The Tcl **info globals** command returns the names of all globalvariables currently defined.

6.6.0.2.2.3.5: Obtaining status and error information

Successful **wscp** commands return a result, typically either alist of information or an empty string. When commands are runinteractively, the Tcl result is displayed by the **wscp**shell. Failed **wscp** commands raise a TclException, which,unless there is a **catch** clause, stops the execution of theenclosing procedure.

When an exception is caught by **wscp**, the stack trace is appendedto the Tcl variable errorInfo. You can view the stack trace by issuingeither of the following commands:

```
wscp> puts $errorInfowscp> set errorInfo
```

The Tcl variable errorCode is set when **wscp** commands areexecuted. A nonzero value represents an error returned by thecommand.

Note:

> Interactive users and writers of Tcl scripts must check this variable todetermine whether a command succeeded or failed.

The errorCode variable is set to an integer (0 indicates success).The static statusToString method ofcom.ibm.ejs.sm.ejscp.WscpStatus can be usedto translate an errorCode value. For example, the following Tclprocedure takes an errorCode value and translates it:

```
# Converts a WscpStatus to its corresponding string translation#proc statusToString {{status -1}} {
global errorCode    if {$status == -1 && $errorCode != "NONE"} {set status $errorCode}    java::call
com.ibm.ejs.sm.ejscp.WscpStatus statusToString $status}
```

Note that this Tcl procedure can be used to convert any error code; ifan error code argument is not provided, the procedure converts the currentvalue of the global errorCode variable.

The sample Tcl script init.tcl contains the statusToString procedureand other useful procedures for debugging. See 6.6.0.2.2.4.1: Initialization and general-purpose procedures.

6.6.0.2.2.3.6: Use of qualified home names in the administrative server

The following property is used by the administrative server to set up theJava Naming and Directory Interface (JNDI) namespace. It specifieswhether enterprise bean homes are looked up in the initial context or in aspecified subcontext. If this property is set to true, bean homes arelooked up in the specified subcontext.

```
com.ibm.ejs.sm.adminServer.qualifyHomeName
```

Both the administrative server and **wscp**, by default, have thisproperty set to true. If the administrative server is running withqualified home names, **wscp** must also run with qualified homenames. The following property controls the use of qualified home namesin **wscp**:

```
wscp.qualifyHomeName
```

Note:

> This property is not related to the fully qualified names used for objectinstances.

You can disable the use of qualified home names in an administrative serverby setting thecom.ibm.ejs.sm.adminServer.qualifyHomeNameproperty to false in the admin.config file. This file is locatedin the directory where Advanced Application Server was installed (typically,on Windows NT systems,installation_drive\Websphere\Appserver\bin). If this propertyis set to false, the corresponding property in **wscp** must also beset to false.

6.6.0.2.2.3.7: Tracing the administrative server, application servers, and the wscp client

Use either of the following methods to enable tracing for a **wscp**client:

- Set trace properties (on the Java command line or in the .wscprcfile).
- Load thecom.ibm.ejs.sm.ejscp.DrAdminExtensionextension and then set trace properties by using the -setTrace option of theDrAdmin local operation. See 6.6.0.2.2.3.8: Enabling tracing with DrAdmin for details.

The following example lines set trace properties. The first lineenables all tracing for all of the wscp classes; the second line enablestracing for only the wscp.commands classes.

```
wscp.traceString=com.ibm.ejs.sm.ejscp.*=all=enabledwscp.traceString=com.ibm.ejs.sm.ejscp.commands.*=all=enabled
```

You can also use the DrAdmin extension to enable and set tracing for theadministrative server and for application (enterprise bean) servers.

If you are not using the DrAdmin extension, you must use the console toenable tracing for an application server. You can, however, use**wscp** to set trace for these servers by modifying the TraceSpecattribute of the server object.

See Administering the product messages,logs, and traces for additional information on tracing.

6.6.0.2.2.3.8: Enabling tracing with DrAdmin

In addition to the EjscpExtension class, WebSphere Advanced ApplicationServer includes the **wscp** DrAdmin operations for tracing. TheDrAdmin operations can be used to trace an administrative server or anyapplication server in a domain.

- The DrAdmin local operation traces the **wscp** clientitself.
- The DrAdmin remote operation traces the administrative server or anapplication server (the server can be local, running on the same machine as**wscp**, or it can be running on a remote machine).

The syntax for both commands is as follows:

```
DrAdmin local  [-setTrace trace_spec] [-setRingBufferSize size] [-dumpRingBuffer file_name]
[-dumpState file_name]DrAdmin remote  server_port [-serverHost host_name [-setTrace trace_spec]
[-dumpRingBuffer file_name] [-dumpState file_name[-setRingBufferSize string] [-stopServer]
[-stopNode] [-dumpThreads]
```

The arguments and options are as follows:

- server port. A port number is displayed as an Auditmessage in the Console Messages window when each server starts and when theadministrative server starts. These audit messages are written to thefile named tracefile in the logs subdirectory of the WebSphere homeinstallation directory.
- -serverHost. Specifies the name of the machine where theadministrative server or application server is running.
- -setTrace. Specifies a trace specification.
- -setRingBufferSize. Specifies the size of the ring buffer inkilobytes. The default is 8 KB.
- -dumpRingBuffer. Specifies a file name where the ring buffer is tobe written. By default, the ring buffer is written to the currentworking directory. If the administrative server is started as a WindowsNT service, the ring buffer is written to the system default directoryC:\WINNT\system32.
- -dumpState. Specifies a file name where state information for theserver is to be written.
- -stopServer. Stops the server being traced.
- -stopNode. Stops the node.
- -dumpThreads. Specifies a file name where thread history and errorinformation is to be written.

In the following example, the DrAdmin extension is used to trace theadministrative server running on port 1078. By default, the trace ringbuffer is written to the system default directory. The second examplespecifies a file name where the trace is to be written. (For Windows NTpathnames, you can use forward slashes (/). If backslashes are used,they must be prefaced with the backslash (\) character so that the backslashesare treated as ordinary characters.) The third example sets a tracespecification that enables tracing for all container classes.

```
wscp> DrAdmin remote 1078Server trace ring buffer dumped into file JmonDump52701921622wscp> DrAdmin
remote 1078 -dumpRingBuffer e:\\wscp\\dradmin.dumpServer trace ring buffer dumped into file
e:\wscp\dradmin.dumpwscp> DrAdmin remote 1078 -setTrace com.ibm.ejs.container.*=all=enabledServer
trace set to com.ibm.ejs.container.*=all=enabledServer trace ring buffer dumped into file
e:\wscp\dradmin.dump
```

6.6.0.2.2.3.9: Manipulating the JNDI context of objects

The **wscp** Context operations allow you to manipulate the JavaNaming and Directory Interface (JNDI) context of objects that use JNDI namebindings, such as J2CConnectionFactory, DataSource, URL and MailSessionobjects.

The following example creates an initial JNDI context:

```
wscp> Context init
```

The following command binds a MailSession object to the JNDI naming contextejsadmin:

```
wscp> Context bind /MailSession:UserMail/  ejsadmin
```

The following command unbinds a DataSource object from itsJNDI namingcontext:

```
wscp> Context unbind /JDBCDriver:DB2drv/DataSource:appData/
```

For more information on JNDI, see JNDI (Java Naming andDirectory Interface) overview.

6.6.0.2.2.3.10: Monitoring performance

Use the **wscp** PmiService operations to monitor application serverperformance. The PmiService operations can be used to enable or disabletracing, check the values of performance counters, and monitor otherperformance measures. The full range of WebSphere performancemonitoring functions is available through Resource Analyzer.

The following command example turns on tracing for the specifiedapplication server:

```
wscp> PmiService enableData /Node:Appserv1/ApplicationServer:sampleServ/ -dd {JVMRuntimeModule}
```

The following command example turns off tracing for the specifiedapplication server:

```
wscp> PmiService disableData /Node:Appserv1/ApplicationServer:sampleServ/ -dd {JVMRuntimeModule}
```

The following command example displays configuration information for thespecified server:

```
wscp> PmiService getConfigs /Node:Appserv1/ApplicationServer:sampleServ/ [output too long to
display]
```

The following command example lists the modules for which performancestatistics are collected:

```
wscp > PmiService listMembers /Node:Appserv1/ApplicationServer:sampleServ/jvmRuntimeModule
threadPoolModule transactionModule
```

The following command example displays performance information for thespecified server:

```
wscp> PmiService get /Node:Appserv1/ApplicationServer:sampleServ/ [output too long to display]
```

The following command example displays performance information for thespecified module:

```
wscp> PmiService gets /Node:Appserv1/ApplicationServer:sampleServ/ -dd jvmRuntimeModule{Description
jvmRuntimeModule.desc} {Descriptor {NamejvmRuntimeModule} {Type 13}
{FullNameroot/wssol2/ApplicationServer:BrokerAppSrv/jvmRuntimeModule} {NodeNamewssol2} {ServerName
ApplicationServer:BrokerAppSrv} {ModuleNamejvmRuntimeModule} {MaxPathLength 3} {DataDescriptor
{NamejvmRuntimeModule} {Type 13} {ModuleName jvmRuntimeModule} {DataId -1}{Path jvmRuntimeModule}}
{Path wssol2 ApplicationServer:BrokerAppSrvjvmRuntimeModule}}
```

The following command example displays the performance level descriptorsfor the specified application server:

```
wscp> PmiService getLevel /Node:Appserv1/ApplicationServer:sampleServ/{pmi/beanModule 0}
{pmi/threadPoolModule 0} {pmi/connectionPoolModule0} {pmi/jvmRuntimeModule 0} {pmi/transactionModule
0}{pmi/webAppModule 0} {pmi/servletSessionsModule 0} {pmi/jvmpiModule 0}
```

For more information on performance monitoring, see TheWebSphere Resource Analyzer.

6.6.0.2.2.3.11: Setting global security defaults

The **wscp** SecurityConfig operations can be used to do thefollowing:

- Enable and disable WebSphere security on a global basis.
- Specify the default authentication mechanism.
- Configure secure sockets layer (SSL) communication.

They cannot be used to configure security for individual applications orcomponents.

Modifying security properties

Before enabling security, you must make the following modifications to theproduct_installation_root/properties/sas.client.propsfile:

```
com.ibm.CORBA.loginSource=propertiescom.ibm.CORBA.loginUserId=userIDcom.ibm.CORBA.loginPassword=password
```

where *userID* is a valid userID and *password* is thecorresponding password.

The sas.client.props file specifies login information forboth the administrative console and **wscp**. However, you canset up **wscp** to use a different login mechanism than theadministrative console --for instance, you can set up **wscp** fora programmatic login and the console for an interactive login. Do thefollowing:

1. Copy the sas.client.props file to another directory.
2. Change the security properties for **wscp** that were describedearlier in this article.
3. In the file setupCmdLine.bat (Windows) or setupCmdLine.sh(Unix), set the value of the WSCPCLIENTSAS variable to the location of thecopied file. For example:

   ```
   WSCPCLIENTSAS=new_directory/sas.client.props
   ```

   where *new_directory* is the directory where the modified copyof the sas.client.props file resides.

See 6.6.0.2.2.2.3: Authenticating to the administrative server for more information about enabling security.

Security configuration examples

The following example command checks whether security is enabled:

```
wscp> SecurityConfig isSecurityEnabled
```

Return values are:

- 1 (true)--Security is enabled.
- 0 (false)--Security is disabled.

The following example command enables security for all applications:

```
wscp> SecurityConfig enableSecurity
```

The following example command disables security for all applications:

```
wscp> SecurityConfig disableSecurity
```

The following example command returns the current authentication mechanismfor security:

```
wscp> SecurityConfig getAuthenticationMechanism
```

Possible return values are:

- LOCALOS -- The underlying operating system's authenticationmechanism. The local operating system supports basic authenticationsuch as checking a user ID and password.
- LTPA -- Lightweight Third Party Authentication (LTPA). LTPAauthenticates users with a Lightweight Directory Access Protocol (LDAP)directory service and supports certificate-based authentication.

Note:

> LTPA cannot be directly configured from the **wscp** command linebecause the configuration settings are too complex. However, you canuse the **wscp** XMLConfig operation to import LTPA configurations thathave been stored in XML files. See 6.6.0.2.2.4.6: Importing and exporting a configuration by using XMLConfig for instructions on how to use this command.

The following example command returns the user ID that can be used in localoperating system (LOCALOS) authentication:

```
wscp> SecurityConfig  getUserid{tym}
```

The following example command sets the authentication method to that of thelocal operating system and authenticates to the user tym:

```
wscp> SecurityConfig  setAuthenticationMechanism  LOCALOS -userid {{tym} {tympwd}}
```

The system uses the operating system's existing securityrepository. The administrative server must be restarted for the changeto take effect.

The following example command displays information about how SSL isconfigured in WebSphere Application Server:

```
wscp> SecurityConfig getSSLConfig{{TrustFileName
${WAS_HOME}/etc/ServerTrustFile.jks}{TrustFilePassword WebAS}
{KeyFileName${WAS_HOME}/etc/ServerKeyFile.jks} {KeyFilePassword WebAS}{KeyFileFormat 0}
{TrustFileFormat 0} {ClientAuthentication false}{UseGlobalDefaults true} {SecurityLevel 0}
{CryptoHardwareEnabledfalse} {CryptoTokenType {}} {CryptoLibraryFile {}} {CryptoPassword
{}}{SSLProperties {}}}
```

The following example command sets various SSL configurationparameters:

```
wscp> SecurityConfig setSSLConfig {{ClientAuthentication true}
{KeyFileName${WAS_HOME}/etc/NewKey.jks} {KeyFilePassword serverAS}}
```

6.6.0.2.2.3.12: Managing security roles

The **wscp** SecurityRoleAssignment operations allow you to managesecurity roles for J2EE applications.

Role-based security enables declarative, customized authentication forapplications. When a J2EE application is assembled, permission toexecute methods is granted to one or more roles, which represent abstractgroups of users. When the application is deployed, actual users orgroups of users are assigned to these roles. When the application isrun, WebSphere Application Server authorizes client requests based on theuser's identification information and what roles the user is assignedto. For a more detailed description of how role-based authentication isimplemented, see article 5.1.3.

The **wscp** SecurityRoleAssignment operations can be used toperform the following tasks:

- List the roles that are defined for an enterprise application.
- List the users and groups that are assigned to each role.
- Add users and groups to a role.
- Delete users and groups from a role.
- Specify the identity under which enterprise bean methods areexecuted.

In addition to individual users and groups, the special groups all usersand all authenticated users can be assigned to security roles.

Note that the SecurityRoleAssignment operations only work with existingsecurity roles; they cannot be used to define new roles. However,you can define and assign security roles when you install an enterpriseapplication with the **wscp** EnterpriseApp install command or installa module with the **wscp** Module install command. See 6.6.0.2.2.4.8: Creating an enterprise application for more information.

The **wscp** SecurityRoleAssignment examples in this section makeuse of the user-to-role mapping in the following table.

Banking enterprise application		**Roles**			
		Teller	**Clerk**	**Supervisor**	**WebTeller**
Users and groups	TellerGroup	Yes			Yes
	Bob	Yes	Yes		Yes
	Mary			Yes	
	ClerkGroup		Yes		
	Supervisor			Yes	
	SupervisorGroup			Yes	

The following example command lists the roles defined for the Bankingenterprise application:

```
wscp> SecurityRoleAssignment listRoles /EnterpriseApp:Banking/Teller Clerk Supervisor WebTeller
```

The following example command lists the roles defined for the Bankingapplication and the users assigned to each role:

```
wscp> SecurityRoleAssignment getUserRoleMapping /EnterpriseApp:Banking/ {Teller {Bob}} {Clerk {}}
{Supervisor {Supervisor}} {WebTeller {Bob}}
```

The following example command lists the users assigned to the Teller rolefor the Banking application:

```
wscp> SecurityRoleAssignment getUserRoleMapping /EnterpriseApp:Banking/ -roles {Teller}{Teller
{Bob}}
```

The following example command lists the roles to which the user Bob isassigned:

```
wscp> SecurityRoleAssignment getUserRoleMapping /EnterpriseApp:Banking/  -users {Bob} {Teller {Bob}}
{WebTeller {Bob}}
```

The following example command lists the roles defined for the Bankingapplication and the groups assigned to each role:

```
wscp> SecurityRoleAssignment getGroupRoleMapping /EnterpriseApp:Banking/ {Teller {TellerGroup}}
{Clerk {ClerkGroup}} {Supervisor {}} {WebTeller {TellerGroup}}
```

The following example command lists the groups assigned to the WebTellerrole for the Banking application:

```
wscp> SecurityRoleAssignment getGroupRoleMapping /EnterpriseApp:Banking/  -roles
{WebTeller}{WebTeller {TellerGroup}}
```

The following example command lists the special role mappings for theBanking application (that is, whether the role has been assigned to all usersor all authenticated users):

```
wscp> SecurityRoleAssignment getSpecialRoleMapping /EnterpriseApp:Banking/
```

Return values are:

- `Everyone`--All users
- `AllAuthenticatedUsers`--All authenticated users

The following example command adds the user Mary to the Teller andWebTeller roles:

```
wscp> SecurityRoleAssignment addUserRoleMapping /EnterpriseApp:Banking/ -userroles {{Teller Mary}
{WebTeller Mary}}
```

Use the -userroles option to specify which users are added to whichroles. Enter either a role-user pair (such as {Teller Mary}) or a listof role-user pairs (such as {{Teller Mary} {WebTeller Mary}}).

The following example command adds the group ClerkGroup to the WebTellerrole:

```
wscp> SecurityRoleAssignment addGroupRoleMapping /EnterpriseApp:Banking/ -grouproles {WebTeller
ClerkGroup}
```

Use the -grouproles option to specify which groups are added to whichsecurity roles. Enter either a role-group pair (such as {WebTellerClerkGroup}) or a list of role-group pairs (such as {{WebTeller ClerkGroup}{WebTeller SupervisorGroup}}).

The following example command adds the special group AllAuthenticatedUsersto the Clerk role:

```
wscp> SecurityRoleAssignment addSpecialRoleMapping /EnterpriseApp:Banking/-specialroles {{Clerk
AllAuthenticatedUsers} {Teller Everyone}}
```

Use the -specialroles option to specify which special groups are assignedto which security roles. Enter either a role-special group pair(such as {Clerk AllAuthenticatedUsers}) or a list of role-special group pairs(such as {{Clerk AllAuthenticatedUsers} {Teller Everyone}}).

The following example command deletes the user Bob from the Teller andWebTeller roles:

```
wscp> SecurityRoleAssignment deleteUserRoleMapping /EnterpriseApp:Banking/-userroles {{Teller Bob}
{WebTeller Bob}}
```

Use the -userroles option to specify which users are deleted from whichroles. Enter either a role-user pair (such as {Teller Bob}) or a listof role-user pairs (such as {{Teller Bob} {WebTeller Bob}}).

The following example command deletes the group ClerkGroup from theWebTeller role:

```
wscp> SecurityRoleAssignment deleteGroupRoleMapping /EnterpriseApp:Banking/-grouproles {WebTeller
ClerkGroup}
```

Use the -grouproles option to specify which groups are deleted from whichsecurity roles. Enter either a role-group pair (such as {WebTellerClerkGroup}) or a list of role-group pairs (such as {{WebTeller ClerkGroup}{WebTeller SupervisorGroup}}).

The following example command deletes the special groups AllUsers andAllAuthenticatedUsers from the Clerk role:

```
wscp> SecurityRoleAssignment deleteSpecialRoleMapping /EnterpriseApp:Banking/ -specialroles {{Clerk
AllAuthenticatedUsers} {Clerk Everyone}}
```

Use the -specialroles option to specify which special groups are deletedfrom which security roles. Enter either a role-special group pair(such as {Clerk AllAuthenticatedUsers}) or a list of role-special group pairs(such as {{Clerk AllAuthenticatedUsers} {Clerk Everyone}}).

The following example command lists the execution identities and roles thatenterprise bean methods run under in the Banking application:

```
wscp> SecurityRoleAssignment getRunAsToUser /EnterpriseApp:Banking/{Supervisor {Supervisor Bob}}
```

The following example command assigns the execution identity Mary forenterprise bean methods that run under the Supervisor role. A passwordmust also be specified; in this case, the password is marypwd.

```
wscp> SecurityRoleAssignment setRunAsToUser /EnterpriseApp:Banking/ -runasroles {Supervisor Mary
marypwd}
```

6.6.0.2.2.3.13: Connecting to remote servers

The **wscp** Remote operations enable **wscp** to administerremote servers as follows:

- Connect to remote servers by using the Remote attach command.
- Disconnect from remote servers by usng the Remote detach command.
- Start a server listener process by using the Remote listen command.

The following example command shows how to create a connection to a remoteserver:

```
wscp> Remote attach greenland.ibm.com:1000:123898
```

where greenland.ibm.com is the server name, 1000 is the portnumber, and 123898 is the key number.

The following example command closes all remote connections:

```
wscp> Remote detach
```

The following example commands opens up a server listener process for thespecified server, port number, and key:

```
wscp> Remote listen greenland.ibm.com:1000:123898Server listening at greenland.ibm.com:1000:123898
```

6.6.0.2.2.3.14: Where to find more information about Tcl

The following are some useful Tcl commands for writing **wscp**scripts. For additional information on Tcl, refer to *Tcl and theTK Toolkit* by John K. Ousterhout (Addison Wesley), or to the Tcldeveloper Web site at http://www.scriptics.com.

- **set**. Creates, reads, and modifies variables.

  ```
  set serv "/Node:dev-pc/ApplicationServer:myAppServer/"
  ```
- **eval**. Accepts any number of arguments, concatenates themwith separator spaces, then executes the result as a Tcl script. Oneuse of **eval** is for generating commands, saving them in variables,and then later evaluating the variables as Tcl scripts.

  ```
  set cmd {ApplicationServer stop /Node:MyNode/ApplicationServer:MyServer/}. . .eval $cmd
  ```
- **exec**. Creates one or more new processes and waits untilthey are complete before returning. Looks for an executable file in theworking directory or uses the PATH environment variable.

  ```
  exec date
  ```
- **global**. Makes global variables available inside aprocedure.

  ```
  global errorCode
  ```
- **lappend**. Appends new elements to a list stored in avariable.

  ```
  set vars {value1 value2 value3}value1 value2 value3lappend vars value4value1 value2 value3 value4
  ```
- **lindex**. Extracts an element from a list.

  ```
  lindex $vars  2value3
  ```
- **lreplace**. Deletes elements from a list and optionallyadds new elements. The first argument is a list, and the second andthird arguments are the indices of the first and last elements to bedeleted.

  ```
  lreplace $vars 1 2value1 value4
  ```
- **lsearch**. Searches a list for an element with aparticular pattern and returns the index of the first matching element that isfound.

  ```
  lsearch $vars value43
  ```
- **proc**. Creates a named procedure and assigns a list ofarguments to be used with that procedure.

  ```
  proc checkStatus {expectedStatus}proc getAttrs {name array args}
  ```

6.6.0.2.2.4: Example wscp commands, procedures, and scripts

This section contains example **wscp** commands, Tcl procedures, andTcl scripts. You can use these examples as provided, or customize themto develop procedures and scripts suited to your application. Many ofthe procedures and scripts are available as stand-alone files in 6.6.0.2.2.5: Sample Tcl procedures and scripts.

Note that not all **wscp** commands are represented in theseexamples. The examples are intended to demonstrate only the morefrequently used commands (such as commands for creating and installingapplication servers or displaying the attributes of an object). Theexamples include several custom procedures and other files that offer guidancein developing your own scripts and procedures.

Consult the **wscp** command-line help for a complete list ofoperations for each object type.

The following list contains a description of the administrative task thateach example performs; states whether the example is a stand-alone**wscp** command, a Tcl procedure, or a Tcl script; and includesthe name of the Tcl procedure or script.

Note:

> Many of the example scripts and procedures require that the fileinit.tcl be loaded prior to running. This file initializesseveral variables used elsewhere and contains useful Tcl procedures foraccessing attributes, obtaining error status information, and othertasks. See 6.6.0.2.2.4.1: Initialization and general-purpose procedures for a description of the contents of this file.

- 6.6.0.2.2.4.1: Initialization and general-purpose procedures
- 6.6.0.2.2.4.2: Configuring objects
    - Creating objects. Stand-alone **wscp** commands. See Creating an object.
    - Viewing the default values of attributes. Stand-alone**wscp** commands. See Working with the default values of attributes.
    - Modifying objects. Stand-alone **wscp** commands. SeeModifying an object.
    - Modifying the Environment attribute. A custom procedure, modEnv,for modifying or replacing values in the Environment attribute. Theprocedure can be customized for modifying the values of other attributes thatalso require a Tcl list of strings. See Modifying an Environment attribute (modEnv procedure). This procedure is available in 6.6.0.2.2.5: Sample Tcl procedures and scripts.
- Starting live repository objects. Stand-alone **wscp**commands for starting and stopping application servers and other liveobjects. See 6.6.0.2.2.4.3: Starting and stopping live repository objects.
- 6.6.0.2.2.4.4: Displaying information about objects
    - Listing instances of an object type, including listing the objectsrecursively. Stand-alone **wscp** commands. See Listing objects.
    - Querying an object (displaying the values of an instance'sattributes). Stand-alone **wscp** commands and a customprocedure named display. See Querying (displaying) attributes. This procedure is available in 6.6.0.2.2.5: Sample Tcl procedures and scripts.
    - Printing the attributes of an object type. Custom procedure,printAttributes, for printing the attributes of all or select objects.See Printing an object's attributes. This procedure is available in 6.6.0.2.2.5: Sample Tcl procedures and scripts.
    - Viewing the containment hierarchy for all object types or for selectedobject types. Custom procedure, printContainment, for displaying thecontainment hierarchy for all or selected objects.

  See Viewing the containment hierarchy. This procedure is available in 6.6.0.2.2.5: Sample Tcl procedures and scripts.

  - Displaying selected attributes of an application server. Customprocedure, showServerStatus, that displays the name and current state of allservers in a domain. Can be customized to display otherattributes. See Displaying select attributes. This procedure is available in 6.6.0.2.2.5: Sample Tcl procedures and scripts.

- Removing objects recursively. Stand-alone **wscp**commands. The commands are useful in testing situations where objectsmust be routinely created and destroyed. See 6.6.0.2.2.4.5: Removing objects and applications.
- Invoking the XML Configuration Management Tool (XMLConfig) from within**wscp**. Stand-alone **wscp** command. See 6.6.0.2.2.4.6: Importing and exporting a configuration by using XMLConfig.
- Creating a JDBCDriver and DataSource object. Stand-alone**wscp** commands. See 6.6.0.2.2.4.7: Creating and installing drivers and data sources.
- Creating an enterprise application. Tcl scripts and stand-alone**wscp** commands that demonstrate installing and starting anenterprise application and installing Enterprise JavaBeans (EJB) and WebArchive (WAR) modules. See 6.6.0.2.2.4.8: Creating an enterprise application. These scripts are available in 6.6.0.2.2.5: Sample Tcl procedures and scripts.
- Configuring server groups and clones. Sample **wscp**commands that exercise most operations associated with server groups andclones. The commands can be used alone or in combination. See 6.6.0.2.2.4.10: Configuring server groups and clones.
- Administering connections to enterprise information system (EIS) backendsystems. Stand-alone **wscp** commands for installing andcreating instances of J2C resource adapters and J2C connectionfactories. See 6.6.0.2.2.4.11: Administering EIS connections.
- Configuring Java Message Service (JMS) clients. Stand-alone**wscp** commands for creating JMS providers, JMS connection factoriesand JMS destinations. See 6.6.0.2.2.4.12: Administering the Java Message Service (JMS).
- Administering JavaMail sessions. Stand-alone **wscp**commands for creating and destroying JavaMail sessions. See 6.6.0.2.2.4.13: Administering JavaMail sessions.
- Administering URLs and URL providers. Stand-alone **wscp**commands for installing and creating instances of URLs and URLproviders. See 6.6.0.2.2.4.14: Administering URL providers and URLs.

6.6.0.2.2.4.1: Initialization and general-purpose procedures

The init.tcl script initializes variables used elsewhere in theexamples. Note that init.tcl was used for a specific test suite,but it contains several procedures that can be generally useful. If youare writing scripts that must accept some common command-line arguments, youcan use or add to the predefined variables as needed. The contents ofinit.tcl are as follows:

- getAttrs and setAttrs. Procedures that get and set an array ofattributes for a specified object.
- getProperty. Procedure that retrieves a specified system property,such as the operating system name and operating system-specific fileseparator.
- which. Procedure that provides the Tcl equivalent of thecorresponding UNIX command. Retrieves the path to the specifiedargument (must be an executable) on java.library.path.
- parseArguments. Sets global variables for the host name, hostinternet address, and node. Also sets the VERBOSE globalvariable. If VERBOSE is set to 1, the **wscp** command beingexecuted is echoed to the screen.
- initConstants. Sets up lists or arrays of lists for **wscp**commands, operations, actions, and status values. Also sets thelocation of home directories for WebSphere Application Server, DB2, the IBMDebugger; the port number of the administrative server; and othermiscellaneous information.
- statusToString. Translates a specified **wscp** status toits corresponding string equivalent, or if called with no arguments,translates the current value of $errorCode.
- checkStatus. Tests whether the expected status matches thespecified status, or as in the previous item, the current value of $errorCodeif no status argument is provided.

The init.tcl script uses the global variable VERBOSE to echo the**wscp** commands as they are being executed. To set thisvariable to true, specify 1 as the value of the -verbose option when runningwscp.bat.

```
C:\> wscp -verbose 1
```

The init.tcl script is available in 6.6.0.2.2.5: Sample Tcl procedures and scripts.

6.6.0.2.2.4.2: Configuring objects

The following examples demonstrate how to create and modify objects usingthe **wscp** interface:

- Creating an object
- Working with the default values of attributes
- Modifying an object
- Modifying an Environment attribute (modEnv procedure)

Creating an object

The **wscp** create operation creates an object instance. Thesyntax is as follows:

```
object_type create object_name -attribute attr_list
```

The arguments are as follows:

- object_type. Specifies the object type of theinstance.
- object_name. Specifies the object instance to becreated.
- -attribute attr_list. Specifies a Tcl list ofattribute-value pairs to set.

The following interactive **wscp** command creates an applicationserver. The object_type argument (ApplicationServer) specifiesthat an application server object is to be operated on. Theoperation argument (create) specifies that an object is to becreated. The object_name argument(/Node:dev-pc/ApplicationServer:myServer/) is the name of theserver object to be created.

```
wscp> ApplicationServer create /Node:dev-pc/ApplicationServer:myServer/
```

The following **wscp** command creates a DataSource object namedds1.

```
wscp> DataSource create /JDBCDriver:Db2Jdbc/DataSource:ds1/
```

You must specify the name of the desired JDBC provider when creating a datasource -- in this case, the driver /JDBCDriver:Db2Jdbc/

Working with the default values of attributes

An object type's attributes can have default values. When youcreate an object instance, the instance inherits any default values (unlessyou explicit set them in the create operation). For example, theSecurityEnabled attribute of an ApplicationServer object has a default valueof False. Some attributes have a default value of null or the emptylist--for example, the value of the Environment attribute ofApplicationServer objects defaults to the empty list.

The defaults operation is used to view the default values forattributes. (All object types support the defaults operation with theexception of the ServerGroup object type.) The syntax is asfollows:

```
<object_type> defaults  [-all] [-attribute <attribute list>]
```

If you do not specify any options, the defaults operation displays thedefault values for attributes of the specified object type. Thefollowing example displays the default values for attributes of the DataSourceobject type. Note that, while other attributes exist for this objecttype, the defaults operation displays only those attributes that have defaultvalues.

```
wscp> DataSource defaults{Description {}} {ConfigProperties {}} {ConnTimeout 180}{DefaultPassword
{}} {DefaultUser {}} {DisableAutoConnectionCleanupFalse} {IdleTimeout 1800} {JNDIName {}}
{MaxPoolSize 10} {MinPoolSize1} {OrphanTimeout 1800} {StatementCacheSize 100}
```

The -all option displays all attributes (those that have default values aswell as those that are not set). The following example displays allattributes of the DataSource object type. Even though unset attributesdo not have an initial setting, they are displayed as having the string valueAttributeNotSet.

```
wscp> DataSource defaults -all{Name AttributeNotSet} {FullName AttributeNotSet} {Description
{}}{ConfigProperties {}} {ConnTimeout 180} {DatabaseName AttributeNotSet}{DefaultPassword {}}
{DefaultUser {}} {DisableAutoConnectionCleanupFalse} {IdleTimeout 1800} {JNDIName {}} {MaxPoolSize
10} {MinPoolSize1} {OrphanTimeout 1800} {StatementCacheSize 100}
```

The -attribute option is used to display the default value for one or morespecified attributes. Its argument must be a Tcl list. If thespecified attribute has a default value, the value is displayed. Thefollowing example displays the default values of the MinPoolSize andMaxPoolSize attributes of DataSource objects:

```
wscp> DataSource defaults -attribute {MaxPoolSize MinPoolSize}{MaxPoolSize 10} {MinPoolSize 1}
```

Modifying an object

The **wscp** modify operation sets the value of one or moreattributes. If a value already exists for an attribute, that value isreplaced. The syntax is as follows:

```
object_type modify  object_name -attribute  attr_list
```

The arguments are as follows:

- object_type. Specifies the object type of theinstance.
- object_name. Specifies the object instance whoseattributes are to be modified.
- -attribute attr_list. Specifies a Tcl list ofattribute-value pairs to set.

The following example shows how to modify the value of the PingIntervalattribute of an application server:

```
wscp> ApplicationServer show /Node:dev-pc/ApplicationServer:myServer/  \-attribute PingInterval
{PingInterval 60}
```

```
wscp> ApplicationServer modify /Node:dev-pc/ApplicationServer:myServer/ \-attribute {{PingInterval
120}}
```

```
wscp> ApplicationServer show /Node:dev-pc/ApplicationServer:myServer/  \-attribute
PingInterval{PingInterval 120}
```

Modifying an Environment attribute (modEnv procedure)

The following Tcl procedure, modEnv, can be used to modify the Environmentattribute of one or more application servers in a domain. The value ofthe Environment attribute is a list of strings of the form*name*=*value*. This procedure modifies a specifiedelement of the list (or adds the element if it is not present) and retains thevalues of the other elements. The procedure takes threearguments: a server name, an environment variable name, and the value towhich the variable is to be set.

The procedure does the following:

1. Sets a Tcl variable named oldEnv to the existing value of the Environmentattribute.
2. Searches the variable names in oldEnv for the variable name supplied as anargument to the command. If the supplied variable name is not inoldEnv, the procedure appends the supplied variable and value to oldEnv in theform name=value, and sets the variable newEnv to the value of oldEnv.If the supplied variable name is in oldEnv, the procedure replaces thevariable's value with the supplied value and sets newEnv to the value ofoldEnv.
3. Modifies the server's Environment attribute so that it is equal tothe value of newEnv.

```
## modEnv - procedure for modifying the Environment attribute of one or# more application servers in
a domain.  The specified environment variable# is modified (or added if it is not present), and the
values of other # variables are retained.## Arguments:## server - the fully qualified name of the
application server whose# Environment attribute is to be modified.## variable - the name of the
environment variable to modify.## value - the new value of the environment variable.## To modify the
Environment attribute of multiple servers, use# the Tcl foreach command, for example# wscp> foreach
server [ApplicationServer list] {modEnv $server TEST_VARIABLE 3.5}## The file init.tcl must be
loaded prior to using this procedure.  proc modEnv {server variable value} {    set oldEnv {}
getAttrs $server attr Environment    if {[info exists attr(Environment)]} {set oldEnv
$attr(Environment)}    # append to environment if not found; replace if it is found    set i
[lsearch -regexp $oldEnv ^$variable=]    if {$i == -1} { set newEnv [lappend oldEnv
"$variable=$value"]    } else { set newEnv [lreplace $oldEnv $i $i "$variable=$value"]    }    set
attr(Environment) $newEnv    setAttrs $server attr}modEnv
```

The following example shows the use of the modEnv procedure to modify thevalue of the PATH variable in two application servers in a domain (or to addthe PATH variable if it does not exist). The existing value of theEnvironment attribute of each server is as follows:

```
wscp> ApplicationServer show $serv1 -attribute {Environment}{Environment PATH=/myPath}wscp>
ApplicationServer show $serv2 -attribute {Environment}{Environment {{OTHERVARIABLE=word1 word2}}}
```

The following calls to the modEnv procedure modify each Environmentattribute. The resulting change is shown for the two exampleservers:

```
wscp> modEnv $serv1 PATH /revisedPathwscp> ApplicationServer show $serv1 -attribute
{Environment}{Environment PATH=/revisedPath}wscp> modEnv $serv2 PATH /revisedPathwscp>
ApplicationServer show $serv2 -attribute {Environment}{Environment {{OTHERVARIABLE=word1 word2}
PATH=/revisedPath}}
```

The modEnv procedure can be used to change the Environment attribute ofmultiple servers of the same type. In the following example, the Tcl**foreach** command is used to call the modEnv procedure on eachApplicationServer instance in a domain. The value TEST_VARIABLE=1 isadded (or appended) to the attribute as needed.

```
wscp> foreach server [ApplicationServer list] {modEnv $server TEST_VARIABLE 1}
```

6.6.0.2.2.4.3: Starting and stopping live repository objects

The **wscp** interface can be used to start and stop any liverepository object. Live repository objects include the following:

- Nodes
- Application servers
- Generic servers
- Enterprise applications
- Modules
- Server groups

The syntax for the start operation and stop operation is as follows:

```
object_type start  object_name object_type stop  object_name
```

The following example starts the application server named myServer:

```
wscp> ApplicationServer start /Node:dev-pc/ApplicationServer:myServer/
```

6.6.0.2.2.4.4: Displaying information about objects

The following examples include **wscp** operations and custom Tclprocedures for displaying information about objects:

- Listing objects
- Querying (displaying) attributes
- Printing an object's attributes
- Viewing the containment hierarchy
- Displaying select attributes

Listing objects

The **wscp** list operation lists all instances of an object typeor, optionally, only those instances that meet the specified criteria.The syntax is as follows:

```
object_type list  [-constraint attr_list]  [-recursive]
```

The arguments are as follows:

- -constraint attr_list. Specifies a Tcl list of attributesto use as a constraint. Only objects with the specified attribute-valuepairs are listed.
- -recursive. Lists all instances of any object type that belongs tothe containment hierarchy of the specified type.

The following example command lists all instances of the ApplicationServerobject type in the domain:

```
wscp> ApplicationServer list{/Node:dev-pc/ApplicationServer:Default Server/}
{/Node:dev-pc/ApplicationServer:AcctServer1/} {/Node:dev-pc/ApplicationServer:AppServerGroup1/}
{/Node:dev-pc/ApplicationServer:AppServerGroup2/} {/Node:dev-pc/ApplicationServer:CustServer1/}
```

The -recursive option lists all instances of any object type that belongsto the containment hierarchy of the specified type. TFor instance, fora JDBCDriver object type, the -recursive option lists all instances ofDataSource.

```
wscp> JDBCDriver list -recursive/JDBCDriver:OracleDrv/ /JDBCDriver:OracleDrv/DataSource:OracleDB/
/JDBCDriver:OracleDrv/DataSource:AppDb1/ /JDBCDriver:DB2Drv/ /JDBCDriver:DB2Drv/DataSource:WAS/
/JDBCDriver:DB2Drv/DataSource:AppDb2/ /JDBCDriver:DB2Drv/DataSource:SampleDB/
```

The following example lists only those application servers whosePingInterval attribute has 60 as its value:

```
wscp> ApplicationServer list -constraint {{PingInterval
60}}{/Node:dev-pc/ApplicationServer:AcctServer1/} {/Node:dev-pc/ApplicationServer:CustServer1/}
{/Node:dev-pc/ApplicationServer:AppServer2/}
```

Querying (displaying) attributes

The **wscp** show operation displays the values of all attributes ora specified subset of attributes for an object instance. The syntax isas follows:

```
object_type show object_name [-all] [-attribute attr_list]
```

The arguments are as follows:

- object_type. Specifies the object type of theinstance.
- object_name. Specifies the object instance whoseattributes are to be displayed.
- -all. Displays the values of all attributes (those that are set aswell as those that are not set). See Working with the default values of attributes for details on attributes.
- -attribute attr_list. Specifies a Tcl list of attributesto display.

The following show operation displays all attributes of a node namedws2. (The variable $node is set to the fully qualified name of thenode.)

```
wscp> Node show $node -all{Name ws2} {FullName /Node:ws2/} {CurrentState Running}{DesiredState
Running} {StartTime 988812380570} {HostNamewssol2.transarc.ibm.com} {HostSystemType sparc}
{ProcessId 10266}{InstallRoot /opt/WebSphere/AppServer} {PathMap
{WSCP0008I:EditorNotDefinedForThisProperty}}
```

The following show operation displays the values of specific attributes(the Name and CurrentState attributes) of an application server:

```
wscp> ApplicationServer show /Node:dev-pc/ApplicationServer:myServer/ \-attribute {Name
CurrentState}
```

```
{Name myServer} {CurrentState running}
```

The **wscp** show operation is implemented somewhat differently forapplication servers because several attributes have very long values.By default, the ApplicationServer show operation displays all applicationserver attributes *except* the JVMConfig, ORBConfig, andWebContainerConfig attributes, which have very long multipart values.However, **wscp** does provide two ways to display the values of theseattributes:

- By using the -attribute flag to explicitly display their values.For example, the following show operation displays the value of the JVMConfigattribute:

  ```
  wscp> ApplicationServer show /Node:node1/ApplicationServer:myServ/ -attribute {JVMConfig}{JVMConfig
  {{JvmPropertiesArray {}} {AdditionalCommandLineArgs {}}{BootClasspathAppend {}}
  {BootClasspathPrepend {}}{BootClasspathReplace {}} {Classpaths {}} {DebugMode false}{DebugString {}}
  ```

```
    {DisableJIT false} {HProfArgs {}} {InitialHeapSize0} {MaxHeapSize 0} {RunHProf false}
    {SystemProperties {}}{VerboseGC false} {VerboseJNI false} {GeneratedCommandLineArgs {}}}}
```

- By using the ApplicationServer showall operation to display the values ofall application server attributes, regardless of their length.

The show and showAttrs operations perform different functions. Allobject types have a show operation for displaying attributes. Servergroup objects have, in addition to the show operation, the showAttrsoperation. For these objects, the show operation displays theattributes associated with the server group--for example, the IfStartedand StartTime attributes. The showAttrs operation displays the defaultattributes of clones associated with the server group. (Theseattributes match the properties for the application server resource.)

The following examples illustrate output for a show and showAttrsoperation. The object ServGrp1 is a server group created for anapplication server:

```
# show commandwscp> ServerGroup show /ServerGroup:ServGrp1/{Name ServGrp1} {FullName
/ServerGroup:ServGrp1/} {StartTime 0}{IfStarted False} {EJBServerAttributes {{Name
AttributeNotSet}{FullName AttributeNotSet} {CurrentState Stopped} {DesiredStateStopped} {StartTime
0} {ProcessId 0} {Environment {}} {Executablejava} {ExecutableActive java} {CommandLineArgs
{}}{CommandLineArgsActive {}} {EnvironmentActive {}} {UserId {}}{UserIdActive {}} {GroupId {}}
{GroupIdActive {}} {WorkingDirectory{}} {WorkingDirectoryActive {}} {Umask 18} {UmaskActive 18}
{Stdin {}}{StdinActive {}} {Stdout /tmp/Broker.stdout} {StdoutActive/tmp/Broker.stdout} {Stderr
/tmp/Broker.stderr} {StderrActive/tmp/Broker.stderr} {MaxStartupAttempts 2} {ProcessPriority
20}{ProcessPriorityActive 20} {PingInterval 60} {PingTimeout 200}{PingInitialTimeout 300} {TraceSpec
{}} {SystemProperties {}}{ThreadPoolConfig {MinimumSize 10} {MaximumSize 50} {InactivityTimeout10}
{IsGrowable false}} {Transports {{{Protocol http} {Host *} {Port9080} {SSLConfig {}} {MaxKeepAlive
25} {MaxReqKeepAlive 100}{KeepAliveTimeout 5} {ConnectionTimeout 5} {BacklogConnections
50}{HttpProperties {}} {SSLEnabled false}}}} {DynamicCacheConfig{CacheSize 1000} {Enabled false}
{CacheGroups {}}}}} {ModuleVisibility3} {ModuleVisibilityActive 3} {UseDomainQualifiedUserNames
False}{UseDomainQualifiedUserNamesActive False} {DefaultDataSource {}}#showAttrs commandwscp>
ServerGroup showAttrs /ServerGroup:ServGrp1/{CurrentState Stopped} {DesiredState Stopped} {StartTime
0} {ProcessId0} {Environment {}} {Executable java} {ExecutableActive java}{CommandLineArgs {}}
{CommandLineArgsActive {}} {EnvironmentActive {}}{UserId {}} {UserIdActive {}} {GroupId {}}
{GroupIdActive {}}{WorkingDirectory {}} {WorkingDirectoryActive {}} {Umask 18}{UmaskActive 18}
{Stdin {}} {StdinActive {}} {Stdout/tmp/Broker.stdout} {StdoutActive /tmp/Broker.stdout}
{Stderr/tmp/Broker.stderr} {StderrActive /tmp/Broker.stderr}{MaxStartupAttempts 2} {ProcessPriority
20} {ProcessPriorityActive 20}{PingInterval 60} {PingTimeout 200} {PingInitialTimeout 300}{TraceSpec
{}} {SystemProperties {}} {ThreadPoolConfig {MinimumSize10} {MaximumSize 50} {InactivityTimeout 10}
{IsGrowable false}}{Transports {{{Protocol http} {Host *} {Port 9080} {SSLConfig {}}{MaxKeepAlive
25} {MaxReqKeepAlive 100} {KeepAliveTimeout 5}{ConnectionTimeout 5} {BacklogConnections 50}
{HttpProperties {}}{SSLEnabled false}}}} {DynamicCacheConfig {CacheSize 1000} {Enabledfalse}
{CacheGroups {}}}}} {ModuleVisibility 3}{ModuleVisibilityActive 3} {UseDomainQualifiedUserNames
False}{UseDomainQualifiedUserNamesActive False} {DefaultDataSource {}}
```

The following custom procedure, display, displays the output of the showoperation in a readable format--one attribute per line. Theprocedure's arguments are an object type and the name of an objectinstance.

```
## display - a procedure for displaying attributes in a readable format.## Arguments:## type - the
object type whose attributes are to be displayed.# # name - the fully-qualified name of the object
instance whose attributes # are to be displayed.# proc display {type name} {        set attrs [$type
show $name]    foreach attr $attrs { puts $attr    }}display
```

The following example demonstrates output from the displayprocedure. The procedure is used display the attributes of a JDBCDriverobject named DB2_Drv:

```
wscp> display JDBCDriver /JDBCDriver:DB2_Drv/Name DB2_DrvFullName /JDBCDriver:DB2_Drv/Description
nullImplClass COM.ibm.db2.jdbc.DB2ConnectionPoolDataSource
```

This procedure is available in 6.6.0.2.2.5: Sample Tcl procedures and scripts.

Printing an object's attributes

The custom Tcl procedure printAttributes uses the attributes operation toprint all or a subset of attributes for one or more object types.

- If no arguments are specified, the procedure prints the names of allattributes for all object types.
- If one or more options of the attributes operation are specified (forexample, if the -required and -readOnly options are specified), the commandprints only those groups of attributes.
- If one or more object types are specified, the command prints theattributes for the specified object types.

```
## printAttributes - prints the attributes for any objects specified or # for all objects if no
objects are specified.## Arguments:# # options - a list of options to control the types of
attributes printed. # Every option in the list of option names must be a valid option to the # wscp
attributes command (for example, -required or -cloneOnly)# AND the option name must begin with a
"-".## objects - a list of objects whose attributes are to be printed.# # The file init.tcl must be
loaded prior to using this procedure.# proc printAttributes {{options all} {objects all} args} {
global OBJECTS    if {[string first "-" $options] != 0} {       set objects $options    set options
""    }    if {[string compare $objects "all"] == 0} {set objects $OBJECTS}    if {$args != ""} {
foreach arg $args {         lappend objects $arg        }    }     foreach o $objects {          set
cmd [concat $o attributes $options] puts "# $cmd"   set result [eval $cmd]  puts $result
}}printAttributes
```

The following example commands demonstrate the use of the printAttributesprocedure. The first example prints only the attributes associated withclones of

ApplicationServer objects.

```
wscp> printAttributes -cloneOnly ApplicationServer# ApplicationServer attributes -cloneOnlyName
FullName
```

The following command prints all required attributes for all objecttypes:

```
wscp> printAttributes -required# JDBCDriver attributes -requiredName ImplClass# VirtualHost
attributes -requiredName AliasList # ApplicationServer attributes -requiredName# DataSource
attributes -requiredName# EnterpriseApp attributes -requiredName OrigEarFile OrigNodeName#
GenericServer attributes -requiredName Executable# J2CConnectionFactory attributes -requiredName#
J2CResourceAdapter attributes -requiredName ArchiveFile# JMSConnectionFactory attributes
-requiredName ConnectionType ExternalJNDIName# JMSDestination attributes -requiredName
ExternalJNDIName DestinationType# JMSProvider attributes -requiredName ExternalInitialContextFactory
ExternalProviderUR# MailSession attributes -requiredName MailTransportHost# Module attributes
-requiredName ModuleType RelativeURI ModuleTypeActive# Node attributes -requiredName# ServerGroup
attributes -requiredName# URL attributes -requiredName Spec# URLProvider attributes -requiredName
Protocol StreamHandlerClassName
```

The following command prints, for ApplicationServer and EnterpriseAppobjects, only those attributes that are required and that are specified atstartup:

```
wscp> printAttributes {-required -startUp} {ApplicationServer EnterpriseApp}# ApplicationServer
attributes -required -startUpName Environment CommandLineArgs UserId GroupId WorkingDirectory
UmaskStdin Stdout Stderr MaxStartupAttempts ProcessPriority TraceSpecSystemProperties TraceOutput
LogFileSpec DebugEnabled SourcePathOLTEnabled OLTServerHost OLTServerPort IsAClone
SecurityEnabledCacheConfig WebContainerConfig ModuleVisibilityUseDomainQualifiedUserNames
DefaultDataSource# EnterpriseApp attributes -required -startUpName OrigEarFile OrigNodeName Bindings
```

This procedure is available in 6.6.0.2.2.5: Sample Tcl procedures and scripts.

Viewing the containment hierarchy

The custom procedure printContainment can be used to display thecontainment hierarchy of a single object type or the entire object typehierarchy. The procedure takes zero or more arguments. Argumentsare object types. If no arguments are supplied, the procedure printsthe containment hierarchy for all object types. If one or morearguments are supplied, the procedure prints the containment hierarchy for thespecified object types only. Prior to using the procedure, you mustalso load the init.tcl file.

```
## printContainment - a procedure that prints the containment hierarchy for # one or more object
types. # # The script init.tcl must be loaded prior to using this procedure.## Arguments:## objects
- one or more object types whose containment hierarchies are to be printed.## The init.tcl file must
be loaded prior to using this procedure.# proc printContainment {{objects all} args} {    global
OBJECTS    if {$objects == "all"} {set objects $OBJECTS}        if {"$args" != "" } {        foreach
elem $args {        lappend objects $elem       }       }     foreach o $objects {      set cmd
[concat $o containment] puts "# $cmd"   set result [eval $cmd]  puts $result    }}printContainment
```

The following example command demonstrates the use of the printContainmentprocedure to display the containment hierarchy of ApplicationServer andJDBCDriver objects:

```
wscp> printContainment {ApplicationServer JDBCDriver}# ApplicationServer containmentNode
ApplicationServer# JDBCDriver containmentJDBCDriver
```

To print the containment heirarchy of all objects, issue theprintContainment command without any options.

The printContainment procedure is available in 6.6.0.2.2.5: Sample Tcl procedures and scripts.

Displaying select attributes

The following custom procedure, showServerStatus, displays the status (thevalue of the CurrentState attribute) of all application servers in adomain. The procedure can be customized to display additionalattributes or attributes of other objects.

```
## showServerStatus - a procedure for displaying the value of the# Name and CurrentState attribute
of all application servers # in a domain.# proc showServerStatus {} {                puts "\nStatus
of servers in the domain:\n"            foreach ejbserver [ApplicationServer list] {    puts
[ApplicationServer show $ejbserver -attribute {Name CurrentState}] }}showServerStatus
```

The following example demonstrates output of the showServerStatusprocedure:

```
wscp> showServerStatusStatus of servers in the domain:{Name {Default Server}} {CurrentState
{Initialization Failed}}{Name {Appl Server1}} {CurrentState Running}{Name {ServerGroup1}}
{CurrentState Stopped}{Name {ServerGroup2}} {CurrentState Stopped}{Name {Appl Server2}}
{CurrentState Running}{Name {My Server}} {CurrentState Stopped}{Name {Test Server}} {CurrentState
Stopped}
```

The display procedure is available in 6.6.0.2.2.5: Sample Tcl procedures and scripts.

6.6.0.2.2.4.5: Removing objects and applications

The **wscp** remove operation removes objects (and all references tothose objects) from the domain. *A live repository object must bestopped before being removed.* The syntax of the remove operationis as follows:

```
object_type remove  object_name  [-recursive]
```

If the object to be removed contains other objects, removal fails unlessyou use the -recursive option. The -recursive option removes allinstances of any object type that belongs to the containment hierarchy of theobject instance being removed. For example, if a node instance isremoved recursively, any object instance belonging to the containmenthierarchy of the node object is also removed. The recursion takes placein a downward direction (removal of object instances *below* thespecified object in the hierarchy).

The following example command recursively removes components of anapplication server named AppServ:

```
wscp> ApplicationServer remove /Node:AppNode/ApplicationServer:AppServ/ -recursive
```

6.6.0.2.2.4.6: Importing and exporting a configuration by using XMLConfig

The XML Configuration Management Tool (XMLConfig) can be invoked fromwithin **wscp**. The syntax is as follows:

```
wscp> XMLConfig export file_name [-partial file_name] wscp> XMLConfig import file_name [-substitute list]
```

For an export, specify the name of an XML output file for thefile_name argument. Specify the name of an XML data file forthe -partial option. For an import, specify the name of an XML datafile for the file_name argument. Specify a list ofvariable-value pairs for the -substitute option as follows:

```
{{variable1 value1}{variable2 value2}}
```

If a value string contains spaces, it must also be enclosed in braces({}).

6.6.0.2.2.4.7: Creating and installing drivers and data sources

The following example **wscp** commands create and install a JDBCdriver and data source. A JDBC provider and data source must beconfigured for each brand and version of database from which applicationservers or enterprise applications require connections.

The script init.tcl must be loaded prior to using these commands(init.tcl initializes NODE, DB2_HOME, and other variables usedhere).

```
wscp> JDBCDriver create /JDBCDriver:DB2Driver/ -attribute \{{ImplClass
com.ibm.db2.jdbc.app.DB2ConnectionPoolDataSource}} # Create a DataSource object wscp> DataSource
create /JDBCDriver:DB2Driver/DataSource:testDataSource/ # Install the JDBCDriver object wscp>
JDBCDriver install /JDBCDriver:DB2Driver/ -node $NODE \ -jarFile
${DB2_HOME}${FILE_SEPARATOR}java${FILE_SEPARATOR}db2java.zip
```

The following example commands remove the DataSource object and uninstallthe JDBCDriver object:

```
# Remove the DataSource object just created    wscp> DataSource remove
/JDBCDriver:DB2Driver/DataSource:testDataSource/    # Uninstall the JDBCDriver object    wscp>
JDBCDriver uninstall /JDBCDriver:DB2Driver/ -node $NODE    # Remove the JDBCDriver object just
created    wscp> JDBCDriver remove /JDBCDriver:DB2Driver/
```

6.6.0.2.2.4.8: Creating an enterprise application

This section discusses the **wscp** operations for deployingenterprise applications and modules.

- Administering enterprise applications describes how to install, start, and stop enterpriseapplications by using the **wscp** EnterpriseApp operations.
- Administering modules describes how to install, start, and stop modules by usingthe **wscp** Module operations. Modules can be deployedindependently of the enterprise applications into which they arepackaged.

Administering enterprise applications

An enterprise application is a collection of resources (such as XML files,enterprise beans, servlets, HTML files, and JSP files) that work together toperform a business function. An application server, combined with a Webserver, makes the enterprise application available to users. Enterpriseapplications are installed and configured as a single unit.

The **wscp** EnterpriseApp commands allow you to perform variousoperations on enterprise applications, including the following tasks:

- Install an enterprise application from an enterprise archive (EAR)file. In addition to installing the application, you can optionallyspecify the default application server on which its modules run, assignsecurity roles, specify JNDI mappings, and specify data sources for enterprisebeans.
- Start and stop instances of an enterprise application.
- List the modules that compose an enterprise application.
- List the nodes onto which an enterprise application is deployed.

The **wscp** EnterpriseApp install operation installs an enterpriseapplication from an EAR file. The syntax of this operation is:

`EnterpriseApp install` *`nodeName earFileName`* `[`*`other options`*`]`

where:

- *nodeName* -- The fully-qualified name of the WebSphereApplication Server node onto which the application is being installed.
- *earFileName*-- The full path name of the EAR filecontaining the application.
- *other options*-- Represents the following enterpriseapplication attributes that can optionally be set at installation:
    - **-defappserver** *appServName*-- Installs allmodules are installed on the specified application server. Specify thefully-qualified name of the application server onto which the enterpriseapplication is installed, such as/Node:DS1/ApplicationServer:DSServ/.

      Note:

      You can specify either this option or the -moduleappservers option, but notboth.
    - **-moduleappservers** {*moduleName appServName*}--Allows enterprise application modules to be installed on different applicationservers. You must specify a Tcl list of module URI-application serverpairs, including the fully-qualified name of the application server (forexample, {shoppingCart.JAR/Node:DS1/ApplicationServer:DSAux/}).

      Note:

      You can specify either this option or the -defappserver option, but notboth.
    - **-appname** *applicationName* -- The name of theenterprise application.
    - **-userroles** {*roleName userName*}-- Defines J2EEsecurity role to user mappings. Specify a Tcl list of role name anduser name pairs, such as {{Manager Mary}}.
    - **-grouproles** {*roleName groupName*}-- Defines J2EEsecurity role to user group mappings. Specify a Tcl list of role nameand group name pairs, such as {{Manager AdminGroup}}.
    - **-specialroles** {*roleName specialGroupName*}--Defines J2EE security role to special group mappings. (Special groupsare the predefined user groups Everyone and AllAuthenticatedUsers.)Specify a Tcl list of role name and special group name pairs, such as{{Manager Everyone}}.
    - **-runasroles** {*roleName identity password*}--Defines the security roles and execution identities that enterprise beanmethods run under in the enterprise application. A password must alsobe specified, for example {Manager Mary marypwd}.
    - **-resourcereferences** *resourceRefName* Specifies howmodule resource references are mapped to JNDI names. The type ofmapping depends on the type of module, but the mapping is always specified asa Tcl list of resource reference-JNDI name pairs.
        - Enterprise bean resource references are mapped as {*module_URI*::*bean_name*::*resource_ref_nameJNDIName*}
        - Web archive (WAR) resource references are mapped as{*module_URI::resource_ref_name*}. The referencesin WAR modules do not require bean names.
    - **-ejbnames**{*module_URI*::*bean_nameJNDIName*}-- Specifies the JNDI names for enterprise beans as aTcl list of pairs, where *module_URI* is the name of the JAR filecontaining the enterprise bean module. This option is only valid forJAR modules.
    - **-modvirtualhosts** {*module_URIvirtual_host_name*}-- Specifies the virtual host mapping for a WARmodules as a Tcl list of pairs, where *module_URI* is the name of theWAR file containing the web archive module. Enter the virtual hostalias for *virtual_host_name* (such as default_host) not thefully-qualified **wscp** virtual host name (such as/VirtualHost:default_host/)
    - **-ejbreferences**--Specifies the JNDI names for enterprisebean references. The type of mapping depends on the type of module, butthe mapping is always specified as a Tcl list of enterprise beanreference-JNDI name pairs.
        - Enterprise bean references are of the form {*module_URI*::*bean_name*::*ejb_ref_nameJNDIName*}
        - Web archive references are of the form{*module_URI::ejb_ref_name*}. The references inWAR modules do not require bean names.
    - **-ejbdatasources** {*module_URIJNDI_name*}--Sets the JNDI name of the default data source forJAR files as a Tcl list of pairs, where *module_URI* is the name ofthe JAR file. This option is only valid for JAR files.
    - **-cmpdatasources** {module_URI::bean_nameJNDIName}--Specifies the data source for entity beans withcontainer-managed persistence (CMP) as a Tcl list of pairs, where*module_URI* is the name of the JAR file.
    - **-redeploy**--Forces the archive to be installed, regardlessof whether it has been deployed before.
    - **-dbname** *name* --Specifies the name of theenterprise application database. It can only be used when an enterpriseapplication is first installed or when an application is reinstalled using the**-redeploy** option.
    - **-schemaname** *schema* -- Specifies the databaseschema to be used in the enterprise application database. It can onlybe used when an enterprise application is first installed or when anapplication is reinstalled using the **-redeploy** option.
    - **-dbtype** *type* -- Specifies the type ofapplication database, where *type* is one of the following supporteddatabases:
        - DB2UDBWIN_V72
        - DB2UDBOS390_V6
        - DB2UDBAS400_V4R5
        - INFORMIX_V92
        - MSSQLSERVER_V7
        - ORACLE_V8
        - SQL92
        - SQL99
        - SYBASE_V1192
        - MYSQL_V323

      It can only be used when an enterprise application is first installed orwhen an application is reinstalled using the **-redeploy**option.

To view a complete list of enterprise application attributes, use theEnterpriseApp attributes command. See Querying (displaying) attributes for more information on displaying object attributes.

The following command example shows how to install an enterpriseapplication. The node under which the application runs is DS1; thedefault application server under which its modules are installed isDSserv; and the security role mappings assign all authenticated users tothe Client role and the site administrator to the Administrator role.

```
wscp> EnterpriseApp install /Node:DS1/ C:/drugstore/DrugStoreApp.ear -defappserver /Node:DS1/ApplicationServer:DSserv/-userroles
{{Administrator SiteAdmin} {Supervisor Mary}} -specialroles {Client AllAuthenticatedUsers}
```

The following command example starts an instance of the DrugStoreenterprise application, then waits 60 seconds to see if it reallystarted. If the application does not start within this time period, theEnterpriseApp start operation is assumed to have failed.

```
wscp> EnterpriseApp start /EnterpriseApp:DrugStore/ -wait 60
```

The following command example stops an instance of the DrugStore enterpriseapplication:

```
wscp> EnterpriseApp stop /EnterpriseApp:DrugStore/
```

The following command example lists the modules that comprise an enterpriseapplication:

```
wscp> EnterpriseApp listmodules
/EnterpriseApp:DrugStore//Node:DS1/ApplicationServer:DSserv/Module:ShoppingCart//Node:DS1/ApplicationServer:DSserv/Module:StoreFront//Node:DS1/ApplicationServer:DSserv/Module:Accounts//Node:DS2/ApplicationServer:DSserv/Module:InventoryMgr/
```

The following command example lists the nodes onto which an enterpriseapplication is deployed:

```
wscp> EnterpriseApp listnodes /EnterpriseApp:DrugStore//Node:DS1/ /Node:DS2/
```

Several Tcl scripts for installing enterprise applications areavailable. Each installs one of the sample applications provided withWebSphere Application Server.

- t1.tcl installs jmsample.ear
- t2.tcl installs perfServletApp.ear
- t3.tcl installs sampleApp.ear
- t4.tcl installs Samples.ear
- t5.tcl installs ServletCacheMonitor.ear
- t6.tcl installs soapsamples.ear
- t7.tcl installs TradeSample.ear

The following example shows the script t1, which is used to install thesample enterprise application jmsample.

```
set mynode xxxx set instdir i:/WebSphere/AppServer/installableApps/ set earfile jmsample.ear set appname "MailSampleApp" set
sname "Default Server" set mailth "AHost" set mailSessName "DefaultMailSession" set vhostName "default_host"   # create
prerequiste objects  # # first, the MailSession set mailtransportattr [list MailTransportHost $mailth] set attributelist [list
$mailtransportattr]   MailSession create /MailSession:$mailSessName/ -attribute $attributelist   # next, the virtual host set
aliaslist [list *:80 *:9080] set aliasattr [list AliasList $aliaslist] set attributelist [list $aliasattr]   VirtualHost create
/VirtualHost:$vhostName/ -attribute $attributelist   # Now install the application # #     construct -modvirtualhosts option set
modhost1 [list mtcomps.war $vhostName] set modhosts  [list $modhost1]   #     construct -resourcereferences option set resref1
[list mtcomps.war::mail/MailSession9 mail/$mailSessName] set resref2 [list deplmtest.jar::MailEJBObject::mail/MailSession9
mail/$mailSessName] set resrefs [list $resref1 $resref2]     EnterpriseApp install /Node:$mynode/ $instdir$earfile -appname
$appname -defappserver /Node:$mynode/ApplicationServer:$sname/ -modvirtualhosts $modhosts -resourcereferences $resrefs
```

This script (and the others listed in this section) is available in 6.6.0.2.2.5: Sample Tcl procedures and scripts.

Administering modules

The resources that compose an enterprise application are grouped accordingto function into modules. Enterprise applications can contain threetypes of modules:

- Enterprise JavaBeans (EJB) modules, which contain Java class files forenterprise beans, Java class files for functionality that is not included withthe J2EE platform, and an EJB deployment descriptor.
- Web modules, which contain Java class files for servlets and applets;JSP files and their helper classes; static HTML, sound, image, video, andother content files; and a Web deployment descriptor.
- Application client modules, which contain the Java classes that implementthe client and an application deployment descriptor.

The **wscp** Module operations allow you to deploy modulesindependently from enterprise applications. Application client and EJBmodules are installed from Java archive (JAR) files; Web modules areinstalled from Web archive (WAR) files. When you install a module,**wscp** actually generates a simple EAR file with the specified JAR orWAR file as its only module. It then installs the EAR file. Youcan optionally assign such things as security roles, specify JNDI mappings,and specify data sources for enterprise beans.

The **wscp** Module install operation installs an enterpriseapplication module. The syntax of this operation is:

`Module install nodeName module_name-moduleappservers module_URI appServName [other options]`

where:

- *nodeName* -- The fully-qualified name of the WebSphereApplication Server node onto which the module is being installed.
- *module_name*-- The full path name of the file containingthe module.
- **-moduleappservers** {*module_URI appServName*}--Specifies the application server onto which a modules is installed.Specify the file name of the JAR or WAR file and the fully-qualified name ofthe application server, such as {shoppingCart.JAR/Node:DS1/ApplicationServer:DSAux/}.
- *other options*-- Represents the following module attributesthat can optionally be set at installation:
  - **--contextroot** *contextroot*--The contextroot for a WAR file.
  - **-appname** *applicationName* -- The name of theenterprise application to which the module belongs.
  - **-userroles** {*roleName userName*}-- Defines J2EEsecurity role to user mappings. Specify a Tcl list of role name anduser name pairs, such as {{Manager Mary}}.
  - **-grouproles** {*roleName groupName*}-- Defines J2EEsecurity role to user group mappings. Specify a Tcl list of role nameand group name pairs, such as {{Manager AdminGroup}}.
  - **-specialroles** {*roleName specialGroupName*}--Defines J2EE security role to special group mappings. (Special groupsare the predefined user groups Everyone and AllAuthenticatedUsers.)Specify a Tcl list of role name and special group name pairs, such as{{Manager Everyone}}.
  - **-runasroles** {*roleName identity password*}--Defines the security roles and execution identities that enterprise beanmethods run under in the enterprise application. A password must alsobe specified, for example {Manager Mary marypwd}.
  - **-resourcereferences** *resourceRefName* Specifies howmodule resource references are mapped to JNDI names. The type ofmapping depends on the type of module, but the mapping is always specified asa Tcl list of resource reference-JNDI name pairs.
    - Enterprise bean resource references are mapped as {*module_URI*::*bean_name*::*resource_ref_nameJNDIName*}
    - Web archive (WAR) resource references are mapped as{*module_URI::resource_ref_name*}. The referencesin WAR modules do not require bean names.
  - **-ejbnames**{*module_URI*::*bean_nameJNDIName*}-- Specifies the JNDI names for enterprise beans as aTcl list of pairs, where *module_URI* is the name of the JAR filecontaining the enterprise bean module. This option is only valid forJAR modules.
  - **-ejbreferences**--Specifies the JNDI names for enterprisebean references. The type of mapping depends on the type of module, butthe mapping is always specified as a Tcl list of enterprise beanreference-JNDI name pairs.
    - Enterprise bean references are of the form {*module_URI*::*bean_name*::*ejb_ref_nameJNDIName*}
    - Web archive references are of the form{*module_URI::ejb_ref_name*}. The references inWAR modules do not require bean names.
  - **-ejbdatasources** {*module_URIJNDI_name*}--Sets the JNDI name of the default data source forJAR files as a Tcl list of pairs, where *module_URI* is the name ofthe JAR file. This option is only valid for JAR files.
  - **-cmpdatasources** {module_URI::bean_nameJNDIName}--Specifies the data source for entity beans withcontainer-managed persistence (CMP) as a Tcl list of pairs, where*module_URI* is the name of the JAR file.
  - **-dbname** *name* --Specifies the name of thedatabase.
  - **-schemaname** *schema* -- Specifies the databaseschema to be used.
  - **-dbtype** *type* -- Specifies the type of database,where *type* is one of the following supported databases:
    - DB2UDBWIN_V72
    - DB2UDBOS390_V6
    - DB2UDBAS400_V4R5
    - INFORMIX_V92
    - MSSQLSERVER_V7
    - ORACLE_V8
    - SQL92
    - SQL99
    - SYBASE_V1192
    - MYSQL_V323
- **-defappserver** *appServName*-- Installs allmodules are installed on the specified application server. Specify thefully-qualified name of the application server onto which the enterpriseapplication is installed, such as/Node:DS1/ApplicationServer:DSServ/.

  Note:

  You can specify either this option or the -moduleappservers option, but notboth.
- **-moduleappservers** {*moduleName appServName*}--Allows enterprise application modules to be installed on different applicationservers. You must specify a Tcl list of module URI-application serverpairs, including the fully-qualified name of the application server (forexample, {shoppingCart.JAR/Node:DS1/ApplicationServer:DSAux/}).

  Note:

  You can specify either this option or the -defappserver option, but notboth.
- **-appname** *applicationName* -- The name of theenterprise application.

To view a complete list of enterprise application attributes, use theEnterpriseApp attributes command. See Querying (displaying) attributes for more information on displaying object attributes.

The following command example shows how to install a Web module. Themodule type and active module type are web; the relative URI of the WARfile containing the module is storefront.war; the context root is/WebApp.

```
wscp> Module install /Node:DS1/ C:/storefront.war -contextroot /WebApp
```

6.6.0.2.2.4.10: Configuring server groups and clones

The **wscp** ServerGroup operations allow you to configure servergroups and clones. Server groups are templates for creating clones ofapplication servers. Clients see the clones associated with a servergroup as a single application server image. The clones of anapplication server automatically participate in workload management.

The **wscp** ServerGroup operations included are as follows:

- Creating a server group from an existing application server instance
- Starting and stopping a server group and its clones
- Adding a clone to a server group

See Managing workloads for information on servergroups, clones, and workload management.

The following example creates a server group from an existing instance ofan application server, specifying its name, server selection policy, and thevalue of an environment variable:

```
wscp> ServerGroup create /ServerGroup:EjbServerGroup/ -baseInstance /ApplicationServer:EjbAppServ/
\-serverGroupAttrs {{EJBServerAttributes {SelectionPolicy roundrobin}} {Environment {VAR1 0}}}
```

The following example starts a server group after a delay of 30seconds:

```
wscp> ServerGroup start /ServerGroup:EjbServerGroup/ -wait 30
```

The following example stops a server group, using the -force option to stopit regardless of any other conditions:

```
wscp > ServerGroup stop /ServerGroup:EjbServerGroup/ -force
```

The following example adds a clone to the server groupEjbServerGroup. The clone runs on node Server1:

```
wscp > ServerGroup clone /ServerGroup:EjbServerGroup/ -cloneAttrs{{Name Clone1}} -node
/Node:Server1/
```

The fully qualified name of the new application server instance is/Node:Server1/ApplicationServer:Clone1/.

The following example lists all clones that are associated with a servergroup:

```
wscp > ServerGroup listClones /ServerGroup:EjbServerGroup/
/Node:Server1/ApplicationServer:EjbServerGroup1/ /Node:Server1/ApplicationServer:EjbServerGroup2/
```

6.6.0.2.2.4.11: Administering EIS connections

The **wscp** J2CResourceAdapter and J2CConnectionFactory operationsallow you to manage connections to enterprise information system (EIS) backendsystems, such as PeopleSoft or Customer Information Control System(CICS). These operations can be used to perform the followingtasks:

- Create a J2C resource adapter by using the J2CResourceAdapter createcommand. A J2C resource adapter is a module that enables connections toa specific back-end system.
- Install a J2C resource adapter on a node by using the J2CResourceAdapterinstall command. You can install a J2C resource adapter on multiplenodes.
- Create an instance of a J2C connection factory by using theJ2CConnectionFactory create command. A J2C connection factory is a setof connection configuration values that are used by the J2C connection poolmanager to specify how applications connect to a backend system.

The following command example creates a J2C resource adapter. TheArchiveFile attribute is required.

```
wscp> J2CResourceAdapter create /J2CResourceAdapter:ResAd/  -attributes {{ArchiveFile
C:/apps/resources.rar}}
```

The following command example installs a J2C resource adapter. Thenode and resource archive (RAR) file attributes are required.

```
wscp> J2CResourceAdapter install /J2CResourceAdapter:myRes/ -node /Node:Appserv1/ -rarfile
"C:/apps/resources.rar"
```

The following command example creates a J2C connection factoryobject. You must specify the name of the J2C resource adapter withwhich the connection factory is associated with. The optionalattributes specify the timeout period in milliseconds and the maximum numberof connections supported by that connection factory.

```
wscp> J2CConnectionFactory create
/J2CResourceAdapter:ResAd/J2CConnectionFactory:factory1/-attributes {{ConnectionTimeout 1000}
{MaxConnections 10}}
```

For more information on administering J2C connections, see Administering J2C related administrative objects.

6.6.0.2.2.4.12: Administering the Java Message Service (JMS)

The Java Messaging Service (JMS) enables applications to exchange data inthe form of messages. The **wscp** operations JMSProvider,JMSConnectionFactory and JMSDestination enable you to configure JMS clientsfrom the command line as follows:

- Create a JMS provider by using the JMSProvider create command. AJMS provider implements the JMS messaging interfaces.
- Install a JMS provider on a node by using the JMSProvider installcommand. You can install JMS providers on multiple nodes.
- Create an instance of a JMS connection factory by using theJMSConnectionFactory create command. A client uses a JMS connectionfactory object to connect to the messaging system.
- Create an instance of a JMS destination by using the JMSDestination createcommand. A client uses a JMS destination object to specify the targetof the messages it sends or the source of the messages it receives.

The following example command creates a JMS provider. TheExternalInitialContextFactory and ExternalProviderURL attributes arerequired.

```
wscp> JMSProvider create /JMSProvider:Prov1/ -attribute{{ExternalInitialContextFactory
com.ibm.websphere.naming.WsnInitialContextFactory}{ExternalProviderURL iiop://localhost}}
```

The following example command installs a JMS provider on the nodedev1:

```
wscp> JMSProvider install /JMSProvider:Prov1/ -node /Node:dev1/ -jarFile
"/opt/WebSphere/ApplicationServer/jars/JMS.jar"
```

The following example command creates an instance of a JMS connectionfactory. The ExternalJNDIName attribute is required.

```
wscp> JMSConnectionFactory create /JMSProvider:Prov1/JMSConnectionFactory:connect1/ -attribute
{{ExternalJNDIName jms/connect1}}
```

The following example command creates an instance of a JMSdestination. The ExternalJNDIName attribute is required.

```
wscp> JMSDestination create /JMSProvider:Prov1/JMSDestination:dest1/ -attribute {{ExternalJNDIName
jms/dest1}}
```

For more information, see Administering messaging and JMSproviders.

6.6.0.2.2.4.13: Administering JavaMail sessions

The **wscp** MailSession operations are used to manage JavaMailsessions. JavaMail enables applications to compose, send and receiveelectronic mail.

The following example command shows how to create a new JavaMail sessionthat uses the SMTP mail protocol. The session name andMailTransportHost attributes are required.

```
wscp> MailSession create /MailSession:Session1/ -attribute {{MailTransportHost mailhost}
{MailTransportProtocol smtp}}
```

The following example command shows how to destroy a JavaMailsession:

```
wscp> MailSession remove /MailSession:Session1/
```

6.6.0.2.2.4.14: Administering URL providers and URLs

The **wscp** URL and URLProvider operations allow you to create andinstall instances of URLs and URL providers.

The following example command creates an instance of a URL provider.The name, Protocol and StreamHandlerClass attributes are required.

```
wscp> URLProvider create /URLProvider:Prov1/ -attribute {{Protocol http}{StreamHandlerClass
com.ibm.ejs.myStreamHandler}}
```

The following example command installs a URL provider on a node:

```
wscp> URLProvider install /URLProvider:AppProv/ -node /Node:AppServ1/ -jarFile
"C:/jars/provider.jar"
```

The following example command creates an instance of a URL. You mustspecify a URL provider for the URL; the Spec attribute is required:

```
wscp> URL create /URLProvider:Prov1/URL:testURL/ -attribute {{Spec mySpec}}
```

6.6.0.2.2.5: Sample Tcl procedures and scripts

The following files contain example Tcl procedures and scripts.

- attrs.tcl script. Procedure for printingthe attributes of all or selected objects.
- contain.tcl script. Procedure fordisplaying the containment hierarchy for all or select objects.
- disp.tcl script. Procedure fordisplaying the output of the show operation in a readable format--oneattribute per line.
- init.tcl script. Many of the examplescripts and procedures require that the file init.tcl be loaded priorto running. This file initializes several variables used elsewhere andcontains useful Tcl procedures for accessing attributes, obtaining errorstatus information, and other tasks.
- modEnv.tcl script. Procedure formodifying or replacing values in the Environment attribute.
- servInfo.tcl script. Procedure fordisplaying the name and current state of all servers in a domain.
- The following scripts install the example enterprise applications providedwith WebSphere Application Server:
    - t1.tcl script. Procedure for installingthe sample application jmsample.ear.
    - t2.tcl script. Procedure for installingthe sample application perfServletApp.ear.
    - t3.tcl script. Procedure for installingthe sample application sampleApp.ear.
    - t4.tcl script. Procedure for installingthe sample application Samples.ear.
    - t5.tcl script. Procedure for installingthe sample application ServletCacheMonitor.ear.
    - t6.tcl script. Procedure for installingthe sample application soapsamples.ear.
    - t7.tcl script. Procedure for installingthe sample application TradeSample.ear.

6.6.0.2.2.6: Migrating wscp scripts from version 3.5.x to version 4.0

This section summarizes some of the changes to **wscp** in version4.0 that can affect existing **wscp** tcl scripts. It isintended as a guide for determining whether these scripts need to be updatedand is not an exhaustive description of **wscp** changes.

You need to retest all scripts created under earlier versions of**wscp** before migrating them to a production environment that runsthe latest version of the software.

Discontinued objects

The following objects are no longer supported in **wscp**.

- EJBContainer
- EnterpriseBean
- Servlet
- ServletEngine
- ServletRedirector
- SessionManager
- WebApplication
- WebResource
- UserProfile

Although they can no longer be administered on an individual basis,enteprise beans, servlets, and Web applications that have been packaged intoenterprise applications or modules can be installed, started, and stopped withthe EnterpriseApp and Module operations. See 6.6.0.2.2.4.8: Creating an enterprise application.

Renamed objects and functional or syntax changes

The following objects have been renamed or their functionality and syntaxhave changed. See the listed articles for more information on their newsyntax and options.

Old name	**New name**	**Functional changes**
EnterpriseApplication	EnterpriseApp	Application components are now packaged in modules. You no longerhave to explicitly install all components of an enterprise application.See 6.6.0.2.2.4.8: Creating an enterprise application.
Model	ServerGroup	Can only create server groups and clones of application servers.See 6.6.0.2.2.4.10: Configuring server groups and clones.
DataSource	DataSource	Syntax changes. See 6.6.0.2.2.4.7: Creating and installing drivers and data sources.

New objects and new functionality

The following objects are new for version 4.0 and represent areas offunctionality that can now be handled in scripts. See the listedarticles for more information on these objects and services.

New objects and services	New functional areas	Documentation
Module	Java 2 Enterprise Edition (J2EE) modules	6.6.0.2.2.4.8: Creating an enterprise application
J2CConnectionFactory	Enterprise information systems (EIS) connectivity	6.6.0.2.2.4.11: Administering EIS connections
J2CResourceAdapter		
JMSProvider	Java Message Service (JMS)	6.6.0.2.2.4.12: Administering the Java Message Service (JMS)
JMSConnectionFactory		
JMSDestination		
URL	URL	6.6.0.2.2.4.14: Administering URL providers and URLs
URL provider		
PmiService	Performance data	6.6.0.2.2.3.10: Monitoring performance
SecurityConfig	Global security settings	6.6.0.2.2.3.11: Setting global security defaults
SecurityRoleAssignment	J2EE security roles	6.6.0.2.2.3.12: Managing security roles

6.6.0.2.2.7: Using the wscpCommand interface

Use the**com.ibm.ejs.sm.ejscp.wscpcommand.WscpCommand**interface to embed **wscp** operations in Java applications.This enables applications to evaluate **wscp** operations withoutrepeated startup costs. The interface is linked through the product_installation_root/lib/wscp.jar file.

Constructors

`WscpCommand(String node, String port)`

where:

- *node*--The name of the WebSphere Application Server node onwhich the **wscp** operation is executed (such as localhost).
- *port*--The port number of the administrative server.

`WscpCommand()`

The default values for this constructor are localhost for node and 900 forport (the default port number of the administrative server).

Public methods

`WscpResult evalCommand(String command)`

This method evaluates a **wscp** operation and returns theresults. Use the same syntax as you do for interactive or scripted**wscp** operations. Using abbreviated command names is notrecommended.

The results of the **evalCommand** method are encoded in a**com.ibm.ejs.sm.ejscp.wscpcommand.WscpResult**object. The following methods on this object can be used to evaluatethe returned value:

- **toString**-- Returns a string representation of the resultsof the wscp operation.

  Note:

  > The WscpCommand interface has no way of knowing the format of the expectedoutput from a **wscp** operation. When the return format is notan attribute-value pair or list, use the **toString** method toevaluate the output. The application programmer must then write code toparse the output.

- **success**-- Returns a boolean value indicating whether thewscp operation was successful.
- **listToVector**-- If the expected result of the wscpoperation is a list, use this method to return a vector containing the listelements.
- **attribPairsToVector**-- If the expected result of the wscpoperation is a list of attribute-value pairs, use this method to return avector containing the list. Each element of the vector is anattribute-value pair.
- **attribPairsToHashTable**-- If the expected result of thewscpoperation is a set of attribute-value pairs, use this method to return a hashtable containing the values keyed on the attributes.

`String getErrorInfo()`

This method returns the status of an **evalCommand** method whoseresults are returned in a WscpResult object.

- If the **wscp** operation specified in the **evalCommand**executed successfully, the **getErrorInfo** method returns an emptystring.
- If the **wscp** operation did not execute successfully, the**getErrorInfo** method returns the exception information that wasreceived from **wscp**.

The following is an example of how to use the **evalCommand** and**getErrorInfo** methods.

```
String cmd = "ApplicationServer list";WscpResult results = wscpCommand.evalCommand(cmd);if
(!results.success()) {   System.out.println("command failed; exception information: " +
results.getErrorInfo());   }
```

Utility class

The **WscpCommand** interface contains a utility class,**WscpQualifiedName**, that allows the manipulation of fully qualfiedwscp object names. The constructor is as follows:

`WscpQualifiedName(fullyqualifiedname)`

where *fullyqualifiedname* is the fully-qualified **wscp**name of an object.

The methods for this class are as follows:

- **getName**(*level*)--Gets the object name at thespecified containment level.
- **getObject**(*level*)--Gets the object type at thespecified containment level.
- **toString**--Returns a string representation of the objectname.
- **numberOfLevels**--Returns the number of containmentlevels.

The **WscpQualifiedName** class contains the following constants,which define the containment levels at which various components arenamed.

- ENTERPRISE_APPLICATION
- DATASOURCE
- JDBC_DRIVER
- NODE
- SERVER_GROUP
- VIRTUAL_HOST
- APPLICATION_SERVER
- GENERIC_SERVER
- WEB_RESOURCE
- WEB_APPLICATION
- MODULE
- JMS_PROVIDER
- JMS_CONNECTION_FACTORY
- JMS_LISTENER
- J2C_RESOURCE_FACTORY
- J2C_CONNECTOR
- MAIL_SESSION
- URL
- URL_PROVIDER

An example of how this class is used is as follows:

```
String name = "{/Node:MyNode/ApplicationServer:MyAppServer/}"WscpQualfiedName qName = new
WscpQualifiedName(name);qName.numberOfLevels() returns 2qName.getObject(2) returns
"ApplicationServer"qName.getName(2) returns
"MyAppServer"qName.getName(WscpQualifiedName.APPLICATION_SERVER)       returns "MyAppServer"
```

Security

The **wscp** command-line tool runs with security enabled because itis started by using SAS from the command line, as you can see in thewscp.bat (Windows) and wscp.sh (Unix) files. However, the**WscpCommand** interface cannot guarantee support for a server withsecurity enabled because it is used in client programs. If the Javaprogram is started using SAS, **WscpCommand** methods execute withsecurity enabled.

6.6.0.4: Overview of editing property files by hand

In general, it is recommended that you use the administrative toolsto edit configuration (property) files, rather than editing them directly. Here are references to the main opportunities for editing product property files by hand:

- Administrative configuration file
- Dynamic caching feature

6.6.0.5: Using the Application Assembly Tool interface

The Application Assembly Tool interface includes the following:

- A main window for creating, viewing, and editing the content and assembly properties of Web modules, EJB modules, application client modules, and J2EE applications.
- Property dialog boxes for creating new objects and for accessing and modifying properties for a module. The tool uses the property values to automatically generate deployment descriptor files for the module.
- Wizards for quickly creating modules. Wizards gather minimum requirements for creating the module and generating the deployment descriptor. They prompt for required information and use default values where applicable.

The main Application Assembly Tool window consists of two panes: the navigation pane onthe left and the property-editing pane on the right. The navigation pane provides a viewof the contents of the module being created or edited. The property-editing pane displaysproperty dialog boxes for the component or property group selected in the opposite pane.

- The **navigation pane**. This is a hierarchical (tree) view of a module's contents, assembly properties, and files. The first object in the tree is the display name of the module or application. An icon in the tree represents each of the following:
    - Modules in the application (EJB modules, Web modules, or application client modules).
    - Components of a module (for example, entity beans and session beans for EJB modules).
    - Groups of properties for the module or component. For example, Web components have properties for security role references, initialization parameters, and page lists.
    - Files making up the module.

  The navigation pane is used to review the content and structure of a module and to select components or property groups. Selection of a component or property group allows its properties to be entered or edited in the property pane.
- The **property pane**. By default, this pane displays property dialog boxes for the item selected in the opposite pane. If the selected item is an object type (for example, entity bean), the instances of the type are displayed in the top portion of the pane. The properties for the instance are displayed on the bottom portion of the pane. When an instance is selected, the corresponding property dialog box is displayed in the bottom portion of the pane.

When the property pane is disabled, its contents are dependent on the item that isselected in the navigation pane, as follows:

- If a module folder is selected (for example, Web Modules), the names of all WAR files are displayed.
- If an archive file is selected (for example, a JAR file), the property groups for that archive are displayed.
- If an object type is selected (for example, Web Components), the instances of that object type are displayed. In this example, servlets and JSP files are displayed.
- If an object instance is selected (for example, an entity bean instance), the property groups for the instance are displayed.
- If a property group icon is selected, the property values are displayed in table form.
- If the Files icon is selected, a complete list of files for the module are displayed.

The property pane can be disabled by clicking **View->Show Property Pane**to toggle the selection.

You can adjust the width of the panes by dragging the split bar left or right. In theproperty-editing pane, you can also adjust the width of the columns of properties bydragging the edge of the column left or right.

The navigation pane

The navigation pane displays contents and associated assembly properties in an indentedtree outline. The tree hierarchy for an application (EAR file) is as follows:

- Application display name
    - EJB Modules (folder)
        - JAR files
            - Session beans
                - Session bean instances
                - Properties for session bean instances
            - Entity beans
                - Entity bean instances
                - Properties for entity bean instances
            - Security roles
            - Method permissions
            - Container transactions
            - Files
    - Web Modules (folder)
        - Web applications
            - Web components
                - Web component instances (servlets or JSP files)
                - Properties for the Web component instance
            - Properties for the Web application
            - Assembly property extensions
            - Files
    - Application Clients (folder)
        - Application client instances
        - Properties for application client instances
    - Security Roles (for the application)
    - Files

Clicking the plus (+) and minus (-) symbols (or double-clicking an item)expands and collapses the tree to reveal the content of each folder or icon. The tree canalso be navigated by using the up- and down-arrow keys to move through the branches andleaves and the left- and right-arrow keys to expand and collapse the tree. Folders andicons can be expanded to reveal their contents. For example, when the Web Modules folderis expanded, the individual Web applications (WAR files) contained in the module arerevealed. When the icon for a Web application is expanded, icons for the Web components(servlets or JSP files), assembly properties, and individual files in the application arerevealed.

In a J2EE application, the first level of the hierarchy is the display name of theapplication. The second level consists of folders, one for each type of module in theapplication and one for the security properties that apply to the application as a whole.Beneath each folder representing a type of module, the module's components are listed. Forexample, beneath the EJB modules folder, icons represent the individual JAR files in themodule. Beneath the JAR file, icons represent session beans, entity beans, assemblyproperties, and files in the JAR. Clicking the icon for entity beans reveals all entitybean instances in the module. Expanding an entity bean instance reveals icons for variouscategories of properties for the bean.

Clicking an item (an archive file, enterprise bean instance, or property icon, forexample) in the navigation pane causes its corresponding property dialog box to bedisplayed in the property pane (if properties have been set).

The property dialog boxes

The property dialog boxes allow you to define the contents of a new module, makemodifications to an existing module, or duplicate a module in full or in part by cuttingand pasting archive files and property values. This method is especially useful forassembling multiple modules with similar contents. Using a property dialog box alsoprovides access to the complete listing of properties, whereas the wizards expose only theminimum required and more-frequently used properties that must be defined for the module.Default values are used where applicable. On the property dialog box, the fields forrequired properties have a red asterisk (*) next to them. When using the property dialogbox, you can access help for any of the properties or examine the definitions of thedefault properties already provided.

A property dialog box is used to enter, view, or modify values. Each property dialogbox contains one or more of the following tabs. Clicking a tab opens a window containing agroup of properties. Note that not all tabs are present for all property groups.

- **General**. These properties represent standard J2EE deployment descriptors.
- **Icons**. These properties specify the location of JPEG or GIF files used as icons to represent the component in a GUI.
- **IBM Extensions**. The IBM deployment descriptor extensions are additions to the standard descriptors for J2EE applications, Web applications, and enterprise beans. The extensions allow you to specify properties that enable Enterprise Edition or legacy (older) systems to work in the WebSphere Application Server environment. They also allow you to specify behavior that is vendor specific, undefined in a current specification, or expected to be included in a future specification. For example, isolation level attributes determine how isolated one transaction is from another. This property can be set for individual methods in an enterprise bean or for all methods in the enterprise bean. Isolation level attributes are defined in the extensions layer.
- **Bindings**. Binding information contains references to the external resources and enterprise beans required by a module. This information is used by the container to locate a resource in the JNDI namespace. The logical names specified for binding properties are later resolved (in the administrative console) to the registered JNDI name for an enterprise bean's home interface and for resource manager connection factories. The bindings specified for a module in the Application Assembly Tool can be changed prior installing the module in an application server.

The wizards

The wizards allow you to quickly create a module with the minimum required information.Using wizards is ideal for testing and prototyping purposes. If you wish to specifyadditional properties for your module, such as extensions and binding information, use theproperty dialog boxes. (After using a wizard, you can continue to edit the properties of amodule or add new properties; this is done by using the property dialog boxes.) Theproperty dialog boxes allow you to specify values for all properties, whereas the wizardsprompt you to specify required properties and other more frequently used properties. Onwizards, the fields for required properties have a red asterisk next to them. When usingthe wizards, you can access help for any of the properties or examine the definitions ofthe default properties already provided.

6.6.0.5.1: Modifying and adding modules to applications

You can modify a module by adding or editing properties, adding components,or removing components. If you are modifying an application (EAR file),you can add new modules or import them. The steps used to modify amodule are the same regardless of the type of module. After you makechanges to a module, you must save the archive again. If the modulecontains enterprise beans, you must regenerate deployment code.

This file contains instructions for the following tasks:

- Adding or editing assembly properties
- Adding or importing modules into applications
- Copying and deleting JAR and WAR files
- Adding enterprise beans and Web components
- Adding files

Note that after deployment code has been generated for an application, thedeployable archive is renamed with the prefix `Deployed_` .Any subsequent changes to the archive from within the Application AssemblyTool are applied to the version of the archive that existed prior to codegeneration. To see changes reflected in your application, you mustregenerate deployment code and re-install the deployable archive.

Adding or editing assembly properties

To add properties or edit the existing properties of a module, perform thefollowing steps:

1. Make sure that the property pane is enabled. From the main menu,click **View**. If necessary, toggle the **Show PropertyPane** menu item so that it is checked.
2. If the module is not already opened, click**File**->**Open** and select the archive file for themodule. The contents of the module are displayed in an ApplicationAssembly Tool window.
3. In the navigation pane, click the icon for the group of properties thatyou want to view or edit. If property values exist, a property dialogbox is displayed in the property editing pane. If no property valuesexist, right-click the property icon and choose **New**. Aproperty dialog box is displayed.
4. Enter or edit values. Click **OK** to close the propertydialog box. Click **Apply** to apply the values but to keep thedialog open for additional edits. Click **Help** to view fieldhelp.

There are several alternative methods of accessing the property dialogboxes. These methods apply only to archive files (JAR, WAR, and EARfiles) or to module components (enterprise beans or JSP files, forexample).

- Right-click the icon representing the archive or component. Choose**Properties**. A property dialog box is displayed. Editthe properties and click **OK**.
- Click the icon representing the archive or component. Click theProperties icon on the tool bar. A property dialog box isdisplayed. Edit the properties and click **OK**.

Adding or importing modules into applications

You can add any combination of JAR or WAR files to an enterpriseapplication (EAR file). There are several ways to include archivefiles:

- Import an existing JAR or WAR file.

- Open an existing EJB or Web module and copy and paste individual beans orWeb components into the application.
- Create a new JAR or WAR file.

The procedure for each task is as follows:

- To import archive files into the application, right-click the iconcorresponding to the type of module (EJB module, Web module, or applicationclient module) and choose **Import**. Use the file browser tolocate and select the archive file. Click **Open**. TheJAR or WAR file is displayed under the appropriate folder. The importedmodule's properties are also imported. Click the plus sign(+) next to the archive's icon to view its contents and editits properties if needed.
- To copy portions of an existing module, open the module by clicking**File**->**Open** and selecting the file. Arrange thewindows so that both the EAR file and the source archive are visible.Use a copy-and-paste operation to copy archives or components to the currentmodule.
- To create a new JAR or WAR file to be included in the application,right-click the folder corresponding to the type of module to create (EJBModules, Web Modules, and Application Clients), and choose**New**. Enter required values as indicated. Optionally,enter values for other properties. Click **OK.** The JARor WAR file is displayed under the appropriate folder. Click the plussign (+) to verify contents and enter assembly properties.

  For EJB modules, you must add at least one enterprise bean.Right-click the folder corresponding to the type of bean to create (sessionbean or entity bean), and choose **New** or **Import**.For entity beans, choose whether you are creating a CMP or BMP bean. Aproperty dialog box is displayed. Enter required values asindicated. Optionally, enter values for other properties. Click**OK.** The enterprise bean is displayed in the navigationpane. Click the plus sign (+) to verify the contents andenter assembly properties.

Copying and deleting JAR and WAR files

To copy a JAR or WAR file from one module to another:

1. In the navigation pane of the source module, right-click the iconrepresenting the JAR or WAR file to be copied. Choose**Copy**.
2. In the navigation pane of the target module, right-click the folderrepresenting EJB modules or Web modules and choose **Paste**.Confirm the values displayed in the dialog box. If the target modulealready contains an archive with the same name, a message is displayed statingthat the archive cannot be overwritten. (You can ignore the warning andlater rename the archive and modify its properties.)

To delete a JAR or WAR file from a module:

1. In the navigation pane, right-click the JAR or WAR file. Choose**Delete**.

Adding enterprise beans and Web components

To add enterprise beans or Web components, use any of the followingmethods:

- Import the item into the module. In the navigation pane,right-click the Web modules, EJB modules, or Application Client folder andchoose **Import**. Browse the file system and locate the desiredarchive file. When the file is located, click **Open**.If you are importing an EAR file, you are prompted to select which WAR or JARfiles to import. Click the desired archive and click**OK**. As prompted, enter or verify properties for the importedarchive.
- Use a copy-and-paste operation to copy components from an existingmodule.
- Create a new item by right-clicking the appropriate icon (for example, theicon representing entity beans

or Web components) and chooseNew. Click **Browse** to locate the class files ofthe enterprise bean or Web component. By default, the root directory orarchive is the current archive. If needed, browse the file system forthe directory or archive where class files reside. After you choose adirectory or archive, its file structure is displayed. Expand thestructure and locate the files that you need. Select the file and click**OK**. In the property dialog box, click **OK**.Verify that the enterprise beans or Web components are added to the module(expand the hierarchy in the navigation pane).

Adding files

To add files, perform the following steps:

1. Right-click the Files icon and choose **Add Files**. If youare adding files to a Web module, right-click the **Class Files**,**Resource Files**, or **JAR Files** icon.
2. Select a root directory or archive. The root directory is used todetermine the relative path of the files being added. If you are addingan entire archive, select the directory that contains the archive.First, click **Browse** to navigate to the desired directory orarchive. Then select the directory and click **Select**.The directory structure is displayed in the left pane.
3. Use the left pane to navigate the contents of the directory. Theright pane displays the contents of the selected directory in the leftpane.
4. From the list of files displayed in the right pane, select the desiredfiles, and click **Add**. If you select a directory and click**Add**, all files in the directory, including the directory, areadded. Relative path names are maintained. To select multiplefiles, press and hold the Control key while selecting the files. To adda range of files, select the initial file, press and hold the Shift key, andthen select the last file in the range.
5. The files to be added are displayed in the Selected Files pane.When the Selected Files pane contains the correct set of files, click**OK**.

6.6.0.5.2: Generating deployment code for modules

Before installing your application in WebSphere Application Server, youmust generate deployment code for the application. This step isrequired for EJB modules and for any EAR files that contain EJBmodules. During code generation, the Application Assembly Tool invokesthe Deployment Tool for Enterprise JavaBeans (ejbdeploy) to do thefollowing:

1. Generate and compile container implementation classes.
2. Process the implementation classes by using the RMIC compiler, whichgenerates the client-side stub and server-side skeletons needed for remotemethod invocation (RMI). (You can bypass this step by selecting theCode Generation Only checkbox on the property dialog box.)
3. Validate the JAR file and display error messages if there are any J2EEspecification violations within the JAR file.
4. For CMP entity beans, generate persistence code.

For CMP entity beans, if the JAR file contains a map and schema document,that schema is used. If the JAR file does not contain a map and schemadocument, the Application Assembly Tool uses a top-down mapping to generatethe following files:

- META-INF\Table.ddl
- META-INF\map.mapxmi
- META-INF\Schema\schema.rdbxmi

Note:

For Advanced Single Server Edition of this product, only the Table.ddlfile is generated.

The user must create the database table manually. The generatedTable.ddl file can be used as a guide, or the user can create the tableby some other means.

If you do not wish to use the default top-down mapping, the recommendedtool is VisualAge for Java. For information on default mappings, seethe documentation for Deployment Tool for Enterprise JavaBeans.

To migrate CMP entity beans from WebSphere Application Server 3.5for use in version 4.0, use the -35 option of the Deployment Tool forEnterprise JavaBeans. This option is recommended only if you need topreserve use of the mapping rules that were used in the 3.5 version ofthe Deployment Tool.

To generate deployment code for a module in the Application Assembly Tool,perform the following steps:

Note:

A working directory is used to save temporary files while creating thedeployed archive. On UNIX systems, if there is insufficient space inthis directory, a segmentation violation error can occur. Make surethat there is sufficient space in the working directory. Required spacevaries depending on the application. By default, the working directoryis `/tmp` on UNIX systems. You can specify a differentdirectory in the Working directory field on the property dialog box.

1. Click **File**->**Open** and select the module. Themodule is displayed in the main window.
2. Right-click the name of the module at the top of the navigation pane andchoose **Generate code for deployment**. A property dialog boxis displayed. Enter or edit values for 6.6.35.0: Assembly properties for generating deployment code. By default, the newly created archive file has thesame name as the current archive file with the prefix`Deployed_`. Click **Generate Now**.
3. View status messages in the window and resolve any errors reported.
4. To close the window, click **Close**.

If the modules in an application are already deployed, the ApplicationAssembly Tool regenerates deployment code for them.

Note: Ensure that your JAR files do not contain source code(.java files). On certain platforms, JAR files containing sourcecode can cause compilation errors during deployment code generation.

6.6.0.5.3: Verifying archives

Verifying archive files is important for successful generation ofdeployment code for the application. During verification, theApplication Assembly Tool checks that an archive is complete, and thatdeployment descriptor properties and references contain appropriatevalues. Specifically, the verification process checks thefollowing:

- Required deployment properties must contain values.
- Values specified for environment entries must match their associated Javatypes.
- In both EAR and WAR files:
  - For EJB references, the target enterprise bean of the link mustexist.
  - For security role references, the target role must exist.
  - Security roles must be unique.
- For EAR files, each module listed in the deployment descriptor must existin the archive.
- For WAR files, the files for icons, servlets, error, and welcome pageslisted in the deployment descriptor must have corresponding files in thearchive.
- For EJB modules:
  - All class files referenced in the deployment descriptor must exist in theJAR file.
  - Method signatures for enterprise bean home, remote, and implementationclasses must be compliant with the EJB 1.1 specification.

To verify a module in the Application Assembly Tool:

1. Click **File**->**Open** and select the module to beverified.
2. Right-click the name of the module at the top of the navigation paneand choose **Verify**.
3. In the **Verify** window, click **Verify**. The tooldisplays a scrolling window for viewing status messages as the verificationproceeds.

6.6.0.5.4: Viewing deployment descriptors

You can view the deployment descriptor for a J2EE application or for anymodule contained in the application. To view a deploymentdescriptor:

1. Click **File**->**Open** and select the module whosedeployment descriptor file is to be viewed.
2. In the navigation pane, right-click the name of the EAR file to view theapplication-level deployment descriptor. Right-click the name of an EJBmodule or Web module to view a module-level deployment descriptor.Choose **View Descriptor**. The tool displays a window forviewing the XML file.
3. Click **Close** to close the window.
4. If necessary, modify the values of the deployment descriptor by editingthe module's properties and then saving the archive. Repeat thisprocedure to view your modifications.

6.6.0.5.5: Specifying JNDI bindings and resolving references

Binding information is used by a container to locate external resourcessuch as enterprise beans and resource connection factories required by anapplication. You can specify the JNDI names of resources when youinstall an application by using the administrative console.Alternatively, you can specify the JNDI names in the Application AssemblyTool. Binding properties specified in the tool can be changed in theadministrative console prior to installing an application. You canspecify binding information for the following:

- Enterprise beans.
- Data sources to be used by entity beans.
- References to enterprise beans and resource manager connectionfactories. For example, an application client, enterprise bean, orservlet can be a client to another enterprise bean. The referencingbean or servlet declares an EJB reference and optionally binds thereference.
- Virtual host names for Web applications.

If you are using the Application Assembly Tool to create application clientmodules, you must also use the Application Client Resource Configuration Toolif you have local resources. This tool allows you to define referencesto local resources (other than enterprise beans) on the machine where theapplication client resides.

References to enterprise beans, resource manager connection factories, andenvironment entries are made available in the JNDI naming contextjava:comp/env. References are relative to this context.The references can then be bound to the actual JNDI name in the targetoperational environment.

To specify binding information for an enterprise bean:

1. In the navigation pane, click the icon representing the enterprise beaninstance. The properties for the enterprise bean are automaticallydisplayed in the property pane.
2. In the property pane, click the **Bindings** tab.
3. Enter the JNDI name. This is the name to which the bean's homeinterface is bound in the global JNDI namespace. As a convention, usethe fully qualified JNDI name. For example,`com/ibm/ejb/samples/ClaimHome`. You use this name when youbind EJB references to the bean from the referencing client.
4. Click **Apply**.
5. Save the archive.

To declare a reference and optionally bind the reference:

1. In the navigation pane, locate the property icon representing thereference (EJB reference, resource reference, or environment entry).
2. Right-click the icon and click **New**. The property dialogbox is displayed. (If property values already exist for the item, theproperty dialog box is automatically displayed in the property pane.)
3. On the **General** tab, enter required properties. Forexample, for EJB references, enter a name, home interface, remote interface,and enterprise bean type. If you enter the name`ejb/ClaimHome`, the client looks up the bean's home interfaceby using the JNDI name `java:comp/env/ejb/ClaimHome`.View the help for 6.6.43.0.1: Assembly properties for EJB references, 6.6.43.0.2 Assembly properties for resource references, and 6.6.34.0.a: Assembly properties for environment entries.

   If a referenced enterprise bean resides in the current module, or in theencompassing application, use the Link property to select the name of thebean. The binding information can be omitted.
4. Click the **Bindings** tab.

5. Enter the global JNDI name of the item being referenced. As aconvention, use the fully qualified JNDI name. For the examplereference `ejb/ClaimHome`, enter`com/ibm/ejb/samples/ClaimHome`. This name must match the JNDIname specified for the ClaimHome bean on its **Bindings** propertydialog box.
6. Click **Apply**.
7. Save the archive.

Some additional notes on binding:

- For CMP entity beans, you can optionally specify the JNDI name of the datasource of the bean.
- Optionally, you can specify a default data source and defaultauthorization information to be used by all entity beans in an EJBmodule. To do this, click the EJB Modules folder to display itsproperty dialog box. Select the **Bindings** tab. Enterthe appropriate values.

6.6.0.5.6: Converting EJB JAR files from 1.0 to 1.1 format

The Application Assembly Tool can convert an EJB 1.0 (serialized)deployment descriptor to an EJB 1.1 XML deployment descriptor.However, some code changes are required to make your EJB 1.0application comply with the J2EE specification.

To convert a JAR file containing enterprise beans that conform to the1.0 EJB specification, do the following:

1. Click **File**->**Open** to read the JAR file into theApplication Assembly Tool. You are prompted whether to make theconversion.
2. Save the JAR file by clicking **File**->**Save As**.Specify a new name for the JAR file. An EJB module containing theconverted JAR file is automatically created.

The resulting JAR file contains the following .xml and .xmifiles instead of the original .ser files. Old information ismapped to the XML format of the EJB 1.1 specification.

- META-INF/MANIFEST.MF
- META-INF/ejb-jar.xml
- META-INF/ibm-ejb-jar-ext.xmi
- META-INF/ibm-ejb-jar-bnd.xmi

6.6.0.5a: Starting and stopping Application Assembly Tool

The Application Assembly Tool can be started from the command line or canbe launched from within the administrative console. On Windows NTsystems, the tool can also be launched from the Start menu.

- To start the tool from the command line, run the assembly.bat file(on Windows NT systems) or assembly.sh on UNIX systems. Thescript is located in the directory where WebSphere Application Server isinstalled. For example, issue the following command on Windows NTsystems:

  ```
  C:\ websphere\appserver\bin> assembly
  ```

  The administrative server does not need to be running to launch or usethe tool from the command line. If the tool cannot be started from thecommand line, verify that the assembly.bat file is in your PATHenvironment variable or specify the fully qualified path to the script.
- To start the tool from within the administrative console, choose**Tools**->**Application Assembly Tool**.

To close the Application Assembly Tool, click**File**->**Exit**.

6.6.0.7: Launching Java application clients in the J2EE application client container

The J2EE specification requires support for a client container that runs standalone Java applications (known as J2EE application clients) and provides J2EE services to theapplications. J2EE services include naming, security, and resource connections.

Before using the tool

You are ready to run your client application using this tool after you have:

1. Written the client application program
2. Assembled and installed an application module (.ear file) in the application server runtime

Invoking the launchClient tool

J2EE application clients are launched using the launchClient shell, located in:

`product_installation_root/bin/launchClient.bat`

This command instantiates an instance of the `launchClient` class.

Application client runtime behavior

The `launchClient` batch command starts the application client runtime, which:

- Initializes the client runtime
- Loads the class that you designatedas the `main` class with the application assembly tool
- Runs the `main` method of the client application program

When your program terminates, the application client runtime cleans up the environment and the Java virtual machine ends.

Passing parameters

You have to pass parameters to the `launchClient` command. You can pass parameters to your client application program as well. The `launchClient` commandallows you to do both.

The `launchClient` command requires thatthe first parameter is either:

- an `Ear` file specifying the client application to launch
- a request for `launchClient` usage information

All other parameters intended for the `launchClient` command must begin with the **`-CC`** prefix.

Parameters that are not Ear files, or usage requests, or that begin with the **`-CC`** prefix, are ignored by the application client runtime, and are passed directly to the client application program.

For more information on the `launchClient` parameters, refer to the`main` method in the `launchClient` javadoc.

`launchClient` retrieves parameters from three places:

1. The command line
2. A properties file
3. System properties

The parameters are resolved in the order listed above, with command line values having the highest priority and system properties the lowest. This prioritization allows you to set and override default values.

Specifying the server name

By default, the `launchClient` command uses the environment variable COMPUTERNAME for the **`BootstrapHost`** property value. This setting is effective for testing your client application when it is installed on the same computer as the server. However, in other cases you need to override this value with the name of your server.

You can override the **`BootstrapHost`** value by invoking `launchClient` with thefollowing parameters:

```
launchClient myapp.ear -CCBootstrapHost=abc.midwest.mycompany.com
```

You can also override the default by specifying the value in a properties file and passingthe file name to `launchClient`.

Running a client application with security enabled

Security is controlled by the server. You do not need to configure security onthe client because the client assumes that security is enabled. If security is notenabled, the server ignores the security request, and the client applicationworks as expected.

Format of the launchClient properties file

You can store launchClient values in a properties file, a good method for distributing default values. A knowledgeable user can then override one or more values on the command line.

The format of the file is one launchClient-CC parameter per line without the -CC prefix. For example:

```
verbose=trueclasspath=c:\mydir\util.jar;c:\mydir\harness.jar;c:\production\G19\global.jarBootstrapHost=abc.westcoast.mycompany.comtracefile=c:\WebSphere\mylog.txt
```

Command syntax reference

You can obtain additional help on this command by invoking it without arguments. Following is the result of one such invocation. The arguments are subject to change. For the definitive arguments and syntax, query the program.

```
IBM WebSphere Application Server, Release 4.0J2EE Application Client Tool, Version 1.0Copyright IBM Corp., 1997-2001WSCL0012I:
Processing command line arguments.Usage: launchClient [<userapp.ear> | -help | -?] [-CC<name>=<value>] [app
args]where:<userapp.ear>     = The path and name of the Ear file containing the client application.-help, -?               =
Print this help message.where the -CC properties are for use by the Application Client Runtime:-CCverbose              =
<true|false> Use this option to display additional                          informational messages. The default is false.
-CCclasspath            = A classpath value. When an application is launched,                          the system classpath is
not used. If you need to                          access classes that are not in the ear file or part                          of
the resource classpaths, specify the appropriate                          classpath here. Multiple paths may be concatenated.
-CCjar                  = The name of the client jar file within the EAR file                          that contains the
application you wish to launch.                          This argument is only necessary when you have
multiple client jar files in the ear file.                          -CCBootstrapHost        = The name of the host server you
wish to connect to                          initially. Format: your.server.ofchoice.com-CCBootstrapPort        = The server port
number. If not specified, the                          WebSphere default value is used.-CCinitonly            = <true|false>
This option is intended for ActiveX                          applications to initialize the Application Client
runtime without launching the client application.                          The default is false.
-CCtrace                = <true|false> Use this option to have WebSphere write                          debug trace information
to a file. You may need this                          information when reporting a problem to IBM Service.
The default is false.                          -CCtracefile            = The name of the file to write trace information.
The default is to output to the console.                          -CCpropfile            = Name of a Properties file containing
launchClient                          properties. In the file, specify the properties                          without the -CC
prefix. For example: verbose=true.                          where "app args" are used by the client application and are ignored
byWebSphere.
```

6.6.0.8: Expanding .ear files

The EARExpander expands Ear files into the format desired by the application server runtime, as described in the application installation instructions. EARExpander can also collapse the expanded format back to a normal Ear (.jar or .zip) format.

Invoking the tool

The tool is located in the following directory:

```
product_installation_root/bin/EARExpander.bat
```

To view syntax, open a command line and invoke the tool without arguments. Hereis a typical result. The line breaks have been changed for better formatting inthis documentation.

```
C:\seaa0122.02\bin>EARExpander.bat IBM WebSphere Application Server, Release 4.0 J2EE J2EE
Application Expansion Tool, Version 1.0Copyright IBM Corp., 1997-2001Required Argument Missing:ear
Usage: java com.ibm.websphere.install.commands.EARExpander-ear -expandDir -operation
[expansionFlags]
```

ExpansionFlags indicate whether you want every JAR file expanded, or just the contained WAR files within the EAR file. The default is all.

Expanding files

The following example command expands the file my.ear into the *product_installation_root*/bin/myEAR directory:

```
EARExpander -ear my.ear -expandDir product_installation_root/bin/myEAR             -operation expand
```

Collapsing files

Using the collapse -operation reverses the format to normal.

```
EARExpander -ear my.ear -expandDir product_installation_root/bin/myEAR
-operation collapse
```

Type each of the above commands on a single line, despite their appearance in this documentation.

6.6.0.9: Application Client Resource Configuration Tool for configuring client resources

The *Application Client Resource Configuration Tool* defines the resources for the client application. These configurations are stored in the client application .ear file, and are used by the WebSphere application client runtime for resolving and creating an instance of the resources for the client application.

Launching the tool

6.6.0.9.3: Removing objects from EAR files with the Application Client Resource Configuration Tool

During this task, you will remove (delete) an object from an EARfile for your application client. You can remove any particular J2EE resource or resource provider, including data sources and data source providers; URLs and URL providers;JMS providers, connection factories, and destinations; and JavaMail sessions. You cannot removethe default JavaMail provider, however.

1. Start the tool and open the EAR file from which you want to remove an object. The EAR file contents will be displayed in a tree view.

   Recommended. If you already had the EAR file open, and have made some changes, click **File** -> **Save** to save your work before preceding to delete an object. See below for a discussion.
2. In the tree, locate the object that you want to remove.
3. Do one of the following:
   - Click the object, then click **Edit** -> **Delete** from the tool menu bar.
   - Right-click the object to display its menu, then click **Delete**.
4. If you are sure that you want the deletion to take effect, click **File** -> **Save**.

⚠ The option to delete an item does not offera confirmation dialog. As soon as you delete the item, it is gone. As a safeguard,consider saving your work right before you begin this task. That way, if you change yourmind after removing an item, you can close the EAR file without saving your changes, which willcancel your deletion. Be sure to do so immediately after the deletion, or you willalso lose any unsaved work that you performed since the deletion.

6.6.0.9a: Starting the Application Client Resource Configuration Tool and opening an EAR file

Start the graphical interface with the command:

```
clientConfig
```

To open an EAR file into the tool:

1. On the menu bar of the tool, click **File** -> **Open**.
2. Browse for the file that you want to open.
3. When you have found the file and selected it, click **Open**.

To save your changes to the file and close the tool:

1. On the menu bar of the tool, click **File** -> **Save**.
2. Click **File** -> **Exit**.

Now begin administering a resource type:

- JDBC providers and data sources
- JavaMail providers and sessions
- URL providers and URLs
- JMS providers, connection factories, and destinations

6.6.0.10: SoapEarEnabler tool

The SoapEarEnabler tool is a Java applicationthat enables a set of SOAP serviceswithin an Enterprise Application Archive(EAR).

> Start with an existing EAR file, either onecreated with the Application Assembly Tool (AAT), or a previously-created, valid, J2EE-compliant EAR.

The SoapEarEnabler guidesyou through the required steps to enable one or moreservices within an application.The SoapEarEnabler tool makes a backup copy of your original EAR in the event you need to remove or add services at a later time.

> SoapEarEnabler will **not** accept a "soap-enabled"EAR file as input.

After you enable web services, you must install the EAR in WebSphere ApplicationServer.

You will be prompted as to whether you wish to add the adminsitration client to the EAR.This is a Web-based client that will allow to list all active services for a specific context froma browser window. With this interface, you can stop and start existing services. You might choose to notadd this interface for security reasons, or you might want to secure the interface before making a service available.

For more information on securing resources, see article Securing SOAP services.

Before invoking SoapEarEnabler,first create an 4.8.2.1: Apache SOAP deployment descriptor for each service to be enabled.

Invoke the SOAP EAR enabler tool from the WebSphere ApplicationServer bin directory using the command, **SoapEarEnabler** on the Windows platform or **SoapEarEnabler.sh** on UNIX platforms.

The tool operates in two modes:

1. interactive
2. silent

Specify all requiredcommand line arguments to use the tool in ***silent*** mode.

Interactive mode

The SoapEarEnabler tool prompts you for all requiredinformation.The following dialog is an example of using the tool in interactive mode.

> In this dialog, user input is in *italics*, and tool output is in **bold**.

SoapEarEnabler (On Windows NT)
SoapEarEnabler.sh (On UNIX platforms)

Please enter the name of your ear file:
..\work\stockquote.ear
How many services would you like your applicationto contain (1...n)?
1

Now prompting for info for service #1:

> **Please enter the file name of the deploymentdescriptor xml file:**
> *..\work\StockQuoteDD.xml*
> **Is this service an EJB (y/n)?**
> *n*
> **How many jar files arerequired for this service (0...n)?**
> *1*
> **Classpath requirement #1: Please choose a file ([1] samples.jar, [2] stockquote.war):**
> *1*
> **Should this service be secured (y/n)?**
> *n*

Please enter a context root for your non-securedservices (e.g. /soap):
/soapsamples
Do you wish to install the administration client?

> **Warning! You should not install this client in a production ear unless you intend to secure the URI to it.**

Install the administration client (y= yes/n= no)?
y

Silent Mode

In silent mode, supply the argumentsin the same order as for the interactive prompts.

> In silent mode, the SOAP Admin GUI will not install.Also, you will not be prompted for the SOAP Admin GUI.

The following example describes how to to use the tool in silent mode:

```
soapearenabler [args]
 where the argumentsmust be specified in the following order:
<ear-file-name>
<number-of-services>
The following block is repeated based onthe number of services specified.
<deployment-descriptor-file-name>
<service-is-an-ejb-(y/n)>
The following argument should be suppliedonly if service-is-an-ejb-(y/n) is y.
<ejb-jar-file-uri-(already-in-ear)>
<number-of-additional-jar-files(0, 1, 2...)>
The following argument is repeated number-of-additional-jar-filestimes.
<classpath-entry-uri-(already-in-ear)>
<secure-this-service-(y/n)>
This following argument is supplied onlyif secure-this-service-(y/n) is n for anyservice.
<context-root-for-non-secured-services,ex: /soap>
This following argument is supplied onlyif secure-this-service-(y/n) is y for anyservice.
<context-root-for-secured-services, ex:/soapsec>
```

Silent Mode examples

The following is an example of deployingone ejb as a non-secured service:

```
soapearenabler soap.ear 1          d:\xml-soap\java\samples\ejbadder\deploymentdescriptor.xmly
adderservice-ejb.jar 1 samples.jar n /soap
```

The following is an example of deployingone ejb as a non-secured service, and onejava class as a secured service:

```
soapearenabler soap.ear 2          d:\xml-soap\java\samples\ejbadder\deploymentdescriptor.xmly
adderservice-ejb.jar 1 samples.jar n
d:\xml-soap\java\samples\stockquote\deploymentdescriptor.xmln 1 samples.jar y /soap /soap-sec
```

The following is an example of deploying2 java classes as non-secured services:

```
soapearenabler soap.ear 2          d:\xml-soap\java\samples\stockquote\deploymentdescriptor.xmln 1
samples.jar n           d:\xml-soap\java\samples\addressbook\deploymentdescriptor.xmln 1 samples.jar
n /soap
```

The line breaks in the above examples have been modified for thisdocumentation. Typically, commands are issued on a single line.

6.6.0.12: About the WebSphere Resource Analyzer

This section introduces the WebSphere Resource Analyzer and contains thefollowing topics:

- Introduction to the WebSphere Resource Analyzer
- Resource categories monitored by the Resource Analyzer
- How performance data is organized
- How performance data is accessed and reported

Introduction to the WebSphere Resource Analyzer

The WebSphere Resource Analyzer is a Graphical User Interface (GUI)performance monitor for IBM WebSphere Application Server Version 4.0,Advanced Edition for Multiplatforms. You can use the Resource Analyzerto retrieve performance data from the application servers. Data iscollected continuously by the application servers and retrieved as needed fromwithin the Analyzer. You can regulate the impact incurred from datacollection by using the Resource Analyzer or the WebSphere AdvancedAdministrative Console. (See Understanding instrumentation levels for data collection.) The Resource Analyzer's graphical interfaceprovides controls that enable you to choose the particular resources andcounters to include in a view.

The Resource Analyzer provides access to a wide range of performance datafor two kinds of resources:

- Application resources (for example, enterprise beans and servlets)
- WebSphere run-time resources (for example, Java Virtual Machine (JVM)memory, application server thread pools, and database connection pools)

Performance data includes simple counters, statistical data (such as theresponse time for each method invocation of an enterprise bean), and load data(such as the average size of a database connection pool during a specifiedtime interval). This data is reported for individual resources andaggregated for multiple resources.

Depending on which aspects of performance are being measured, you can usethe Resource Analyzer to accomplish the following tasks:

- View data in real time or view historical data from log files.
- View data in chart form, allowing comparisons of one or more statisticalvalues for a given resource on the same chart. In addition, differentunits of measurement can be scaled to enable meaningful graphicdisplays.
- Record current performance data in a log and replay performance data fromprevious sessions.
- Compare data for a single resource to an aggregate (group) of resources ona single node.

Data collection and reporting incurs a cost--it can affect theperformance of your distributed applications. Although performance datais automatically *collected* at all times, it is not*reported* by default. Instead, you must explicitly enable thereporting of the data that is collected for your monitored system. TheWebSphere Advanced Administrative Console provides a dialog box for specifyingwhich data is to be collected for which resources. If data is not beingcollected for a particular resource, the data point is not registered andtherefore cannot be displayed in the Resource Analyzer. Even when datareporting is enabled, the types of data displayed depend on user-definedinstrumentation levels. See Understanding instrumentation levels for data collection.

To ensure the efficient operation of WebSphere Application Server, datacollection for the machine on which the application server resides needs to bemaintained with a minimum of overhead. To accomplish this requirement,no significant calculations are made on the server side. Instead, theResource Analyzer uses the Performance Monitoring Infrastructure (PMI) clientto hold and manipulate raw data that is reported by the server.

Resource categories monitored by the Resource Analyzer

The Analyzer collects and reports performance data for the followingresource categories (modules):

- **Enterprise beans**. Data for this category reports loadvalues, response times, and life cycle activities for enterprise beans.Examples include the average number of active beans and the number of timesbean data is loaded or written to the database. It reports informationfor enterprise bean methods, which are the remote interfaces used by anenterprise bean. Examples include the number of times a method wascalled and the average response time for the method. It also reportsinformation on the size and usage of a cache of bean objects (enterprise beanobject pools). Examples include the number of calls attempting toretrieve an object from a pool and the number of times an object was foundavailable in the pool.
- **Database connection pools**. Data for this categoryreports usage information about connection pools for a database.Examples are the average size of the connection pool (number of connections),the average number of threads waiting for a connection, the average wait timein milliseconds for a connection, and the average time the connection was inuse.
- **J2C Connectors**. Data for this category reports usageinformation about the J2EE (Java 2 Enterprise Edition) Connector Architecturethat enables enterprise beans to connect and interact with procedural back-endsystems, such as Customer Information Control System (CICS), and InformationManagement System (IMS). Examples are the number of managed connections(physical connections) and the total number of connections (connectionhandles).
- **JVM run time**. Data for this category reports memory usedby a process as reported by the JVM. Examples are the total memoryavailable and the amount of free memory for the JVM.

  JVMPI run time. In addition, the Resource Analyzer makesuse of a Java Virtual Machine Profiler Interface (JVMPI) to enable a morecomprehensive performance analysis. This profiling tool enables thecollection of information about the Java Virtual Machine (JVM) that runs theapplication server. See Enabling JVMPI data reporting.
- **Servlet session manager**. Data for this category reportsusage information for HTTP sessions. Examples include the total numberof sessions being accessed, the average amount of time it takes for a sessionto perform a request, and the average number of concurrently active HTTPsessions.
- **Thread pools**. Data for this category reports informationabout the pool of Object Request Broker (ORB) threads that an applicationserver uses to process remote methods and the Web container pools that areused to process HTTP requests coming into the application server.Examples include the number of threads created and destroyed, the maximumnumber of pooled threads allowed, and the average number of active threads inthe pool.
- **Transaction manager**. Data for this category reportstransaction information for the container. Examples include the averagenumber of active transactions, the average duration of transactions, and theaverage number of methods per transaction.
- **Web applications**. Data for this category reportsinformation for the selected server. Examples include the number ofloaded servlets, the average response time for completed requests, and thenumber of requests for the servlet.

How performance data is organized

The Resource Analyzer organizes performance data in a centralized hierarchyof the following objects:

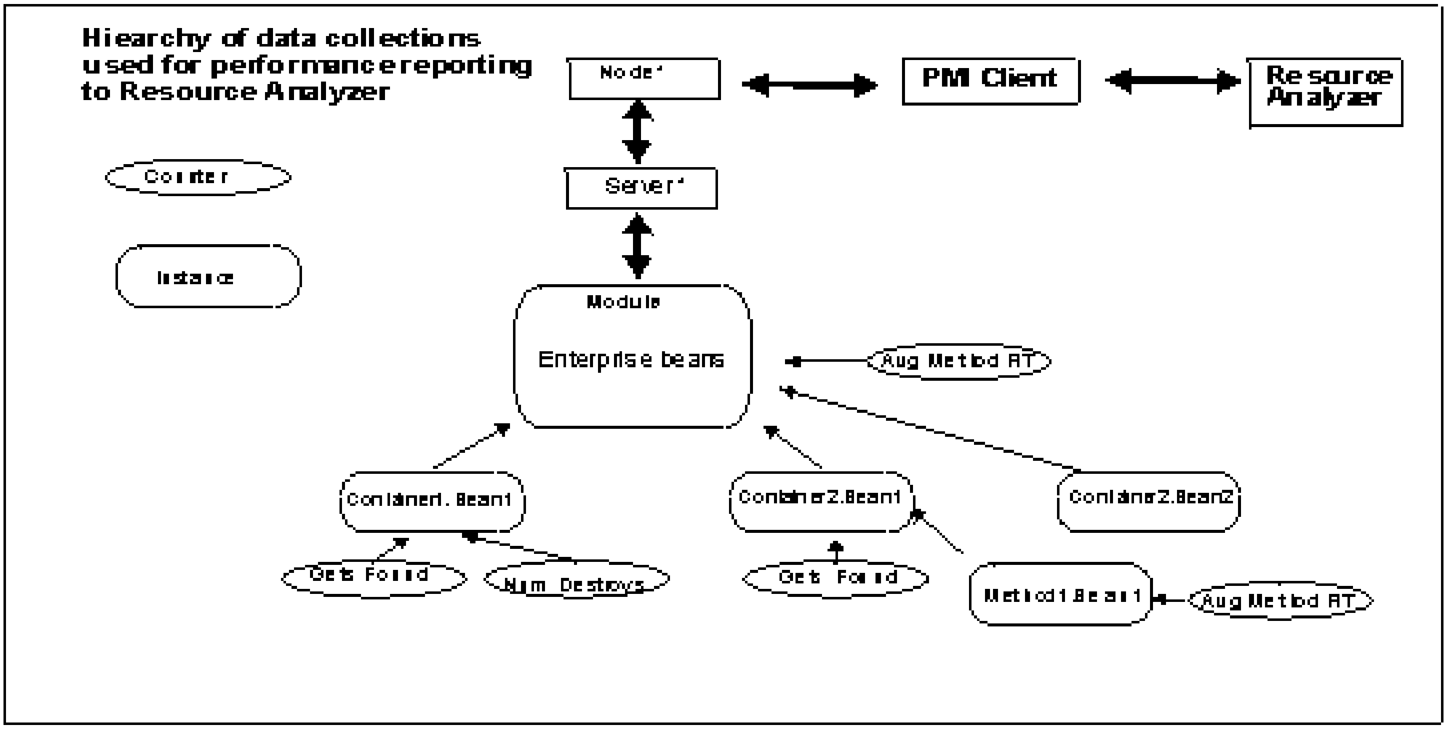
- **Node.** A node represents a physical machine in theWebSphere administrative domain. No performance data is collected forthe node itself.

- **Server.** A server is a functional unit that providesservices to clients over a network. No performance data is collectedfor the server itself.
- **Module.** A module represents one of the resourcecategories for which data is collected to be reported to the ResourceAnalyzer: enterprise beans, database connection pools, J2C Connectors,JVM run time, JVMPI run time, servlet session manager, thread pools,transaction manager, Web applications.
- **Method.** A method is software that implements a specifiedoperation within a Java class. For the Resource Analyzer, informationis collected on instance methods.
- **Servlet.** A servlet is a Java program that extends somefunctionality of the Web server. It generates dynamic data in responseto client requests. For the Resource Analyzer, information is collectedon the servlets in Web application.
- **Counter.** A counter is a data type used to holdperformance information for analysis.

Each resource category (module) has an associated set of counters.The counters track activities that can have an impact on systemperformance. Counters measure some aspect of a running system, forexample, the number of active enterprise beans, the time spent in respondingto a servlet request, or the number of kilobytes of available memory.

Modules can have instances, which are single instantiations of an objectwithin a class. Counters can be assigned to any number of modules andinstances or to no group at all. For example, the diagram in Figure 1 shows that the counter Avg Method Rt (average methodresponse time) belongs to the Enterprise Beans module, and in addition, themethod associated with the Container2.Bean1 instance also uses the AvgMethod Rt counter. The diagram in Figure 1 shows a hierarchy of data collections that are organized forreporting to the Resource Analyzer.

Figure 1. Example hierarchy of collections of performance data reported for Resource Analyzer



Each instance can have counters activated or deactivated for each module(resource category), for example, the enterprise beans category. Inaddition, each module has its own a set of counters. The modules andinstances can include or exclude any or all of the counters that areavailable, based on the object type and the instrumentation level. (SeeUnderstanding instrumentation levels for data collection.)

Instances uniquely identify objects within them to modules by using aconcatenation of the name of the instance and the name of the object.For example, Figure 1 shows instances named Container1 and Container2. Theinstances contain objects named Bean1. Each instance.object isuniquely identified by its full name: Container1.Bean1,Container2.Bean1, and Container2.Bean2.

The object Bean1 belongs in both instances, Container1 andContainer2. Container1.Bean1 and Container2.Bean1 aredistinct instances with different activities occurring within each. Theinformation that is reported for the same object in two instances will differaccording to the difference in activity occurring in each instance. Forexample, Container1.Bean1 and Container2.Bean1 both have theGets Found counter enabled. The data reported by the Gets Found counterin Container1.Bean1 reflects only the activity occurring inContainer1. Similarly, the Gets Found counter inContainer2.Bean1 only reports activity in Container2.

The information that is reported on Container1.Bean1 andContainer2.Bean1 also differs according to which counters are enabledwithin each instance. For example, the Num Destroys counter is enabledin Container1.Bean1, but not in Container2.Bean1.Therefore, no report is made to Resource Analyzer on the number of beanobjects that were destroyed in Container2.Bean1.

If the instrumentation level for the enterprise bean category is set tohigh, then the counter Avg Method Rt (average method response time) isavailable to be enabled (because its impact cost rating is medium) in theEnterprise Beans module. When the instrumentation level is set at themodule level, the same instrumentation level is automatically invoked for allinstances occurring within the module. Because aggregate data isreported at the module level, the average response time on all methods foreach instance is reported.

However, it is possible to select any single instance and reset theinstrumentation level for that specific instance. So, for example, ifthe value for the instrumentation level for Container2.Bean2 is resetto low, then no report from Container2.Bean2 is included in theaggregate value reported by the Avg Method Rt counter at the module levelbecause the counter's medium impact cost rating is excluded when theinstance's instrumentation level is set to low. Detailedinformation is provided in Rating the impact of performance monitoring.

How performance data is accessed and reported

The Resource Analyzer uses the Performance Manager Infrastructure (PMI) tomanipulate the raw data that is collected about the application server.As an application runs, raw data is saved into the application server'smemory. The PMI client takes the raw data and completes the complexcalculations that change the raw data into meaningful performancestatistics. The PMI client removes the overhead of performing thesecalculations at the application server. This enables a minimumperformance impact for data collection on the application server and ensuresthe optimal efficiency of the running application. Figure 2 shows how the PMI client reports performance data toanalytical tools.

Figure 2. How the PMI client gathers and reports performance data

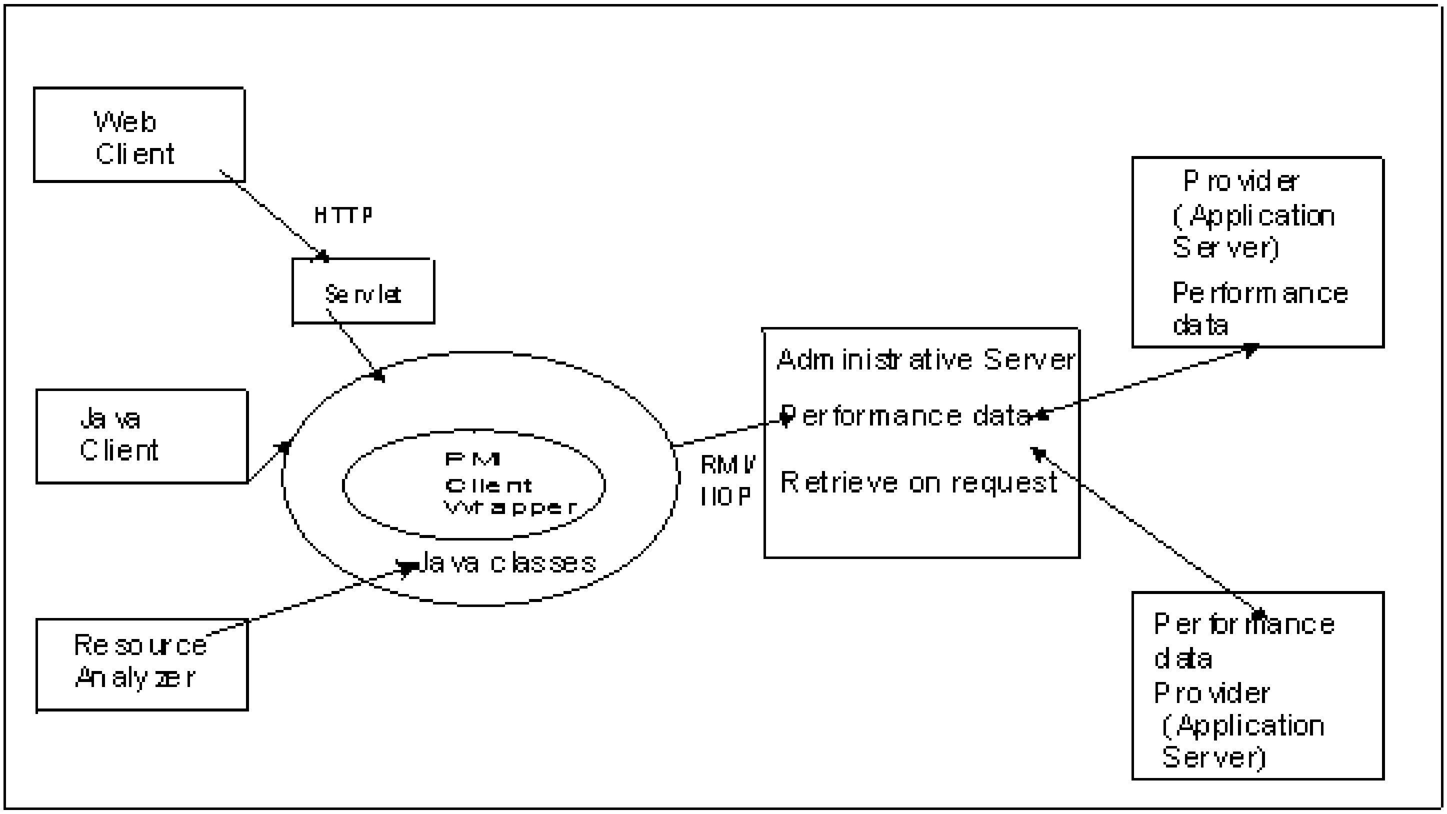


Figure 2 shows the three types of analytical tools that can accessperformance data reported by the PMI client:

- A Web client can use the Hypertext Transfer Protocol (HTTP) to activate aservlet that is designed to gather performance data on the application serverby using the PMI client. The PMI client retrieves the raw data from theapplication server or servers and completes the required calculations fordisplay by the servlet.
- A Java client can access the PMI client directly to get meaningfulperformance data. The PMI client uses Remote MessagingInterface/Internet Inter-ORB Protocol (RMI/IIOP) to exchange data with theadministrative server.
- The Resource Analyzer uses the calculations reported by the PMI client toconstruct graphic and tabular displays representing applicationperformance.

The application server records the sum, the count, and the sum of thesquares (for standard deviation calculations). It also maintainscalculations that provide a time-weighted dimension to the statisticaldata: the time-weighted sum (which is updated each time the counter ischanged), the time when the counter was created, the time when the counter waslast updated, and the current value of the counter.

With this data, the Resource Analyzer can be used to do the following typesof analysis:

- Monitor real-time performance, such as response times for servlet requestsor enterprise bean methods
- Detect trends by analyzing logs of data over time
- Determine the efficiency of a configuration of resources (such as theamount of allocated memory, the size of database connection pools, and thesize of a cache for enterprise bean objects)
- Gauge the load on application servers and the average wait time forclients

6.6.0.12.1: Rating the impact of performance monitoring

This section explains the ratings for the impact costs of counters andinstrumentation levels. The following topics are included:

- Understanding the impact ratings for data counters
- Understanding instrumentation levels for data collection

Understanding the impact ratings for data counters

By default, performance data is automatically collected for WebSphereresources and for run-time resources. All data collection affectsperformance in some way, and the impact on performance varies depending on thecounter. The Resource Analyzer represents the overhead cost associatedwith each counter as a rating of low, medium, or high. If a counter hasa low cost rating, its performance cost is minor and usually involves a singleaddition or subtraction. A counter with a high cost rating has a higherperformance impact. A high cost rating usually indicates that severalcalculations, including multiplication, division, or both, are involved ingathering the data for the counter.

For example, collecting the number of times a bean instance is passivatedhas a low impact on performance. Calculating the average response timeon all methods of an enterprise bean's remote interface has a mediumimpact on performance; calculating the average number of methods beingprocessed concurrently has a high impact. The associated cost rating(low, medium, or high) for each counter is listed in Performance data.

Understanding instrumentation levels for data collection

The resources in a WebSphere administrative domain are*instrumented* so that statistical data can be collected.Instrumentation refers to the mechanism by which some aspect of the runningsystem is measured (analogous to a meter attached to a resource). Eachresource category has an *instrumentation level* (different from theimpact rating for the counters in that category). The instrumentationlevel determines *which* counters are available to be collected forthat category.

The instrumentation level is set through the **Monitoring Settings**menu option in the Resource Analyzer, or through the Properties dialog box,which is accessed from the WebSphere Administrative Console. See Setting the instrumentation level for information about this task.

For example, if a resource category has an instrumentation level setting oflow, only counters having a low-impact cost rating are available forselection. If the instrumentation level is set to medium, then countershaving low- impact and medium-impact cost ratings are available forselection. Similarly, when the instrumentation level is set to high,all counters with low-, medium-, and high-impact cost ratings are availablefor selection.

An instrumentation level also can be set to maximum, which enables theavailability of all counters and, in addition, increases the level ofgranularity when reporting on enterprise methods. This setting has ahigher impact on a system's performance. Conversely, theinstrumentation level can be set to none, which disables performance reportingand eliminates any impact of monitoring on system performance.Initially, the instrumentation levels are set to none.

When you set an instrumentation level at the root level of the resource,all instances of that resource acquire the same setting. Any individualinstance can be selected and separately set to a different instrumentationlevel.

The instrumentation levels rate the approximate performance costs forcounters. However, the ratings do not reflect the impact of countersthat can become numerous.

See Performance data for more information on instrumentation levels and countersfor each resource category.

Indentifying the instrumentation level represented in the graphic display

The instrumentation level can be set for each resource category and foreach resource instance. The current instrumentation level isrepresented by a small icon with a dial setting. The meanings of thedial settings are as follows:

- In the Monitoring Settings dialog box, icons in the form of dials show theinstrumentation rating associated with each resource category. Theseicons use both color and dial position to convey information:
  - When the instrumentation level is deactivated, no dial is displayed on theyellow icon, and a zero (0) is located over the vertical bar. In thiscase, the instrumentation level for the associated resource category is set tonone.
  - When the instrumentation level for a category is set to low, a blue dialis positioned at 10 o'clock on the yellow icon. In this case, onlythe counters that have an impact rating of low are displayed in the CounterSelection panel of the Data Monitoring pane.
  - When the instrumentation level for a category is set to medium, a greendial is positioned at 12 o'clock on the yellow icon. In this case,only the counters that have impact ratings of low or medium are displayed forselection in the Counter Selection panel of the Data Monitoring pane.
  - When the instrumentation level for a category is set to high, a red dialis positioned at 3 o'clock on the yellow icon. In this case, thecounters for all impact ratings: low, medium, and high, are displayed inthe Counter Selection panel of the Data Monitoring pane.
  - When the instrumentation level for a category is set to maximum, a reddial is positioned at three o'clock on the yellow icon, and anexclamation point is displayed to the right of the dial. In this case,in addition to the counters with low-, medium-, and high- impact ratings beingavailable, reporting for individual enterprise bean methods isavailable.

See Performance data for more information on instrumentation levels and countersfor each resource category.

6.6.0.12.2: Exploring the Resource Analyzer's graphic interface

This section describes the graphical interface for the ResourceAnalyzer. The following topics are included:

- Layout of the console
- The Resource Selection pane
- The Data Monitoring pane
- The menu bar
- The toolbar icons
- The node icons
- The status bar

Layout of the console

The Resource Analyzer main window consists of two panes: the ResourceSelection pane and the Data Monitoring pane. The Resource Selectionpane, located on the left, provides a view of resources for which performancedata can be displayed. The Data Monitoring pane, located on the right,displays numeric and statistical data for the resources that are highlighted(selected) in the Resource Selection pane.

See The Resource Selection pane and The Data Monitoring pane for more information.

You can adjust the width of the Resource Selection and Data Monitoringpanes by dragging the split bar left or right. You can rearrange theorder of the table columns in the Data Monitoring pane by dragging the columnheading left or right. You can also adjust the width of the columns bydragging the edge of the column left or right.

The Resource Selection pane

The *Resource Selection* pane provides a hierarchical (tree) viewof resources and the types of performance data available for thoseresources. Use this pane to select which resources to monitor and tostart and stop data retrieval for those resources.

The Resource Selection pane displays resources and associated resourcecategories in an indented tree outline. Clicking the plus (+) and minus(-) symbols expands and collapses the tree to reveal the categories for thevarious resource instances. The resource tree can also be navigated byusing the up and down arrow keys to cycle through the branches and leaves andthe left and right arrow keys to expand and collapse the tree ofresources. Resource instances can be expanded to reveal the instancesthey contain, if applicable. For example, when a container instance isexpanded, the enterprise bean instances in the container are revealed.The Data Monitoring pane automatically displays the appropriate selection ofcounters for any objects highlighted in the Resource Selection pane.

The first level of the hierarchy includes all nodes (machines) in theadministrative domain, followed by all application servers on the node.Below each application server, all resource categories are listed. Ifthe enterprise beans category is expanded, all container instances in theserver are displayed. All enterprise bean instances within thatcontainer appear below that in the hierarchy. A Methods resource isassociated with each bean. Clicking an individual bean or containerinstance causes its corresponding counters to be displayed in the DataMonitoring pane. For enterprise beans, the counters displayed depend onwhether the bean is an entity bean or a session bean. For containers,the counters are aggregate counters for all enterprise beans in thecontainer.

The Data Monitoring pane

The *Data Monitoring* pane enables the selection of multiplecounters and displays the resulting performance data for the currentlyselected resource. It contains two panels: the Data Display panelabove and the Counter Selection panel below.

The Counter Selection panel

The *Counter Selection* panel shows the counters that areavailable to be selected for the resource performance category.

Two factors determine the list of available counters in the CounterSelection panel:

- Only counters associated with the resource that is selected in theResource Selection pane are displayed.
- Only counters having impact cost ratings within or below theinstrumentation level that is set for that resource in the WebSphere AdvancedAdministrative Console are displayed.

The first three counters shown for each resource performance category areselected by default. All counters can be selected or deselected, andthe resulting output, shown in the top panel, automatically reflects theselection.

The columns in the Counter Selection panel supply the following informationfor each counter:

- **Name**. The names of the counters that are available forselection with this resource.
- **Description**. A brief description of the function of eachcounter.
- **Value**. The value for the counter, displayed according tothe display mode in effect. Values are actual values (not scaled valuesused for the chart, if applicable).
- **Select**. A check box that indicates whether a counter isto be reflected in the chart. To hide data, clear the check box.The column representing that counter is then removed from the View Datawindow, and the graphic display for that counter is removed from the ViewChart window.
- **Scale**. A value indicating whether data has been scaled(amplified or diminished) from its actual value to fit on the chart.This value is reflected only in the View Chart window.

  The value for the **Scale** column can be set manually by editingthe value of the **Scale** field. See Scaling the chart display manually for information on manually setting the scale.

The Data Display panel

When a counter on the list in the Counter Selection panel is selected, thestatistics gathered from that counter are displayed in the *DataDisplay* panel at the top of the Data Monitoring pane.

The View Data window shows the counter's output in table format;the View Chart window displays a graph with time represented on the x axis andthe performance value represented on the y axis. One or moreperformance counters can be simultaneously graphed on a single chart.The chart plots data from *n* data points, where *n* is thecurrent table size (number of rows).

The display of multiple resources and aggregate data

When a single resource is selected in the Resource Selection pane, the DataMonitoring pane displays a choice of a table view or a chart view. Ifmultiple resources are selected, the Data Monitoring pane displays a singledata sheet for viewing summary information for the selected resources.The data sheet displays the tables for all objects of similar type for theselected resources. For example, if three servlet instances areselected, the data sheet displays a table of counter values for all theservlets.

The Resource Analyzer provides aggregate data at the module level.See Figure 1 for a depiction of the data collection hierarchy used by theResource Analyzer. If aggregate data is available for a group, it isdisplayed in the Data Monitoring pane. For example, for each enterprisebean home interface, counters track the number of active enterprise beans ofthat home. Each container has an aggregate value that is the sum of allthe enterprise beans in that container. The enterprise beans resourcecategory (module) within the application server has an aggregate value that isthe sum of all enterprise beans in all containers.

The menu bar

The menu bar contains the following options:

- **File** menu. Used to change to current mode (from loggingmode), to open an existing log file, and to exit from the ResourceAnalyzer. The **File** menu contains the following items:
    - **Current System**. Resumes the display of real-time data intables and charts. This menu option is used to stop viewing data from alog file and return to viewing real-time data.
    - **Open Log File**. Displays a dialog box for specifying thename and location of an existing log file to be replayed.
    - **Exit**. Closes the Resource Analyzer. If you madechanges to the instrumentation levels of any resources during the session, adialog box opens to ask whether you want to save the changed settings beforeclosing the tool.
- **Actions** menu. Used to start and stop the reporting ofdata, and to clear and refresh data. The **Actions** menucontains the following items:
    - **Start**. Starts retrieving performance data for the group(or groups) selected in the Resource Selection pane.
    - **Stop**. Stops retrieving performance data for the group(or groups) selected in the Resource Selection pane.
    - **Stop All**. Stops data retrieval for all groups for whichdata is being reported.
    - **Refresh**. Queries the administrative server for any newlystarted resources since data retrieval began or for additional counters toreport. This operation also is recursive over all componentssubordinate to the selected resource.
    - **Clear Values**. Deletes the values currently displayed intables and charts. For example, after stopping a counter, you can usethis operation to remove the remaining data from a table.
    - **Reset to Zero**. Resets the "clock" used for reportingdata for counters of the selected performance group.
    - **Get Value**. Obtains the current values for all countersthat belong to the selected resource. The values are displayedindependently of the refresh rate. Data reporting does not need to beenabled for the resource.
    - **Monitoring Settings**. Displays the Performance MonitoringSettings dialog box to enable changes in the instrumentation levels forresource performance categories.
    - **Rewind**. Rewinds the log file to the beginning.

  Note that right-clicking a resource in the Resource Selection pane displaysa menu that provides the following subset of the **Action** menuoptions: **Start**, **Stop**, **Refresh**, **GetValue**, **Clear Values**, **Reset to Zero**, and**Monitoring Settings**.
- **Logging** menu. Provides **On** and **Off**options that are used to start and stop recording data in a log file.The **Off** action saves the log file. If you start a new logfile and specify the same file name, the file is

overwritten.

- **Options** menu. Used to change settings. The**Options** menu contains the following items:
  - **View Data As**. Specifies how counter values aredisplayed. You can choose whether to display absolute values, changesin values, or rates of change. How data is displayed differs slightlydepending on where you are viewing data. The choices follow:
    - **Raw Value**. Displays the absolute values.
    - **Change in Value**. Displays the change in the currentvalue from a previous value.
    - **Rate of Change**. Displays the ratio*change*/(*T1 - T2*), where*change* is the difference between the current value and a previousvalue, *T1* is the time when the current value was retrieved, and*T2* is the time when the previous value was retrieved.
  - **Play Speed**. Specifies how fast to replay a logfile.
  - **Set Refresh Rate**. Displays a pop-up field for specifyinghow often data is retrieved, in seconds. The default is 10seconds.
  - **Set Table Size**. Displays a pop-up field for entering thenumber of rows (and therefore the number of data points) to be displayed inthe View Data window. The default value is 40 rows.
- **Help** menu. Provides information for users.

The toolbar icons

Toolbar icons provide shortcuts to frequently used commands. Thetoolbar includes the following icons:

- **Refresh**. Updates data and structures for the selectedresources. That is, it polls the administrative server to retrieve newinformation about additional counters to display or new servers recently addedto the domain.
- **Start**. Starts reporting data for a resource. Inlogging mode, plays a log file.
- **Stop**. Stops retrieving performance data for the group(or groups) selected in the Resource Selection pane.
- **Stop All**. Stops retrieving data for all groups for whichdata is being reported.
- **Clear Values**. Deletes the values currently displayed inall tables and charts.
- **Reset to Zero** icon. Resets the counters.
- **Get Value** icon. Obtains the current values for allcounters associated with the selected resource. The values aredisplayed independently of the refresh rate. The resource does not needto be running in order to get the values of the counters.

The node icons

In the Resource Selection pane, the color of the node icon indicates thecurrent state and availability of the application server in the domain.

- Green--The resource is running and available.
- Red--The resource is stopped.

This use of icon colors in the Resource Selection pane is identical to theuse of colors for the node icons shown in the administrative console.

In addition, in the Resource Selection pane, the following icon colorsindicate whether performance data is being reported for a resource:

- Blue--Data is being reported for this resource.
- Red--No data is being reported for this resource.

Note that even though data is continuously being *collected* for aparticular resource, this data is not *reported* (not displayed bythe Resource Analyzer) unless you explicitly enable data reporting for aresource from within the Resource Analyzer. See Setting the instrumentation level for instructions.

The status bar

The status bar across the bottom of the Resource Analyzer windowdynamically displays the current state of the reporting values. Thefollowing state information is reported in the status bar:

- The number of resources (groups) currently selected in the ResourceSelection pane
- The number of available counters for the currently selected resource orresources
- The current setting for the refresh rate
- The table size in use in the current Data Display panel
- The display mode in use in the current Data Display panel
- The current state of the logging setting

6.6.0.13: Log Analyzer main window

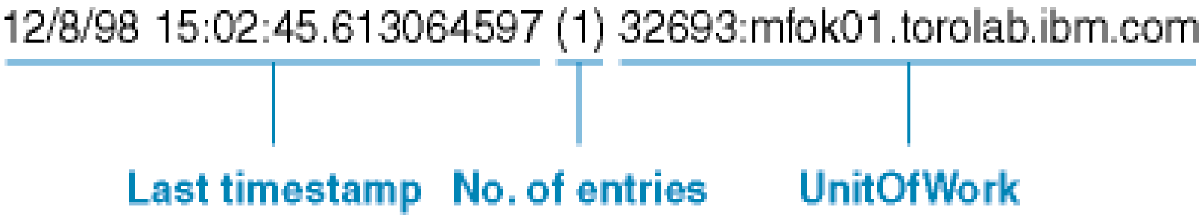
The Log Analyzer takes one or more activity or trace logs, merges all the data, and, by default, displays the entries in unit of work groupings. It analyzeserror conditions in the log entries to provide error message explanations. The Log Analyzer's main window has the following interface:

- Three window panes
- Status line
- Menu bar
- Pop-up actions

Window panes
The Log Analyzer window has three panes:

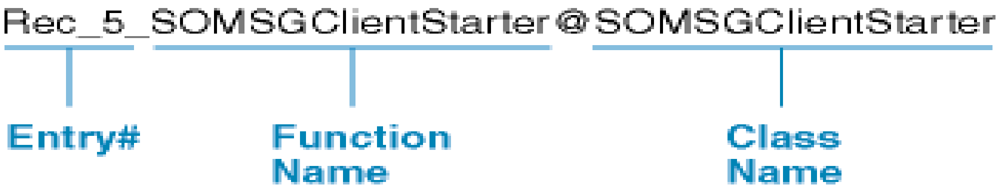
- **Logs pane (left)**
  You may find the unit of work UOW grouping useful when you are trying to find related entries in the activity log or when you are debugging problems across multiple machines. By default, Log Analyzer's Logs pane displays log entries by unit of work. It lists all the unit of work (UOW) instances and its associated entries from the logs that you have opened. The file name of the first log that you opened is shown in the pane's title bar. By default, the log entries are shown in UOW sequence. There is a root folder and under it, each UOW has a folder 📁 icon which you can expand to show all the entries for that UOW. All log entries *without* any UOW identification are grouped into a single folder in this tree view. The UOW folders are sorted to show the UOW with the latest timestamp at the top of the list. The entries *within* each UOW are listed in the reverse sequence, that is the first (earliest) entry for that UOW is displayed at the top of the list. If you have merged several logs in the Log Analyzer, all the log entries are merged in timestamp sequence within each UOW folder, as if they all came from the same log.

  Each UOW line has the following format:

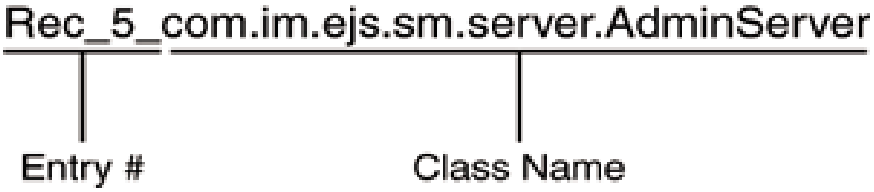
  

  Click the ⊞ icon next to the UOW folder to see all the log entries for the UOW. Each log entry's identification has the following format:

  For the WebSphere Enterprise Edition:

  For WebSphere Advanced Edition:



Every log entry is assigned an entry number, Rec_*nnnn*, when a log is opened in the Log Analyzer. If more than one file is opened in the Log Analyzer (merged files), the Rec_*nnnn* identification will not be unique because the number is relative to the entry sequence in the original log file and not to the merged data that the Log Analyzer is displaying. This Rec_*nnnn* appears in the first line (**RecoredId**) in the Records pane.

By default, each entry in this pane is color-coded to help you quickly identify the ones that have high severity errors.

- Non-selected log entry with background color of:
    - Pink indicates that it has a severity 1 error.
    - Yellow indicates that it has a severity 2 error.
    - White indicates that it has a severity 3 error.
- Selected log entry with background color of:
    - Red indicates that it has a severity 1 error.
    - Green indicates that it has a severity 2 error.
    - Blue indicates that it has a severity 3 error.

These colors are configurable and can be changed in the Log Analyzer's Preferences Log page. See Background color for different error severity levels and for more information on how to do this.

The Log Analyzer can display the log entries in different groupings. Use the Log Analyzer Preferences notebook: Logs page to set the grouping filters.

After the Analyze action has been invoked, each analyzed log entry has the following icons:

- ✓ indicates that the entry has some analysis information in one or more pages in the Analysis pane.
- ↻ indicates that the entry has some analysis information and that it has a reraised or remapped exception. You may want to look at the log entry prior to this one when diagnosing problems.
- ? indicates that the entry has either a severity 1 or 2 error but no additional analysis information is available for it.
- ✗ indicates that the entry has a severity 3 error and it has no analysis information.

- **Record pane (upper right)**
  When you select an entry under the unit of work in the Logs pane, you see the entry in the Record pane. The entry's identification is shown in the pane's title bar. Right-click in this Record pane to see the actions that you can perform on the entry. There is a drop down arrow next to **Record** which allows you to go back to look at the last ten records that you have viewed. The cache for the historical data (10, by default) is set in the Preferences General page.

  Note:

  1. The associated analysis data for these cached records are not shown. To see analysis information for cached data, reselect the entry from the Logs pane.

2. The timestamp format that appears in the Record pane must match the format that was used by the showlog command when formatting the log.

You can enable/disable line wrap mode for the Record Pane using the **Log Analyzer Preferences notebook: Record**. To print contents of this pane, select **Record > Print** when the Record pane is in focus.

- **Analysis pane (lower right)**
  When analyze action has been invoked *and* additional information is available, the information will appear in the Symptom page. The page, GPF, is filled in only when the analyze action is invoked *and* there is a GPF dump. If the page tab is grayed out, there is no information in that page. The pages of the Analysis pane are:
  - **Symptom**
    The Log Analyzer provides a database of information to help you recover from some common WebSphere errors. As a part of the analyze action, if such recovery information is found in the database for the selected log entry, the information is displayed in this page.

  You can set line wrap for information that appears in the Symptom page in the Analysis pane using the **Log Analyzer Preferences notebook -- Analyzer output**.

  The Symptoms page can be detached into separate windows to make it easy for you to view all the information by selecting **View > Detach into a new window**.

  You can set line wrap for information that appears in the Symptoms page in the Analysis pane using the **Log Analyzer Preferences notebook -- Analyzer output**.

Status line
There is a status line at the bottom of the window showing the status of actions.

Menu bar
The menu bar in the Log Analyzer's main window, has the following selections:

- **File**
  - **Open...**
    Opens a new log file in the analyzer. You can select either a formatted activity/trace log or a previously saved XML file. If you want the Log Analyzer to format a raw log file (by running the showlog command) prior to opening it, name the log file with suffix.log. If the Log Analyzer finds that the .log file contains formatted data, it skips the showlog formatting step.

    If you want to merge data from another log, select **Merge with**.
  - **Merge with...**
    When another log file is already opened in the Log Analyzer, use the **Merge with** action to open subsequent logs. The Log Analyzer merges the data from all the logs that it opens and displays all the entries within timestamp sequence in the UOW folders. The data appears as if they came from one log.

    If you want the Log Analyzer to run the showlog utility prior to opening it, name the log file **activity.log**.
  - **Redisplay logs**
    To redisplay the logs using the recently set filters.
  - **Save as**...
    Saves the log as an XML file (or text file). If analyze action has been performed, all the Symptom analysis information is also saved. If logs are merged in the Log Analyzer, the saved file contains entries of all the merged logs in the sequence that is shown in the Logs pane.

    Note: If the merged logs have different timestamp formats, you should not save the merged information because the Log Analyzer only recognizes a single timestamp format for each file that it opens.

    - **Save**
      Is only enabled if the first file that you opened is an XML file. It resaves the XML file with all the data that is currently displayed in the Log Analyzer. Any Symptom information is also saved. If logs are merged in the Log Analyzer, the saved file contains entries of all the merged logs in the sequence that is shown in the Logs pane. **Note:** If the merged logs have different timestamp formats, you should not save the merged information because the Log Analyzer only recognizes a single timestamp format for each file that it opens.
    - **Print Log...**
      Prints all the entries that the Log Analyzer is displaying. If logs are merged in the Log Analyzer, the output contains entries of all the merged logs in the sequence that is shown in the Logs pane. If analyze action has been performed, all Symptom analysis information is also printed. To print parts of the log, use **Record > Print**.
    - **Close**
      Closes the opened log.
    - **Update Database**
      Updates the symptom database which is used for Symptom analysis. It downloads the latest version of the symptom database from the URL specified in the ivblogbr.properties file.
    - **Preferences...**
      Lets you configure and change the appearance of the Log Analyzer window and its contents.
    - **Exit**
      Exits the Log Analyzer and closes its window.
- **Edit**
    - **Copy**
      Copies the selected text in the Record or Analysis pane to the clipboard. If you have not selected any text, **Copy** does not appear in the menu.
    - **Find**
      Allows you to find text strings in the focused pane.
- **View**
    - **Logs**
      Toggles the visibility of the Logs pane.
    - **Record**
      Toggles the visibility of the Record pane.
    - **Symptom**
      Toggles the visibility of the Symptom page in the Analysis pane.
- **Record**
  All the actions under this menu applies to the focused pane.

  To select several entries, hold down the **Ctrl** key when making the selection. When a folder is selected, the action applies to all the entries in that folder.

    - **Analyze**
      Retrieves and displays additional documentation on known errors and error messages in the Analysis pane (Symptom page). Select the folder(s) and/or entries in the Logs pane, right-click to select the **Analyze** action, or from the menu bar, select **Record > Analyze**. **Note:** If you invoke **Analyze** for the root folder, then all the entries in the log that you are viewing will be analyzed.

  If some analysis information is available for an entry, it will either have a or icon next to it in the Logs pane. If the analyze action has already been performed, the selection will be grayed out.

  - **Save to file**
    Saves the selected entries in the Logs pane. If UOW folders are selected, all the entries in the folder are saved. Any retrieved analysis information is also saved. If the focused pane is either the Records or Analysis pane, then only that pane's information is saved.
  - **Print**
    - If the focused pane is Logs, the action prints the selected folder(s) and/or entries. Any retrieved analysis information for those entries is also printed.
    - If the focused pane is Record, the action prints the entry that is currently in the Record pane. Any retrieved analysis information is not printed.
    - If the focused pane is Analysis, the action prints the Symptom page contents.
- **Windows**
  If you have detached the Symptom page in the Analysis pane into separate windows, all the windows will be listed under this menu and you can select the windows that you want to bring to the foreground.
- **Help**
  Provides a list of WebSphere's on-line documentation for additional information.

Pop-up actions
In the focused pane, right-click to bring up a list of actions in a pop-up menu. Actionsthat you cannot perform are grayed out. When a folder is selected in Logs pane, the actionapplies to all the entries in that folder. To select several folders or entries in theLogs pane, hold down the **Ctrl** key when making the selection.

6.6.0.13.1: Log Analyzer Find window

The Log Analyzer Find window allows you to look for text strings in the focused pane.For example, if you remember the Unit of Work identification, you can enter that text string in theFind window to quickly locate the Unit of Work folder in the Logs pane.

6.6.0.13.2: Log Analyzer Preferences notebook -- General

The Log Analyzer Preferences notebook's General page lets you specify the behavior ofpanes in the Log Analyzer's window:

- **Show title bars**
  Shows title bars of window and its panes.
- **Highlight selected pane**
  Highlights the pane that is in focus.
- **Pane history cache size**
  Specify a number of records to save in the cache. The Log Analyzer keeps a history of the (specified number of) records that you have viewed. You can use the drop down list next to **Record** in the pane's title bar to see these cached entries.

  Note: The associated analysis data for these records are not saved. To see analysis information, reselect the entry from the Logs pane.
- **Show logo at startup**
  Shows the logo when you start-up the Log Analyzer.

When you are finished, click **OK** to apply your changes and close thePreferences notebook.

6.6.0.13.3: Log Analyzer Preferences notebook -- Appearance

The Log Analyzer Preferences notebook's Appearance page lets you define the overallappearance of the Log Analyzer. You can select the family of products and its textureschemes that you want the Log Analyzer's window to emulate.

When you are finished, click **OK** to apply your changes and close thePreferences notebook.

6.6.0.13.4: Log Analyzer Preferences notebook -- Toolbars

The Log Analyzer Preferences notebook's Toolbars page lets you customize the appearanceand contents of the toolbar in the Log Analyzer window. You can select whether there istext and/or icon in the toolbar, as well as, the functions that you want in the toolbar.

When you are finished, click **OK** to apply your changes and close thePreferences notebook.

6.6.0.13.5: Log Analyzer Preferences notebook -- Help

The Log Analyzer Preferences notebook's Help page lets you select the browser that willbe used to display online help files.

For Windows NT, the default webbrowser will be used and you do not need to update any settings unless there areproblems when bringing up the default browser.

For AIX, HP-UX, and Solaris, you haveto update the following settings, especially the browser's full path in the **Browserlocation** entry.

- **Help browser**
  Select the web browser you want to use.
- **Browser location**
  Select the location of the browser executable file. This should be correct by default, but if you cannot access help then you may need to explicitly enter the browser location.

When you are finished, click **OK** to apply your changes and close thePreferences notebook.

6.6.0.13.6: Log Analyzer Preferences notebook -- Logs

The Logs page of the Log Analyzer Preferences notebook lets you group the entries inthe logs by different entry fields for viewing. For example, you can select to group thelog entries by TimeStamp or clientHostName when they are displayed in the Logs pane.

Filter 1
Use this filter to set the first level of grouping when log entries are displayedin the Logs pane. By default, the log entries are grouped by UnitOfWork.

Filter 2
Use this filter to set the second level of grouping (that is, within the groupingof filter 1 folder) when log entries are displayed in the Logs pane.

All the entries within the grouped folders are always sorted in timestamp sequence withthe earliest entry at the top of the list.

Redisplay log file immediately
Select this box to immediately regroup the logs entries (after you have clicked **OK**)based on the new filter settings. The entries in the Logs Pane are redisplayed accordingto the new grouping. If you want to delay the grouping, then do not select this box and,at a later time, you can use the **File > Redisplay logs...** menuselection to regroup and display the log entries based on the changed filter settings.

When you are finished, click **OK** to apply your changes and close thePreferences notebook.

6.6.0.13.7: Log Analyzer Preferences notebook -- Severity

The Log Analyzer Preferences notebook lets you change the background colors of logentries that appear in the Logs pane. The colors are useful to help you quickly identifyentries that have high severity errors and the entry that you have currently selected.

Use colors to indicate level of error severity
Select this checkbox to color-code the background of log entries and folders. Whenselected, the radio button selections in this page are enabled.

Background color
For each folder and entry in the Logs pane, there is some text describing the entry.

To choose a background color for selected log entry that has a severity 1 error, do thefollowing:>

1. Select **Selected node**.
2. Select **Severity 1**.
3. Select the color by clicking on the color **Swatches**. To use the default setting, click **Restore Default**. To see the results of your change, look at the **Preview** box.
4. Click **Apply** to save that setting.

Repeat similar steps to change the background color for selected log entries that haveseverity 2 and 3 errors.

To choose a background color for an unselected log entry that has a severity 1 error,do the following:

1. Select **Unselected node**.
2. Select **Severity 1**.
3. Select the color by clicking on the color **Swatches**. If you want to use the default setting, click **Restore Default**. If you want to see the result of your change, look at the **Preview** box.
4. Click **Apply** to save that setting.

Repeat similar steps to change the background color for unselected log entries thathave severity 2 and 3 errors.

Sample
You can see the result of you color change prior to applying the change. Look atthe nodes shown in the **Sample** box. For color changes of *selected*nodes, click on the node in the sample box to see the color change.

When you are finished, click **OK** to apply your changes and close thePreferences notebook.

6.6.0.13.8: Log Analyzer Preferences notebook -- Record

The Log Analyzer Preferences notebook lets you set the line wrap mode for the data thatis displayed in the Record pane. It also lets you set the time and date format of thetimestamp so that it matches the timestamp format of the log that you are opening.

Enable line wrap mode for record pane
Select this checkbox to enable line wrapping for the information that appears in theRecord pane.

Date format

When viewing logs, this date format only changes the timestamp format that isdisplayed in the Record pane. The timestamp format in the log file and the timestamp shownin the Logs pane are not affected by this setting.

Time format

When viewing logs, this time format only changes the timestamp format that isdisplayed in the Record pane. The timestamp format in the log file and the timestamp shownin the Logs pane are not affected by this setting.

When you are finished, click **OK** to apply your changes and close thePreferences notebook.

6.6.0.13.9: Log Analyzer Preferences notebook -- Analyzer output

The Log Analyzer Preferences notebook lets you enable line wrap for information thatappears in the Analysis pane.

Set line wrap
Select the appropriate checkbox to enable line wrap for the following page that appears inthe Analysis pane:

- Symptom page

When you are finished, click **OK** to apply your changes and close thePreferences notebook.

6.6.0.14: XML-SOAP Admin tool

Use the SOAPEarEnabler toolto add administrative interfaces to your EAR files. You can then use the XML-SOAP Admin tool with these EAR files.

WebSphere Application Server provides a modified version of theApache SOAP XML-Admin interface (or XML-SOAP Admin tool) for each SOAP-enabled EAR file.This interface allows you to do the followingfor each context root:

- List configured services, showing active and stopped services
- Stop a service
- Start a service
- View the Apache Soap deployment descriptor for a service

Accessing the XML-SOAP Admin tool

Access the XML-SOAP Admin tool through a Web browser by specifying:

`http://localhost/<`**`contextroot`**`>/admin/index.html`

Note: The *context root*, in this example, is the context specified when installing the SOAP-enabled `.ear` file.The context root for SOAP samples is `soapsamples`.

Therefore to use this interface with the SOAP samples,enter:

`http://localhost/soapsamples/admin/index.html`

You cannot use the XML-SOAP Admin tool to add or remove a service. Usethe SOAP Ear Enabler tool to add or removeservices. A "stopped" service is persistedacross starts and stops of the applicationserver. Therefore, if you stopa service, it will remain stopped until thenext time you use the XML-SOAP Admin toolto start it again.

You can add the XML-SOAP Admin tool interface to an enterpriseapplication when you SOAP-enable the EAR file.In interactive mode, you are asked whether you want to add the XML-SOAP Admin tool interface.Replying "yes" will add the necessaryJSP files and bindings that allow you to accessthe XML-SOAP Admin tool interface for the application.The interface is an optional addition becauseyou may not want to expose itin a production environment. Optionally,you may choose to use the application assemblytool to assign roles to the XML-SOAP Admin toolso that it is secure.

Updating an existing SOAP-enabled Enterprise Application

The Application Assembly tool is used toupdate the contents and configuration ofan enterprise application. For example, tosecure the XML-SOAP Admin tool interfacefor a particular EAR, use the application assemblytool to secure the resource. (See article Securing applicationsfor security information.) However, you cannot use the application assembly toolto add or remove a Web service.

To add or remove a service to the XML-SOAP Admin tool, startwith the original EAR file and execute the enabling process again.

Note: The SOAPEar Enabler tool saves a backup copy of the EAR file.

6.6.0.15: Deployment Tool

Main topics

- Generating deployment code for EJB enterprise beans

Other topics

- Meet-in-the-middle mapping support
- Modifying the schema document
- Modifying the map document
- Implementing custom finders in home interfaces for CMP entity beans

Migrating CMP beans from Version 3.5

To preserve the existing (older) table/schema rules when migrating CMP entity beans from WebSphere Application Server Version 3.5 to Version 4.0, use the -35 option ofthe deployment tool. The deployment tool is used to generate code for deployment on Version 4.0. Its -35 option will generate table mappings based on the 3.5 rules. Combine the-35 option with the -dbschema EJB option to map to the 3.5 generated schema that was used.

To migrate manually, the EJB JAR requiring migration must be invoked through the deployment tool command line, using the arguments:

```
 -35 -dbschema EJB
```

Without these options, the database table will follow a new set of rules that is more aligned with the defaults for VisualAge for Java.

The .java and .class file time stamps must differ

If an erroroccurs when you are generating code for deployment, ensure that the time stampsof the .class files and corresponding .java files in your EJB JAR file do notmatch. If the .class and .java files have the same time stamp, one solution isto perform an operation on the .java files (such as "touch" on a UNIX system)to cause their time stamps to change without changing the file content.

6.6.0.15.1: Generating EJB deployment code from the command line

Before you can successfully run your enterprise beans on either a test orproduction server, you need to generate deployment code for the enterprisebeans. The Deployment Tool for Enterprise JavaBeans (referred to in theremainder of this topic as simply the EJB Deploy Tool) provides a command-lineinterface that you can use to generate enterprise bean deployment code. The tool employs a command-line environment that allows you to runa build process overnight and have the deployment tool automatically invoked togenerate your deployment code in batch mode.

The EJB Deploy Tool is invoked from the command line using the **ejbdeploy**command, whichaccepts an input EJB JAR file that contains one or more enterprise beans. Itthen generates an output deployed JAR file that contains deployment codein the form of .class files.

Specifically, when you run the ejbdeploy command, the following activitiesoccur:

- Code is imported from the input JAR file
- Deployment code is generated
- RMIC is run
- The deployment code is compiled
- Code is exported to the output JAR file

These activities are discussed further in some of the descriptions for theoptions that can be specified with the ejbdeploy command.

The EJB Deploy Tool supports meet-in-the-middle mapping, EJB single andmultiple table inheritance, and associations that have been defined in VisualAgefor Java. It also supports the use of converters, which translate a databaserepresentation to a Java object type, and composers, which are used to map asingle complex bean field to multiple database columns.

For information about migrating entity beans from WebSphere ApplicationServer Version 3.5, see the sectionbelow entitled "Migrating CMP entity beans from WebSphere ApplicationServer, Version 3.5". For information aboutgenerating deployment code for VisualAge for Java enterprise beans and runningthem in WebSphere Application Server, see the section below entitled"Deploying and running VisualAge for Java enterprise beans in WebSphere ApplicationServer".

Syntax

ejbdeploy *input_JAR_name working_directoryoutput_JAR_name* [**-codegen**] [**-cp** *classpath*][**-dbname** *name*] [**-dbschema** *name*] [**-dbvendor** *name*][**-ignoreErrors**] [**-keep**] [**-noinform**] [**-novalidate**] [**-nowarn**] [**-quiet**] [**-rmic**"*options*"] [**-trace**] [**-35**]

Command arguments

ejbdeploy

The command to generate deployment code. If run without any arguments, the ejbdeploy command displays a list of arguments that can be run with the command.

input_JAR_name

The fully qualified name of the input JAR file that contains the enterprise beans for which you want to generate deployment code; for example, c:\ejb\inputJARs\myEJBs.jar. (This argument is required.) If your input JAR file contains CMP beans, the ejbdeploy command automatically creates the schemas and mappings for the CMP beans. It is important to note that the ejbdeploy command no longer uses what is specified on the system class path. Instead, the dependent classes need to be contained in a JAR file and included in the command processing using the **-cp** option. You must ensure that the .class files of each enterprise bean's home and remote classes are packaged in the input JAR file. (Note that the best way to specify dependent JAR files is through the Java extension mechanism and by listing the required JAR files in the manifest file of the input JAR file. The EJB Deploy Tool will respect these extensions and you won't need to specify them using a command-line option.)

working_directory

The name of the directory where temporary files are stored that are required for code generation. (This argument is required.) If the working directory that you specify already exists prior to running the ejbdeploy command, the temporary files are generated into the working directory and are not removed. However, if the working directory does not already exist prior to running the command, the directory is created and the temporary files are generated into it, but the directory and all of the temporary files are later automatically removed when the deployment code generation is complete. (If you do not want the working directory to be removed when the deployment code generation is complete, use the -keep option.)

output_JAR_name

The fully qualified name of the output JAR file that is created by the ejbdeploy command and that contains the generated classes required for deployment; for example: c:\ejb\outputJARs\myEJBs.jar.
(This argument is required.) The directories specified in the fully-qualified name must already exist before you run the ejbdeploy command. (Note that when you specify a name for the output JAR file and then run the ejbdeploy command, any existing output JAR file of the same name will be overwritten without warning.)

-codegen

Restricts the ejbdeploy command to just (a) importing code from the input JAR file (b) generating the deployment code, and (c) exporting code to the output JAR file. (If you do not specify the -codegen option, the ejbdeploy command performs the same operations as those performed using the -codegen option, but it will additionally compile the generated deployment code and run RMIC.)

-cp *classpath*

If you intend to run the ejbdeploy command against JAR files that have dependencies on other JAR files, you can use the -cp option to specify the classpath of the other JAR files. Using the -cp option, you can specify multiple JAR files as arguments. However, the JAR file names must be fully qualified, separated by semicolons, and enclosed in double quotation marks. For example:
-cp "*path*\myJar1.jar; *path*\myJar2.jar; *path*\myJar3.jar"

-dbname *name*

The name of the database you want to create. If the name of the database contains any spaces, the entire name must be enclosed in double quotes. For example:
-dbname "my database"

-dbschema *name*

The name of the schema you want to create. If the name of the schema contains any spaces, the entire name must be enclosed in double quotes. For example:

-dbschema "my schema"
(Note that this option is only used when creating a database definition in the top-down mode of operation. The database information is then saved in the schema document in the JAR file, meaning that the options do not need to be re-specified. It also means that when a JAR is generated by VisualAge for Java or WebSphere Studio Application Developer, the correct database must be defined at that point because it cannot be changed later.)

-dbvendor *name*

The name of the database vendor, which is used to determine database column types, mapping information, Table.ddl, and other information. The valid database vendor names are:
SQL92 (Generic SQL 92)
SQL99 (Generic SQL 99)
DB2UDBWIN_V61 (DB2 for Windows, V6.1)
DB2UDBWIN_V71 (DB2 for Windows, V7.1)
DB2UDBOS390_V6 (DB2 for OS/390, V6)
DB2UDBAS400_V4R5 (DB2 for OS/400, V4 R5)
ORACLE_V8 (Oracle, V8.0)
INFORMIX_V92 (Informix Dynamic Server.2000, V9.2)
SYBASE_V1192 (Sybase Adaptive Server Enterprise, V11.9.2)
MSSQLSERVER_V7 (MS SQL Server, V7)
MYSQL_V323 (MySQL, V3.23)

-ignoreErrors

Specifies that processing should continue even if compilation or validation errors are detected.

-keep

Retains the working directory after the ejbdeploy command has run.

-noinform

Suppresses informational messages.

-novalidate

Prevents the validators from running. (When you generate deployed code in batch mode from the command line, all of the installed validators are automatically run on the specified JAR file unless you specify this -novalidate option, which prevents all validators from running.) Note that the **-nowarn** option is generally a better options to use. It will hide any warnings but still display errors that may prevent the bean from deploying or running on the server.

-nowarn

Suppresses warning and informational messages.

-quiet

Suppresses status messages (but does not suppress error messages).

-rmic "*options*"

Enables you to pass RMIC options to RMIC. The options, which are described in Sun's RMI Tools documentation, must be separated by a space and enclosed in double quotation marks. For example:
-rmic "-nowarn -verbose"

-trace

Generates trace information.

-35

For CMP entity beans, the EJB Deploy Tool looks in the JAR file for an existing schema and map to use when generating deployment code. If no existing schema and map are found, a schema and map are created using improved top-down mapping rules. If you want to use the same top-down mapping rules for CMP entity beans that are used in the EJB Deploy Tool provided with WebSphere Application Server, Version 3.5, then specify the -35 option. This may be desirable in some situations. However, if you do not specify the -35 option, a form of top-down mapping is used that is an improvement over what has previously been available. (In the top-down mapping approach, you already have existing enterprise beans and their design determines the database design. You generate a schema and map and the generated schema contains one table for each entity bean with container-managed persistence. In these tables, each column corresponds to a CMP field of the enterprise bean, and the generated mapping maps the field to the column.)

Migrating CMP entity beans from WebSphere Application Server, Version 3.5

To migrate CMP entity beans from WebSphere Application Server, Version 3.5, to WebSphere Application Server, Version 4.0, the existing EJB applications should be migrated over and redeployed using the **-35** option of the EJBDeploy Tool so that the existing (older) table rules are kept. To migrate the beans manually, the EJB JAR requiring migration must be invoked through the command line and the -35 option should be used. Without this option, the database table will follow a new set of rules which is more aligned with the defaults for VisualAge for Java.

Deploying and running VisualAge for Java enterprise beans in WebSphere Application Server

To run an enterprise bean created in VisualAge for Java on WebSphereApplication Server Version 4.0, you must first generate deployment code for itusing the EJB Deploy Tool.

(Note that you cannot use the **EJB => Generate deployed code** menuitem in VisualAge for Java, because it is designed to support the embeddedVisualAge for Java WebSphere Test Environment that is based on WebSphereApplication Server, Version 3.5. Also, you cannot use the EJB Deploy Tool thatis shipped with the June 2001 release of WebSphere Application Server, Version4.0, because it does not support all of the function that is available inVisualAge for Java.)

To deploy and run a VisualAge for Java enterprise bean on WebSphereApplication Server, Version 4.0:

1. In the VisualAge for Java EJB Development Environment, select the **EJB => Export => Export EJB 1.1 JAR** menu item. This calls the Export Tool for Enterprise Java Beans 1.1, which enables you to create a JAR file that can be deployed to WebSphere Application Server. (If you don't use the Export Tool for Enterprise JavaBeans 1.1, no mappings, inheritance, or associations will be exported.)
2. Open a command window and change to the directory where you installed the EJB Deploy Tool.
3. Run the ejbdeploy command and use the JAR that you created in VisualAge for Java as your input JAR

file. This will create an output deployed JAR file.

4. Install the deployed JAR file into WebSphere Application Server, Version 4.0. Depending on your edition of WebSphere Application Server, this can be accomplished using either the SEAppInstall tool or the WebSphere Application Assembly Tool. If your edition of WebSphere Application Server includes the SEAppInstall tool and you want to use the SEAppInstall tool from the command line, ensure that you specify the "-ejbdeploy false" option, so that the earlier version of the EJB Deploy Tool that shipped with WebSphere Application Server Version 4.0 is not invoked.

Note that for CMP entity beans, a data definition language (DDL) that can be used to create corresponding database tables for a JAR file is contained within the META-INF directory and entitled Table.ddl. (In VisualAge for Java, Version 3.5, this information used to be directed into the generated persister. The table creation is now a task left up to the user, but it is made more convenient by the Table.ddl file.)

6.6.0.15.2: Meet-in-the-middle mapping support

By default, the ejbdeploy command automatically generates top-down mapping.However, after completion of the top-down operation, you can modify theresulting map and schema to fit your existing data or models(meet-in-the-middle). Follow the steps below if you want to modify the default schema and/or map.

To start this process, you need a standard EJB-1.0 or 1.1 JAR file.

1. Run the ejbdeploy utility using the -codegen argument.
2. Modify the database schema.
3. Modify the map.
4. Generate deployment code for your enterprise beans byrunning the ejbdeploy utility again. This produces a deployed JAR thatcan be installed directly into WebSphere Application Server 4.0.

6.6.0.15.3:Modifying the schema document

A schema document is created during the top-down mappinggeneration done by the ejbdeploy tool. The schema document contains descriptions of tables and their columns. After completing the top-downoperation, the resulting schema may need to be modified to fit existing dataor models (meet-in-the-middle mapping). This topic discusses that process.

XMI vocabulary: Schema.rdbxmi

This topic discusses the structure and vocabulary of an XMI schemadocument. The following is a sample XMI code block from the schema.rdbxmi file that describes the BankAccount table, all itscolumns and constraints (Primary Key).

```
 <xmi:XMI xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI" xmlns:RDBSchema="RDBSchema.xmi">
<RDBSchema:RDBDatabase xmi:id="TopDownDB_ID" name="TopDownDB" schemata="USERID_ID"
tableGroup="BankAccount">    <dataTypeSet href="UDBV7_Primitives.xmi#SQLPrimitives_1"/>
</RDBSchema:RDBDatabase>   <RDBSchema:RDBSchema xmi:id="USERID_ID" name="USERID"
database="TopDownDB_ID" tables="BankAccount"/>   <RDBSchema:RDBTable xmi:id="BankAccount"
name="BankAccount" primaryKey="SQLReference_1" database="TopDownDB_ID"        schema="USERID_ID">
<namedGroup xsi:type="RDBSchema:SQLReference" xmi:id="SQLReference_1" members="RDBColumn_1"
table="BankAccount" constraint="Constraint_BankAccountPK"/>    <constraints
xmi:id="Constraint_BankAccountPK" name="BankAccountPK" type="PRIMARYKEY"
primaryKey="SQLReference_1"/>    <columns xmi:id="RDBColumn_1" name="accountNum" allowNull="false"
group="SQLReference_1">      <type xsi:type="RDBSchema:SQLExactNumeric" xmi:id="SQLExactNumeric_1">
<originatingType xsi:type="RDBSchema:SQLExactNumeric"
href="UDBV7_Primitives.xmi#SQLExactNumeric_1"/>      </type>    </columns>    <columns
xmi:id="RDBColumn_2" name="customerName">      <type xsi:type="RDBSchema:SQLCharacterStringType"
xmi:id="SQLCharacterStringType_1" characterSet="800" length="32">        <originatingType
xsi:type="RDBSchema:SQLCharacterStringType" href="UDBV7_Primitives.xmi#SQLCharacterStringType_3"/>
</type>    </columns>    <columns xmi:id="RDBColumn_3" name="balance">      <type
xsi:type="RDBSchema:SQLNumeric" xmi:id="SQLNumeric_1" precision="5" scale="0">
<originatingType xsi:type="RDBSchema:SQLNumeric" href="UDBV7_Primitives.xmi#SQLNumeric_3"/>
</type>    </columns>   </RDBSchema:RDBTable> </xmi:XMI>
```

XML element descriptions

RDBSchema:RDBDatabase - defines the name and id of the database.

RDBSchema:RDDTable - defines thename and id of the table.

RDBSchema:SQLConstraint - defines the constraints,this sample only includes the primary key constraint. The constraintpoints to another object that describes the type of constraint. In thiscase, the SQLReference "BankAccountPK" describes the primary key, and what members(columns) are included.

RDBSchema:RDBColumn - defines the column type by referencing a typeinstance in a sqlprimitives document. The primitives documents are located in the /plugins/sqlmodel/runtime/primitivesdirectory.

Changing type definitions

If you need to change between two closely-related types, such as, VARCHAR to LONGVARCHAR, you only need to change the type id in thehref. A column might looks like this:

```
<columnsxmi:id="RDBColumn_52" name="location">
  <type xsi:type="RDBSchema:SQLCharacterStringType"xmi:id="SQLCharacterStringType_20" characterSet="800" length="1">
    <originatingType xsi:type="RDBSchema:SQLCharacterStringType"
    href="ORACLEV8i_Primitives.xmi#SQLCharacterStringType_2"/>
  </type>
</columns>
```

In this case, the href is referring to the following type inORACLEV8i_Primitives.xmi:

```
<types xsi:type="RDBSchema:SQLCharacterStringType"
  xmi:id="SQLCharacterStringType_2" externalName="CHARACTERVARYING" name="VARCHAR2"
  dbcEnumType="12 domain="ORACLE_V8" requiredUniqueInstance="true"renderedString="VARCHAR2"
  typeEnum="CHARACTERVARYING"formatterClassName="com.ibm.etools.rdbschema.formatter.oracle.CharacterTextFormatter"
  characterSet="800" length="1"/>
```

If you just need to change the length, you can do so by directly changing thelength (seen in red in the code snippet above) on the type object. If you want to switch to one of the othercharacter types, you can change the type reference (seen in red in the code snippet above) to one of the other similar types in the primitives doc. For example, SQLCharacterStringType_1 to SQLCharacterStringType_6.As long as you are switching between closely-relatedtypes, you do not need to change the syntax of the <type>.

For numerics, the same thing will apply. You shouldonly have to modify the precision and scale. However, you can get the more specificnumeric type you want by switching between the SQLNUMERIC_ types.

Adding table columns

Adding a new column to the table is easy if you use the followingsteps:

1. Add the new column under the columns tag. Make sure the order ofthe column in the list corresponds to the order in the database.

```
<columns xmi:id="RDBColumn_1" name="accountNum" allowNull="false" group="SQLReference_1">
```

2. Add the column type.

```
<type xsi:type="RDBSchema:SQLExactNumeric" xmi:id="SQLExactNumeric_1">   <originatingType
xsi:type="RDBSchema:SQLExactNumeric" href="UDBV7_Primitives.xmi#SQLExactNumeric_1"/></type>
```

Make sure all of the id's chosen are unique in the entire XMIdocument.

More examples of type definitions

The following type definition examples were taken from UDBV6_Primitives.xmi. This and other type definition example files can befound in the following location: X:\..\eclipse\plugins\sqlmodel\runtime\primitives\

CHARACTER FOR BIT DATA	<types xmi:type="RDBSchema:SQLBinaryLargeObject" xmi:id="SQLBinaryLargeObject_3" externalName="BINARY LARGE OBJECT" name="CHARACTER () FOR BIT DATA" jdbcEnumType="-2" domain="DB2UDBNT_V61" requiredUniqueInstance="true" renderedString="CHARACTER" typeEnum="BINARYLARGEOBJECT" formatterClassName="com.ibm.etools.rdbschemagen.formatter.db2nt.BinaryTextFormatter" length="1"/>
VARCHAR FOR BIT DATA	<types xsi:type="RDBSchema:SQLBinaryLargeObject" xmi:id="SQLBinaryLargeObject_5" externalName="BINARY LARGE OBJECT" name="VARCHAR () FOR BIT DATA" jdbcEnumType="-3" domain="DB2UDBNT_V61" requiredUniqueInstance="true" renderedString="VARCHAR" typeEnum="BINARYLARGEOBJECT" formatterClassName="com.ibm.etools.rdbschema.formatter.db2nt.BinaryTextFormatter" length="1"/>
SMALLINT	<types xsi:type="RDBSchema:SQLExactNumeric" xmi:id="SQLExactNumeric_4" externalName="SMALLINT" name="SMALLINT" jdbcEnumType="5" domain="DB2UDBNT_V61" requiredUniqueInstance="false" renderedString="SMALLINT" typeEnum="SMALLINT" formatterClassName="com.ibm.etools.rdbschema.formatter.db2nt.SimpleTextFormatter"/>
FLOAT	<types xsi:type="RDBSchema:SQLApproximateNumeric" xmi:id="SQLApproximateNumeric_4" externalName="DOUBLE PRECISION" name="FLOAT" jdbcEnumType="8" domain="DB2UDBNT_V61" requiredUniqueInstance="false" renderedString="DOUBLE" typeEnum="DOUBLEPRECISION" formatterClassName="com.ibm.etools.rdbschema.formatter.db2nt.SimpleTextFormatter"/>
DOUBLE	<types xsi:type="RDBSchema:SQLApproximateNumeric" xmi:id="SQLApproximateNumeric_2" externalName="DOUBLE PRECISION" name="DOUBLE" jdbcEnumType="8" domain="DB2UDBNT_V61" requiredUniqueInstance="false" renderedString="DOUBLE" typeEnum="DOUBLEPRECISION" formatterClassName="com.ibm.etools.rdbschema.formatter.db2nt.SimpleTextFormatter"/>
LONG VARCHAR FOR BIT DATAoka	<types xsi:type="RDBSchema:SQLBinaryLargeObject" xmi:id="SQLBinaryLargeObject_8" externalName="BINARY LARGE OBJECT" name="LONG VARCHAR FOR BIT DATA" jdbcEnumType="-4" domain="DB2UDBNT_V61" requiredUniqueInstance="false" renderedString="LONG VARCHAR" typeEnum="BINARYLARGEOBJECT" formatterClassName="com.ibm.etools.rdbschema.formatter.db2nt.BinaryTextFormatter" length="0"/>

DATE	<types xsi:type="RDBSchema:SQLDate" xmi:id="SQLDate_1" externalName="DATE" name="DATE" jdbcEnumType="91" domain="DB2UDBNT_V61" requiredUniqueInstance="false" renderedString="DATE" typeEnum="DATE" formatterClassName="com.ibm.etools.rdbschema.formatter.db2nt.SimpleTextFormatter"/>
SMALLINT	<types xsi:type="RDBSchema:SQLExactNumeric" xmi:id="SQLExactNumeric_4" externalName="SMALLINT" name="SMALLINT" jdbcEnumType="5" domain="DB2UDBNT_V61" requiredUniqueInstance="false" renderedString="SMALLINT" typeEnum="SMALLINT" formatterClassName="com.ibm.etools.rdbschema.formatter.db2nt.SimpleTextFormatter"/>
DECIMAL (20,2)	<types xsi:type="RDBSchema:SQLNumeric" xmi:id="SQLNumeric_1" externalName="NUMERIC" name="DECIMAL" jdbcEnumType="2" domain="DB2UDBNT_V61" requiredUniqueInstance="true" renderedString="NUMERIC" typeEnum="NUMERIC" formatterClassName="com.ibm.etools.rdbschema.formatter.db2nt.NumericTextFormatter" precision="5" scale="0"/>
REAL	<types xmi:type="RDBSchema:SQLApproximateNumeric" xmi:id="SQLApproximateNumeric_1" externalName="REAL" name="REAL" jdbcEnumType="7" domain="DB2UDBNT_V61" requiredUniqueInstance="false" renderedString="REAL" typeEnum="REAL" formatterClassName="com.ibm.etools.rdbschemagen.formatter.db2nt.SimpleTextFormatter"/>
INTEGER	<types xsi:type="RDBSchema:SQLExactNumeric" xmi:id="SQLExactNumeric_1" externalName="INTEGER" name="INTEGER" jdbcEnumType="4" domain="DB2UDBNT_V61" requiredUniqueInstance="false" renderedString="INTEGER" typeEnum="INTEGER" formatterClassName="com.ibm.etools.rdbschema.formatter.db2nt.SimpleTextFormatter"/>
BIGINT	<types xsi:type="RDBSchema:SQLExactNumeric" xmi:id="SQLExactNumeric_3" externalName="INTEGER" name="BIGINT" jdbcEnumType="-5" domain="DB2UDBNT_V61" requiredUniqueInstance="false" renderedString="BIGINT" typeEnum="INTEGER" formatterClassName="com.ibm.etools.rdbschema.formatter.db2nt.SimpleTextFormatter"/>
TIME	<types xsi:type="RDBSchema:SQLTime" xmi:id="SQLTime_1" externalName="TIME" name="TIME" jdbcEnumType="92" domain="DB2UDBNT_V61" requiredUniqueInstance="true" renderedString="TIME" typeEnum="TIME" formatterClassName="com.ibm.etools.rdbschema.formatter.db2nt.SimpleTextFormatter" precision="2" timezone="false"/>
TIMESTAMP	<types xsi:type="RDBSchema:SQLExactNumeric" xmi:id="SQLExactNumeric_1" externalName="INTEGER" name="INTEGER" jdbcEnumType="4" domain="DB2UDBNT_V61" requiredUniqueInstance="false" renderedString="INTEGER" typeEnum="INTEGER" formatterClassName="com.ibm.etools.rdbschema.formatter.db2nt.SimpleTextFormatter"/>
BIGINT	<types xsi:type="RDBSchema:SQLExactNumeric" xmi:id="SQLExactNumeric_3" externalName="INTEGER" name="BIGINT" jdbcEnumType="-5" domain="DB2UDBNT_V61" requiredUniqueInstance="false" renderedString="BIGINT" typeEnum="INTEGER" formatterClassName="com.ibm.etools.rdbschema.formatter.db2nt.SimpleTextFormatter"/>
CLOB	<types xmi:type="RDBSchema:SQLCharacterLargeObject" xmi:id="SQLCharacterLargeObject_1" externalName="CHARACTER LARGE OBJECT" name="CLOB" jdbcEnumType="2005" domain="DB2UDBNT_V61" requiredUniqueInstance="true" renderedString="CLOB" typeEnum="CHARACTERLARGEOBJECT" formatterClassName="com.ibm.etools.rdbschemagen.formatter.db2nt.LargeObjectTextFormatter" characterSet="800" length="1" multiplier="K"/>
BLOB	<types xsi:type="RDBSchema:SQLBinaryLargeObject" xmi:id="SQLBinaryLargeObject_1" externalName="BINARY LARGE OBJECT" name="BLOB" jdbcEnumType="2004" domain="DB2UDBNT_V61" requiredUniqueInstance="true" renderedString="BLOB" typeEnum="BINARYLARGEOBJECT" formatterClassName="com.ibm.etools.rdbschema.formatter.db2nt.LargeObjectTextFormatter" length="1" multiplier="K"/>

6.6.0.15.4: Modifying the map document

A map consists of mapping the model attributes andrelationships to the schema. The map connects the object modeldescription with the schema description. Once the map is defined, thepersistence support for your data store is created during codegeneration.

XMI vocabulary: Map.mapxmi

After completing the top-down operation, the resulting map may need tobe modified to fit existing data or models.

This topic discusses the structure and vocabulary of an XMI mapdocument. The following is a sample XMI code block from themap.mapxmi file that shows the mapping between the BankAccount CMP, andthe BANKACCOUNT table.

```
<ejbrdbmapping:EjbRdbDocumentRoot xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI"
xmlns:ejbrdbmapping="ejbrdbmapping.xmi" xmlns:Mapping="Mapping.xmi" xmlns:ejb="ejb.xmi"
xmlns:RDBSchema="RDBSchema.xmi" xmlns:java="java.xmi"    xmi:id="EjbRdbDocumentRoot_1"
outputReadOnly="false" topToBottom="true">  <helper xsi:type="ejbrdbmapping:RdbSchemaProperies"
xmi:id="RdbSchemaProperies_1" primitivesDocument="DB2UDBNT_V71"/>  <inputs xsi:type="ejb:EJBJar"
href="META-INF/ejb-jar.xml#ejb-jar_ID"/>  <outputs xsi:type="RDBSchema:RDBDatabase"
href="META-INF/Schema/Schema.dbxmi#TopDownDB_ID"/>  <nested xsi:type="ejbrdbmapping:RDBEjbMapper"
xmi:id="RDBEjbMapper_1">    <helper xsi:type="ejbrdbmapping:PrimaryTableStrategy"
xmi:id="PrimaryTableStrategy_1">      <table href="META-INF/Schema/Schema.dbxmi#BankAccount"/>
</helper>    <inputs xsi:type="ejb:ContainerManagedEntity" href="META-INF/ejb-jar.xml#BankAccount"/>
<outputs xsi:type="RDBSchema:RDBTable" href="META-INF/Schema/Schema.dbxmi#BankAccount"/>    <nested
xmi:id="BankAccount_accountNum---BankAccount_accountNum">      <inputs xsi:type="ejb:CMPAttribute"
href="META-INF/ejb-jar.xml#BankAccount_accountNum"/>      <outputs xsi:type="RDBSchema:RDBColumn"
href="META-INF/Schema/Schema.dbxmi#RDBColumn_1"/>      <typeMapping
href="JavatoDB2UDBNT_V71TypeMaps.xmi#int-INTEGER"/>    </nested>    <nested
xmi:id="BankAccount_customerName---BankAccount_customerName">      <helper
xsi:type="ejbrdbmapping:EJBConverter" xmi:id="EJBConverter_1">        <targetClass
href="java:/java.lang#String"/>        <transformerClass
href="java:/com.ibm.vap.converters#VapTrimStringConverter"/>      </helper>      <inputs
xsi:type="ejb:CMPAttribute" href="META-INF/ejb-jar.xml#BankAccount_customerName"/>      <outputs
xsi:type="RDBSchema:RDBColumn" href="META-INF/Schema/Schema.dbxmi#RDBColumn_2"/>      <typeMapping
href="JavatoDB2UDBNT_V71TypeMaps.xmi#String-VARCHAR"/>    </nested>    <nested
xmi:id="BankAccount_balance---BankAccount_balance">      <inputs xsi:type="ejb:CMPAttribute"
href="META-INF/ejb-jar.xml#BankAccount_balance"/>      <outputs xsi:type="RDBSchema:RDBColumn"
href="META-INF/Schema/Schema.dbxmi#RDBColumn_3"/>      <typeMapping
href="JavatoDB2UDBNT_V71TypeMaps.xmi#BigDecimal-DECIMAL"/>    </nested>  </nested>  <typeMapping
xsi:type="Mapping:MappingRoot"
href="JavatoDB2UDBNT_V71TypeMaps.xmi#Java_to_DB2UDBNT_V71_TypeMaps"/></ejbrdbmapping:EjbRdbDocumentRoot>
```

XML element descriptions

helper - defines helper objects that are used to set schemaproperties, table strategy, and as converters between attribute and column types.

inputs - definesCMP references using an href tag that uses a path statement relative to theMETA-INF directory.

outputs - defines database table referencesusing an href tag that uses a path statement relative to the META-INF directory.

nested - defines children mappings forattributes and columns. These mappings use the mapping object todescribe the inputs, outputs, and helpers.

typeMapping - an optional element in the map document.

Adding mappings

The procedure for adding a new mapping is easy if you use the followingsteps:

1. Under the nested tag, add a new Mappingobject.

   ```
   <helper xsi:type="ejbrdbmapping:RdbSchemaProperies" xmi:id="RdbSchemaProperies_1"
   primitivesDocument="DB2UDBNT_V71"/> <inputs xsi:type="ejb:EJBJar"
   href="META-INF/ejb-jar.xml#ejb-jar_ID"/> <outputs xsi:type="RDBSchema:RDBDatabase"
   href="META-INF/Schema/Schema.dbxmi#TopDownDB_ID"/>  <nested xsi:type="ejbrdbmapping:RDBEjbMapper"
   xmi:id="RDBEjbMapper_1">    <helper xsi:type="ejbrdbmapping:PrimaryTableStrategy"
   xmi:id="PrimaryTableStrategy_1">      <table href="META-INF/Schema/Schema.dbxmi#BankAccount"/>
   </helper>
   ```

2. Add the CMPAttribute and RDBColumn references.

   ```
    <nested xmi:id="BankAccount_accountNum---BankAccount_accountNum">   <inputs
   xsi:type="ejb:CMPAttribute" href="META-INF/ejb-jar.xml#BankAccount_accountNum"/>   <outputs
   xsi:type="RDBSchema:RDBColumn" href="META-INF/Schema/Schema.dbxmi#RDBColumn_1"/>   <typeMapping
   ```

```
    href="JavatoDB2UDBNT_V71TypeMaps.xmi#int-INTEGER"/> </nested>
```

Make sure all of the id's chosen are unique in the entiredocument.

6.6.0.15.5: Implementing custom finder helpers for CMP entity beans

To work with custom finder helpers for CMP entity beans, there are two tasks thatyou need to complete:

- Implement the custom finder helpers
- Add the custom finder helpers to the home interface

The task of adding custom finder helpers to the home interface is found in theWebSphere Studio Application Developer help topic "Adding methods to thehome and remote interfaces". The task of implementing custom finder helpers is described in the remainderof this topic in the following sections:

- Introduction to working with custom finder helpers
- Implementing custom finders using EJB query language and SQL
- Defining custom finder helpers in deployment descriptor extension documents
- The AutoWorld example
    - Using the finder helper interface

Introduction to working with custom finder helpers

To implement custom finder helpers, you can use one of two supported querylanguages:

- EJB query language
- SQL

EJB query language is recommended for implementing custom finder helpers, butthe use of SQL is still supported.

To implement custom finder helpers, you should use a deployment descriptorextension document rather than a helper finder interface. Both EJB query language strings and SQL query strings and method declarationscan be used in a deployment descriptor extension document.

The use of helperfinder interfaces has been deprecated but is still supported to someextent. If you are working with existing EJB 1.0 JARfiles, you can continue to define SQL query strings or method declarations inthe helper finder interface. (The SQL query string is actually a field on theinterface describing the full SELECT statement or just the WHERE clause.)

However, for any new development work that requires you to work with EJB JARfiles at a level higher than 1.0, it is required that you use a deploymentdescriptor extension document rather than the finder helper interface to defineyour queries or method declarations.

Implementing custom findersusing EJB query language and SQL

The EJB query language defines finder methods for entity beans with container-managed persistence. The definition uses a language based on SQL that allowssearches on the persistent attributes of an Enterprise Java Bean and associatedbean attributes. The query language is independent of the bean's mapping to arelational datastore and is portable. The query language is compiled into SQL atdeployment time based on the schema mapping for the bean.

An EJB query is a string that contains

- an optional SELECT clause that specifies the EJB objects to return
- a FROM clause that names the bean collections
- an optional WHERE clause that contains search predicates over the collections
- an optional ORDER BY clause that specifies the ordering of the result collection.

An EJB query also contains input parameters that correspond to the argumentsof the finder method.

Additional information is found in the WebSphere Studio Application Developer help topic "EJBquery language - overview".

There are four types of custom finders that are currently supported for enterprise beans and that can be used in combination:

- SQL SELECT
- SQL WHERE
- SQL Method
- EJB query language

The SELECT, WHERE, and method custom finders are standard SQL techniques used to query a database column.

For each finder method that is defined in the EJB home interface (other than findByPrimaryKey and those finder methods generated to support associations), one of the following queries or declarations must be defined in the deployment descriptor extension document or in the finder helper interface (in file *beanClassName*FinderHelper.java):

- An SQL SELECT or WHERE query tag (for extension documents) or string (for finder helper interfaces)
- A method declaration (for method custom finders)

- An EJB query language tag or string

The return type java.util.Enumeration or java.util.Collection on the finder in the home interface indicates that this finder may return more than one bean. Using the remote interface as the return type indicates that a single bean is returned. This is true for all of the supported types of custom finders. The code generated into the persister handles this distinction.

For SQL SELECT, WHERE, and method custom finders, there may be situations where the finder methods access different databases. In this case, it is necessary to ensure that SQL compatibility is maintained across the different databases. For instance, it is possible that the SQL syntax used by each database is different. In these situations, use the SQL extensions defined by JDBC to resolve the database differences. For example, assume that you are developing a CMP entity bean that requires a finder method that involves a timestamp/date field. Also assume that this bean will be deployed to DB2 and Oracle databases. The problem is that the format of the timestamp/date fields in DB2 and Oracle are different, which causes difficulties in defining one WHERE clause for use with both DB2 and Oracle. The solution to this particular problem is to use the SQL Escape sequence.

Defining custom finder helpers in deployment descriptor extension documents

If you are working with existing EJB 1.0 JAR files, you can continue todefine SQL query strings, method declarations, or EJB query language strings inthe helper finder interface. However, this mechanism for defining queries hasseveral restrictions, such as the inability to create polymorphic finders.Therefore, for any new development work, it is recommended that you use adeployment descriptor extension document rather than the finder helper interfaceto define your queries and method declarations. The extension document you needto use is saved in the JAR file and is named:

META-INF/ibm-ejb-jar-ext.xmi

The extension document contains query tags rather than query strings. You canedit the extension document much like a deployment descriptor file. Thefollowing example uses standard SQL and EJB query language finder syntax andwill help you understand the format of the extensions document. You can use theexample as a template and copy/paste the tagging for your own use.

```
<ejbext:EJBJarExtension xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI"
xmlns:ejbext="ejbext.xmi" xmlns:ejb="ejb.xmi" xmlns:ecore="ecore.xmi"
xmi:id="ejb-jar_ID_Ext"><ejbJar href="META-INF/ejb-jar.xml#ejb-jar_ID"/><ejbExtensions
xsi:type="ejbext:ContainerManagedEntityExtension"      xmi:id="Department_Ext"
name="Department"><enterpriseBean xsi:type="ejb:ContainerManagedEntity"
href="META-INF/ejb-jar.xml#Department"/><finderDescriptors
xsi:type="ejbext:FullSelectFinderDescriptor"
xmi:id="Department_findAllQueryString_FullSelect_ID"      selectStatement="SELECT * FROM
T1.DEPARTMENT"><finderMethodElements xmi:id="MethodElement_10" name="findAll"      parms=""
type="Home"><enterpriseBean xsi:type="ejb:ContainerManagedEntity"
href="META-INF/ejb-jar.xml#Department"/></finderMethodElements></finderDescriptors><finderDescriptors
xsi:type="ejbext:WhereClauseFinderDescriptor"     xmi:id="Department_findByName_WhereClause_ID"
whereClause="T1.NAME LIKE ?"><finderMethodElements xmi:id="MethodElement_11" name="findByName"
parms="java.lang.String" type="Home"><enterpriseBean xsi:type="ejb:ContainerManagedEntity"
href="META-INF/ejb-jar.xml#Department"/></finderMethodElements></finderDescriptors><finderDescriptors
xsi:type="ejbext:EjbqlFinderDescriptor"     xmi:id="Department_findByName_ejbql_ID"
ejbqlQueryString="select e from DepartmentBean name like ?"><finderMethodElements
xmi:id="MethodElement_12" name="findByEjbName"     parms="java.lang.String"
type="Home"><enterpriseBean xsi:type="ejb:ContainerManagedEntity"
href="META-INF/ejb-jar.xml#Department"/></finderMethodElements></finderDescriptors><finderDescriptors
xsi:type="ejbext:UserFinderDescriptor"
xmi:id="Department_findSpecial_User_ID"><finderMethodElements xmi:id="MethodElement_13"
name="findSpecial"type="Home"><enterpriseBean xsi:type="ejb:ContainerManagedEntity"
href="META-INF/ejb-jar.xml#Department"/></finderMethodElements></finderDescriptors></ejbExtensions></ejbJar>
```

For each finder method in the extension document, you need to provide afinderDescriptor element that consists of two parts: the finderMethodElement,which identifies the method to which the finder descriptor applies, and thequery string, which can be specified as one of the following types:

- SQL "whereClause" (corresponds to finder helper interface strings)
- SQL "selectStatement" (corresponds to finder helper interface strings)
- EJB query language string (specifies the query in terms of the EJB 2.0 EJB Query Language)

These three types of query string have the following unique xsi:type:

- ejbext:WhereClauseFinderDescriptor
- ejbext:FullSelectFinderDescriptor
- ejbext:EjbqlFinderDescriptor

The name of the class that contains the custom finder method is *beanClassName*FinderObject,where *beanClassName* is the name of the implementation class for the bean.The FinderObject class must be in the same package as the bean implementationclass.

In the extensions document, you can also employ a user finder descriptor,which the deployment tool uses to determine the finder implementation methodprovided by the user. In the above example, the last finderDescriptors tagspecifies a user finder descriptor. This descriptor simply serves as aplaceholder, because it contains no "string" like the otherdescriptors. The name of the custom finder is there, however. It is namedfindSpecial. As a result, in the *beanClassName*FinderObject class providedby the bean developer, there must be a corresponding method called findSpecial.

Since two separate findSpecial methods are used, care must be taken tounderstand the difference between the two. The essential difference is that thebean developer defines one findSpecial method in the home interface and theother findSpecial method in the FinderObject class. Both take the samearguments, but they return different types and serve different purposes. Theyare given the same name as a convenient way to tie them together.

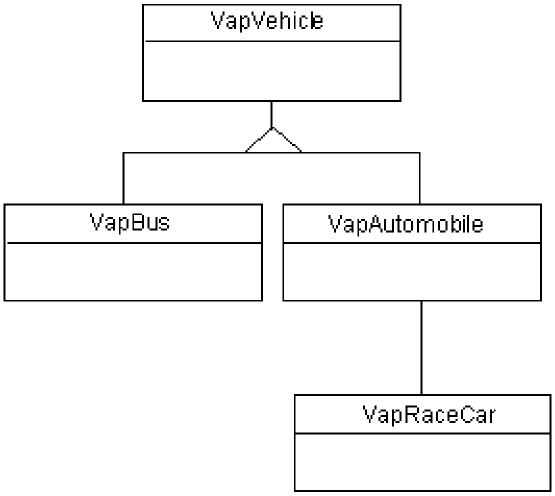
When a bean developer needs a custom finder, the developer defines it in thehome interface. For CMP entity beans, the EJB Deploy Tool generates the codethat implements the custom finder. For all but the custom method finder, allthat is needed is a string. For the custom method finder, the bean developerneeds to actually implement a method in the FinderObject. The deploy tool stillgenerates the code that implements the findSpecial custom finder, but thatgenerated code will call the findSpecial method in the FinderObject.

During deployment, if a JAR file contains finder helper interfaces, the EJBDeploy Tool will migrate the appropriate SQL strings into a field containedwithin the extensions document (ibm-ejb-jar-ext.xmi). This string is what isused during deployment to emit code contained within the emitted JDBC persister.In WebSphere Application Server, Version 3.5, the finder helper interface wasused directly. Once the information exists in the extensions document, thatinformation will be used and any subsequent changes to the finder helperinterface will be ignored. This field can be modified by hand or by using theApplication Assembly tool.

Note that when enterprise beans are exported from VisualAge for Java,Enterprise Edition, Version 4.0 using the Export Tool for Enterprise Java Beans1.1, or when 1.0 EJB JAR files are deployed with the deployment tool, the findermeta data is migrated to the new form from the finder helper interfaces. If theenterprise beans are exported using the Export Tool for Enterprise Java Beans1.1, the redundant classes are filtered from the exported JAR. If the enterprisebeans are not exported using the Export Tool for Enterprise JavaBeans 1.1 and are imported along with the redundant classes, the classes aresimply ignored.

The AutoWorld example

To introduce this AutoWorld example, considerthe following hierarchy of CMP entity beans:



Outside of this CMP bean hierarchy, there are two other existing CMP entitybeans: VapGarage and VapMotorVehiclePart. An association exists betweenVapGarage and VapVehicle, in which a garage can hold many vehicles. Anotherassociation exists between VapMotorVehiclePart and VapVehicle, in which avehicle can contain many parts.

In the remainder of this topic, the AutoWorld example is used to show you howto work with queries defined in the finder helper interface.

Using the finder helperinterface

In this section, the AutoWorld example is used to show you how the followingcustom finders are defined in the finder helper interface:

- SQL SELECT custom finders
- SQL WHERE custom finders
- Method custom finders

SQL SELECT custom finders

A SELECT custom finder is used to enter an entire select statement in the finder helper interface to define the SQL query.

The use of SELECT custom finders is supported for compatibility with earlier releases. Using SELECT custom finders is discouraged in this and future releases.

SQL WHERE custom finders

A custom finder in which you enter only the filtering WHERE clause into the finder helper interface is called a WHERE custom finder. Using the same garage example as before, the finder helper interface would look like this:

```
public interface VapGarageBeanFinderHelper {    public final static String
findCapacityGreaterThanWhereClause =          "T1.CAPACITY > ?"; }
```

Notice that any dependency on the shape of the results is removed from the string. Two dependencies still exist:

- The name of the column (CAPACITY)
- The alias for the table (T1)

The name of the column would change only if you took action to change it.

The alias for the table will be the same from one generation to the next unless tables are added to or removed from the mapping. In single-table cases, this may not seem significant (the alias is always T1). When multiple tables are used, this is very important.

For example, consider the VapVehicle hierarchy. When it is mapped using root/leaf inheritance mapping, there is at least one table for each class in the hierarchy, and the query into which the WHERE clause would be inserted would have multiple subselects. The WHERE clause would be inserted into each subselect, and it would be valid SQL syntax for each subselect to use a different alias for the same table. However, our query generation ensures that the same table has the same alias across the entire query. If it did not, this WHERE clause substitution technique could not work. In these cases, the code generated into the persister knows to inject any finder parameters into the query multiple times.

There is one very important restriction when using this form of custom finder in a mapped hierarchy: The WHERE clause can only reference tables that map the bean in which the finder is defined or tables that map one of bean's parent beans.

For example, a WHERE clause in the VapVehicleBeanFinderHelper can only reference columns in the table used to map the VapVehicle bean. However, a WHERE in the VapAutomobileBeanFinderHelper can reference columns from either the table that maps VapAutomobile or VapVehicle (its superclass).

Any table references in your handwritten SQL code must match the table aliases set up in the genericFindSqlString field. This is declared in the enterprise bean's generated persister.

As in the SELECT custom form, the number of finder parameters must match the number of injection points (the **?** characters) in the WHERE clause. Also, as in the SELECT form, the type of the parameter will be used to determine which java.sql.PreparedStatement set call will be used to inject each parameter. The parameter types must be compatible with the column types. If the type is an object type and the parameter is null, a setNull call will be used.

For example, the home interface may contain the following method:

```
public java.util.Enumeration findGreaterThan (int threshold) throws   java.rmi.RemoteException,
javax.ejb.FinderException;
```

In this finder helper interface, the WHERE custom finder is one of the forms that you can provide. For example (line broken for publication):

```
public static final String findGreaterThanWhereClause =    "T1.VALUE > ?";
```

Note, however, that if you have an SQL statement that contains no WHERE clause, such as `SELECT * FROM MYTABLE`, you should use a query string that always evaluates to *true*. For example:

```
public static final String findALLWhereClause = "1 = 1";
```

Method custom finders

A custom finder in which you enter a method signature into the finder helper interface is called a method custom finder. It is the most flexible type of custom finder, but it requires more work on your part. Using the same garage example as before, the finder helper interface would look like this:

```
public interface VapGarageBeanFinderHelper {   public java.sql.PreparedStatement
findCapacityGreaterThan(int threshold)       throws Exception;}
```

Unlike the SELECT and WHERE forms, however, this is not enough for method custom finders. An implementation of this method is needed. You provide your implementation of the method in a class that follows these rules:

- The name of the class is *beanClassName*FinderObject (VapGarageBeanFinderObject, in this example) and is in the same package as the bean class.
- The class must extend com.ibm.vap.finders.VapEJSJDBCFinderObject and must implement the bean's finder helper interface.
- Any table references in your handwritten SQL code must match the table aliases set up in the genericFindSqlString field. This is declared in the enterprise bean's generated persister.

To finish off our example, the finder object would look like this.

```
/** * Implementation class for methods in * VapGarageBeanFinderHelper. */ public class
VapGarageBeanFinderObject extends   com.ibm.vap.finders.VapEJSJDBCFinderObject implements
VapGarageBeanFinderHelper {    public java.sql.PreparedStatement   findCapacityGreaterThan(int
threshold)    throws Exception {       PreparedStatement ps = null;       int mergeCount =
getMergedWhereCount();       int columnCount = 1;       ps = getMergedPreparedStatement("T1.CAPACITY
> ?");       for (int i=0; i<(columnCount*mergeCount); i=i+columnCount) {          ps.setInt(i+1,
threshold);       }       return ps;    }}
```

In the case of any method custom finder, the generated persister uses your implementation to create the PreparedStatement to be executed. The persister will execute the PreparedStatement and handle the results. The implementation needs help from the persister to make sure the result set for the query has the correct shape. The com.ibm.vap.finders.VapEJSJDBCFinderObject base class provides several important helper methods, some of which are shown in the above example. In the following table, the complete set of helper methods are listed and described:

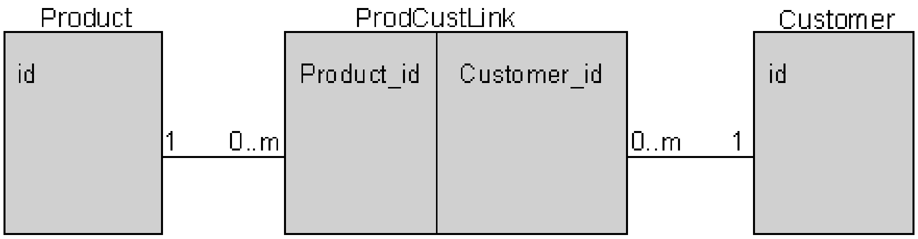
Method	Description
getMergedPreparedStatement	Takes a WHERE clause and returns a PreparedStatement with the WHERE clause merged into the appropriate places. The PreparedStatement will have the correct result set shape.
getMergedWhereCount	Returns the number of times the WHERE clause is merged into the PreparedStatement. This is needed to know how many times to inject your query parameters into the PreparedStatement.
getPreparedStatement	Takes a complete query string and returns a PreparedStatement. This can be used if for some reason you need to do your own WHERE clause merging. This should be a very rare case. The next two functions are provided to help in these extreme cases.
getGenericFindSqlString	Returns the query string into which WHERE clauses are merged.
getGenericFindInsertPoints	Returns an array of integers that defines the point or points in the getGenericFindSqlString returned query string at which the WHERE clause is merged. The first point in the array is the last point in the string. It is usually best to merge from the end of the query string since a merge at the end will not change the location of merges earlier in the string. The size of the array is the same as the value returned from getMergedWhereCount.

This is a rather simple case that can better be handled by a WHERE custom finder. More complex examples are possible that a WHERE custom finder simply could not handle. For example, suppose you wanted a finder that took a more complex object and injected it into multiple columns in a WHERE clause. You could end up with a finder method that looked like this:

```
public java.sql.PreparedStatement   findWithComplexObject(BigObject big) throws Exception {
PreparedStatement ps = null;    int mergeCount = getMergedWhereCount();    int columnCount = 3;
int anInt = big.getAnInt();    String aString = big.getAString();    String aLongAsString =
com.ibm.vap.converters.VapStringToLongConverter.      singleton().dataFrom(big.getLongObject());
   ps = getMergedPreparedStatement("(T1.ANINT > ?) AND        (T1.ASTRING = ?) AND (T2.ALONGSTR <
?)");    for (int i=0; i<(columnCount*mergeCount); i=i+columnCount) {        ps.setInt(i+1, anInt);
       if (aString == null)           ps.setNull(1, java.sql.Types.VARCHAR);       else
ps.setString(i+2, aString);        if (aLongAsString == null)           ps.setNull(1,
java.sql.Types.VARCHAR);       else          ps.setString(i+3, aLongAsString);    }    return ps; }
```

Even more complex examples are possible. For instance, an object could be passed that contains the WHERE clause (or instructions on how to create it) in addition to the data. Or, there could be multiple parameters, each representing different conditions in the WHERE clause.

The following example is a logical representation of how a many-to-many association could be accomplished using a complex method custom finder. The example involves a many-to-many association between Product and Customer beans, using a intermediary bean (ProdCustLink) and two 1:m associations:



You can write method custom finders to span the relationship in either direction with just one method call. For this example, consider one direction only: a finder in Customer that retrieves all Customer instances that are associated with a given product key.

Customer's home interface contains the appropriate method signature, as follows:

```
java.util.Enumeration   findCustomersByProduct(prod.cust.code.ProductKey inKey)   throws
java.rmi.RemoteException, javax.ejb.FinderException;
```

Customer's finder helper interface contains the signature for the corresponding finder method:

```
public java.sql.PreparedStatement   findCustomersByProduct(prod.cust.code.ProductKey inKey)
throws Exception;
```

The finder object (CustomerBeanFinderObject) builds and caches the query string for the finder as well as implements the finder method.

```
public class CustomerBeanFinderObject    extends com.ibm.vap.finders.VapEJSJDBCFinderObject
implements CustomerBeanFinderHelper {   private String cachedFindCustomersByProductQueryString =
null;   .   .   .}
```

Through lazy initialization in the finder object, the accessor method for the query-string field builds up the query string by first merging the WHERE condition into the query template and then adding a reference to the intermediate table into the FROM clause.

The first half of the accessor method uses a genericFindInsertPoints array to locate and update each WHERE clause. Then, the second half of the method counts forward from the beginning of each FROM clause, inserts the reference to the intermediate table into the query string as needed, and updates the query-string field.

```
protected String getFindCustomersByProductQueryString() {   if
(cachedFindCustomersByProductQueryString == null) {      // Do the WHERE first    // so that the
genericFindInsertPoints are correct.      int i;      int[] genericFindInsertPoints =
getGenericFindInsertPoints();      StringBuffer sb = new StringBuffer(getGenericFindSqlString());
for (i = 0; i < genericFindInsertPoints.length; i++) {         sb.insert(genericFindInsertPoints[i],
"(T1.id = T2.Customer_id) AND (T2.Product_id = ?)");      }   // Make sure to update every FROM
clause.      String soFar = sb.toString();      int fromOffset = soFar.06060015_indexOf(" FROM ");
while (fromOffset != -1) {         sb.insert((fromOffset+5)," ProdCustLink T2, ");         soFar =
sb.toString();         fromOffset = soFar.06060015_indexOf(" FROM ", (fromOffset+5));
}      cachedFindCustomersByProductQueryString = sb.toString();   }   return
cachedFindCustomersByProductQueryString;}
```

After this method call, the query string looks something like the following:

```
 SELECT <columns> FROM ProdCustLink T2, CUSTOMER T1     WHERE((T1.id = T2.Customer_id) AND
(T2.Product_id = ?))
```

Also in the finder object, the implemented finder uses the query string to create a PreparedStatement. Last but not least, the product ID value is added into each WHERE clause by using the superclass method getMergedWhereCount() in the iteration loop.

```
public java.sql.PreparedStatement    findCustomersByProduct(ProductKey inKey)    throws
java.lang.Exception {   // Get the full query string and make a PreparedStatement.
java.sql.PreparedStatement ps =       getPreparedStatement(getFindCustomersByProductQueryString());
// Inject the product id parameter into each merged WHERE clause.   for (int i = 0; i <
getMergedWhereCount(); i++) {      if (inKey != null)         ps.setInt(i+1, inKey.id);      else
ps.setNull(i+1, 4);   }   return ps;}
```

6.6.0.16: Dynamic fragment cache configuration

Dynamic fragment caching can be enabled and configured using two XML configuration files, or through WebSphere Application Server's administrative console. The articles in this section describe building the XML files, although all the values discussed (except where specifically documented otherwise) have identical counterparts in the administrative console.

To enable dynamic caching, you can build two XML configuration files:

- ***dynacache.xml*** - the global caching administration file
- ***servletcache.xml*** - the individual cache policy configuration file

Both XML files use UTF-8 (the 8-bit Universal Character Set transformation format) character encoding.

6.6.0.16.1: Global administration

You can enable and configure dynamic fragment caching through a XML configuration fileor through the administrative console.

XML configuration file

The *dynacache.xml* file enables and configures dynamic fragment (also known as servlet) caching. If this file is not found in the *`product_installation`*`\properties` directory, the Web container processes servlets in the usual manner. However, if this file is found in the *`product_installation`*`\properties` directory, then dynamic fragment caching is enabled.

Within this file, users configure the overall operation of the cache, such as its size, and register the external caches that are used by the application.

> **Note:** The ***dynacache.xml*** file is read only once, at WebSphere Application Server startup.So if changes are made to the file, WebSphere Application Server must be restarted for the changes to take effect.

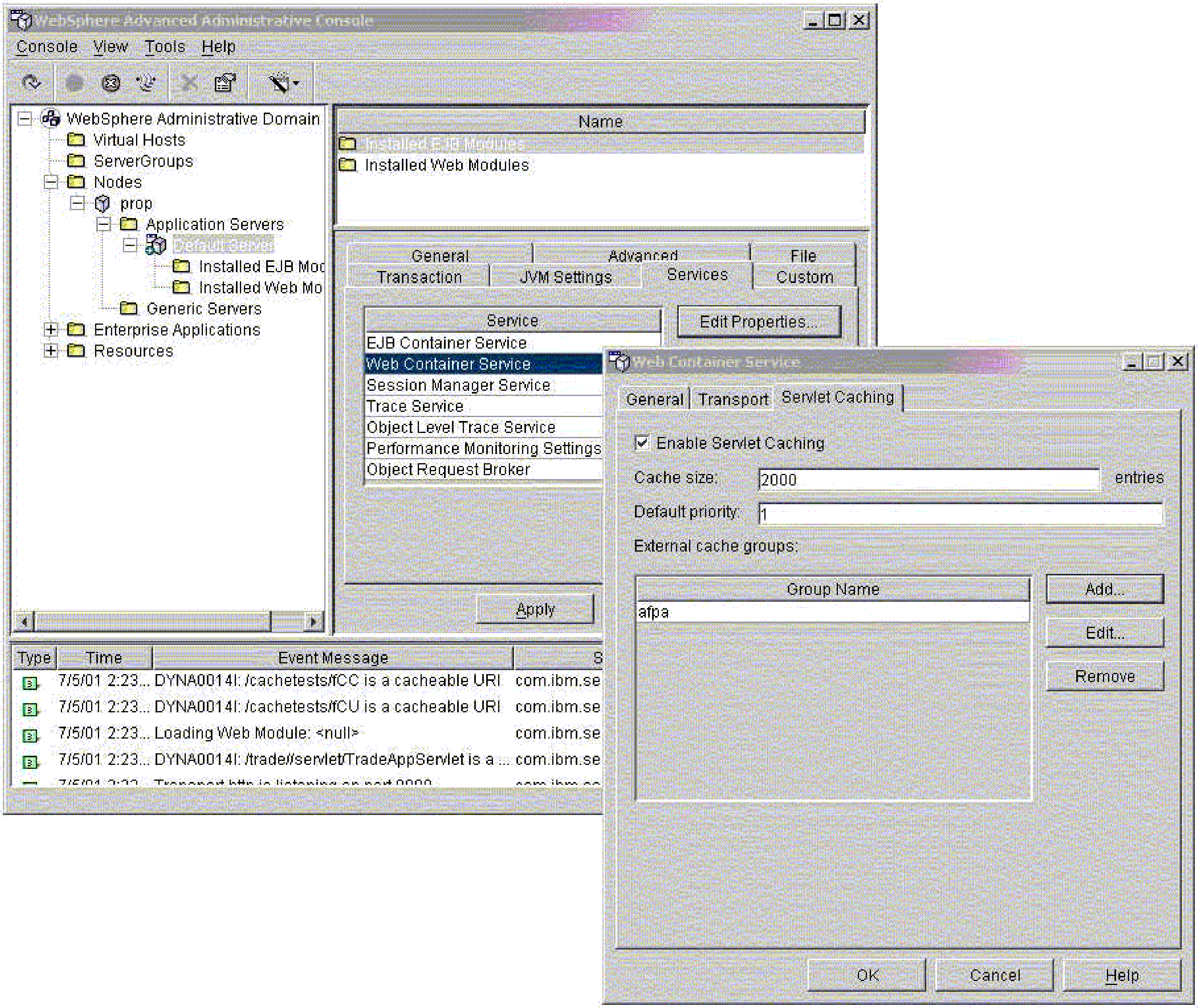
The DTD for the *dynacache.xml* file is specified in the *dynacache.dtd* file, which ships with WebSphere Application Server and is located in the*`product_installation`*`\properties` directory. The *dyncache.dtd* file defines the root element, <cacheUnit>, and should beincluded in the *dynacache.xml* file through the DOCTYPE declaration.

The beginning of the *dynacache.xml* should have the following processing instructions:

```
<?xml version="1.0"?><!DOCTYPE cacheUnit SYSTEM "dynacache.dtd">
```

Console configuration

You can also enable dynamic fragment caching through the administrative console.Cache properties are located under a server's Web Container:



☑ Enable Servlet Caching

Cache size: 2000 entries

Default priority: 1

External cache groups:

Group Name
afpa

```
Type  Time          Event Message                                          S
      7/5/01 2:23... DYNA0014I: /cachetests/fCC is a cacheable URI          com.ibm.se
      7/5/01 2:23... DYNA0014I: /cachetests/fCU is a cacheable URI          com.ibm.se
      7/5/01 2:23... Loading Web Module: <null>                             com.ibm.se
      7/5/01 2:23... DYNA0014I: /trade//servlet/TradeAppServlet is a ...    com.ibm.se
```

Description of elements in the *dynacache.xml* file

The elements defined in the *dynacache.xml* file provide the following two categories of function:

- Configuring global cache operations
- Controlling external caches

Configuring global cache operations

The root <cacheUnit> element contains one <cache> element, which provides global settings for all Application Servers on a node.

Cache

`<cache size="entries" priority="default_priority"/>`

entries: An integer defining the maximum number of entries the cache will hold. Values are usually in the thousands, with no set maximum or minumum

The size of the cache is limited not by the amount of available memory (for example, 20 megabytes), but by the number of distinct elements the available memory can hold. The cache only adds a few bytes of overhead for each entry (less than 32 bytes), so the size of an entry in memory is effectively equal to the amount of data being stored in that entry.

default_priority: An integer that defines the default priority for cacheable servlets defined in servletcache.xml. The recommended value for *default_priority:* is one.

Priority is an extension of the Least Recently Used (LRU) caching algorithm. It represents the number of cycles through the LRU algorithm an entry is guaranteed to stay in the cache. See article Dynamic fragment cache policy configurationfor more information on priority settings.

So, a cache storing at most 10,000 entries, giving all entries a default priority of one, would be declared as:

```
<cache size="10000" priority="1" />
```

Controlling external caches

WebSphere Application Server can control external caches. You can define different groups of external caches, each with its own set of member caches. The interface between WebSphere Application Server and one of the member caches is an adapter bean, typically written by the vendor of the external cache. An administrator registers external cache groups within an <externalCacheGroups> element, which can then be used in cache policies. Each group is defined within a <group> element, which can contain <member> elements. See the External cache adapter building article for information on configuring WebSphere Application Server to control IBM Edge Server or IBM HTTP Server.

Group

`<group id="group_name" >    ... </group>`

group_name: A string identifier for this group of external caches.

This name is used to specify the external group.For a servlet to be externally cacheable, it must be configured in the `servletcache.xml` file that is included in a group.

Member

`<member address="address" adapterBeanName="adapter_class" />`

address: The hostname or IP address for this external cache.

This string will be passed to the adapter bean's setAddress method.

adapter_class: The package and class name of the adapter for this external cache.

Example of the *dynacache.xml* file

An entire dynacache.xml file based on the element definitions previously described might look like the following example:

```
<?xml version="1.0"?>

<!DOCTYPE cacheUnit SYSTEM "dynacache.dtd">

<cacheUnit>

<cache size="10000" priority="1" />

<externalCacheGroups>

<group id="edgeservers" type="shared">

<member address="edgeone" adapterBeanName="my.package.EdgeAdapter" />

<member address="edgetwo" adapterBeanName="other.package.OtherAdapter" />

</group>

</externalCacheGroups>

</cacheUnit>
```

Quick Reference of the *dynacache.xml* file

1. <?xml version="1.0"?>
2. <!DOCTYPE cacheUnit SYSTEM "dynacache.dtd">
3. <cacheUnit>
4. <cache size="*number of elements*" priority="*default priority*" />
5. <externalCacheGroups>
6. <group id="*group name*">
7. <member address="*server name*" adapterBeanName="*some.package.Adapter*"/>
8. </group>
9. </externalCacheGroups>
10. </cacheUnit>

6.6.0.16.2: Policy configuration

Cache policies can be configured using the XML configuration file, *servletcache.xml*, or the Application Assembly Tool. This article describes building the XML file, which currently supports more caching features than the AAT. Unless stated otherwise, the XML elements have identical counterparts in the AAT. See the Application Assembly Tool documentation to learn how to build cache policies with the AAT.

XML configuration

Administrators define the servlets to be cached inside the *servletcache.xml* file. The *servletcache.xml* file, like the*dynacache.xml* file, is located in the`product_installation/properties`directory.

The root element of this XML file is <servletCache>, which contains <servlet> elements.Within the <servlet> element, you specify parameters that:

1. Identify the servlets to be cached
2. Govern the creation of entry ids for servlets
3. Describe how entries generated from servlets are deleted from the cache

Typically you declare several <servlet> elements inside a *servletcache.xml* file.

Just like the *dynacache.xml* file, the *servletcache.xml* file is read only once, when WebSphere Application Server is first started. So if changes are madeto the file, WebSphere Application Server must be restarted for the changes to take effect.

The DTD for the *servletcache.xml* file is specified in the *servletcache.dtd* file, which ships with WebSphere Application Server and is located in the`product_installation\properties` directory. The *servletcache.dtd* file defines the root element <servletCache>. The *servletcache.dtd* file is included in the *servletcache.xml* file through the DOCTYPE declaration.

The beginning of the *servletcache.xml* should have the following processing instructions:

```
<?xml version="1.0"?><!DOCTYPE servletCache SYSTEM "servletcache.dtd">
```

Identifying What to Cache

The cache will parse the *servletcache.xml* file on startup, and extract from each <servlet> element a set of configuration parameters. Then, every time a new servlet is initialized, the cache attempts to match that servlet to each of the different servlet elements to find the configuration information for that servlet.

You can specify which servlets to cache in two ways:

1. By specifying the class name of the servlet, or
2. By using the servlet's full URI, beginning after the host name/IP address

To cache a servlet class, use the <servletimpl> tag.

Servletimpl

```
<servletimpl class="ClassName" />
```

ClassName: The class to be cached. This class must extend HttpServlet.

Whenever a servlet of the specified class is initialized, the dynamic caching functionwill match that servlet with the element configuration.

Note: This comparison is done by matching strings; subclasses of the specified servlet implementation will not match this declaration.

Defining servlets by class name is ***not*** supported by the Application Assembly Tool.

Alternatively, you can specify a web path for your servlets. You do this with the <path> tag.

Path

`<path uri="`*`web_path`*`" />`

web_path: The full URI of the servlet, from the hostname to the CGI variable, but not including the CGI variable. The Web application's Web path must be a part of this parameter.

So if you access a cacheable servlet with the URL *http://servername.ibm.com/webappname/servlet/MyServlet?arg1=1&....* then that servlet's path element should read `<path uri="/webappname/servlet/MyServlet" />`

You can specify different path elements referring to the same servlet. Also, the same path can appear in two different <servlet> elements, but when configuring a servlet invoked from that path, dynacache will use the configuration from the first valid <servlet> element that matches.

<servlet> <servletImpl class="**CalcServlet**" /> : (other config info) : </servlet>	<servlet> <path uri="/tools/Calc" /> <path uri="/tools/servlet/CalcServlet" /> <path uri="/tools/Calculator.jsp" /> : (other config info) : </servlet>
This configuration will match any servlet of class **CalcServlet**. Also, if class **ScientificCalculatorServlet** extends **CalcServlet**, **ScientificCalculatorServlet**s will not match this element	This configuration yields slightly different behavior. In it we are working within a hypothetical tools Web application. The first path will catch any request for the /tools/Calc URI, regardless of that servlet's class. The second path will catch any calls to the Invoker Servlet for the **CalcServlet** class. The third path allows you to cache on a JSP implementation of the calculator.
This example of a general setup will cache every servlet of class CalcServlet across the entire server, regardless of the URI	This example of a specific setup will cache the GUI defined servlet Calc (which may or may not be of the CalcServlet class), and will only cache CalcServlets if they are specifically invoked

Removing entries from the cache

After you declare which servlets to cache, defining how to cache those servlets has two parts:

- Building unique entries for different requests, and
- Removing those entries at the appropriate time

You can do this using the *servletcache.xml* file, or by writing your own class to handle invalidation.

The cache removes entries in the following circumstances:

1. One of the cache's invalidation methods (see com.ibm.websphere.servlet.cache.Cache in Javadoc) was called directly, inside the servlet code
2. Running a servlet triggered a rule based invalidation

3. The timeout for the entry expired
4. The cache was full and a new entry replaced an old one

The first case is not relevant in this context, as it is done programmatically inside an application. The second case results from the "invalidate" attribute used with a cache variable, and third and fourth cases are configured using the <timeout> and <priority> tags.

Timeout

<timeout seconds="*time_in_cache*" />

time_in_cache: The length of time, in seconds, after creation of an entry, that it should be removed from the cache. This value is required. If this value is zero or negative, the entry will not timeout, and can only be removed when the cache is full or programmatically from within an application.

If the <timeout> element is not present, then the <servlet> element in which it is contained will be invalidated, and will not be cached. When entries with a positive timeout are created, the timeout is added to the current time to determine when the entry will be invalidated. When that time is reached, if the entry remains in the cache, the cache will force its invalidation.

Priority

<priority val="*priority*" />

priority: This value is a zero or positive integer, designating the length of time an entry will stay in the cache before becoming eligible for removal. If this element is not present, then the default value, defined in the dynacache.xml file, will be used.

Priority is an extension of the Least Recently Used caching algorithm. It represents the number of cycles through the LRU algorithm an entry is guaranteed to stay in the cache. On each cycle of the algorithm, the priority of an entry is decremented. Once it reaches zero, it is eligible to be invalidated. If an entry is requested while in the cache, its priority is reset to this value, keeping it in the cache longer. Therefore, a higher priority will provide a relatively longer availability for an entry, and more frequently requested entries will stay in the cache longer, with an entry's timeout as a hard cap.

In a high volume application where space in the cache is at a premium, designers should consider increasing the priority of a servlet or JSP if calculating its output is significantly harder than average, or if it will be executed much more often than average.

Note: Priority values should be kept low, as higher values will not yield a relative improvement, but will use extra LRU cycles. Declaring a servlet with priority 3 and another with priority 4 will generate the same invalidation behavior as servlets with priority 1 and 2, only marginally slower. When dealing with caches in the thousands of entries and greater, that slowdown becomes significant.

Use the timeout variable to guarantee the validity of an entry. Use priority to rank the relative importance of that entry. Giving all elements equal priority results in a standard LRU cache that increases performance significantly, but you can tailor the cache's operation to the application with careful use of the priority variable.

Externalcache

<externalcache id="*group_name*" />

group_name: The name of an external cache group defined in the global configuration of the cache.

When this page is cached, a copy of it will now be pushed to this external cache group. See the external cachearticle for more information.

IdGenerator

<idgenerator class="*class_name"* />

class_name: The full package and class name of a class entending `com.ibm.servlet.dynacache.IdGenerator`.

See the "Custom Id and MetaData Generators" section for more information.

MetaDataGenerator

<metadatagenerator class="*class_name"* />

class_name: The full package and class name of a class entending `com.ibm.servlet.dynacache.MetaDataGenerator`.

See the "Custom Id and MetaData Generators" section for more information.

Specifying how to cache

Each time a servlet is called and WebSphere Application Server generates a corresponding `HttpServletRequest` object, the cache uses information in that object to build an id string to represent the call.

A servlet's cache policy contains the rules that determine which pieces of information are used in that id string. It always contains certain default information, such as the URI of the requested fragment and its character encoding. The rules give extra information about what variables should be used in the id, and how they should be treated.

Using cache variables

A cache variable is a generic term for a variable whose data should be used in caching a fragment. The four types of cache variables that correspond to the main sources of input for a fragment are:

- request parameters
- request attributes
- session attributes
- cookies

A client browser can set request parameters with CGI when submitting a form, while a servlet can set attributes on an HttpServletRequest object, then forward or include that request to another servlet. Finally, WebSphere Application Server can maintain session attributes that a servlet might want to access, and set cookies for later servlets to process.

In addition to building cache ids, these variables are used to controlthe following:

- grouping of cache entries
- the invalidation of other groups
- determining whether to cache a servlet based on that variable's value

The cache can decide whether or not to cache an invocation depending on the value of its input variables using the <exclude> tag.

Groups are handled by data ids, which consist of strings specified in the cache variable combined with the variable's value. Using the data_id attribute builds a group name and adds a servlet's cache entries to that group.

The invalidate attribute causes the eviction of a group from the cache. It is possible to have an entry that does no caching, and only invalidates groups. The <invalidateonly> tag is provided to save processing time for this case.The syntax for the various attibutes follow.

Request, Parameter, and Attribute

```
<request>
   <parameter id="parameter_name"
         data_id="group_identifier"
         invalidate="group_identifier"
         ignorevalue="true|false"
         required="true|false" >
     <exclude value="exclude_value"/>
   </parameter>

<attribute id="attribute_name"
          method="method_name"
           data_id="group_identifier"
          invalidate="group_identifier"
         ignorevalue="true|false"
         required="true|false" >
     <exclude value="exclude_value"/>
    </attribute>
</request>
```

parameter_name: The name of a request parameter, the value of which will be used to generate cache entry ids.

attribute_name: The name of a request attribute. The output of one or more of this object's methods will be used to generate cache entry ids.

method: The name of a method in this request attribute. The output of this method will be used to generate cache entry ids. If this value is not specified "toString" is assumed. This method may not take any arguments, and parenthesis should not be included in the method.

group_identifier: A string that, when combined with the value of this request variable, generates a group name for this cache entry. If used by a "data_id" attribute, this cache entry will be placed in that group. If used in an "invalidate" entry, then that group will be invalidated, and this entry will then be placed in the cache. If this variable is not present, the group id will not be used, and neither action will occur.

ignorevalue: Indicates whether this variable's value is relevant to the cache id, or whether only the variable's presence is important. If true, the cache id will reflect that the variable was present, but its value will not be used. This value defaults to false when not set.

required: Indicates whether this value must be present in the request. If this is set to true and either the parameter/attribute is not present or (when defining an attribute) does not contain the specified method, then this request will not be cached. This value defaults to false when not set.

You can define unlimited parameter and attribute objects within a <request> element, but you should only define one <request> element per <servlet> element.

The cache's handling of session attributes is identical to its handling of request attributes.

Session

```
<session id="session_attribute_name"
          method="method_name"
          data_id="group_identifier"
          invalidate="group_identifier"
          ignorevalue="true|false"
          required="true|false" >
     <exclude value="exclude_value"/>
 </session>
```

session_attribute_name: The name of a session attribute, the value of which will be used to generate cache entry ids.

method: The name of a method in this session attribute. The output of this method is used to generate cache entry ids. If this value is not specified "toString" is assumed. This method may not take any arguments, and parenthesis should not be included in the method.

group_identifier: A string that, when combined with the value of this request variable, generates a group name for this cache entry. If used by a "data_id" attribute, this cache entry will be placed in that group. If used in an "invalidate" entry, then that group will be invalidated, and this entry will then be placed in the cache. If this variable is not present, the group id will not be used, and neither action will occur.

ignorevalue: Indicates whether this variable's value is relevant to the cache id, or if only the variable's presence is important. If true, the cache id will reflect that the variable was present, but its value will not be used. This value defaults to false when not set.

required: Indicates whether this value must be present in the session. If this is set to true and either the session parameter is not present or does not contain the specified method, then this request is not cached. This value defaults to false when not set.

As with request parameters/attributes, there can be any number of session attributes declared in a servlet.

Invalidateonly

<invalidateonly />

When this tag is used, no caching is performed for this servlet, though invalidations triggered by it will take effect.

Exclude

The <exclude/> tag attaches to a cache variable and is used to keep a servlet from being cached when that variable is present on the request. It can only be applied for certain values, or for all values of a cache variable. When the exclude tag is used without its "value" attribute, no caching is performed for this servlet when the variable is present. Alternatively, users can repeat the <exclude> tag with the "value" attribute defined several times to specify a set of values that keep the servlet from being cached. This helps prevent caching of control servlets.

exclude_value: When the cache variable is present on a request and its value is equal to this, the servlet is not cached.

For example, the following setup indicates that whenever the request parameter "nocache" is present, the servlet should not be cached:

```
<servlet>
   <path uri="/myServlet"/>
   <timeout seconds="-1" />
   <request>
     <parameter id="nocache" >
            <exclude/>
     </parameter>
   </request>
</servlet>
```

In this example, the servlet will be cached unless the request parameter "cache" is present and its value equals "no" or "false":

```
<servlet>
    <path uri="/myServlet" />
    <timeout seconds="-1" />
    <request>
```

```
      <parameter id="cache" >
         <exclude value="no"/>
         <exclude value="false"/>
      </parameter>
    </request>
</servlet>
```

Exclude tags can be applied to any cache variable.

Notes:

Invalidating has a higher performance cost than caching. Avoid using the same variable to invalidate a group and to group an entry.You will see less performance benefit than when just using the variable to define a group id.

Data_id and invalidate tags that are within the same <servlet> element should have different values, usually. If they have the same value, the group is invalidated, and the entry for the current servlet is put into that group. The invalidate tag that corresponds to a data_id will occur in different <servlet> elements.

Quick Reference of the *servletcache.xml* file

1. <?xml version="1.0"?>
2. <!DOCTYPE cacheUnit SYSTEM "dynacache.dtd">
3. <servletCache>
4. <servlet>
5. <invalidateonly/>

   <servletimpl class="*some.package.SomeClass*"/> OR <path uri="*MyServlet*" />
   <path uri="*servlet/SomeClass*"/>
6. <timeout seconds= "*timeout value*" /> OR <metadatagenerator
   <priority value="*priority value*" /> class="package.GeneratorClass" />
7. <externalcache id="external cache group name">
8. OR
9. <request>
10. <parameter id="*parameter name*"

    data_id="*group identifier 1*"
    invalidate="*group identifier 2*"
    ignorevalue="true|false"
    required="true|false" >

      <exclude value="*exclude value*"/>
    </parameter>
11. <attribute id ="*attribute name*"

    method= "*aMethodName*"
    data_id="*group identifier 1*"
    invalidate="*group identifier 2*"
    ignorevalue="true|false"
    required="true|false" />

12. </request>
13. <session id="*session attribute name*"
    method="*aMethodName*"
    data_id="*group identifier 1*"
    invalidate="*group identifier 2*"
    ignorevalue="true|false"
    required="true|false" />
14. cookie id="*cookie name*"
    method="*aMethodName*"
    data_id="*group identifier 1*"
    invalidate="*group identifier 2*"
    ignorevalue="true|false"
    required="true|false" />
15. <idgenerator class="package.IdGeneratorClass" />
16. <metadatagenerator class="package.MetaDataGeneratorClass"/>
17. </servlet>
18. </servletCache>

Notes:
Line 6: Timeout < 1 implies the value will not time out. Required.
Line 10: The method, called from the request parameter, defaults to `toString`, and `required` defaults to false.
Line 10: The exclude tag can be used in any of the four cache variable types.
Line 10: The method called on the request attribute defaults to toString, and `required` defaults to false.
Line 13: The method called on the session attribute defaults to toString, and `required` defaults to false.
Line 14: The method called on the cookie defaults to toString, and `required` defaults to false.
Line 8: OR applies to lines 10-15

6.6.0.16.3: Dynamic fragment cache XML examples

In this example, Calculator Servlet is defined inside a 'tools' Web application.It takes an operation 'operation' and two arguments, 'arg1' and 'arg2'.The values for these variables are received in the query string from an external user,that is, request parameters from a browser or applet.You invoke the servlet to calculate 2+3 to get the answer 5, with the URI:

`/tools/Calc?arg1="2"&arg2="3"&operation="+"`

To cache the output of this servlet, you distinguish results using the request parameters, so that 2,3,+ has a different cache entry than 4,5,*. For this example, you would define the following <servlet> element:

```
<servlet>
   <path uri="/tools/Calc" />
   <request>
   <parameter id="arg1"      required="true" />
   <parameter id="arg2"      required="true" />
   <parameter id="operation" required="true" />
   </request>
   <timeout seconds="-1" />
</servlet>
```

In a second example, the news servlet of class CoastalNewsServlet displays either west coast news or east coast news, depending on a user's location.This servlet has a session object named 'location' of class 'LocationBean' with a method getCoast() that returns "east" or "west". If this object is not present on the session, then the servlet returns the news for both coasts, which you also want to cache. So whether or not the location is present on the session, you want to cache the output of the servlet, andyou do not want the entries to time out:

```
<servlet>
         <servletimpl class="CoastalNewsServlet" />
         <session id="location" method="getCoast" />
         <timeout seconds="-1" />
</servlet>
```

To group cache entries based on the coast, or to invalidate the cache entries for a coast whenever the location bean gets updated by a control servlet, you must consider the design of the application.

If a variable is not available to a servlet at execution (that is, the request/session variable has not been set), then, even if the servlet is cacheable, no group id will be generated based on that variable.In the CoastalNewsServlet example, a missing session parameter causes the servlet to display the news for both coasts. Naturally, in this case, you want the cache entry to belong to both the east and west coast groups. However, because the cache does not handle data ids when variables are missing, this is not possible.

The simplest solution to this problem marks the location bean as a required variable for caching. This means the output of the news servlet will not be cached if location is undefined. Therfore, you can now do all the grouping knowing that the location bean will be present to place entries into the correct groups. Servletcache.xml needs two sets of changes to finish configuring groups:

- the CoastalNewsServlet entry must be modified to put entries into groups, and
- a new entry for the control servlet that updates the location bean must be added to allow invalidation of these groups.

```
<servlet>
      <servletimpl class="CoastalNewsServlet" />
      <session id="location" method="getCoast" data_id="coast" required="true" />
      <timeout seconds="-1" />
</servlet>


<servlet>
       <invalidateonly/>
       <servletimpl class="LocationUpdateServlet" />
       <request>
       <attribute id="new_coast" invalidate="coast" />
  </request>
</servlet>
```

Now, when the news servlet is invoked, the cache will take the data_id "coast" and append "=" and the value of location.getCoast() to create a group name to identify that entry. In the update servlet, a new_coast string is expected as a request attribute, and will be used in the same fashion to build group names for removal from the cache.

Using HttpSession and request attributes

The dynamic cache can use objects stored in an HttpSession or request attribute in caching requests. The Servlet specification allows the Servlet or JSP programmer to put any number of objects into the session or request, and index these values with a String key name. Possible uses range from the storage of minimal data and simple types (for example, String, Integer, Boolean) to very complex and large (Vectors and Hashtables of complex User Objects).

Given this background, this is what the WebSphere Application Server Servlet/JSP cache supports:

- When building cache policies the user can specify:
  1. A key name for a session/request attribute value and

2. An [optional] method name used for retrieving a value from the stored object.

- In the case where the method name is not specified, the default method toString() is used to transform the object into a String for use in the cache id.

  Consider the following object:

  ```
  class User {    public String getName() {...};        public String getPrimaryGroup() {...};    }
  ```

  If an instance of this object is stored in the session using the key "user", you might specify the following cache policy:(in this example, imagine that different user groups view a different home page):

  ```
  <servlet>
          <path uri="/index.jsp" />
          <session id="user" method="getPrimaryGroup" />
  <servlet>
  ```

Caching personalized pages

Let's say we want a fragment of a page to be a stock list owned by an user. The fragment has two servlets: the first obtains the stock list from the database and forwards the list of stock symbols to the second servlet, which gets quotes for the stocks from the back end. How can we cache this fragment?

Depending on exactly how this stock list is generated, caching will have varying effectiveness, but it will provide a benefit. The answer comes from applying a key concept in servlet caching. The biggest performance wins come from caching servlets that obtain information from outside WebSphere Application Server. This means that while caching a simple presentation JSP file will give moderate performance gains, caching servlets that request information from Enterprise Java Beans or databases, saves WebSphere Application Server processing power and decreases load on the back end.

In this example, both sets of actions can be cached (since they are both reads from the database). The first servlet is a classic example of a cacheable servlet. It is a simple database lookup, using one value as input (a user id), and producing the list as output. The interesting case (an example of designing an application with caching in mind) is how the servlets go from a list of symbols to actually looking up the quotes.

The simple design involves one servlet that does all the work and a presentation JSP. The servlet gets a list of stock symbols for a user from the database (presumably that user's database record holds the symbols in their portfolio), then immediately goes back to the database and looks up current quotes for those symbols. It builds a list of quotes and forwards them to a presentation JSP.

This design can be cached. You base everything off the user id, and cache the current quotes for that user. There are missed opportunities here, though. First, since everything is based off of an individual user, you cannot reuse the stock quotes gathered for other users who own the same stocks. The usefulness of this cache entry is limited to one user. Second, whenever one of the quotes, or the list of symbols changes, the whole page changes. This entry is unstable, and will not be correctvery long.

A better design fragments the data requests into 3 different servlets/JSP files, separating each of the two database reads into its own servlet/JSP file. The first servlet would use the user id to get the list of symbols. It would forward the list to a presentation JSP that would format the list, and get individual quotes from a third servlet that takes a symbol as input.

All three of these servlets/JSP files can be cached individually. The first servlet is caching the user specific list of symbols only. Changes to a user's portfolio will not affect the cache entries for the stock quotes. The presentation JSP file only changes the list of symbols supplied to it. Caching in this example is as useful as caching the first servlet, and could be even more useful if different users have the same list of stocks. The quotes are cached with the last servlet, which will have a different entry for each stock. If one stock quote changes, then the request for that individual stock will have to be reinvoked from the application server. The rest of the requests can all be served from the cache. Most importantly, these quotes are reusable in any request for a customer's portfolio. By narrowing the responsibility of individual pieces of the application, you can provide selective caching, and achieve bigger performance gains.

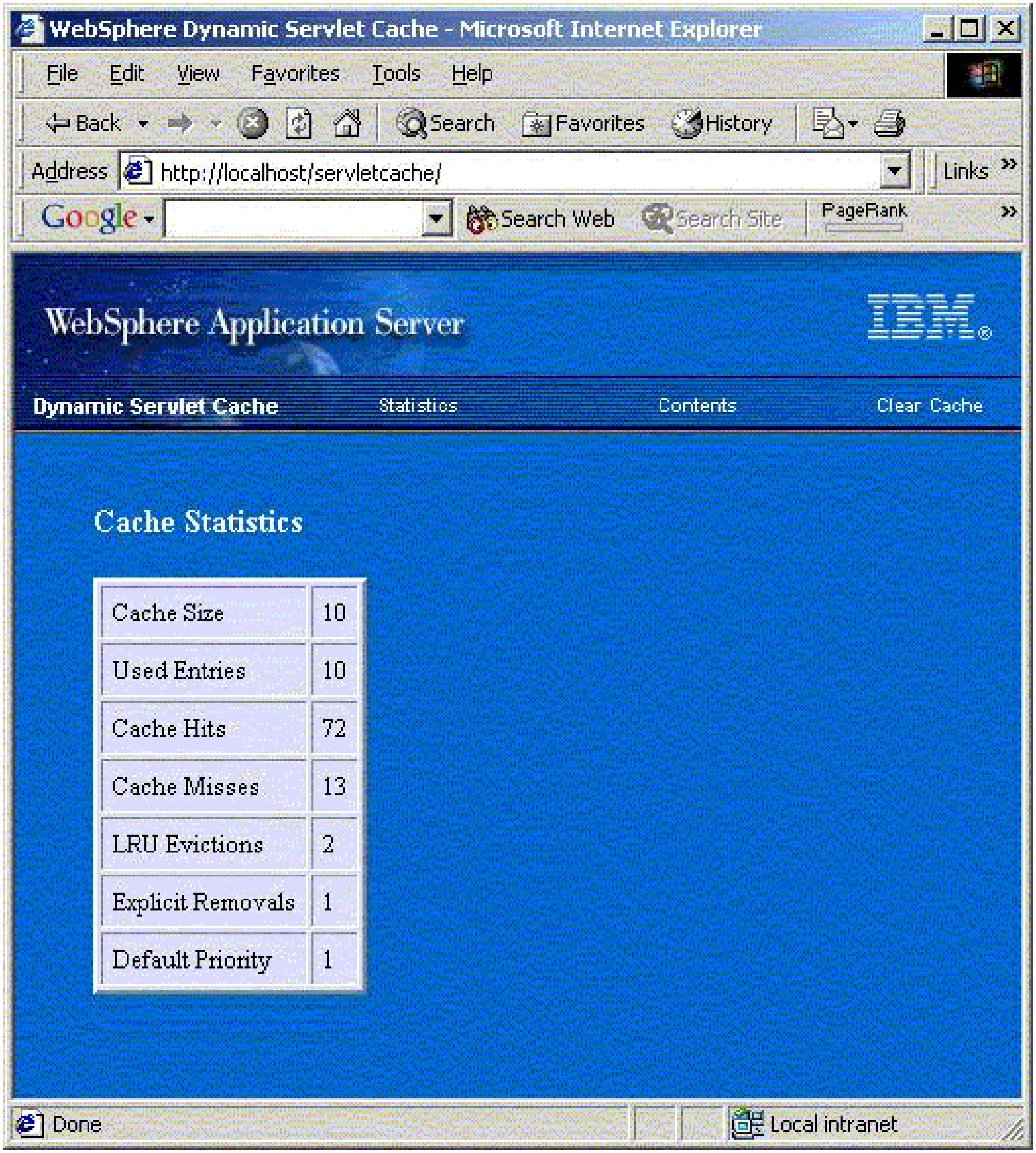
6.6.0.16.4: Dynamic fragment caching monitor

WebSphere Application Server provides the Servlet Cache Monitor application for inspecting the contents and behavior of the fragment cache.To use the Servlet Cache Monitor, install the *ServletCacheMonitor.ear* file, located in the`product_installation/installableApps` directory,on each application server that uses the dynamic fragment cache function.

After the *ServletCacheMonitor.ear* file is installed,access the Servlet Cache Monitor through the *`/servletcache`* URI.

To monitor different application servers on the same node, modify the *ServletCacheMonitor.ear* file's Web path using the Application Assembly Tool. This allows you to accessthe different servers' cache monitors with different URIs.

Cache statistics



Cache Statistics

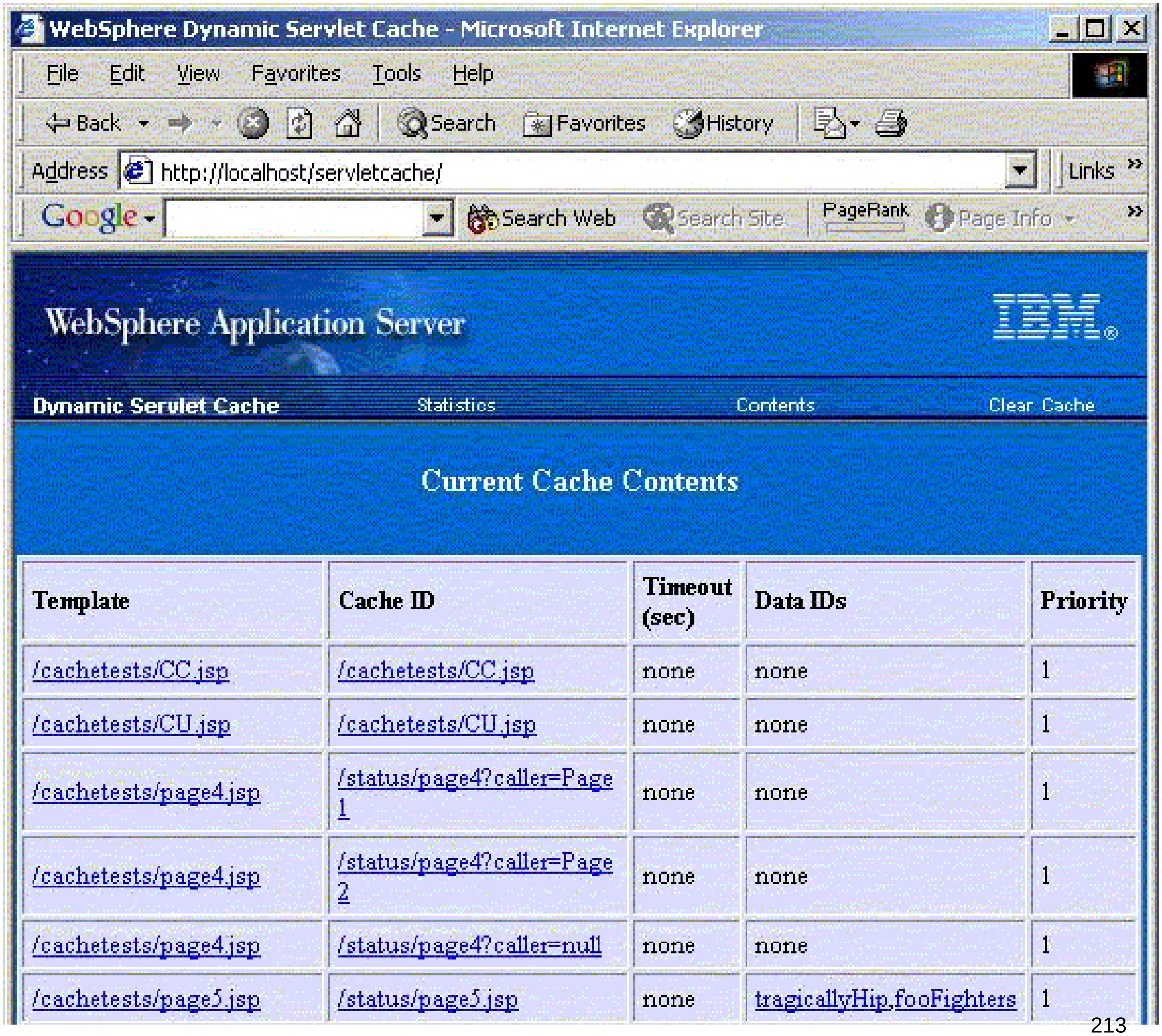
Cache Size	10
Used Entries	10
Cache Hits	72
Cache Misses	13
LRU Evictions	2
Explicit Removals	1
Default Priority	1

The statistics page is the main page of the monitor and describes the following properties:

- Cache Size - The maximum number of cache entries as defined in the global properties of the fragment cache.
- Used Entries - The number of entries currently contained in the cache
- Cache Hits - The number of servlet/JSP requests, both external (from a browser) and internal (included/forwarded from another servlet/JSP) that have been served from the cache since startup.
- Cache Misses - The number of requests for cacheable servlets or JSPs, that were not served from the cache, i.e. resulted in executing the fragment and storing the result in the cache.
- LRU Evictions - The number of entries that have been removed from the cache by the LRU algorithm when the cache was full and new entries needed to be added.
- Explicit Removals - The number of entries removed from the cache through cache policy rules, or through the cache monitor.
- Default Priority - The default value for a cache entry's priority as defined in the global properties of the fragment cache. See the Cache Policies section of the fragment cache documentation for more information about priority.

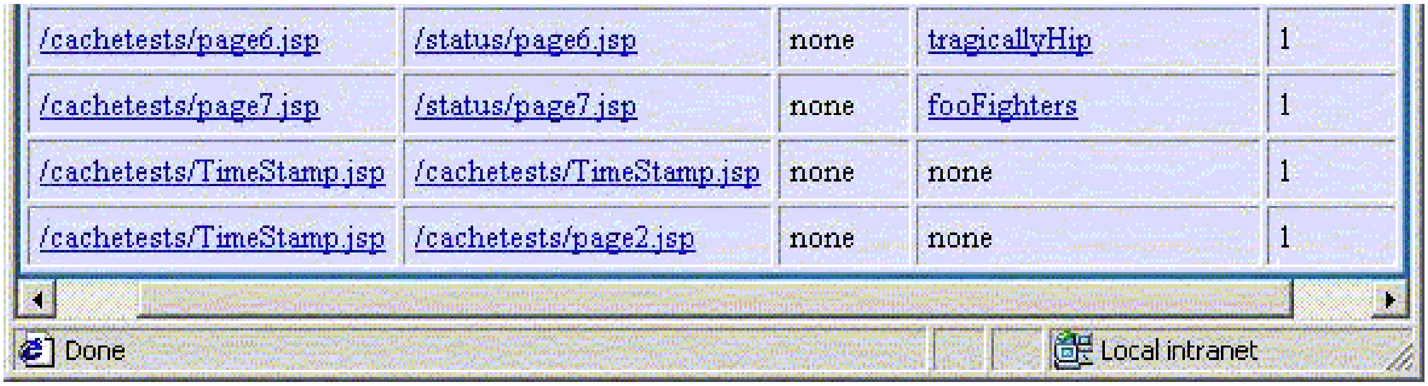
Cache contents

The contents of the fragment cache can be viewed in the "Cache contents view" panel. A list of all entries, an individual entry, or lists of entries with identical group ids (also referred to as data ids) are available.



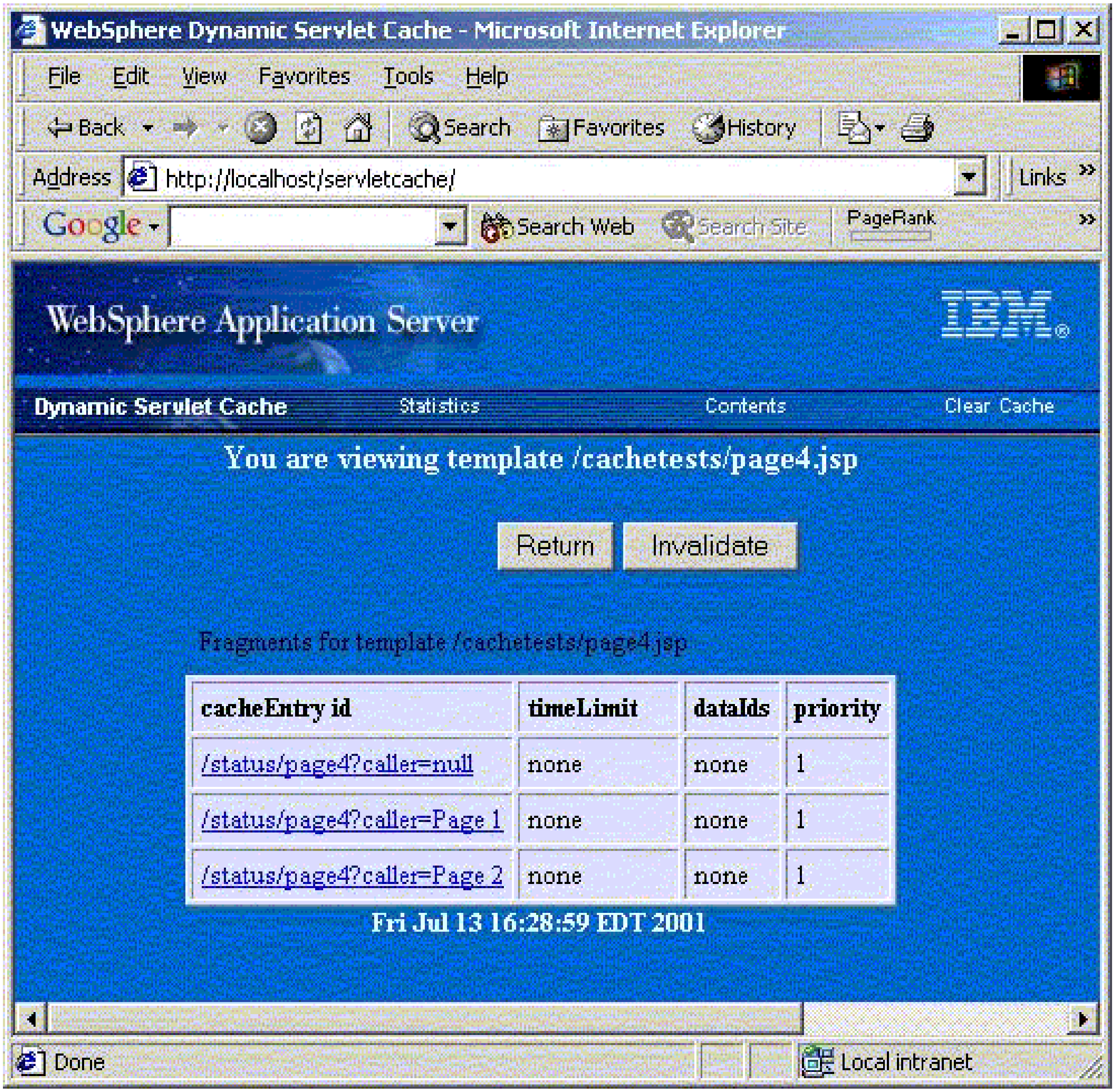
Current Cache Contents

Template	Cache ID	Timeout (sec)	Data IDs	Priority
/cachetests/CC.jsp	/cachetests/CC.jsp	none	none	1
/cachetests/CU.jsp	/cachetests/CU.jsp	none	none	1
/cachetests/page4.jsp	/status/page4?caller=Page 1	none	none	1
/cachetests/page4.jsp	/status/page4?caller=Page 2	none	none	1
/cachetests/page4.jsp	/status/page4?caller=null	none	none	1
/cachetests/page5.jsp	/status/page5.jsp	none	tragicallyHip,fooFighters	1



/cachetests/page6.jsp	/status/page6.jsp	none	tragicallyHip	1
/cachetests/page7.jsp	/status/page7.jsp	none	fooFighters	1
/cachetests/TimeStamp.jsp	/cachetests/TimeStamp.jsp	none	none	1
/cachetests/TimeStamp.jsp	/cachetests/page2.jsp	none	none	1

For each entry you can see several of its properties, including template, cache id, and data id. Clicking a template or data id displays a list of entries with the same template/data id.



WebSphere Dynamic Servlet Cache - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites History
Address http://localhost/servletcache/
Links
Google Search Web Search Site PageRank
WebSphere Application Server
IBM
Dynamic Servlet Cache
Statistics
Contents
Clear Cache
Return
Invalidate
Done
Local intranet

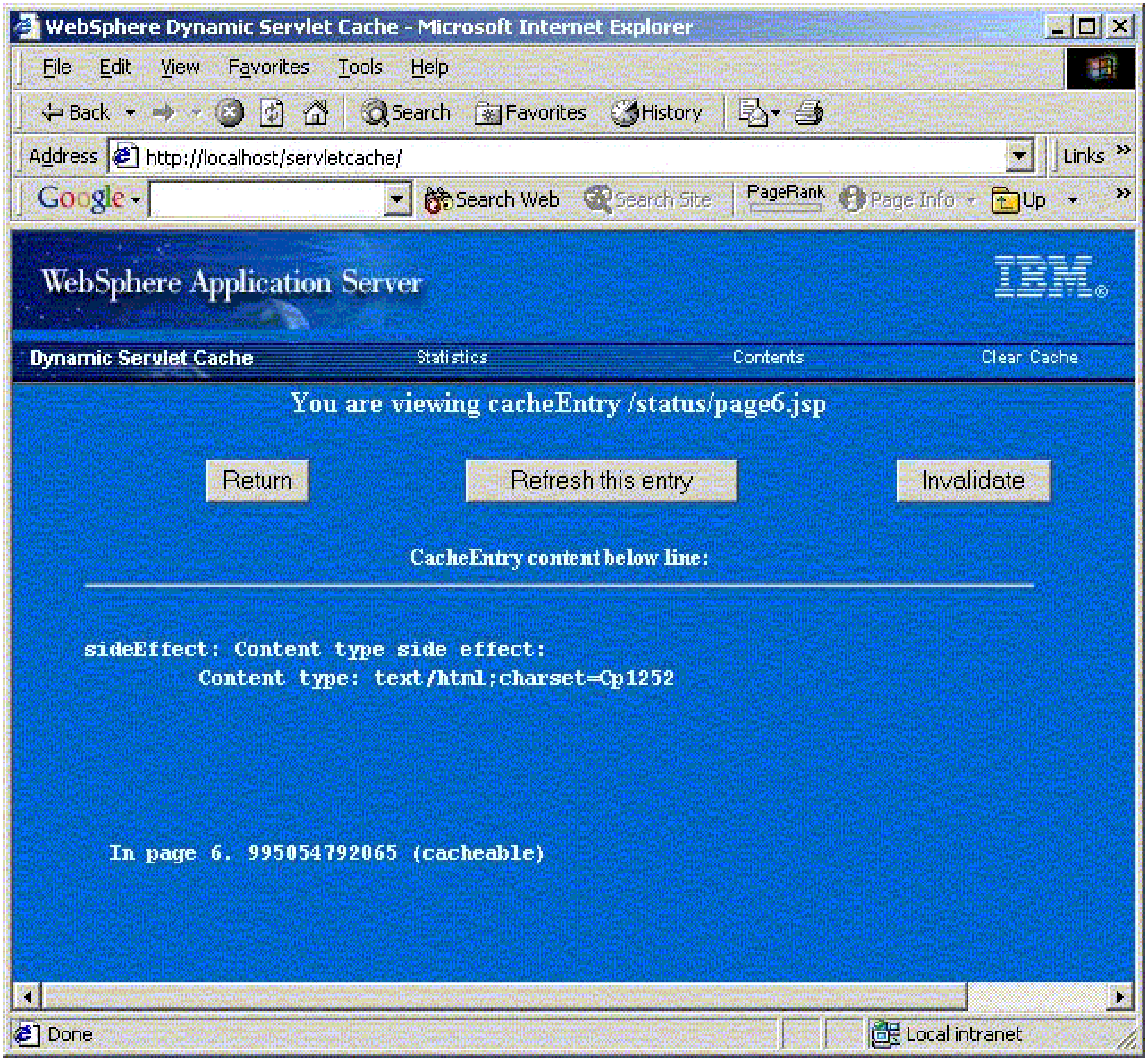
You are viewing template /cachetests/page4.jsp

Fragments for template /cachetests/page4.jsp

cacheEntry id	timeLimit	dataIds	priority
/status/page4?caller=null	none	none	1
/status/page4?caller=Page 1	none	none	1
/status/page4?caller=Page 2	none	none	1

Fri Jul 13 16:28:59 EDT 2001

Clicking a cache id in any of the menus in the "Cache list of entries" view, displays a detailed description of the entry. From this panel, you can invalidate the entry, reset its position in the LRUalgorithm (mimicking a cache hit on that entry), or return to the main list of entries.


WebSphere Dynamic Servlet Cache - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites History
Address http://localhost/servletcache/
Links
Google Search Web Search Site PageRank Page Info Up
WebSphere Application Server
IBM
Dynamic Servlet Cache
Statistics
Contents
Clear Cache
Return
Refresh this entry
Invalidate
Done
Local intranet

You are viewing cacheEntry /status/page6.jsp

CacheEntry content below line:

```
sideEffect: Content type side effect:
        Content type: text/html;charset=Cp1252
```

```
In page 6. 995054792065 (cacheable)
```

Clear the cache

This removes every entry currently in the cache.

6.6.1: Administering applications (overview)

Administration of applications consists of the following:

- Assembling the modules of the application, setting deployment descriptor properties, and generating code for deployment using the Application Assembly Tool
- Performing additional configuration tasks using the console

Classpath considerations

An important classpath-related setting to note is the Module Visibility. This application server setting impacts the portability of applications and standalone modules from other WebSphere Application Server versions and editions. If your existing module does not run as-is when you transfer it to Version 4.0, you might need to reassemble an existing module or change the module visibilitysetting.

See the information on setting classpaths for a full discussion of classpath considerations. See the applicationserver property reference for information about the module visibility setting.

6.6.1.0: Enterprise application properties

Key:

Applies to Java administrative console of Advanced Edition Version 4.0

Applies to Web administrative console of Advanced Single Server Edition Version 4.0

Applies to Application Client Resource Configuration Tool

Archive URL

The path to the EAR file for the application

Context Root

The context root of the Web application contained in this enterprise application.

The context root is combined with the defined servlet mapping (from the WAR file) to compose the full URL that users type to access the servlet. For example, if the context root is /gettingstarted and the servlet mapping is MySession, then the URL is http://host:port/gettingstarted/MySession.

EnterpriseApp Name **or Name**

A logical name for the application. The name will be displayed in administration and configuration tools.

Execution State

For Advanced Edition: The current state of the application, such as running or stopped

For Advanced Single Server Edition: The execution state that you would like the application to have, the next time you start the product

Node

The administrative node with which the application is associated.

Path, or Remote Path on Server

The fully qualified path to the .ear, .jar, or .war file for which you are configuring an enterpriseapplication

Use the first set of fields if the console and application files are on the same machine (whether or not the server is on that machine, too).

Use the second set of fields (including **Remote Path...**) if the application files already reside on the server machine, and the console is being run from a remote machine.

During application installation, application files are typically uploaded from a client machine containing the administrative console to the server machine, where they are deployed. In such cases, the Web browser running the administrativeconsole is used to select EAR, WAR, or JAR modules to upload to the servermachine.

In some cases, however, the application files will already reside on the file system of the machine running the application server. To have the application server install these files, (bypassing the upload stage), use the Remote Path option.

An example of the field value on Windows NT is C:\WebSphere\AppServer\installableApps\test.ear. You can optionally enter the application name to use for standalone modules (JAR, WAR) and the context root (used only for Web modules).

User/Role Mappings, Run As Mappings, Roles, Users, and Groups

See the properties for mapping security roles and "run as" roles to users and groups.

6.6.1.0.1.a: Assembly properties for security roles (application)

A security role is a logical grouping of principals. Access tooperations (such as EJB methods) is controlled by granting access to arole.

Role Name (Required, String)

Specifies the name of a security role that is unique to theapplication.

Description

Contains a description of the security role.

Binding -- Groups -- Name

Specifies user groups that are granted the security role.

Binding -- Users -- Name

Specifies users that are granted the security role.

Binding -- Special Subjects -- Name

Specifies one of two special categories of users to which roles can begranted: Everyone or All authenticated users. If the specialsubject Everyone is granted a role, then all users, including those who didnot authenticate, are granted the role. In other words, a method on anenterprise bean or a URI is unprotected if any of the required roles for thatmethod are granted to the special subject Everyone. In the case of Allauthenticated users, any user who can authenticate by using a valid user IDand password is considered to be granted that role.

6.6.1.0.aa: Assembly properties for enterprise applications

File name

Specifies the file name of the application.

Alternative DD

Specifies the file name for an alternative deployment descriptor file touse instead of the original deployment descriptor file in theapplication's JAR file. This file is the postassembly version ofthe deployment descriptor file. (The original deployment descriptorfile can be edited to resolve dependencies and security information.Directing the use of the alternative deployment descriptor allows you to keepthe original deployment descriptor file intact). The value of theAlternative DD property must be the full path name of the deploymentdescriptor file relative to the application's root directory. Byconvention, the file is in the ALT-INF directory. If this property isnot specified, the deployment descriptor file is read directly from themodule's JAR file.

Display name (Required, String)

Specifies a short name intended to be displayed by GUIs. It is usedto identify the application at deployment time.

Small icon

Specifies a JPEG or GIF file containing a small image (16x16pixels). The image is used as an icon to represent the application in aGUI.

Large icon

Specifies a JPEG or GIF file containing a large image (32x32pixels). The image is used as an icon to represent the application in aGUI.

Description

Contains a description of the application.

Reload interval

Specifies the time interval, in seconds, at which the EJB modules are tobe reloaded.

Enterprise application name

Specifies the global JNDI name of the application.

Run-as bindings - Authorization Data - User ID

Specifies the security principal to be used for the execution of theapplication's enterprise bean methods. This security principal isgranted the named role.

Run-as bindings - Authorization Data - Password

Specifies the password of the security principal.

Run-as bindings - Role name

Specifies a security role name. This name appears as the value ofthe Use Identity Assigned to Specified Role property of the enterprisebean.

6.6.1.1: Administering enterprise applications with the Java administrative console

Use the Java administrative console to administer enterprise applications.

Work with resources of this type by locating them in the tree view:

WebSphere Administrative Domain -> Enterprise Applications -> ***application_name***

Expand the tree further to see the Installed EJB Modules and Installed Web Modulesof an application.

6.6.1.1.1: Installing enterprise applications and modules with the Java administrative console

To install an enterprise application into an application server runtime, use the ApplicationInstallation Task Wizard. Access it by clicking **Console -> Wizards -> Install Enterprise Application** on the console menu bar.

Using the Application Installation wizard

You use the Application Installation wizard to install an application (EAR file)or module (JAR or WAR file). It leads you through some or all of the followingtasks, depending on whether you are working with an application, EJB module, orWeb module:

- Specifying the application or module
- Mapping users to roles
- Mapping RunAs roles to users
- Binding EJBs to JNDI names
- Mapping EJB references to EJBs
- Mapping resource references to resources
- Specifying the default datasource for EJBModule
- Specifying datasources for CMP beans
- Specifying virtual hosts for WAR modules
- Selecting the target application server
- Completing the application or module installation

Please take the term "EJBs" in the console to refer to "enterprise beans."

⚠ Users and groups defined in the application EAR file will be removedduring application installation if they do not exist in the specified system registry.

⚠ Note that if the EAR file or EJB module that youare installing contains method permissions, you will be prompted whether to deny all accessto unprotected methods. If you select to deny access, the method permission for *all*methods in *all* enterprise beans in the application will be set to DenyAllRole.

Specifying the application or module

On the Specifying the Application or Module panel:

1. Select to work with either an application or a standalone module.
2. Specify the application or module path, the application name and, for modules, the context root for the web module.

   View properties reference for:

   - Enterprise applications
   - EJB modules
   - Web modules

3. Click **Next**.

Mapping users to roles

For assistance with this task, see the associated instructions.

When finished, click **Next**.

Mapping RunAs roles to users

On the Mapping EJB RunAs Roles to Users panel:

1. Select a RunAs role from the table and click **Select**.
2. Enter the name of a user. The user must exist in the user registry and be a member of the role to which the RunAs role is being assigned.
3. Enter the password for the user in the **Password** field.
4. Repeat steps 1 and 4 for each role that needs to be mapped.
5. Click **Next**.

Binding EJBs to JNDI names

On the Bind EJB to JNDI Name panel:

1. Select an EJB from the table and click **Edit JNDI Name**.
2. Select a JNDI name for the EJB.

   View properties reference for:

   - EJB Modules
3. Repeat steps 1 and 2 for each EJB that needs binding to a JNDI name.
4. Click **Next**.

Mapping EJB references to EJBs

On the Mapping EJB references to EJB panel:

1. Select an EJB reference from the table and click **Select EJB**.
2. Select an EJB and provide its JNDI name.

   View properties reference for:

   - EJB modules
   - Web modules
3. Repeat steps 1 and 2 for each EJB reference that needs to be mapped.
4. Click **Next**.

Mapping resource references to resources

On the Mapping Resource References to Resources panel:

1. Select a resource reference from the table and click **Select Resource**.

2. Select a resource.

   View properties reference for:

   - EJB modules

3. Repeat steps 1 and 2 for each resource reference that needs to be mapped.
4. Click **Next**.

Specifying the default datasource for EJBModule

On the Specifying the Default Datasource for EJBModule panel:

1. Select one or more EJB modules from the table and click **Select DataSource**. To select multiple modules, press the Ctrl key and click the modules.
2. Select a datasource.

   View properties reference for:

   - EJB modules

3. Repeat steps 1 and 2 for EJB modules that need a datasource.
4. Click **Next**.

Specifying datasources for CMP beans

On the Specifying Datasources for Individual CMP Beans panel:

1. If your Container Managed Persistence (CMP) beans need datasources, select one or more CMP beans from the table and click **Select DataSource**. To select multiple CMP beans, press the Ctrl key and click the beans.
2. Select a datasource.

   View properties reference for:

   - EJB modules

3. Repeat steps 1 and 2 for CMP beans that need a datasource.
4. Click **Next**.

Specifying virtual hosts for WAR modules

On the Specifying the Virtual Hosts for WAR Modules panel:

1. Select one or more web modules from the table and click **Select Virtual Host**. To select multiple modules, press the Ctrl key and click the modules.
2. Select a virtual host.

   View properties reference for:

   - Virtual hosts

3. Repeat steps 1 and 2 for web modules that need a virtual host.
4. Click **Next**.

Selecting the target application server

On the Selecting the Application Server panel:

1. Select one or more modules from the table and click **Select Server**. To select multiple modules, press the Ctrl key and click the modules.
2. Select an application server or server group.

   View properties reference for:

   - Application servers
3. Repeat steps 1 and 2 for modules that need an application server or server group.
4. Click **Next**.

Completing the application or module installation

The Completing the Application Installation Wizard panel lists the node onto which the application will be installed and indicates whether you must copyone or more files to a specific directory.

To have the wizard create a new application with the values you specified, click **Finish**. If the wizard encounters an error, a message will display explaining why an application could not be created.

To change any values specified, click **Back** to return to the appropriatepanel(s), make any needed changes, and then click **Finish** on this panel.

6.6.1.1.2: Viewing deployment descriptors of enterprise applications with the Java administrative console

To view the deployment descriptor of an enterprise application:

1. Select the Enterprise Applications folder in the tree view of the administrative console. The details view of the console will display a list of zero or more installed applications (as .ear files).
2. Right-click your application in the details view and select **View Deployment Descriptor** from the popup menu.

An Enterprise Application window opens, providing descriptive information about the selected application's Application.xml and modules.

6.6.1.1.3: Showing the status of enterprise applications, with the Java administrative console

During this task, you will view the status (such as running or stopped) of each EJB or Webmodule in an enterprise application.

1. Locate the enterprise application.
2. Right-click it and select **Show Status** to display the Module Status window.
3. When finished viewing status, close the window.

6.6.1.1.4: Exporting enterprise applications with the Java administrative console

You can export your application to a location of your choice:

1. Select the Enterprise Applications folder in the tree view of the administrative console. The details view of the console will display the list of zero or more installed applications.
2. Right-click your application in the details view and select **Export Application** from the popup menu.
3. In the Export Application dialog that opens:
    a. From the **Node** drop-down list, select a node to which the application should be exported.
    b. In the **Export directory** field, specify the target drive and directory for the application.
    c. Click **OK**.

The application will be exported to the specified node and directory, and will have the name *enterpriseApplicationName*.ear.

6.6.1.5: Administering applications with Application Assembly Tool

A J2EE$^{(TM)}$ application consists of one or more Web modules, EJB modules, orapplication client modules. The Application Assembly Tool is used tocreate and edit an application's modules, verify the archive files, andgenerate deployment code. See the related topics for links to concepts,instructions for creating an application, and field help.

6.6.1.5.1: Creating an application

J2EE applications can be created by using property dialog boxes or by usinga wizard.

- Using the property dialog boxes
- Using the Create Application wizard

Using the property dialog boxes

Creating a new enterprise application consists of specifying the archivefiles that are to be included in the application and then adding assemblyinstructions. Follow these steps to create a new application:

1. Click **File**->**New**-> **Application**. Thenavigation pane displays a hierarchical structure used to build the contentsof the application. The icons represent EJB modules, Web modules,application client modules, assembly properties, and files comprising theapplication. A property dialog box containing general information aboutthe application is displayed in the property pane.
2. By default, the EAR file name and the application display name are thesame. It is recommended that you change the display name in theproperty pane.
3. By default, the temporary location of the EAR file isinstallation_directory`/bin`. You must specify a newfile name and location by clicking **File**->**Save**.You must add at least one module (archive) before saving theapplication.
4. Enter values for other properties as needed. View the help for 6.6.1.0.a: Assembly properties for enterprise applications.
5. Add archive files to the application. You can add any combinationof JAR or WAR files. There are several ways to include archivefiles:
    - Import an existing JAR or WAR file.
    - Open an existing EJB or Web module and copy and paste individual beans orWeb components into the application.
    - Create a new JAR or WAR file.
6. To import archive files into the application, right-click the iconcorresponding to the type of module (EJB module, Web module, or applicationclient module) and choose **Import**. Use the file browser tolocate and select the archive file. Click **Open**. TheJAR or WAR file is displayed under the appropriate folder. The importedmodule's properties are also imported. Click the plus sign(+) next to the archive's icon to view its contents and editits properties if needed.
7. To create a new JAR or WAR file to be included in the application,right-click the folder corresponding to the type of module to create (EJBModules, Web Modules, and Application Clients), and choose**New**. Enter required values as indicated. Optionally,enter values for other properties. Click **OK.** The JARor WAR file is displayed under the appropriate folder. Click the plussign (+) to verify contents and enter assembly properties.

    For EJB modules, you must add at least one enterprise bean.Right-click the folder corresponding to the type of bean to create (sessionbean or entity bean), and choose **New** or **Import**.For entity beans, choose whether you are creating a CMP or BMP bean. Aproperty dialog box is displayed. Enter required values asindicated. Optionally, enter values for other properties. Click**OK.** The enterprise bean is displayed in the navigationpane. Click the plus sign (+) to verify the contents andenter assembly properties.
8. To copy portions of an existing module, open the module by clicking**File**->**Open** and selecting the file. Arrange thewindows so that both the EAR file and the source archive are visible.Use a copy-and-paste operation to copy archives or components to the currentmodule.
9. Repeat the procedure for additional JAR files or WAR files representingEJB modules, Web modules, or

application clients.

10. Define security properties for the application. Right-click theSecurity Roles icon and click **New**. Enter values on theproperty dialog box and then click **OK**. View the help for 6.6.1.0.1.a: Assembly properties for security roles (application).
11. Add supplementary files needed by the application.Right-click the Files icon in the navigation pane and choose **AddFiles**. Click **Browse** to navigate to the desireddirectory or archive. Click **Select**. If you are addingan entire archive, select the directory that contains the archive. Thedirectory structure is displayed in the left pane. Browse the directorystructure. From the right pane, select one or more files to be addedand click **Add**. If you select a directory and click**Add**, all files in the directory, including the directory, areadded. The selected files are displayed in the Selected Fileswindow. Relative path names are maintained. When the SelectedFiles window contains the correct set of files, click **OK**.
12. Review the contents of the application and make any desiredchanges.
13. Save the application EAR file. Click**File->Save**. Enter a name for the application, forexample, temp.ear. Click **Save**. A dialog boxconfirming the save operation is displayed. Click **OK**.

Using the Create Application wizard

Use this wizard to create an enterprise application. An enterpriseapplication can consist of one or more EJB modules, Web modules, orapplication clients, and a deployment descriptor. During creation ofthe application, you specify the archive files for each module to be includedin the application. You also specify other information about theapplication, such as security roles.

You can create new EJB modules, Web modules, and application clientmodules, or import existing modules. When the wizard is completed, yourapplication is packaged in an EAR file and resides in the directory locationthat you specified.

To create an enterprise application, click the **Wizards** icon onthe tool bar and then click **Application**. Follow theinstructions on each panel.

- Specifying application properties
- Adding supplementary files
- Choosing application icons
- Adding EJB modules
- Adding Web modules
- Adding application client modules
- Adding security roles
- Setting additional properties and saving the archive

Specifying application properties

On the **Specifying Application Properties** panel:

1. Specify a display name for the application (required). The displayname is used to identify your application in the Application Assembly Tool andcan be used by other tools.
2. Specify a file name for the application (required). The file namespecifies a location on your system where the EAR file is to becreated.
3. Provide a short description of the application (optional).

4. Click **Next**.

Adding supplementary files

On the **Adding Supplementary Files** panel, specify the location ofadditional files to be included in your application. These are filesother than the J2EE modules to be included--for example, libraries andutilities needed by the application. To add or remove files:

1. Click **Add**. Use the file browser to locate and choose oneor more files to add to the application. First, browse for the rootdirectory or archive where the files are located and click**Select**. If you are adding an entire archive, select thedirectory that contains the archive.
2. The directory structure is displayed in the left pane. Browse thedirectory structure. From the right pane, select one or more files tobe added and click **Add**. If you select a directory and click**Add**, all files in the directory, including the directory, areadded. Relative path names are maintained. The selected filesare displayed in the Selected Files window. Click **OK**.The files are displayed in a table on the wizard panel.
3. If you want to remove a file, select the file in the table and then click**Remove**.
4. Continue to add or remove files until you have the correct set offiles. Click **OK**.
5. Click **Next**.

Choosing application icons

On the **Choosing Application Icons** panel:

1. Specify the full path name of a small icon, or click **Browse** tolocate and select the file. The icon must be a GIF or JPEG image 16x16pixels in size.
2. Specify a full path name of a large icon, or click **Browse** tolocate and select the file. The icon must be a GIF or JPEG image 32x32pixels in size.
3. Click **Next**.

Adding EJB modules

On the **Adding EJB Modules** panel, add one or more EJB JARfiles.

1. To add JAR files, click **Add**. Use the file browser tolocate and select the files, and then click **Open**. Asprompted, verify or enter a file name and alternate deployment descriptor forthe JAR file. Click **OK**. The JAR files are displayedin a table on the wizard panel.
2. To remove a JAR file, select the file and then click**Remove**.
3. Continue adding or removing JAR files as needed. When you arefinished adding JAR files, click **Next**.

Adding Web modules

On the **Adding Web Modules** panel, add one or more WAR files tothe application.

1. To add WAR files, click **Add**. Use the file browser tolocate and select the files and then click **Open**. Asprompted, verify or enter a file name and context root for the WARfile. Click **OK**. The WAR files are displayed in atable on the wizard panel.
2. To remove a WAR file, select the file and then click**Remove**.
3. Continue adding or removing files as needed. When you are finishedadding WAR files, click **Next**.

Adding application client modules

On the **Adding Application Client Modules** panel, add one or moreapplication client JAR files to the application.

1. To add JAR files, click **Add**. Use the file browser tolocate and select the files and then click **Open**. Asprompted, verify or enter a file name and alternate deployment descriptor forthe JAR file. Click **OK**. The JAR files are displayedin a table on the wizard panel.
2. To remove a JAR file, select the file and click **Remove**.
3. Continue adding or removing files as needed.
4. When you are finished adding application client JAR files, click**Next**.

Adding security roles

On the **Adding Security Roles** panel, specify a security role thata principal must be granted in order to access the application. Thesesecurity roles must be unique to the application. Any security rolesdefined in the application's modules are also displayed in the table onthis panel.

1. Click **Add**. Type a role name (required).Optionally, enter a description of the role. View the help for 6.6.5.0.5: Assembly properties for security roles. Click **OK**. The role and itsdescription are displayed in a table on the wizard panel.
2. Continue to add security roles as needed. If you need to remove arole, select the role in the table and then click **Remove**.

Setting additional properties and saving the archive

Click **Finish** to complete the wizard. To change settingsfor properties, click **Back** to return to the appropriatepanel. Make any needed changes, and then click**Finish**.

After you click **Finish**, the contents of the archive aredisplayed in the navigation pane. You can continue adding or modifyingproperties as needed. For example, you can add bindinginformation. When you are finished editing the archive, click**File**->**Save** to save the archive file.

6.6.2: Administering nodes (overview)

Administrative server node configurations provide informationabout machines (physical hosts) involved in the IBM WebSphereApplication Server environment.

WebSphere Application Server supports one administrativeserver per physical machine. Therefore, it does not matterwhether the administrator considers the physical machineor the administrative server to be the "node."

(The current exception is for theAS/400 operating system, on which the product supports multipleadministrative servers on each physical machine).

A node can be in the "running" state, which indicates notonly that the physical machine and operating system are running,but that the WebSphere administrative server on the machine is alsorunning.

The administrative server must be running in orderfor the node to be active and operational in the administrativedomain.

Regenerating the WebSphere plug-in for the Web server

It is suggested you review the information about administering WebSphereplug-ins for Web servers, because regenerating (updating) the plug-inis a task associated with the administrative node.

See instructions for regenerating the plug-in usingthe Java-based administrative console.

6.6.2.0: Node properties

Key:

Applies to Java administrative console of Advanced Edition Version 4.0

Applies to Web administrative console of Advanced Single Server Edition Version 4.0

Applies to Application Client Resource Configuration Tool

Name or Node Name

A logical name for the node, such as "localhost" or the short (unqualified) DNS name of the node. Machine names must be unique within an administrative domain.

Status

The current state of the node, such as running or stopped

The Advanced Edition administrative console has tabbed pages showing the installed JDBC providers, URL providers, JMS providers, and J2C Adapters.

6.6.2.1: Administering nodes with the Java administrative console

Use the Java administrative console to administer WebSphere administrative nodes.

Work with resources of this type by locating them in the tree view:

WebSphere Administrative Domain -> Nodes -> ***node_name***

6.6.2.1.1: Configuring new nodes with the Java administrative console

There is no administrative task for configuring a new node.The product automatically displays a node in the tree view when the administrative server for that node is configured to use the same administrative database as other nodes in the domain displayed by the tree view. In other words, in a centralized administrationsituation, as described in the administrative overview of administrative servers.

6.6.2.1.2: Reconnecting administrative nodes with the Java administrative console

1. Locate the node in the tree view.
2. Right-click it, then select **Reconnect**.

6.6.2.1.3: Regenerating WebSphere plug-in configurations for entire administrative nodes with the Java administrative console

During this task, you will manually trigger an update of the configuration forthe WebSphere plug-in for the Web server you are using.

1. Locate the node in the tree view.
2. Right-click it, then select **Regen Webserver Plugin**.

⚠ This task can overwrite manual configuration changes that youmight want to preserve instead. Before performing this task,understand its implications as described in the articles about administering Web server plug-ins.

6.6.2.1.4: Configuring tracing on administrative nodes with the Java administrative console

1. Locate the node in the tree view.
2. Right-click it, then select **Trace...**.
3. Use the tree view to select the Components or Groups to trace. By selectingan item in the tree, you select it and all of the items in the subtree beneathit.
4. Specify the ring buffer size and other properties as described in trace administration properties reference.
5. Click **OK** when finished.

The Trace... dialog can be used to perform various tracing tasks, includingdumps, as described in the article about administering traces, messages, and logs.

6.6.3: Administering application servers

An application server configuration provides information forstarting and managing a server process to handle requests for enterpriseapplications and their components.

The WebSphere administrator configures one or more applicationservers with settings for:

- Giving the server a unique name by which to manage it
- Associating the server with a unique transport port
- Specifying Java command line arguments and environment variables for starting theapplication server process
- Enabling WebSphere security for the server
- Associating the server with an operating system user ID and security group
- Specifying how many times to try to start or ping the server before giving up
- Specifying the process priority on the operating system
- Defining standard in, standard error, standard out streams for the server runtime
- Specifying a working directory
- Conducting transactions
- Specifying a workload management policy for distributing work among the server and itsclones
- Specifying tracing of the server and its child components

Each application server HTTP transport needs a unique port number and a host alias

Each application server in an administrative domain needs tohave a unique port number for the HTTP transport of its Web container. Furthermore,the virtual host associated with the application server must have an aliases associated with the transport port. If there is a problem with these settings (such as two application servers with the same virtual host are trying to usethe same HTTP transport port), the typical symptom is that the second applicationserver will not start.

See virtual host administrative overview for further details and instructions.

Regenerating the WebSphere plug-in for the Web server

It is suggested you review the information about administering WebSphereplug-ins for Web servers, because you will need to regenerate the plug-in configurationeach time you create a new application server.

See instructions for regenerating the plug-in usingthe Java-based administrative console.

6.6.3.0: Application server properties

Key:

☕ Applies to Java administrative console of Advanced Edition Version 4.0

🌐 Applies to Web administrative console of Advanced Single Server Edition Version 4.0

C:\ Applies to Application Client Resource Configuration Tool

An application server contains a variety of configurations:

- General, Advanced, and File properties ☕ or Process Definition 🌐
- Transaction properties ☕ 🌐
- Custom service properties ☕ 🌐
- JVM properties ☕ 🌐
- EJB container service properties ☕ 🌐
- Web container service properties ☕ 🌐
- Transport configuration properties ☕ 🌐
- Session Manager service properties ☕ 🌐
- Trace service properties ☕ 🌐
- Object Level Trace service properties ☕ 🌐
- Performance Monitor service properties ☕
- Object Request Broker (ORB) service properties ☕ 🌐
- Security service properties 🌐

... as well as these additional settings:

Application Server Name ☕ **or Name** 🌐

A logical name for the application server. The name must unique within the node (physical machine) containing the application server.

Execution State 🌐 **or Node Startup State** ☕

For the Advanced Single Server Edition: The state that you would like for the application server to be in, the next time the *product* is stopped and started again.

For the Advanced Edition: The state you would like the application server to be in, the next time *the node containing the application server* is stopped and started again.

Choices are Stopped, Running, and Last State (meaning the last state that the applicationserver was in, before node was stopped).

Module Visibility

The classloader isolation mode to use for the application server. By changing the default visibility level, it is possible to have visibility of classes in other modules, or even otherapplications.

- Specify "SERVER" to allow all application classloaders on the system to have visibility of all other application classloaders in the system. Search order is the same order as when the modules were initialized into the system.
- Specify "APPLICATION" to allow all classloaders in a J2EE application to have visibility of other classloaders in the same application. Search order is the order the modules are defined in the application.xml for the EAR.
- Specify "MODULE" to use one classloader per module. Each module (EAR, JAR, or WAR) has its own unique classloader. Visibility to other modules in the application is only achieved when MANIFEST Class-Path entries are added to a module.
- Specify "COMPATIBILITY" for compatibility with applications from WebSphere Application Server 3.5.x and 3.0.2.x. In this mode, all EJB module classloaders have visibility of all other EJB module classloaders and all Web Application modules have visibility of the EJB classloaders. Search order for the EJB classloaders is determined by the order in which the EJB modules were initialized.

Portable J2EE applications should be written with Module-level visibility. The othermodes are provided to accomodate applications that have different visibility requirementsthat cannot be modified.

The default module visibility differs between *Advanced Edition* and *Advanced Single Server Edition*.

- The *Advanced Edition* default is MODULE
- The *Advanced Single Server Edition* default is COMPTABILITY

Suppose you have a WAR module that depends on an EJB JAR installed in a different application or the same application, but the WAR module lacks a manifest classpath entry. Such a WAR module will run on *Advanced Single Server Edition* using the default settings, but will not run on *Advanced Edition* until you reset the Module Visibility field to COMPATIBILITY.

If you do not want to run *Advanced Edition* WAR modules in COMPATABILITY mode, then specify the classpath during assembly for the WAR module that you are going to run on *Advanced Edition*.

Name

See Application Server Name

Node

The administrative node on which the application server resides

Node Startup State

See Execution State

6.6.3.1: Administering application servers with the Java administrative console

Use the Java administrative console to administer WebSphere application servers.

Work with resources of this type by locating them in the tree view:

WebSphere Administrative Domain -> Nodes -> ***node_name*** **->Application Servers ->** ***application_server_name***

Expand the tree further to see the Installed EJB Modules and Installed Web Moduleson an application server.

6.6.3.1.1: Configuring new application servers with the Java administrative console

The product offers several ways to configure new application servers:

- By clicking **Console -> Wizards -> Create Application Server** from theconsole menu bar.
- By clicking **Create Application Server** from the drop-down list on theWizards toolbar button.
- Using menus on resources in the tree view

The first two methods lead to the Create Application Server task wizard, for which detailed help is provided here.

Using the Create Application Server wizard

You use the Create Application Server wizard to define a new application server and toset values for some properties pertaining to application servers. To set all values for these properties, use the properties dialogs for application server and web container.

- Naming the application server
- Specifying services used by the application server
- Completing the application server

Naming the application server

On the Specifying Application Server Properties panel:

1. Name the application server. The name must be unique within the node containing the application server.
2. Specify the node (physical machine) on which the application server will reside. For example, if the application server will reside on your machine, specify the hostname of your machine.
3. Click **Next**.

Specifying services used by your application server

On the Enabling Other Services panel:

1. If your application server will use services other than those already indicated or if you want to edit a service's property values, select a service from the table and click **Edit Service Properties**.
2. Edit service properties as needed.

   View EJB container properties help
   View Web container properties help
   View Session Manager properties help
   View Trace Service properties help
   View Object Level Trace properties help
   View Resource Analyzer settings help
   View Object Request Broker properties help
3. Repeat steps 1 and 2 to add another service or to edit another service's properties.
4. Click **Next**.

Completing your application server

The Completing the Create Application Server Wizard panel lists the application server name, the node onto which the server should be installed, and any other property values you specified for your application server.

If you do not want to change the values specified, click **Finish**. Thewizard will create a new application server with the values you specified, and display a message indicating where the application server resides on your system. If the wizard encounters an error, a message will display explaining why an application server could not be created.

To change the values specified, click **Back** to return to the appropriatepanel(s), make any needed changes, and then click **Finish** on this panel.

6.6.3.1.2: Starting and stopping application servers with the Java administrative console

1. Make sure the administrative server is running.
2. Open the Java administrative console.
3. Locate the application server in the tree view.
4. Right-click it and select **Start** or **Stop**.

Stopping application servers more gracefully

An application server can be stopped in a more graceful mannerif you set a quiesce period to occur prior to shutdown.During the quiesce period,incoming request activity is turned away (which, in a clustered environment, allows the request to be rerouted to a different application server clone).

Two configurable system properties affect the behaviorof the quiesce period.These two properties can be set on the command lines ofindividual application servers or on the server group command linein the administrative console.They are as follows:

`com.ibm.ejs.sm.server.quiesceTimeout=180`

This property represents the longest time (in seconds)to wait for server activity to end.The default is 180 seconds (3 minutes).A time-out value of 0 disables the quiesce feature;the application server stops without a quiesce period.The graceful stop usually occurs before the time-out period expires.The quiesce period ends after all requests that are currently in the application server have finished and there has not been a newincoming request for at least the quiesceInactiveRequestTime number of seconds(see the following property).If these conditions are not met before the time-out period ends,the application shuts down anyway.The default of 180 seconds should be reasonable in most environments,but you might need to modify the setting whenthe average time for a request to finish in an application server is longer.

`com.ibm.ejs.sm.server.quiesceInactiveRequestTime=5`

This property represents how much time should pass withoutnew incoming requests to the application server.The default value is 5 seconds.In a clustered environment,clients are informed of the impending shutdown and should stop sending requests before the end of the quiesce period.

Important:this quiesce feature is enabled only for the *stop* action.The *force stop* action is not affected;this immediately ends the application server process.

See also information about variations of starting andstopping.

6.6.3.1.3: Editing services of application servers with the Java administrative console

During this task, you will edit the settings of one of the many services thatcomprise an application server.

1. Locate the application server in the tree view.
2. Click it to display its properties in the properties view.
3. Click the **Services** tabbed page. (If editing a custom service, click the **Custom**tabbed page instead).
4. Select the service from the list of services.
5. Click **Edit Properties**. (On the **Custom** tab, it is simply **Edit**).
6. Specify properties for the service. See the properties references for:
    - EJB container service
    - Web container service
    - Session Manager service
    - Trace service
    - Object Level Trace service
    - Performance Monitoring settings
    - ORB service
    - Custom service

    If you cannot see all of the services (except Custom service) on the **Services**tabbed page, expand the console window vertically.
7. Click **OK** when finished, to close the service properties window.
8. Click **Apply** on the application server properties view.

6.6.4: Administering EJB containers (overview)

A container configuration provides information about an enterprisebean container. The administrator can specify several properties toaddress basic questions about the container location and behavior.

Specifying the server in which the container will reside

Each enterprise bean container resides in a particular application server.

When the administrator adds a new container to the WebSphere administrativedomain, he or she must associate the container with a particular server (alsoknown as the container's parent).

An application server can host multiple containers.

Specifying how beans in the container will get database connections

Every container can support the two main bean types, session beans and entitybeans:

- Entity beans require database connections because they store permanent data.
- Session beans do not *require* database access, though they can obtain it indirectly (as needed) by accessing entity beans.

A data source is an administrative resource that defines a pool of database connections. Servlets and enterprise beans use data sources to obtain database connections.

When configuring a container, the administrator can specify a default data source for the container. This data source will be the default data source used by any entity beans installed in the container that use container managed persistence (CMP).

When configuring a CMP entity bean, the administrator can specify which data source the container must use for managing the persistent state of the entity bean. If the administrator specifies a data source for an individual CMP entity bean then this data source will override any data source specified on the container.

Specifying a default data source is optional if each CMP entity bean in the container has a data source specified in it configuration. If a default data source is not specified and a CMP entity bean is installed in that container without specifying a data source for that bean then it will not be possible to start that CMP entity bean.

The default data source for a container is secure. When specifying it, the administrator must provide the user ID and password for accessing the data source.

Specifying how the container will manage cached bean instances

Each container keeps a cache of bean instances for ready access. The WebSphere administrator specifies settings governing the cache size and a policy for removing unused items from the cache.

Specifying where the container will passivate beans to make room in its cache

A container can *passivate* session beans to make room in its cache. The container saves a serialized session bean to a file. It restores the bean tothe cache when more room is available.

The WebSphere administrator specifies a passivation directory inwhich to keep the files.

6.6.4.0: EJB container properties

Key:

Applies to Java administrative console of Advanced Edition Version 4.0

Applies to Web administrative console of Advanced Single Server Edition Version 4.0

Applies to Application Client Resource Configuration Tool

Application Server

The application server of which the EJB container is a part

Default Data Source

The data source to use to connect to a JDBC-compliant database, such as DB2

Cache clean-up interval or Clean-up Interval

The interval at which the container attempts to remove unused items from the cache to reduce the total number of items in the cache to the value of the Cache preferred limit property.

The cache manager tries to maintain some unallocated entries that can be allocated quickly, as needed. A background thread attempts to free some entries while ensuring that some unallocated entries are maintained. If the thread runs while the application server is idle, then when the application server needs to allocate new cache entries, it does not pay the performance cost of removing entries from the cache.

In general, increase this parameter as the cache size increases.

For the Advanced Single Server Edition: This value must be a positive integer specified in milliseconds.

Cache size

The number of buckets in the cache hash table

If you change this value, change the Cache absolute limit property to correspond. For example, if you change the cache size to 3000, change the cache absolute limit to 3000, unless for some reason you do not want all of the available cache to be used.

For Advanced Single Server Edition: This value must be a positive integer.

Inactive Pool Cleanup Interval

The interval at which inactive pools will be reduced to their minimum size.

For Advanced Single Server Edition: The value must be a positive integer specified in milliseconds.

Installed EJB Modules

The EJB modules that are installed in the EJB container of this server

Node

The node with which the application server is associated

Passivation Directory

The directory into which the container will save the persistent state of passivated session beans

Session beans are passivated when the container needs to reclaim space in the bean cache. At passivation time, the container serializes the bean instance to a file in the passivation directory and discards the instance from the bean cache. If, at a later time, a request arrives for the passivated bean instance, the container retrieves it from the passivation directory, deserializes it, returns it to the cache, and dispatches the request to it. If any step fails (for example, if the bean instance is no longer in the passivation directory), then the method invocation fails.

6.6.4.1: Administering enterprise bean containers with the Java administrative console

Use the Java administrative console to administer EJB container services.

Work with resources of this type by locating them in the application server properties:

1. Locate the EJB container service among the application serverservices.
2. When the application server properties are displayed in the properties view, click the **Services** tabbed page.
3. Locate the EJB container service in the list of services.

6.6.4.1.1: Configuring the EJB container services of application servers with the Java administrative console

During this task, you will specify settings for the EJB container service ofan existing application server.

1. Locate the service. (Select it in the list of services).
2. Click **Edit Properties** to display the properties of the service.
3. Specify values for the EJB container service properties.
4. When finished, click **OK**.

6.6.5: Administering EJB modules (overview)

Administration of EJB modules consists of the following:

- Creating the module, setting deployment descriptor properties, and generatingcode for deployment using the Application Assembly Tool
- Setting additional configuration properties using the console

Classpath considerations

An important classpath-related setting to note is the Module Visibility. This application server setting impacts the portability of applications and standalone modules from other WebSphere Application Server versions and editions. If your existing module does not run as-is when you transfer it to Version 4.0, you might need to reassemble an existing module or change the module visibilitysetting.

See the information on setting classpaths for a full discussion of classpath considerations. See the applicationserver property reference for information about the module visibility setting.

6.6.5.0: EJB module properties

Key:

Applies to Java administrative console of Advanced Edition Version 4.0

Applies to Web administrative console of Advanced Single Server Edition Version 4.0

Applies to Application Client Resource Configuration Tool

Application or Application Ref

The application installation binding within which the module-to-server installation binding is contained. This is typically the logical name of the enterprise application you configured to contain this EJB module.

Default Data Source

The data source to use for this EJB module, unless a different one is specified

Execution State

The state that you would like the enterprise bean module to be in, the next time the product is stopped and started again

Name or Module Name

An administrative name for the EJB module

Password

The password corresponding to the specified user ID

Server

The application server on which the EJB module is installed

URI

A URI that, when resolved relative to the application URL, specifies the location of the module archive on a file system. The URI must match the URI of a ModuleRef URI in the deployment descriptor of an application if the module was packaged as part of a deployed application (EAR).

User ID

The user ID for accessing the default data source

See also the other application server properties.

6.6.5.0.1: Assembly properties for entity beans

EJB name (Required, String)

Specifies a logical name for the enterprise bean. This name must beunique within the EJB module. There is no relationship between thisname and the JNDI name.

Display name

Specifies a short name that is intended to be displayed by GUIs.

Description

Contains text describing the entity bean.

Home Interface (Required, String)

Specifies the full name of the enterprise bean's home interfaceclass--for example,com.ibm.ejs.doc.account.AccountHome.

Remote Interface (Required, String)

Specifies the full name of the enterprise bean's remote interfaceclass--for example,com.ibm.ejs.doc.account.Account.

EJB class (Required, String)

Specifies the full name of the enterprise bean class--for example,com.ibm.ejs.doc.account.AccountBean.

Persistency

Specifies whether an entity bean manages its own persistent storage orwhether storage is managed by the container. The valid values areBean-managed and Container-managed.

Primary key field

Specifies the name of a simple primary key. Simple primary keys mapto a single field in the entity bean class and are made up of primitive Javadata types (such as integer or long). If exactly one CMP field is theprimary key, it can be specified here.

Primary key class (Required, String)

Specifies the full name of the bean's primary key class--forexample,com.ibm.ejs.doc.account.AccountKey.Composite primary keys map to multiple fields in the entity bean class (or todata structures built from the primitive Java data types), and must beencapsulated in a primary key class. More complicated enterprise beansare likely to have composite primary keys, with multiple instance variablesrepresenting the primary key. A subset of the container-managed fieldsis used to define the primary key class associated with each instance of anenterprise bean.

Reentrant

Specifies whether the entity bean is reentrant or not. If anenterprise bean is reentrant, it can invoke methods on itself or call anotherbean that invokes a method on the calling bean. Only entity beans canbe reentrant. If an entity bean is not reentrant and a bean instance isexecuting a client request in a transaction context and another client usingthe same transaction context makes a request on the same bean instance, theEJB container throws the java.rmi.RemoteException exception tothe second client. If a bean is reentrant, the container cannotdistinguish this type of illegal loopback call from a legal concurrent calland the bean must be coded to detect illegal loopback calls.

Small icon

Specifies a JPEG or GIF file containing a small image (16x16pixels). The image is used as an icon to represent the entity bean in aGUI.

Large icon

Specifies a JPEG or GIF file containing a large image (32x32pixels). The image is used as an icon to represent the entity bean in aGUI.

Inheritance root

Specifies whether or not the enterprise bean is at the root of aninheritance hierarchy.

Bean Cache -- Activate at

Specifies the point at which an enterprise bean is activated and placed inthe cache. Removal from the cache and passivation is also governed bythis setting. Valid values are Once and Transaction. Onceindicates that the bean is activated when it is first accessed in the serverprocess, and passivated (and removed from the cache) at the discretion of thecontainer--for example, when the cache becomes full. Transactionindicates that the bean is activated at the start of a transaction andpassivated (and removed from the cache) at the end of the transaction.The default value is Transaction.

Bean Cache -- Load at

Specifies when the bean loads its state from the database. Thevalue of this property implies whether the container has exclusive or sharedaccess to the database. Valid values are Activation andTransaction. Activation indicates that the bean is loaded when it isactivated (Once or Transaction) and implies that the container has exclusiveaccess to the database. Transaction indicates that the bean is loadedat the start of a transaction and implies that the container has shared accessto the database. The default is Transaction.

The settings of the Activate at and Load at properties govern which commitoptions are used. The commit options are described in detail in theEnterprise JavaBeans specification, version 1.1.

- For Commit Option A (implies exclusive DB access), use Activate at = Onceand Load at = Activation. This option reduces database I/O (avoidscalls to the ejbLoad function) but serializes all transactions accessing thebean instance. Option A can increase memory usage by maintaining moreobjects in the cache, but could provide better response time if bean instancesare not generally accessed concurrently by multiple transactions.
- For Commit Option B (implies shared DB access), use Activate at = Once,Load at = Transaction. Option B can increase memory usage bymaintaining more objects in the cache. However, because eachtransaction creates its own copy of an object, there can be multiple copies ofan instance in memory at any given time (one per transaction), requiring thatthe database be accessed at each transaction. If an enterprise beancontains a significant number of calls to the ejbActivate function, usingOption B can be beneficial because the required object is already in thecache. Otherwise, this option does not provide significant benefit overOption A.
- For Commit Option C (implies shared DB access), use Activate at =Transaction and Load at = Transaction. This option can reduce memoryusage by maintaining fewer objects in the cache, however, there can bemultiple copies of an instance in memory at any given time (one pertransaction). This option can reduce transaction contention forenterprise bean instances that are accessed concurrently but notupdated.

Locale location

Specifies the language used when the enterprise bean retrieves anddisplays message catalogs: the local language of the client that invokedthe bean method or the local language of the server where the bean isrunning. Valid values are server and caller.

Local Transactions -- Boundary

Specifies when a local transaction begins. The default behavior isthat the local transaction begins when the method begins and ends when themethod ends.

Local Transactions -- Unresolved action

Specifies the action the container must take if resources are uncommittedby an application in a local transaction. A local transaction contextis created when a method executes in what the EJB specification refers to asan unspecified transaction context. Valid values are Rollback andCommit. The default is Rollback.

Local Relationship Roles - Name

Information is not available.

Local Relationship Roles - Source EJB Name

Information is not available.

Local Relationship Roles - is Forward

Information is not available.

Local Relationship Roles - is Navigable

Information is not available.

JNDI name

Specifies the JNDI name of the bean's home interface. This isthe name under which the enterprise bean's home interface is registeredand therefore is the name that must be specified when an EJB client does alookup of the home interface.

Data Source - JNDI name

Specifies the JNDI name for the bean's data source.

Default Authorization - User ID

Specifies the default user ID for connecting to a data source.

Default Authorization - Password

Specifies the default password for connecting to a data source.

6.6.5.0.2: Assembly properties for CMP fields

Container-managed persistence (CMP) fields define the variables in the beanclass for which the container must handle persistence management.

Name (Required, String)

Specifies a subset of public variables in an enterprise bean'simplementation class.

6.6.5.0.3: Assembly properties for method extensions

Method type

Specifies the type of the enterprise bean method. Valid values areHome methods, Remote methods, and All methods.

Name

Specifies the name of an enterprise bean method, or the asterisk character(*) . The asterisk is used to denote all the methods of the specifiedinterface--for example, all methods in the remote interface.

Parameters

Contains a list of fully qualified Java type names of the methodparameters. Used to identify a single method among multiple methodswith an overloaded method name.

Isolation level attributes

The transaction isolation level determines how isolated one transaction isfrom another. This property can be set for individual methods in anenterprise bean or for all methods in the enterprise bean. An asteriskis used to indicate all methods in the bean. However, within atransactional context, the isolation level associated with the first methodinvocation becomes the required isolation level for all methods invoked withinthat transaction. If a method is invoked with a different isolationlevel from that of the first method, thejava.rmi.RemoteException exception is thrown.

Isolation level

Specifies the level of transactional isolation. The valid valuesare as follows:

- Serializable. This level prohibits the following types ofreads:
    - Dirty reads, where a transaction reads a database row containinguncommitted changes from a second transaction.
    - Nonrepeatable reads, where one transaction reads a row, a secondtransaction changes the same row, and the first transaction rereads the rowand gets a different value.
    - Phantom reads, where one transaction reads all rows that satisfy an SQLWHERE condition, a second transaction inserts a row that also satisfies theWHERE condition, and the first transaction applies the same WHERE conditionand gets the row inserted by the second transaction.
- Repeatable read. This level prohibits dirty reads and nonrepeatablereads, but it allows phantom reads.
- Read committed. This level prohibits dirty reads, but allowsnonrepeatable reads and phantom reads.
- Read uncommitted. This level allows dirty reads, nonrepeatablereads, and phantom reads.

The container uses the transaction isolation level attribute asfollows:

- Session beans and entity beans with bean-managed persistence(BMP)--For each database connection used by the bean, the container setsthe transaction isolation level at the start of each transaction unless thebean explicitly sets the isolation level on the connection.
- Entity beans with container-managed persistence (CMP)--The containergenerates database access code that implements the specified isolationlevel.

Security Identity

Specifies that a principal's credential properties are to be handledas indicated in the Run-As mode property. Checking this box makes theRun-as Mode properties editable.

Description

Contains text describing or commenting on the securityinstructions.

Run-As Mode

Credential information is used by the security service to determine thepermissions that a principal has on various resources. At appropriatepoints, the security service determines whether the principal is authorized touse a particular resource based on the principal's permissions. Ifthe method invocation is authorized, the security service does the followingwith the principal's credential properties based on the value of theRun-as Mode property of the enterprise bean:

- Use Identity of Caller -- the security service makes no changes tothe principal's credential properties.
- Use Identity of EJB Server -- the security service alters theprincipals credential properties to match the credential properties associatedwith the EJB server.
- Use Identity Assigned to Specified Role -- A principal that has beenassigned to the specified security role is used for the execution of thebean's methods. This association is part of the applicationbinding where the role is associated with a user ID and password of a user whois granted that role.

Role Name

Specifies the name of a security role. If the Use Identity Assignedto Specified Role button is selected, a principal that has been granted thisrole will be used.

Description

Contains a description of the security role.

Access Intent - Intent type

Specifies whether the method is a read-only method or whether the methodcan update data (or invoke other methods that can update data, in the sametransaction). The legal values are read or update (read/write).

Finder descriptor - User

Specifies that the user has provided a finder helper class in the entitybean's home interface. The class contains specialized findermethods.

Finder descriptor - EJB QL

Describes the semantics of a finder method using the EJB QL (EnterpriseJavaBeans query language). EJB QL is a declarative, SQL-like languageintended to be compiled to the target language of the persistent data storeused by a persistence manager. The language is independent of thebean's mapping to a relational datastore and is thereforeportable. The EJB query specifies a search based on the persistentattributes and relationships of the bean. An EJB query contains aSELECT clause (optional), a FROM clause (required), a WHERE clause (optional),and an ORDER BY clause (optional). The SELECT clause specifies the EJBobjects to return. The FROM clause specifies the collections of objectsto which the query is to be applied. The WHERE clause contains searchpredicates over the collections. The ORDER BY clause specifies theordering of the resulting collection.

Finder descriptor - Full select

Note: For information on restrictions, see the documentation forDeployment Tool for Enterprise JavaBeans.

Describes the semantics of the finder method using a SQL SELECTstatement. The SELECT statement indicates the EJB objects toreturn.

Finder descriptor - Where clause

Note: For information on restrictions, see the documentation forDeployment Tool for Enterprise JavaBeans.

Describes the semantics of the finder method using a SQL WHEREclause. This clause restricts the results that are returned by thequery.

6.6.5.0.4: Assembly properties for session beans

EJB name (Required, String)

Specifies a logical name for the enterprise bean. This name must beunique within the EJB module. There is no relationship between thisname and the JNDI name.

Display name

Specifies a short name that is intended to be displayed by GUIs.

Description

Contains text describing the session bean.

Home interface (Required, String)

Specifies the full package name of the enterprise bean's homeinterface class, for example,com.ibm.ejs.doc.account.AccountHome.

Remote interface (Required, String)

Specifies the full package name of the enterprise bean's remoteinterface class, for example,com.ibm.ejs.doc.account.Account.

EJB class (Required, String)

Specifies the full package name of the enterprise bean class, for example,com.ibm.ejs.doc.account.AccountBean.

Session type

Specifies whether the enterprise bean maintains a conversational state (isa stateful session bean) or does not (is a stateless session bean).Valid values are stateful and stateless.

Transaction type

Specifies whether the enterprise bean manages its own transactions orwhether the container manages transactions on behalf of the bean. Validvalues are container or bean.

Small icon

Specifies a JPEG or GIF file containing a small image (16x16pixels). The image is used as an icon to represent the session bean ina GUI.

Large icon

Specifies a JPEG or GIF file containing a large image (32x32pixels). The image is used as an icon to represent the session bean ina GUI.

Timeout

Specifies the idle timeout value for the enterprise bean inseconds. A zero (0) value indicates that idle bean instances time outafter the maximum allowable timeout period elapses. By default, thetimeout is 600 seconds or 10 minutes. This property does not apply tosession beans.

Inheritance root

Specifies whether the enterprise bean is at the root of an inheritancehierarchy.

Bean Cache -- Activate at

Applies to stateful session beans only, not to stateless beans.Specifies the point at which an enterprise bean is activated and placed in thecache. Removal from the cache and passivation are also governed by thissetting. Valid values are Once and Transaction. Once indicatesthat the bean is activated when it is first accessed in the server process andpassivated (and removed from the cache) at the discretion of thecontainer--for example, when the cache becomes full. Transactionindicates that the bean is activated at the start of a transaction andpassivated (and removed from the cache) at the end of the transaction.The default value is Once.

Bean Cache -- Load at

This property does not apply to session beans.

Locale location

Specifies the language used when the enterprise bean retrieves anddisplays message catalogs: the local language of the client that invokedthe bean method or the local language of the server where the bean isrunning. Valid values are server and caller.

Local Transactions -- Boundary

Specifies when a local transaction begins. The default behavior isthat the local transaction begins when the method begins and ends when themethod ends. This property is not applicable for session beans.

Local Transactions -- Unresolved action

Specifies the action the container must take if resources are uncommittedby an application in a local transaction. A local transaction contextis created when a method executes in what the EJB specification refers to asan unspecified transaction context. Valid values are Rollback andCommit. The default is Rollback.

JNDI name

Specifies the JNDI name of the bean's home interface. This isthe name under which the enterprise bean's home interface is registeredand therefore is the name that must be specified when an EJB client does alookup of the home interface.

6.6.5.0.5: Assembly properties for security roles

A security role is a logical grouping of principals. Access tooperations (such as EJB methods) is controlled by granting access to arole.

Role name (Required, String)

Specifies the name of a security role.

Description

Contains text describing the security role.

If specifying security roles at the application level (EAR file), thefollowing properties apply:

Role Name (Required, String)

Specifies the name of a security role that is unique to theapplication.

Description

Contains text describing the security role.

Binding -- Groups -- Name

Specifies user groups that are granted the security role.

Binding -- Users -- Name

Specifies users that are granted the security role.

Binding -- Special Subjects -- Name

Specifies one of two special categories of users to which roles can begranted: Everyone or All authenticated users. If the specialsubject Everyone is granted a role, then all users, including those who didnot authenticate, are granted the role. In other words, a method on anenterprise bean or a URI is unprotected if any of the required roles for thatmethod are granted to the special subject Everyone. In the case of Allauthenticated users, any user who can authenticate by using a valid user IDand password is considered to be granted that role.

6.6.5.0.6: Assembly properties for method permissions

A method permission is a mapping between one or more security roles and oneor more methods that a member of the role can invoke. Assemblyproperties for method permissions include an optional description, a list ofsecurity role names, and a list of methods. The security roles must bedefined, and the methods must be methods defined in the enterprise bean'sremote or home interfaces.

Method permission name

Specifies a name for the mapping between method permissions and securityroles.

Description

Contains text describing the mapping between method permissions andsecurity roles.

Methods - Name

Specifies the name of an enterprise bean method, or the asterisk (*)character. The asterisk is used to denote all the methods of thespecified interface--for example, all methods of the remoteinterface.

Methods - Enterprise Bean

Specifies the name of the enterprise bean that contains the method.

Methods - Type

Distinguishes between a method with the same signature that is defined inboth the home and remote interface. Valid values are All methods,Remote methods, or Home methods.

Methods - Parameters

Contains a list of fully qualified Java type names of the methodparameters. This property is used to identify a single method amongmultiple methods with an overloaded method name.

Roles -- Role Name

Specifies the security role that must be granted in order to invoke themethod.

6.6.5.0.7: Assembly properties for container transactions

Container transaction properties specify how the container must managetransaction scopes for the enterprise bean's method invocations.Specify one or more methods and associate a transaction attribute with eachmethod.

Name

Specifies a name for the mapping between a transaction attribute and oneor more methods.

Description

Contains text describing the mapping.

Transaction Attribute

Specifies how the container must manage the transaction boundaries whendelegating a method invocation to an enterprise bean's businessmethod. The legal values are Never, Mandatory, Requires New, Required,Supports, Not Supported, and Bean Managed. The default is NotSupported.

- Mandatory. Directs the container to always invoke the bean methodwithin the transaction context associated with the client. If theclient attempts to invoke the bean method without a transaction context, thecontainer throws the javax.jts.TransactionRequiredExceptionexception to the client. The transaction context is passed to any EJBobject or resource accessed by an enterprise bean method.

  EJB clients that access these entity beans must do so within an existingtransaction. For other enterprise beans, the enterprise bean or beanmethod must implement the Bean Managed value or use the Required or RequiresNew value. For non-enterprise bean EJB clients, the client must invokea transaction by using the javax.transaction.UserTransactioninterface.
- Supports. Directs the container to invoke the bean method within atransaction context if the client invokes the bean method within atransaction. If the client invokes the bean method without atransaction context, the container invokes the bean method without atransaction context. The transaction context is passed to anyenterprise bean objects or resources that are used by this bean method.
- Never. Directs the container to invoke bean methods without atransaction context.
  - If the client invokes a bean method from within a transaction context, thecontainer throws the java.rmi.RemoteException exception.
  - If the client invokes a bean method from outside a transaction context,the container behaves in the same way as if the Not Supported transactionattribute was set. The client must call the method without atransaction context.
- Requires New. Directs the container to always invoke the beanmethod within a new transaction context, regardless of whether the clientinvokes the method within or outside a transaction context. Thetransaction context is passed to any enterprise bean objects or resources thatare used by this bean method.
- Not Supported. Directs the container to invoke bean methods withouta transaction context. If a client invokes a bean method from within atransaction context, the container suspends the association between thetransaction and the current thread before invoking the method on theenterprise bean instance. The container then resumes the suspendedassociation when the method invocation returns. The suspendedtransaction context is *not* passed to any enterprise bean objects orresources that are used by this bean method.
- Required. Directs the container to invoke the bean method within atransaction context. If a client invokes a bean method from within atransaction context, the container invokes the bean method within the clienttransaction context. If a client invokes a bean method outside atransaction context, the container creates a new transaction context andinvokes the bean method from within

that context. The transactioncontext is passed to any enterprise bean objects or resources that are used bythis bean method.

- Bean Managed. Notifies the container that the bean class directlyhandles transaction demarcation. This property can be specified onlyfor session beans, and it cannot be specified for individual beanmethods.

Method Elements - Name

Specifies the name of an enterprise bean method, or the asterisk character(*). The asterisk is used to denote all the methods of the specifiedinterface--for example, all methods of the remote interface.

Method Elements - Enterprise bean

Specifies which enterprise bean contains the methods indicated in the Nameproperty.

Method Elements - Type

Used to distinguish between a method with the same signature that isdefined in both the home and remote interface. Valid values are Allmethods, Remote methods, or Home methods. Use All methods if atransaction attribute applies to all methods of the bean.

Method Elements - Parameters

Contains a list of fully qualified Java type names of the methodparameters. This property is used to identify a single method amongmultiple methods with an overloaded method name.

6.6.5.0.aa: Assembly properties for EJB modules

File name (Required, String)

Specifies the file name of the EJB module, relative to the top level ofthe application package.

Alternative DD

Specifies the file name for an alternative deployment descriptor file touse instead of the original deployment descriptor file in the module'sJAR file. This file is the postassembly version of the deploymentdescriptor file. (The original deployment descriptor file can be editedto resolve dependencies and security information. Directing the use ofthe alternative deployment descriptor allows you to keep the originaldeployment descriptor file intact). The value of the Alternative DDproperty must be the full path name of the deployment descriptor file relativeto the module's root directory. By convention, the file is in theALT-INF directory. If this property is not specified, the deploymentdescriptor file is read directly from the module's JAR file.

Classpath

The path containing additional classes required by the application but notcontained in the module's archive file. The class loader uses thispath. Specify the values relative to the root of the EAR file andseparate the values with spaces. Absolute values that reference filesor directories on the hard drive are ignored. To specify classes thatare not in JAR files but are in the root of the EAR file, use a period andforward slash (./). Consider the following example directorystructure in which the file myapp.ear contains an EJB module namedmyejb.jar. Additional classes reside in class1.jar andclass2.zip. A class named xyz.class is not packaged in aJAR file but is in the root of the EAR file.

```
myapp.ear/myejb.jarmyapp.ear/class1.jarmyapp.ear/class2.zipmyapp.ear/xyz.class
```

Specify `class1.jar class2.zip ./` as thevalue of the Classpath property. (Name only the directory for.class files.)

Display name

Specifies a short name that is intended to be displayed by GUIs.

EJB client JAR

Specifies the location of the JAR file containing a subset of deployedclasses needed by the client.

Description

Contains text describing the module.

Small icon

Specifies a JPEG or GIF file containing a small image (16x16pixels). The image is used as an icon to represent the module in aGUI.

Large icon

Specifies a JPEG or GIF file containing a large image (32x32pixels). The image is used as an icon to represent the module in aGUI.

Generalizations -- Subtype

Information is not available.

Generalizations -- Supertype

Information is not available.

EJB Relationships -- Name

Information is not available.

EJB Relationships -- Source EJB name

Information is not available.

EJB Relationships -- Is forward

Information is not available.

EJB Relationships -- Is navigable

Information is not available.

Default Data Source -- JNDI name

Specifies the default JNDI name for the data source. This defaultis used if binding information is not specified in the deployment descriptorfor an individual enterprise bean.

Default Authorization - User ID

Specifies the default user ID for connecting to an enterprise bean'sdata store.

Default Authorization - Password

Specifies the default password for connecting to an enterprise bean'sdata store.

6.6.5.1: Administering EJB modules with the Java administrative console

Work with resources of this type by locating them in the tree view.

To locate modules according to their membership in enterprise applications, click:

WebSphere Administrative Domain -> Nodes -> ***node_name*** **-> Enterprise Applications ->** ***application_name*** **->EJB modules**

To locate modules installed on a particular application server, click:

WebSphere Administrative Domain -> Nodes -> ***node_name*** **-> Application Servers ->** ***application_server_name*** **->Installed EJB modules**

In either case, the instances of the module will be displayed in thedetails view.

6.6.5.1.1: Installing EJB modules with the Java administrative console

Use the application installation wizard to install EJB modules onto an application server.

6.6.5.1.2: Viewing deployment descriptors of EJB modules with the Java administrative console

To view the deployment descriptor of an EJB module:

1. Locate the EJB module instance.
2. Right-click your EJB module in the details view and select **View Deployment Descriptor** from the popup menu.

A window opens, providing descriptive information about the selected module's EJB JAR and its enterprise beans and assembly descriptors.

6.6.5.1.3: Showing the status of EJB modules with the Java administrative console

During this task, you will view the status (such as running or stopped) of each EJB module in an enterprise application.

1. Locate the EJB module instances.
2. Right-click an instance and select **Show Status** to display the Module Status window.
3. When finished viewing status, close the window.

6.6.5.1.4: Exporting table DDLs of EJB modules with the Java administrative console

You can export the DDL for each EJB module that contains container managed persistence (CMP) entity beans. You can save the DDL files to the location of your choice.

Later, you (or a database administrator) can use the saved DDL with the database vendor product to create database tables that correspond to the EJB modules. See Recreating database tables form the exported table DDL for instructions.

The data store that the DDL schema is loaded into should be the same data store specified in the container managed entity bean data source bindings. The bindings should be those that were specified for each enterprise bean, using the Application Installation wizard.

To export a DDL:

1. Locate the EJB module instance.
2. Right-click your EJB module in the details view and select **Export Table DDL** from the popup menu.
3. In the Export Table DDL dialog that opens:
    a. From the **Node** drop-down list, select a node to which the DDL should be exported.
    b. In the **Export directory** field, specify the target drive and directory for the DDL.
    c. Click **OK**.

The DDL will be exported to the specified node and directory, and will have the name *enterpriseApplicationName_ejbModuleName_*Table.ddl.

6.6.5.1.5: Moving EJB modules to other application servers with the Java administrative console

You can move an EJB module from one application server to a different application server or server group. To move a module:

1. Locate the EJB module instance.
2. Right-click your EJB module in the details view and select **Move** from the popup menu.
3. In the Select a target server or ServerGroup dialog that opens, specify the target application server or server group and click **OK**
4. Copy EAR files containing the module to the node that contains the destination server. If the destination is a server group, then copy the EAR file to all nodes that contain the clones of the destination server group.

6.6.5.5: Administering EJB modules with Application Assembly Tool

An EJB module is used to assemble one or more enterprise beans into asingle deployable unit. The Application Assembly Tool is used to createand edit EJB modules, verify the archive files, and generate deploymentcode. See the related topics for links to concepts, instructions forcreating an EJB module, and field help.

6.6.5.5.1: Creating an EJB module

EJB modules can be created by using the property dialog boxes or by using awizard.

- Using the property dialog boxes
- Using the Create EJB Module wizard

Using the property dialog boxes

The steps for creating an EJB module are as follows:

1. Click **File**->**New**->**EJB Module**. Thenavigation pane displays a hierarchical structure used to build the contentsof the module. The icons represent the components, assembly properties,and files for the module. A property dialog box containing generalinformation about the module is displayed in the property pane.
2. By default, the archive file name and the module display name are thesame. It is recommended that you change the display name in theproperty pane.
3. By default, the temporary location of the EJB module isinstallation_directory`/bin`. You must specify a newfile name and location by clicking **File**->**Save**.You must add at least one enterprise bean to the module before savingit. This is a requirement for a valid archive file.
4. Enter values for other properties as needed. View the help for 6.6.5.0.a: Assembly properties for EJB modules.
5. Add enterprise beans to the module. You must add at least oneenterprise bean. First, click the icon representing the type of beanbeing added (Session Beans or Entity Beans). There are several ways ofadding beans to a module:
   - Import an existing JAR or EAR file containing enterprise beans.Right-click the icon representing the enterprise bean type and choose**Import**. Click **Browse** to browse the file systemand locate the desired JAR file. When the file is located, click**Open**. Select the JAR file in the left window. Theenterprise beans in the selected JAR file are displayed in the rightwindow. Select the beans to be added and click **Add**.The selected items are displayed in the Selected Components window.Click **OK**. The property dialog box for the enterprise bean isautomatically populated with required values. Click the plus sign(+) next to the icon representing the bean type to verify that thebeans are included in the module.
   - Use a copy-and-paste operation to copy archive files from an existingmodule.
   - Create a new enterprise bean. Right-click the icon representing theappropriate bean type and choose **New**. For entity beans,choose whether the bean has container-managed or bean-managedpersistence. In the property dialog box, browse for and select theclass files that make up the bean. By default, the root directory orarchive is the current archive. If needed, browse the file system forthe directory or archive where the class files reside. After you choosea directory or archive, its file structure is displayed. Expand thestructure and locate the files that you need. Select the file and click**OK**. In the property dialog box, click **OK**.Verify that the beans are added to the module (expand the hierarchy for thebean type in the navigation pane). If there are one or more beans,display the properties for each bean by clicking the bean in the top part ofthe property pane. The corresponding property dialog box is displayedin the bottom part of the pane.
6. Specify properties for each enterprise bean. Expand the hierarchyfor each bean type. Click a bean instance and, if needed, edit or enterproperties for that bean. View the help for 6.6.5.0.1: Assembly properties for entity beans or 6.6.5.0.4: Assembly properties for session beans.
7. Add assembly properties for each bean. Click the plus sign(+) next to the bean instance to reveal

property groups.Click the icon representing a group of properties. If properties arealready defined (for example, for an imported bean), edit the properties inthe property pane. If properties are not defined, right-click theproperty icon and click **New**. A property dialog box isdisplayed. Enter values for the properties and click**OK**.

   - Specify Environment Entries. View the help for 6.6.34.0.a: Assembly properties for environment entries.
   - Specify EJB References. View the help for 6.6.43.0.1: Assembly properties for EJB references.
   - Specify Resource References. View the help for 6.6.43.0.2 Assembly properties for resource references.
   - Specify Security Role References. View the help for 6.6.43.0.3: Assembly properties for security role references.
   - Specify CMP Fields. View the help for 6.6.5.0.2: Assembly properties for CMP fields.
   - Specify Method Extensions. View the help for 6.6.5.0.3: Assembly properties for method extensions.

8. Add assembly properties for the EJB module. In the navigation pane,right-click each property group's icon. Choose **New** toadd new values.
   - Specify Security roles. View the help for 6.6.5.0.5: Assembly properties for security roles.
   - Specify Method permissions. View the help for 6.6.5.0.6: Assembly properties for method permissions.
   - Specify Container transactions. View the help for 6.6.5.0.7: Assembly properties for container transactions.
9. Add files needed by the application. Right-click the Filesicon in the navigation pane and choose **Add Files**. Click**Browse** to navigate to the desired directory or archive.Click **Select**. If you are adding an entire archive, selectthe directory that contains the archive. The directory structure isdisplayed in the left pane. Browse the directory structure. Fromthe right pane, select one or more files to be added and click**Add**. If you select a directory and click **Add**, allfiles in the directory, including the directory, are added. Theselected files are displayed in the Selected Files window. Relativepath names are maintained. When the Selected Files window contains thecorrect set of files, click **OK**.
10. Click **File**->**Save** to save the archive.

Using the Create EJB Module wizard

Use this wizard to create an EJB module. The module can then be usedas a stand-alone application, or it can become part of a J2EE applicationcontaining other modules. An EJB module consists of one or moreenterprise beans. You can use existing EJB JAR files (import them), orcreate new ones.

During creation of the EJB module, you specify the files for eachenterprise bean to be included in the module. You also specify otherinformation about the bean, such as security roles and references to otherenterprise beans and to resource connection factories. After definingthe enterprise beans to be included in the module, you specify assemblyproperties that apply to the module as a whole. Both bean and moduleinformation are used to create a deployment descriptor.

You can specify either of the following:

- One or more EJB 1.1 JAR files, either created manually or withVisualAge for Java. The enterprise beans must conform to J2EEspecifications. The JAR files can be deployed or undeployed.
- One or more EJB 1.0 JAR files. The Application Assembly Toolautomatically converts the file to the

EJB 1.1 specification format,but you must specify dependent classpaths if any.
- Enterprise bean class files (not residing in a JAR file).

The wizard creates an EJB module in the file location you specify.

To create an EJB module, click the **Wizards** icon on the tool barand then click **EJB Module**. Follow the instructions on eachpanel.

- Specifying EJB module properties
- Adding files
- Specifying EJB Client JAR and classpath
- Choosing EJB module icons
- Adding enterprise beans
- Adding security roles
- Adding method permissions
- Adding container transactions
- Setting additional properties and saving the archive

Specifying EJB module properties

On the **Specifying EJB Module Properties** panel:

1. Indicate the application to which this module is to be added. If aparent application is not indicated, the module is created as a stand-aloneapplication.
2. Specify a display name for the module. The display name is used bythe Application Assembly Tool to identify the module and can be used by othertools.
3. Specify a file name for the module. The file name specifies alocation on your system for the JAR file to be created.
4. Provide a short description of the module (optional).
5. Click **Next**.

Adding files

On the **Adding Files** panel, specify supplementary files (such aslibrary and utility files) that are to be included in your EJB module.To add or remove files:

1. Click **Add**. Use the file browser to choose one or morefiles. First, browse for the root directory or archive where the filesare located and click **Select**. If you are adding an entirearchive, select the directory that contains the archive. The directorystructure is displayed in the left pane. Browse the directorystructure. From the right pane, select one or more files to be addedand click **Add**. If you select a directory and click**Add**, all files in the directory, including the directory, areadded. Relative path names are maintained. The selected filesare displayed in the Selected Files window. Click **OK**.The files are displayed in a table on the wizard panel.
2. If you want to remove a file, select the file in the table and then click**Remove**.
3. Continue to add or remove files until you have the correct set offiles.
4. Click **Next**.

Specifying EJB Client JAR and classpath

On the **Specifying EJB Client JAR and Classpath** panel:

1. Specify the EJB Client JAR file. This is the location of the JARfile containing deployed classes needed by a client program for accessing theenterprise beans in this module.
2. Specify the path containing additional classes required by theapplication. This path is used by the classpath loader.
3. Click **Next**.

Choosing EJB module icons

On the **Choosing EJB Module Icons** panel, specify icons for yourmodule.

1. Specify the full path name of a file containing a small icon. Theicon must be a GIF or JPEG image 16x16 pixels in size.
2. Specify a full path name of a file containing a large icon. Theicon must be a GIF or JPEG image 32x32 pixels in size.
3. Click **Next**.

Adding enterprise beans

The **Adding Enterprise Beans** panel is used to add new enterprisebeans, import existing beans, or (if you are modifying this module or make amistake) remove beans.

To add a new enterprise bean:

1. Click **New**. On the dialog box, choose a bean type(session bean, entity bean with BMP, or entity bean with CMP). Click**OK**.
2. On the **Specifying Enterprise Bean Properties** panel, entervalues for the properties of the bean. Click **Browse** tolocate the root directory or archive where the bean class files reside.The files are displayed in a window. Locate and click the appropriatefile. Click **OK**. View the help for 6.6.5.0.1: Assembly properties for entity beans or 6.6.5.0.4: Assembly properties for session beans. Click **Next**.
3. On the **Specifying Container-managed persistence (CMP) fields**panel, define the variables for which the container must manage persistencemanagement. View the help for 6.6.5.0.2: Assembly properties for CMP fields. Click **Next**. This panel is visibleonly for entity beans with CMP.
4. On the **Specifying Specific Enterprise Bean Type Properties**panel, enter values for the properties of the bean. View the help for 6.6.5.0.1: Assembly properties for entity beans or 6.6.5.0.4: Assembly properties for session beans. Click **Next**.
5. On the **Choosing Enterprise Bean Icons** panel, specify icons forthe bean. Specify the full path name of a file containing a small iconand large icon. The icon must be a GIF or JPEG image (16x16 pixels or32x32 pixels in size).
6. On the **Adding Environment Entries** panel, enter values forenvironment entries. Click **Add** and enter a name and type(required). Click **OK**. The entry is displayed in thetable on the wizard panel. To remove an entry, select the entry andthen click **Remove**. View the help for 6.6.34.0.a: Assembly properties for environment entries. Click **Next**.
7. On the **Adding Security Role References** panel, enter values forsecurity role references. Click **Add** to enter a rolename. Click **OK**. The role name is displayed in a tableon the wizard panel. To remove a role, select the role in the table andthen click **Remove**. View the help for 6.6.43.0.3: Assembly properties for security role references. Click **Next**.
8. On the **Adding Resource References** panel, enter references forresource connection factories. Click

Add to add areference. You must enter values for a name, type, and authorizationmode. Click **OK**. The reference is displayed in a tableon the wizard panel. To remove a reference, select the reference in thetable and then click **Remove**. View the help for 6.6.43.0.2 Assembly properties for resource references. Click **Next**.

9. On the **Adding EJB References** panel, enter values for EJBreferences. Click **Add** to add a reference. You mustenter a value for the name, home interface, remote interface, and type.Click **OK**. The reference is displayed in a table on thewizard panel. To remove a reference, select the reference in the tableand then click **Remove**. View the help for 6.6.43.0.1: Assembly properties for EJB references.
10. Click **Finish**.
11. Continue to add more enterprise beans as needed.

To import an existing enterprise bean:

1. Click **Import**.
2. Browse the file system to locate the desired archive. The contentsof the archive are displayed in a window. Select one or more JARfiles. The enterprise beans in the JAR file are displayed in the rightwindow. Select an enterprise bean and then click **Add**.The enterprise beans are added to the Selected Components window. Click**OK**.

To remove an enterprise bean, select the enterprise bean in the table andthen click **Remove**.

Continue adding and removing enterprise beans as necessary. Click**Next**.

Adding security roles

On the **Adding Security Roles** panel:

1. Click **Add**. Type a role name and, optionally, type adescription. Click **OK**. The role name is displayed ina table on the wizard panel. View the help for 6.6.5.0.5: Assembly properties for security roles.
2. Continue to add security roles as needed. If you need to remove arole, select the role in the table and then click **Remove**.
3. Click **Next**.

Adding method permissions

On the **Adding Method Permissions** panel, indicate which securityroles are permitted to invoke which methods.

1. To add method permissions, click **Add**. Enter a name forthe method permission. View the help for 6.6.5.0.6: Assembly properties for method permissions.
2. Click **Add** next to the table of methods. Locate themethod in the JAR file, select it, and then click **OK**.
3. Click **Add** next to the table of security roles. Selectthe appropriate security role and click **OK**.
4. Verify the information and click **OK**. The methodpermission is displayed in a table on the wizard panel.
5. To add multiple method permissions, click **Add** on the wizardpanel and repeat the process.
6. Continue to add and remove methods and corresponding security roles asneeded. If you need to remove a method permission, select the item andthen click **Remove**.
7. Click **Next**.

Adding container transactions

On the **Adding Container Transactions** panel, indicate transactionattributes for the methods of the enterprise bean.

1. To add a container transaction, click **Add**. Enter a nameand choose a transaction attribute from the menu. View the help for 6.6.5.0.7: Assembly properties for container transactions.
2. Click **Add** to choose which methods are to be governed by thisattribute. Locate the method in the JAR file, select it, and then click**OK**.
3. Verify the information and click **OK**. The containertransaction is displayed in a table on the wizard panel.
4. To add multiple container transactions, click **Add** on the wizardpanel and repeat the process.
5. Continue adding or removing container transactions as needed. Ifyou need to remove a container transaction, select the item and then click**Remove**.

Setting additional properties and saving the archive

Click **Finish** to complete the wizard. To change settingsfor properties, click **Back** to return to the appropriatepanel. Make any needed changes, and then click**Finish**.

After you click **Finish**, the contents of the archive aredisplayed in an Application Assembly Tool window. Review the contentsin the navigation pane. You can continue adding or modifying propertiesas needed. For example, you can add binding information. Whenyou are finished editing the archive, click **File**->**Save**to save the archive file.

6.6.7: Administering Web containers (overview)

A Web container configuration provides information about the applicationserver component that handles servlet requests forwarded bythe Web server. The administratorspecifies Web container properties including:

- Application server on which the Web container runs
- Number and type of connections between the Web server and Web container
- Port on which the Web container listens

6.6.7.0: Web container properties

Web containers contain other resource types, whose properties are listedin separate property reference files. If you do not find a property in the followinglist, see below for links to the property references of other resource types comprising Web containers.

Key:

☕ Applies to Java administrative console of Advanced Edition Version 4.0

🌐 Applies to Web administrative console of Advanced Single Server Edition Version 4.0

C:\ Applies to Application Client Resource Configuration Tool

Allow thread allocation beyond maximum ☕

Allows the number of threads to increase beyond the maximum size configured for the thread pool

Application Server ☕

The application server associated with this Web container

Cache Size ☕ 🌐

A positive integer defining the maximum number of entries the cache will hold. Values are usually in the thousands, with no maximum or minimum.

Can Be Grown 🌐

Allows the number of threads to increase beyond the maximum size configured for the pool

Default Priority ☕ 🌐

The default priority for servlets that can be cached. It determines how long an entry will stay in a full cache. The recommended value is 1.

Dynamic Properties 🌐

A set of name-value pairs for configuring properties beyond those displayed in the interface

Enable Dynamic Cache 🌐 **or Enable Servlet Caching** ☕

Enable the servlet and JSP dynamic JNDI caching feature

External Cache Groups ☕

For servlets that can be cached, specifies the external groups to which their entries should be sent

Enable Servlet Caching ☕

See Enable Dynamic Cache

HTTP Transports 🌐 **or Transport** ☕

The HTTP transports associated with this Web container. See also transport properties

Inactivity Timeout 🌐 **or Thread Inactivity Timeout** ☕

The period of time after which a thread should be reclaimed due to inactivity

Installed Web Modules

The Web modules that are installed into the Web container of this server

Maximum Size **or Maximum Thread Size**

The maximum number of threads to allow in the pool

Minimum Size **or Minimum Thread Size**

The minimum number of threads to allow in the pool

Name (External Cache Group)

See External Cache Groups

Node

The node with which this Web container is associated

Session Manager

The Session Manager associated with this Web container. See also Session Manager properties

Thread Inactivity Timeout

See Inactivity Timeout

Thread Pool

The thread pool settings for the Web container

Transport

See HTTP Transports

Type

Only shared external cache groups are supported at this time

Additional properties related to Web containers

Web containers contain other resource types, whose properties are listedin separate property reference files. If you do not find the property in theabove list ...

See also the:

- application server properties
- HTTP Transport properties

For Advanced Single Server Edition, see also the:

- Session Manager properties
- Web module properties

6.6.7.1: Administering Web containers with the Java administrative console

Use the Java administrative console to administer Web containers.

Work with resources of this type by locating them in the application server properties:

1. Locate the application server instance containing the service.
2. When the application server properties are displayed in the properties view, click the **Services** tabbed page.
3. Locate the Web container service in the list of services.

6.6.7.1.1: Configuring the Web container services of application servers with the Java administrative console

During this task, you will specify settings for the Web container service ofan existing application server.

1. Locate the Web container service in the application server properties.
2. Click **Edit Properties** to display the container service properties dialog.
3. Specify values for the Web container service properties.
4. When finished, click **OK**.

6.6.7.1.2: Administering transports of Web container services of application servers with the Java administrative console

HTTP transports that the Web server uses to relay requests to the applicationserver are accessed through the Web container configuration. See administering transports with the Java administrative consolefor a full description of settings and tasks.

6.6.8: Administering Web modules (overview)

Classpath considerations

An important classpath-related setting to note is the Module Visibility. This application server setting impacts the portability of applications and standalone modules from other WebSphere Application Server versions and editions. If your existing module does not run as-is when you transfer it to Version 4.0, you might need to reassemble an existing module or change the module visibilitysetting.

See the information on setting classpaths for a full discussion of classpath considerations. See the applicationserver property reference for information about the module visibility setting.

Identifying a welcome page for the Web application

The default welcome page for your Web application is assumed to be named index.html. For example, if you have an application with a Web path of:

```
/webapp/myapp
```

then the default page named index.html can be implicitly accessed using the following URL:

```
http://hostname/webapp/myapp
```

To identify a different welcome page, modify the properties of theWeb module when you are assembling it. See the article aboutassembling Web modules with the ApplicationAssembly Tool (article 6.6.8.5).

Web application URLs are now case-sensitive on all operating systems

Please note that in Version 4.0.x, Webapplication URLs are now case-sensitive on all operating systems,for security and consistency.

For example, suppose you have a Web client application that runs successfully on Version 3.5.x. When running the same application on Version 4.0, you encounter an error that the welcome page (typically index.html), or HTML files to which it refers, cannot be found:

```
    Error 404: File not found: Banner.html     Error 404: File not found: HomeContent.html
```

Suppose the content of the index page is as follows:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Frameset//EN"><HTML><TITLE>Insurance Home Page</TITLE>
<frameset   rows="18,80">    <frame      src="Banner.html"           name="BannerFrame"
SCROLLING=NO>     <frame      src="HomeContent.html"      name="HomeContentFrame">
</frameset></HTML>
```

but the actual file names in \WebSphere\AppServer\installedApps\... directory in whichthe application is deployed are:

```
banner.htmlhomecontent.html
```

To correct the problem, modify the index.html file to change the names "Banner.html"and "HomeContent.html" to "banner.html" and "homecontent.html" to match the names of thefiles in the deployed application.

6.6.8.0: Web module properties

Key:

Applies to Java administrative console of Advanced Edition Version 4.0

Applies to Web administrative console of Advanced Single Server Edition Version 4.0

Applies to Application Client Resource Configuration Tool

Application or Application Ref

The application installation binding within which the module-to-server installation binding is contained. This is typically the logical name of the enterprise application you configured to contain this Web module.

Context Root

The context root of the Web application contained in an enterprise application.

The context root is combined with the defined servlet mapping (from the WAR file) to compose the full URL that users type to access the servlet. For example, if the context root is /gettingstarted and the servlet mapping is MySession, then the URL is http://host:port/gettingstarted/MySession.

Execution State

The state that you would like the Web module to be in, the next time the product is stopped and started again

Name or Module Name

An administrative name for the Web module

Server

The application server on which the Web module is installed

URI

A URI that, when resolved relative to the application URL, specifies the location of the module archive on a file system. The URI must match the URI of a ModuleRef URI in the deployment descriptor of an application if the module was packaged as part of a deployed application (EAR).

6.6.8.0.1: Assembly properties for Web components

Component name (Required, String)

Specifies the name of the servlet or JavaServer Pages(TM) (JSP)file. This name must be unique within the Web module.

Display name

Specifies a short name that is intended to be displayed by GUIs.

Description

Contains a description of the servlet or JSP file.

Component type

Specifies the type of Web component. Valid values are servlet orJSP file.

Class name (Required, String)

Specifies the full path name for the servlet's class.

JSP file (Required, String)

Specifies the full path name for the JSP file.

Load on startup

Indicates whether this servlet is to be loaded at the startup of the Webapplication. The default is false (the checkbox is notselected). Also specifies a positive integer indicating the order inwhich the servlet is to be loaded. Lower integers are loaded beforehigher integers. If no value is specified, or if the value specified isnot a positive integer, the container is free to load the servlet at any timein the startup sequence.

Small icon

Specifies a JPEG or GIF file containing a small image (16x16pixels). The image is used as an icon to represent the Web component ina GUI.

Large icon

Specifies a JPEG or GIF file containing a large image (32x32pixels). The image is used as an icon to represent the Web component ina GUI.

6.6.8.0.2: Assembly properties for initialization parameters

Initialization parameters are sent to a servlet in its HttpConfig objectwhen the servlet is first started.

Parameter name (Required, String)

Specifies the name of an initialization parameter.

Parameter value (Required, String)

Specifies the value of the initialization parameter.

Description

Contains text describing the use of the parameter.

6.6.8.0.3: Assembly properties for page lists

Page lists allow you to avoid hardcoding URLs in servlets and JSPfiles. A page list specifies the location where a request is to beforwarded, but automatically tailors that location depending on the MIME typeof the servlet. These properties allow you to specify a markup languageand an associated MIME type. For the given MIME type, you also specifya set of pages to invoke. For example, if you define a markup languagenamed VXML and associate it with a vxml MIME type, you can then define Pagenames and URIs to be invoked for that particular MIME type. The Pagenames end in .page and are the same name for all markuplanguages. However, the URIs are set to point to files that areparticular to the given MIME type. For example, if a page is calledShowAccount.page and is in a markup language named VXML, the URI isShowAccountVXML.jsp. In a markup language named HTML, the URI isShowAccountHTML.jsp. When the servlet refers toShowAccount.page, the actual file to which the request maps depends onthe servlet's MIME type.

Name

Specifies the name of the markup language--for example, HTML, WML,and VXML.

MIME Type

Specifies the MIME type of the markup language--for example,text/html and text/x-vxml.

Error Page

Specifies the name of an error page.

Default Page

Specifies the name of a default page.

Pages - Name

Specifies the name of the page to be served, for example,StockQuoteRequest.page.

Pages - URI

Specifies the URI of the page to be served, for example,examples/StockQuoteHTMLRequest.jsp.

6.6.8.0.4: Assembly properties for security constraints

Security constraints declare how Web content is to be protected.These properties associate security constraints with one or more Web resourcecollections. A constraint consists of a Web resource collection, anauthorization constraint, and a user data constraint.

- A Web resource collection is a set of resources (URL patterns) and HTTPmethods on those resources. All requests that contain a request paththat matches the URL pattern described in the Web resource collection issubject to the constraint. If no HTTP methods are specified, then thesecurity constraint applies to all HTTP methods.
- An authorization constraint is a set of roles that users must be grantedin order to access the resources described by the Web resourcecollection. If a user who requests access to a specified URI is notgranted at least one of the roles specified in the authorization constraint,the user is denied access to that resource.
- A user data constraint indicates that the transport layer of theclient/server communications process must satisfy the requirement of eitherguaranteeing content integrity (preventing tampering in transit) orguaranteeing confidentiality (preventing reading while in transit).

If multiple security constraints are specified, the container uses the"first match wins" rule when processing a request to determine whatauthentication method to use, or what authorization to allow.

Security constraint name

Specifies the name of the security constraint.

Authorization Constraints - Roles

Specifies the user roles that are permitted access to this resourcecollection.

Authorization Constraints - Description

Contains a description of the authorization constraints.

User Data Constraints - Transport guarantee

Indicates how data communicated between the client and the server is to beprotected. Specifies that the protection for communications between theclient and server is None, Integral, or Confidential. None means thatthe application does not require any transport guarantees. Integralmeans that the application requires that the data sent between the client andthe server must be sent in such a way that it cannot be changed intransit. Confidential means that the application requires that the datamust be transmitted in a way that prevents other entities from observing thecontents of the transmission. In most cases, Integral or Confidentialindicates that the use of SSL is required.

User Data Constraints - Description

Contains a description of the user data constraints.

6.6.8.0.5: Assembly properties for Web resource collections

A Web resource collection defines a set of URL patterns (resources) andHTTP methods belonging to the resource. HTTP methods handle HTTP-basedrequests, such as GET, POST, PUT, and DELETE. A URL pattern is apartial Uniform Resource Locator that acts as a template for matching thepattern with existing full URLs in an attempt to find a valid file.

Web resource name (Required, String)

Specifies the name of a Web resource collection.

Web resource description

Contains a description of the Web resource collection.

HTTP methods

Specifies the HTTP methods to which the security constraintsapplies. If no HTTP methods are specified, then the security constraintapplies to all HTTP methods. The valid values are GET, POST, PUT,DELETE, HEAD, OPTIONS, and TRACE.

URL pattern

Specifies URL patterns for resources in a Web application. Allrequests that contain a request path that matches the URL pattern are subjectto the security constraint.

6.6.8.0.8: Assembly properties for context parameters

A servlet context defines a server's view of the Web applicationwithin which the servlet is running. The context also allows a servletto access resources available to it. Using the context, a servlet canlog events, obtain URL references to resources, and set and store attributesthat other servlets in the context can use. These properties declare aWeb application's parameters for its context. They convey setupinformation, such as a webmaster's e-mail address or the name of a systemthat holds critical data.

Parameter name (Required, String)

Specifies the name of a parameter--for example,dataSourceName.

Parameter value (Required, String)

Specifies the value of a parameter--for example, jdbc/sample.

Description

Contains a description of the context parameter.

6.6.8.0.9: Assembly properties for error pages

Error page locations allow a servlet to find and serve a URI to a clientbased on a specified error status code or exception type. Theseproperties are used if the error handler is another servlet or JSPfile. The properties specify a mapping between an error code orexception type and the path of a resource in the Web application. Thecontainer examines the list in the order that it is defined, and attempts tomatch the error condition by status code or by exception class. On thefirst successful match of the error condition, the container serves back theresource defined in the Location property.

Error Code

Indicates that the error condition is a status code.

Error Code (Required, String)

Specifies an HTTP error code, for example, 404.

Exception

Indicates that the error condition is an exception type.

Exception type name (Required, String)

Specifies an exception type.

Location (Required, String)

Contains the location of the error-handling resource in the Webapplication.

6.6.8.0.10: Assembly properties for MIME mapping

A Multi-Purpose Internet Mail Extensions (MIME) mapping associates a filename extension with a type of data file (text, audio, image). Theseproperties allow you to map a MIME type to a file name extension.

Extension (Required, String)

Specifies a file name extension, for example, .txt.

MIME type (Required, String)

Specifies a defined MIME type, for example, text/plain.

6.6.8.0.11: Assembly properties for servlet mapping

A servlet mapping is a correspondence between a client request and aservlet. Servlet containers use URL paths to map client requests toservlets, and follow the URL path-mapping rules as specified in the JavaServlet specification. The container uses the URI from the request,minus the context path, as the path to map to a servlet. The containerchooses the longest matching available context path from the list of Webapplications that it hosts.

URL pattern (Required, String)

Specifies the URL pattern of the mapping. The URL pattern mustconform to the Servlet specification. The following syntax must beused:

- A string beginning with a slash character (/) and ending with the slashand asterisk characters (/*) is used as a path mapping.
- A string beginning with the characters *. is used as an extensionmapping.
- All other strings are used as exact matches only.
- A string containing only the slash character (/) indicates that theservlet specified by the mapping becomes the default servlet of theapplication. In this case, the servlet path is the request URI minusthe context path, and the path info is null.

Servlet (Required, String)

Specifies the name of the servlet associated with the URL pattern.

6.6.8.0.12: Assembly properties for tag libraries

Java ServerPages (JSP) tag libraries contain classes for common tasks suchas processing forms and accessing databases from JSP files.

Tag library file name (Required, String)

Specifies a file name relative to the location of the web.xmldocument, identifying a tag library used in the Web application.

Tag library location (Required, String)

Contains the location, as a resource relative to the root of the Webapplication, where the Tag Library Definition file for the tag library can befound.

6.6.8.0.13: Assembly properties for welcome files

A Welcome file is an entry-point file (for example, index.html) fora group of related HTML files. Welcome files are located by using agroup of suggested partial URIs. A Welcome file is an ordered list ofpartial URIs that the container uses to attempt to find a valid file when theinitial URI cannot be found. The container appends these partial URIsto the requested URI to arrive at a valid URI. For example, the usercan define a Welcome file of index.html so that a request to a URL suchas host:port/webapp/directory (where directory is a directoryentry in the WAR file that is not mapped to a servlet or JSP file) can befulfilled.

Welcome file (Required, String)

> The Welcome file list is an ordered list of partial URLs with no trailingor leading slash characters (/). The Web server appends each file inthe order specified and checks whether a resource in the WAR file is mapped tothat request URI. The container forwards the request to the firstresource in the WAR that matches.

6.6.8.0.14: Assembly properties for MIME filters

Filters transform either the content of an HTTP request or response and canalso modify header information. MIME filters forward HTTP responseswith a specified MIME type to one or more designated servlets for furtherprocessing.

MIME Filter - Target

Specifies the target virtual host for the servlets.

MIME Filter - Type

Specifies the MIME type of the response that is to be forwarded.

6.6.8.0.15: Assembly properties for JSP attributes

JSP attributes are used by the servlet that implements JSP processingbehavior.

JSP Attribute (Name)

Specifies the name of an attribute.

JSP Attribute (Value)

Specifies the value of an attribute.

6.6.8.0.16: Assembly properties for file-serving attributes

File serving allows a Web application to serve static file types, such asHTML. File-serving attributes are used by the servlet that implementsfile-serving behavior.

File Serving Attribute (Name)

Specifies the name of an attribute.

File Serving Attribute (Value)

Specifies the value of an attribute.

6.6.8.0.17: Assembly properties for invoker attributes

Invoker attributes are used by the servlet that implements the invocationbehavior.

Invoker Attribute (Name)

Specifies the name of an attribute.

Invoker Attribute (Value)

Specifies the value of an attribute.

6.6.8.0.18: Assembly properties for servlet caching configurations

Dynamic caching can be used to improve the performance of servlet andJavaServer Pages (JSP) files by serving requests from an in-memorycache. Cache entries contain the servlet's output, results of theservlet's execution, and metadata.

The properties on the General tab define a cache group and govern how longan entry remains in the cache. The properties on the ID Generation tabdefine how cache IDs are built and the criteria used to cache or invalidateentries. The properties on the Advanced tab define external cachegroups and specify custom interfaces for handling servlet caching.

Caching group name (Required, String)

Specifies a name for the group of servlets or JSP files to becached.

Priority

An integer that defines the default priority for servlets that arecached. The default value is 1. Priority is an extension of theLeast Recently Used (LRU) caching algorithm. It represents the numberof cycles through the LRU algorithm that an entry is guaranteed to stay in thecache. The priority represents the length of time that an entry remainsin the cache before being eligible for removal. On each cycle of thealgorithm, the priority of an entry is decremented. When the priorityreaches zero, the entry is eligible for invalidation. If an entry isrequested while in the cache, its priority is reset to the priorityvalue. Regardless of the priority value and the number of requests, anentry is invalidated when its timeout occurs. Consider increasing thepriority of a servlet or JSP file when it is difficult to calculate the outputof the servlet or JSP file or when the servlet or JSP file is executed moreoften than average. Priority values should be low. Higher valuesdo not yield much improvement but use extra LRU cycles. Use timeout toguarantee the validity of an entry. Use priority to rank the relativeimportance of one entry to other entries. Giving all entries equalpriority results in a standard LRU cache that increases performancesignificantly.

Timeout

Specifies the length of time, in seconds, that a cache entry is to remainin the cache after it has been created. When this time elapses, theentry is removed from the cache. If the timeout is zero or a negativenumber, the entry does not time out. It is removed when the cache isfull or programmatically, from within an application.

Invalidate only

Specifies that invalidations for a servlet are to take place, but that nocaching is to be performed for the servlet. For example, this propertycan be used to prevent caching of control servlets. Control servletstreat HTTP requests as commands and execute those commands. By default,this checkbox is not selected.

Caching group members

Specifies the names of the servlets or JSP files to be cached. TheURIs are determined from the servlet mappings.

Use URIs for cache ID building

Specifies whether or not the URI of the requested servlet is to be used tocreate a cache ID. By default, URIs are used.

Use specified string

Specifies a string representing a combination of request and sessionvariables that are to be used to create cache IDs. (This propertydefines request and session variables, and the cache uses the values of thesevariables to create IDs for the entries.)

Variables - ID

The name of a request parameter, request attribute, session parameter, orcookie.

Variables - Type

Indicates the type of variable specified in the ID field. The validvalues are Request parameter, Request attribute, Session parameter, orCookie.

Variables - Method

The name of a method in the request attribute or session parameter.The output of this method is used to generate cache entry IDs. If thisvalue is not specified, the toString method is used by default.

Variables - Data ID

Specifies a string that, combined with the value of the variable,generates a group name for the cache entry. The cache entry is placedin this group. This group can later be invalidated.

Variables - Invalidate ID

Specifies a string that is combined with the value of the variable on therequest or session to form a group name. The cache invalidates thegroup name.

Required

Indicates whether a value must be present in the request. If thischeckbox is selected, and either the request parameter, request attribute, orsession parameter is not specified, or the method is not specified, therequest is not cached.

External cache groups - Group name

Specifies the name of the external cache group to which this servlet willbe published.

ID generator

Specifies a user-written interface for handling parameters, attributes,and sessions. The value must be a full package and class name of aclass extendingcom.ibm.websphere.servlet.cache.IdGenerator.The properties specified in the Application Assembly Tool will still be usedand passed to the IdGenerator in the initialize method inside acom.ibm.websphere.servlet.cache.CacheConfigobject.

Meta data generator

Specifies a user-written interface for handling invalidation, prioritylevels, and external cache groups. The value must be the full packageand class name of a class extendingcom.ibm.websphere.servlet.cache.MetaDataGenerator.The properties specified in the Application Assembly Tool will still be usedand passed to the MetaDataGenerator in the initialize method inside acom.ibm.websphere.servlet.cache.CacheConfigobject.

6.6.8.0.aa: Assembly properties for Web modules

File name (Required, String)

Specifies the file name of the Web module, relative to the top level ofthe application package.

Alternative DD

Specifies the file name for an alternative deployment descriptor file touse instead of the original deployment descriptor file in the module'sJAR file. This file is the postassembly version of the deploymentdescriptor file. (The original deployment descriptor file can be editedto resolve dependencies and security information. Directing the use ofthe alternative deployment descriptor allows you to keep the originaldeployment descriptor file intact). The value of the Alternative DDproperty must be the full path name of the deployment descriptor file relativeto the module's root directory. By convention, the file is in theALT-INF directory. If this property is not specified, the deploymentdescriptor file is read directly from the module's JAR file.

Context root (Required, String)

Specifies the context root of the Web application. The context rootis combined with the defined servlet mapping (from the WAR file) to composethe full URL that users type to access the servlet. For example, if thecontext root is /gettingstarted and the servlet mapping is MySession, then theURL is http://host:port/gettingstarted/MySession.

Classpath

Specifies the full class path for the Web application. Specify thevalues relative to the root of the EAR file and separate the values withspaces. Absolute values that reference files or directories on the harddrive are ignored. To specify classes that are not in JAR files but arein the root of the EAR file, use a period and forward slash(./). Consider the following example directory structure inwhich the file myapp.ear contains a Web module namedmywebapp.war. Classes reside in class1.jar andclass2.zip. A class named xyz.class is not packaged in aJAR file but is in the root of the EAR file.

```
myapp.ear/mywebapp.warmyapp.ear/class1.jarmyapp.ear/class2.zipmyapp.ear/xyz.class
```

Specify `class1.jar class2.zip ./` as thevalue of the Classpath property. (Name only the directory for.class files.)

Display name

Specifies a short name that is intended to be displayed by GUIs.

Description

Contains a description of the Web module.

Distributable

Specifies that this Web application is programmed appropriately to bedeployed into a distributed servlet container.

Small icon

Specifies a JPEG or GIF file containing a small image (16x16pixels). The image is used as an icon to represent the module in aGUI.

Large icon

Specifies a JPEG or GIF file containing a large image (32x32pixels). The image is used as an icon to represent the module in aGUI.

Session configuration

Indicates that session configuration information is present.Checking this box makes the Session timeout property editable.

Session timeout

Specifies a time period, in seconds, after which a client is consideredinactive. The default value is zero, indicating that the sessiontimeout never expires.

Login configuration -- Authentication method

Specifies an authentication method to use. As a prerequisite togaining access to any Web resources protected by an authorization constraint,a user must authenticate by using the configured mechanism. A Webapplication can authenticate a user to a Web server by using one of thefollowing mechanisms: HTTP basic authentication, HTTP digestauthentication, HTTPS client authentication, and form-basedauthentication.

- HTTP basic authentication is not a secure protocol because the userpassword is transmitted with a simple Base64 encoding and the target server isnot authenticated. In basic authentication, the Web server requests aWeb client to authenticate the user and passes a string called the realm ofthe request in which the user is to be authenticated.

- HTTP digest authentication transmits the password in encryptedform.
- HTTPS client authentication uses HTTPS (HTTP over SSL) and requires theuser to possess a public key certificate.
- Form-based authentication allows the developer to control the appearanceof login screens.

The Login configuration properties are used to configure the authenticationmethod that should be used, the realm name that should be used for HTTP basicauthentication, and the attributes that are needed by the form-based loginmechanism. Valid values for this property are Unspecified, Basic,Digest, Form, and Client certification.

Note: HTTP digest authentication is not supported as a loginconfiguration in this product. Also, not all login configurations aresupported in all of the product's global security authenticationmechanisms (Local Operating system, LTPA, and custom pluggable userregistry). HTTP basic authentication and form-based loginauthentication are the only authentication methods supported by the LocalOperating system user registry. Because Advanced Single Server Editionuses the local operating system as the user registry for authentication, itcan only support these two login methods. LTPA and the custom pluggableuser registry are capable of supporting HTTP basic authentication, form-basedlogin, and HTTPS client authentication. LTPA and the custom pluggableuser registry is available only in Advanced Edition.

Login configuration -- Realm name

Specifies the realm name to use in HTTP basic authorization. It isbased on a user name and password, sent as a string (with a simple Base64encoding). An HTTP realm is a string that allows URIs to be groupedtogether. For example, if a user accesses a secured resource on a Webserver within the "finance realm," subsequent access to the same or differentresource within the same realm does not result in a repeat prompt for a userID and password.

Login configuration -- Login page (Required, String)

Specifies the location of the login form. If form-basedauthentication is not used, this property is disabled.

Form Login Config -- Error page (Required, String)

Specifies the location of the error page. If form-basedauthentication is not used, this property is disabled.

Reload interval

Specifies a time interval, in seconds, in which the Web application'sfile system is scanned for updated files. The default is 0(zero).

Reloading enabled

Specifies whether file reloading is enabled. The default isfalse.

Default error page

Specifies a file name for the default error page. If no other errorpage is specified in the application, this error page is used.

Additional classpath

Specifies an additional class path that will be used to referenceresources outside of those specified in the archive. Specify the valuesrelative to the root of the EAR file and separate the values withspaces. Absolute values that reference files or directories on the harddrive are ignored. To specify classes that are not in JAR files but arein the root of the EAR file, use a period and forward slash(./). Consider the following example directory structure inwhich the file myapp.ear contains a Web module namedmywebapp.war. Additional classes reside in class1.jar andclass2.zip. A class named xyz.class is not packaged in aJAR file but is in the root of the EAR file.

```
myapp.ear/mywebapp.warmyapp.ear/class1.jarmyapp.ear/class2.zipmyapp.ear/xyz.class
```

Specify `class1.jar class2.zip ./` as thevalue of the Additional classpath property. (Name only the directoryfor .class files.)

File serving enabled

Specifies whether file serving is enabled. File serving allows theapplication to serve static file types, such as HTML and GIF. Fileserving can be disabled if, for example, the application contains only dynamiccomponents. The default value is true.

Directory browsing enabled

Specifies whether directory browsing is enabled. Directory browsingallows the application to browse disk directories. Directory browsingcan be disabled if, for example, you want to protect data. The defaultvalue is true.

Serve servlets by classname

Specifies whether a servlet can be served by requesting its classname. Usually, servlets are served only through a URI reference.The class name is the actual name of the servlet on disk. For example,a file named SnoopServlet.java compiles intoSnoopServlet.class. (This is the class name.)SnoopServlet.class is normally invoked by specifying snoop in theURI. However, if Serve Servlets by Classname is enabled, the servlet isinvoked by specifying SnoopServlet. The default value is true.

Virtual hostname

Specifies a virtual host name. A virtual host is a configurationenabling a single host machine to resemble multiple host machines. Thisproperty allows you to bind the application to a virtual host in order toenable execution on that virtual host.

6.6.8.1: Administering Web modules with the Java administrative console

Work with resources of this type by locating them in the tree view.

To locate modules according to their membership in enterprise applications, click:

WebSphere Administrative Domain -> Nodes -> ***node_name*** **-> Enterprise Applications ->** ***application_name*** **->Web modules**

To locate modules installed on a particular application server, click:

WebSphere Administrative Domain -> Nodes -> ***node_name*** **-> Application Servers ->** ***application_server_name*** **->Installed Web modules**

In either case, the instances of the module will be displayed in thedetails view.

6.6.8.1.1: Installing Web modules with the Java administrative console

Use the application installation wizard to install Web modules onto an application server.

6.6.8.1.2: Viewing deployment descriptors of Web modules with the Java administrative console

To view the deployment descriptor of a Web module:

1. Locate the Web module instance.
2. Right-click your Web module in the details view and select **View Deployment Descriptor** from the popup menu.

A Web Module window opens, providing descriptive information about the selected module's Web.xml, servlets, servlet mappings, welcome file, security constraints and role, login configuration, and EJB reference.

6.6.8.1.3: Showing the status of Web modules with the Java administrative console

During this task, you will view the status (such as running or stopped) of each Web module in an enterprise application.

1. Locate the application instance.
2. Click the Installed EJB modules folder of the application to display its Web module instances in the details view.
3. Right-click an instance and select **Show Status** to display the Module Status window.
4. When finished viewing status, close the window.

6.6.8.1.5: Moving Web modules to other application servers with the Java administrative console

You can move a Web module from one application server to a different application server or server group. To move a module:

1. Locate the Web module instance.
2. Right-click your Web module in the details view and select **Move** from the popup menu.
3. In the Select a target server or ServerGroup dialog that opens, specify the target application server or server group and click **OK**
4. Copy WAR files containing the module to the node that contains the destination server. If the destination is a server group, then copy the WAR file to all nodes that contain the clones of the destination server group.

6.6.8.5: Administering Web modules with Application Assembly Tool

A Web module is used to assemble one or more servlets, JavaServer Pages(JSP) files, Web pages, and other static content into a single deployableunit. The Application Assembly Tool is used to create and edit modules,verify the archive files, and generate deployment code. See the relatedtopics for links to concepts, instructions for creating a Web module, andfield help.

6.6.8.5.1: Creating a Web module

Web modules can be created by using property dialog box or by using awizard.

- Using the property dialog boxes
- Using the Create Web Module wizard

Using the property dialog boxes

The steps for creating a Web module are as follows:

1. Click **File**->**New**->**Web Module**. Thenavigation pane displays a hierarchical structure used to build the contentsof the module. The icons represent the components, assembly properties,and files for the module. A property dialog box containing generalinformation about the module is displayed in the property pane.
2. By default, the archive file name and the module display name are thesame. It is recommended that you change the display name in theproperty pane. Enter values for other properties as needed. Viewthe help for 6.6.8.0.a: Assembly properties for Web modules.
3. By default, the temporary location of the Web module isinstallation_directory`/bin`. You must specify a newfile name and location by clicking **File**->**Save**.You must first add at least one Web component (servlet or JSP file) beforesaving the archive.
4. Add Web components (servlets or JSP files) to the module. You mustadd at least one Web component. There are several ways of addingcomponents to a module:
    - Import an existing WAR file containing Web components. In thenavigation pane, right-click the Web Components icon and choose**Import**. Click **Browse** to browse the file systemand locate the desired archive file. When the file is located, click**Open**. The Web applications in the selected archive file aredisplayed. Select a Web application. Its Web components aredisplayed in the right window. Select the servlets or JSP files to beadded and click **Add**. The components are displayed in theSelected Components window. Click **OK**. The propertiesassociated with the archive are also imported and the property dialog boxesare automatically populated with values. Double-click the WebComponents icon to verify that the servlets or JSP files are included in themodule.
    - Use a copy-and-paste operation to copy archive files from an existingmodule.
    - Create a new Web component. Right-click the Web Components icon andchoose **New**. Enter a component name and choose a componenttype. Browse for and select the class files. By default, theroot directory or archive is the current archive. If needed, browse thefile system for the directory or archive where the class files reside.After you choose a directory or archive, its file structure isdisplayed. Expand the structure and locate the files that youneed. Select the file and click **OK**. In the New WebComponent property dialog box, click **OK**. Verify that the Webcomponent has been added to the module (double-click the Web Components iconin the navigation pane). The Web components are also listed in the topportion of the property pane. Click the component to view itscorresponding property dialog box in the bottom portion of the pane.
5. Enter properties for the Web component as needed. View the help for6.6.8.0.1: Assembly properties for Web components.
6. Enter assembly properties for each Web component. Click the plussign (+) next to the component instance to reveal propertygroups. Right-click each property group's icon. Choose**New** to add new values, or edit existing values in the propertypane.
    - Specify Security Role References. View the help for 6.6.43.0.3: Assembly properties for security

role references.

    - Specify Initialization Parameters. View the help for 6.6.8.0.2: Assembly properties for initialization parameters.
    - Specify Page List Extensions. View the help for 6.6.8.0.3: Assembly properties for page lists.

7. Specify additional properties for the Web module. Right-click eachproperty group's icon. Choose **New** to add new values, oredit existing values in the property pane.
    - Specify Security Constraints. View the help for 6.6.8.0.4: Assembly properties for security constraints. If you add a security constraint, you must add atleast one Web resource collection.
    - Specify Web resource collections, HTTP methods, and URL patterns.View the help for 6.6.8.0.5: Assembly properties for Web resource collections.
    - Specify Context Parameters. View the help for 6.6.8.0.8: Assembly properties for context parameters.
    - Specify EJB references. View the help for 6.6.43.0.1: Assembly properties for EJB references.
    - Specify Environment Entries. View the help for 6.6.34.0.a: Assembly properties for environment entries.
    - Specify Error Pages. View the help for 6.6.8.0.9: Assembly properties for error pages.
    - Specify MIME Mappings. View the help for 6.6.8.0.10: Assembly properties for MIME mapping.
    - Specify Resource References. View the help for 6.6.43.0.2 Assembly properties for resource references.
    - Specify Security Roles. View the help for 6.6.5.0.5: Assembly properties for security roles.
    - Specify Servlet Mapping. View the help for 6.6.8.0.11: Assembly properties for servlet mapping.
    - Specify Tag Libraries. View the help for 6.6.8.0.12: Assembly properties for tag libraries.
    - Specify Welcome Files. View the help for 6.6.8.0.13: Assembly properties for welcome files.
8. Optionally, specify assembly property extensions. In the navigationpane, double-click the icon for Assembly Property Extensions.
    - Specify MIME filters. View the help for 6.6.8.0.14: Assembly properties for MIME filters.
    - Specify JSP Attributes. View the help for 6.6.8.0.15: Assembly properties for JSP attributes.
    - Specify File Serving Attributes. View the help for 6.6.8.0.16: Assembly properties for file-serving attributes.
    - Specify Invoker Attributes. View the help for 6.6.8.0.17: Assembly properties for invoker attributes.
    - Specify Servlet Caching Configurations. View the help for 6.6.8.0.18: Assembly properties for servlet caching configurations.
9. Add any other files needed by the application. In the navigationpane, click the plus sign (+) next to the Files icon.Right-click **Add Class Files**, **Add JAR Files**, or **AddResource Files**. Choose **Add Files**. Click**Browse** to navigate to the desired directory or archive and thenclick **Select**. If you are adding an entire archive, selectthe directory that contains the archive. The directory structure isdisplayed in the left pane. Browse the directory structure. Fromthe right pane, select one or more files to be added and click**Add**. If you select a directory and click **Add**, allfiles in the directory, including the directory, are added. Relativepath names are maintained. When the Selected Files window contains thecorrect set of files, click **OK**.

10. Click **File**->**Save** to save the archive.

Using the Create Web Module wizard

Use this wizard to create a Web module. The module can then be usedas a stand-alone application, or it can become part of a J2EE applicationcontaining other modules. A Web module consists of one or more servletsand JSP files. You can use existing archives (by importing them), orcreate new ones.

During creation of the Web module, you specify the files for each servletor JSP file to be included in the module. You also specify assemblyproperties for the servlets and JSP files, such as references to enterprisebeans and resource connection factories, and security roles. Thecontent information and assembly properties are used to create a deploymentdescriptor.

Before you start the wizard, you must have the required files for yourservlet or JSP file. When the wizard is completed, your Web module (WARfile) is created in the directory that you specify.

To create a Web module, click the **Wizards** icon on the tool barand then click **Web Module**. Follow the instructions on eachpanel.

- Specifying Web module properties
- Adding files
- Specifying optional Web module properties
- Choosing Web Module icons
- Adding Web components
- Adding security roles
- Adding servlet mappings
- Adding resource references
- Adding context parameters
- Adding error pages
- Adding MIME mappings
- Adding Tag Libraries
- Adding Welcome Files
- Adding EJB references
- Setting additional properties and saving the archive

Specifying Web module properties

On the **Specifying Web Module Properties** panel:

1. Indicate the application to which this module is to be added. If aparent application is not indicated, the module is created as a stand-aloneapplication.
2. Specify a file name and display name for the module. The displayname is used to identify your module in the Application Assembly Tool and canbe used by other tools. The file name specifies a location on yoursystem where the WAR file is to be created.
3. Provide a short description of the module.
4. Click **Next**.

Adding files

On the **Adding Files** panel, specify the files that are to beassembled for your Web module.

1. Click **Add Resource Files**, **Add Class Files**, or**Add JAR files**, depending on the type of file you are adding.First, browse for the root directory or archive where the files are locatedand click **Select**. If you are adding an entire archive,select the directory that contains the archive. The directory structureis displayed in the left pane. Browse the directory structure.From the right pane, select one or more files to be added and click**Add**. If you select a directory and click **Add**, allfiles in the directory, including the directory, are added. Relativepath names are maintained. The selected files are displayed in theSelected Files window. Click **OK**. The files are listedin a table on the wizard panel.
2. If you want to remove a file, select the file in the table and then click**Remove**.
3. Continue to add or remove files until you have the correct set offiles.
4. Click **Next**.

Specifying optional Web module properties

On the **Specifying Optional Web Module Properties** panel:

1. Indicate whether the module can be installed in a distributable Webcontainer. The default value is false.
2. Specify the full classpath for the Web application.
3. Click **Next**.

Choosing Web Module icons

On the **Choosing Web Module icons** panel, specify icons for yourmodule.

1. Specify the full path name of a GIF or JPEG file. The icon must be16x16 pixels in size.
2. Specify a full path name of a GIF or JPEG file. The icon must be32x32 pixels in size.
3. Click **Next**.

Adding Web components

On the **Adding Web components** panel, add new servlets or JSPfiles or import existing ones.

To add a new Web component:

1. Click **New**.
2. On the **Specifying Web Component Properties** panel, specify thecomponent name and enter values for other properties. View the help for6.6.8.0.1: Assembly properties for Web components.
3. On the **Specifying Web Component Type** panel, indicate the typeof Web component and specify the servlet class name or JSP file.
4. On the **Choosing Web Component Icons** panel, specify a filecontaining a JPEG or GIF image.
5. On the **Adding Security Role References** panel, enter values forsecurity role references. Click **Add** to enter a rolename. Click **OK**. The role name is displayed in thetable on the wizard panel. To remove a role, select the role in thetable and then click **Remove**. Repeat as necessary.View the help for 6.6.43.0.3: Assembly properties for security role references. Click **Next**.
6. On the **Adding Initialization Parameters** panel, enter values forthe Web component's initialization parameters. Click**Add** to add a parameter. You must enter a name andvalue. Click **OK**. The parameter is displayed in atable on the wizard panel. To remove a parameter, select the parameterand click **Remove**. Repeat as necessary. View the helpfor 6.6.8.0.2: Assembly properties for initialization parameters.

7. Click **Finish**.

To import an existing Web component:

1. Click **Import**.
2. Browse the file system to locate the desired archive. The contentsof the archive are displayed in a window. Select the desired componentand then click **Add**. The components are added to the SelectedComponents window. Click **OK**.

To remove a Web component, select the component name in the table and click**Remove**.

When you are finished adding Web components, click **Next**.

Adding security roles

On the **Adding Security Roles** panel:

1. Click **Add**. Type a role name and, optionally, type adescription. Click **OK**. The role name is displayed ina table on the wizard panel. View the help for 6.6.5.0.5: Assembly properties for security roles.
2. Continue to add security roles as needed. If you need to remove arole, select the role in the table and then click **Remove**.
3. Click **Next**.

Adding servlet mappings

On the **Adding Servlet Mappings** panel:

1. Click **Add**. Enter a URL pattern and select a servlet fromthe menu. View the help for 6.6.8.0.11: Assembly properties for servlet mapping. Click **OK**. The servlet mappings aredisplayed in a table on the wizard panel.
2. Continue to add and remove URL patterns and corresponding servlets asneeded. If you need to remove mapping, select the entry in the tableand then click **Remove**.
3. Click **Next**.

Adding resource references

On the **Adding Resource References** panel, enter references forresource connection factories.

1. Click **Add** to add a reference. You must enter a value fora name, type, and authorization mode. View the help for 6.6.43.0.2 Assembly properties for resource references. Click **OK**. The reference isdisplayed in the table on the wizard panel.
2. To remove a reference, select the reference in the table and then click**Remove**.
3. Continue to add and remove references as needed.
4. Click **Next**.

Adding context parameters

On the **Adding Context Parameters** panel, enter values for contextparameters.

1. Click **Add** to add a parameter. You must enter a name andvalue. View the help for 6.6.8.0.8: Assembly properties for context parameters. Click **OK**. The parameter isdisplayed in the table on the wizard panel.
2. To remove a parameter, select the parameter and then click**Remove**.

3. Continue to add and remove parameters as needed.
4. Click **Next**.

Adding error pages

On the **Adding Error Pages** panel, enter values for errorpages.

1. Click **Add** to add a page. You must enter alocation. Then choose **Error Code** or **ErrorException**. Enter a name for the error code or exception.View the help for 6.6.8.0.9: Assembly properties for error pages. Click **OK**. The error page isdisplayed in the table on the wizard panel.
2. To remove an error page, select the item in the table and then click**Remove**.
3. Continue to add and remove error pages as needed.
4. Click **Next**.

Adding MIME mappings

On the **Adding MIME Mappings** panel, enter values for MIMEmappings.

1. Click **Add** to add a mapping. You must enter an extensionand a MIME type. View the help for 6.6.8.0.10: Assembly properties for MIME mapping. Click **OK**. The mapping is displayedin the table on the wizard panel.
2. To remove a mapping, select the mapping and then click**Remove**.
3. Continue to add and remove mappings as needed.
4. Click **Next**.

Adding Tag Libraries

On the **Adding Tag Libraries** panel, enter values for taglibraries.

1. Click **Add** to add a tag library. You must enter a tagfile name and library location. View the help for 6.6.8.0.12: Assembly properties for tag libraries. Click **OK**. The tag library isdisplayed in the table on the wizard panel.
2. To remove a tag library, select the library and then click**Remove**.
3. Continue to add and remove tag libraries as needed.
4. Click **Next**.

Adding Welcome Files

On the **Adding Welcome Files** panel, enter values for welcomefiles.

1. Click **Add**. Enter a file name or use the file browser tolocate the file. View the help for 6.6.8.0.13: Assembly properties for welcome files. Click **OK**. The file name isdisplayed in the table on the wizard panel.
2. To remove a file, select the file in the table and then click**Remove**.
3. Continue to add and remove files as needed.
4. Click **Next**.

Adding EJB references

On the **Adding EJB References** panel, enter values for EJBreferences.

1. Click **Add** to add a reference. You must enter a value forthe name, home interface, remote interface, and type. View the help for6.6.43.0.1: Assembly properties for EJB references. Click **OK**. The reference isdisplayed in the table on the wizard panel.
2. To remove a reference, select the entry in the table and then click**Remove**.
3. Continue to add and remove references as needed.

Setting additional properties and saving the archive

Click **Finish** to complete the wizard. To change settingsfor properties, click **Back** to return to the appropriatepanel. Make any needed changes, and then click**Finish**.

After you click **Finish**, the contents of the archive aredisplayed in the Application Assembly Tool window. In the navigationpane, continue adding or modifying properties as needed. For example,you can add binding information. When you are finished editing thearchive, click **File**->**Save** to save the archivefile.

6.6.11: Administering HTTP session support (overview)

When configuring the Session Manager, the WebSphere administrator can:

- Specify which session tracking mechanism to use to pass the sessionid between the browser and the servlet
- Specify whether to save session data in a database (persistent sessions)
- Define the persistent sessions characteristics
- Define other characteristics of the Session Manager environment

6.6.11.0: Session Manager properties

Key:

☕ Applies to Java administrative console of Advanced Edition Version 4.0

🌐 Applies to Web administrative console of Advanced Single Server Edition Version 4.0

🖥 Applies to Application Client Resource Configuration Tool

Application Server ☕

The application server with which the Session Manager is associated

Allow Overflow ☕ 🌐

Whether to allow the number of sessions in memory to exceed the value specified by Max In Memory Session Count property

Cookie Domain ☕ **or Domain** 🌐

The value of the domain field of a session tracking cookie. This value will dictate to the browser whether or not to send a cookie to particular servers. For example, if you specify a particular domain, session cookies will be sent only to hosts in that domain. The default domain is the server.

Cookie Maximum Age ☕ **or Maximum Age** 🌐

The amount of time that the cookie will live on the client browser.

Specify that the cookie will live only as long as the current browser session, or to a maximum age. If youchoose the maximum age option, specify the age in seconds.

This value corresponds tothe Time to Live (TTL) value described in the Cookie specification.

For *Advanced Single Server Edition*, use -1 to specify the current browser session.

Cookie Name ☕ **or Name** 🌐

A unique name for the cookie. The name must be JSESSIONID as specified in the Servlet 2.2 API

Confirm Password ☕

Enter the password a second time to ensure it recorded correctly

Cookie Path ☕ **or Path** 🌐

This dictates browser whether cookie is sent to the URI requested based on the path. Specify any string representing a path on the server. "/" indicates root directory.

Specify a value in order to restrict the paths to which the cookie will be sent. By restricting paths, you can keep the cookie from being sent to certain URLs on the server. If you specify the root directory, the cookie will be sent no matter which path on the given server is accessed.

Data Source JNDI Name 🌐 **or Data Source** ☕

The JNDI name of the non-JTA enabled data source from which the Session Manager will obtain database connections.

For example, if the JNDI name of the datasource is "jdbc/sessions", specify "jdbc/sessions."

The data source represents a pool of database connections and a configuration for that pool (such as the pool size). The data source must already exist as a configured resource in the environment.

Data Source

See Data Source JNDI Name

DB2 Row Size

The tablespace page size configured for the sessions table, if using a DB2 database. Possible values are 4, 8, 16, and 32 kilobytes (K).

The default row size is 4K. In DB2, it can be updated to a larger value. This can help databaseperformance in some environments.

When this value is other than 4, you must specify Table Space Name to use. For 4K pages, the Table Space Name is optional.

Domain

See Cookie Domain

Enable Cookies

Whether session tracking will use cookies to carry session IDs. If cookies are enabled, session tracking will recognize session IDs that arrive as cookies and try to use cookies for sending session IDs. If cookies are not enabled, session tracking will use URL rewriting instead of cookies (if URL rewriting is enabled). Enabling cookies takes precedence over URL rewriting.

⚠ For *Advanced Single Server Edition*, do not disable cookies in the Session Manager of the application server that is running the administrative application, because it will cause the administrative application not to function after a restart of the server. An alternative is to run the administrative application in a separate process from your applications.

Enable Persistent Sessions

When persistent sessions are enabled, the Session Manager will persist session information into the data source specified by the data source connection settings. Otherwise, the Session Manager will discard the session data when the server shuts down.

Enable Protocol Switch Rewriting

Whether the session ID is added to a URL when the URL requires a switch from HTTP to HTTPS or from HTTPS to HTTP. If rewriting is enabled, the session ID is required in order to go between HTTP and HTTPS.

Enable Security Integration **or Integrate with WebSphere Security**

When security integration is enabled, the Session Manager will associate the identity of users with their HTTP sessions.

⚠ Do not enable this property if the application server contains a Web application that has Form Login configured as the authentication method and local operating system is the authentication mechanism. It will cause authorization failures when user agents try to use the Web application.

Enable SSL Tracking or Enable SSL ID Tracking

Whether session tracking uses SSL to carry session IDs. Enabling SSL tracking takes precedence over cookie-based session tracking and URL rewriting.

Enable SSL ID Tracking

See Enable SSL Tracking

Enable URL Rewriting

Specifies whether the Session Manager uses rewritten URLs to carry the session IDs. If URL rewriting is enabled, the Session Manager recognizes session IDs that arrive in the URL ifthe encodeURL method is called in the servlet

First Hour or First Time of Day (0 - 23)

The first hour during which the invalidated persistent sessions will be cleared from the database. This value must be a positive integer between 0 and 23. This is valid only when schedule invalidation is enabled.

First Time of Day (0 - 23)

See First Hour

Integrate with WebSphere Security

See Enable Security Integration

Invalidation Timeout

Specifies how long a session is allowed to go unused before it will be considered valid no longer.

For Advanced Edition: Specify either "Set timeout" or "No timeout." If you select to set the timeout, the value must be atleast two minutes, specified in minutes.

For Advanced Single Server Edition: Use a -1 to specify that the session will not be invalidated.

The value of this setting is used as a default when the session timeout is not specified in a Web module deployment descriptor.

Note, to preserve performance, the invalidation timer is not accurate "to the second." It is safe to assume that the timer is accurate to within two minutes. When the Write Frequency is time based, this value should be at least twice as large as the write interval.

Maximum Age

See Cookie Maximum Age

Max In Memory Session Count or Maximum In Memory Session Count

Specifies the maximum number of sessions to maintain in memory.

The meaning differs depending on whether you are using in-memory or persistent sessions. For in-memory sessions, this value specifies the number of sessions in the base session table. Use the Allow Overflow property to specify whether to limit sessions to this number for the entire Session Manager, or to allow additional sessions to be stored in secondary tables.

For persistent sessions, this value specifies the size of the memory cache for sessions. When the session cache has reached its maximum size and a new session is requested, Session Manager removes the least recently used session from the cache to make room for the new one.

Name

See Cookie Name

Node

The administrative node with which the Session Manager is associated

Password

The password for database access

Path

See Cookie Path

Restrict exchange of cookies to secure sessions or Secure

Whether session cookies include the secure field. Enabling the feature will restrict the exchange of cookies only to HTTPS sessions.

Schedule Invalidation or Specify session database cleanup schedule for invalidated sessions

Enables the scheduled invalidation process for cleaning up the invalidated HttpSessions from the sessions database. Enable this option to reduce the number of database updates that are requiredto keep the HttpSessions alive.

When this option is not enabled, the invalidator process runs everyfew minutes to remove invalidated HttpSessions.

When this option is enabled, this setting specifies the two hours of a day in which the invalidator process cleans up the invalidated persistent sessions in the database. Specify the times at which there will be less activity in the session database.

Second Hour or Second Time of Day (0 - 23)

The second hour during which the invalidated persistent sessions will be cleared from the database. This value must be a positive integer between 0 and 23.This is valid only when schedule invalidation is enabled.

Secure

See Restrict exchange of cookies to secure sessions

⚠ For *Advanced Single Server Edition*, do not enable this feature in the Session Manager of the application server that is running the administrative application, because it will cause the administrative application not to function after a restart of the server. An alternative is to run the administrative application in a separate process from your applications.

Specify session database cleanup schedule for invalidated sessions

See Schedule Invalidation

Table Space Name

Tablespace to be used for the sessions table. This value is required when the DB2 Page Size is other than 4K and session persistence is enabled.

User ID **or Username**

The user ID for database access

Use Multirow Sessions **or Using Multirow Schema**

Whether to place each instance of application data in a separate row in the database, allowing larger amounts of data to be stored for each session. This can yield better performance in certain usage scenarios. If using multirow schema is not enabled, instances of application data can be placed in the same row.

Username

See User ID

Using Multirow Schema

See Use Multirow Sessions

Write Contents

Whether only updated attributes should be written to the database. Otherwise, all of the session attributes will be written to the database, whether or not they have changed.

- If you specify only updated attributes, only the updated attributes will be written to the database
- If you specify all session attributes, all attributes will be written to the database

Write Frequency

When the session will be written to the database.

- If you specify end of servlet service (END_OF_SERVLET_SERVICE), a session will be written to the database after the servlet completes execution.
- If you specify manual update (MANUAL_UPDATE), programmatic sync on the IBMSession object will be required in order to write the session data to the database.
- If you specify time based (TIME_BASED_WRITE), session data will be written to the database based on the specified Write Interval value.

Write Interval

If the Write Frequency is specified as TIME_BASED_WRITE, this value specifies how often the session data will be written to the database. The value must be a number of seconds, specified as a positive integer in the range from 5 to 9999. At minimum, a configured Invalidation Timeout should be twice as large as the configured write interval.

See also the other application server properties.

6.6.11.1: Administering session support with the Java administrative console

Use the Java administrative console to administer Session Managers, which providesupport for HTTP sessions.

Work with resources of this type by locating them in the application server properties:

1. Locate the application server instance containing the service.
2. When the application server properties are displayed in the properties view, click the **Services** tabbed page.
3. Locate the Session Manager service in the list of services.

Changes to Session Manager properties take effect after you stop the serverand start it again.

6.6.11.1.1: Configuring new Session managers

Session Managers cannot be created directly. One Session Manager service is created for you when you create an application server.

6.6.11.5: Procedure for configuring persistent session support

To configure Session Manager for persistent sessions:

1. Create a JDBC Provider.
2. Create a data source pointing to existing database, using the JDBC providerthat you created. Note the JNDI name of the data source.
3. Configure the Session Manager and save your changes:
    1. Enable session persistence.
    2. Specify the Data Source name from the previous step.
    3. Specify the database user ID and password for accessing the database.

6.6.12: Configuring user profile support

When configuring the User Profile Manager, the administrator can specify:

- A user ID and password for accessing the user profile database and table
- The "data wrapper" class that implements user profile support
- The data source for storing user profile data in a table
- The enterprise bean classes for accessing user profiles
- Whether to enable user profiles at this time

The User Profile Manager settings require significant understanding of the classes that implement user profilesupport, particularly if the implementation involves enterprise beans.

To configure a user profile, follow these steps:

1. Develop a servlet that accesses the User Profile Manager API.
2. Create an enterprise application that uses the user profile implementation beans.(These beans can be found in installation_root/lib/userprofile.jar.) Use the same JNDI datasource name for both UP_ReadOnly and UP_ReadWrite.Make a note of the JNDI datasource name, because you must also specify this in userprofile.xml (step 4).
3. Deploy the enterprise application in the application server.
4. To installation_root/properties, add a file named userprofile.xml in the following format.Specify enterprise bean class names;data wrapper class name; andJNDI names for the read-only bean, read/write bean, and datasource (from step 2). You must also add user ID and password information for the JNDI datasource.An example of userprofile.xml follows:

   ```
   <?xml version="1.0"?>   <userprofile>      <userprofile-enabled>true</userprofile-enabled>
   <userprofile-wrapper-class>         <classname>
   com.ibm.servlet.personalization.userprofile.UserProfile         </classname>
   </userprofile-wrapper-class>      <userprofile-manager-name>         User Profile Manager
   </userprofile-manager-name>      <userprofile-bean>         <readonly-interface>
   com.ibm.servlet.personalization.userprofile.UP_ReadOnly         </readonly-interface>
   <readwrite-interface>            com.ibm.servlet.personalization.userprofile.UP_ReadWrite
   </readwrite-interface>         <readonlyhome-interface>
   com.ibm.servlet.personalization.userprofile.UP_ReadOnlyHome         </readonlyhome-interface>
   <readwritehome-interface>            com.ibm.servlet.personalization.userprofile.UP_ReadWriteHome
   </readwritehome-interface>
   <readonly-JNDI-lookupName>UP_ReadOnlyHome</readonly-JNDI-lookupName>
   <readwrite-JNDI-lookupName>UP_ReadWriteHome</readwrite-JNDI-lookupName>      </userprofile-bean>
   <userprofile-store>         <database-userid></database-userid>
   <database-password></database-password>         <database-datasource></database-datasource>
   </userprofile-store>    </userprofile>
   ```
5. Start the enterprise application.

6.6.13: Administering transports (overview)

Transports define the characteristics of the connections between theWeb server and the application server, across which requests for applicationsare routed. Specifically, they define theconnection between the Web server plug-in and the Web container ofthe application server.

Use transport properties to specify:

- How to manage the set of connections; for example, how many concurrentrequests to allow
- Whether to secure the connections with SSL
- Host and IP information for the transport participants

Administering the transport is closely related to administeringthe WebSphere plug-ins for Web servers. Indeed, without the plug-inconfiguration, the transport configuration is of little use.

You might also need to perform special configuration tasks to redirectapplication requests from the Web server machine to the applicationserver machine if the Web server is located on a machine remote from the application server. See section 7.3.

To avoid port contention,you must specify a unique port number for each application server instanceon a given machine.

The internal transport

For applications in a test or development environment (in other words,a non-production environment), you can use the *internal HTTP transport*system to serve servlets without an Web server plug-in. Simply use theinternal HTTP transport port (typically on port 9080).

For example, to serve the "snoop" servlet without an HTTP server, use the URL:

`http://`*`your.server.name`*`:`*`port`*`/servlet/snoop`

with *port* being the internal transport port number (typically 9080)and *your.server.name* being localhost if theApplication Server is on the local machine.

For a product environment, *do not* use the internal transport,as it lacks the performance available when using a Web server plug-in.

At times, the transport might be configured to use a port other than 9080. (The transport configuration is a part of the Web container configuration. In such cases, you need adjust your virtual host aliasand what you type into the Web browser, as described in the virtual host administrative overview.

Regenerating the WebSphere plug-in for the Web server

It is suggested you review the information about administering WebSphereplug-ins for Web servers, because you will need to regenerate the plug-in configurationeach time you specify or change an HTTP transport.

6.6.13.0: Properties of tranports

Key:

Applies to Java administrative console of Advanced Edition Version 4.0

Applies to Web administrative console of Advanced Single Server Edition Version 4.0

Applies to Application Client Resource Configuration Tool

HTTP transports

Dynamic Properties or Properties

Special configuration properties for the transport. Each transport type can contain custom properties specified by the transport provider.

For the Advanced Edition: The properties are displayed as fields intowhich you can enter values.

For the Advanced Single Server Edition: Enter the property names and their corresponding valuesinto the Properties area.

The properties are as follows:

Connection Backlog or MaxConnectBacklog

The maximum number of outstanding connect requests that the operating system will buffer while it waits for the application server to accept the connections. If a client attempts to connect when this operating system buffer is full, the connect request will be rejected.

Set this value to the number of concurrent connections that you would like to allow. Keep in mind that a single client browser browser might need to open multiple concurrent connections (perhaps 4 or 5); however, also keep in mind that increasing this value consumes more kernel resources.

The value of this property is specific to each transport.

ConnectionIOTimeOut or I/O Timeout

The maximum time (in seconds) to wait when trying to read or write data DURING a request.

The value of this property, after it is specified for the first transport, will be applied globally to all other HTTP transports in this administrative domain.

ConnectionKeepAliveTimeOut or Keep Alive Timeout

The maximum time (in seconds) to wait for the next request on a keep alive connections.

The value of this property, after it is specified for the first transport, will be applied globally to all other HTTP transports in this administrative domain.

I/O Timeout

See ConnectionIOTimeout

Keep Alive Timeout

See ConnectionKeepAliveTimeout

MaxConnectBacklog

See Connection Backlog

Maximum Keep Alives or MaxKeepAliveConnections

The maximum number of concurrent keep alive (persistent) connections across all HTTP transports. (HTTP 1.1 allows multiple requests to be sent on the same TCP/IP connection.) The default value is 90% of the maximum number of threads in the Web container thread pool. This prevents all of the threads from being held by keep alive connections so that there are threads available to handle new incoming connect requests.

The value of this property, after it is specified for the first transport, will be applied globally to all other HTTP transports in this administrative domain.

Maximum Requests per Keep Alive or MaxKeepAliveRequests

The maximum number of requests which can be processed on a single keep alive connection.

The value of this property, after it is specified for the first transport, will be applied globally to all other HTTP transports in this administrative domain.

Enable SSL or SSL Enabled

Whether to protect connections between the WebSphere plug-in and application server with Secure Socket Layer (SSL). See also the security properties

External

Whether this transport is for internal or external use

Host or Host Name or Transport Host

The host IP address to which to bind for the transport

Host Name

See Host

Port or Transport Port

The port to which to bind for the transport

Properties

See Dynamic Properties

SSL Enabled

See Enable SSL

Transport Host

See Host

Transport Port

See Port

Use global SSL default configuration

See the general security properties

See also the other application server properties.

6.6.13.1: Administering transports with the Java administrative console

Use the Java administrative console to administer transports.

Work with resources of this type by locating them in the application server properties:

1. In the tree view, click

   WebSphere Administrative Domain -> Application Servers -> ***application_server_name***
2. When the application server properties are displayed in the properties view, click the **Services** tabbed page.
3. Locate the Web container service in the list of services.
4. Click **Edit Properties**.
5. Click the **Transport** tabbed page.

6.6.13.1.1: Creating new transport configurations with the Java administrative console

1. Locate the transport settings.
2. Click **Add** to view the HTTP transport properties dialog.
3. Specify transport properties.
4. When finished, click **OK** to exit the transport properties.
5. Click **OK** to exit the Web container properties dialog.
6. Click **Apply** on the application server properties dialog.

6.6.13.1.2: Editing existing transport configurations with the Java administrative console

1. Locate the transport settings.
2. Select a transport from the list of HTTP Transports.
3. Click **Edit** to view the HTTP transport properties dialog.
4. Edit the transport properties.
5. When finished, click **OK** to exit the transport properties.
6. Click **OK** to exit the Web container properties dialog.
7. Click **Apply** on the application server properties dialog.

6.6.13.1.3: Removing transport configurations with the Java administrative console

1. Locate the transport settings.
2. Select a transport from the list of HTTP Transports.
3. Click **Remove**.
4. Click **OK** to exit the Web container properties dialog.
5. Click **Apply** on the application server properties dialog.

6.6.14: Administering database connections (overview)

The administrator of Advanced Edition (non-Single Server Edition) establishesdatabase connections for supporting data access by:

- Applications installed in the server runtime, possibly including the WebSphere Samples
- The WebSphere administrative server for obtaining its administrative data

For information about the latter topic, see the article about administeringthe administrative server and database.

The WebSphere administrator has an important role in establishingand maintaining connection pools through data sources and drivers:

- Configuring the resources used in connection pooling -- JDBC drivers and data sources

  The administrator needs to configure data sources and JDBC drivers for each brand and version of database from which enterprise applications or individual resources will require connections.
- Adjusting connection pooling parameters for optimal performance.

  Connection pools provide a way to share the connection overhead among multiple requests, but it is possible to configure too large a connection pool. The administrator needs to determine the optimal value for the pool size and other settings, based on environmental factors such as the operating system in use.

JDBC Drivers and data sources are used by Java components requiringdatabase access, such asservlets and enterprise beans. All entity beans require a datasource, either defined as a property of the enterprise bean, or as a property of the enterprise bean container.

6.6.14.0: Properties of JDBC and data source providers

The administrative tools use the following terms for the same type of configuration:
JDBC provider or data source provider

Key:

Applies to Java administrative console of Advanced Edition Version 4.0

Applies to Web administrative console of Advanced Single Server Edition Version 4.0

Applies to Application Client Resource Configuration Tool

Classpath or Server Class Path

The path to the JAR file containing the implementation code for the database driver, such as db2java.zip file for DB2.

See the reference below for a list of default locations.

Custom Properties

Name-value pairs for setting additional properties

Description

A description of the driver, for your administrative records

Implementation Class or Implementation Classname

The name of the Java data source class provided by the database vendor.

DB2:

- COM.ibm.db2.jdbc.DB2ConnectionPoolDataSource (one phase commit protocol)
- COM.ibm.db2.jdbc.DB2XADataSource (two phase commit protocol)

DB2 on iSeries - toolbox driver

- com.ibm.as400.access.AS400JDBCConnectionPoolDataSource (one phase commit protocol)
- com.ibm.as400.access.AS400JDBCXADataSource (two phase commit protocol)

DB2 on iSeries - native driver

- com.ibm.db2.jdbc.app.DB2StdConnectionPoolDataSource (one phase commit protocol)
- com.ibm.db2.jdbc.app.DB2StdXADataSource (two phase commit protocol)

Oracle:

- oracle.jdbc.pool.OracleConnectionPoolDataSource (one phase commit protocol)
- oracle.jdbc.xa.client.OracleXADataSource (two phase commit protocol)

Sybase:

- com.sybase.jdbc2.jdbc.SybConnectionPoolDataSource (one phase commit protocol)
- com.sybase.jdbc2.jdbc.SybXADataSource (two phase commit protocol)

Merant:

  - com.merant.sequelink.jdbcx.datasource.SequeLinkDataSource (one and two phase commit protocol)

Informix

  - com.informix.jdbcx.IfxConnection PoolDataSource (one phase commit protocol)
  - com.informix.jdbcx.IfxXADataSource (two phase commit protocol)

Name

A name for the driver. It is recommended you enter a name that is suggestive of the database product you are using, such as DB2JdbcDriver.

Node

The node (machine) on which to install the driver.

For Advanced Edition: Use the buttons on the Nodes panel to access dialogs for installing drivers on specific nodes and for uninstalling drivers.

Property Set

See the resource provider properties (only valid if you are using *Advanced Single Server Edition*).

Server Class Path

See Classpath

URL prefix

The URL prefix with which this driver is associated. The URL prefix is comprised of the protocol and subprotocol, separated by a colon (":").

The Database Name of the data source is appended to the URL prefix to produce the full JDBC URL of the database, such as jdbc:db2:was

Locating JDBC providers on your operating system

The following table lists the default locations of JDBC provider files. Seethe product prerequisites Web site for the most up-to-date information on the operating system brands and databases supported by IBM WebSphere Application Server. Column 1 in the table lists the operating system, and column 2 shows a list of the drivers that are available for use with each database supported for the operating system.

Operating system	Drivers
AIX	● DB2: *$DB2_HOME*/java12/db2java.zip or *$DB2_HOME*/java/db2java.zip ● Oracle: *$ORACLE_HOME*/jdbc/lib/classes12.zip ● Sybase: *sybase_install_root*/jConnect-5_2/classes/jconn2.jar Merant SequelLink: ❍ Complimentary copies with WebSphere Application Server: ■ *product_installation_root*/lib/sljc.jar ■ *product_installation_root*/lib/sljcx.jar

<table>
<tr><td>HP-UX</td><td><ul><li>DB2: <i>$DB2_HOME</i>/java12/db2java.zip or <i>$DB2_HOME</i>/java/db2java.zip</li><li>Oracle: <i>$ORACLE_HOME</i>/jdbc/lib/classes12.zip</li><li>Merant SequelLink:<ul><li>Complimentary copies with WebSphere Application Server:<ul><li><i>product_installation_root</i>/lib/sljc.jar</li><li><i>product_installation_root</i>/lib/sljcx.jar</li></ul></li></ul></li></ul></td></tr>
<tr><td>Linux (Intel)</td><td><ul><li>DB2: <i>$DB2_HOME</i>/java12/db2java.zip or <i>$DB2_HOME</i>/java/db2java.zip</li><li>Oracle: <i>$ORACLE_HOME</i>/jdbc/lib/classes12.zip</li></ul></td></tr>
<tr><td>Linux on S/390</td><td><ul><li>Oracle: <i>$ORACLE_HOME</i>/jdbc/lib/classes12.zip</li></ul></td></tr>
<tr><td>Solaris</td><td><ul><li>DB2: <i>$DB2_HOME</i>/java12/db2java.zip or <i>$DB2_HOME</i>/java/db2java.zip</li><li>Oracle: <i>$ORACLE_HOME</i>/jdbc/lib/classes12.zip</li><li>Sybase: <i>sybase_install_root</i>/jConnect-5_2/classes/jconn2.jar Merant SequelLink:<ul><li>Complimentary copies with WebSphere Application Server:<ul><li><i>product_installation_root</i>/lib/sljc.jar</li><li><i>product_installation_root</i>/lib/sljcx.jar</li></ul></li></ul></li></ul></td></tr>
<tr><td>Windows</td><td><ul><li>DB2: C:\SQLLIB\java\db2java.zip</li><li>Oracle: C:\Oracle\Ora81\jdbc\lib\classes12.zip</li><li>Merant SequelLink:<ul><li>Complimentary copies with WebSphere Application Server:<ul><li><i>product_installation_root</i>\lib\sljc.jar</li><li><i>product_installation_root</i>\lib\sljcx.jar</li></ul></li></ul></li></ul></td></tr>
</table>

6.6.14.0.1: Properties of data sources

Key:

Applies to Java administrative console of Advanced Edition Version 4.0

Applies to Web administrative console of Advanced Single Server Edition Version 4.0

Applies to Application Client Resource Configuration Tool

Category

An optional category string that can be used to classify or group the resource

Confirm Password **or Re-Enter Password**

Confirm the password that you entered in the preceding field

Connection timeout

The maximum time in seconds that requests for a connection wait if the maximum number of connections is reached and all connections are in use

This value must be a positive integer.

Custom Properties **or Property Set**

A set of custom name-value pairs describing properties of the data source

The following are the required properties for each database type:

DB2:

No required properties.

DB2 on iSeries -- toolbox driver:

serverName

The name of the server from which the data source will obtain connections, such as "MyServer"

DB2 on iSeries -- native driver:

No required properties.

Oracle:

URL

The url indicating the database from which the data source will obtain connections, such as "jdbc:oracle:thin:@myServer:1521:myDatabase," where "myServer" is the server name, "1521" is the port it is using for communication, and "myDatabase" is the database name.

Set the user and password in the field provided in the console.

Sybase:

serverName

The name of the database server, such as "db_machine" portNumber

The tcpip port number through which all communications to the server take place, such as 4100.

Merant:

serverName

The name of the server where SequeLinkServer resides, such as "MyServer"

portNumber

The TCP/IP port SequeLinkServer uses for communication. By default, SequeLinkServer uses port 19996, such as "19996".

disable2Phase

By default, two phase connections ae used by Merant always, because the same data source class is used for one phase and two phase commit protocols.

To use one phase connections, set disable2Phase to true.

Set the user and password in the field provided in the console.

Informix:

ifxIFXHOST

The physical machine name

serverName

The name of the Informix instance on the physical machine

portNumber

The port number of the Informix instance

informixLockModeWait

By default, Informix throws an exception when it cannot acquire a lock, rather than waiting for the current owner of the lock to release it. To modify this behavior, set this property to the number of seconds to wait for a lock. The default is 0 seconds. Any negative value means to wait forever.

The following are some additional, optional properties for various database types:

Sybase:

connectionProperties

CHARSET_CONVERTER_CLASS=com.sybase.jdbc2.utils.TruncationConverter

Setting the CHARSET_CONVERTER_CLASS can prevent exceptions such as this one when performing a dataSource.getConnection() call:

```
java.io.IOException: JZ0I6: An error occured converting
UNICODE to the charset used by the server. Error
message: java.io.CharConversionException:
java.io.UnsupportedEncodingException: hp-roman8
```

Set additional connectionProperties by specifying them using the same pattern, separated by commas: *PROPERTY_NAME*=value;*PROPERTY_NAME*=value; ...

Database Name

The name of the database used to store entity bean data

This is required for DB2, and sometimes required for Sybase, Merant, and Informix (depending on your database configuration), and ignored for Oracle.

Description

A description of the data source, for your administrative records

Default Password **or Password**

The password for connecting to the database when no user ID and password pair is specified by the application. If the default password is specified, the default user ID must also be specified.

Default User ID **or User**

The user name for connecting to the database when no user ID and password pair is specified by the application. If the default user ID is specified, the default password must also be specified.

Disable Auto Connection Cleanup

Keeps the connection pooling software from automatically closing connections from this data source at the end of a transaction. This behavior is needed if you want to reuse the same connection across multiple transactions. When this is set, you must be sure to close the connection programmatically when you are through using it.

Idle timeout

The maximum time in seconds that an idle (unallocated) connection can remain in the pool before being removed to free resources.

This value must be a positive integer.

JDBC Provider

The JDBC provider (also known as data source provider) with which this data source is associated. It is used to connect to a relational database.

JNDI Name

The JNDI name for the resource, including any naming subcontexts. This name is used as the linkage between the platform's binding information for resources defined in the client application's deployment descriptor and actual resources bound into JNDI by the platform.

Maximum Connection Pool Size **or Maximum Pool Size**

The maximum number of connections that can be in the pool. If the maximum number of connections is reached and all connections are in use, additional requests for a connection wait up to the number of seconds specified in the Connection timeout property.

This value must be a positive integer.

Maximum Pool Size

See Maximum Connection Pool Size

Minimum Connection Pool Size **or Minimum Pool Size**

The minimum number of connections in the pool.

This value must be a positive integer.

Minimum Pool Size

See Minimum Connection Pool Size

Name

A name by which to administer the data source.

It is recommended that you enter a name that is suggestive of the database you will use to store entity bean data, such as WASDataSource, where WAS is the database name. The default value for this property is the value of the Name property prefixed with "jdbc/" (such as "jdbc/DataSourceName").

Orphan timeout

The maximum number of seconds that an application can hold a connection without using it before the connection can be returned to the pool.

This value must be a positive integer.

Note that the actual amount of time before a connection is closed is approximately twice the orphan timeout value.

Password

See Default Password

Property Set

See the custom properties

Re-Enter Password

See Confirm Password

Statement Cache **or Statement Cache Size**

The maximum number of prepared statements to cache for the data source. The limit is shared among all connections. The default value is 100.

User

See Default User ID

6.6.14.1: Administering database connections with the Java console

Use the Java administrative console to administer JDBC providers and data sources.

Work with resources of this type by locating them in the tree view:

WebSphere Administrative Domain -> Resources -> JDBC Providers -> *provider_name*

Expand the tree further to see **Data Sources -> *datasource_name*** for each provider.

Data source instances will only be visible if they have been created.

6.6.14.1.1: Configuring new database connections with the Java administrative console

To configure JDBC providers and data sources, the resources necessary to allow applications to connect to databases, click **Console -> Wizards -> Create Data Source** from the console menu bar.This leads to the Datasource task wizard.

Using the Datasource wizard

You use the Datasource wizard to define and configure connections toa relational database that supplies a JDBC provider. Using the wizard sets some properties pertaining to data sources. To set all values for all of the properties, use theproperties dialogs for data access.

- Naming the data source
- Selecting whether to use an existing or new JDBC provider
- Creating a new JDBC provider
- Completing the data source

Naming the data source

On the Specifying Datasource Resources panel:

1. Name your data source.

   It is recommended that within the name you identify the database to which the data source will provide connections. For example, enter `WASDatasource`, where *WAS* is the name of the database. The name of the JNDI lookup for such a data source would be *jdbc/WASDatasource*.
2. Describe your data source.
3. If required by your database, specify the name of the database being connected to.

   The name is required for DB2, and sometimes required for Sybase, Merant, and Informix (depending on your database configuration), and ignored for Oracle.

   For example, for DB2, you could enter `WAS`, and your data source would, as a result, point to *jdbc:db2:WAS*. For Oracle, you would leave this field blank.

   View data source property reference
4. Click **Next**.

Selecting whether to use an existing or new JDBC provider

On the Specifying a JDBC Provider panel:

1. If your data source will use a JDBC provider previously configured, select the provider from the drop-down list for **Use an existing JDBC provider**. Otherwise, opt to create a new JDBC provider.
2. Click **Next**.

Creating a new JDBC provider

If you opted to create a new JDBC provider, you will see the Creating a JDBC Provider panel:

1. Name the provider. It is recommended that the name identify the database using the JDBC provider. For example, enter `DB2JdbcDriver`.
2. Describe the provider.
3. Select the Java class corresponding to the database being used. click the ellipses to open a dialog in which you can select the implementation class or enter an implementation class in the text field of the ellipses field.

   View data access properties help
4. Click **Next**.

Completing the data source

The Completing the Datasource Wizard panel lists the database and provider names, and the implementation class.

If you do not want to change the values specified, click **Finish**. Thewizard will define a data source with the values you specified, and display a message indicating whether the data source was successfully created. If the wizard encounters an error, a message will display explaining why a data sourcecould not be defined.

To change the values specified, click **Back** to return to the appropriatepanel(s), make any needed changes, and then click **Finish** on this panel.

⚠ For many databases, additional propertiesneed to be added after the Datasource Wizard is completed, before the data sourcecan be used. For a list of the properties required for the data source for eachdatabase type, see the **Custom Properties** field description in the data source property reference.

6.6.14.5: Additional administrative tasks for specific databases

For your convenience, this article provides instructions for enabling some popular database drivers, and performing other administrative tasks often required in order to provide dataaccess to applications running on WebSphere Application Server. These tasks are performedoutside of the WebSphere Application Server administrative tools, often using the databaseproduct tools. Always refer to the documentation accompanying your database driver as the authoritative andcomplete source of driver information.

See the WebSphere Application Server product prerequisites for the latest information about supported databases, drivers, and operating systems.

- Enabling JDBC 2.0
  - For DB2 on Windows NT
  - For DB2 on UNIX
- Sourcing the db2profile script on UNIX
- Using JTA drivers
  - For DB2 on Windows NT
  - For DB2 on UNIX
- For Oracle 8.1.7 two phase commit support
- For Sybase on AIX

Enabling JDBC 2.0

Ensure that your operating system environment is set up to enable JDBC 2.0 use.This is required in order to use data sources created through WebSphere ApplicationServer.

The following steps make it possible to find the appropriate JDBC 2.0 driverfor use with WebSphere Application Server administration:

Enabling JDBC 2.0 with DB2 on Windows NT

To enable JDBC 2.0 use on Windows NT systems:

1. Stop the DB2 JDBC Applet Server service.
2. Run the following batch file:
   `SQLLIB\java12\usejdbc2.bat`
3. Stop WebSphere Application Server (if it is running) andstart it again.

Perform the steps once for each system.

Determining the level of JDBC in use for DB2 on Windows NT

To see whether the JDBC level in use on your system:

- If JDBC 2.0 is in use, this file will exist:
  `SQLLIB\java12\inuse`
- If JDBC 1.0 is in use, this file will exist:
  `SQLLIB\java11\inuse`
  or there will be no java11 directory

Enabling JDBC 2.0 with DB2 on UNIX

Before starting WebSphere Application Server, you need to call $INSTHOME/sqllib/java12/usejdbc2 to use JDBC 2.0. For convenience, you might want to put this in your root user's startup script.For example on AIX, add the following to the root user's .profile:

```
if [ -f /usr/lpp/db2_07_01/java12/usejdbc2 ] ; then    . /usr/lpp/db2_07_01/java12/usejdbc2fi
```

Determining the level of JDBC in use for DB2 on your UNIX system

To determine if you are using JDBC 2.0, you can echo $CLASSPATH. If it contains

```
$INSTHOME/sqllib/java12/db2java.zip
```

then JDBC 2.0 is in use.

If it contains

```
$INSTHOME/sqllib/java/db2java.zip
```

then JDBC 1.0 is in use.

Sourcing the db2profile script on UNIX

Before starting WebSphere Application Server to host applications requiringdata access, source the db2profile:

```
.~db2inst1/sqllib/db2profile
```

where *db2inst1* is the user created during DB2 installation.

Using JTA drivers

Instructions are available for using JTA drivers on particular operatingsystems. See your operating system's documentation for more information.

With the JDBC programming model underlying WebSphere Application Server Version 4.0x, the term "JTA enabled" becomes somewhat odd, with respect toits Version 3.5 meaning. The goal of this section about "Using JTA drivers" is to provide information about the steps that make DB2 work nicely with applicationsutilizing XA classes -- that is, those whose dataSourceClasses implement javax.sql.XADataSource.

Using JTA drivers for DB2 on Windows NT

To enable JTA drivers for DB2 on Windows NT, follow these steps:

1. Stop all DB2 services.
2. Stop the IBM WebSphere Application Server administrative service.
3. Stop any other processes that use the db2java.zip file. (Note: If the **JVIEW** process is active, you must use the **Task Manager** utility to stop it.)
4. Make sure that you already enabled JDBC 2.0.
5. Start the DB2 services.
6. Configure DB2 to use JTS as the transaction processing (TP) monitor. From the DB2 Control Center, follow these steps:
   a. Right-click the DB2 instance that contains the database that is to be enabled for JTA access.
   b. Click **Multisite Update**, **Configure** to start the Smartguide utility.
   c. Click the **Use the TP monitor named below** radio button.
   d. Select **JTS** as the TP monitor.
   e. Click **Done**.
7. Bind the necessary packages to the database. From the **DB2 Command Line Processor** window, issue the following commands:

   ```
   db2=> connect to mydb2jtadb2=> bind db2home\bnd\@db2cli.lstdb2=> bind db2home\bnd\@db2ubind.lstdb2=>
   disconnect mydb2jta
   ```

   where *mydb2jta* is the name of the database that is to be JTA enabled, and *db2home* is the DB2 root installation directory path (for example, D:\ProgramFiles\SQLLIB\bnd\@db2cli.lst).
8. When you use an IBM WebSphere Application Server administrative client (suchas the WebSphere Administrative Console) to configure a JDBC provider, specifythe following settings:
   - **Server class path** = `%DB2_ROOT%/Sqllib/java/db2java.zip`
   - **Implementation class name** = `COM.ibm.db2.jdbc.DB2XADataSource`

Using JTA drivers for DB2 on UNIX

To enable JTA drivers on UNIX, follow these steps:

1. Stop all DB2 services.
2. Stop the IBM WebSphere Application Server administrative service.
3. Stop any other processes that use db2java.zip file.
4. Make sure that you already enabled JDBC 2.0.
5. Start the DB2 services.
6. Bind the necessary packages to the database.From the DB2 **Command Line Processor** window,issue the following commands:

   ```
   db2=> connect to mydb2jtadb2=> bind db2home\bnd\@db2cli.lstdb2=> bind db2home\bnd\@db2ubind.lstdb2=>
   disconnect mydb2jta
   ```
7. When you use an IBM WebSphere Application Server administrative client (suchas the WebSphere Administrative Console) to configure a JDBC provider, specifythe following settings:
   - **Server class path** = `$INSTHOME/sqllib/java12/db2java.zip`

     For example, if $INSTHOME is `/home/test`, the path will be `/home/test/sqllib/java12/db2java.zip`
   - **Implementation class name** = `COM.ibm.db2.jdbc.DB2XADataSource`

For Oracle 8.1.7 two phase commit support

The Oracle 8.1.7 thin driver can be used for JTA two phase support with the following restrictions:

- The thin driver that comes shipped with 8.1.7 may or may not work. Futurepatches from Oracle may work as well, but have not been tested. The driver that was available from the Oracle Technology Network Web site as of February 20, 2001 does work and is the recommended driver. Later versions on this Web site are expected to work, but have not been tested.

  To obtain the driver from the Oracle support Web, visit:

  ```
  http://technet.oracle.com/
  ```

You will need to be a registered user for the Oracle Technology Network to get the driverfrom this site. Contact Oracle for how to get access. After you have access:

1. On the left hand side of the screen, select "Software".
2. On "Download Oracle Products, Drivers, and Utilities"
3. On the "Select a Utility or Driver" selection, select "Oracle JDBC Providers".
4. Download the 8.1.7 driver for the platforms you use and follow the instructions for installing the new driver.

The above instructions are subject to change with no notice to IBM. This versionof the instructions could become inaccurate. Consult the Oracle site for the definitiveinstructions.

- The 8.1.7 driver must be used with 8.1.7 databases. 8.1.6 databases do notsupport the recover() and forget() methods and other problems have been encountered runningwith 8.1.6. Oracle does not support JTA with 8.1.6.
- For Oracle, JTA can only be used with container managed (CMP) beans.
- In order for the bean to create the table, the bean must first be started the JTA set tofalse. After the bean has created the table, JTA can be set back to true.
- An entity bean that accesses Oracle with JTA set to true must be configuredas follows:
  - In the deployment descriptor properties, under Transactions, under the Remote tab, set the Trasnaction Attribute to TX_REQUIRED.
  - Under Isolation, under the Remote tab, set the Isolation Level to TRANSACTION_READ_COMMITTED.
- A session bean that is used with an entity bean that accesses Oracle with JTA set totrue must be configured as follows:
  - In the deployment descriptor properties, Under Transactions, under the Remote tab, set the Transaction Attribute to TX_BEAN_MANAGED.
  - Under Isolation, under the Remote tab, set the Isolation Level to TRANSACTION_READ_COMMITTED.

Using JTA drivers for Sybase on AIX

To enable JTA drivers for use with Sybase on the AIX operating system,follow these steps:

1. At a command prompt, enable the Data Transaction Manager (DTM) by issuing these commands (one per line):

   ```
   isql -Usa -Ppassword -Sservername       sp_configure "enable DTM", 1    go
   ```

2. Stop the Sybase Adaptive Server database and start it again.
3. At a command prompt, grant the appropriate role authorization to the EJB user:

   ```
   isql -Usa -Ppassword -Sservername       grant role dtm_tm_role to EJB                  go
   ```

Notes about Sybase JTA drivers

Do not use a Sybase JTA connection in an enterprise bean method with an unspecified transaction context. A Sybase JTA connection does not support the local transaction mode. The implication is that the Sybase JTA connection must be used in a global transaction context.

6.6.14.6: Notes about various databases

This article provides miscellaneous tips for using supported databases. See also the related links.

Alwaysconsult the product prerequisites Web site for a list of the database brands and versions that aresupported by your particular Application Server version, edition, and fix pack.

- Do not drop the administrative database while the administrative server is running.
- DB2 performance on a local machine can be improved by setting up a local database as a remote instance. On UNIX systems, this is required. The configuration uses TCP/IP instead of shared memory.

  Oracle and Sybase also support client/server connections. Consult their product documentation for specifics.
- If using local DB2 databases for data access by session clients, in some cases, multiple connections for session clients cannot be established successfully. To avoid stale connections when there are large numbersof session clients, catalog the DB2 databases using a TCP/IP loopback.
  1. Set up a TCP/IP port in /etc/services, if a port for remote DB2 clients has not been established yet.
  2. Ensure that the TCP/IP communication protocol has been specified in the DB2COMM registry parameter.
     - To check the current setting of the DB2COMM parameter, enter `db2set DB2COMM`.
     - To update the DB2COMM registry variable to include TCP/IP, use the db2set command.

     For example:

     ```
             db2set DB2COMM=existing_protocol_names, tcpip
     ```
  3. Update the SVCENAME database manager configuration parameter to the connection service name as defined in /etc/services (step 1). For example:

     ```
             db2 update dbm cfg using svcename connection service name
     ```
  4. Catalog the loopback node. For example:

     ```
     db2 catalog tcpip node node_name remote 127.0.0.1 server connection_service_name
     ```
  5. Catalog the database as follows:

     ```
             db2 catalog db database_name as database_alias        db2 uncatalog db database_name
     db2 catalog db database_alias as database_name at node node_name
     ```

     This allows you to implement a TCP/IP loopback without needing to change the application to connect to the new alias and the USER and USING parameters.
  6. Stop DB2 and start it again to refresh the directory cache.
- When using Sybase 11.x, you might encounter the following error when HttpSession persistence is enabled:

  ```
  DBPortability W Could not create database table: "sessions" com.sybase.jdbc2.jdbc.SybSQLException:
  The 'CREATE TABLE' command is notallowed within a multi-statement transaction in the 'database_name'
  database
  ```

  where 'database_name' is the name of the database for holding sessions.

  If you encounter the error, issue the following commands at the Sybase command line:

  ```
  use database_namegosp_dboption db,"ddl in tran ",truego
  ```
- Sybase 12.0 does not support local transaction modes with a JTA enabled data source. To usea connection from a JTA enabled data source in a local transaction, Sybase patch EBF9422 mustbe installed.

6.6.14.8: Recreating database tables from the exported table DDL

To recreate database tables from the exported table DDL, you execute the exported table DDL file to create a table for CMP beans by hand. For instructionsfor exporting the table DDL, see the following:

- Exporting DDLs of EJB modules

Create the tables in the database of the data source configuration in the following order:

1. If you specified data source for the CMP bean, then use the data source for the bean.
2. If you specified data source for the EJB module of that CMP bean, then use the data source for the EJB module.
3. If neither of the above two conditions applies, use the default data source for the EJBContainer of the application server onto which the EJM module was installed.

The content of the table DDL file is the SQL strings needed to create a table. The content differs for different databases. The DDL file is essentially an SQL file, and each database has a different command to execute an SQL file.

Copy or paste the content of the table DDL file to a database command line, or use the command line option to take a SQL file as a parameter. The syntax of the command for supported databases is:

For DB2:

```
db2 -tf table_DDL_file_name
```

For Oracle:

```
sqlplus user_name/password
```

After a new window displays:

```
@table_DDL_file_name
```

For Sybase:

```
isql -Uuser_name -Ppassword -Sserver_name -i table_DDL_file_name
```

6.6.14.9: Administering data source providers and data sources with the Application Client Resource Configuration Tool

Use the Application Client Resource Configuration Tool to edit the configurations of data source providers (such as JDBC providers)and data sources, which are used by your application clients to access data fromdatabases.

Work with objects of this type by locating them in the tree that is displayed by the tool when you use it to openan EAR file. If file with which you are working contains data source providers and data sources, its tree will contain one or more of the following:

Resources -> ***application*****.jar** -> **Data Source Providers** -> ***data_source_provider_instance***

where ***data_source_provider_instance*** isa particular data source provider.

If you expand the tree further, you will also see the **Data Sources** folders containing the data source instances for each data source provider instance.

6.6.14.9.1: Configuring new data source providers (JDBC drivers) with the Application Client Resource Configuration Tool

During this task, you will create new data source providers (also known as JDBC providers)for your client application. Note, in a separate administrative task, the Java code for the required data source provider must be installed on the client machine on which the client application resides.

To configure a new data source provider:

1. Start the tool and open the EAR file for which you want to configure the new data source provider. The EAR file contents will be displayed in a tree view.
2. From the tree, select the JAR file in which you wantto configure the new data source provider.
3. Expand the JAR file to view its contents.
4. Click the folder called **Data Source Providers**. Do one of the following:
    - Right-click the folder and select **New Provider**.
    - On the menu bar, click **Edit** -> **New**.
5. In the resulting property dialog, configure the data source provider properties.
6. When finished, click **OK**.
7. On the menu bar, click **File** -> **Save** to save your changes.

6.6.14.9.1.1: Configuring new data sources with the Application Client Resource Configuration Tool

During this task, you will create new data sources for your client application.

1. In the tree, click the data source provider for which you want to create a data source.
   - Configure a new data source provider.
   - Or, click an existing data source provider.
2. Expand the data source provider to view its **Data Sources** folder.
3. Click the folder. Do one of the following:
   - Right-click the folder and select **New Factory**.
   - On the menu bar, click **Edit** -> **New**.
4. In the resulting property dialog, configure the data source properties.
5. When finished, click **OK**.
6. On the menu bar, click **File** -> **Save** to save your changes.

6.6.14.9.3: Removing data source providers (JDBC drivers) and data sources with the Application Client Resource Configuration Tool

Please see "Removing objects from EAR files with theApplication Client Resource Configuration Tool", as this task is similar for all object types supported by the tool.

6.6.14.9.4: Updating data source and data source provider configurations with the Application Client Resource Configuration Tool

During this task, you will modify (update) the configuration of an existing data source ordata source provider.

1. Start the tool and open the EAR file containing the data source or data source provider. The EAR file contents will be displayed in a tree view.
2. From the tree, select the JAR file containing the data source or data source providerthat you want to update.
3. Expand the JAR file to view its contents.
4. Keep expanding the JAR file contents until you locate the particular data source or datasource provider that you want to update. When you find it, do one of the following:
   - Right-click the object and select **Properties**
   - On the menu bar, click **Edit** -> **Properties**
5. In the resulting property dialog, update the properties. For detailed field help, see:
   - Data source provider properties
   - Data source properties
6. When finished, click **OK**.
7. On the menu bar, click **File** -> **Save** to save your changes.

6.6.15: Administering custom services (overview)

Configuring a custom service provides the ability to plug into a WebSphere application server to define a hook point to be executed at server startup and shutdown. It allows configurable extensions to the application server runtime.

The application server runtime will load the classname specified in thecustom service properties and call its initialize() method, passing in a Properties object that contains the configuration information for the service. The configuration information for the custom service can be arbitrary name-value pair properties, or information contained is an external file, such as an XML file on disk somewhere.

Custom service property in the administrative configuration file

For the multiple server edition of *Advanced Edition*, the administrative server configuration file containsthe property:

```
         com.ibm.websphere.preconfiguredCustomServices
```

that is a list of files containing XML configuration of custom services. The files in the list are read every time a new application server is configured.Any custom services contained in those files are automatically added to the configuration of the application server. This allows, for example, extensions to be defined in those files that will be made part of every server, automatically.The file with these preconfigured custom services must be in the same format as the XML config files used by *Advanced Single Server Edition*, (such as server-cfg.xml). The preconfigured custom service file format must include Domain, Node, and Server elements. The parser expects these and will fail unless these elements are in the file surrounding the actual custom service elements. The precise value for these elements (Domain, Node, Server) does not matter; they just must be present in the configuration file.

Attached is a sample service.xml that defines a preconfigured CustomService.

```
<!--      This file defines a custom service for the automatic generation of      plugin
configuration data. When this service is enabled, the plugin      configuration files will be
regenerated whenever you start an      application server. To enable this custom service by default,
change      the "enable" property from "false" to "true".      NOTE: Changes to this file will not
affect existing application      servers. When a new application server is created, it will contain
a custom service defined by the properties in this file. To change      the properties of this
custom service on an existing application server,      you must edit the service's properties in the
administration client. --><applicationserver:Domain xmi:version="2.0"
xmlns:xmi="http://www.omg.org/XMI"
xmlns:applicationserver="applicationserver.xmi"                           xmlns:server="server.xmi"
xmi:id="">  <nodes xmi:id="">    <servers xsi:type="applicationserver:ApplicationServer" xmi:id="">
<customServices xmi:id="PluginCfgService"   enable="false"  description="If enabled, the plugin
configuration files will be                      regenerated when the application server is started"
displayName="Automatic Generation of Plugin Configuration"      externalConfigURL=""
classname="com.ibm.websphere.plugincfg.initializers.AEPluginCfgService">      </customServices>
</servers>  </nodes></applicationserver:Domain>
```

6.6.15.0: Properties of custom services

Key:

Applies to Java administrative console of Advanced Edition Version 4.0

Applies to Web administrative console of Advanced Single Server Edition Version 4.0

Applies to Application Client Resource Configuration Tool

Classname

The classname of the service implementation. The class must implement the Services API of the product.

Classpath

Classpath for locating the ZIP and JAR files that contain the classes implementing this service

Configuration File URL **or External Config URL**

The URL of a custom service configuration file

Each application server has a separate file containing the configuration properties for its custom service.Different application servers can run with the same custom service, but with different custom service configuration values. Specify the URL for an external file containing addition custom service configuration data that could be more complex than simple name-value config properties.

Custom Properties **or Dynamic Properties**

Name-value pairs that will be passed to the init() method of the service

Description

An optional description for the custom service

Display Name **or Name**

A logical name for the service

Enable **or Enabled**

Whether the server will attempt to start and initialize the specified service

External Config URL

See Configuration File URL

Name

See Display Name

See also the other application server properties.

6.6.15.1: Administering custom services with the Java administrative console

Use the Java administrative console to administer custom services.

Work with resources of this type by locating them in the application server properties:

1. Display the application server properties.
2. When the application server properties are displayed in the properties view, click the **Custom** tabbed page.
3. Select the service from the list of services.

6.6.15.1.1: Adding and removing custom services with the Java administrative console

1. Locate the custom service instance. (Select it in the listof services).
2. Click the **Add** or **Remove** button.
3. If adding a custom service, specify the custom service properties and click **OK**.
4. Click **Apply** on the application server properties dialog.

6.6.15.1.2: Updating custom services with the Java administrative console

1. Locate the custom service instance. (Select it in the listof services).
2. Click the **Edit** button.
3. Specify the custom service properties.
4. When finished, click **OK**.
5. Click **Apply** on the application server properties dialog.

6.6.16: Administering virtual hosts (overview)

To define a virtual host, the administrator specifies informationsuch as:

- An administrative name for the virtual host
- One or more domain name aliases for the virtual host
- Mime types and extensions to recognize
- Whether this is the default virtual host

Ensuring there is a virtual host alias for each HTTP transport port

There must be a virtual host alias corresponding to each portbeing used by an HTTP transport. There is one HTTP transport in eachWeb container, with one Web container in each application server.

The following procedure describes how to find out (or set) the port for the HTTP transport, then create a corresponding virtual host alias. You will need to do so in the following cases:

- You are using the internal HTTP transport (described in the transport administration overview) with a portother than the default of 9080, or for some reason the virtual host does notcontain the usual entry for port 9080
- You have created multiple application servers (either standalone, or might have created a server group) that are using the samevirtual host. Because each server must be listening on a differentHTTP transport port, you need a virtual host alias for each one's transport port.

To discover or edit the transport port number for a given application server, and then create an alias corresponding to the port number:

1. View or edit the transport properties . Note the value in the fieldnamed **Port** or **Transport Port**, such as 9082.
2. Configure the virtual host to containan alias for the port number, such as *:9082, if9082 is port number in use by transport.
3. When you enter the URL for the application into a Web browser, includethe port number in the URL, such as:

   `http://localhost:9082/wlm/SimpleServlet`

   using the port number from the previous step.

6.6.16.0: Properties of virtual hosts

Key:

Applies to Java administrative console of Advanced Edition Version 4.0

Applies to Web administrative console of Advanced Single Server Edition Version 4.0

Applies to Application Client Resource Configuration Tool

Aliases

The list of one or more DNS aliases by which the virtual host is known. For example: *:9082

Default MIME Entries

A collection of MIME type extension mappings defined for the virtual host. If MIME entries are not specified at the Web module level, these MIME entries will apply.

Extension or Extensions

A list of file extensions that are mapped to the MIME type

Host Aliases

Configuration for aliases. An alias is the DNS host name and port number used by a client to form the URL request for a Web application resource (such as a servlet, JSP, or HTML page). For example, it is the "myhost:8080" portion of http://myhost:8080/servlet/snoop. When no port number is specified, the default port 80 is used.

Host Name

The IP address, DNS host name with domain name suffix, or just the DNS host name, used by a client to request a Web application resource (such as a servlet, JSP, or HTML page). For example: www.VHTest.ibm.com

MIME Entry or MIME Types

Configuration for the mapping of MIME types to extensions

MIME Types or Type

A MIME type that is mapped to the specific extension. For example: text, image, or application

Name

A logical name used for configuring Web applications to a particular host name. The default virtual host is typically suitable for most simple configurations.

Port

A port value can be specified in conjunction with the host name to indicate the port for which the Web server has been configured to accept client requests.

Type

See MIME type

6.6.16.1: Administering virtual hosts with the Java administrative console

Use the Java administrative console to administer virtual hosts.

Work with resources of this type by locating them in the tree view:

WebSphere Administrative Domain -> Virtual Hosts

The virtual host instances will be displayed in the details view.

6.6.16.1.1: Configuring new virtual hosts with the Java administrative console

1. Locate the **Virtual Hosts** folder.
2. Specify virtual host properties.
3. Open an editor on the hosts file (for example, on Windows NT the file is C:\WINNT\system32\drivers\etc\hosts) and add an entry for the virtual host. For example, add the line:

   ```
   127.0.0.1    www.VHTest.ibm.com
   ```
4. Stop the administrative server and start it again.

6.6.16.1.2: Adding and removing virtual host aliases with the Java administrative console

1. Locate the virtual host instance in the details view.
2. Right-click the instance and select **Properties** to display its properties dialog.
3. On the **General** tabbed page:
    - To remove an alias, select the alias from the Aliases list and click **Remove**.
    - To add an alias, click **Add** and type its information into the alias list.
4. Click **OK** to exit the properties window for the virtual host instance.

6.6.16.1.3: Adding, removing, and editing virtual host MIME types with the Java administrative console

1. Locate the virtual host instance in the details view.
2. Right-click the instance and select **Properties** to display its properties dialog.
3. On the **Advanced** tabbed page, select the MIME type from the list.
4. Click the **Add**, **Remove**, or **Edit** button.
    - To remove a MIME type, select it from the list and click **Remove**.
    - To add a MIME type, click **Add** and type its information into the list.
    - To edit a MIME type, select it from the list and click **Edit**. Specify properties. Click **OK**.
5. Click **OK** to exit the properties window for the virtual host instance.

6.6.18: Securing applications

For purposes of security, Application Server categorizes assetsinto two classes: resources and applications.

- *Resources* are individual components, such as servlets and enterprise beans.
- *Applications* are collections of related resources.

Security can be applied to applications and to individual resources. Setting up security involves the following general steps:

1. Setting global values for use by all applications.
2. Refining settings for individual applications.

Securing applications with IBM WebSphere ApplicationServer product security involves a series of tasks. Completing thetasks results in a set of policies defining *which*users have access to *which* methods or operations in *which*applications.

For example, the security administrator establishes policies specifyingwhether the user *Bob* is permitted to use the company's Inventoryapplication to perform a write operation, such as changing the numberunits of merchandise recorded in the company's inventory database.

The product security server works withthe selected user registry or directory product to enforce thepolicies whenever a user tries toaccess a protected application. For example, *Bob* might beprompted for a digital certificate verifying his identity when hetries to use the Inventory application.

6.6.18.0: General security properties

Key:

Applies to Java administrative console of Advanced Edition Version 4.0

Applies to Web administrative console of Advanced Single Server Edition Version 4.0

Applies to Application Client Resource Configuration Tool

Cache Timeout **or Security Cache Timeout**

Time after which the authentication cache will be refreshed. Caching can improve performance with respect to authentication lookups.

Specify this value in seconds, with a minumum of 30.

Default SSL Configuration **or Use global SSL default configuration**

Apply the default SSL configuration to the entire administrative domain.

For *Advanced Edition*, see Configuring SSL support instructions.

Enabled **or Enable Security**

Whether global security is enabled. When security is not enabled, all other security settings are not validated or used.

⚠ For *Advanced Edition* (non-Single Server), when security is enabled for the first timewith the LTPA authentication mechanism selected, you will be prompted toenter a password for encrypting and decrypting LTPA keys. Makesure you remember the password! For more information about LTPA keys, refer tothe article about making LTPA-secured calls across WebSphere domains.

Security Cache Timeout

See Cache Timeout

Use Domain Qualified User Names

When the value of this setting is true, user names returned by calls such as getUserPrincipal() will be qualified with the security domain in which they reside

Use global SSL default configuration

See the Default SSL Configuration field description

6.6.18.0.1: Properties for configuring Secure Socket Layer (SSL) support

Key:

Applies to Java administrative console of Advanced Edition Version 4.0

Applies to Web administrative console of Advanced Single Server Edition Version 4.0

Applies to Application Client Resource Configuration Tool

For *Advanced Edition* (non-Single Server), see Configuring SSL support instructions.

Client Authenticationor Enable Client Authentication

Whether the server and client should prove their identities through an exchange of keys

The SSL server is always authenticated to the client. If client authentication is enabled, the SSL client is also authenticated to the server. By default, client authentication is disabled.

Confirm Password

Type the password again, to confirm the correct spelling

Dynamic Properties

Name-value pairs that you can use to configure additional SSL settings beyondthose available in the administrative interface

com.ibm.ssl.protocol

This is the SSL protocol to be used (including its version). The possible values are SSL, SSLv2, SSLv3, TLS, or TLSv1. The default value, SSL, is backward-compatible with the other SSL protocols.

com.ibm.ssl.keyStoreProvider

The name of the key store provider to use. Specify one of the security providers listed in your java.security file which has a key store implementation. The default value is IBMJCE.

com.ibm.ssl.keyManager

The name of the key management algorithm to use. Specify any key management algorithm that is implemented by one of the security providers listed in your java.security file. The default value is IbmX509.

com.ibm.ssl.trustStoreProvider

The name of the trust store provider to use. Specify one of the security providers listed in your java.security file which has a trust store implementation. The default value is IBMJCE.

com.ibm.ssl.trustManager

The name of the trust management algorithm to use. Specify any trust management algorithm that is implemented by one of the security providers listed in your java.security file. The default value is IbmX509.

com.ibm.ssl.trustStoreType

The type or format of the trust store. The possible values are JKS, PKCS12, JCEK. The default value is JKS.

com.ibm.ssl.enabledCipherSuites

The list of cipher suites to enable. By default, this is not set and the set of cipher suites used are determined by the value of the SecurityLevel (HIGH, MEDIUM, or LOW). A cipher suite is a combination of cryptographic algorithms used for an SSL connection.

Enter a space-separated list of any of the following cipher suites:

```
SSL_RSA_WITH_RC4_128_MD5SSL_RSA_WITH_RC4_128_SHASSL_RSA_WITH_DES_CBC_SHASSL_RSA_WITH_3DES_EDE_CBC_SHASSL_DHE_RSA_WITH_DES_CBC_SHASSL_DHE_RSA_WITH_3DES_EDE_CBC_SHASSL_DHE_DSS_WITH_DES_CBC_SHASSL_DHE_DSS_WITH_3DES_EDE_CBC_SHASSL_RSA_EXPORT_WITH_RC4_40_MD5SSL_RSA_EXPORT_WITH_DES40_CBC_SHASSL_RSA_EXPORT_WITH_RC2_CBC_40_MD5SSL_DHE_RSA_EXPORT_WITH_DES40_CBC_SHASSL_DHE_DSS_EXPORT_WITH_DES40_CBC_SHASSL_RSA_WITH_NULL_MD5SSL_RSA_WITH_NULL_SHASSL_DH_anon_WITH_RC4_128_MD5SSL_DH_anon_WITH_DES_CBC_SHASSL_DH_anon_WITH_3DES_EDE_CBC_SHASSL_DH_anon_EXPORT_WITH_RC4_40_MD5SSL_DH_anon_EXPORT_WITH_DES40_CBC_SHA
```

Enable Client Authentication

See Client Authentication

Enable Crypto Token Support

Whether cryptographic token support is enabled. If this is selected, then the values on the **Crypto Token** panel are used. After enabling Crypto Token support, stop your application serverand start it again for the change to take effect.

A crypto token is a hardware or software device which has a built-in key store implementation. The exact values for the following fields should be documented in the documentation of the supported cryptographic device.

Enable SSL

Whether to enable SSL support

Key File Format

The format of the key file. Possible values are JKS, PKCS12, and JCEK

Key File Name

The fully qualified path to the key file that contains public keys and perhaps private keys. See below for a note about the name.

An SSL key file can be created with the IKeyMan key management utility, or it may correspond to a hardware device if one is available. In either case, this specifies the source for personal certificates, as well as for signer certificates unless a trust file is specified.

⚠ The default Key File and the default Trust File contains a test certificate, and is only intended for use in a test environment. The default key files should never be used in a production environment because the private keys are same on all the WebSphere installations. Please refer to the the introduction to security certificates for information about creating and managing digital certificates for your WebSphere domain.

Key File Password

The password for accessing the key file

Library File

The DLL or shared object which implements the interface to the cryptographic device

Password

The password for the cryptographic device

Security Level

The security level can be HIGH, MEDIUM, or LOW and is a user-friendly way of enabling a certain set of cipher suites. The Security Level can be overridden by giving an explicit value to the dynamic property named **com.ibm.ssl.EnabledCipherSuites** (a **Dynamic Property** described previously). The mapping of security level to enabled cipher suites is as follows.

If the security level is HIGH, the enabled cipher suites are:

SSL_RSA_WITH_RC4_128_MD5
SSL_RSA_WITH_RC4_128_SHA
SSL_RSA_WITH_DES_CBC_SHA
SSL_ RSA_WITH_3DES_EDE_CBC_SHA
SSL_DHE_RSA_WITH_DES_CBC_SHA
SSL_DHE_RSA_WITH_3DES_EDE_CBC_SHA
SSL_DHE_DSS_WITH_DES_CBC_SHA
SSL_DHE_DSS_WITH_3DES_EDE_CBC_SHA

If the security level is MEDIUM, the enabled cipher suites are:

SSL_RSA_EXPORT_WITH_RC4_40_MD5
SSL_RSA_EXPORT_WITH_DES40_CBC_SHA
SSL_RSA_EXPORT_WITH_RC2_CBC_40_MD5
SSL_DHE_RSA_EXPORT_WITH_DES40_CBC_SHA
SSL_DHE_DSS_EXPORT_WITH_DES40_CBC_SHA

If the security level is LOW and a *server* SSL configuration, the enabled cipher suites are:

SSL_RSA_WITH_NULL_MD5
SSL_RSA_WITH_NULL_SHA
SSL_DH_anon_WITH_RC4_128_MD5
SSL_DH_anon_WITH_DES_CBC_SHA
SSL_DH_anon_WITH_3DES_EDE_CBC_SHA
SSL_DH_anon_EXPORT_WITH_RC4_40_MD5
SSL_DH_anon_EXPORT_WITH_DES40_CBC_SHA

If the security level is LOW and a client SSL configuration, the enabled cipher suites are:

SSL_RSA_WITH_NULL_MD5
SSL_RSA_WITH_NULL_SHA

Token Type

The type of token, such as PKCS#11

Trust File Format

The format of the specified trust file

Trust File Name

The fully qualified path to a trust file containing the public keys. See below for a note about the name.

As with the SSL key file, this can be created with the IKeyMan utility, or it may correspond to a hardware device. Unlike the SSL key file, no personal certificates are referenced; only signer certificates are retrieved. If a trust file is not specified but the key file is specified, then the SSL key file is used for retrieval of signer certificates as well as personal certificates.

Trust File Password

A password for accessing the trust file

Note about key and trust file names: The Default SSL configuration data and the LDAP SSL configuration data managed by the WebSphere Application Server Security Center are shared by multiple nodes in the same security domain. Machines in the same security domain can host different operating systems, such as AIX and Windows 2000. Moreover, WebSphere Application Server installation path can be different on different host machines.

Hence it is not always possible to use absolute file path when specifying the location of the key store and the trust store. IBM WebSphere Application Server uses a symbolic link WAS_HOME (which equates to *product_installation_root*) to locate key store and trust store. For example, the key file name can be defined by

```
${WAS_HOME}/etc/ServerKeyFile.jks
```

The ServerKeyFile.jks must exist on all the host machine under the "etc" subdirectory of the *product_installation_root*. The contents in the key files can be different on different nodes, but the file names should match.

6.6.18.0.2: Properties for configuring security using local operating system

Key:

Applies to Java administrative console of Advanced Edition Version 4.0

Applies to Web administrative console of Advanced Single Server Edition Version 4.0

Applies to Application Client Resource Configuration Tool

Authentication Mechanism

Select how to authenticate users that try to access applications.

- Against the local operating system user registry, or
- Against an LTPA based LDAP registry or custom registry

Note that the local operating system user registry is intended for single machineand single application server environments. *Advanced Single Server Edition* supports only the local operating system mechanism.

When form-based login is used with local operating system authentication, the user information is stored in the HTTP session. Using an HTTP connection is not very secure, meaning the information can be obtained by others. Using SSL connections (HTTPS) between the browser and the Web server will improve security.

When security is enabled for the first timewith the LTPA authentication mechanism selected, you will be prompted toenter a password for encrypting and decrypting LTPA keys. Makesure you remember the password! For more information about LTPA keys, refer tothe article about making LTPA-secured calls across WebSphere domains.

Security Server ID **or Server ID**

The user ID under which the server runs, for security purposes. This ID is not associated with the system process. This ID refers to the application security context within the WebSphere Application Server product.

If using local operating system authentication, the following conditions apply:

- On UNIX operating systems, the ID must be root or have root authority.
- On Windows operating systems, the account must be a member of the Administrators group and must have the rights to "Log on as a service" and "Act as part of the operating system." If the Windows machine is a member of an NT domain, then the ID must also be an administrator in the NT domain. Do not use an account whose name matches the name of your machine or Windows Domain.

If using LDAP or custom registry authentication (not available for *Advanced Single Server Edition*), the following conditions apply:

- The user should be a valid user in the LDAP or custom registry
- The user should *not* be a root DN or administrator DN because those users are not always in the directory in all LDAP implementations.

Security Server Password **or Server Password**

The password corresponding to the server ID

6.6.18.0.3: Properties for configuring security using Lightweight Third Party Authentication (LTPA)

Key:

Applies to Java administrative console of Advanced Edition Version 4.0

Applies to Web administrative console of Advanced Single Server Edition Version 4.0

Applies to Application Client Resource Configuration Tool

Domain

Restrict SSO to servers in the domain you specify in this field. This domain name is used when creating HTTP cookies for Single Sign On. It determinesthe scope to which Single Sign On applies.

For example, a domain of austin.ibm.com would allow Single Sign On to work between WebSphere application server A at serverA.austin.ibm.comand WebSphere application server B at serverB.austin.ibm.com. Note that cross-domainSingle Sign On is not supported. That is, a server at austin.lotus.com, and anotherat austin.ibm.com cannot partipicate in WebSphere Single Sign On.

Enable Single Sign On

Causes your LTPA directory service to store extra information in the tokens so that other applications can accept clients as already authenticated by WebSphere Application Server. When clients try to access the other applications, they will not be interrupted and asked to log in.

When you enable Single Sign On, the **Domain** field will be enabled. You must enter a DNS domain name. See the **Domain** field description for more information. The **Limit to SSL connections only**check box will also be enabled. The **Import Keys** and **Export Keys** button will also be enabled.

Enable Web Trust Associations

When enabled, one or more trust associations will be active. Trust associations enable a third party reverse proxy server to perform authentication on behalf of the WebSphere Application Server security component. To do so, you need to create a corresponding interceptor for the reverse proxy server and determine how "trust" will be established between them. See the security documentation in the InfoCenter for additional information.

Limit to SSL connections only

Specifies to use a connection with SSL for Single Sign On, to prevent the SSO token from flowing over non-secure connections. When this is set, form-based authentication will not work when resources are accessed over HTTP. The resources can be accessed only over HTTPS.

If this property is set and form-based login is used for authentication, the resources can be accessed only using secure connections (HTTPS). Connections that are not secure (HTTP) will not work. If basic login for authentication is used and the access is through an connection that has not been secured, then SSO will not work. The user will be prompted to log in again.

Token Expiration

How many minutes can pass before a client using an LTPA token must authenticate again. LTPA uses tokens to store the authenticated status of a client.

A positive integer indicates the token life, in minutes

6.6.18.0.4: Properties for mapping security roles and "run as" roles to users and groups

Key:

Applies to Java administrative console of Advanced Edition Version 4.0

Applies to Web administrative console of Advanced Single Server Edition Version 4.0

Applies to Application Client Resource Configuration Tool

Note, clicking **Cancel** in the Security Center will not undo the changes made to the roles.

Roles

Roles to which you want to map users and groups in order to give the users and groups permissionto run as those roles.

Users

Users to which you want to map roles. The users must be defined in your chosen authentication mechanism.

Groups

Groups to which you want to map roles. The groups must be defined in your chosen authentication mechanism.

6.6.18.0.5: Properties for configuring using custom user registry (pluggable user registry)

Key:

Applies to Java administrative console of Advanced Edition Version 4.0

Applies to Web administrative console of Advanced Single Server Edition Version 4.0

Applies to Application Client Resource Configuration Tool

Display these settings by selecting the **Custom User Registry** radio buttonlocated in the middle of the **Authentication** tabbed page when LTPA is the selected authentication mechanism.

To add or remove custom settings, besides thoseavailable in the administrative console, click the **SpecifyCustom Settings** button.

Custom User Registry Classname

The name of the custom user registry implementation class file. This should be a dot separated class name.

For example, if the implementation file is com/myCompany/sampleRegistry.java, then enter com.myCompany.sampleRegistry. The class file should be in the WebSphere Application Server classpath. (See InfoCenter article 6.4.1 about setting classpaths.)

Security Server ID

The user ID under which the server runs, for security purposes. This user should be a valid user in the custom user registry.

Security Server Password

The password corresponding to the Security Server ID.

6.6.18.0.6: Custom properties for custom user registry

Key:

Applies to Java administrative console of Advanced Edition Version 4.0

Applies to Web administrative console of Advanced Single Server Edition Version 4.0

Applies to Application Client Resource Configuration Tool

Use the **Add** button to enter new name-value pairs. Use **Remove** to remove a selectedsetting.

Name

The name of any user defined custom registry properties.

Value

The value for the corresponding property.

6.6.18.0.7: Properties for configuring LDAP support

Key:

Applies to Java administrative console of Advanced Edition Version 4.0

Applies to Web administrative console of Advanced Single Server Edition Version 4.0

Applies to Application Client Resource Configuration Tool

Display these settings by selecting the **LDAP** radio buttonlocated in the middle of the **Authentication** tab when LTPA is the selected authentication mechanism.

Click the **Advanced** button to set advanced LDAP properties. Click the **SSL Configuration** button to set SSL properties for LDAP.

Base Distinguished Name

The base distinguished name of the directory service, indicating the starting point for LDAP searches of the directory service. (See RFC 1779 for a discussion of this technique).For example, for a user with a DN of`cn=John Doe, ou=Rochester, o=IBM, c=US`, the base DNcan be specified as any of (assuming a suffix of c=us):

- `ou=Rochester, o=IBM, c=us`
- `o=IBM, c=us`
- `c=us`

This field is not case sensitive.

This field is required for all LDAP directories except theDomino Directory. If you are using the Domino Directory andyou specify a Base Distinguished Name, you will *not* beable to grant permissions to individual Web users for resourcesmanaged by your WebSphere application server.

Bind Distinguished Name

The distinguished name for application server to use to bind to the directory service. If no name is specified, the application server binds anonymously. See the Base Distinguished Name field description for examples of distinguished names.

Bind Password

The password for the application server to use to bind to the directory service

Directory Type

The directory service product to use to locate information against which to authenticate users and groups.

Modifications to the default values in the advanced LDAP properties will cause this field valueto change to Custom.

Host

The host ID (IP address or DNS name) of the LDAP server

Port

The host port of the LDAP server. The port number will default to 389 if none is specified.

If multiple WebSphere application servers are installed and configured to run in the sameSingle Sign On domain, or if the WebSphere application server will inter-operate with a previousversion of WebSphere application server, then it is important that the port number match inall configurations.

For example, if the LDAP port is explicitly specified as 389 in a Version 3.5.x configuration, and a Version 4.0 application server is going to inter-operate with the V3.5.xserver, then port 389 should also be specified explicitly for the Version 4.0 server. Notethat this is true even though the default port number is 389 -- if the port is specifiedexplicitly in one server configuration, it should be specified explicitly in allserver configurations.

Security Server ID

The user ID under which the server runs, for security purposes

If using LDAP or custom registry authentication (not available for *Advanced Single Server Edition*), the following conditions apply:

- The user should be a valid user in the LDAP or custom registry
- The user should *not* be a root DN or administrator DN because those users are not always in the directory in all LDAP implementations.

Security Server Password

The password corresponding to the Security Server ID

6.6.18.0.8: Properties for Select Users/Groups window

Key:

Applies to Java administrative console of Advanced Edition Version 4.0

Applies to Web administrative console of Advanced Single Server Edition Version 4.0

Applies to Application Client Resource Configuration Tool

The following three options can be selected in any combination. See belowfor important usage notes.

Everyone

Grants anyone and everyone the access to the role. This choice basically provides no security protection.

All Authenticated Users

Grants users who are authenticated access to the resource.

Select Users/Groups

Grants users or groups whom you select access to the role.

Generally, it is preferable to grant groups rather than individual users access to a role. It is easier to manage roles mapped to groups because there are typically fewer groups than users, users can be added to or removed from groups outside of WebSphere, and the authorization table has fewer entries, which can improve performance.

Usage notes

- If "Everyone" is selected then any other selections will be ignored.
- If "All authenticated users" is selected, but "Everyone" is not, then "Select users/groups" will be ignored.
- When "Select users/groups" is selected, the search button can be used to select users and groups using a pattern.

  For better performance, avoid using general wildcard search (* for example) if the target registry contains a large number of users or groups. Currently, only the first 1000 users and the first 1000 groups will be displayed. The display name is attached to the security name in the "Available Users/Groups" panel.

6.6.18.0.9: Advanced properties for configuring LDAP support

Key:

Applies to Java administrative console of Advanced Edition Version 4.0

Applies to Web administrative console of Advanced Single Server Edition Version 4.0

Applies to Application Client Resource Configuration Tool

If any of the user and group filters are modified from their default value, the **Directory Type** field value on the **Authentication** tabbed page will change to Custom.

Certificate Filter

If you specified the filter Certificate Mapping, use this property to specifythe LDAP filter to use to map attributes in the client certificate to entries in LDAP.Note that if more than one LDAP entry matches the filter specification at runtime,then authentication will fail because it results in an ambiguous match.

The syntax or structure of this filter is:

```
LDAP attribute=${Client certificate attribute}
```

For example:

```
uid=${SubjectDN}
```

The left side of the filter specification is an LDAP attribute that depends on theschema that your LDAP server is configured to use. The right side of the filter specificationis one of the public attributes in your client certificate. Note that the right side mustbegin with ${ and end with }.

The following certificate attribute values may be used on the right side of the filterspecification. Note that the case of the strings *is* important.

- ${UniqueKey}
- ${PublicKey}
- ${Issuer}
- ${NotAfter}
- ${NotBefore}
- ${SerialNumber}
- ${SigAlgName}
- ${SigAlgOID}
- ${SigAlgParams}
- ${SubjectDN}
- ${Version}

To enable this field, select CERTIFICATE_FILTER for the Certificate Mapping.

Certificate Mapping

Whether to map X.509 Certificates into an LDAP directory by EXACT_DN or CERTIFICATE_FILTER. Specify CERTIFICATE_FILTER to use the specified Certificate Filter for the mapping.

Group Filter

An LDAP filter clause for searching the registry for groups. It is typically used for Security Role to Group assignment. It specifies the property by which to look up groups in the directory service. For more information about this syntax, see the LDAP directory service documentation.

Group ID Map

An LDAP filter that maps the short name of a group to an LDAP entry. Specifies the piece of information that should represent groups when groups are displayed.

For example, to display groups by their names, specify *:cn. The * is a wildcard character that searches on any object class in this case. This field takes multiple objectclass:property pairs delimited by a semicolon (";").

Group Member ID Map

An LDAP filter that identifies User to Groups memberships. Specifies which property of an objectclass stores the list of members belonging to the group represented by the objectclass. This field takes multiple objectclass:property pairs delimited by a semicolon (";"). For more information about this syntax, see the LDAP directory service documentation.

Initial JNDI Context Factory

Java classname of the initial context factory of a provider

User Filter

An LDAP filter clause for searching the registry for users. It is typically used for Security Role to User assignment. It specifies the property by which to look up users in the directory service.

For example, to look up users based on their user IDs, specify (ampersand(uid=%v)(objectclass=inetOrgPerson) where ampersand is the ampersand symbol.

For more information about this syntax, see the LDAP directory service documentation.

User ID Map

An LDAP filter that maps the short name of a user to an LDAP entry. Specifies the piece of information that should represent users when users are displayed.

For example, to display entries of the type object class = inetOrgPerson by their IDs, specify inetOrgPerson:uid. This field takes multiple objectclass:property pairs delimited by a semicolon (";").

6.6.18.0.10: Properties for mapping "Run As" roles to users

Key:

Applies to Java administrative console of Advanced Edition Version 4.0

Applies to Web administrative console of Advanced Single Server Edition Version 4.0

Applies to Application Client Resource Configuration Tool

Security Name

For LDAP, a security name is the full distinguished name, such as CN=Bob Smith, o=austin.ibm.com.

For the Windows operating system, it is the user name with the hostname or domain name attached, such as myDomain\user1.

Short Name

For LDAP, a short name can be the uid, such as bob.

For the Windows operating system, it is the user name without the hostname or domain name attached, such as user1.

Password

The password corresponding to the User

User

The user **Short Name** or **Security Name** as entered in other fields

The user name entered here depends on the selection in the **Select Users/Groups/Group** panel under the **Role Mapping** tabbed page of the Security Center.

If "Everyone (no authentication)" is selected, the user name defined in this panel is optional. Any user name in the current registry is valid.

If "Everyone (no authentication)" is not selected but the "All authenticated users" is selected then the user name is required. Any user name in the current registry is valid.

If the "Select users/groups" is the only selection then the user name is required. This user name must have been assigned to the same role in the **Role Mapping** panel or belong to a group that has been assigned to the same role.

6.6.18.0.11: Properties for encrypting and decrypting LTPA keys

Key:

☕ Applies to Java administrative console of Advanced Edition Version 4.0

🌐 Applies to Web administrative console of Advanced Single Server Edition Version 4.0

C:\ Applies to Application Client Resource Configuration Tool

Password ☕

The password to encrypt and decrypt the LTPA keys. This password should be used when importing these keys into other WebSphere Application Server administrative domain configurations (if any) and when configuring SSO for Domino Server.

6.6.18.1: Securing applications with the Java administrative console

To configure security, use the Security Center. Access the Security Centerby clicking **Console -> Security Center** on the console menu bar.

With it, you can complete the following security tasks:

- Enable product security
- Define a security realm and set of valid users
- Specify how to authenticate users seeking access to applications
- Grant users permissions to access applications

6.6.18.1.1a: Specifying global settings with the Java administrative console

1. Start the Security Center by clicking **Console -> Security Center** from the console menu bar.
2. Complete the task, referring to the information below for assistance.
3. Stop the administrative server and start it again for the changes to takeeffect.

   The next time the administrator opens the WebSphere AdministrativeConsole, the administrator will be prompted to log in (if security has been enabled), using an ID and passwordspecified during Security Centerconfiguration.

General

Use the General tab to specify whether to enable security. If the check box is *not*selected, any other security settings you specify will be disregarded.

This page also contains an option for setting a security cache timeout. The securitysystem caches authentication lookup information it receives from the user registry ordirectory service. Use this field to specify how long to cache the information (inseconds). Caching can improve lookup performance.

Authentication

Use the Authentication tabbed page to specify how to authenticate theinformation presented by users trying to access an application or resources.

The administrator can have users or groups authenticated against either the local operating system user registry (such as Windows NT User Manager program) or an LDAP or custom user registry.

Role Mapping

Use the Role Mapping page to assign users in particular groups to specific roles.Role mapping gives particular users or groups authorization to access one ormore applications defined by a role.

The users, groups and roles were defined when the application was installed or configured.

Run As Role Mapping

Use the Run As Role Mapping page to assign only one user to a specific role.The application is delegated to that user. Any user who knows the assigned user'sID and password can access the application.

Administrative Roles

Use the Administrative Roles page to map an administrative role to at least one user or group.

6.6.18.1.2: Securing cloned applications

In an environment containing server groups (formerly called *models*)and clones, each server group and clone must be secured individually. Securing a server group does not automatically secure its clones.

For example, if you clone an application server that contains secure enterprise applications, then you need to secure those same enterpriseapplications (if you want to) on the cloned application servers.

Secure a cloned application as you would secure any new application.

6.6.18.1.4a.4.1: Supported directory services

For a list of supported directory services, see the prerequisites Web site discussed in the article about the site. An additional Custom option is available for tailoring any of the default filters to fit a *supported* LDAP directory service.

6.6.18.1a: Summary of security settings with the Java administrative console

Use the Security Center task wizard to specify global and default security settings for all applications:

- Global settings apply to existing and future applications and cannot be customized.
- Default settings apply only to future applications and can be customized.

The default settings are used as a template or starting point for configuring individual applications. The administrator should still explicitly configure security settings foreach application.

Task	Wizard page description	Global or default?
Enable security; specify how long to cache authentication lookup results	6.6.18.1a.1: General	Global
Specify how to authenticate users	6.6.18.1a.2: Authentication	Default
Select users and groups for roles	6.6.18.1a.3: Role Mapping	Global
Assign one user to each role	6.6.18.1a.4: Run As Role Mapping	Global
Select users and groups for administrative roles	6.6.18.1a.5: Administrative Roles	Global
Making LTPA-secured calls across WebSphere domains	6.6.18.1a.6: Authentication	Global
Configuring SSL support	6.6.18.1a.7: General	Default

IBM WebSphere Application Server provides security at several levels.The security characteristics of an individual application can come fromany of these levels. At the most general level are the global securitycharacteristics set up to act as application defaults. This filebriefly describes these global values.

In WebSphere, the global defaults for security apply to allapplications. Some of the values can be changed on anapplication-by-application basis, and others remain constant acrossall applications.

An example of a value that can be set on a per-application basisis the type of authentication procedure. You must establish adefault procedure, but this value is used for applicationsthat do not explicitly indicate how they will authenticate users.

An example of value that cannot be changed on a per-applicationbasis is whether to ignore security or not. In Application Server,security is either enabled or disabled. If it is enabled, allapplications are secured according to their configurations. Ifsecurity is disabled, all applications run unsecurely, regardlessof their configurations.

6.6.18.1a.1: Enabling security with the Java administrative console

IBM WebSphere Application Server security can be enabled or not enabled. If securityis not enabled, all other security settings are ignored.

Selecting how to enable security

The administrator can enable server security by selecting the **Enable Security**check box on the **General** tabbed page of the Security Center. The administrator can use the **General** tabbed page to specify additional general settings.

6.6.18.1a.2: Specifying how to authenticate users with the Java administrative console

Use the Authentication tab of the Security Centerwizard to specify how to authenticate or verify the user data receivedas a result of a challenge (such as a logon screen).

The WebSphere security server must havesome way to check the user ID and password, digital certificate, or otheruser identification for credibility. It relies on the authenticationmechanism specified by the administrator.

Selecting how to authenticate user data

Users can be authenticated by one of two authentication mechanisms, either theoperating system user registry or Lightweight Third-Party Authentication (LTPA).

The operating system user registry simply compares users to valid usersin the underlying operating system. When the administrator selects the Local Operating Systemauthentication mechanism, the Authentication tabbed page changes to allow theadministrator to set a security server ID and password under which the application will run. Theinformation is used for delegation of the application resource.

The Local Operating System authentication mechanism supports the basic challenge type. If the administrative server is running as a non-root user, then the Local OperatingSystem cannot be used. LTPA authentication in connection with LDAP or with the Custom User Registry must be used to enable security. Similarly, if the administrative server is beingused in a multi-node configuration, LTPA authentication must be used.

When the administrator selects **Lightweight Third-Party Authentication (LTPA)** as theauthentication mechanism, the **Authentication** tabbed page changes. This change enables theadministrator to specify LTPA settings and information about the Lightweight Directory Access Protocol (LDAP)-compliant directory service product to be used, or the custom user registry. LTPA causes a search to be performedagainst the selected registry (LDAP or custom user registry). LTPA supports both the basic and certificate challenge types.

The help files that describe the OS, LTPA, LDAP, and custom user registry settings provide guidance forcompleting options on the **Authentication** tabbed page.

6.6.18.1.a.3: Selecting users and groups for roles with the Java administrative console

Use the Role Mapping tab of the Security Center wizard to assign users or groups to a particular role. Different roles can have different security authorizations. Mapping users or groups to a role authorizes those users or groups to accessapplications defined by the role.

Users, groups and roles are defined when an application is installed or configured.

Mapping users or groups to roles

The administrator maps a user or group as follows:

1. In the Role Mapping tabbed pane, the administrator selects an application.
2. Click **Edit Mapping** to open the Role Mapping dialog.
3. In the Role Mapping dialog, the administrator selects a role and clicks on **Select** to open the Select Users/Groups dialog.
4. In the Select Users/Groups dialog, the administrator selects who is authorized access for the role.
5. Click **OK** when finished mapping a user or group to a role.
6. The administrator repeats the previous two steps for other roles or as needed.
7. Click **OK** to exit the **Role Mapping** dialog.
8. Click **OK** or **Apply** on the Security Center.

6.6.18.1.a.4: Assigning users to Run As roles using the Java administrative console

Use the **Run As Role Mapping** tab of the **Security Center** wizard to assign a user only to a particular Run As role. During delegation the user assigned to the Run As Role will be used when making invocation to other methods. See InfoCenter section 5.1.4, "The WebSphere delegation model," for detail description.

Before performing this subtask

Before completing the Run As Role Mapping subtask, the administrator needs to completesubtasks in the Role Mapping tabbed pane of the Security Center and map users or groupsto the roles.

Mapping users to Run As roles

To map users to Run As roles:

1. In the **Run As Role Mapping** tabbed pane, the administrator selects an application.
2. Click **Edit Mapping** to open the **Run As Role Mapping** dialog.
3. In the **Run As Role Mapping** dialog, the administrator selects a role and clicks on **Select** to open the Select User dialog.
4. In the Select User dialog, the administrator enters the User ID/Password of a user who should havebeen granted the same role or who belongs to a group that has been granted the same role(in the Role Mapping task).
5. Click **OK** when finished mapping a user to a Run As Role.
6. The administrator repeats the previous two steps for other roles or as needed.
7. Click **OK** to exit the **Run As Role Mapping** dialog.
8. Click **OK** or **Apply** on the Security Center.

6.6.18.1.a.5: Selecting users and groups for administrative roles with the Java administrative console

Use the **Administrative Roles** tabbed page of the **Security Center** wizard to assign users or groups to the administrative role.WebSphere security model has the configuration capability to assign any user or group to have the WebSphere administrator authority. This is encapsulated with the notion of an "AdminRole" which is scoped to the WebSphere administrative application. Any user who has been granted the administrative role, or is part of a group which has been granted the administrative role, will be able to administer the WebSphere administrative domain. This role will grant such a user or a group the capability to perform any WebSphere administrative function. For example, the administrator can create a new application server, stop a running server, deploy an application, and configure security settings.

Mapping users or groups to administrative roles

The administrator maps a user or group as follows:

1. In the **Administrative Roles** tabbed pane, the administrator selects an application.
2. Click **Edit Mapping** to open the **Administrative Roles** dialog.
3. In the Select Users/Groups dialog, the administrator selects who is authorized access for the role.
4. Click **OK** when finished mapping a user or group to a role.
5. Click **OK** to exit the **Administrative Roles** dialog.
6. Click **OK** or **Apply** on the Security Center.

6.6.18.1a.6: Making LTPA-secured calls across WebSphere domains with the Java administrative console

If applications in two different WebSphere Application Server domainsneed to be able to communicate, the two WebSphere application servers mustshare security information so that the servers themselves cancommunicate. Specifically, the LTPA component of the administrativeservers in both domains must use the same LTPA key. This allows the twoservers to communicate securely with each other, and it allows the called serverto decrypt security information from the calling server. Otherwise, the WebSphereapplication server in the calling domain cannot authenticateto the application server in the called domain.

See below for an example.

This article describes the procedure for making LTPA-secured calls:

1. Generate keys
2. Export the key information
3. Make the file accessible to the second domain
4. Import the key information

Generate keys

Use the **Generate Keys** button on the **Authentication** tabbedpage to generate LTPA keys.

When LTPA keys generated, you must provide a password that is used to protected the keys. This password is required when the keys are imported from a file into another WebSphere Application Server domain.

Export the key information

You must export the calling domain's LTPA keys to a fileso that the key can be made available to another domain,where the keys are imported from the file.

Before LTPA keys can be exported, they have to be created.Such keys are typically created when security is enabled for the first time using the LTPA authentication mechanism for the domain, or can be created any time by clicking the **Generate Keys** button. When the LTPA keys are created,you must provide a password that is used to protect the keys. Thispassword is required when the keys are imported from a fileinto another application, so you *must* have this password.

To export the LTPA key information, perform these steps:

1. Start the administrative server for the domain, if necessary.
2. Start the administrative console, if necessary.
3. Click on the **Console**action bar and then choose **Security Center** from the drop-downmenu.
4. Click the **Authentication** tab in the Security Center.
5. Ensure that LTPA is selected as the authentication mechanism.
6. Click the **Export Key** button.
7. When prompted, specify the name and location of the fileto contain the LTPA keys. You can use any file name and extension.Note the name and extension you specify; this file must laterbe imported by the application in the second domain.

8. Click **Save** to save the file.
9. Click **Cancel** to close the wizard. (This procedurehas not changed any global security setting, so there are nonew settings to save.)

Make the file accessible to the second domain

The file containing the exported keys must be installed in a locationwhere the importing administrative server can find it. For example, to move thefile from one machine to another, you can put it on a floppy disk andinstall it on the second machine. This file contains security keys,so treat it with care. Some sites have policies describing howsuch transfers can be done.

Import the key information

You must import the LTPA keys of calling domain from thefile. This allows the called domain to decrypt informationencrypted by the calling domain.

To import the key information from a file, perform these steps:

1. Start the administrative server for the domain, if necessary.
2. Start the administrative console, if necessary.
3. Click on the **Console**action bar and then choose **Security Center** from the drop-downmenu.
4. Click the **Authentication** tab in the Security Center.
5. Ensure that LTPA is selected as the authentication mechanism.
6. Click the **Import Key** button.
7. When prompted, select the file that was generatedduring the export step.
8. Click **Open**.
9. When prompted, type the LTPA password established wheninitially generating the keys.
10. Click **OK** to import the keys.
11. Stop and restart the administrative server.

Example of LTPA-secured calls across domains

Suppose that a servlet running in DomainA needs to call an enterprise bean running in Domain B.Before this exchange can take place, the two WebSphere applicationservers have to exchange LTPA key information. To exchange the necessaryinformation between the two domains, three things must be done:

1. The keys for the LTPA component in the calling application'sdomain must be exported to a file. In the example scenario,the calling application is the servlet.
2. The file must be made accessible to the administrative serverof the called WebSphere Application Server domain.
3. The key information from the calling domain must be importedby the LTPA component of the called domain. In the examplescenario, the called application is the enterprise bean.

6.6.18.1a.7: Configuring SSL in WebSphere Application Server

- "What is Secure Socket Layer?" and related concepts
- Overview: WebSphere Application Server's use of SSL
- Configuring SSL for browsers
- Configuring SSL for Web servers
- Configuring SSL for IBM HTTP Server, specifically
- Configuring SSL for WebSphere plug-ins for Web servers
- Configuring SSL for WebSphere Application Server

Overview: WebSphere Application Server's use of SSL

SSL (Secure Socket Layer) is used by several WebSphere Application Server components in order to provide secure communication. In particular, SSL is used by:

- HTTPS: the application server's built-in HTTPS transport.
- ORB. the application server's client and server ORB.
- LDAPS: the admin server's secure connection to the LDAP registry used for authentication. This is available only in WebSphere Application Server Advanced Edition.

The administrative model in WebSphere Application Server allows these various SSL components to be centrally managed by configuring the *default SSL Settings*. Furthermore, any of the *default settings* can be overridden by configuring the specific SSL settings for HTTPS, ORB, and LDAPS. This provides both central administration as well as individual configurability which may be required for the various uses of SSL.

Configuring SSL for the browser

Configuring SSL for the browser is browser-specific. Consult your browser documentation for instructions.

Generally speaking, when the you type "https://..." instead of "http://...", the browser creates an SSL connection instead of a simple TCP connection to the Web server. The browser then typically either prompts the user or fails to connect if it was unable to validate the Web server or to agree upon the level of security options (the strength of the encryption algorithm to use).

Configuring SSL for the Web server

Configuring SSL for the Web server depends on the type of Web server. Consult your Web server documentation forinstructions.

Generally speaking, when SSL is enabled, an SSL key file is required. This key file should contain both the CA certificates (signer certificates) as well as any personal certificates. Client authentication can also be enabled; by default, it is disabled.

In order for the client certificate (the certificate from the browser) to be forwarded by the WebSphere Web server plug-in to the WebSphere Application Server, client authentication must be enabled for the Web server. Enabling client authentication in WebSphere Application Server itself is not required unless you want to authenticate the WebSphere Web server plug-in (or any other clients connecting directly to the WebSphere Application Server over SSL).

Configuring SSL for IBM HTTP Server, specifically

This section provides a brief example of configuring SSL for IBM HTTP Server. See the IBM HTTP Server documentation for the most recent and complete instructions. Note also that the *httpd.conf.sample* file of your Web server provides examples of all directives, including the SSL-related directives.

1. Create a keyfile using the IHS key management utility.
    1. Create a directory at a location such as *"product_installation_root*/myKeys"

       This directory will be used to hold all of your SSL key files and certificates.
    2. Start the Key Management Utility from the IBM HTTP Server start menu.

       To start this utility on a Windows platform, click: **Start -> Programs -> IBM HTTP Server -> Start Key Management Utility**
    3. Click the **Key Database File** menu and select **New**.
    4. Specify settings and click **OK**:
        - Key Database Type: CMS Key Database File
        - File Name: WebServerKeys.kdb
        - Location: The path to your "myKeys" directory
    5. Enter a password for your SSL key file (twice for confirmation).
    6. Check the "Stash the password to a file?" option. Click **OK**.

       This causes a file named "WebServerKeys.sth" to be created containing an encoded form of the password. Note that this encoding prevents a casual viewing of the password but is not highly secure. Therefore, operating system permissions should be used to prevent all access to this file by unauthorized persons.

    7. When you see the list of default **Signer Certificates**, click the **Signer Certificates** menu and select **Personal Certificates**.

       If you have a server certificate from a CA (for example, Verisign), you can click **Import** to import this certificate into your SSL key file. You will be prompted for the type and location of the file containing the server certificate.

       If you do not have a valid server certificate from a CA, but want to test your system, click **New Self-Signed**.

       You will be prompted minimally to enter a **Key Label** such as "Test" and **Organization**, such as "IBM". Choose to use the default values for other values.
    8. Click the **Key Database File** menu and select **Close**.
2. Add the following lines to the bottom of your httpd.conf file:

```
        LoadModule ibm_ssl_module modules/IBMModuleSSL128.dll        Listen 443        SSLEnable
Keyfile "product_installation_root/myKeys/WebServerKeys.kdb"        # SSLClientAuth required
```

   This causes the Web server to listen on port 443 (the default SSL port).

   Uncomment the last line containing "SSLClientAuth required" if you want to enable client authentication. This will cause IHS to send a request for a certificate to the browser. Your browser may prompt you to choose a certificate to send to the Web server in order to perform client authentication.
3. Start your IBM HTTP Server.
4. Test your configuration from a browser by entering a URL such as:

```
https://localhost
```

   If you are using a self-signed certificate, instead of a certificate issued by a CA such as Verisign, then your browser should prompt you to see if you want to trust the unknown signer of the server's certificate. Additionally, if you enabled client authentication, then your browser may prompt you to select a certificate to send to the Web server in order to perform client authentication. The page should then be displayed.

Configuring SSL for WebSphere plug-ins for Web servers

After SSL is working between your browser and Web server, proceed to configure SSL between the Web server plug-in and the WebSphere Application Server product. This is not required if the link between the plug-in and application server is known to be secure or if your applications are not sensitive. If privacy of application data is a concern, however, this connection should be an SSL connection.

Step 1: Creating an SSL key file for the WebSphere Web server plug-in

When configuring SSL, you must first create an SSL key file.

Note that if you are using the IBM HTTP Server, you *may* use the same SSL key file which the Web server is using; however, it is recommended that separate SSL key files be used because the trust policy for the connection to the web server will likely be different than the trust policy for the connection to the application server.

For example, we may want to allow many browsers to connect to the Web server's HTTPS port, whereas we only want to allow a small, well-known number of WebSphere plug-ins to connect directly to a WebSphere application server's HTTPS port. The following is an example of how to create an SSL key file for your WebSphere plug-in which will only allow the plug-in to connect to the application server on it's SSL port.

1. Create the directory *product_installation_root*\myKeys if you have not already done so.

   This directory will contain all of the SSL key files and extracted certificates that you will create.
2. Start the key management utility of GSKit.

   GSKit is the SSL implementation used by the WebSphere plug-in, which is the same implementation used by the IBM HTTP Server.

   The default path on Windows to this utility is C:\Program Files\ibm\gsk5\bin\gsk5ikm.exe.
3. Click the **Key Database File** pulldown and select **New**.
4. Specify settings and click **OK**:
    - **Key database type**: CMS Key Database File
    - **File name**: plug-inKeys.kdb
    - **Location**: your myKeys directory
5. Enter a password for your SSL key file (twice for confirmation).
6. Check the **Stash the password to a file?** option. Click **OK**.

   This causes a file such as "*product_installation_root*\myKeys\plug-inKeys.sth to be created containing an encoded form of the password. This encoding prevents a casual viewing of the password but is not highly secure. Therefore, operating system permissions should be used to prevent all access to this file by unauthorized persons.
7. When you see the list of default **Signer Certificates**, select the first certificate and click **Delete**.
8. Repeat the previous step until all of the signer certficates have been deleted.
9. Create a self-signed certificate:
    1. Click the **Signer Certificates** menu and select **Personal Certificates**.
    2. Click **New Self-Signed**.
    3. Enter "plug-in" for the **Key Label** and "IBM" for the **Organization**.

    4. Click **OK**.
10. Extract the certificate so that you can import it into the application server key file later.
    1. Click **Extract Certificate**.
    2. Specify settings:
        - **Base64-encoded ASCII data**: Data Type
        - **Certificate file name**: plug-in.arm
        - **Location**: path to your myKeys directory
    3. Click **OK**.
11. Click the **Key Database File** menu and select **Close**.

Step 2: Modifying the WebSphere Web server's plug-in configuration file

Now that you have created the SSL key file for the plug-in, edit the plug-in configuration file so that it references your key file.

The following is an example of the plug-in configuration file. This configuration causes the plug-in to forward HTTP requests to the HTTP port of the application server, and to forward HTTPS requests to the HTTPS port of the application server.

The SSL configuration information is specified for *secureServer1*, which is the only member of the *secureServers* group. All HTTPS requests are forwarded to the *secureServers* group. (A server group is a concept that issupported only in *Advanced Edition*, not in *Advanced Single Server Edition*.)

The SSL key file is specified by the **keyring** property, and the stash file (which contains an encoded password) is specified by the **stashfile** property. Make sure that the path of this file is specified in your Web server configuration (for example, in "httpd.conf" for IHS).

```
<?xml version="1.0"?> <Config>     <Log LogLevel="Error"
Name=<"product_installation_root\logs\native.log">  <VirtualHostGroup Name="standardHost">
<VirtualHost Name="*:80"/>     </VirtualHostGroup>     <VirtualHostGroup Name="secureHost">
<VirtualHost Name="*:443"/>     </VirtualHostGroup>     <UriGroup Name="WebSphereURIs">         <Uri
Name="/servlet/snoop/*"/>         <Uri Name="/servlet/snoop"/>         <Uri
Name="/servlet/snoop2/*"/>        <Uri Name="/servlet/snoop2"/>         <Uri Name="/servlet/hello"/>
<Uri Name="/ErrorReporter"/>         <Uri Name="/servlet/*"/>         <Uri Name="/servlet"/>
<Uri Name="*.jsp"/>         <Uri Name="/j_security_check"/>         <Uri Name="/webapp/examples"/>
<Uri Name="/WebSphereSamples"/>         <Uri Name="/WebSphereSamples/SingleSamples"/>         <Uri
Name="/theme"/>     </UriGroup>    <ServerGroup Name="standardServers">         <Server
Name="standardServer1">            <Transport Hostname="localhost" Port="9080" Protocol="http"/>
</Server>     </ServerGroup>    <ServerGroup Name="secureServers">         <Server
Name="secureServer1">            <Transport Hostname="localhost" Port="9443" Protocol="https">
<Property name="keyring" value="product_installation_root\myKeys\plug-inKeys.kdb">
<Property name="stashfile" value="product_installation_root\myKeys\plug-inKeys.sth">
</Transport>         </Server>     </ServerGroup>     <Route VirtualHostGroup="standardHost"
UriGroup="WebSphereURIs" ServerGroup="standardServers"/>     <Route VirtualHostGroup="secureHost"
UriGroup="WebSphereURIs" ServerGroup="secureServers"/>     </Config>
```

⚠ The XML implementation of the plug-in configuration file could changebefore this documentation is updated again. Consult the actual configuration file installed onyour system with your current product version and fix pack level as the most current and correct version of the XML syntax.

Configuring SSL for WebSphere Application Server

The administrative console provides the following access points to SSL settings.

Use the Default SSL Settings to centrally manage SSL settings for resources in the administrative domain. Any of the default settings can be overridden in the settings for an individual resource type -- the transport, ORB, or LDAPs security settings.

- Default SSL Settings

  Open the Security Center and click **Default SSL Configuration**.
- HTTPS SSL settings for the HTTP transport of a Web container

  Edit the transport properties. In particular, select the **Enable SSL** check box.
- ORB SSL settings

  The ORB currently uses the default SSL settings.
- LDAPS SSL settings

  Use the Security Center with LTPA selected as the **Authentication Mechanism**in order to display the LDAP configuration settings. Click **SSL Configuration**.

The above ettings that can be configured through any of these SSL settings is described by the:

- SSL property reference

In the SSL settings dialog, note the **Crypto Token** button for configuring settings for supported cryptographic devices.

Configuring SSL for the application server's HTTPS transport

In order to configure SSL, you must first create an SSL key file. The contents of this file depend on whom you want to allow to communicate *directly* with the application server over the HTTPS port (in other words, you are defining the HTTPS server security policy).

This article presents a restrictive security policy, in which only a well-defined set of clients (the WebSphere plug-ins for the Web server) are allowed to connect to the application server HTTPS port. The following procedure for creating an SSL key file without the default signer certificates follows that restrictive trend.

Step 1: Create an SSL key file without the default signer certificates

1. Start IKeyMan.

   On Windows, start IKeyMan from the WebSphere Application Server entry on the Windows Start menu.
2. Create a new key database file.
   1. Click **Key Database File** and select **New**.
   2. Specify settings:
      - **Key database type**: JKS
      - **File Name**: appServerKeys.jks
      - **Location**: your myKeys directory, such as *"product_installation_root*\myKeys
   3. Click **OK**.
   4. Enter a password (twice for confirmation) and click **OK**.
3. Delete all of the signer certificates.
4. Click **Signer Certificates** and select **Personal Certificates**.
5. Add a new self-signed certificate.
   1. Click **New Self-Signed** to add a self-signed certificate.
   2. Specify settings.
      - **Key Label**: appServerTest
      - **Organization**: IBM
   3. Click **OK**.
6. Extract the certificate from this self-signed certificate so that it can be imported into the plug-in's SSL key file.
   1. Click **Extract Certificate**.
   2. Specify settings:
      - **Data Type**: Base64-encoded ASCII data
      - **Certificate file name**: appServer.arm
      - **Location**: the path to your myKeys directory
   3. Click **OK**.
7. Import the plug-in's certificate.
   1. Click **Personal Certificates** and select **Signer Certificates**.
   2. Click **Add**.
   3. Specify settings:
      - **Data Type**: Base64-encoded ASCII data
      - **Certificate file name**: appServer.arm
      - **Location**: the path to your myKeys directory
   4. Click **OK**.
8. Enter "plug-in" for the label and click **OK**.
9. Click **Key Database File**.
10. Select **Exit**.

Step 2: Add the signer certificate of the application server to the plug-in's SSL key file

1. Start the key management utility.
2. Click the **Key Database File** menu and select **Open**.
3. Select the file *product_installation_root*\myKeys\plug-inKeys.kdb.
4. Enter the associated password and click **OK**.
5. Click **Personal Certificates** and select **Signer Certificates**.
6. Click **Add**.
7. Specify settings.
   - **Data Type**: Base64-encoded ASCII data
   - **Certificate File Name**: appServer.arm
   - **Location**: the path to your myKeys directory.

8. Click **OK**.
9. Click **Key Database File** and select **Exit**.

Step 3: Reference the key file in WebSphere Application Server systems administration

Reference the appropriate SSL key file in the default SSL settings configuration panel or in the HTTPS SSL settings configuration panel. Here, we will use the default SSL settings panel.

1. Start the administrative console.
2. Open the Security Center.
3. Specify settings in the default SSL configuration.
    - **Key File Name**: *product_installation_root*/myKeys/appServer.jks
    - **Key File Password**: *enter your password*
    - **Key File Format**: JKS
    - **Trust File Name**: (empty)
    - **Trust File Password**: (empty)
    - **Client Authentication**: selected
4. Save your changes.

Step 4: Stop the servers and start them again

The configuration is complete. In order to activate the configuration, stop and restart both the Web server and the application server.

6.6.18.1a.8: Selecting users and groups with the Java administrative console

1. Display the page for selecting users and groups by clicking**Console -> Security Center** on the console menu bar, thenselecting the **Role Mapping** or **Administrative Role** tabbed page.

   You might also encounter this tabbed page as part of installingan enterprise application or module.
2. Select a role from the table and click **Select**.
3. Select who should be assigned the role.

   View properties reference for:
   - Select Users/Groups window

   At runtime, the authorization checking will grant access in the following order: **Everyone**, **All authenticated users**, and then **Select users/groups**. If a user or group is in more than one of these roles, the first match will grant access.

   If you opt to select a user or group for the role, then enter a name in the search field or enter a search pattern. For important search usage notes, see:
   - Select Users/Groups window

   After a result of the search is displayed in the **Available Users/Groups** tree view, select one or more users or groups and click **Add**.
4. Click **OK** to commit the role to user or group mapping.
5. Repeat the steps for each role that needs to be mapped.

Related references

Properties for mapping security roles and "run as" roles to users and groups

6.6.18.6: Avoiding known security risks in the runtime environment

Securing the properties files

WebSphere Application Server depends on several configuration filescreated during installation. These files contain password informationand should be protected accordingly. Although the files are protectedto a limited degree during installation, this basic level of protectionis probably not sufficient for your site. You should ensure that thesefiles are protected in compliance with the policies of your site.

The files are found in the bin and properties subdirectories in theWebSphere *<product_installation_root>*.The configuration files include:

- In the bin directory: admin.config
- In the properties directory:
    - sas.server.props
    - sas.client.props
    - sas.server.props.future

ℹ Failure to adequately secure these files can lead to abreach of security in your WebSphere applications.

Securing properties files on Windows NT

To secure the properties files on Windows NT, follow this procedurefor each file:

1. Open the Windows Explorer for a view of the files and directories on the machine.
2. Locate and right-click the file to be protected.
3. On the resulting menu, click Properties.
4. On the resulting dialog, click the Security tab.
5. Click the Permissions button.
6. Remove the Everyone entry.
7. Remove any other users or groups who should *not* be granted access to the file.
8. Add the users who should be allowed to access the file. At minimum, add the identity under which the administrative server runs.

Securing properties files on UNIX systems

This procedure applies only to the ordinary UNIX filesystem. If yoursite uses access-control lists, secure the files by using that mechanism.

For example, if your site's policy dictates that the only user who shouldhave permission to read and write the properties files is the root user,to secure the properties files on a UNIX system follow this procedurefor each file:

1. Go to the directory where the properties files reside.
2. Ensure that the desired user (in this case, root) owns each file and that the owner's permissions are appropriate (for example, rw-).
3. Delete any permissions given to the "group".

4. Delete any permissions given to the "world".

Any site-specific requirements can affect the desired owner, group andcorresponding privileges.

Risks illustrated by example applications

The level of security appropriate to a resource varies with thesensitivity of the resource. Information considered confidentialor secret deserves a higher level of security than public information,and different enterprises will assess the same information differently.Therefore, a security system needs to be able to accommodate a widerange of needs. What is reasonable in one environment can be considereda breach of security in another.

The following describes some user practices and their potential risks.When applicable, it uses components of the example application to illustrate the point.

Invoker Servlet

Purpose: The invoker servlet serves servlets by class name.For example, if you invoke /servlet/com.test.HelloServlet, the invokerwill load the servlet class (if it is in its classpath) and executethe servlet.

Security consideration: By using this servlet, a user can accessany other servlet in the application, without going through the proper channels. For example, if /servlet/testHello is a URI associated with com.test.HelloServlet, and if that URI is protected, user must beauthenticated to invokes /servlet/testHello, but any user can invoke/servlet/com.test.HelloServlet, circumventing the security on the URI.This is a security exposure if you have secured servlets installed inthe system.

Solution: Avoid installing this servlet in your configuration.

An application's error page

Purpose: In case of application errors, users are redirectedto an error page associated with the Web application. This can beany type of Web resource to which customers should be redirectedin case of an error.

Security consideration: This page should be unprotected. Ifit is protected, the server cannot authenticate the user from the context and therefore cannot send the user to the error page whenan error occurs.

Solution: Do not secure these resources.

The web application "examples"

Purpose: This application is available as part of the defaultinstallation.

Security consideration: The servlets available in this application can export sensitive information, for example, theconfiguration of your server.

Solution: The "examples" Web application should not beinstalled in a production environment.

Avoiding other known security risks

This file addresses specific problem areas. As always, periodically check the product Web site Library page for the latest information. See alsothe product Release Notes.

- To avoid a security risk, ensure that the WebSphere Application Server document root and the Web server document root are different. Keep your JSP files in the WebSphere Application Server document root or it will be possible for users to view the source code of the JSP files.

WebSphere Application Server checks browser requests against its list of virtual hosts. If the host header of the request does not match any host on the list, WebSphere Application Server lets the Web server serve the file. Suppose the requested file is a JSP file in the Web server document root -- the JSP file is served as a regular file.

This problem has been noticed in scenarios using Netscape Enterprise Server. Due to the nature of the problem, it is possible that other Web server brands are susceptible.

- **Microsoft Internet Information Server users:**
  To use the Microsoft Internet Information Server with security enabled, in combination with IBM WebSphere Application Server security, you need to:
    - Configure IIS authentication settings to Anonymous.
    - Disable NTLM (Windows NT Challenge/Response) in the Microsoft Management Console
    - Disable Basic Authentication on the Microsoft Management Console

  Look for the setting on the Directory Security tab of the WWW Services properties.

  Problems are common when Internet Information Server NTLM is enabled along with IBM WebSphere Application Server security. The above settings are recommended to avoid problems.
- **Users of Distinguish Names (DN) in LDAP:**
  Make sure you use Distinguished Names (DNs) that your directory service product supports. Although WebSphere Application Server security supports valid LDAP DNs, some directory-service products support only a subset. For example, testing revealed that some directory services do not support all valid LDAP DNs. Specifically, a valid DN of the form OID.9.2.x.y.z=foo was rejected by one or more of the supported directory services.

  Also, directory services vary in how they represent DNs, and DNs are both case- and space-sensitive. In some cases, this leads to situations in which a user enters a valid DN and is authenticated but is still refused access. This problem is often solved by using the Common Name (the short name) rather than the full Distinguished Name.
- **Users of digital certificates with European characters:**
  If you use the iKeyman GUI tool to obtain manage certificates that contain European characters in names, the GUI will not display them. For example, a digital certificate contains the name of the company that owns the certificate and the name of the company that issued the certificate. In the US, there are companies that use symbols instead of letters in their names, like @Home and $mart $hopper. European characters in certificate names will not be displayed by the GUI.

6.6.18.7: Protecting individual application components and methods

Protecting enterprise beans after redeployment

All methods in enterprise beans and Web applications are unprotectedby default.

Security is not automatically updated when changes are made to a bean. Itwill be updated after the old application is stopped, the new application is deployed into the runtime, and the new application is started.

Adding a method to a bean

If you add a method to a bean, you must use the Application AssemblyTool to associate the new method with a role.

Modifying a method on a bean

If you modify a method on a bean, you must use the ApplicationAssembly Tool to ensure that the method still has a role associatedwith it.

Unprotecting resources

All methods in enterprise beans and Web applications are unprotected by default. If you have add a single method-to-role mapping to an enterprise-bean method, the user will be given an option to assign "DenyAllRole" role to all other unprotected methods during application installation. If the unprotected methods are assigned the "DenyAllRole" role, then these methods are protected; nobody is permitted to use them. If the unprotected methods are not assigned the "DenyAllRole" role, these methods are not protected and anyone can access those methods.

Unprotecting an entire application

During application assembly, if you have assigned roles to methods withan application, you have protected those methods. To unprotectthe methods, you can do either of the following:

- Use the Application Assembly Tool to remove the method-to-role mappings for every method in the application
- Assign the Everyone subject to all of the roles in the application, either during application installation or using the **Security Center** after installation

Unprotecting a Method

The only way to unprotect a specific method is to use the ApplicationAssembly Tool to edit the method-to-role mapping. Change the role associatedwith the method to a different role, one that is associated only withthe Everyone subject.

6.6.18.8: Using Microsoft Active Directory as an LDAP Server

To use Miscrosoft Active Directory as the LDAP server for authenticationwith WebSphere Application Server, there are some specific steps you musttake. By default, Microsoft Active Directory does not allowanonymous LDAP queries. To make LDAP queries or browse thedirectory, an LDAP client must bind to the LDAP server usingthe distinguished name (DN) of an account that belongs to theAdministrator group of the Windows system.

To set up Microsoft Active Directory as your LDAP server, followthis procedure:

1. Determine the full DN and password of an account in the Administrators group. For example, if the Active Directory administrator creates an account in the Users folder of the Active Directory Users and Computers Windows NT/2000 control panel and the DNS domain is ibm.com, the resulting DN has the following structure:
   `cn=<adminUsername>, cn=users, dc=ibm, dc=com`
2. Determine the short name and password of any account in the Microsoft Active Directory. This does not have to be the same account as used in the previous step.
3. Use the WebSphere Application Server administrative console to set up the information needed to use Microsoft Active Directory:
   1. Start the administrative server for the domain, if necessary.
   2. Start the administrative console, if necessary.
   3. On the administrative console, click **Console -> Security Center** on the console menu bar.
   4. Select the **Authentication** tabbed page. On it, select Lightweight Third Party Authentication (LTPA) as the authentication mechanism.
   5. Enter the following information in the LDAP settings fields:
      - **Security Server ID**: The short name of the account chosen in 2
      - **Security Server Password**: the password of the account chosen in step 2
      - **Directory Type**: Active Directory
      - **Host**: The DNS name of the machine running Microsoft Active Directory
      - **Base Distinguished Name**: the domain components of the DN of the account chosen in step 1. For example:
        `dc=ibm, dc=com`
      - **Bind Distinguished Name**: the full DN of the account chosen in step 1. For example:
        `cn=<adminUsername>, cn=users, dc=ibm, dc=com`
      - **Bind Password**: the password of the account chosen in step 1
   6. Click **OK** button to save the changes.
   7. Stop and restart the administrative server to make the changes take effect.

6.6.18.9: Specifying authentication options in sas.client.props

You can use the sas.client.props file to direct WebSphere ApplicationServer to authenticate users by prompting or by using a user ID and password set in the properties file. The following steps describe theprocedure:

1. Locate the sas.client.props file. By default, it is located in the properties directory under the *<product_installation_root>* of your WebSphere Application Server installation.
2. Edit the file to set up the authentication procedure:
   - To authenticate by prompting, set the loginSource property to the value "prompt":

     ```
     com.ibm.CORBA.loginSource=prompt
     ```

     Note that when using prompt, a graphical panel is presented for the user for collecting the user ID and password. Pure Java clients must call the JDK API System.exit(0) at the end of the program in order to properly end the Java process. This is because the JDK starts a backward AWT thread that is not killed when the login prompt disappears. If you choose not to use a System.exit(0) call, pressing Ctrl-C ends the process.
   - To authenticate by prompting on the console (stdout), set the loginSource property to the value "stdin". The user is then prompted for user ID and password by using a non-graphical console prompt. This is currently only supported by a pure Java client.
   - To authenticate by the values configured in the file, set the loginSource property to the value "properties" and set the desired values for the loginUserid and loginPassword properties:

     ```
               com.ibm.CORBA.loginSource=properties          com.ibm.CORBA.loginUserid=<user_ID>
     com.ibm.CORBA.loginPassword=<password>
     ```
3. Save the file.

6.6.18.10: The demo keyring

Do *not* use the demo keyring in production systems. It includesa self-signed certificate for testing purposes, and the privatekey for this certificate can be obtained easily, which puts the securityof all certificates stored in the file at risk. This keyringis intended only for testing purposes.For information on obtainingproduction certificates, see Requestingcertificates; for information on creating keyring files,see Tools for managing certificates and keyrings.(The links will only work if you are reading this as part of the InfoCenter that you can obtain fromhttp://www.ibm.com/software/webservers/appserv/infocenter.html).

6.6.18.12: Crytographic token support

To understand how to make WebSphere Application Server (both the runtime and the IKeyMan key management utility) work correctly with any crypto hardware, you should become familiar with the JSSE documentation available from the Application Server product installation:

`product_installation_root``/java/docs/jsse/readme.jsse.ibm.html`

Be sure to unzip the file:

`product_installation_root``/java/docs/jsse/native-support.zip`

to the appropriate location; otherwise, link errors will occur.

Follow the documentation that accompanies your device in order to install your crypto hardware. Installation instructions for IBM crypto hardware devices can be found athttp://www.ibm.com/security/cryptocards/html/library.shtml

The product supports the use of the following cryptographic devices.

These can be used by an SSL client or server:

- IBM 4758-23
- nCipher nForce
- Rainbow Cryptoswift

These can be used by SSL clients:

- IBM Security Kit Smartcard
- GemPlus Smartcards
- Rainbow iKey 1000/2000 (USB "Smartcard" device)
- Eracom CSA800

IBM HTTP Server Version 1.3.19 supports the following cryptographic devices. [Thisinformation is provided for convenience. Consultthe IBM HTTP Server Web site and documentation as the ultimate authority].

Cryptographic devices	Client or server	Interface	Operating system
Rainbow Cryptoswift	Client or server	BSAFE 3.0	Windows NT, Solaris, HP-UX
nCipher nFast	Client or server	BHAPI plugin under under BSAFE 4.0	Windows NT, Solaris
nCipher nForce accelerator mode	Client or server	BHAPI/BSAFE	Windows NT, Solaris
nCipher nForce - key storage mode	Client or server	PKCS11	Windows NT, Solaris, HP-UX, AIX, Linux
IBM4758	Client or server	PKCS11	Windows NT, AIX

⚠ Be sure to check the WebSphere Application Server prerequisites Web sitefor the currently supported version(s) of IBM HTTP Server.

6.6.19: Administering the product messages, logs, and traces (overview)

Messages, logs, and traces provide information about the execution of IBM WebSphere Application Server components, such as the administrative server andclients, application servers, and other WebSphere processes in the environment.

6.6.19.0: Properties for tracing, logging, and messages

Please select a sub-topic to access settings pertaining to:

- The Event Viewer
- The trace administration dialog
- Application server trace

6.6.19.0.1: Event Viewer properties

Key:

Applies to Java administrative console of Advanced Edition Version 4.0

Applies to Web administrative console of Advanced Single Server Edition Version 4.0

Applies to Application Client Resource Configuration Tool

See also the other application server properties.

The event viewer shows FATAL, ERROR, WARNING, and AUDIT events that happened during a certain period of time preceding the current time as defined by the system clock. Associated with the Event viewer there are two secondary windows: Event Viewer and the Event Details.

Event viewer properties

Log Limit

How many serious event records to keep at one time. In otherwords, this specifies the size of the serious event pool.

When the log limit is reached and a new serious event occurs,the record for the least recent event is discarded. A record iscreated for the new event.

Serious Event Pool Interval

How often to update the serious event pool with newlydiscovered serious events.

Serious Event Readback Interval

See the description below.

Show Audit Messages

Whether to consider AUDITs, or informational statements aboutsignificant events, to be serious events. Terminate events are also included. If the check box is not enabled, neither AUDITs nor terminate events will be displayed on the console as serious events.

Show Severe Errors

Whether to consider fatal trace events and error trace events to be serious events. If the check box is notenabled, they will not be displayed as serious events.

Show Warning Messages

Whether to consider WARNINGs to be serious events. If thecheck box is not enabled, WARNINGs will not be displayed as serious events.

Warnings indicate a problem has occurred that must be fixed toprevent a more serious problem.

Event details (read only)

ID

The number of the serious event, with respect to the log limit. For example, the event might be number 4 of the 50 most recent serious events.

Node

The node (physical machine) on which the serious event occurred

Server

The application server on which the serious event occurred

Source

The code file in which the serious event occurred

Thread ID

The ID of the thread in which the serious event occurred

Time

The time at which the serious event occurred

Type

The type of serious event, such as an AUDIT, FATAL, orWARNING

In the Event Viewer, the type column shows Fatal and Errors with the red icon. Warnings are showed with a yellow icon. Audits and Terminates are shown as a green icon. In the Details window, the type is shown as FATAL, ERROR, WARNING, AUDIT, or TERMINATE.

Serious Event Readback Interval

By default, the event viewer displaysevents that happened in the last five (5) minutes. To change the value of this*SeriousEventReadbackInterval*, create a file *product_installation_root*/bin/ejsconsole.properties. Inthe file, set the directive:

```
SeriousEvenReadbackInterval=n
```

where *n* is an integer specifying the number of minutes.

6.6.19.0.2: Trace Administration properties

Key:

Applies to Java administrative console of Advanced Edition Version 4.0

Applies to Web administrative console of Advanced Single Server Edition Version 4.0

Applies to Application Client Resource Configuration Tool

See also the other application server properties.

Ring buffer size

The size of the ring buffer for tracing. The ring buffer size is used to specify the number of events to cache.

The default is 1, meaning 1 multiplied by 1024, or 1024 entries. The actual amount of memory consumed by the ring buffer will be determined by the size of the cached events.

When the buffer is full, the oldest trace record is discarded to make room for theaddition of the newest trace record.

Dump file name

The path to the file into which to record a core dump ofthe application server being traced. The entire ring buffer contents aredumped to the file.

If you specify just the file name, the file will reside in thedefault working directory of the server.

6.6.19.0.3: Server trace properties

These are settings for tracing the internal components of the application server.

Key:

Applies to Java administrative console of Advanced Edition Version 4.0

Applies to Web administrative console of Advanced Single Server Edition Version 4.0

Applies to Application Client Resource Configuration Tool

Application Server

The application server whose internal components you are tracing

Dynamic Properties

Name-value pairs for configuring additional properties beyond those available in the administrative interface

Enable Trace File Overwrite

When a named file is specified for trace output, and overwrite is enabled, the named trace file will be truncated each time the application server starts. When overwrite is not enabled, trace data is appended instead to the trace file.

Click "View..." to view the contents of the trace file. If the server is not running, then "View" will display the trace file from the last time the server ran.

Enable Trace String

Whether the specified trace string should be put into use

Node

The administrative node on which you are performing tracing

Trace File

The file to which to log trace events. Note that serious events are always logged to the standard output stream and to the specified trace file.

For Advanced Edition: On an administrative server, the property is called traceOutput.

Specify one of these values:

- stderr
- stdout
- *filename* (or Specify)

where *filename* is a file of your choice. If you do not specify thefull path to the file (drive letter and directories), the file will be created inthe working directory of the application server, as specified by the propertiesof the application server.

For an application server, the literals `stderr` and`stdout` refer files named stderr.txt and stdout.txt (respectively). For an administrative server (as in the case of Advanced Edition), they actually refer to the standard error and standard out streams, such as an error file or the screen.

It is recommended you trace to stderr or stdout instead of to a file. Tracing to stderror stdout is more efficient than tracing to a specified file.

Trace Host Name

The host name to be used for trace access. It is the host name of the machine on which the application server is running.

Trace Output File

See Trace File

Trace Port

The port provided by the application server for trace access.

For Advanced Single Server Edition: You might need to change the default value if the port is already in use by another application. Specify -1 to use automatic port selection. However, a value of -1 prevents you from performing the runtime tracing tasks, such as sending trace strings to a running server.

Trace Specification or Trace String

Specifies the application server components to trace. On an administrative server (in the case ofAdvanced Edition), the property is called traceString.

For *Advanced Edition* (non-Single Server), when the application server is running, you can click the ellipses button ("...") located next to the **Trace Specification** text field to open a graphical tree interface for specifying trace. You can also specify the the trace administration settings using the dialog.

For *Advanced Single Server Edition*, when the application server is running, you can use a graphical tree viewerto select components to trace. If the server is not running, then you will needto type a trace specification string into a text field, using the correct format.

The valid format is:

```
<comp1>=<type>=[enabled|disabled],...:<comp2>=<type>=[enabled|disabled],...
```

where:

comp

Specifies the component name. The component can be a class, package, or group of classesor packages.

Specify the full name of the component, or use a wildcard ("*") character.When you terminate a component name with a wild card character, the enable|disableaction applies to all components whose names begin with the specified prefix.

For example, if components named "a.b.c.d" and "a.b.c.e" are registered, then specify"a.b.c.d" to trace only the "a.b.c.d" component. Specify "a.b.c.*" to trace both components.

type

Specifies the type of tracing to perform:

- **debug -** Provides information for debugging purposes.
- **entry/exit -** Indicates that a process has entered or exited a method.
- **event -** Indicates that a significant event took place, such as a state change.
- **all -** Conducts all three types of tracing.

enabled|disabled

Specifies whether tracing is enabled or disabled for the component. Type either"enabled" or "disabled."

Here is an example of a valid trace specification or traceString. Thestatement:

```
com.ibm.ejs.container.*=all=enabled:com.ibm.ejs.jts.*=entry=enabled
```

enables all tracing on the container and entry/exit tracing on the JTS.

Trace String

See Trace Specification

View Log File

To view a log file, select the log file, determine which region of the file to view, and click "Refresh." The names of particular log files are as configured elsewhere in the server configuration. Note that it might be extremely slow, or impossible, to load the entire contents of a very large log file.

See also the other application server properties.

6.6.19.1: Administering the product messages, logs, and traces

The Application Server product provides various levels of information, from high-level messages in the console to more detailed serverexecution logs and comprehensive tracing.

The remainder of this article, and associated article in section 6.6.19.1,describe how to use the following access points to configure messages andtracing.

Quick summary of trace access points:

- Message filtering
- **Console** -> **Trace** -> **Enabled** menu option
- **Console** -> **Trace** -> **Settings** menu option
- The Trace Service on the **Services** tabbed page of application server properties
- The **Trace...** menu option when you right-click a running administrative node instance, which displays the trace dialog box

For extensive information about logs, traces, and messages, see the Problem Determination section of the InfoCenter (section 8).

Accessing the trace service configuration

1. Locate the application server properties dialog.
2. When the application server properties are displayed in the properties view, click the **Services** tabbed page.
3. Locate the trace service in the list of services.

General approach to tracing a problem

You should not need to trace IBM WebSphere Application Serverunless directed to do so by IBM support personnel. In case you would like to perform your own tracing, here is a procedure:

1. Determine the activity during which the problem usually occurs.
2. Configure the server to take a trace, using one of the optionsdescribed below: dumping the server trace or tracing a server from startup.
3. Perform the activity during which the problem usually occurs.
4. After the problem occurs, gather the trace information as appropriate. (For example, dump the contents of the ring buffer to a file).
5. Optionally, analyze the trace output and speculate which server classesor components are involved in the problem. Repeat the procedure, thistime narrowing the scope of the trace to just those components, isolating the problem area.

Dump the server trace to a file

Sometimes the administrator might find it useful to take a trace on a server process that is already running. The administrator will use the trace dialog box to select the server components to trace. After the administrator has enabled trace, the trace output will be routed to the in-memory ring buffer. The contents of the ring buffer can then be dumped to a file when the trace is completed.

1. Access the trace service as described previously.
2. In the trace dialog box, specify the classes to trace. Verify the dump file name.
3. Now or later, click the Dump button.

   To isolate the activity you are trying to trace, you might want to delay clicking the Dump button until after the server performs theactivity. See the general trace approachsection above for more details about this strategy.

Trace an administrative server from startup

To trace an administrative server from the time that it starts,modify its properties related to tracing:

1. Open the admin.config file in the bin directory of the WebSphereApplication Server installation root.
2. Set the traceString property to specify the server classes to trace. For valid syntax,see the trace properties help.
3. Set the traceOutput property to specify the file to whichto send the trace data. For valid syntax,see the server trace properties help.
4. Save the file.
5. Start the server (or stop it and start it again) for the new traceparameters to take effect.

Trace an application server from startup

To trace an application server from the time that it starts,modify its properties related to tracing:

1. In the tree view, locate and click the application server todisplay its properties on the right side of the console.
2. Set the trace-related properties. Consult the server trace properties help for syntax.
3. Save the property changes.
4. Start the server (or stop it and start it again) for the new traceparameters to take effect.

Finding trace output

The administrator specifies the location of trace output for various servers. Thetable summarizes where to find trace output:

Type of output	Property that specifies location
Application or administrative server dump	Trace dialog box
Administrative server trace	traceOutput property in admin.config file inbin directory of product installation root
Application server trace	Trace output file property of the application server

6.6.19.1.1: Administering messages with the Java administrative console

Please select a sub-topic.

6.6.19.1.1.1: Filtering messages with the Java administrative console

An administrator can specify the types of messages to be listed in the messages area of the Java console (WebSphere Administrative Console) and limit the message log size. In the messages area,click **Options** to open a dialog for setting message preferences.Change the properties shown in the dialog as needed, and click **OK**.

6.6.19.1.2: Viewing logs and messages

Please select a sub-topic.

6.6.19.1.2.1: Viewing messages with the Java administrative console

The messages area is located in the bottom part of the WebSphere Administrative Console. For the messages area to be visible, the **Show Console Messages** toggle on the console **View** menu must be enabled (the default).

Messages displayed in this area are administrative console messages and administrative server serious event messages.

The **Type** column indicates the level of severity of a message:

1	FATAL or ERROR
2	WARNING
3	AUDIT or information

The **Event Message** column lists console trace or event messages. Resize or scroll through the area as needed to review messages that provide a high-level indication of administrative domain health.

The **Source** column states the name of the file from which a trace or eventmessage was generated.

To view a message that is not fully visible, click the **Details**push button. The dialog that opens shows the full message and also provides additional details on serious events. It shows the type of message and distinguishes between fatal and error messages for events generated from the trace facility. This dialog shows only some of the detailsfor administrative console messages; to see all of the details, look at messages generated from the trace interface.

Note that the serious event messages are also saved in the serious event database table, and are also viewable through the activity log.

6.6.19.1.2.3: Viewing logs

Article 8.3 in the problem determination section provides instructions for viewing andinterpreting logs.

6.6.19.1.3: Updating the trace services of application servers with the Java administrative console

During this task, you will specify settings for the EJB container service ofan existing application server.

1. Locate the EJB container service among the application server services.
2. Click **Edit Properties** to display the properties of the service.
3. Specify values for the trace service properties.
4. When finished, click **OK**.

6.6.20: Administering transactions (overview)

Administrators do not usually need to intervene in server transactions. In fact, an administrator can introduce inconsistencies into server data by forcing atransaction to the wrong outcome.

Usually, it is sufficient simply to monitor the progress of transactions. Theadministrator should become familiar with transaction states and identifiers.

Circumstances that could require intervention

Timeouts associated with transactions usually prevent any one transaction from holding resources at a server for too long.

For example, if two transactions are competing for the same resource(one holds a lock on a resource and the other is requesting that lock, and the lock modesconflict), timeouts will eventually abort one of the transactions: the idle timeout willabort a transaction that is inactive too long, and the operation timeout will abort anactive transaction that is taking too long.

Nevertheless, a transaction can hangindefinitely if, for example, the transaction is prepared but the coordinator isunreachable.

If a hung transaction is interfering with the operation of a server (perhaps holdinglocks that other critical transactions are waiting for), the administrator might need to intervene.

An unprepared transaction can be aborted at any time. Unprepared transaction statesinclude active and preparing.

Aborting an unprepared transaction does not affect theconsistency of data -- the transaction's participants have not yet notified a coordinator oftheir readiness to commit. If the administrator aborts an unprepared transaction, any work that wascompleted on behalf of the transaction is rolled back.

With transactions that have already prepared, any action that the administrator takes is moresignificant. It is best to allow the system to resolve the transaction (allow normalprocessing to take place). However, the administrator will need to force an outcome if a transaction can never be completed otherwise.

6.6.20.0: Transaction properties

Key:

Applies to Java administrative console of Advanced Edition Version 4.0

Applies to Web administrative console of Advanced Single Server Edition Version 4.0

Applies to Application Client Resource Configuration Tool

See also the other application server properties.

Client Inactivity Timeout or Transaction Inactivity Timeout

The number of milliseconds a transaction can remain inactive before it is aborted.

Dynamic Properties

Name-value pairs for configuring additional properties beyond those available in the administrative interface

Enable

Enable the transaction service. If the service is not enabled, all other transactionsettings will be ignored.

Total Tran Lifetime Timeout or Transaction Timeout

The number of seconds to allow a transaction to proceed before aborting it because it is taking too much time.

Transaction Inactivity Timeout

See Client Inactivity Timeout

Transaction Log File

The log file for transaction service logging. If specifying multiple log file names, separate them with commas. For example, specify:
"W:\WebSphere\AppServer\logs\tran1.log,W:\WebSphere4_0\AppServer\logs\tran2.log"

If this property is blank, the server will use in-memory logging. In a production environment, be sure to set a value because transaction rollback capability requires transaction logging.

Transaction Timeout

See Total Tran Lifetime Timeout

6.6.20.1: Administering transactions with the Java administrative console

Use the Java administrative console to administer support for transactions.

Work with resources of this type by locating the application server properties in the tree view:

WebSphere Administrative Domain -> Nodes -> *node_name* ->Application Servers -> *application_server_name*

Click the application server instance to display its properties in the properties view. Click the **Transaction** tab. Do not forget to click **Apply** if you modify the properties.

For second location of transaction settings, with different settings thanthose available in the application server properties:

1. Locate a running application server in the tree view.
2. Right-click it and select **Transactions**.
3. Click **Show Details** to display the properties.

6.6.20.1.1: Viewing transactions and forcing outcomes with the Java administrative console

1. Locate a running application server or node in the tree view.
2. Right-click it and select **Transactions**.
3. In the resulting dialog box, you can view the transactions, forcing outcomesif necessary.

6.6.21: Performing tasks with the Resource Analyzer

This section describes how to start the Resource Analyzer, how to setinstrumentation levels, and how to enable data reporting with the Java VirtualMachine Profiler Interface (JVMPI). It also provides step-by-stepinstructions for monitoring data with the Resource Analyzer. Itcontains the following topics:

- Opening the Resource Analyzer
- Setting the instrumentation level
- Enabling JVMPI data reporting
- Displaying current data
- Viewing and modifying chart data
- Storing and replaying data from a log file
- Changing display settings

A full listing of all counters for all resource categories is shown in Performance data.

Opening the Resource Analyzer

The Resource Analyzer is included with WebSphere Application Server4.0. You can access the Analyzer through the following menuoptions:

- In the WebSphere Advanced Administrative Console menu, click**Tools** --> **Resource Analyzer**. This method isavailable for all platforms.
- On Windows systems, select **Start** --> **Programs**--> **IBM WebSphere** --> **IBM Resource Analyzer4.0**.

If the administrative server is not running, you can start the ResourceAnalyzer from the command line, for example, when you want to run the Analyzerin logging mode to view data from previously logged sessions.

To start the Resource Analyzer from the command line, invoke a scriptprovided in the bin directory where WebSphere Application Server isinstalled.

- On Windows systems, the command is as follows:

  ```
  C:\WebSphere\AppServer\bin\ra.bat
  ```

- On UNIX systems, the command is as follows:

  ```
  # /opt/websphere/appserver/bin/ra.sh
  ```

By default, the Resource Analyzer looks for the administrative server onthe machine where the Analyzer is started. If the Resource Analyzer isrunning on a separate machine, you can start it from the command line,specifying the host and port number. For example, on Windows systems,use the following command:

```
C:\> WebSphere\AppServer\bin\ra.bat  [host_name [port_number]]
```

Due to memory requirements, it is recommended that you run the ResourceAnalyzer on a separate machine from the administrative server.

Setting the instrumentation level

Use the Performance Monitoring Settings dialog box to set instrumentationlevels in the Resource Analyzer. The levels are set separately for eachapplication server. You can access the Performance Monitoring Settingsdialog box from within the Resource Analyzer or from the WebSphere AdvancedAdministrative Console. The access methods are discussed in thefollowing topics:

- Setting the instrumentation level through the Resource Analyzer
- Setting the instrumentation level from the administrative console

In the Performance Monitoring Settings dialog box, perform the followingsteps to set the instrumentation level for a resource category:

1. Click the icon labeled **Performance**, which is shown in theCounter Settings window, to expand all the resource performancecategories.
2. Highlight the desired resource category and choose an instrumentationlevel from the selections on the right (either **None**,**Low**, **Medium**, **High**, or**Maximum**). The dial icon associated with the choseninstrumentation level is added to the resource category under the**Performance** icon.
3. Click **OK** to apply the chosen setting and close the PerformanceMonitoring Settings dialog box.

Choosing an instrumentation level causes the same level to be appliedrecursively to all elements below the selected resource. To ensure thatan instrumentation level is applied to the desired resource, expand the nodeand make sure that only the desired resource is selected before setting thelevel. The instrumentation level for any element below a resource canbe selected and individually set to a different instrumentationsetting.

If the instrumentation level excludes a counter, that counter does notappear as an entry in the tables or charts in the Resource Analyzer.For example, if the instrumentation level is set to low on the AdministrativeConsole, counters for that resource having a impact rating of medium or highare not displayed on the selection list in the Counter Selection panel of theData Monitoring pane.

Setting the level for individual enterprise bean methods

To avoid the overhead cost of monitoring individual remote methods,individual methods in enterprise beans are not displayed in the performancepanel unless the methods level is set to maximum. To display individualmethods and specify their instrumentation levels, set the instrumentationlevel for the methods category to maximum by following the procedure describedin Setting the instrumentation level. Close and reopen the Performance Monitoring Settingsdialog box. Individual methods are now displayed, and you can set themethod for individual methods.

Note that only methods that have been called by an application aredisplayed. If a remote method has not been called since the applicationserver was started, it does not appear in the performance panel.

Setting the instrumentation level through the Resource Analyzer

The instrumentation level for a resource category is set in the PerformanceMonitoring Settings dialog box. To open this dialog box from theResource Analyzer, do one of the following:

- In the Resource Selection pane (the topology tree in the left-handwindow), highlight the resource. On the menu bar, click **Actions--> Monitoring Settings**.
- In the Resource Selection pane, right-click the resource. From thedrop-down menu that is displayed, click **Monitoring Settings.**

Follow the instructions in Setting the instrumentation level to change the instrumentation levels.

Alternatively, clicking a resource with an instrumentation level of nonecauses the Performance Monitoring Settings dialog box to open. To usethis access method, follow these steps:

- Double-click the resource in the Resource Selection pane. If theinstrumentation rating for the resource is set to none, a dialog box isdisplayed. (If the instrumentation level for the selected resource isset to a value other than none, the hierarchy tree for the resource expands toshow its subcomponents.)

- The dialog box reports that performance data monitoring is not enabled forthe resource, and it asks whether you want to set the level now. Choose**Yes** to access the Performance Monitoring Settings dialog box;choose **No** to close the dialog box without enabling performance datamonitoring for the resource; Choose **do not ask again** todisable future displays of this dialog box.

Setting the instrumentation level from the administrative console

The instrumentation level for a resource performance category is changedthrough the Performance Monitoring Settings dialog box. To access thisdialog box from the WebSphere Advanced Administrative Console, perform thefollowing steps:

1. In the left-hand window of the Administrative Console, which shows theresource topology tree, right-click the application server instance.Choose **Properties** from the pop-up menu. The ApplicationServer Properties dialog box is displayed.
2. Click the **Services** tab. A Service window displays a listof all the resources that can be monitored.
3. Select **Performance Monitoring Settings** and click **EditProperties**. The Performance Monitoring Settings dialog boxopens.
4. Follow the instructions given in Setting the instrumentation level to change the instrumentation levels.
5. Click **Apply** to make the settings persistent.

Enabling JVMPI data reporting

The Resource Analyzer leverages a Java Virtual Machine Profiler Interface(JVMPI) to enable a more comprehensive performance analysis. Thisprofiling tool enables the collection of information, such as data aboutgarbage collection, about the Java Virtual Machine (JVM) that runs theapplication server.

JVMPI is a two-way function call interface between the JVM and anin-process profiler agent. The JVM notifies the profiler agent ofvarious events, such as heap allocations and thread starts. Theprofiler agent can activate or inactivate specific event notifications, basedon the needs of the profiler.

JVMPI supports partial profiling by enabling the user to choose which typesof profiling information is to be collected and to select certain subsets ofthe time during which the JVM is active. JVMPI moderately increases theperformance impact.

This functionality is available on the Windows, AIX, and Solarisplatforms.

To enable JVMPI reporting for each individual application server, do thefollowing on the Administrative Console of the WebSphere ApplicationServer:

1. Right-click the application server in the hierarchical tree located in theleft pane of the Administrative Console. A pop-up menu isdisplayed.
2. Choose **Properties** from the menu. The Properties dialogbox opens.
3. Click the **JVM Settings** tab. The JVM Settings sheet isdisplayed.
4. Click **Advanced JVM Settings** on the bottom left of the JVMSettings sheet. The Advanced JVM Settings sheet is displayed.
5. In the **Command line arguments** text box, type the followingcase-sensitive command:

   ```
   -XrunpmiJvmpiProfiler
   ```
6. Click **OK** to exit from the Advanced JVM Settings sheet.
7. Start the application server, or restart the application server if it wasalready running.

8. Refresh the Resource Analyzer.

Displaying current data

The following instructions describe how to display current performance datain the Data Monitoring pane.

- Starting data retrieval
- Stopping data retrieval
- Refreshing data
- Getting current data
- Clearing values from tables and charts
- Resetting counters to zero
- Viewing and modifying chart data

Starting data retrieval

To start data retrieval, do the following:

1. Click one or more resources in the Resource Selection pane. Toselect multiple resources, press and hold the Control key and click eachresource.
2. Click **Actions** --> **Start**. Alternatively,right-click the resource and choose **Start**. Theresource's icon turns blue.

If a new module or instance (servlet or JSP file) is loaded by WebSphereApplication Server after the Resource Analyzer is started, perform thefollowing steps to display performance data for a newly loaded servlet or JSPfile:

- From the Resource Analyzer menu, click **Actions** -->**Refresh**.
- In the Resource Selection pane, right-click the newly loaded resource andchoose **Monitoring Settings** from the drop-down menu. The**Performance Monitoring Settings** dialog box opens.
- In the Performance Monitoring Settings dialog box, highlight the newlyloaded resource and change the instrumentation rating, which by default is setto none, to the desired level.

Only default servlets, for example, the ErrorReporter servlet in theDefault application server, have a the value for the **Load atstartup** option set to **true**. The option is set to**false** for other servlets. To change this setting, followthese steps:

- In the WebSphere Advanced Administrative Console, right-click the resourceand select **Properties** from the drop-down menu.
- In the **Advanced** window of the servlet's Properties sheet,set the **Load at startup** option to **true**. When theapplication server is started, performance data will be collected for theservlet.

Stopping data retrieval

To stop data retrieval, do the following:

1. Click one or more resources in the Resource Selection pane. Toselect multiple resources, press and hold the Control key and click eachresource.
2. Click **Actions** --> **Stop**. Alternatively,right-click the resource and choose **Stop**. Theresource's icon turns red.

If you want to stop data reporting for all resources, click**Actions** --> **Stop All**.

When a running application server is stopped in the administrative console,the Resource Analyzer icon for the server turns from green to red. Inaddition, data reporting for this server automatically stops, and icons forthe server's resources turn red.

Refreshing data

New performance data can become available in either of the followingsituations:

- An administrator uses the console to change the instrumentation level fora resource (from medium to high, for example, or from none to low).
- An administrator uses the console to add a new resource (for example, anenterprise bean or a servlet) to the run time.

In both cases, if the resource in question is already being polled by theResource Analyzer (or the parent of the resource is being polled), the systemis automatically refreshed. That is, if more counters are added for agroup that the Resource Analyzer is already polling, the Resource Analyzerautomatically detects this and adds the counters to the table or chartviews. If the parent of the newly added resource is being polled, thenew resource is detected automatically and added to the Resource Selectiontree. Otherwise, the Resource Selection tree, or parts of it, can berefreshed manually by selecting the appropriate node and clicking the**Refresh** icon (or by right-clicking a resource and choosing**Refresh**).

When an application server is running, the Resource Analyzer's tree isautomatically updated with the server's local structure (its containersand enterprise beans) to reflect changes on the server. However, if astopped server is started *after* the Resource Analyzer starts, amanual refresh operation is required so that the server's structure isaccurately reflected in the Resource Selection tree.

To query the administrative server for new resources or counters, do thefollowing:

1. Click one or more resources in the Resource Selection pane.
2. Click **Actions** --> **Refresh**.Alternatively, right-click the resource and choose **Refresh**.

Note that the refresh operation is a local, not global, operation.It applies only to those resources that are selected. Also, the refreshoperation is recursive; all subordinate (children) resources are alsorefreshed when a selected resource is refreshed.

Getting current data

To obtain the current values for all counters that belong to aresource:

1. Click one or more resources in the Resource Selection pane.
2. Click **Actions** --> **Get Value**.Alternatively, right-click the resource and choose **GetValue**.

The values are displayed independently of the refresh rate. Theresource does not need to be running in order to get the values of thecounters.

Clearing values from tables and charts

After stopping a resource, you can use the **Clear Values**operation to remove remaining data from a table or chart. You can thenbegin populating the table or chart with new data. To clear the valuescurrently displayed, do the following:

1. Click one or more resources in the Resource Selection pane.
2. Click **Actions** --> **Clear Values**.Alternatively, right-click the resource and choose **ClearValues**.

Resetting counters to zero

To reset the start time for calculating aggregate data, do thefollowing:

1. Click one or more resources in the Resource Selection pane.
2. Click **Actions** --> **Reset to Zero**.Alternatively, right-click the resource and choose **Reset toZero**.

The reset operation sets the "clock" used for reporting aggregate data forcounters of the selected performance category. Instead of reportingdata from the time the server was started, reporting now begins from the timeof the reset action. Not all counters can be reset. If you usethe reset operation for a group containing counters which cannot be reset, thereset action has no effect. You can select multiple performance groupsand reset them simultaneously.

Viewing and modifying chart data

The **View Chart** window displays a graph with time as the x axisand the performance value as the y axis.

To view data in a chart, do the following:

1. Click a resource in the Resource Selection pane.
2. Click the **View Chart** tab in the Data Monitoring pane.

The default scaling factor is 1. Negative results are displayed aszero (0). When selected counters are using measurement units that arenot proportionally similar, the scaling factor can be set manually to enable ameaningful graphic display. See Scaling the chart display manually for information on manually scaling the graphic display forthe **View Chart** window.

Scaling the chart display manually

You can manually adjust the scale for counters so that the graphic displaysenable meaningful comparisons between graphs of different counters.Follow these steps:

- In the Counter Selection panel, the bottom panel of the Data Monitoringpane, double-click the **Scale** column for the counter that you wantto modify.
- Type the desired value into the field for that counter's**Scale** value.

The **View Chart** display immediately reflects the change in thescaling factor.

The possible values for the **Scale** field range from 0 to 100 andshow the following relationships:

- A value equal to 1 indicates that the value is the actual value.This is the default setting.
- A value greater than 1 indicates that the variable value has beenamplified by the factor shown. For example, a scale setting of1.5 means that the data points for the variable are amplified to oneand one-half times their actual values.
- Negative results are displayed as zero (0).

This value is reflected only in the **View Chart** window.

Storing and replaying data from a log file

The Resource Analyzer can be used to both store and view data saved inprevious sessions.

- Starting to record data

- Stopping the recording of data
- Replaying a log file
- Changing the speed for replaying a log
- Rewinding the log file

Starting to record data

All data being reported by the Resource Analyzer can be saved in a logfile. The data is written to the log as serialized Java objects or asan XML document. To start recording data, do the following:

1. Click **Logging** --> **On**.
2. In the **Save** dialog box, specify the name, location, and formattype of the log file. The **Files of type** field allows anextension of *.lra (log resource analyzer) for binary files or*.xml for XML format. The XML format provides a flexibility thatenables analysis by using third-party tools.
3. Click **OK**.

Stopping the recording of data

To stop recording data, click **Logging** -->**Off**.

Replaying a log file

Only log files saved in binary format (*.lra) can be replayed byusing the Resource Analyzer. Log files saved in XML format(*.xml) provide the option of analysis using third-party tools.

To replay a log file, do the following:

1. Click **File** --> **Open Log File**.
2. In the **Open** dialog box, locate the name of the file to replayand double-click the file.

By default, the data is replayed at the same rate it was collected (writtento the log). If data was collected every minute, it is displayed everyminute. You can change the speed at which the log is replayed byclicking **Options** --> **Play Speed**. If the datawas collected every minute and the speed factor is set to 60x, then data isdisplayed every second.

While replaying the log, you can choose different groups to view byselecting them in the Resource Selection pane. You can also view thedata in either of the views available in the tabbed Data Monitoringpane.

You can stop and resume the log at any point. However, data cannotbe replayed in reverse.

Changing the speed for replaying a log

The log speed is used to control how fast to replay a log file. Theavailable speeds are 1x, 5x, 20x, and 60x. The 1x speed plays the logat the same speed at which the data was originally collected (written to thelog). The 5x speed plays the log five times faster than the rate atwhich the data was collected, and so on. To change the log speed, dothe following:

1. Click **Options** --> **Play Speed**.
2. Move the cursor to choose the desired speed.

Rewinding the log file

To rewind the log file, click **Actions -->Rewind**. Alternatively, use the **Rewind** icon on thetoolbar.

Changing display settings

You can change several aspects of data monitoring. The followingsections describe the possible changes:

- Changing the View Data As setting
- Changing the speed for replaying a log (discussed in Storing and replaying data from a log file)
- Changing the refresh rate
- Changing the table size

Changing the View Data As setting

The **View Data As** setting determines whether counter values represent absolute values, changes in values, or rates of change. The meanings for these settings differ slightly depending on where you are viewing data. Thechoices follow:

- **Raw Value**. Displays the absolute values.If the counter represents load data (for example, the average number ofconnections in a database pool or the average number of live bean objects),the Resource Analyzer displays the current value followed by theaverage--for example, 18(avg:5).
- **Change in Value**. Displays the change in the current value from aprevious value.
- **Rate of Change**. Displays the ratio*change*/(*T1* - *T2*), where*change* is the change in the current value from a previous value,*T1* is the time when the current value was retrieved and*T2* is the time when the previous value was retrieved.

Changing the refresh rate

By default, the Resource Analyzer retrieves data from the administrativeserver every 10 seconds. To change the rate at which data is retrievedfrom the server, do the following:

1. Click **Options** --> **Set Refresh Rate**.
2. In the **Set Refresh Rate** dialog box, type a positive integerrepresenting the number of seconds. The integer must be 1 orgreater.
3. Click **OK**.

Changing the table size

By default, the View Data window displays 40 rows, correspondingto the values of the last 40 data points retrieved from the administrativeserver. To change the size of the table (number of rows displayed), dothe following:

1. Click **Options** --> **Set Table Size**.
2. In the **Set Table Size** dialog box, type the number of rows todisplay.
3. Click **OK**.

- 6.6.0.12: About the Resource Analyzer
- 6.6.0.12.1: Rating the impact of performance monitoring
- 6.6.0.12.2: Exploring the Resource Analyzer's graphic interface

6.6.21.0: Performance data reported with the Resource Analyzer

The following tables describe the counters for each resourcecategory. Each table includes the following types of information:

- **Name**. This column shows the name of the counter as itappears in the **Counter Selection** panel.
- **Granularity.** This column indicates the unit to whichdata collection is applied for that counter.
- **Impact.** This column identifies the level of performanceimpact--low, medium, or high-- that is expected when the counter isenabled.
- **Data Type.** This column shows the digital representationformat used for expressing the measurable units that are collected by thecounter. The following types are included:
  - **Long**--This is a signed integer.
  - **Stat**--This is basic statistical data that is used tocalculate means, variations, and standard deviation information. TheResource Analyzer uses the following raw data, which is stored on theapplication server: the sum, the count, and the sum of squares (forstandard deviation calculations).
  - **Load**--This value adds a time dimension to calculatetime-weighted statistics. The following raw data, which is stored onthe application server, is used to perform the calculations::
    - The time-weighted sum, which is updated each time the counter is changed
    - The time when the counter is created
    - The time when the counter was last updated
    - The current value of the counter
- For the enterprise beans resource category, **Bean Type** is alsogiven. Values can be entity, stateful session, stateless session, andall.

Counters are available for the following resource categories:

- Counters for enterprise beans
- Counters for database connection pools
- Counters for J2C connectors
- Counters for the JVM run time
- Counters for the JVMPI profiler
- Counters for the servlet session manager
- Counters for ORB and Web container thread pools
- Counters for the transaction manager
- Counters for Web applications

Each resource category has an instrumentation level. Theinstrumentation level determines *which* counters are being collectedfor that category. Each counter has a cost rating (high, medium, orlow), indicating its impact on an application's performance if data iscollected for that counter. If a resource category has aninstrumentation level of high, for example, all counters in that category witha high cost rating *and lower* are collected.

By default, instrumentation levels for each resource category are set tonone. For information about how to set instrumentation levels, see Understanding instrumentation levels for data collection.

Counters for enterprise beans

The available performance data for enterprise beans depends on the type ofbean: entity bean, stateful session bean, or stateless sessionbean. Every enterprise bean home interface has a single module ofperformance data, and all bean instances of that home are reflected in thestatistics for that home. (The instances update the same set ofcounters as those selected in the single module for the performancedata.) An *object pool* is a cache of bean objects.There is a cache for each home interface. Performance data for anobject pool can be used to determine how effective the pool is (how often abean object was available in the pool) and how many resources the poolused. In addition to enterprise bean data, data is collected for everymethod of a bean's remote interface.

The performance data for enterprise beans, object pools, and methods islisted in the following table. The notation of (ms) in the Name columnindicates that the counter value is reported in milliseconds.

Table 1. Counter information for enterprise beans

Name	Granularity	Impact	Bean type	Data type	Description
Num Creates	Per type/ per home	Low	Entity and stateful session	long	Number of create calls.
Num Removes	Per type/ per home	Low	Entity and stateful session	long	Number of remove calls.
Num Activates	Per type/ per home	Low	Entity and stateful session	long	Number of times a bean instance was activated.
Num Passivates	Per type/ per home	Low	Entity and stateful session	long	Number of times a bean instance was passivated.
Num Instantiates	Per type/ per home	Low	All	long	Number of times bean objects were created.
Num Destroys	Per type/ per home	Low	All	long	Average number of times bean objects were destroyed (garbagecollected).
Num Loads	Per type/ per home	Low	Entity	long	Number of times bean data was loaded.
Num Stores	Per type/ per home	Low	Entity	long	Number of times bean data was written to the database.
Concurrent Actives	Per type/ per home	High	Entity and stateful session	load	Average level as a function of time of bean instances of the home thatare in the ready state (active beans). A measure of how busy the serveris.
Concurrent Lives	Per type/ per home	High	All	load	Average level as a function of time of bean objects that exist in the runtime, whether active or pooled (objects that were instantiated but not yetdestroyed). A measure of how many resources the home interface isconsuming.
Total Method Calls	Per type/ per home	Low	All	long	Total number of methods being processed. A measure of how busy theserver is.

Avg Method Rt (ms)	Per type/ per home	Medium	All	stat	Average response time on all methods of the remote interface for thisbean.
Avg CreateTime (ms)	Per persister/ per home	Medium	All	stat	Average method response time for create.
Avg Remove Time (ms)	Per persister/ per home	Medium	All	stat	Average method response time for remove.
Active Methods	Per type/ per home	High	All	load	Average number of invocations being processed concurrently for all themethods.
Gets from Pool	Per home/ object pool	Low	All	long	Number of calls retrieving an object from the pool.
Gets Found	Per home/ object pool	Low	All	long	Number of times a retrieval call found an object available in thepool.
Returns to Pool	Per home/ object pool	Low	All	long	Number of calls returning an object to the pool.
Returns Discarded	Per home/ object pool	Low	All	long	Number of times the returned object was discarded because the pool wasalready full.
Drains from Pool	Per home/ object pool	Low	All	long	Number of times the pool was found idle and an attempt was made to removeidle objects.
Avg Drain Size	Per home/ object pool	Medium	All	load	Average number of objects discarded each time the pool was emptied ofidle objects.
Pool Size	Per home/ object pool	High	All	load	Average number of objects in the pool.

Counters for database connection pools

Each data source (database instance) has a connection pool.Performance data for database connection pools is listed in the followingtable. The notation of (ms) in the Name column indicates that thecounter value is reported in milliseconds.

Table 2. Counter information for database connection pools

Name	Granularity	Impact	Data type	Description
Num Creates	Per connection pool	Low	long	Number of connections created.
Num Destroys	Per connection pool	Low	long	Number of connections released.
Num Returns	Per connection pool	Low	long	Number of connections returned to the pool by applications.
Num Allocates	Per connection pool	Low	long	Number of connections allocated.
Pool Size	Per connection pool	High	load	Average size of the pool (number of connections).
Concurrent Waiters	Per connection pool	High	load	Average number of threads waiting for a connection.

Avg Wait Time (ms)	Per connection pool	Medium	stat	Average time that a client waited to be granted a connection.
Faults	Per connection pool	Low	long	Number of connection pool timeouts.
Percents Used	Per connection pool	High	load	Average percent of the pool in use.
Percents Maxed	Per connection pool	High	load	Average percent of the time that all connections are in use.
PrepStmt Cache Discards	Per connection pool	Low	long	Number of prepared statements discarded from the cache.

Counters for J2C connectors

Performance data for J2C Connectors is listed in the followingtable.

Table 3. Counter information for J2C Connectors

Name	Granularity	Impact	Data type	Description
Num Managed Connections	Per connection factory	Low	long	The number of physical connections in the pool.
Num Connections	Per connection factory	Low	long	The total number of connection handles in use by the applicationserver.

Counters for the JVM run time

The performance data for the Java Virtual Machine (JVM) run time is listedin the following table.

Table 4. Counter information for JVM run-time memory

Name	Granularity	Impact	Data type	Description
Total Memory (bytes)	Per JVM	Low	long	The total amount of memory in the JVM run time
Free Memory (bytes)	Per JVM	Low	long	The amount of free memory in the JVM run time
Used Memory (bytes)	Per JVM	Low	long	The total amount of memory used in the JVM run time.

Counters for the JVMPI profiler

Resource Analyzer leverages a Java Virtual Machine Profiler Interface(JVMPI) to enable a more comprehensive performance analysis. Thisprofiling tool enables the collection of information about the Java VirtualMachine (JVM) that runs the application server. For information on thistool, see Enabling JVMPI data reporting.

Table 5. Counter information for the JVMPI interface

Name	Granularity	Impact	Data type	Description
Num Calls	Per JVM	Low	long	The number of garbage collection calls issued.

Avg Time Between Calls	Per JVM	Low	stat	The average time in seconds between two garbage collection calls.
Avg Duration	Per JVM	High	stat	The average time a garbage collection call is in process.
Num Waiters	Per JVM	Low	long	The number of times a thread waits for a lock.
Avg Wait Time	Per JVM	Low	stat	The average time that a thread waits for a lock.
Num Objects Alloc	Per JVM	Low	long	The number of objects allocated.
Num Objects Freed	Per JVM	Low	long	The number of objects freed.
Num Objects Moved	Per JVM	High	long	The number of objects moved in the heap.
Num Threads Started	Per JVM	Low	long	The number of threads started.
Num Threads Dead	Per JVM	Low	long	The number of threads dead.

Counters for the servlet session manager

Performance data for HTTP sessions is listed in the following table.

Performance data for a servlet is collected only if the servlet is loadedwhen the application server is started. To ensure that data iscollected for a servlet, see Starting data retrieval.

Table 6. Counter information for the servlet session manager

Name	Granularity	Impact	Data type	Description
Created Sessions	Per servlet	Low	long	Number of sessions that were created.
Invalidated Sessions	Per servlet	Low	long	Number of sessions that were invalidated.
Finalized Sessions	Per servlet	Low	load	Number of sessions that were finalized.
Session Lifetime	Per servlet	Medium	stat	Average lifetime of a session.
Session Invalidate Time	Per session manager	Medium	stat	Average time from when a session is invalidated to when it isfinished.
Active Sessions	Per servlet	High	load	Total number of sessions that currently are being accessed byrequests.
Live Sessions	Per servlet	High	load	Total number of valid sessions in the server.

Counters for ORB and Web container thread pools

The performance data for ORB and Web container thread pools is listed inthe following table.

Table 7. Counter information for thread pools

Name	Granularity	Impact	Data type	Description
Thread Creates	Per thread pool	Low	long	Number of threads created.
Thread Destroys	Per thread pool	Low	long	Number of threads destroyed.

Active Threads	Per thread pool	High	load	Average number of active threads in the pool.
Pool Size	Per thread pool	High	load	Average number of threads in the pool.
Percent Maxed	Per thread pool	High	load	Average percent of the time the number of threads in the pool reached orexceeded the desired maximum number.

Counters for the transaction manager

The following performance data is available for containertransactions.

Table 8. Counter information for transactions

Name	Granularity	Impact	Data type	Description
Global Trans Begun	Per transaction manager/server	Low	long	Total number of global transactions begun at the server.
Global Trans Involved	Per transaction manager/server	Low	long	Total number of global transactions involved at the server (begun andimported).
Local Trans Begun	Per transaction manager/server	Low	long	Total number of local transactions begun at the server.
Active Global Trans	Per transaction manager/server	High	load	Average number of concurrently active global transactions.
Active Local Trans	Per transaction manager/server	High	load	Average number of concurrently active local transactions.
Global Trans Duration	Per transaction manager/server	Medium	stat	Average duration of global transactions.
Local Trans Duration	Per transaction manager/server	Medium	stat	Average duration of local transactions.
Global Before Completion Duration	Per transaction manager/server	Medium	stat	Average duration for before_completion for global transactions.
Global Prepare Duration	Per transaction manager/server	Medium	stat	Average duration for prepare for global transactions.
Global Commit Duration	Per transaction manager/server	Medium	stat	Average duration for commit for global transactions.
Local Before Completion Duration	Per transaction manager/server	Medium	stat	Average duration for before_completion for local transactions.
Local Commit Duration	Per transaction manager/server	Medium	stat	Average duration for commit for local transactions.
Num Optimizations	Per transaction manager/server	Low	long	Number of global transactions converted to single phase foroptimization.
Global Trans Committed	Per transaction manager/server	Low	long	Total number of global transactions committed.

Local Trans Committed	Per transaction manager/server	Low	long	Number of local transactions committed.
Global Trans RolledBack	Per transaction manager/server	Low	long	Total number of global transactions rolled back.
Local Trans RolledBack	Per transaction manager/server	Low	long	Number of local transactions rolled back.
Global Trans Timeout	Per transaction manager/server	Low	long	Number of global transactions timed out.
Local Trans Timeout	Per transaction manager/server	Low	long	Number of local transactions timed out.

Counters for Web applications

Performance data for Web applications is shown in the followingtable. Note that the first four counters are simply aggregate valuesfor all of the servlets in the application. The notation of (ms) in theName column indicates that the counter value is reported inmilliseconds.

Table 9. Counter information for Web applications

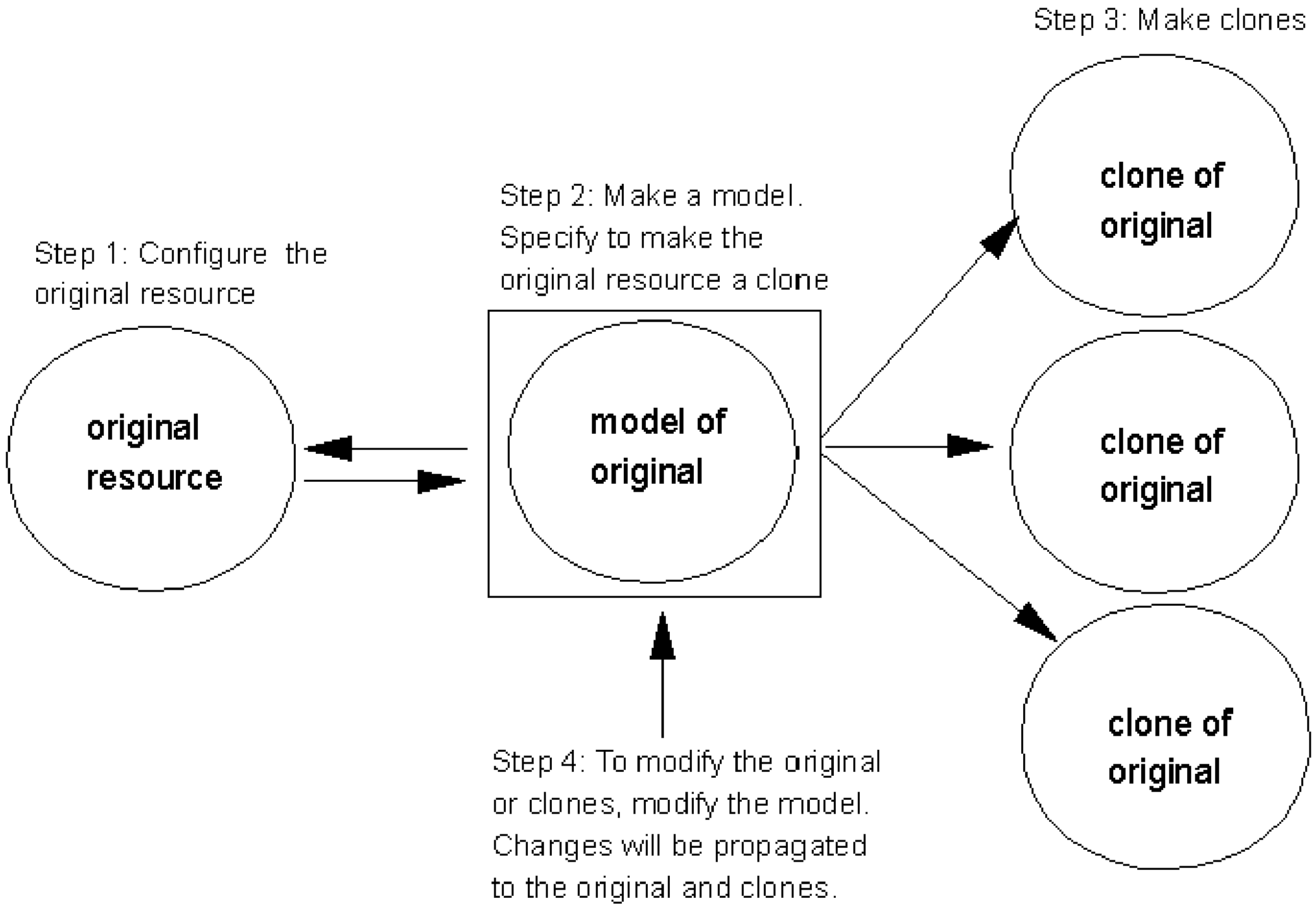
Name	Granularity	Impact	Data type	Description
Num Loaded Servlets	per servlet/JSP file	Low	long	Current number of loaded servlets.
Num Reloads	per servlet/JSP file	Low	load	Number of times a servlet has been reloaded.
Total Requests	per servlet/JSP file	Low	long	Total number of requests for the servlets.
Concurrent Requests	per servlet/JSP file	High	stat	Number of requests that are concurrently processed.
Response Time (ms)	per servlet/JSP file	Medium	long	Response time until the request is finished.
Num Errors	per servlet/JSP file	Low	long	Number of errors or exceptions that have occurred in the servlets.

6.6.21.1: Analyzing resources with the Java administrative console

To launch the Resource Analyzer tool, click**Tools** -> **Resource Analyzer** on the console menu bar.

6.6.22: Administering server groups (overview)

Administrators can manage clones efficiently by managing the server group on which they are based:


Step 3: Make clones
Step 2: Make a model. Specify to make the original resource a clone
Step 1: Configure the original resource
original resource
model of original
clone of original
clone of original
clone of original
Step 4: To modify the original or clones, modify the model. Changes will be propagated to the original and clones.

After cloning an application server, modifying the application server group (model) automatically propagates the same changes to all of the application server clones. It is recommended that the administrator convert the original application server into one of the clones.

🛈 Note that if an application server is used as a base to create a server group, its modules will be taken away from it, and put into server group context. Do not be alarmed if the modules seem to be missing from the original applicationserver!

Properties of server groups and clones

The properties of server groups and clones are of two main types:

- Properties matching the properties of the original application server
- Properties that uniquely identify the server group or clone

The majority of a server group or clone's properties match those ofthe original application server. However, the server group or clone will have a uniquename and location.

Server group and clone modifications

It is important to remember that to perform an administrative action on a clone, such as modifying the clone's properties, the administrator must perform the action on the associated server group instead.

For example, to add or remove a bean from an application server, theadministrator adds or removes the bean from the server group container within the server group application server. With one action, the administrator can add or remove the bean from all clones of the application server.

Some properties must be set at each clone.For example, the port used for transport must be unique for all cloneson a given machine.

6.6.22.0: Server group properties

Key:

Applies to Java administrative console of Advanced Edition Version 4.0

Applies to Web administrative console of Advanced Single Server Edition Version 4.0

Applies to Application Client Resource Configuration Tool

⚠ Some properties should only be modifiedat the server group level, while others should only be modified for individual serversin the server group. See below for a guide.

Server groups have the same properties as application server, with these additions.

Server Group Name

An administrative name for the server group

Workload management selection policy

Specifies the selection method to be used by clients in deciding which server to bind to, of the servers in the cluster. The choices are:

Random

Application server instances are selected randomly from within the group of clones associated with a server group.

Round robin

An application server instance is initially selected at random from an ordered list. Other application server instances are selected from the ordered list in turn, until the initially selected server is selected again. The sequence is repeated. If a particular application server instance is stopped or otherwise unavailable, that instance is skipped (no attempt is made to select it) until it can be verified as being back in service.

Random prefer local

Application server instances on the same host as the client are selected according to the random server selection policy. If no local application server instances are available, server instances on remote hosts are selected according to the random server selection policy.

Round robin prefer local

Application server instances on the same host as the client are selected according to the round-robin server selection policy. If no local application server instances are available, server instances on remote hosts are selected according to the round robin server selection policy.

The random prefer local and round-robin prefer local policies try to select application server instances within the same Java Virtual Machine (JVM) process. For instance, if a client and an application server clone are running in the same JVM process, the local clone is always picked over a clone in a separate JVM process.

Properties at the server group and individual server level

In general, after you have included an application server in a server group, you should only modify properties at the server group level, and *not* modifythe properties of individual servers.

However,there are exceptions, noted with an "X" in the table below. The exceptions are known as 'clone-only' properties. They can be edited for individual clone servers. Clones inherit the value of these properties from the server group when the clone is first created. From that point on, changes to the version of the property for the server group do not effect the individual clones.

DisplayName	Set this property for each server (as opposed to setting it in the server group properties)
General tab	
Application Server Name	X
Environment	X
Working directory	X
Node Startup State	
Maximum startup attempts	
Module visibility	
Advanced tab	
Ping interval	
Ping timeout	
Ping initial timeout	
Process priority	X
Use domain qualified user names	
File tab	
Standard input	X
Standard output	X
Standard error	X
File permissions	
Transaction tab	
Transaction timeout	
Transaction inactivity timeout	
JVM Settings tab	
Initial java heap size:	
Maximum java heap size	
Classpaths	X
System Properties	
Advanced JVM Settings dialog	
Command line arguments	
Enable debug mode	X
Debug string	X
Enable IBM distributed debugger	X
Source path	X
Run HProfs	
Hprof arguments	
Disable JIT	X:
Enable class verbose mode	X
Enable garbage collection verbose mode	X

Enable JNI verbose mode	X
Boot classpath (replace)	X
Boot classpath (prepend)	X
Boot classpath (append)	X
Services tab	
EJB Container service -- General tab	
Passivation directory	X
Cache size	
Cache clean-up interval	
Default datasource	
Web Container service -- General tab	
Minimum thread size	X
Maximum thread size	X
Allow thread allocation beyond maximum	X
Thread inactivity timeout	X
Web Container service -- Transport tab	
HTTP Transports	X
HTTP Transport Properties dialog -- General page	
Transport host	X
Transport port	X
Connection backlog	X
Enable SSL	X
Use global SSL default configuration	X
Key file name	X
Key file password	X
Key file format	X
Trust file name	X
Trust file password	X
Trust file format	X
Enable client authentication	X
Security Level	X
HTTP Transport Properties dialog -- Advanced page	
Enable Crypto Token Support	X
Token type	X
Library file	X
Password	X
Dynamic Properties	X
Maximum keep alives	X
Maximum requests per keep alive	X
Keep alive timeout	X
I/O timeout	X
Web Container service -- Servlet Caching tab	

Enable Servlet Caching	
Cache size	
Default priority	
External cache groups	
External cache groups dialog	
Cache group name	
Members	
-- Adapter Bean Name	
-- Hostname	
Session Manager Service -- General tab	
Enable SSL ID tracking	
Enable cookies	
Enable URL rewriting	
Session Manager Service -- Advanced tab	
Allow overflow	
Integrate with WebSphere security	
Enable protocol switch rewriting	
Invalidation timeout	
Session Manager Service -- Cookies tab	
Cookie domain	
Cookie path	
Restrict exchange of cookies to secure sessions	
Cookie maximum age	
Session Manager Service -- Persistence tab	
Enable persistent sessions	
(values for above?)	
Configure Persistence Tuning	
Write frequency	
Time based:	
Write contents	
Specify session database cleanup schedule for invalidated sessions	
First time of day	
Second time of day	
Session Manager Service -- Database tab	
Data source	
Username	
Password	
DB2 row size	
Table space name	
Use multirow sessions	
Trace Service -- General tab	

Trace specification	X
Trace output file	X
Object Level Trace Service -- General tab	
Object Level Tracing enabled	X
OLT server host	X
OLT server port	X
Performance Monitoring Settings -- General tab	
Enable performance counter monitoring	X
(counter settings)	X
Object Request Broker -- General tab	
Listener port	X
Request timeout	
Locate request timeout	
Connection cache maximum	
Connection cache minimum	
Thread pool size	X
Enable ORB tracing	X
Object Request Broker -- Advanced tab	
RMI remote codebase	
Request retry	
Request retry delay	
Enable HTTP tunneling	
Tunnel agent URL	
External config URL	
Pass by reference	
ORB SSL Configuration Properties - General tab	
Enable SSL	
Use global SSL default configuration	
Key file name	
Key file password	
Key file format	
Trust file name	
Trust file password	
Trust file format	
Enable client authentication	
Security level	
ORB SSL Configuration Properties - Advanced tab	
Enable Crypto Token Support	
Token type	
Library file	
Password	
SSL Properties	

Edit Custom Service dialog - General tab	
Name	
Description	
Classpath	
Classname	
Configuration file URL	
Enabled	
Edit Custom Service dialog - Custom tab	
Custom properties	

6.6.22.1: Administering server groups with the Java administrative console

Use the Java administrative console to administer server groups.

Work with resources of this type by locating them in the tree view:

WebSphere Administrative Domain -> ServerGroups

The instances will be displayed in the details view.

6.6.22.1.1: Specifying server groups with the Java administrative console

A server group is a template for creating additional, nearly identical copiesof an administrative resource and its contents. The administrative resourceis usually an application server.

To create a server group, click **Console -> Wizards -> Create Server Group** from the console menu bar. This leads to the Create Server Group task wizard. Note, this article also contains somesteps you need to do after completing the wizard.

Using the Create Server Group wizard

You use a Create Server Group wizard to define a new server group or todefine one based on an existing server. The wizard enables you to set values for some server group properties. To view or change all properties, use the server group properties dialog.

1. Identifying the server group
2. Specifying services used by the server group
3. Using application servers as clones in the server group
4. Completing the server group wizard
5. Copying the application installation directories to other nodes
6. ⚠ Ensuring each server has a unique port number for its HTTP transport

Identifying your server group

On the Specifying Server Group Properties panel:

1. If you want to define a new server group that is not based on an existing server, specify a server group name. Alternatively, if you want to base your new server group on an existing server, select the existing application server from the drop-down list and specify a server group name.

   Note that you can also create a server group from an existing stand-alone application server by right-clicking on the application server in the console tree view and selecting the **Create Server Group** popup menu option.
2. Specify a workload management selection policy.

   View server group properties help
3. Click **Next**.

Specifying services used by your server group

On the Enabling Services panel is a list of services available to your server group.To view or change the property values for a service, select the service, click **Edit Service Properties**, and change the property valuesas needed. Then, click **Next**.

Using application servers as clones in your server group

On the Creating Application Servers as Clones panel:

1. Name your application server clone. A *clone* is a copy of a server group. A clone can handle a request for the server group. A common use of a clone is to provide distributed function by locating clones on different machines.
2. Specify the node on which to install the application server.
3. Click **Add**.
4. Repeat steps 1 through 3 to create another application server clone.
5. Click **Next**.

   View server group properties help

Completing the server group wizard

The Completing the Create Server Group Wizard panel lists the server group nameand the names of any clones.

If you do not want to change the values specified, click **Finish**. Thewizard will create a new server group, and display a message indicating whether theserver group was successful created. If the wizard encounters an error, a message will display explaining why a server group could not be created.

To change the values specified, click **Back** to return to the appropriatepanel(s), make any needed changes, and then click **Finish** on this panel.

Copying the application installation directories to other nodes

As it is completing, the wizard instructs you to "Copy the install directory of each enterprise application to the $WAS_HOME/installedApps directory of all the other nodes." The following scenario illustrates how to do this step, which is only necessary ifthe server group spans multiple nodes (it could be that all of the servers in the group reside on the same administrative node).

The following steps NEED VERIFICATION and COMPLETION - defect 103733: have called LW on Austin GUI team for explicit instructions

Suppose you have created a server group based on Application_Server_A on Administrative_Node_A. Application_Server_A contains the enterprise applicationcalled My_Application. The server group is comprised of two servers: Application_Server_Aand another Application_Server_B on a remote administrative node called Administrative_Node_B. (All of these names are the logical names by which theresources are displayed and administered in the administrative console tree view).

$WAS_HOME refers to the *product_installation_root*.

In such a scenario, you would do the following:

1. View the product directory structure on Administrative_Node_A (on Windows NT, you can use Windows Explorer).
2. Under the *product_installation_root* directory, locatethe "installedApps" directory.
3. In the installedApps directory, locate the directory named for My_Application. Itshould have a name such as: ?????
4. Copy the My_Application directory structure, including all of its contents and subdirectories.
5. On Administrative_Node_B, locate the installedApps directory, again under the *product_installation_root*.
6. Put the My_Application directory structure, which you copied from Administrative_Node_Ain a previous step, into the installedApps directory on Administrative_Node_B.

Repeat the steps for each administrative node that contains a server in the servergroup. For this scenario involving only Administrative_Node_A and Administrative_Node_B, no repetitions are required.

Ensuring each server has a unique port number for its HTTP transport

An important step remains after you finish the wizard. Each server in a servergroup must have a unique port specified for the HTTP transport of its Web container.Furthermore, the port must have a corresponding virtual host alias.

For each server in the server group:

1. Edit the HTTP transport properties.
2. In the transport properties, specify a unique value forthe Port (or Transport Port) field and save your change. (Typically, the field value will be 9080 until you change it).
3. Follow the instructions in virtual host administrative overviewfor creating an alias corresponding to the port number.

6.6.22.1.2: Editing services of server groups with the Java administrative console

During this task, you will edit the settings of one of the many services thatcomprise an application server.

1. Locate the server group in the tree view.
2. Click it to display its properties in the properties view.
3. Click the **Services** tabbed page. (If editing a custom service, click the **Custom**tabbed page instead).
4. Select the service from the list of services.
5. Click **Edit Properties**. (On the **Custom** tab, it is simply **Edit**).
6. Specify properties for the service. See the properties references for:
    - EJB container service
    - Web container service
    - Session Manager service
    - Trace service
    - Object Level Trace service
    - Performance Monitoring settings
    - ORB service
    - Custom service
7. Click **OK** when finished, to close the service properties window.
8. Click **Apply** on the server group properties view.

6.6.24: Administering application client modules (overview)

Administration application client modules consists of the following.

Use the Application Assembly Tool to:

1. Creating the module
2. Setting deployment descriptor properties
3. Generating code for deployment

Use the Application Client Resource Configuration Tool to set additional configuration properties.

Classpath considerations

An important classpath-related setting to note is the Module Visibility. This application server setting impacts the portability of applications and standalone modules from other WebSphere Application Server versions and editions. If your existing module does not run as-is when you transfer it to Version 4.0, you might need to reassemble an existing module or change the module visibility setting.

See the information on setting classpaths for a full discussion of classpath considerations.

6.6.24.0: Application client module properties

These are set using the Application Assembly Tool. Refer to the "Assembly properties for client modules" property reference to set values for these properties.

6.6.24.0.a: Assembly properties for application client modules

File name (Required, String)

Specifies the file name of the application client module, relative to thetop level of the EAR file. If this is a stand-alone module, the filename is the full pathname of the archive.

Alternative DD

Specifies the file name for an alternative deployment descriptor file touse instead of the original deployment descriptor file in the module'sJAR file. This file is the postassembly version of the deploymentdescriptor file. (The original deployment descriptor file can be editedto resolve dependencies and security information. Directing the use ofthe alternative deployment descriptor allows you to keep the originaldeployment descriptor file intact). The value of the Alternative DDproperty must be the full path name of the deployment descriptor file relativeto the module's root directory. By convention, the file is in theALT-INF directory. If this property is not specified, the deploymentdescriptor file is read directly from the module's JAR file.

Classpath

Specifies the full class path containing the dependent code that is notcontained in the application client JAR file. Specify the valuesrelative to the root of the EAR file and separate the values withspaces. Absolute values that reference files or directories on the harddrive are ignored. To specify classes that are not in JAR files but arein the root of the EAR file, use a period and forward slash(./). Consider the following example directory structure inwhich the file myapp.ear contains an application client JAR file namedclient.jar. Additional classes reside in class1.jar andclass2.zip. A class named xyz.class is not packaged in aJAR file but is in the root of the EAR file.

```
myapp.ear/client.jarmyapp.ear/class1.jarmyapp.ear/class2.zipmyapp.ear/xyz.class
```

Specify `class1.jar class2.zip ./` as thevalue of the Classpath property. (Name only the directory for.class files.)

Display name (Required, String)

Specifies a short name that is intended to be displayed by GUIs.

Small icon

Specifies a JPEG or GIF file containing a small image (16x16pixels). The image is used as an icon to represent the applicationclient in a GUI.

Large icon

Specifies a JPEG or GIF file containing a large image (32x32pixels). The image is used as an icon to represent the applicationclient in a GUI.

Description

Contains text describing the application client.

Main class (Required, String)

Specifies the full path name of the main class for this applicationclient.

6.6.24.5: Administering application clients with Application Assembly Tool

An application client is a standalone Java program (in contrast to a Webbrowser-based program). An application client module is used toassemble the files that make up the application client into a singleunit. The Application Assembly Tool is used to create and edit modules,verify the archives, and generate deployment code. See the relatedtopics for links to concepts, instructions for creating an application clientmodule, and field help.

If the application client requires local resources, you must also use theApplication Client Resource Configuration Tool. This tool allows you todefine references to local resources other than enterprise beans (such as JDBCand JMS resources) on the machine where the application client resides.

6.6.24.5.1: Creating an application client

Application client modules can be created by using property dialog boxes orby using a wizard.

- Using the property dialog boxes
- Using the Create Application Client wizard

Using the property dialog boxes

Creating a new application client consists of specifying the files thatmake up the client and then adding assembly properties. To create a newapplication client:

1. Click **File** ->**New** -> **ApplicationClient**. The navigation pane displays a hierarchical structureused to build the contents of the module. The icons represent thecomponents, assembly properties, and files for the module. A propertydialog box containing general information about the application client isdisplayed in the property pane.
2. By default, the application client JAR file and the display name are thesame. It is recommended that you change the display name in theproperty pane.
3. By default, the temporary location of the application client JAR file isinstallation_directory`/bin`. You must specify a newfile name and location by clicking **File**->**Save**.
4. Enter the main class file name (required). Click **Browse**to locate the class file. By default, the root directory or archive isthe current archive. If needed, browse the file system for thedirectory or archive where class files reside. Click**Select**. The archive's file structure is displayed inthe window. Expand the structure and locate the files that youneed. Select the file and click **OK**.
5. Enter values for other properties as needed. View the help for 6.6.24.0.a: Assembly properties for application client modules. Click **OK**.
6. Define assembly properties for the application client.
    - Right-click the EJB References icon and click **New**. Aproperty dialog box for EJB References is displayed. View the help for 6.6.43.0.1: Assembly properties for EJB references. Enter values for each property and then click**OK**. Repeat to specify multiple EJB references. Thetop portion of the property pane lists each reference. Select thereference to view its corresponding property dialog box.
    - Right-click the Resource References and click **New**. Aproperty dialog box for Resource References is displayed. View the helpfor 6.6.43.0.2 Assembly properties for resource references. Enter values for each property and then click**OK**. Repeat to specify multiple resource references.
    - Right-click the Environment Entries and click **New.** Aproperty dialog box for Environment Entries is displayed. View the helpfor 6.6.34.0.a: Assembly properties for environment entries. Enter values for each property and then click**OK**. Repeat to define multiple environment variables.
7. Add files for the application client. In the navigation pane,right-click the Files icon and then choose **Add Files**. Usethe file browser to locate files to add. First, browse for the rootdirectory or archive where the files are located and click**Select**. If you are adding an entire archive, select thedirectory that contains the archive. The directory structure isdisplayed in the left pane. Browse the directory structure. Fromthe right pane, select one or more files to be added and click**Add**. If you select a directory and click **Add**, allfiles in the directory, including the directory, are added. Relativepath names are maintained. The selected files are displayed in theSelected Files window. Click **OK**. The file names,extensions, modification dates, sizes, and path names are displayed in theproperty pane.

8. Review the contents of the module and make any desired changes.
9. Click **File**->**Save** to save the module.

Using the Create Application Client wizard

Use this wizard to create an application client JAR file. Duringcreation, you specify the files that make up the application client.You also specify information such as references to other (external) enterprisebeans needed by the client.

Before you start the wizard, you must have the class files and other filesbelonging to the application client. When the wizard is completed, yourapplication client JAR file resides in the location that you specified.

To create an application client, click the **Wizards** icon on thetoolbar, and then click **Application Client**. Follow theinstructions on each panel.

- Specifying application client module properties
- Adding files
- Specifying additional application client module properties
- Choosing application client module icons
- Adding EJB references
- Adding resource references
- Adding environment entries
- Setting additional properties and saving the archive

Specifying application client module properties

On the **Specifying Application Client Module Properties**panel:

1. Indicate the application to which this module is to be added. If aparent application is not indicated, the module is created as a stand-aloneapplication.
2. Specify a display name for the application client (required). Thedisplay name is used to identify your application in the Application AssemblyTool and can be used by other tools.
3. Specify a file name for the application client (required). The filename specifies a location on your system for the JAR file to becreated.
4. Provide a short description of the application client (optional).
5. Click **Next**.

Adding files

On the **Adding Files** panel, specify the files that are to beassembled for the application client. To add or remove files:

1. Click **Add**. Use the file browser to locate files toadd. First, browse for the root directory or archive where the filesare located and click **Select**. If you are adding an entirearchive, select the directory that contains the archive. The directorystructure is displayed in the left pane. Browse the directorystructure. From the right pane, select one or more files to be addedand click **Add**. If you select a directory and click**Add**, all files in the directory, including the directory, areadded. Relative path names are maintained. The selected filesare displayed in the Selected Files window. Click **OK**.The files are displayed in a table on the wizard panel.

2. If you want to remove a file, select the file in the table and then click **Remove**.
3. Continue to add or remove files until you have the correct set of files.
4. Click **Next**.

Specifying additional application client module properties

On the **Specifying Additional Application Client Module Properties** panel:

1. Specify the classpath and main class for the application client. View the help for 6.6.24.0.a: Assembly properties for application client modules.
2. Click **Next**.

Choosing application client module icons

On the **Choosing Application Client Module Icons** panel, specify icons for your module.

1. Specify the full path name of a file containing a small icon, or click **Browse** to locate and select the file. The icon must be a GIF or JPEG image 16x16 pixels in size.
2. Specify a full path name of a file containing a large icon, or click **Browse** to locate and select the file. The icon must be a GIF or JPEG image 32x32 pixels in size.
3. Click **Next**.

Adding EJB references

On the **Adding EJB References** panel, specify the enterprise beans required by the application client.

1. Click **Add**. Enter the name of the enterprise bean to be used by the application client, the names of the bean's home and remote interfaces, and the bean type (required). View the help for 6.6.43.0.1: Assembly properties for EJB references. The EJB reference is displayed in a table on the wizard panel.
2. If the referenced bean is located in the module being created (or in the encompassing application), enter a name in the Link field (optional). If the bean is external to the module, leave the Link field blank. You can specify JNDI binding information later (by using the property dialog boxes or by using the administrative console).
3. To remove a reference, select the entry in the table and then click **Remove**.
4. Continue to add and remove references as needed.
5. Click **Next**.

Adding resource references

On the **Adding Resource References** panel, enter references to connection factory objects for resource managers.

- Click **Add**. Enter values for the name, type, and authorization mode of the resource reference, then click **OK**. View the help for 6.6.43.0.2 Assembly properties for resource references. The resource reference information is displayed in a table.
- To remove a reference, select the reference in the table and click **Remove**.
- Continue adding or removing references as needed.
- Click **Next**.

Adding environment entries

On the **Adding Environment Entries** panel, add environment entriesfor the application client.

- Click **Add**. Enter the name and type of the entry, thenclick **OK**. View the help for 6.6.34.0.a: Assembly properties for environment entries.
- To remove an environment entry, select the entry in the table and thenclick **Remove**.
- Continue adding or removing environment entries as needed.
- Click **Next**.

Setting additional properties and saving the archive

Click **Finish** to complete the wizard. To change settingsfor properties, click **Back** to return to the appropriatepanel. Make any needed changes, and then click**Finish**.

After you click **Finish**, the contents of the archive aredisplayed in the main window. You can continue adding or modifyingproperties as needed. For example, you can add bindinginformation. When you are finished editing the archive, click**File**->**Save**. Specify a name for the archive andclick **Save**.

6.6.26: Administering application server process definitions (overview)

The process definition is an *Advanced Single Server Edition* concept thatencompasses the runtime attributes of the application server process, such the working directory,startup attempts, file permissions, and process priority on the operating system. It is a sub-component of the application server.

In *Advanced Edition* (non-Single Server), the "process definition" is not part of the formal model, but the application server has many of the same properties. For this reason, the property reference for process definitions is shared among*Advanced Edition* and *Advanced Single Server Edition*.

For the many other application server properties, see the application server property reference.

6.6.26.0: Process definition properties

Key:

Applies to Java administrative console of Advanced Edition Version 4.0

Applies to Web administrative console of Advanced Single Server Edition Version 4.0

Applies to Application Client Resource Configuration Tool

See also the other application server properties.

Environment

Name-value pairs for configuring environment variable values for the running process

Executable Name

The executable name of the process

File Permissions

Read, Write, and Execute permissions for the Owner, User, and Group security roles

Group ID

The name of the operating system user under which to run the server

Note that the operating system user must exist on the machine where the server is to run before the server is started. This user must be assigned the necessary operating system privileges for performing operations such as creating output files on the local file system.

JVM Settings

Java virtual machine configuration settings for the application server's process

Maximum Startup Attempts

The maximum number of times to attempt to start the application server before giving up

Ping Initial Timeout

The maximum time in seconds for the server to finish initialization.After this time elapses, the administrative server attempts to restart the server.

Ping Interval

The frequency of communication attempts between the parent process, such as a process manager, and the process it is trying to spawn. Adjust this value based on your requirements for restarting failed servers. Decreasing the value detects failures sooner; increasing the value reduces the frequency of pings, reducing system overhead.

Ping Timeout

When a parent process is spawning a child process, such as when a process manager spawns a server, the parent process pings the child process to see whether the child was spawned successfully. This value specifies the number of milliseconds that the parent process should wait (after pinging the child process) before assuming that the child process failed.

Process Priority

Operating system priority for the process. Only root users can change this value.

Run As User

Runs the process as a specific user

Run As Group

Runs the process as a member of a specific group

Standard Input

The file from which the standard input stream will be read

Standard Output **or Stdout File Name**

The file to which the standard output stream will be directed. The file name can include a symbolic path name defined in the path map entries.

For the Advanced Single Server Edition: Click "View..." to view the contents of the stdout file.

Standard Error **or Stderr File Name**

The file to which the standard error stream will be directed. The file name can include a symbolic path name defined in the path map entries.

For the Advanced Single Server Edition: Click "View..." to view the contents of the stderr file.

Stderr File Name

See Standard Error

Stdout File Name

See Standard Output

Umask

The user mask that the process runs under (the file-mode permission mask)

User ID

The name of the operating system user under which to run the server

Note that the operating system user must exist on the machine where the server is to run before the server is started. This user must be assigned the necessary operating system privileges for performing operations such as creating output files on the local file system.

Working directory

The local directory in which to run the server

This directory is used to determine the locations of input and output files with relative path names.

After you start a server, it is recommended that you do not change the server's working directory.

Passivated enterprise beans are placed in the current working directory of the application server on which the beans are running. Make sure the working directory is a known directory under the root directory of the Application Server product.

6.6.28: Administering Object Level Trace and Distributed Debugger support (overview)

Complimentary copies of IBM Object Level Trace and IBM Distributed Debugger software are provided for tracing and debugging the applications running on the application server.

See the related topics for instructions for enabling and configuring OLT and Debugger support through the WebSphere Application Server administrative console.

See also the following OLT and Debugger documentation, included with the InfoCenter:

- IBM Object Level Trace documentation
- IBM Distributed Debugger documentation

6.6.28.0: Object Level Trace and Distributed Debugger properties

Key:

Applies to Java administrative console of Advanced Edition Version 4.0

Applies to Web administrative console of Advanced Single Server Edition Version 4.0

Applies to Application Client Resource Configuration Tool

Dynamic Properties

Name-value pairs for configuring properties beyond those available in the administrative interface

Enable IBM Distributed Debugger

Whether to enable the IBM Distributed Debugger support. If you do not, all other settings related to the Distributed Debugger will be ignored.

Enabing IBM Distributed Debugger makes the execution of your application enabled for the debugger.With this selection, you can debug your application in standalone mode, or through Object Level Trace.

This is a necessary step for using the Object Level Trace to trace code running on the application server. Note, you must also enable Object Level Trace.

This field and the **Enable Debug Mode** field in JVM properties cannot be enabled at the same time.

See the InfoCenter documentation for more information about using OLT and Debugger.

Enable Object Level Trace or Object Level Trace Enabled

Whether to enable support for IBM Object Level Trace. If you enable OLT, also enable the Distributed Debugger.

This is a necessary step for using the Object Level Trace to trace code running on the application server. You use the JVM Settings properties to configure debugging support. See the InfoCenter documentation for more information about using OLT and Debugger.

Host Name or OLT Server Host

The fully qualified host name of the machine on which the OLT trace viewer is located. Trace results will be directed to the machine.

Object Level Trace Enabled

See Enable Object Level Trace

OLT Server Host

See Host Name

OLT Server Port or Port

The port number corresponding to the OLT Host Name. The default is 2102.

Port

See OLT Server Port

Source Path

The root directory from which to obtain the source code for debugging purposes. If the directoryis already set in your CLASSPATH environment variable, then you do not need to set it here.

6.6.28.1: Administering OLT services of application servers with the Java administrative console

Use the Java administrative console to administer Object Level Trace and IBM DistributedDebugger support.

Follow this procedure to enable and configure OLT and Debugger support in the serverruntime:

1. Open the application server properties dialog.
2. When the application server properties are displayed in the properties view, click the **JVM Settings** tabbed page.
3. Select the **Advanced JVM Settings**.
4. Use the JVM properties to:
   - Enable the IBM Distributed Debugger.
   - Specify the "source path" if you know that the source files of your application reside in a different directory than your class files. The source path is the list of directories that contain the source files of your application.
5. Click **OK** to save the changes.
6. Also in the application server properties, click the **Services** tabbed page.
7. Select the **Object Level Trace** service in the list of services.
8. Click **Edit** to edit the service.
9. Configure the OLT service properties, including:
   - Enable Object Level Tracing.
   - Provide the hostname of the machine when the OLT and debugger user interface will be running.
   - Set the OLT server port to the default value of 2102.
10. Click **OK** to save the changes.
11. Back on the application server properties dialog, click **Apply**.
12. Start the application server.

6.6.28.1.1: Configuring the Object Level Trace services of application servers with the Java administrative console

During this task, you will specify settings for the Object Level Trace (OLT) service ofan existing application server.

1. Locate the Object Level Trace service in the application server properties.
2. Click **Edit Properties** to display the OLT service properties dialog.
3. Specify values for the OLT service properties.
4. When finished, click **OK**.
5. Click **Apply** on the application server properties view.

6.6.30: Administering Object Request Brokers (overview)

The WebSphere Application Server uses the ORB to manage communication between client applications and server applications, as well as, communication between WebSphere components, such as the application server and administrative server . During WebSphere Application Server installation, default values are established, called ORB properties, which are used when WebSphere is started and the ORB is initialized. These properties control the run-time behavior of the ORB and can also affect the behavior of WebSphere components which are tightly integrated with the ORB, such as security.

It might be necessary to modify some of the ORB properties under certain circumstances. For example, it may be necessary to modify the default timeout delay the ORB uses when waiting for a response to a request, or to modify the maximum number of active connections the ORB caches for better performance, and so on.

In every request/response exchange, there is a client-side ORB and a server-side ORB, depending on which side initiated the exchange. It is important that the ORB properties be set for either the client-side or server-side as necessary because each ORB is, in general, independent of the other.

Do not use multiple ORB instances

IBM WebSphere Application Server security does not support the programming model of using multiple ORB instances.

To obtain its ORB instance, every application should always make calls to:

```
com.ibm.ejs.oa.EJSORB.getORBinstance()
```

After the ORB instance has been established with a process, the process should not change ORB-related properties because property changes will trigger a new ORB instance to be created. A multiple ORB scenario will occur, which is not supported.

Finding information about the Request Interceptor interface

For information about the Request Interceptor Interface of the IBM Java ORB, see the IBM Web site:

`http://www.ibm.com/software/webservers/appserv/request_interceptors.html`

6.6.30.0: Object Request Broker properties for Advanced Edition

These field descriptions apply strictly to the Advanced Edition. They do not applyto Advanced Single Server Edition.

Configure SSL

See the Secure Socket Layer property help

Connection Cache Maximum

The maximum amount of connections in use kept in the connection cache table at any one time.

The default value is 240, with a range from 0 to 256.

Connection Cache Minimum

The number of connections that must be in the connectioncache in order for the ORB to clean up any connections that are not busy.

For example, if the default value is 100, then when the cache size reaches 101, the ORB will beginshutting any connections that are open but not busy. It will stop when there are 100 or fewer connections in the cache.

The default value is 100, with a range from 0 to 255.

Enable HTTP Tunneling

Controls how the client ORB attempts to use HTTP tunneling. The values of this optional property can be:

- **always** - use HTTP tunneling immediately, without trying TCP connections first
- **never** - disable HTTP tunneling. If a TCP connection fails, a CORBA system exception COMM_FAILURE is thrown
- **whenrequired** - use HTTP tunneling if TCP connections fail

Enable ORB Tracing

Whether to enable the ORB Comm Trace

External Config URL

The URL of an ORB configuration file

Listener Port

The port on which this server will listen for incoming requests. The default is the next available system assigned port number. The range for this optional property is from 0 to 2147483647 (maximum Java Int).

Locate Request Timeout

The number of seconds to wait before timing out on a LocateRequest message. The default value of this optional property is 180. The range is from 0 to 2147483647 (maximum Java Int).

Pass by Reference

When parameters are passed locally, they are passed "by value" using the standard IIOP copy semantics. Use this property to select "pass by reference" instead. Depending on your application design, using pass by reference might introduce side effects.

Request Retry

The number of times the ORB will attempt to send an indirect-IOR request in case a server failed. If a server has been determined to have failed, the ORB will give the Location Service Daemon an opportunity to issue a new IOR. The default value of this optional property is 1. The range is from 0 to 2147483647 (maximum Java Int).

Request Retry Delay

The amount of delay time between request retries. The default value of this optional property is 0. The range is from 0 to 2147483647 (maximum Java Int).

Request Timeout

The number of seconds to wait before timing out on a request message. The default value of this optional property is 180. The range is from 0 to 2147483647 (maximum Java Int).

RMI Remote Codebase

The URL locations to download classes that are published by this JVM.

Thread Pool Size

Specifies the starting size of the thread pool for the application server.

Each application server has its own thread pool or cache from which it uses threads to process remote method invocations.

The size of a server's thread pool varies throughout the server's lifetime. Threads are created when needed and destroyed when there are too many idle threads.

Server Socket Queue Depth, using com.ibm.CORBA.ServerSocketQueueDepth=*value*

The property can be specified to change the queue depth of a server.

The field is not available from the property sheet for ORB administration,but is included here for completeness of the ORB property reference. Instead, for an application server,the argument can be specified using the command line arguments ofthe application server JVM properties.

For *Advanced Edition* (non-Single Server): Use the administrative configuration file settings for the administrative server,as described in administrative server configuration file properties.

Tunnel Agent URL

The URL of the servlet used to support HTTP tunneling. It must be a properly formed URL, such as "http://w3.mycorp.com:81/servlet/com.ibm.CORBA.services.IIOPTunnelServlet" or, for applets, "http://*applethost*:*port*/servlet/com.ibm.CORBA.services.IIOPTunnelServlet." There is no default value. This field is required if com.ibm.CORBA.ForceTunnel is set.

6.6.30.1: Administering the ORB services of application servers with the Java administrative console

Use the Java administrative console to administer Object Request Brokers.

Work with resources of this type by locating them in the application server properties:

1. Locate the application server instance containing the service.
2. When the application server properties are displayed in the properties view, click the **Services** tabbed page.
3. Locate the ORB service in the list of services.

6.6.30.1.1: Configuring the ORB services of application servers with the Java administrative console

During this task, you will specify settings for the Object Level Trace (OLT) service ofan existing application server.

1. Locate the ORB service in the application server properties.
2. Click **Edit Properties** to display the ORB service properties dialog.
3. Specify values for the OLT service properties.
4. When finished, click **OK**.
5. Click **Apply** on the application server properties view.

6.6.30.5: Setting the ORB timeout value

If the network goes down when a Java application is attempting to contact a server, it can take too long for the ORB in the Java application to get an exception. This causes the Web browser originating the request to time out before the Java application can respond to the connection failure (network down). Setting the ORB timeout value to be less than the browser timeout value will allow the Javaapplication to handle the exception before the browser times out. This providesthe opportunity for a more graceful, user-friendly response to down servers.

The setting should onlybe used if you cannot set or adjust a timeout at the JDK level. At the time of this writing, the JDK does not have such a setting. Also, you should be fairlysure of the problem before attempting this solution. Note the considerations.

Considerations for setting the ORB timeout

It is recommended that you do not adjust the ORB timeout unless experiencing a problem, because configuring a valuethat is inappropriate for the environment can create a problem. If you set the value, experimentation will be needed to find the correct value for the particular environment. Configuringan incorrect value can produce results worse than the original problem.

That said, if Web clients accessing Java applications running in the WebSphere environment are consistently experiencingproblems with their requests, and the problem cannottraced to other sources and addressed through other solutions, consider setting an ORB timeout value and adjusting it for your environment.

Web browsers vary in their language forindicating that they have timed out. Usually, the problem is announced as a "connection failure" or "no path to server" message.

Tuning the ORB timeout

When tuning this parameter, aim to set the ORB timeout value to *less than* the time after which a Web client eventually times out. Because it can be difficult to tell how long Web clients will wait before timing out, setting the ORB timeout requiressome experimentation. Another difficulty is that the idealtesting environment will feature some simulated networkfailures for testing the proposed setting value.

Empirical results from limited testing indicate that30 seconds is a reasonable starting value. Mainly, you needto ensure that the setting is not too low. To fine tune thesetting, find a value that is not too low. Then graduallydecrease the setting until reaching the threshhold at whichthe value becomes too low. Set the value a little abovethe threshhold.

When the ORB timeout is set too low, the symptom is numerous CORBA 'NO_RESPONSE' exceptions, which occur evenfor some requests that should have been valid. If requests that should have been successful (for example, the server is not down) are being lost or refused, then the value is likely too low.

How to set the ORB timeout

To set the ORB timeout, add a command line argument to theJava application that the Web clients are trying to access. TheJava application could be your own application, or a Javaprocess internal to WebSphere Application Server, such as anapplication server (and its related servlet engine). It could be the WebSphere thin servlet redirector.

On the Java command line for the application, specify:

```
-Dcom.ibm.CORBA.requestTimeout=30
```

where 30 is a value that you think will be appropriate forthe environment.

6.6.31: Monitoring and tuning performance (overview)

IBM WebSphere Application Server provides the following for monitoring andtuning the performance of applications installed in the server runtime:

- The Resource Analyzer, an interface for displaying and analyzing data collected by application servers:
    - Introduction to the Resource Analyzer
    - How to operate the Resource Analzyer
- A set of performance monitoring properties for collecting the data used by the Resource Analyzer
- A Performance Tuner wizard that helps you tune some key settings to improve performance

6.6.31.0: Performance monitor properties

Key:

Applies to Java administrative console of Advanced Edition Version 4.0

Applies to Web administrative console of Advanced Single Server Edition Version 4.0

Applies to Application Client Resource Configuration Tool

See also the other application server properties.

Enable

Whether to enable performance monitoring

Specification

A specification string or tree structure for enabling performance monitoring for components

6.6.31.1: Monitoring and tuning performance with the Java administrative console

Use the Java administrative console to monitor and tune the performanceof applications and other components.

Locate the performance monitoring settings in the application server properties.

1. Locate the application server instance.
2. When the application server properties are displayed in the properties view, click the **Services** tabbed page.
3. Locate the Performance Monitoring settings in the list of services.

The above settings are used with the Resource Analyzer.

For tuning performance, use the Performance Tuner Wizard.

6.6.31.1.1: Using the Performance Tuner Wizard of the Java administrative console

Open the Performance Tuner Wizard by clicking **Console-> Wizards-> Performance Tuner** from the WebSphere Advanced Edition administrative console.

Using the Performance Tuner Wizard

Use this wizard to tune the most common performance related settingsassociated with one of your application servers or one of your server groups.Optimize performance by working with settings for yourapplications, servlets, enterprise beans, data sources, and expected load.

Application Server

Select the application server or server group to tune.

If you are tuning a server group, all updates will be reflected in the servergroup and only some will be propagated to existing clones (see the server group property reference for details). Subsequently created clones will be based on the updated server group values.

Web Container

Update the pool size within the normal range by adjusting the slider. Edit the numeric field toupdate the pool size within, or outside the normal range. Think about specifying the maximum number of Web container threadsless than the number of connections accepted by the Web server. Consider setting the Web containermaximum threads number significantly smaller than the number of connections accepted by the Web server, for a site with many static pages returned directly by the Web server.

ORB Properties

You can tune the following object request broker properties:

- **Pass by Reference**

  This option can provide better performance. Select Pass by Reference, only if appropriate for yourapplication. Selecting this option can break remote transparency, since you can modify objectspassed to an EJB method. Know your application before using this option.
- **ORB Threads Pool Size**

  A thread is needed for each EJB request. Enterprise beans are typically invoked from servlets in another JVM,using RMI/IIOP and remote EJB client applications, using RMI/IIOP. The ORB thread pool size should accommodate both request sources.

Data Source

Select a data source that is associated with the selected application server. After selecting thedata source, click **Next**. You are given the opportunity to tune the corresponding:

- Connection pool size
- Prepared statement cache size
- Database (for DB2 only)

Data Source: Connection Pool Size

Each data source contains a pool of connections to the corresponding database. An upper bound for your application comes from the sum of the number of Web container threads and ORB threads.

The servlet contribution to the connection pool size can be significantly smaller than the Web container maximum threads, if only a small percentage of servlet requests use these database connections.

Data Source: Prepared Statement Cache Size

Make your prepared statement cache large enough for all prepared statements, by setting the cache sizeto the product of:

- The number of SQL prepared statements in your application
- The maximum number of configured data source connections

Database: (DB2 Only)

This panel is only available for DB2 databases.

Tune the database after this wizard has completed, by selecting **Tune database** and entering the DB2 SYSADM ID and password. If you provided a cataloged database alias name when configuringthe data source and this alias name differs from the real database name, enter the *real database name*in the field provided.

After database tuning is selected, and you have completed the remaining panels, click **Finish** on the Summary panel. The tuning wizard then calls the DB2SmartGuide API, to tune the DB2 database associated with the data source. Stop and restart the database instance, for the DB2SmartGuide changes to take effect.

Note: DB2SmartGuide tuning works better if the database is already populated. It isnot necessary or recommended to tune the repository database (WAS).

Before tuning a database, you might want to use the DB2 DBA utility to back up (db2cfexp) the databaseconfiguration. If the tuning fails, you can then restore (db2cfimp) your database configuration. You can also use:

- **`DB2 RESET DATABASE CONFIGURATION FOR database-name`**

  to restorethe database to default values
- **`DB2 RESET DATABASE MANAGER CONFIGURATION`**

  to restore the DBM to defaultvalues

The database being tuned must reside on a DB2 Server at Version 7.2.1, or higher. This tuning optionis not available in DB2 servers for OS/390, OS/400, VM or VSE.

Java Virtual Machine (JVM) Heap Size

The Java Virtual Machine (JVM) Heap Size settings influence garbage collection of Java objects. Ifyou increase the heap size, garbage collection occurs less frequently, but takes longer.

These settings depend strongly on your application and on the amount of physical memory available.Consider:

- whether the JVM Heap for the selected application server shares physical memory with otherapplication server JVM Heaps on the same machine.
- specifying JVM Heaps to reside in physical memory and prevent swapping to disk.

- setting the starting JVM Heap Size to one quarter of the maximum JVM Heap Size.
- setting the maximum JVM Heap Size to the following, if you have only one application server on themachine:
    - 128 MB, for small systems with less than 1 GB of memory
    - 256 MB, for systems with 2 GB of memory
    - 512 MB, for larger systems

Note: A value of 0, or blank, indicates that no starting or maximum heap size is passed, when initializing the JVM.

On OS/400, the JVM Heap Size is quite different and you should never set the maximum heap size.

Summary:

The following performance settings are saved in the WebSphere Application Server configuration when youclick **Finish**. Correct a setting without updating the configuration, by clicking **Back**.

After you click Finish, warnings are displayed for any of the following conditions:

- Any of the following parameters are changed to a value greaterthan the highest value on the corresponding slider:
    - Web Container Maximum Threads
    - ORB Thread Pool Size
    - Data Source Maximum Connections
    - Data Source Prepared Statement Cache Size
    - JVM Maximum Heap Size
- A database is ready for tuning. The warning notifies you about restarting the DB2 database instanceand suggests that you back up your database configuration before tuning.
- The number of Web Container Threads and ORB Threads is greater than a Data Source Connection Pool.

If these conditions are acceptable to you, then continue.If these conditions are not acceptable, return to the wizard to change asetting, or cancel the tuning session.

6.6.31.1.2: Configuring the Performance Monitor services of application servers, with the Java administrative console

During this task, you will specify performance monitoring settings ofan existing application server, which define how the application servercollects data for use by the Resource Analyzer.

1. Locate the Performance Monitoring settings in the application server properties.
2. Select **Edit** to display the Performance Monitoring settings properties dialog.
3. Configure the performing monitoring properties.
   1. Click **Enable performance counter monitoring**. (Later, you can use this check box to disable monitoring without losing the settings you have configured).
   2. Select an item in the Counter Settings tree view. (Expand the tree if you need to, by clicking the plus sign (**+** in the root of the tree).
   3. Specify a monitoring level for the item. This will affect the item and all of its children in the Counter Settings tree view.
4. When finished, click **OK**.
5. Click **Apply** on the application server properties view.

6.6.34: Administering environment entries

Environment entries are defined during application assembly. Please see the related information topics.

6.6.34.0: Properties of environment entries

Environment entries are configuredduring application assembly. Refer to the"Assembly properties forenvironment entries".

6.6.34.0.aa: Assembly properties for environment entries

Environment entries define variables used to customize the business logicof an application at run-time. This eliminates the need to access orchange the application's source code. The container makes theapplication's environment entries available in a JNDI naming context(java:comp/env). An example environment entry is maxExceptions,describing the maximum number of tax exemptions that are allowed. Theenvironment entry's expected type is java.lang.Integer andits value is 15.

Name (Required, String)

Specifies the name of the environment entry, relative to thejava:comp/env context.

Value

Contains the value of the environment entry. The value must be astring that is valid for the constructor of the specified type that takes asingle string parameter.

Type (Required, String)

Specifies the fully qualified Java type of the environment entry valuethat is expected by the module's code. The following are the legalvalues for this field: java.lang.Boolean,java.lang.String, java.lang.Integer,java.lang.Double, java.lang.Byte,java.lang.Short, java.lang.Long, andjava.lang.Float.

Description

Contains text describing the environment entries.

6.6.35: Administering generation of deployment code

The Application Assembly Tool contains settings foradministering how deployment code will be generated.Please see the sub-topics for details.

6.6.35.0: Assembly properties for generating deployment code

Deployed module location (Required, String)

Specifies the name and location where the deployed archive will besaved. By default, this is the directory where you saved the archivefile or the directory from which you launched the Application AssemblyTool. The deployed archive file is given the same name as theundeployed archive, prefixed with `deployed_`.

Working directory (Required, String)

Specifies the full path name of the working directory. Thisdirectory is used to save temporary files while creating the deployedarchive. On UNIX systems, if there is insufficient space in the workingdirectory, a segmentation violation error can occur. Make sure thatthere is sufficient space in the working directory. Required spacevaries depending on the application. By default, the working directoryis `/tmp` on UNIX systems. You can specify a different workingdirectory by using this field.

Dependent classpath

Specifies the class path where dependent class files can be found, ifneeded by the module being deployed. List only additional JAR filesreferred to by the JAR file being deployed. On Windows NT systems,separate each path by a semicolon (;). On UNIX systems, separateeach path by a colon (:). Use one entry for each directory or JARfile to be searched. Files without the extension .jar or.zip are ignored.

Code generation only

Specifies that only stub and skeleton files are to be generated.The RMIC command and javac compiler are not to be run. The default isfalse (that is, all steps are executed).

Verify archive

Specifies whether verification is to take place during deployment.The default is false. Verification checks the completeness of thearchive and ensures that the classes in the EJB JAR files are in compliancewith the EJB 1.1 specification. Note that verifying archivefiles is important for successful generation of deployment code for theapplication. If an archive file is not valid, code generation canfail.

RMIC options

Specifies additional options to be passed to the RMIC command.These additional options are appended to the following options, which arealways passed to the RMIC command and cannot be overridden: -keep (keepsintermediate generated files), -iiop (generates stubs for IIOP), -ddirectory_name (writes the files to the specified directory),-sourcepath directory_name (locates the source files in thespecified directory), and -nowrite (compiled classes are not written to thefile system).

Database type

Specifies the database type, operating system, and version number.This information is used when generating the SQL code for creating databasetables for entity beans with container-managed persistence (CMP).

Database name

Specifies the name of the database. This information is used whengenerating the SQL code for creating database tables for entity beans withcontainer-managed persistence (CMP). The database name is used in thegenerated DDL file.

Schema name

Specifies the name of the database schema. This information is usedwhen generating the SQL code for creating database tables for entity beanswith container-managed persistence (CMP). The schema name is used inthe generated DDL file and in the generated persistence code.

6.6.36: Administering Java Virtual Machine settings (overview)

Please see the related information links.

6.6.36.0: JVM properties

Key:

Applies to Java administrative console of Advanced Edition Version 4.0

Applies to Web administrative console of Advanced Single Server Edition Version 4.0

Applies to Application Client Resource Configuration Tool

Boot Classpath

Some JVMs contain an option to specify bootstrap classes and resources.

For the Advanced Single Server Edition: This property can contain multiple paths separated by colons (":") or semicolons (";") depending on the operating system of the node.

For the Advanced Edition: Use the three fields for specifying:

- Total replacement of the default bootclasspath
- A string to *prepend* to the beginning of the bootclasspath
- A string to *append* to the end of the bootclasspath

Classpaths **or Classpath**

The standard classpath in which the Java virtual machine looks for classes.

For the Advanced Single Server Edition: Enter the classpath into a single text field, separating multiple paths by colons (":") or semicolons (";") depending on the operating system of the node.

For the Advanced Edition: Enter each classpath entry into a table row; you do not need to add the colon or semicolon at the end of each entry.

Command line arguments **or Generic Command line arguments**

The command line arguments to pass to the Java virtual machine (JVM) that starts the application server process.

You can find the most updated information about the command line arguments by entering at a command prompt `java` for the standard options and `java -X` for the non-standard options. In case that is not convenient, check this summary list of the arguments.

Debug Arguments **or Debug String**

When Debug Mode for the Java virtual machine is enabled, this property allows additional debug arguments to be passed to the Java virtual machine.

Debug Mode **or Enable Debug Mode**

Enables the JVM debug output. When the property is set, the application server will start with the java_g -debug argument, which is necessary to allow the Distributed Debugger, or any Java debugger, to attach to the application server.

This field and the **Enable IBM Distributed Debugger** field cannot be enabled at the same time.

Disable JIT

Disables the Just In Time (JIT) compiler option of the JVM

Enable Class Verbose Mode **or Verbose Mode**

Enables verbose debugging output for class loading

Enable Debug Mode

See Debug Mode

Enable IBM Distributed Debugger

See the OLT and Debugger property reference

Generic Command line arguments

See Command Line Arguments

Enable Garbage Collection Verbose Mode **or Verbose Mode Garbage Collection**

Enables verbose debug output for garbage collection

Enable JNI Verbose Mode **or Verbose Mode JNI**

Enables verbose debugging output for native method invocation

Generated Command Line Arguments (read only)

Displays the commands that will be added to the Java command line, based on your configuration ofthe JVM settings (including the Advanced JVM Settings).

Hprof Arguments

Additional profiler arguments to use when Run HProfs is used

Initial Heap Size **or Initial Java Heap Size**

The initial heap size available to the JVM, in megabytes

Initial Java Heap Size

See Initial Heap Size

Maximum Heap Size **or Maximum Java Heap Size**

The maximum heap size available to the JVM, in megabytes

Maximum Java Heap Size

See Maximum Heap Size

Run HProfs

Enable HProf profiler support

For the Advanced Edition: Use the checkbox to enable or disable HProf options. Use the text field to specify the options.

For the Advanced Single Server Edition: To use a different profiler, use the command line property to specify settings for the custom profiler.

Source Path

See the OLT and Debugger property reference

System Properties

A list of name-value pairs that can be passed into the virtual machine for initializing it. -D*name*=*value* pairs are created from the values entered.

Verbose Mode

See Enable Class Verbose Mode

Verbose Mode Garbage Collection

See Enable Garbage Collection Verbose Mode

Verbose Mode JNI

See Enable JNI Verbose Mode

See also the other application server properties.

6.6.36.0.1: Java command line arguments reference

This section provides a reference of Java Virtual Machine (JVM) commandline arguments related to performance and debugging. The WebSphereadministrator can configure application servers and other managed Java processesto start with these parameters as command line arguments.

Additional command line arguments are valid, beyond those listed in this section. Fora list of the command line arguments currently available on your operating system with your JavaDevelopment Kit (JDK) level, type `java` or `java -X` at a system command prompt.

Command line arguments related to performance

Goal	Argument	Values	Notes
Specify the maximum heap size the Java interpreter will use for dynamicallyallocated objects and arrays	-Xmx	Specify the value in bytes, with a value greater than 1000	On AIX, the default is 32M
Specify how much memory is allocated for the heap when the JVM starts	-Xms	Specify the value in bytes, with a value greater than 1000	On AIX, the default is 1M
Specify the size of each thread Java code stack	-oss	Specify the value in bytes, with a value greater than 1000	On AIX, the default is 400K
Specify the size of each thread native code stack	-ss	Specify the value in bytes, with a value greater than 1000	On AIX, the default value is 256K

Command line arguments related to debugging

Goal	Argument	Values	Notes
Disable the JIT compiler	-Djava.compile=none	None	Not available on AIX or Solaris
Specify whether to run the byte-code verifier on all loaded classes	-verify	*true*|*false*	None
Verify classes read in over the network	-verifyremote	None	None
Avoid verifying any classes	-noverify	None	None
Specify whether to print a message whenever the garbage collector frees memory	-verbosegc	None	None
Prevent asynchronous garbage collection	-noasyncgc	None	None
Disable class garbage collection	-Xnoclassgc	None	None
Specify whether to print a message each time the JVM loads a class	-verbose	None	None

6.6.37: Administering mail providers and mail sessions (overview)

WebSphere Application Server's implementation ofJavaMail does not provide mail servers. Even if your application components can communicatewith mail servers, you still must configure separate IMAP and SMTP serversto enable the mail functions.

Only the SMTP and IMAP service providers are shipped with WebSphere Application Server.To use other protocols, install the appropriate service providers for those protocols.

When you configure your mail servers, use fully qualified internet host names.

For information on how to install a service provider, see Chapter 5, *The Mail Session*, in theJava Mail API design specification.

The parameters used to configure a mail session resource can be dividedinto two groups:

1. mail send (transport) and
2. mail store access

If your enterprise application references a mail sessionresource, your application will use one of these functions:

- only mail-send
- both mail-send and mail-store access
- only mail-store access

> Use of the *only mail store-access* option is rare. As such, the store access part of the configuration is treated bySystem Management as optional while the *mail-send* function is treated as mandatory.

6.6.37.0: Properties of JavaMail providers

Key:

Applies to Java administrative console of Advanced Edition Version 4.0

Applies to Web administrative console of Advanced Single Server Edition Version 4.0

Applies to Application Client Resource Configuration Tool

Class Path

The path to the JAR file containing the implementation class for the JavaMail provider

Custom Properties

Name-value pairs for setting additional properties beyond those available in the administrative interface

Description

Optional description for your administrative records

Name

Administrative name of the JavaMail provider

6.6.37.0.1: Properties of JavaMail sessions

Key:

Applies to Java administrative console of Advanced Edition Version 4.0

Applies to Web administrative console of Advanced Single Server Edition Version 4.0

Applies to Application Client Resource Configuration Tool

Category

Optional category for classifying this resource for your administrative records

Custom Properties

Name-value pairs for setting additional properties

Confirm Password **or Re-Enter Password**

Confirm the password that you entered in the preceding field

Description

Optional description for your administrative records

Enable Store

Enable JavaMail sessions

J2EE Resource Provider

The JavaMail provider with which this JavaMail session is associated

JNDI Binding Path **or JNDI Name**

The JNDI name for the resource, including any naming subcontexts. This name is used as the linkage between the platform's binding information for resources defined in the client applications deployment descriptor and actual resources bound into JNDI by the platform

JNDI Name

See JNDI Binding Path

Name

Administrative name of the JavaMail session object

Mail From **or Outgoing Mail Originator**

Value of replyTo field in mail messages sent through the mail transport host. The Internet email address that by default will bedisplayed in the received message as either the "From" or the "Reply-To" address. The recipient's reply will come to the specified address.

Mail Store Access Host **or Mail Store Host**

The server to which to connect when reading mail. This property combines with the mail store user ID and password to represent a valid mail account. For example, if the mail account is john_william@my.company.com, enter my.company.com.

Mail Store Access Password or Mail Store Password

The password to use when connecting to the mail store host. This property combines with the mail store user ID and password to represent a valid mail account. For example, if the mail account is john_william@my.company.com, enter the password corresponding to john_william.

Mail Store Access Protocol or Mail Store Protocol

The protocol to be used when reading mail. Should be IMAP.

Mail Store Access User Name or Mail Store User

The user ID to use when connecting to the mail store host.This property combines with the mail store user ID and password to represent a valid mail account. For example, if the mail account is john_william@my.company.com, enter john_william.

Mail Store Host

See Mail Store Access Host

Mail Store Password

See Mail Store Access Password

Mail Store Protocol

See Mail Store Access Protocol

Mail Store User

See Mail Store Access User

Mail Transport Host or Outgoing Mail Server

The server to which to connect when sending mail. Specify the fully qualified Internet host name of the mail server, also known as the SMTP server

Mail Transport Password or Outgoing Mail Password

The password to provide when connecting to the mail transport host. This property is rarely used for the default SMTP protocol. You can leave this field blank unless you use a transport protocol that requires a user ID and password.

Mail Transport Protocol or Outgoing Mail Protocol

The transport or protocol to use when sending mail, such as "POP3", "IMAP4", or "SMTP"

The default is set to SMTP, and usually you should not change it. To use a different protocol, first install the required service provider, and then enter the protocol name in this field.

Mail Transport User or Outgoing Mail User Name

The user ID to provide when connecting to the mail transport host. This property is rarely used for the default SMTP protocol. You can leave this field blank unless you use a transport protocol that requires a user ID and password.

Outgoing Mail Originator

See Mail From

Outgoing Mail Server

See Mail Transport Host

Outgoing Mail Password

See Mail Transport Password

Outgoing Mail Protocol

See Mail Transport Protocol

Outgoing Mail User Name

See Mail Transport User

Re-Enter Password

See Confirm Password

6.6.37.1: Administering JavaMail support resources with the Java console

Use the Java administrative console to administer JavaMail sessions.

Work with resources of this type by locating them in the tree view:

WebSphere Administrative Domain -> Resources -> JavaMail Sessions

The instances will be displayed in the details view.

6.6.37.1.1: Configuring new JavaMail support resources with the Java administrative console

1. Click **Console -> New -> JavaMail Session** from the console menu bar.
2. Specify JavaMail session properties.
3. Click **OK**.

Alternatively, use the **Console -> New** menu options pertaining to JavaMail resources.

6.6.37.9: Administering JavaMail providers and sessions with the Application Client Resource Configuration Tool

Use the Application Client Resource Configuration Tool to edit the configurations of JavaMail sessions to be used by your application clients.

Work with objects of this type by locating them in the tree that is displayed by the tool when you use it to openan EAR file. If file with which you are working contains JavaMail sessions, its tree will contain one or more of the following:

Resources -> ***application*.jar** -> **JavaMail Providers** -> **Mail Provider** (a default mail provider) ->**JavaMail Sessions**

Inside the **JavaMail Sessions** folder will be JavaMail sessioninstances.

6.6.37.9.1: Configuring new JavaMail sessions with the Application Client Resource Configuration Tool

During this task, you will configure new mail sessions for your application client. The mail sessions will be associated with the preconfigured default mail provider supplied by the product.

To configure a new JavaMail Session:

1. Start the tool and open the EAR file for which you want to configure the new JavaMail session. The EAR file contents will be displayed in a tree view.
2. From the tree, select the JAR file in which you wantto configure the new JavaMail session.
3. Expand the JAR file to view its contents.
4. Click **JavaMail Providers** -> **MailProvider** -> **JavaMail Sessions**. Do one of the following:
    - Right-click the **JavaMail Sessions** folder and select **New Factory**.
    - On the menu bar, click **Edit** -> **New**.
5. In the resulting property dialog, configure the JavaMail session properties.
6. When finished, click **OK**.
7. On the menu bar, click **File** -> **Save** to save your changes.

6.6.37.9.3: Removing JavaMail sessions with the Application Client Resource Configuration Tool

Please see "Removing objects from EAR files with theApplication Client Resource Configuration Tool", as this task is similar for all object types supported by the tool.

6.6.37.9.4: Updating JavaMail session configurations with the Application Client Resource Configuration Tool

During this task, you will modify (update) the configuration of an existing JavaMail session. Note, you cannot update the default JavaMail provider, but you can viewits properties by performing similar steps.

1. Start the tool and open the EAR file containing the JavaMail session. The EAR file contents will be displayed in a tree view.
2. From the tree, select the JAR file containing the JavaMail session that you want to update.
3. Expand the JAR file to view its contents.
4. Keep expanding the JAR file contents until you locate the particular JavaMail session that you want to update. When you find it, do one of the following:
   - Right-click the object and select **Properties**
   - On the menu bar, click **Edit** -> **Properties**
5. In the resulting property dialog, update the properties. For detailed field help, see:
   - JavaMail session properties
6. When finished, click **OK**.
7. On the menu bar, click **File** -> **Save** to save your changes.

6.6.38: Administering URL providers and URLs (overview)

A Default URL Provider is included in the initial product configuration. It utilizes the URL support provided by the JDK. Any URL resources with protocols supported by the JDK (such as HTTP, FTP, FILE) can use the Default URL Provider.

Please see the related information links for tasks and settings relatedto URL support.

6.6.38.0: Properties of URL providers

Key:

☕ Applies to Java administrative console of Advanced Edition Version 4.0

🌐 Applies to Web administrative console of Advanced Single Server Edition Version 4.0

C:\ Applies to Application Client Resource Configuration Tool

Classpath ☕ **or Class Path** C:\ **or Server Classpath** 🌐

The path to the JAR file containing the implementation classes for the URL provider

Custom Properties C:\

Name-value pairs for setting additional properties

Description ☕ 🌐 C:\

Optional description of the URL Provider, for your administrative records

Name ☕ 🌐 C:\

Administrative name for the URL Provider

Node ☕

The node with which the URL Provider is associated

If using the Java-based administrative console, use the buttons on the node panel to access dialogs for installing drivers on specific nodes, and for uninstalling drivers.

Protocol ☕ 🌐 C:\

The protocol supported by this stream handler.

For Advanced Single Server Edition: This field is required

For the Application Client Resource Configuration Tool: This refers to a *custom*protocol. If you are going to use a standardprotocol, such as "nntp," "smtp," or "ftp," then leave this field blank

Server Classpath 🌐

See Classpath

Stream Handler Class ☕ C:\ **or Stream Handler Class Name** 🌐

Fully qualified name of Java class that implements the stream handler for the protocol specified by the **Protocol** property

6.6.38.0.1: Properties of URLs

Key:

☕ Applies to Java administrative console of Advanced Edition Version 4.0

🌐 Applies to Web administrative console of Advanced Single Server Edition Version 4.0

C:\ Applies to Application Client Resource Configuration Tool

Category 🌐

Administrative category for the URL

Custom Properties C:\

Name-value pairs for setting additional properties

Description ☕ 🌐 C:\

Optional description of the URL, for your administrative records

JNDI Binding Path ☕ **or JNDI Name** 🌐 C:\

The JNDI name for the resource, including any naming subcontexts. This name is used to link the platform's binding information. The binding associates the resources defined in the deployment descriptor of the module to the actual (physical) resources bound into JNDI by the platform.

JNDI Name 🌐 C:\

See JNDI Binding Name

Name ☕ 🌐 C:\

Administrative name for the URL

Spec 🌐 **or URL** ☕

The string from which to form a URL

URL ☕ C:\

See Spec

URL Provider ☕

The URL Provider that implements the protocol for this URL

6.6.38.1: Administering URL resources with the Java administrative console

Use the Java administrative console to administer URL providers and URLs.

Work with resources of this type by locating them in the tree view:

WebSphere Administrative Domain -> Resources -> URL Providers -> *provider_name* -> URLs

The URL instances will be displayed in the details view.

6.6.38.1.1: Configuring URL resources with the Java administrative console

URL resources allow you to define specific locations for a resource on a network.

To create a URL resource, click **Console -> Wizards -> Create URL Resource** from the console menu bar. This leads to the URL Resource task wizard.

Alternatively, use the **Console -> New** menu options pertaining to URL resources.

Using the URL Resource wizard

You use a URL Resource wizard to define a resource that uses the Default URL Provider. Using the wizard sets basic properties pertaining to URL resources. To set other property values for URL resources, use the properties dialogs for URL resources.

- Naming the URL resource and specifying its URL string
- Completing the URL resource

Naming the URL resource and specifying its URL string

On the Specifying the URL Resource panel:

1. Name your resource.
2. Describe your resource.
3. Specify the string for the URL. A sample string is `http://www.ibm.com`.

   View URL resources properties help
4. Click **Next**.

Completing the URL resource

The final panel lists the URL resource name and the URL string.

If you do not want to change the values specified, click **Finish**. Thewizard will define a URL resource with the values you specified, and display a message indicating whether the resource was successfully created. Your specifiedURL will be created using the Default URL Provider. If the wizard encounters an error, a message will display explaining why a URL resource could not be defined.

To change the values specified, click **Back** to return to the appropriatepanel(s), make any needed changes, and then click **Finish** on this panel.

6.6.38.1.2: Installing and uninstalling URL providers on nodes, with the Java administrative console

To install a provider:

1. Locate a URL provider instance in the tree view.
2. Click the **Nodes** tabbed page.
3. Click the **Install New** button.
4. Configure the new provider:
    1. Select a node on which to install the provider.
    2. Specify or browse for the classpath directory (see URL provider properties).
    3. Click **Install**.
5. Click **Apply** on the URL provider properties view.

To uninstall a provider:

1. Locate a URL provider instance in the tree view.
2. Click the **Nodes** tabbed page.
3. Select a provider from the list of providers.
4. Click **Uninstall**.
5. Click **Apply** on the URL provider properties view.

You can also click a node instance in the tree view todisplay a similar configuration dialog.

6.6.38.9: Administering URL providers and URLs with the Application Client Resource Configuration Tool

Use the Application Client Resource Configuration Tool to edit the configurations of URL providers and URLs to be used by your application clients.

Work with objects of this type by locating them in the tree that is displayed by the tool when you use it to openan EAR file. If file with which you are working contains URL providers and URLs, its tree will contain one or more of the following:

Resources -> ***application*.jar** -> **URL Providers** -> ***url_provider_instance***

where ***url_provider_instance*** isa particular URL provider.

If you expand the tree further, you will also see the **URLs** folders containing the URL instances for each URL provider instance.

6.6.38.9.1: Configuring new URL providers and URLs with the Application Client Resource Configuration Tool

During this task, you will create URL providers and URLs for your client application. Note, in a separate administrative task, the Java code for the required URL provider must be installed on the client machine on which the client application resides.

To configure a new URL provider:

1. Start the tool and open the EAR file for which you want to configure the new URL provider. The EAR file contents will be displayed in a tree view.
2. From the tree, select the JAR file in which you wantto configure the new URL provider.
3. Expand the JAR file to view its contents.
4. Click the folder called **URL Providers**. Do one of the following:
   - Right-click the folder and select **New Provider**.
   - On the menu bar, click **Edit** -> **New**.
5. In the resulting property dialog, configure the URL provider properties.
6. When finished, click **OK**.
7. On the menu bar, click **File** -> **Save** to save your changes.

6.6.38.9.1.1: Configuring new URLs with the Application Client Resource Configuration Tool

During this task, you will create URLs for your client application.

1. In the tree, click the URL provider for which you want to create a URL.
    - Configure a new URL provider.
    - Or, click an existing URL provider.
2. Expand the URL provider to view its **URLs** folder.
3. Click the folder. Do one of the following:
    - Right-click the folder and select **New Factory**.
    - On the menu bar, click **Edit** -> **New**.
4. In the resulting property dialog, configure the URL properties.
5. When finished, click **OK**.
6. On the menu bar, click **File** -> **Save** to save your changes.

6.6.38.9.3: Removing URL providers and URLs with the Application Client Resource Configuration Tool

Please see "Removing objects from EAR files with theApplication Client Resource Configuration Tool", as this task is similar for all object types supported by the tool.

6.6.38.9.4: Updating URL and URL provider configurations with the Application Client Resource Configuration Tool

During this task, you will modify (update) the configuration of an existing URL or URL provider.

1. Start the tool and open the EAR file containing the URL or URL provider. The EAR file contents will be displayed in a tree view.
2. From the tree, select the JAR file containing the URL or URL providerthat you want to update.
3. Expand the JAR file to view its contents.
4. Keep expanding the JAR file contents until you locate the particular URL or URL provider that you want to update. When you find it, do one of the following:
   - Right-click the object and select **Properties**
   - On the menu bar, click **Edit** -> **Properties**
5. In the resulting property dialog, update the properties. For detailed field help, see:
   - URL provider properties
   - URL properties
6. When finished, click **OK**.
7. On the menu bar, click **File** -> **Save** to save your changes.

6.6.39: Administering messaging and JMS providers (overview)

Please see the related information links.

6.6.39.0: Properties of JMS providers

Key:

☕ Applies to Java administrative console of Advanced Edition Version 4.0

🌐 Applies to Web administrative console of Advanced Single Server Edition Version 4.0

C:\ Applies to Application Client Resource Configuration Tool

Binding Classname ☕ **or Binding Class** C:\ **or JNDI Binding Mechanism** 🌐

Java classname to be used for namespace binding. This value is required only for providers with non-standard binding requirements.

Classpath C:\ **or Class Path** ☕ **or Server Class Path** 🌐

The class path that identifies the location of the driver classes (the JMS initial context factory)

Context Factory Class C:\ **or Context Factory Classname** ☕

The Java class name of the JMS providers initial context factory

Custom Properties C:\

Name-value pairs for setting additional properties

Description ☕ 🌐 C:\

Optional description for your administrative records

External Initial Context Factory 🌐

Java classname of the initial context factory of a provider

External Provider URL 🌐 **or Provider URL** ☕

JMS provider URL for external JNDI lookups

JNDI Binding Mechanism 🌐

See Binding Classname

Name ☕ 🌐 C:\

Administrative name for this provider

Nodes ☕

The nodes with which this provider is associated. See also node properties.

If using the Java-based console, use the buttons on the node panel to access dialogs for installing providers on specific nodes, and for uninstalling providers.

Provider URL ☕ C:\

See External Provider URL

Server Class Path

See Classpath

6.6.39.0.1: Properties of JMS connection factories

Key:

Applies to Java administrative console of Advanced Edition Version 4.0

Applies to Web administrative console of Advanced Single Server Edition Version 4.0

Applies to Application Client Resource Configuration Tool

Category

Optional category for your administrative records

Connection Type

Whether the JMS Destination is a queue or topic. Queues are used for point-to-point messaging. Topics are used for publish-and-subscribe messaging.

Custom Properties

Name-value pairs for setting additional properties

Description

Optional description for your administrative records

External JNDI Name **or External JNDI Path**

Namespace location of JMS created connection factory

External JNDI Path

See External JNDI Name

JMS Provider

The JMS provider with which this connection factory is associated

JNDI Name

Namespace location of JMS created connection factory, including any naming subcontexts.

The name is used to link the platform binding information. The binding associates the resources defined the deployment descriptor of the module to the actual (physical) resources bound into JNDI by the platform.

Name

Administrative name for this JMS connection factory

6.6.39.0.2: Properties of JMS destinations

Key:

☕ Applies to Java administrative console of Advanced Edition Version 4.0

🌐 Applies to Web administrative console of Advanced Single Server Edition Version 4.0

C:\ Applies to Application Client Resource Configuration Tool

Category 🌐

Optional category for your administrative records

Custom Properties C:\

Name-value pairs for setting additional properties

Description ☕ 🌐 C:\

Optional description for your administrative records

Destination Type C:\ 🌐

Whether the JMS Destination is a queue or topic. Queues are used for point-to-point messaging. Topics are used for publish-and-subscribe messaging.

External JNDI Name C:\ 🌐 **or External JNDI Path** ☕

Namespace location of JMS created destination

External JNDI Path ☕

See External JNDI Name

JMS Provider ☕

The JMS provider with which this connection factory is associated

JNDI Name C:\ 🌐

Namespace location of JMS created connection factory, including any naming subcontexts.

The name is used to link the platform binding information. The binding associates the resources defined the deployment descriptor of the module to the actual (physical) resources bound into JNDI by the platform.

Name ☕ 🌐 C:\

Administrative name for this JMS destination

6.6.39.1: Administering JMS support resources with the Java administrative console

Use the Java administrative console to administer JMS providers, JMS connectionfactories, and JMS destinations.

Work with resources of this type by locating them in the tree view:

WebSphere Administrative Domain -> Resources -> JMS Providers -> ***provider_name***

Expand the tree further to see the **JMS Connection Factories** and **JMS Destinations**folders. The corresponding instances will be displayed in the details view.

6.6.39.1.1: Configuring JMS resources with the Java administrative console

To create a JMS connection factory or destination, click **Console -> Wizards -> Create JMS Resource** from the console menu bar. This leads to the JMS Resource task wizard.

Alternatively, use the **Console -> New** menu options pertaining to JMS resources.

Using the JMS Resource wizard

You can use a JMS Resource wizard to define a connection factory or destination. Using the wizard sets basic properties pertaining to JMS resources. To set other property values, use the properties dialogs for JMS resources.

- Selecting to create a new connection factory or destination
- Naming the JMS resource and specifying its JNDI path
- Specifying a JMS provider
- Creating a new JMS provider
- Completing the connection factory or destination

Selecting to create a new connection factory or destination

On the Specifying a JMS Resource panel, select to create a new JMS connectionfactory or a new destination, and then click **Next**.

Naming the JMS resource and specifying its JNDI path

On the Creating a JMS Connection Factory or Creating a JMS Destination panel:

1. Specify the JNDI path where the JMS administrative tool has bound this resource.

   If a connection factory is being created, it is recommended that within the name you identify the JMS resource to which the factory will provide connections. For example, enter `QueueConnectionFactory`.
2. Describe your resource.
3. Specify the external JNDI path where your JMS resource should be stored. A sample path is `jms/QueueConnectionFactory`.

   View JMS resources properties help
4. Click **Next**.

Specifying a JMS provider

On the Specifying a JMS Provider panel, select an existing JMS provider or opt to create a new JMS provider, and then click **Next**.

A JMS provider represents a library that implements the JMS interfaces fora specific vendor.

Creating a new JMS provider

On the Creating a JMS Provider panel:

1. Name your JMS provider. Further, specify a context factory class name and a URL for the provider. Optionally, describe your JMS provider and specify a binding class name.

   View JMS resources properties help

2. Click **Next**.

Completing the connection factory or destination

The final panel lists the JMS resource name, the external JNDI path where the resource will reside, and the provider name. If a new provider is being created, the panel also lists the context factory class name and the provider URL.

If you do not want to change the values specified, click **Finish**. Thewizard will define a connection factory or destination with the values you specified, and display a message indicating whether the resource was successfully created. If the wizard encounters an error, a message will display explaining why a connection factoryor destination could not be defined.

To change the values specified, click **Back** to return to the appropriatepanel(s), make any needed changes, and then click **Finish** on this panel.

6.6.39.1.2: Installing and uninstalling JMS providers on nodes, with the Java administrative console

To install a provider:

1. Locate a JMS provider instance in the tree view.
2. Click the **Nodes** tabbed page.
3. Click the **Install New** button.
4. Configure the new provider:
    - Select a node on which to install the provider.
    - Specify or browse for the classpath directory (see JMS provider properties).
    - Click **Install**.
5. Click **Apply** on the JMS provider properties view.

To uninstall a provider:

1. Locate a JMS provider instance in the tree view.
2. Click the **Nodes** tabbed page.
3. Select a provider from the list of providers.
4. Click **Uninstall**.
5. Click **Apply** on the JMS provider properties view.

You can also click a node instance in the tree view todisplay a similar configuration dialog.

6.6.39.9: Administering JMS providers, connection factories, and destinations with the Application Client Resource Configuration Tool

Use the Application Client Resource Configuration Tool to edit the configurations of JMS providers, JMS connectionfactories, and JMS destinations to be used by your application clients.

Work with objects of this type by locating them in the tree that is displayed by the tool when you use it to openan EAR file. If file with which you are working contains JMS providers, JMS connectionfactories, and JMS destinations, its tree will contain one or more of the following:

Resources -> ***application*.jar** -> **JMS Providers** -> ***jms_provider_instance***

where ***jms_provider_instance*** isa particular JMS provider.

If you expand the tree further, you will also see the **JMS Connection Factories** and **JMS Destinations** folders containing the connection factory and destination instances for each JMS provider instance.

6.6.39.9.1: Configuring new JMS providers with the Application Client Resource Configuration Tool

During this task, you will create new JMS provider configurations for your applicationclient. The client application can make use of a messaging service through the Java Message Service APIs. A JMS provider provides two kinds of J2EE factories. One is a JMS Connection factory, and the other is aJMS destination factory.

Note, in a separate administrative task, the JMS client must be installed on the client machine where the client application resides. The messaging product vendor must provide an implementation of the JMS client. For more information, see your messaging product documentation.

To configure a new JMS provider:

1. Start the tool and open the EAR file for which you want to configure the new JMS provider. The EAR file contents will be displayed in a tree view.
2. From the tree, select the JAR file in which you wantto configure the new JMS provider.
3. Expand the JAR file to view its contents.
4. Click the folder called **JMS Providers**. Do one of the following:
    - Right-click the folder and select **New Provider**.
    - On the menu bar, click **Edit** -> **New**.
5. In the resulting property dialog, configure the JMS provider properties.
6. When finished, click **OK**.
7. On the menu bar, click **File** -> **Save** to save your changes.

6.6.39.9.1.1: Configuring new connection factories with the Application Client Resource Configuration Tool

During this task, you will create a new JMS connection factory configuration for your applicationclient.

To configure a new connection factory:

1. In the tree, click the JMS provider for which you want to create a connection factory.
    - Configure a new JMS provider.
    - Or, click an existing JMS provider.
2. Expand the JMS provider to view its **JMS Connection Factories** folder.
3. Click the folder. Do one of the following:
    - Right-click the folder and select **New Factory**.
    - On the menu bar, click **Edit** -> **New**.
4. In the resulting property dialog, configure the JMS connection factory properties.
5. When finished, click **OK**.
6. On the menu bar, click **File** -> **Save** to save your changes.

6.6.39.9.1.2: Configuring new JMS destinations with the Application Client Resource Configuration Tool

During this task, you will create new JMS destination configuration for your applicationclient.

To configure a new destination:

1. In the tree, click the JMS provider for which you want to create a destination.
   - Configure a new JMS provider.
   - Or, click an existing JMS provider.
2. Expand the JMS provider to view its **JMS Destinations** folder.
3. Click the folder. Do one of the following:
   - Right-click the folder and select **New Factory**.
   - On the menu bar, click **Edit** -> **New**.
4. In the resulting property dialog, configure the JMS destination properties.
5. When finished, click **OK**.
6. On the menu bar, click **File** -> **Save** to save your changes.

6.6.39.9.3: Removing JMS providers, connection factories, and destinations with the Application Client Resource Configuration Tool

Please see "Removing objects from EAR files with theApplication Client Resource Configuration Tool", as this task is similar for all object types supported by the tool.

6.6.39.9.4: Updating JMS provider, connection factory, and destination configurations with the Application Client Resource Configuration Tool

During this task, you will modify (update) the configuration of an existing JMS provider, connection factory, or destination.

1. Start the tool and open the EAR file containing the JMS provider, connection factory, or destination.
   The EAR file contents will be displayed in a tree view.
2. From the tree, select the JAR file containing the JMS provider, connection factory, or destinationthat you want to update.
3. Expand the JAR file to view its contents.
4. Keep expanding the JAR file contents until you locate the particular JMS provider, connection factory, or destination that you want to update. When you find it, do one of the following:
   - Right-click the object and select **Properties**
   - On the menu bar, click **Edit** -> **Properties**
5. In the resulting property dialog, update the properties. For detailed field help, see:
   - JMS provider properties
   - JMS connection factory properties
   - JMS destination properties
6. When finished, click **OK**.
7. On the menu bar, click **File** -> **Save** to save your changes.

6.6.41: Administering WebSphere administrative domains (overview)

Please see the related information links.

6.6.41.0: Administrative domain properties

Key:

Applies to Java administrative console of Advanced Edition Version 4.0

Applies to Web administrative console of Advanced Single Server Edition Version 4.0

Applies to Application Client Resource Configuration Tool

Name

A logical name for the WebSphere administrative domain

6.6.41.1: Administering WebSphere administrative domains with the Java administrative console

Use the Java administrative console to administer WebSphere administrative domains.

Work with resources of this type by locating them in the tree view:

WebSphere Administrative Domain

6.6.41.5: Establishing multiple administrative domains on a machine

This procedure describes how to install multiple WebSphere 4.0 domains on thesame physical machine and allow them to run concurrently. This configuration isuseful for allowing developers or testers to isolate their WebSphere domain from other WebSphere administrative domains on the same machine. It also provides the ability to make changes to the overall domain without affecting other domains.

The procedure describes how to set up this configuration using examples from Solaris 8 and Windows NT. The Solaris examples can be extrapolated to other UNIX operating systems supported by IBMWebSphere Application Server. It is intended for experienced administrators who need to setup and maintain multiple WebSphere instances on multiprocessor servers.

The UNIX configuration used in this document is:

- Hardware: Sun Ultra 80 4x300MHz with 1024 MB of RAM
- Operating system: Sun Solaris 8
- Database: IBM DB2 Version 7.1.0.43
- Web Server: IBM HTTP Server 1.3.19.0

The Windows NT configuration used in this document is:

- Hardware: Dual 400 MHz Pentium II Netfinity with 650 MB of RAM
- Operating system: Windows NT 4.0 Service Pack 6a
- Database: IBM DB2 Version 7.1.0.43
- Web Server: IBM HTTP Server 1.3.19.0

Overview:

1. Install and configure separate HTTP servers and their WebSphere plug-ins
2. Run the WebSphere installation twice with the appropriate configuration information.
3. Change the default ports for the bootstrap service, LSD service, and Trace Service.
4. Adjust the port values to eliminate conflicts.
5. Start the servers in the configuration.
6. Test each Web server and application server combination.

Install the first Web server instance

Install the Web server as you usually would, making note of:

- The port on which the Web server is listening. Look for the port line in the httpd.conf file.
- The path in which the Web server is installed

Note, IBM HTTP Server is available during IBM WebSphere Application Serverinstallation, which is the next step.

Install the first IBM WebSphere Application Server instance

Before installing the product, create two databases:
`was40awas40b`
for use by the upcoming two installations of WebSphere Application Server.

See the InfoCenter PDF view for case-specificinstallation documents, and section 2 for more information.

Make note of:

- The path in which IBM WebSphere Application Server is installed, such as:
  `AIX:            /opt/WebSphere/AppServer40aWindows NT:     C:\WebSphere\AppServer40a`
- The name of the database WebSphere is using, such as was40a.

Test the single instance setup

1. Start the Web server. Ensure it is working.
2. Start the WebSphere Application Server product.
3. Start the administrative console.
4. Start the application server.
5. View the Web container properties.

   Record the port numbers on which the Web container HTTP transport is listening, for future reference.
6. Try to access the "snoop servlet" for verification:
   `http://HTTP_Server_hostname/servlet/snoop`

   This confirms thatthe "one instance" setup has basic functionality.
7. Stop the Web server.
8. Stop the application server.

Install the second Web server instance

To install the second instance of the Web server, you need eitherto create a new Web server instance using the Web server configuration application,as used by Netscape Enterprise Server, or to make copies of the necessaryconfiguration files and hand edit them inserting appropriate values, as isnecessary when using IBM Web server or Apache Server.

In this example, we willhand edit the configuration files, because we are using IBM Web server.

1. NT Perform this step if using Windows NT. Otherwise,skip this step.

   To install a second copy of the IBM HTTP Server on Windows NT, first you must remove an entry in the Windows Registry so the installprogram does not detect the already installed version.
   1. Start regedit
   2. Carefully, remove just this one key:
      ```
      My Computer\HKEY_LOCAL_MACHINE\SOFTWARE\IBM\HTTP Server
      ```
2. Copy the httpd.conf and apachectl files in their corresponding directories, suchas:
   ```
   httpd.conf copy to httpd2.confapachectl copy to apachectl2
   ```
3. Edit the httpd2.conf file. Change the values slightly (such as adding **2** to their values).
   - Change the port to an unused port, such as: Port 81
   - Change the error log, such as (for Solaris): ErrorLog /opt/IBMHTTPD/logs/error_log2
   - Change the custom log, such as (for Solaris): CustomLog /opt/IBMHTTPD/logs/access_log2 common
   - Change the pid file, such as (for Solaris): PidFile /opt/IBMHTTPD/logs/httpd2.pid
   - Change the score board file, such as (for Solaris): ScoreBoardFile /opt/IBMHTTPD/logs/httpd2.scoreboard
4. Save the httpd2.conf file.
5. Open the apachectl2 file in a text editor and make similar changes. For example, for Solaris:
   - PIDFILE=/opt/IBMHTTPD/logs/httpd2.pid
   - HTTPD='/opt/IBMHTTPD/bin/httpd -f /opt/IBMHTTPD/conf/httpd2.conf'

   (Note the single quotation marks in the HTTPD command).
6. Save the apachectl2 file.

Install the second WebSphere Application Server instance

Install the second WebSphere Application Server, with these notes:

- Use the Custom installation option.
- Skip the migration assistant if it is present.
- Do not "Backup and Uninstall" the previous version.
- When the installion notifies you that it has found another copy ofWebSphere 4.0, simply click **OK** to continue.
- Do not install the Web server or the database.
- Install the appropriate Web server plug-in.
- For *Advanced Edition* (non-Single Server), name the database something unique, such as "was40b" if the first installation's database was "was40a."
- For the Destination Directory, enter a unique directory for this instance,such as (for Solaris): /opt/WebSphere/AppServer40b
- ⚠ The installation program will onlyaccept a file named httpd.conf when it asks the location of the httpd.conf file. To get past this correctly, move thehttpd.conf file to httpd.confA and move httpd2.conf to httpd.conf.

  After the installation program updates thehttpd.conf file, move the files back to their original names, httpd.conf andhttpd2.conf.

Test the second instance installation, just as you did with the first instance to verify that each set of HTTP/AppServer functions correctly alone.(Note, you might need to restart the Web server to have it read in the plugin-cfg.xml file.)

Adjust the port values to eliminate conflicts

With the configuration as it stands, you can run each HTTPServer and Application Server combination separately, but you cannot run bothinstances concurrently due to conflicting ports.

To allow both instances to run at the same time, edit the port settings.

1. Add the following lines to the administrative server configuration file of one of the WebSphere Application Server installations.
   ```
   com.ibm.ejs.sm.adminServer.bootstrapPort=901com.ibm.ejs.sm.adminServer.lsdPort=9001
   ```
2. Start this instance of WebSphere Application Server.

   Remember that the installation directory is named uniquely, such as WebSphere/AppServer40b instead of just WebSphere/AppServer.
3. After the server has started, start the administrative console for this WebSphere Application Server instance. Open a command prompt.

   NT For Windows NT, run:
   ```
   adminclient hostname 901
   ```
   UNIX For Solaris, run:
   ```
   adminclient.sh hostname 901
   ```
4. Append a new port number to all of the virtual host settings of thisinstance of WebSphere. This is the value that the corresponding HTTPServer will listen on.

   Change the default ports of the host aliases:
   ```
   From *:80 to *:81From *:9080 to *:9081
   ```
   Editing virual host settings

5. Verify that each Web Container port does not conflict with any otherWeb Container port, both on this instance of WebSphere Application Server and otherinstances.
    - Edit the HTTP transport settings.
    - Now regenerate the WebSphere plug-in configuration.
    - Stop the Web server and start it again to ensure it picks up the regenerated configuration.
6. NT For Windows NT, change the HTTP server port numberfrom 80 to another valid port number, such as 81, in httpd.conf.

Start the servers in the configuration

To run this configuration, you will need to use a command prompt to startupeach Web server and each WebSphere Application Server.

1. NT For Windows NT, stop all of the running Webserver and application server services. Set each of the services to **manual** sothat they will not start automatically. The services to set are:
    `IBM HTTP Administration` `IBM HTTP Server` `IBM WS AdminServer`
2. Start each Web server.
3. Start each appliction server by changing directory tothe bin directory of each of the two installations andrunning the startupServer script.

Test each Web server and application server combination

To test that each Web server and application server pair has basicfunctionality, start the default application server in each WebSphere ApplicationServer instance. Try the following URLs to verify that basic functionality exists:

http://*HTTP_server_hostname*/servlet/snoophttp://*HTTP_server_hostname*:81/servlet/snoophttp://*HTTP_server_hostname*/webapp/examples/HitCounthttp://*HTTP_server_hostname*:81/webapp/examples/HitCount

Additional usage notes

To use the WebSphere administrative console with the instanceof WebSphere Application Server for which you changed the port settings, you must invoke it with the hostname and new port number on the command line,such as:

adminclient.sh *MyServer* 901

Whenever you add a new Web Container to either instance ofWebSphere Application Server, you must make sure that the port number of the WebContainer Service does not conflict with another Web Container Serviceon either instance of WebSphere Application Server.

6.6.42: Administering J2C related administrative resources (overview)

To deploy and assemble your application, you use a combination of the:

- Batch EJB Deployment tool
- Application Assembly Tool (AAT)
- Administrative console

In general, the flow looks like this:

1. Convert the EJB 1.0 jar file (.ser) to EJB 1.1 jar (.xml) and generate the deployment code.
2. Assemble the EJB JAR file and application client JAR file into a J2C application (.ear).
3. Define the J2C Resource Adapters and Connection Factories.
4. Install the application and resolve any References.
5. Run the application.

Convert the EJB JAR file and generate the deployment code

During this step, you will convert the EJB 1.0 JAR file (.ser) to an EJB 1.1 JAR file (.xml), and generate the deployment code.

1. Open a system command prompt.
2. Change directory to the *product_installation_root*/bin directory.
3. Execute the command:

   ```
   ejbdeploy YourFilename.jar tmp YourFilenameDep.jar
   ```

   where *YourFilename*.jar is the fully-qualified EJB 1.0 JAR file tobe converted, *tmp* is a directory for temporary files, and *YourFilename*Dep.jaris the fully-qualified name of the new file for the converted EJB 1.1 JAR.

Assemble the EJB JAR file and application client JAR file into a J2EE application

1. Open the Application Assembly Tool.
2. Using the the AAT instructions, create an application:
   1. Under **File -> New -> Application**, provide properties and change the **Display name:** to (for example) *YourApp*.ear.
   2. **Import** the EJB 1.1 jar (the deployed .jar file you created in the previous step (Convert the EJB JAR file and generate the deployment code).
3. Use **File -> SaveAs** to save the .ear (enterprise application archive) file in the sought location.
4. Exit the Application Assembly Tool.

Define the J2C Resource Adapters and Connection Factories

Use the administrative console to define the J2C Resource Adapters and Connection Factories.

To define the J2C Resource Adapter, complete the following steps:

1. Open the administrative console.
2. Expand the tree view to **Resources -> J2C Resource Adapters**.
3. Right click the **J2C Resource Adapter** folder and click **New...**.
4. Enter the resource adapter properties.

   Use the **Browse [...]** button located next to the **Archive file name** on the properties dialog to browse for the .rar file that you downloaded from the Web sitein Installing the J2C (Beta) Resource Adapters, such as cicseci.rar.
5. Click **OK**. If you look on the **Node** tab you should now see your ResourceAdapter installed and configured on your node.

Notes:

- After finding and loading the .rar file, the `Connections` and `Advanced`tabs should contain the read-only contents of the ra.xml descriptor withinthe .rar file. These values will be propagated to the ConnectionFactory when you create it.
- By default, the Resource Adapter is created on the current node. Anothertabbed page in J2C Resource Adapter properties enables you to createthe Resource Adapter on additional nodes.

To define the Connection Factory, complete the following steps:

1. Open the administrative console.
2. Expand the tree view to **Resources -> J2C Resource Adapters ->** ***adapter_name*** **-> J2C Connection Factories**.
3. Right click the **J2C Connection Factories** folder and click **New...**.
4. Enter the resource adapter properties.
   - Enter a name such as cicsDeptA.
   - Modify the values on the `Connections` and `Advanced` tabbed pages as required by your installation.

     See the Resource Adapter documentation in the HowTo.html or similar file in the RAR archive for specific instructions.)
5. Click **OK**. You should now see your Connection Factory createdand associated with your Resource Adapter.

For another way to define J2C Resource Adapters and ConnectionFactories, see Specifying a J2C connectionfactory.

Install the application and resolve any References

Here, the procedure is outlined for installing the application, with linksto detailed instructions for each step. Depending on the complexity of the .earfile that you are installing, you might need to specify values for fields thatare not discussed here. See the enterprise application properties.

1. Open the administrative console.
2. Create a new application server if you do not have one already.
3. Install the enterprise application (.ear file).

Notes:

- When you get to the Mapping Resource References to Resources panel, the *resource-ref* entries as defined by your bean should be listed on the left side. For each *resource-ref*, click **Select Resource** and select the defined (J2C Connection Factory) resource that you previously configured.
- On the Deploy window, click **No** when asked whether the application should be redeployed.

Run the application

To run the application, you start and use a configured application serverjust as you would for any other IBM WebSphere enterprise application. Formore information, see the administrative overview of application servers.

6.6.42.0: Properties of J2C Resource Adapters

Key:

Applies to Java administrative console of Advanced Edition Version 4.0

Applies to Web administrative console of Advanced Single Server EditionVersion 4.0

Applies to Application Client Resource Configuration Tool

Archive File Name or Rar File

Path to the .rar file containing the module for this resource adapter

Custom Connection Properties

Connection properties that are specific to the given J2C adapter, includinga name, description, data type, and value for each property

Classpath

The class path that identifies the location of theJ2C adapter classes

Description

An optional description for your administrative records

Name

Logical name for the resource adapter, to be displayed in administrativetools

Node

The administrative node on which the J2C adapter is installed

Vendor Properties (Advanced Properties)

Connection properties that are specific to the given J2C adapter. See youradapter documentation for suggested name-value pairs.

6.6.42.0: Properties of J2C connection factories

Key:

Applies to Java administrative console of Advanced Edition Version 4.0

Applies to Web administrative console of Advanced Single Server EditionVersion 4.0

Applies to Application Client Resource Configuration Tool

Connection Timeout

If the maximum number of connections has been reached,the connection will wait for [Connection Timeout] milliseconds before abortingand throwing a ResourceAllocationException. If the maximum numberof connections has not been reached, or is set to 0, Connection Timeoutwill not be used.

If Connection Timeout is 0, the pool manager willwait indefinitely.

Description

An optional description for your administrative records

Name

Logical name for the connection factory, to be displayed in administrativetools

J2C Connection Factory

One set of connection configuration values. Application components, suchas enterprise beans, have resource-ref descriptors that refer to a specific connection factory, not to the resource adapter.

You can think ofthe connection factory as a holder of a list of configuration properties.In addition to the arbitrary set of configuration properties defined bythe vendor of the resource adapter, there are several standard configurationproperties that apply to the connection factory. These standard propertiesare used by the J2C connection pool manager in the application server runtimeand are not known by the vendor supplied resource adapter code.

J2C Resource Adapter

This read-only field identifies the J2C Resource Adapter on which this J2C Connection Factory is based.

JNDI binding path

The JNDI name that will be used to bind the J2C ConnectionFactory into the application server's name space. The default valuefor this field is eis/*Name*, where *Name* is the logical nameof the connection factory.

Maximum Connections

The maximum number of managed connections that can be created by a particularManagedConnectionFactory. After this number is reached, no new connectionsare created and either the requester waits for theConnection Timeout or a ResourceAllocationException is thrown. If Maximum Connections is set to 0. The number of connections can grow indefinitely. Maximum Connections must be larger than or equal to Minimum Connections.

Minimum Connections

The minimum number of managed connections to maintain. If this number isreached, the garbage collector will not discard any managed connections.Note, if the actual number of connections is lower than the value specifiedby the minimum connections settings, no attempt will be made to increasethe number of connections to the minimum. Minimum Connections must be lessthan or equal to Maximum Connections.

Pool Name

The name used by the pool manager to group managed connections createdby different ManagedConnectionFactories.

Reap Time

Number of seconds between runs of the garbage collector. The garbage collectordiscards all connections that have been unused for the value specifiedby the unused timeout.

To disable thegarbage collector, set the reap time to 0 or set the unused timeoutto 0.

Re-Authentication Support

Whether the resource adapter supports re-authenticationof existing managed connection instances

Subpool Name

The name used by the pool manager to group managed connections createdby different ManagedConnectionFactories within a particular pool.

Unused Timeout

Number of milliseconds after which an unused connection is discarded. Settingthis value to 0 disables the garbage collector.

6.6.42.1: Administering J2C resources with the Java administrative console

Use the Java administrative console to administer enterprise applications.

Work with resources of this type by locating them in the treeview:

WebSphere Administrative Domain -> Resources -> J2C Resource Adapters-> ***adapter_name*** **-> Connection Factories**

The connection factory instances will be displayed in the detailsview.

6.6.42.1.1: Configuring J2C connection factories with the Java administrative console

A J2C connection factory is an object used to access application resources.

To create a J2C connection factory, click **Console -> Wizards -> Create J2C Connection Factory** from the console menu bar. This leads to the J2C Connection Factory task wizard.

You can also use:

- **Console -> New -> J2C Resource Adapter**
- **Console -> New -> J2C Connection Factory**

Using the J2C Connection Factory wizard

You use a J2C Connection Factory wizard to define a connection factory that givesaccess to application resources. Using the wizard sets basic properties pertaining to J2C connection factories. To set other property values for J2C connection factories, use the properties dialogs for J2C resources.

- Naming the J2C connection factory
- Selecting to use an existing J2C resource adapter or to create a new one
- Creating a new J2C resource adapter
- Completing the J2C connection factory

Naming the J2C connection factory

On the Specifying a Connection Factory panel, name and describe your connection factory session, and then click **Next**.

Selecting to use an existing J2C resource adapter or to create a new one

On the Specifying a J2C Resource Adapter panel, select to use an existing J2C resourceadapter or to create a new J2C resource adapter.

A resource adapter represents a library that supports connections from an application to an enterprise information system (EIS) resource.

If you opt to use an existing adapter, select one from the drop-down list, and then click **Next**.

Creating a new J2C resource adapter

If you opt to create a new J2C resource adapter, the wizard shows the Creating a J2C Resource Adapter panel. Name and describe the resource adapter, and specify a full path name for the adapter archive. Then, click **Next**.

View J2C resources properties help

Completing the J2C connection factory

The final panel lists the J2C connection factory name and resource adapter name.

If you do not want to change the values specified, click **Finish**. Thewizard will define a J2C connection factory with the values you specified, and display a message indicating where the factory resides on your system. If the wizard encounters an error, a message will display explaining why a factory could not be defined.

To change the values specified, click **Back** to return to the appropriatepanel(s), make any needed changes, and then click **Finish** on this panel.

6.6.42.1.2: Installing and uninstalling J2C adapters on nodes, with the Java administrative console

To install an adapter:

1. Locate a J2C adapter instance in the tree view.
2. Click the **Nodes** tabbed page.
3. Click the **Install New** button.
4. Configure the new adapter:
    1. Select a node on which to install the adapter.
    2. Specify or browse for the classpath directory (see J2C adapter properties).
    3. Click **Install**.
5. Click **Apply** on the J2C adapter properties view.

To uninstall an adapter:

1. Locate a J2C adapter instance in the tree view.
2. Click the **Nodes** tabbed page.
3. Select a adapter from the list of adapters.
4. Click **Uninstall**.
5. Click **Apply** on the J2C adapter properties view.

You can also click a node instance in the tree view todisplay a similar configuration dialog, only the **Classpath directory** field is named **Driver file** instead.

6.6.42.6: Installing the Connector Architecture for WebSphere Application Server (J2C)

Before you can use this connectivity,you must install the Connector Architecture (J2C) runtime and at least one J2C (Beta)Resource Adapter.

Installing and uninstalling the Connector Architecture (J2C) runtime

After you have installed the base IBM WebSphere Application Server product, install the Connector Architecture(technology preview) runtime.

1. Insert the WebSphere Application Server product CD and change to the root directory.
2. From a command prompt, run the script:

   ```
   WebSphere/extras/techpreview/connector/installJ2C.bat|sh
   ```

Running the script will install the appropriate .jar files and modify any environment settingsthat are required to support the Connector Architecture (technology preview) runtime.

At the end of the installation process, the utility `uninstallJ2C.bat`is created in the *product_installation_root* directory. If youdecide later to removethe J2C runtime from your IBM WebSphere Application Server environment,run this utility from a command prompt.

Installing the J2C (Beta) Resource Adapters

Note, the Resource Adapters are labeled "Beta" level, while the overall connectivity isat the "technical preview" level. This is primarily a terminology difference, in this particularcase.

The following J2C (Beta) Resource Adapters are available for IBM WebSphere Application Server:

- CICS Transaction Gateway (CTG) V4.0
- ECI and EPI
- IBM WebSphere Adapter for IMS 1.0
- HOD Connector Architecture (J2C) 6.0
- IBM WebSphere Adapter for mySAP.com V1.0
- IBM WebSphere Adapter for PeopleSoft V1.0
- IBM WebSphere Adapter for Oracle Applications V1.0
- IBM WebSphere Adapter for J.D. Edwards V1.0

ℹ All of these initial J2C Resource Adapters are either 0 PhaseCommit (0PC) or 1 Phase Commit (1PC) capable connectors. None of theseJ2C Resource Adapters are 2 Phase Commit (2PC) XA capable.

To install a J2C (Beta) Resource Adapter, complete the following steps:

1. Download the specific J2C (Beta) Resource Adapter package from the WebSphere Application Server Web site.
2. Each of the J2C (Beta) Resource Adapters has its own documentation for installation.
3. After the installation is complete, the J2C (Beta) Resource Adapter's archive files(.rar files) will be somewhere in the installation subdirectory structure.

Next, administer your J2C Resource Adapter, which willinclude identifying the location of the .rar file.

6.6.43: Administering references

References are logical names used to locate external resources for enterprise applications. References include EJB references and resource references (for external resources such asdatabases and messaging systems).

References are defined in the application's deployment descriptor file by using the ApplicationAssembly Tool. At deployment, the references are bound to the physical location (JNDI name) of the resource in the target operational environment.

6.6.43.0: Properties related to references

References are defined in an application's deployment descriptor by using the Application Assembly Tool. Refer to the assembly propertieslisted in the related information.

6.6.43.0.1: Assembly properties for EJB references

An EJB reference is a logical name used to locate the home interface of anenterprise bean used by the application. At deployment, the EJBreference is bound to the enterprise bean's home in the targetoperational environment. The container makes the application's EJBreferences available in a JNDI naming context. It is recommended thatreferences to enterprise beans be organized in the ejb subcontext of theapplication's environment (in java:comp/env/ejb).

Name (Required, String)

Specifies the JNDI name of the enterprise bean's home, relative tothe java:comp/env context. For example, if ejb/EmplRecord isspecified, the referencing code looks up the enterprise bean's home atjava:comp/env/ejb/EmplRecord. This JNDI name is a JNDI name aliasused by the code (the actual JNDI name is specified on the Bindingtab).

Link

Used to link an EJB reference to an enterprise bean in the current module(the same module as the one making the reference) or in another module withinthe same encompassing J2EE application. The value of this property isthe name of the target enterprise bean. (The target enterprise bean canbe in any EJB module in the same J2EE application as the referencingmodule.) To avoid having to rename enterprise beans to have uniquenames within an entire J2EE application, specify the path name of the EJB JARfile containing the referenced enterprise bean and append the targetbean's name, separated by a # symbol, for example,...products/product.jar#ProductEJB. Thepath name is relative to the referencing module's JAR file. If alink is not specified, the reference must be resolved to a JNDI name duringinstallation.

Home (Required, String)

Specifies the fully qualified name of the enterprise bean's homeinterface. An example iscom.ibm.ejbs.EmplRecordHome.

Remote (Required, String)

Specifies the fully qualified name of the enterprise bean's remoteinterface. An example iscom.ibm.ejbs.EmplRecord.

Type (Required, String)

Specifies the expected type of the referenced enterprise bean. Thevalue must be either Entity or Session.

Description

Contains text describing the EJB reference.

JNDI Name

Binding information is used by the run-time environment to resolve thelocation of a resource. For EJB references, the JNDI name must matchthe JNDI name of the enterprise bean as specified on the Binding tab in theEJB module containing the bean.

6.6.43.0.2 Assembly properties for resource references

A resource reference declares a logical name used to locate a connectionfactory object. These objects define connections to external resourcessuch as databases and messaging systems. The container binds thesereferences to actual resource manager connection factories in the targetoperational environment. It is recommended that resource references beorganized in the subcontexts of the application's naming environment,using a different subcontext for each resource manager type. Forexample, all JDBC DataSource references can be declared in thejava:comp/env/jdbc context. All JMS connection factories can bedeclared in the java:comp/env/jms subcontext. All JavaMailconnection factories can be declared in the java:comp/env/mailsubcontext. All URL connection factories can be declared in thejava:comp/env/url subcontext. All J2C connection factories can bedeclared in the java:comp/env/eis subcontext.

Name (Required, String)

Specifies the JNDI name used to retrieve a connection factory for aresource manager (relative to the java:comp/env context). Forexample, if the name specified is jdbc/EmployeeAppDB, the referencing codelooks up the factory under the namejava:comp/env/jdbc/EmployeeAppDB.

Type (Required, String)

Specifies the Java programming language type of the connection factoryobject. For obtaining JDBC API connections, usejavax.sql.DataSource. For obtaining JMS connections, usejavax.jms.QueueConnectionFactory orjavax.jms.TopicConnectionFactory. For obtaining JavaMailconnections, use javax.mail.Session. For obtaining URLconnections, use java.net.URL.

Authentication (Required, String)

Specifies whether the enterprise bean (or servlet) code signs onprogrammatically to the resource manager, or whether the container signs on tothe resource manager on behalf of the bean (or servlet). In the lattercase, the container uses information that is supplied by the deployer.The value of this field must be one of the following: Application orContainer (for enterprise beans) and Servlet or Container (for Webapplications). Note for J2C (Connector Architecture for WebSphereApplication Server) resources: This property is valid only for sessionbeans. There is no CMP support yet. WebSphere Application Serversupports only component-managed sign-on (Option C in the J2EE/ConnectorArchitecture specification) in this release. As a result, the value forthis property is ignored and processed as Application. This means thateither the session bean needs to pass the user ID and password credentials onthe getConnection call or the J2C Connection Factory needs to have its user IDand password fields filled in (this is done by using the WebSphereadministrative console).

Description

Contains text describing the connection factory object.

JNDI name

Specifies the name of the connection factory in the global JNDInamespace.

6.6.43.0.3: Assembly properties for security role references

Role name (Required, String)

Specifies the name of a security role reference used in the applicationcode, for example, "boss." The AccountBean can make a decision based onwhether the user executing a method is granted the role of a "boss."

Link

Specifies the name of a security role defined in the encompassingapplication. The role reference will be linked to this name. Forexample, the AccountBean code uses a role named "boss." The AccountBean is a part of an enterprise application, FinanceApp, that has a role named"Manager." If the link specifies "Manager," then when the bean makes acall to isCallerInRole("boss"), the result will be true if and only if theuser, who invoked the method, has been granted the FinanceApp's Managerrole. The security role reference is the name used by an applicationcomponent (module), and the link name is the name defined in the deploymentdescriptor of the encompassing application. The link maps the name usedin the component to a corresponding name in the application.

Description

Contains text describing the security role.

6.6.44: Administering Web services (overview)

Please see the related information links. A main tool for administering Web services is the XML-SOAP administrative tool.

6.6.45: Administering WebSphere plug-ins for Web servers

The WebSphere Application Server works with a Web server to handle requests for Web applications. The Web server and application server communicate usingthe WebSphere plug-in for the Web server.

When you install WebSphere ApplicationServer, it modifies the Web server configuration file automatically to establish the plug-in. The exception is for installationsusing Domino Server, in which you need to perform manual stepsto update the Web server configuration file after installing WebSphere ApplicationServer. For more information, see "Modifications to Web server configurations during product installation."

Administering Web servers

Refer to your Web server documentation as theultimate source of information about administering your Web server. See also the subtopics of this article.

Ensure that Web servers are configured to perform operations required by Web applications, such asGET and POST. Typically, this involves setting a directive in the Web server configuration file (such as httpd.conf for IBM HTTP Server). Refer to the Web server documentation for instructions. If an operation is not enabled when a servlet or JSP page requiring the operation is accessed, anerror message is displayed, such as this one from IBM HTTP Server:

```
HTTP method POST is not supported by this URL.
```

The internal HTTP transport and the Web server plug-in

A Web server and WebSphere plug-in for the Web server are required for usein a production environment, for performance reasons. But strictly speaking, they are not required in order to start the application server or the administrative console. In a test or development environment, you might wantto use the internal HTTP transport described in the transportadministration overview, instead ofthe Web server plug-in and Web server.

The remainder of this article and its sub-topics (6.6.45.*) discuss theWeb server plug-in.

Administering WebSphere plug-ins for Web servers

Version 4.x introduces the HTTP transport plug-ins as the preferredmethod of communication between the Web server and the application server. TheHTTP plug-in introduces the following advantages:

- XML-based configuration file
- Standard protocol recognized by firewall products
- Security using HTTPS, replacing proprietary OSE over SSL

The remainder of this article discusses the main plug-in tasks and when to perform them:

- Manually triggering a regeneration of the plug-in configuration
- Manually editing the plug-in configuration

Manually triggering a regeneration of the plug-in configuration

The WebSphere administrative console supplies a manual way to force plug-in configurations to update (regenerate) themselves. For a summary of the occassions on which to update the plug-in configuration, see the instructions for regenerating the plug-in.

Time required for plug-in regeneration

Regenerating the configuration might take a while to complete. After it is finished, all objects in the administrative domain will use their newest settings, of which the Web server is now aware.

Whether triggered manually or occurring automatically, plug-in regeneration requires about 30 - 60 seconds to complete, when the application server product is on the same physicalmachine (node) as the Web server. In other cases, it takes more time.

The delay is important because it determines how soon the new plug-in configuration will take effect. Suppose you add a new served path for a servlet, then regenerate the plug-in configurations. The regeneration requires 40 seconds, after which a user should be able to access the servlet by the new served path. You can see how dynamic plug-in configuration enables you to make newly configured resources availableto users right away.

For an HTTP plug-in, the length of the delay is determined by the Refresh Intervalattribute of the Config element in the plugin-cfg.xml file. The plug-in polls the disk at this interval to see whether the configuration has changed. The defaultinterval is 60 seconds.

To regenerate the plug-in configurations requires twice the refresh interval.

In a development environment in which you are frequently changing settings in the administrative console, it is recommended you set the refresh interval to 3 to 5 seconds.

In a production environment, set a longer refresh interval, perhaps as long as 30 minutes, depending on the frequency of changes.

Example of regenerating the plug-in configuration

After using the administrative console to make configuration changes thatinvolve the served paths of Web applications,manually trigger the regeneration of the plug-in configuration (or manually edit the file if that is what you have been doing).

When you trigger the regeneration of the plug-in configuration:

1. The Web container containing the Web application registers the configuration changes to those objects.
2. The Web container unloads the Web application, then loads it again, causing theWeb application to adopt the new settings.

   Note that the state and data of any servlets running in the Web application will be lost, unless the Web application usually persists data (for example, it savesservlet-generated session and user profile data in a database).
3. The plug-in configuration is regenerated. The Web server is aware of the newWeb application configuration. If a user requests the servlet using the path specified by the newWeb resource, the request should be successful.

Manually editing the plug-in configuration

The Web server plug-in property reference describes the location of the plug-in configuration files.

Use care when you are hand-editing the HTTP plug-in configuration, because your changes will be overwritten if the plug-in configuration is regenerated. However, regenerating the plug-in configuration cannot guarantee a correct configuration filefor advanced configuration scenarios. If you seeing strange behavior after triggering aregeneration of the plug-in configuration, consider manuallyediting the configuration.

6.6.45.0: Properties of WebSphere plug-ins for Web servers

- Finding the plug-in configuration files
- HTTP plug-in properties
- See also Transport Properties

Finding the plug-in configuration files

The working or active versions of the HTTP plugin-cfg.xml file resides in the directory:
`product_installation_root``/config`

HTTP plug-ins

The plugin-cfg.xml file includes the following elements and attributes:

Config (exactly one)

This element starts the WebSphere HTTP plug-in configuration file. It contains all other elements and attributes of the configuration.

Refresh interval (exactly one)

The time interval (in seconds) at which the plug-in should check the configuration file to see if updates or changes have occured. The plug-in checks the file for any modifications that have occurred since the last time the plug-in configuration was loaded.

In a development environment in which changes are frequent, a lower setting than the default setting of 60 is recommended. In production, a higher value then the default is a good idea since updates to the configuration will not occur as often. If the plug-in reload fails for some reason, a message is written to the plug-in log file and the previous configuration is used until the plug-in config reload is successful. If you are not seeing the changes you made to your plug-in configuration, check the plug-in log file for indications of the problem.

Log (zero or one for each Config)

The log describes the location and level of log messages that should be written by the plug-in. If a Log is not specified within the Config, then in some cases log messages will be written to the Web server error log.

Name (exactly one)

The fully qualified path to the log file to which the plug-in will write error messages.

If the file does not exist then it will be created. If the file already exists then it will be opened in append mode and the previous plug-in log messages will remain.

LogLevel (zero or one for each Log)

The level of detail of the log messages that the plug-in should write to the log. Values for this attribute are one of the following:

- Trace
- Warn
- Error

If a LogLevel is not specified with the Log, then the default value is Error.

Trace allows you to see the steps in the request process in detail. Warn and Error means that only information about abnormal request processing will be logged.

Be careful when setting the level to Trace. A lot of error messages are logged at this level which can cause the disk space to fill up very quickly. A Trace setting should never be used in a normally functioning environment as it affects performance.

```
<ServerGroup name="Servers">    <Server name="Server1">         <Transport Hostname="localhost"
Port="9080" Protocol="HTTP"/>           <Transport Hostname="localhost" Port="9443"
Protocol="HTTPS">                  <Property name="Keyring"
value="c:/WebSphere/AppServer/keys/keyring.kdb"/>                    <Property name="Stashfile"
value="c:/WebSphere/AppServer/keys/keyring.sth"/>   </Server></ServerGroup>
```

ServerGroup (one or more for each Route)

A group of servers that are generally configured to service the same types of requests.

In the simplest case, the server group will contain only one server definition. In the case inwhich more than one server is defined, the plug-in will load balance across the defined servers using either a Round Robin or a Random algorithm. The default is Round Robin.

Name (exactly one for each ServerGroup)

The logical or administrative name to be used for this group of servers.

LoadBalance (one for each ServerGroup)

The default load balancing type is Round Robin

The Round Robin implementation has a random starting point. This means that the first server picked will be done so randomly and then round robin will be used from that point forward. This is so that in multiple process based Web servers all of the processes don't start up by sending the first request to the same application server.

RetryInterval (one for each ServerGroup)

An integer specifying the length of time that should elapse from the time that a server is marked down to the time that the plug-in will retry a connection. The default is 60 seconds.

RemoveSpecialHeaders (one for each ServerGroup)

The plug-in adds special headers to the request before it is forwarded to the application server. These headers store information about the request that will need to be used by the application. By default the plug-in will remove these headers from incoming requests before adding the headers it is supposed to add.

The value can be true or false.

Setting the attribute to false introduces a potential security exposure by not removing headers from incoming requests.

Server (one or more for each ServerGroup)

A WebSphere Application Server instance that is configured to handle requests routed to it given the routing rules of the plug-in configuration. The Server should correspond to an application server running on either the local machine or a remote machine.

Name (exactly one for each Server)

The administrative or logical name for the server.

CloneID (zero or one for each Server)

If this unique ID is present in the HTTP Cookie header of a request (or the URL if using URL rewriting), the plug-in routes the request to this particular server, provided all other routing rules are met. If a CloneID is not specified in the Server, then Session Affinity is not enabled for this server.

Used in conjunction with Session Affinity. When this attribute is set the plug-in will check the incoming cookie header or URL for jsessionid. If the jsessionid is found then the plug-in will look for a clone ID or clone IDs. If clone IDs are found and a match is made to this attribute then the request will be sent to this server rather than being load balanced across the server group.

If you are not using Session Affinity then it is best to remove these clone IDs from the configuration because there is added request processing in the plug-in when these are set. If clone IDs are not in the plug-in then it is assumed that session affinity is not on and the request is load balanced across the server group.

WaitForContinue (one for each Server)

This true or false attribute allows you to specify whether to use the HTTP 1.1 100 Continue support before sending the request content to the application server. The default is false.

This function is necessary when configuring the plug-in to work with certain types of proxy firewalls. The plug-in by default does not wait for the 100 Continue response from the application server before sending the request content because it is a performance hit.

Transport (one or more for each Server)

The transport for reading and writing requests to a particular WebSphere application server instance. The transport provides the information needed to determine the location of the application server to which the request will be sent. If the Server has multiple transports defined to use the same protocol, the first one will be used.

It is possible to configure the Server to have one non-secure transport and one thatuses SSL. In this configuration, a match of the incoming request protocol will be performed to determine the appropriate transport to use to send the request to the application server.

Hostname (exactly one for each Transport)

The hostname or IP address of the machine on which the WebSphere application server instance is running.

Port (exactly one for each Transport)

The port on which the WebSphere application server instance is listening.

Protocol (exactly one for each Transport)

The protocol to use when communicating over this transport -- either HTTP or HTTPS.

Property (zero, one, or more for each Transport)

When the Protocol of the Transport is set to HTTPS, use this element to supply the various initialization parameters, such as keyfile and stashfile.

Name (exactly one for each Property)

The name of the Property being defined. Supported names recognized by the transport are keyring, stashfile, and password.

Value (exactly one for each Property)

The value of the Property being defined.

```
<VirtualHostGroup name="Hosts"> <VirtualHost name="www.x.com"/> <VirtualHost name="www.x.com:443"/>
<VirtualHost name="*:8080"/>    <VirtualHost name="www.x.com:*"/>       <VirtualHost
name="*:*"/></VirtualHostGroup>
```

VirtualHostGroup (zero, one, or more for each Config)

A group of virtual host names that will be specified in the HTTP Host header. Enables youto group virtual host definitions together that are configured to handle similar types of requests.

Name (exactly one for each VirtualHostGroup)

The logical or administrative name to be used for this group of virtual hosts.

VirtualHost (one or more for each VirtualHost)

The name used for a virtual or real machine used to determine if incoming requests should be handled by WebSphere Application Server or not. Use this element to specify hostnames that will be in the HTTP Host header that should be seen for requests that need to be handled by the application server. You can specify specific hostnames and ports that incoming requests will have or specify a * for either the hostname, port, or both.

Name (exactly one for each VirtualHost)

The actual name that should be specified in the HTTP Host header in order to match successfully with this VirtualHost.

The value is a hostname or IP address and port combination, separated by a colon.

The plug-in can be configured to route requests to the application server based on the incoming HTTP Host header and port for the request. Name allows you to specify what those combinations are.

Wildcarding is available for this attribute. The only acceptable solutions are either a * for the hostname, a * for the port, or a * for both. A * for both means that any request will match this rule. If no port is specified in the definition the default HTTP port of 80 is assumed.

```
<UriGroup name="Uris">  <Uri name="/servlet/snoop"/>    <Uri name="/webapp/*"/> <Uri
name="*.jsp"/></UriGroup>
```

UriGroup (one or more for each Config)

A group of uris that will be specified in the HTTP request line. They will be able to be handled by the same application server. The route will compare the incoming uri with the uris in the group to determine if the application server will handle the request.

Name (exactly one for each UriGroup)

The logical or administrative name for this group of uris.

Uri (one or more for each UriGroup)

The virtual path to the resource that will be serviced by WebSphere Application Server. Each Uri specifies the incoming URLs that need to be handled by the application server. Wildcarding is available in these definitions.

Name (exactly one for each Uri)

The actual string that should be specified in the HTTP request line in order to match successfully with this Uri. Wildcarding within the uri definition is acceptable. You can specify rules such as *.jsp or /servlet/* to be handled by WebSphere Application Server.

```
<Route VirtualHostGroup="Hosts" UriGroup="Uris" ServerGroup="servers/>
```

Route (one or more for each Config)

A request routing rule by which the plug-in will determine if an incoming request should be handled by a WebSphere application server.

The route definition is the central element of the plug-in configuration. It specifies how the plug-in will handle requests based on certain characteristics of the request. The route definition contains the other main elements: a required ServerGroup, and either a VirtualHostGroup, UriGroup, or both.

Using the information that is defined in the VirtualHostGroup and the UriGroup for the route, the plug-in determines if the incoming request to the Web server should be sent on to the ServerGroup defined in this route.

VirtualHostGroup (zero or one for each Route)

The group of virtual hosts that should be used in route determination. The incoming host header and server port are matched to determine if this request should be handled by the application server.

It is possible to omit this from the route definition. If it is not present then every request will match during the virtual host match portion of route determination.

UriGroup (zero or one for each Route)

The group of uris to use for determining the route. The incoming uri for the request are matched to the defined uris in this group to determine if this request should be handled by the application server.

It is possible to omit this from the route definition. If it is not present than every request will match during the uri match portion of route determination.

ServerGroup (exactly one for each Route)

The server group to which to send request that successfully match the route.

The server group that should be used to handle this request. If both the uri and the virtual host matching is successful for this route then the request will be sent to one of the servers defined within this server group.

6.6.45.0.1: Modifications to Web server configuration files during product installation

Based on the plug-ins selected, the following modifications are made to theWeb server configuration files during the installation process. For the exact Web server brands, products, and versions supported by your version and editionof WebSphere Application Server, see the product prerequisites Web site.

The purpose of thisarticle is to help you identify the additions in the event that you need to restorethe configuration file to its original form, and do not have a backup copy. Refer alsoto the updated configuration file itself, as it could contain slightly different and more current modifications as this documentation ages and product fix packsare released.

In the actual configuration file, paths shown in the example below, such as:

```
C:/WebSphere/AppServer/bin/mod_app_server_http.dll
```

should reflect the *product_installation_root* foryour particular installation and operating system. The default Windows installation root is used in the example.

If installing on a UNIX-based system and you findthat few of the modifications were made to the Web server configurationfile, try logging on directly as root to perform the installation. Installation problems of this nature have been reported when usingsu from another account, rather than the root account.

Apache (httpd.conf)

For the HTTP plug-in:

```
LoadModule app_server_module C:/WebSphere/AppServer/bin/mod_app_server_http.dll
```

Apache (srm.conf)

For the HTTP plug-in:

```
Alias /IBMWebAS/samples/  "C:/WebSphere/AppServer/samples/"Alias /IBMWebAS/
"C:/WebSphere/AppServer/web/"NcfAppServerConfig BootFile
C:\WebSphere\AppServer\config\plugin-cfg.xml
```

Domino (httpd.cnf)

For the HTTP plug-in, updating the configuration file is a manual process asdescribed in "Manually updating the Domino Web server configuration file."

IBM HTTP Server (httpd.cnf)

For the HTTP plug-in:

```
Alias /IBMWebAS/samples/  "C:/WebSphere/AppServer/samples/"Alias /IBMWebAS/
"C:/WebSphere/AppServer/web/"LoadModule ibm_app_server_module
C:/WebSphere/AppServer/bin/mod_app_server_http.dllNcfAppServerConfig BootFile
C:\WebSphere\AppServer\config\plugin-cfg.xml
```

Netscape or iPlanet (obj.conf)

For the HTTP plug-in:

```
Init fn="load-modules" funcs="as_init, as_handler,
as_term"shlib="C:/WebSphere/AppServer/bin/ns41_http.dll"Init fn="init_exit"
bootstrap.properties="C:/WebSphere/AppServer/config/plugin-cfg.xml"Service fn ="as_handler"
```

6.6.45.5: Controlling where the WebSphere plug-ins for Web servers are installed

Sometimes, you might want to install the WebSphere plug-ins for Web servers tolocations other than their default locations.

Installing Web server plug-ins to non-default locations

Depending on the operating system, when you select to install a Web server plug-in, the plug-in may be installed silently if the Web server is installed in a standard location for that Web server brand. In such a case, the WebSphere Application Server installation does not prompt you for the configurationfile location.

If you have installed two Web servers on the same machine, the "standard"place for installing the plug-in might be okay for one of the servers, but will not succeed in installing the plug-in for the server that is in thenon-standard location.

For example, suppose IBM HTTP Server "installation 1" is installed in the standard directory in which the Web server is installed on Solaris, /opt/IBMHTTPD. When the WebSphere plug-in for this first Web server is installed, it will silently modify the httpd.conf file in /opt/IBMHTTPD/conf, which is okay.

But now suppose that you installed IBM HTTP Server "installation 2" in /opt/IHS2. When the Web server plug-in for this second Web server is installed, it too will be installed in httpd.conf in /opt/IBMHTTPD/conf.

One way around this is to manually edit the httpd.conf file of the second IBM HTTP Server installation, rather than installing the WebSphere plug-in. You couldcopy the modified configuration file from the first Web server and make itthe configuration file of the second server. Of course, this approach assumesthat you have not customized the configuration file of the second server, andfind it acceptable for its configuration to match that of the first server.

Another way is to install both Web servers in some non-standard place,such as /opt/IHS1 and /opt/IHS2. When the WebSphere Application Server installationprogram cannot find either httpd.conf file, you will be prompted for whichone to edit. Specify the location of one of the Web server files. Then repeatthe plug-in installation, specifying the other location this time.

Installing WebSphere plug-ins on non-default IIS servers

When the product installs the WebSphere plug-in for Microsoft Internet Information Server (IIS), it assumes the user wants to attach the plug-in to the default IIS server. The following instructions explain how to attach the plug-in to an IIS server *other than* the default.

The procedure might be necessary for a user implementing multiple Web sites that separate the pages they serve along some logical boundary, such as security level. Follow the steps to allow a newly defined site (using an IIS instance other than the default) to exercise servlets in conjunction with an existing site or sites.

The instructions apply to IIS Versions 4.x and 5.x.

1. Use the Internet Service Manager (from Microsoft IIS) to create a new site with a name, port number, and base subdirectory.
2. Open the WebSphere Application Server administrative console.
3. Configure the virtual host to containan alias for the port number used by the site.
4. Create a virtual directory for the new site.
    1. Open the Internet Service Manager for IIS.

2. Select the new site in the Tree View.
3. Right-click to display a menu. Select New -> Virtual Directory.
4. Ensure the values are set appropriately:
    - The name of the virtual directory should be set to SEPLUGINS (using all capital letters, as shown).
    - The physical path should be set to the WebSphere bin directory (such as c:\WebSphere\AppServer\bin on Windows NT).

      Note, the directory must have the EXECUTE permission set, but setting any other permissions (or allowing it to inherit other permissions) is a security risk.
5. Start the new site.
6. Issue a browser request to verify that the configuration works, such as:

   http://mymachine:8080/servlet/snoop
7. The administrator must ensure that the SEPLUGINS retains its EXECUTE permission.

   It is possible for virtual directories under a site to inherit properties from the site. The administrator must ensure that the SEPLUGINS virtual directory does not inherit permissions from changes to the site, if those changes involve withdrawing the EXECUTE permission.

6.6.45.6: Regenerating the Web server plug-in configuration

Sometimes you might want to trigger (manually) the WebSphere plug-in toregenerate its configuration. For example, you might installed or removedan enterprise application, or added or removed servlets and mappings froma particular application. Failure to regenerate the plug-in after introducinga new application will likely result in a "404 File Not Found" error when a user tries to access the new Web application.

> ⚠ This task can overwrite manual configuration changes that youmight want to preserve instead. Before performing this task,understand its implications as described in the article about administering Web server plug-ins.

You can run the following script:

```
product_installation_root/bin/GenPluginCfg.sh|bat
```

You can also use the administrative node to manually trigger an update of the configuration forthe WebSphere plug-in for the Web server you are using:

1. Locate the node in the tree view.
2. Right-click it, then select **Regen Webserver Plugin**.

To configure automatic regeneration:

1. Display the application server properties.
2. Select the **Custom** tabbed page.
3. Edit **Automatic Generation of Plug-in**.
4. Save your changes.

6.6.45.7: What to do after changing Web server ports

Suppose you change a Web server port. You will need changethe WebSphere virtual host configurations associated with the Webserver.

Specifically, modify the virtual host aliases forthat Web server to reflect the new port number. Afterdoing so, stop the application servers associated withthe affected virtual hosts and start them again.

For example, if you change the Web server port to 9082:

1. For each virtual host whose configuration contains aliases for the Web server,
2. use the administrative console to display virtual host properties.
   1. Locate the alias settings.
   2. Append :1111 to every alias pertaining to the Web server with the changed port.
   3. Save the changes.
3. Identify the application servers associated with the virtual host.
4. Stop the application servers and start them again.
5. Regenerate the HTTP plug-in configuration.

A virtual host configuration assumes the Web server port tobe 80, unless otherwise specified. If you change the portfrom some other number to 80, either append :80 tothe virtual host aliases, or make sure you remove theprevious port number from any aliases.

6.6.45.8: Checking your IBM HTTP Server version

1. Change directory to the installation root of the Web server. For example, this is /opt/IBMHTTPD on a Solaris machine.
2. Locate the file named version.signature.
3. At a system command prompt in that directory, enter:

   ```
   more version.signature
   ```

The version will be reported as IBM HTTP Server 1.3.12.*xx* .

6.6.45.9: Manually updating the Domino Web server configuration file

At the time of this writing, you must update the Domino Web server configuration manually, after installing IBMWebSphere Application Server, and before running the product. See"Administering WebSphere plug-ins for Web servers" for more information aboutconfiguration file updates.

Note, the following third party product instructions are subject to change, withoutnotice to IBM. Refer to your Web server documentation for the latest and most detailed instructions.

1. Start the Domino server.
2. Start the Domino administrative interface.
3. In the Domino administrative interface, edit the server.
    1. Click the **Configuration** tab.
    2. On the left side, click **Server** -> **All Server Documents**.
    3. When a list of servers is displayed on the right side, double-click the Domino server whose configuration file you need to update.
    4. When the **Edit Server** button is displayed, click it to display the server properties.
    5. Click the **Internet Protocols** tab.
    6. Place your cursor between the brackets next to **DSAPI filter file names**. Type the full path to the WebSphere DSAPI plug-in, such as:

       ```
       product_installation_root/bin/domino5_http.dll
       ```

    7. Click the **Save and Close** button.
4. Close the Domino administrative interface.

6.6.46: Administering WebSphere administrative servers

The administrative server manages configurations and states ofapplication servers, applications, and other resources in the WebSphereadministrative domain. It keeps its administrative data in a database that you define during product installation.

- Configuring the administrative server and database
- Creating the database tables and default configuration
- Establishing centralized administration (shared administrative database)
- What to do if you must drop the administrative database tables
- Miscellaneous topics

Configuring the administrative server and database

There are a couple of ways to configure the administrative server, andprovide information about its administrative database.

Administrative configuration file

The administrative server obtains its own configuration from the properties inthe administrative configuration file. Some values are set during product installation,while others are primarily for modification later.

Adding Java command line arguments for the administrative server

The admin.config file contains many administrative server properties you can set. In addition, you might want to pass the administrative serversome generic Java command line arguments.

In the admin.config file, add the properties and their values as arguments on the com.ibm.ejs.sm.util.process.Nanny.adminServerJvmArgs property. Add them using standardJava command line syntax:

```
-D property_name=property_value
```

For example, this setting passes performance-related Java parameters to the Java process constituting the administrative server:

```
com.ibm.ejs.sm.util.process.Nanny.adminServerJvmArgs=-ms32m -mx128m
```

The *admin.config* file contains both default properties that apply to most configurations and non-default properties.

Creating the database tables and default configuration

In most cases (see below for special cases), the administrative databasetables are created, and populated with a default configuration, when youinstall WebSphere Application Server. During product installation, youmust provide information regarding the administrative database youwould like the installation's administrative server to use.

Settings in the administrative configuration file record the databaseinformation, and determine whether the database tables are created and (or)populated with the default configuration.

Special cases requiring additional steps after product installation

In some cases, the database tables are not created automatically duringproduct installation, or the tables are not populated with the defaultconfiguration. If you need to, trigger the actions manually by setting propertiesin the administrative configuration file and starting the administrativeserver again.

⚠ Users of WebSphere Application Server *Advanced Edition* Version 4.0 on Solaris and AIX operating systems with an Oracle database as the administrative database should note the following. In those cases,the default server is not created when you start the administrative server for thefirst time after installing the product. To create the default server and other defaultresources, set:

```
install.initial.config=true
```

in the administrative configuration file and start the administrative again.

⚠ If you use an Oracle database user ID other than EJSADMIN, the default value assigned during the WebSphere ApplicationServer installation, you will need to edit the administrative configuration file.

Ensure that the user ID and schema lines reflect your actual database user IDand password:

```
com.ibm.ejs.sm.adminServer.dbuser=your_user_namecom.ibm.ejs.sm.adminServer.dbSchema=your_schema_name
```

Then start the administrative server.

Establishing centralized administration

Multiple administrative servers can be configured to use the same administrative database,allowing configuration data to be kept in a centralized place for administration of WebSphereApplication Server across multiple machines (known as administrative nodes). In a successfulmultiple node configuration, the administrative console tree view will show each of the administrative nodes and its contents.

Configuring the 2nd through Nth machine in a cluster

Do not create the database tables or populate them with the default configuration.

When you are setting up a cluster of administrative nodes, only the WebSphere applicationserver product installation needs to install the default configuration:

1. On the first machine, ensure that the database tables have been created, and populatedwith the configurations for the default resources. In the administrative configurationfile, this means:

   ```
   install.initial.config=truecom.ibm.ejs.adminServer.createTables=true
   ```

2. On all subsequent machines sharing the same administrative database, ensurethe database tables are *not* created and the default resources are *not*installed a second time.

   ```
   install.initial.config=falsecom.ibm.ejs.adminServer.createTables=false
   ```

If you have already installed the second (or later) machine, with install.initial.config setto true, then the problem should work itself out after all machines in the cluster havebeen shut down and started again. Until then, you might notice these problems:

- Absence of the **WebSphereAdministrative Domain** resource in the tree view of the administrative console, orother peculiarities in the administrative topology
- Administrative server on the second or later machine will throw exceptions when you first start it

Configuring a remote DB2 database as the administrative database

If the administrative server is going to storeits administrative data in a remote DB2 database, you need to catalog the remotedatabase. See your database documentation for instructions.

Setting the dbserverName and dbportNumber for the data source (flags in the administrativeconfiguration file) is not working in the case of a remote DB2 database. For example, if youhave the application server and the DB2 client is installed on one machine, and are trying to connect the client to the DB2 server on another machine.

This problem *could* be overcome in a subsequent WebSphere Application Server fixpak. Be sureto consult the release notes for each fixpak to see whether the problem is fixed.

What to do if you must drop the administrative database tables

If you must drop the tables in the administrative database for some reason,then reset the pertinent settings in the administrative configuration file tocreate the tables again.

You will always need to set:

```
com.ibm.ejs.adminServer.createTables=true
```

If the machine is the first machine in a cluster of machines sharing anadministrative database, then populate it with the default configuration:

```
install.initial.config=true
```

When you start the administrative server again, the settings will be applied.

Miscellaneous topics

The remainder of this article discusses miscellaneous topics pertainingto the administrative server and database.

Backup copies of admin.config

Note, commented out lines in admin.config will be removed completely when theadministrative server starts.

It is recommended you periodically save a backup copy of admin.config. For example,you might delete lines that you want to restore later.

Running the administrative server in the background

To run the server as a background process, add this parameter to admin.config:

```
com.ibm.ejs.sm.adminServer.processPriority=>number
```

where *number* is 28 for AIX and 24 for Solaris. It could also be thedefault priority of the non-root user ID under which the administrative serveris running. (See also the information on running as non-root).

You will also need to edit the properties of each application server associatedwith this administrative server. Modify the process priority of each server tohave the same process priority (*number*) you assigned to the administrativeserver.

Configuring the nanny process

On UNIX-based systems, the nanny process tries to start an administrative server when the administrative server fails. The nanny settings are available in the admin.config file for the administrative server that the nanny is tending.

6.6.46.0: Administrative server configuration file properties

The admin.config file is located in:

`product_installation_root/bin/admin.config`

Directives in admin.config are similar to their counterparts on the Java command line for the administrativeserver. The command line argument name is appended to astandard package name for the administrative server. For example, the command line argument:

`-lsdPort`

becomes

`com.ibm.ejs.sm.adminServer.lsdPort`

in admin.config.

Some admin.config file properties are platform specific.Review your admin.config file to see which properties appear in your installation.Click any property in the table for a description of that property.Platform specific properties are identified by an icon.

Administrative server and database settings

com.ibm.ejs.sm.adminServer.adminDomain

The logical name specified for the WebSphere administrative domain.

com.ibm.ejs.sm.adminServer.adminDomain.bootstrap

The description is unavailable at this time.

com.ibm.ejs.sm.adminServer.agentMode

Whether this administrative server should run as a full administrative server (false), or in administrative server agent mode (true).

- true
- false

com.ibm.ejs.sm.adminServer.bootstrapHost

Fully qualified host name of the machine containing the administrative server

com.ibm.ejs.sm.adminServer.bootstrapPort

Port number for the administrative server. See also the information about administering ports.

com.ibm.ejs.sm.adminserver.classpath

Values for the administrative server classpath, such as:

```
d\:/WebSphere/AppServer/properties;
d\:/WebSphere/AppServer/lib/bootstrap.jar
```

com.ibm.ejs.sm.adminServer.connectionPoolsize

The number of concurrent database connections to allow from the administrative serverto the administrative database. The default value is 5.

com.ibm.ejs.sm.adminServer.connectionProperties

Use this property to pass arguments into setConnectionProperties() for the WebSphereadministrative server Java process.

For example, if using Sybase as the administrative database, you can setCHARSET_CONVERTER_CLASS=com.sybase.jdbc2.utils.TruncationConverterto prevent exceptions such as this one when performing a dataSource.getConnection() call:

```
java.io.IOException: JZ0I6: An error occured converting UNICODE to the charset used by the server. Error message:
java.io.CharConversionException: java.io.UnsupportedEncodingException: hp-roman8
```

Set additional connectionProperties by specifying them using the same pattern, separated by commas: *PROPERTY_NAME*=value;*PROPERTY_NAME*=value; ...

com.ibm.ejs.sm.adminServer.createTables

Whether to create the administrative database tables the next time you start the administrative server (typically, the first time you start the administrative serverafter product installation).

- true
- false

By default, this flag is initially set to true. Thefirst time you run the administrative server, the administrative server creates the database table or tables that it needs to store administrative data. It sets this flag to false.

See the information about administering administrativeservers for additional discussion of this setting.

In Version 3.5, this setting is called com.ibm.ejs.sm.adminServer.dbInitialized.

com.ibm.ejs.sm.adminServer.dbdataSourceClassName

The data source corresponding to the administrative database.

`COM.ibm.db2.jdbc.DB2ConnectionPoolDataSource`

com.ibm.ejs.sm.adminServer.dbpassword

The password for the administrative database

com.ibm.ejs.sm.adminServer.dbSchema

The database schema for the administrative database

Data source properties - com.ibm.ejs.sm.adminServer.dbUrl

What was called com.ibm.ejs.sm.adminServer.dbUrl in Version 3.5 is replaced byspecific data source properties for each database brand, including a dbUrl property for certain brands.

com.ibm.ejs.sm.adminServer.dbuser

The administrative database user ID, such as db2user

com.ibm.ejs.sm.adminServer.diagThreadPort

The port on which DrAdmin listens

com.ibm.ejs.sm.adminServer.disableAutoServerStart

Whether to disable the feature that tries to start the application servers on an administrativenode when the administrative node is started.

- true - start the administrative server, but not any of the application servers on it
- false - when the administrative server is started, try to start the application servers whose administrative settings indicate they should be started

com.ibm.ejs.sm.adminServer.disablePMI

Whether to disable the performance monitoring classes, to avoid collectingperformance data when it is not needed.

- true - disable PMI classes
- false

This disablePMI setting was called disableEPM in Version 3.5.x.

com.ibm.ejs.sm.adminServer.earFile

The full path to a file containing configuration data for starting the administrativeserver, and product in general.

com.ibm.ejs.sm.adminServer.jarFile

Paths to the JAR files required by the administrative server, such as:

`product_installation_root/lib/repository.jar;product_installation_root/lib/tasks.jar`

com.ibm.ejs.sm.adminServer.logFile

The log files for recording administrative server events, such as:

`product_installation_root/tranlog/your_server_name_tranlog1,product_installation_root/tranlog/your_server_name_tranlog2`

com.ibm.ejs.sm.adminServer.lsdHost

The host name of the administrative server

com.ibm.ejs.sm.adminServer.lsdPort

Port number for the administrative server Location Service Daemon. See also the information about administering ports.

com.ibm.ejs.sm.adminServer.managedServerClassPath

Used by the application server

UNIX com.ibm.ejs.sm.adminServer.nannyPort

The port on which the *nanny* process listens

UNIX com.ibm.ejs.sm.adminServer.processPriority

Possible values are:

- 24 for Solaris
- 28 for AIX

UNIX com.ibm.ejs.sm.util.process.Nanny.adminServerJvmArgs

JVM arguments for the nanny process

UNIX com.ibm.ejs.sm.util.process.Nanny.path

Classpath for the nanny process, such as:

`product_installation_root\bin;product_installation_root\SQLLIB\bin;product_installation_root\SQLLIB\\function;product_installation_root\jdk\bin`

com.ibm.CORBA.ConfigURL

The path to the file specifying ORB configuration values, such as:

`file\:/product_installation_root/properties/sas.server.props`

com.ibm.websphere.preconfiguredCustomServices

Path to a file containing definitions for Custom Services that are to be preconfigured on every application server defined in the domain.

com.ibm.CORBA.ServerSocketQueueDepth

The ServerSocket maximum queue depth. Specify an integer greater than 50. Values less than 50 or invalid integer values will cause a warning to be logged to the message file and the default value of 50 to be used.

This property allows the ServerSocket maximum queue depth to be increased beyond its default value of 50. The default value is according to the JDK122 javadoc for java.net.ServerSocket.

The ServerSocket queue is where incoming socket connection requests from clients are held until the server can accept the request. When the queue becomes full, other open socket requests from clients are rejected with a java.net.ConnectException. This exception will be returned to the client application. If clients are receiving java.net.ConnectExceptions,it is an indication that you should consider increasing the queue depth.

Note, some operating systems impose other limitations on the server socket queue depth. Become familiar with the guidelines imposed by your operating system. Setting this property to a value beyond the platform-imposed limitation will likely not have any effect.

install.initial.config

If set to *true*, attempts to create the default resources (such as the default applicationserver) the next time you start the product administrative server.

- false
- true

For Version 4.0, you need to set both this setting and com.ibm.ejs.adminServer.createTables totrue if you have dropped your WebSphere administrative database tables and need to create them again. Thisis a change from Version 3.5, in which you needed only to reset this setting. See the informationabout administrative servers for more information.

server.root

See the installation root reference for the default values.

com.ibm.ejs.sm.adminServer.seriousEventLogSize

The number of entries to record in the serious events log. Allowing the log to grow too largecan affect performance negatively. The default value is 1000 entries.

UNIX com.ibm.ejs.sm.util.process.Nanny.errtraceFile

The trace file for the nanny process, such as:

`product_installation_root\logs\adminserver_stderr.log`

com.ibm.ejs.sm.adminServer.qualifyHomeName

This value can be:

- true
- false

install.initial.config.file

The full path to the property file defining the default configuration, such as:

`product_installation_root/properties/initial_setup.config`

com.ibm.ejs.sm.adminServer.traceString

The trace specification for the administrative server. See the trace propertyreference for a description.

com.ibm.ejs.sm.adminServer.traceOutput

The trace output file for the administrative server. See the trace propertyreference for a description.

UNIX com.ibm.ejs.sm.util.process.Nanny.traceFile

The trace file for the nanny process, such as:

`product_installation_root/logs/nanny.trace`

UNIX com.ibm.ejs.sm.util.process.Nanny.maxtries

How many time the nanny process should attempt to start the administrative server before giving up. The default is 3.

com.ibm.ws.jdk.path

The path to the root directory of the JDK installed with the product, such as:

`product_installation_root/jdk`

6.6.46.5: Having administrative servers start automatically

By default, the administrative server does not start automatically when a machineis started (or rebooted). This article provides information about starting theserver automatically on Windows NT, triggered by system reboots and other events.

Starting the server when Windows NT starts

On Windows NT, you can use the Services panel to configure the administrative server service to start automatically.

You might also create batch files to run the service. Note than when started from a command line on Windows NT, forexample:

```
net start IBM WS AdminServer
```

the administrative server falsely indicates that it has failed to start. Use the NT Services Panel to verify whether it actually started.

Starting the server when IBM DB2 starts

To have the WebSphere administrative server start automaticallyafter DB2 starts on Windows NT, use the following procedure.

⚠ Take great care in modifying registry entries. The procedure is straightforward,but could cause unintended consequences if you make a mistakewhile entering data. Use this procedure at your ownrisk. If possible, try the procedure on a test machine first.

1. Open the Windows NT Services Panel.
2. Verify that the DB2 and WebSphere administrative serverservices are set to start automatically.
3. Start regedt32 (**not** regedit).
    1. Locate HKEY_LOCAL_MACHINE\Current Control Set\Services\IBM WS AdminServer
    2. If this value name and type does not exist, create it:
        - Value Name: DependOnService
        - Value Type: REG_MULTI_SZ

        To create it, use the Edit -> Add Value menu option. When creating it, you will be prompted to specify data...
    3. Assign two pieces of data to the value:
        - DB2
        - DB2DAS00 (last two digits are zeroes)

        List one piece of data on each line.
4. Save the changes and close regedt32.
5. Reboot the machine. Verify that the administrative server started automaticallyafter the DB2 service started.

6.6.46.6: Using administrative server agents

The product installation program always installs full administrative servers. To produce an administrativeagent, edit the administrative server configuration file.

1. Install an administrative server on a machine.
2. Edit the admin.config file in the bin directory of the*product_installation_root*.
    - Remove the name service JAR property:

      ```
      com.ibm.ejs.sm.adminServer.nameServiceJar=jar-file
      ```
    - Specify the dbUrl, dbUserId, and dbPassword properties:

      ```
      com.ibm.ejs.sm.adminServer.dbUser=user_namecom.ibm.ejs.sm.adminServer.dbUrl=locationcom.ibm.ejs.sm.adminServer.dbPassword=password
      ```
    - Set:

      ```
      install.initial.config=falsecom.ibm.ejs.sm.adminServer.createTables=false
      ```
    - Add or update the property:

      ```
      com.ibm.ejs.sm.adminServer.primaryNode=value
      ```

      where *value* is the name of the node containing the full administrative server to which this agent will connect. To be safe, use the fully qualified name.
    - Add or update the property:

      ```
      com.ibm.ejs.sm.adminServer.bootstrapHost=value
      ```

      where *value* is the host name of the machine containing the full administrative server. To be safe, use the fully qualified name.
    - Add or update the property:

      ```
      com.ibm.ejs.sm.adminServer.agentMode=true
      ```
    - Optionally, specify the properties:

      ```
              com.ibm.ejs.sm.adminServer.bootstrapPort=value  com.ibm.ejs.sm.adminServer.lsdPort=value
      ```

      if the default values (9000 and 900, respectively) are not being used by the full administrative server. Both nodes must use the same values.
3. Save the file.
4. Start the full administrative server.
5. Start the administrative agent the same way you started the full administrative server.
6. Start the Java administrative console onthe machine containing the administrative server agent. Pass the administrativeconsole the host name of the machine containing the full administrativeserver:

   ```
   adminclient full_adminserver_hostname
   ```

   To be safe, use the fully qualified host name.
7. Start the Java administrative console on thefull administrative server machine. Confirm that the administrativeserver agent is displayed as a node in the tree view.

   If the console was already running when you configured and started the agent,you might need to refresh the console in order to see the new node for the agent.

6.6.47: Administering generic servers

By making generic servers or processes known to the WebSphere administrative domain,the administrator can stop and start these processes from the administrative consolewhen stopping or starting the applications that rely on them.

The WebSphere administrator configures one or more generic servers with settings for:

- Specifying the machine on which the server resides
- Specifying Java command line arguments and environment variables for starting theserver process (if applicable)
- Associating the server with an operating system user ID and security group
- Specifying how many times to try to start or ping the server before giving up
- Specifying the process priority on the operating system
- Defining standard in, standard error and standard out streams for the server runtime
- Specifying a working directory

6.6.47.0: Properties of generic servers

Key:

Applies to Java administrative console of Advanced Edition Version 4.0

Applies to Web administrative console of Advanced Single Server Edition Version 4.0

Applies to Application Client Resource Configuration Tool

Name

A logical name for the generic server

The remainder of generic server properties are standard process definition properties and one JVM property: Command Line Arguments.

6.6.47.1: Administering generic servers with the Java administrative console

Use the Java administrative console to administer generic servers.

Work with resources of this type by locating them in the tree view:

WebSphere Administrative Domain -> Nodes -> *node_name* -> Generic Servers

The server instances will be displayed in the details view.

6.6.47.1.1: Adding generic servers with the Java administrative console

To add a generic server:

1. Locate the **Generic Servers** folder in the tree view.
2. Right-click the folder and select **New**.
3. Specify generic server properties.
4. Click **OK**.

6.6.48: Administering ports

The product uses the following ports. See below for port assignment guidelines, particularly notes about the effects of port changes on other WebSphere components. For example, changing the bootstrap port from the product default requires a new parameter when you start the Java administrative console.

Bootstrap port

- One for each application server
- Default value is 900
- To change it, use the administrative console to modify ORB properties

Location Service Daemon (LSD) port

- One for each application server
- Default value is 9000
- To change it, edit the LSD property in the administrative server configuration file

ORB listener port for RMI/IIOP

- One for each application serverand one for each administrative server
- Value is assigned randomly
- To make the value static, add **-Dcom.ibm.CORBA.ListenerPort=xxxx** where*xxxx* is a valid TCP port (see guidelines below).
    - For application servers, add the argument to the Command Line Arguments setting of the application server properties.
    - For administrative servers, add the argument to the com.ibm.ejs.sm.util.process.Nanny.adminServerJvmArgs parameter in the administrative server configuration file.

Port for internal HTTP transport

- One for each application server
- Default value is 9080

Secure port for internal HTTP transport

- One for each application server
- Default value is 9443

DrAdmin port

- One for each application server
- Value is 7000
- This port is assigned dynamically.

Object Level Trace and Debugger port

- One for each application server
- Value is 2012
- To change it, use the administrative console to modify OLT and Debugger properties

Guidelines for assigning ports

Though WebSphere Application Server provides default values, the above port numbers can be modified by the system or network administrator. Here are some guidelines:

- Ports can range from 1024 to 64000. Choose a port that does notconflict with existing ports in use. To check ports in use:
  - Use the netstat command (netstat -a)
  - View the /etc/services file on UNIX
  - View the *drive*\winnt\system32\drivers\etc\services file on Windows NT
- Ports must be unique in the scope of each physical host. That is,two servers on the same machine cannot have the same port values.
- The same port numbers *can* be used for servers running on different physical hosts. That is, the administrative server on Machine Acan have bootstrap port 900, while administrative servers on othermachines have this same number.
- For administrative servers, pick port numbers above 1024 if not using the default port and running as non-root.
- Changing the bootstrap portfrom its default affects the administrative clients. When starting theJava administrative console, you must specify the new port number:

  On UNIX-based operating systems:

  ```
  adminclient.sh hostname port
  ```

  On Windows-based operating systems:

  ```
  adminclient hostname port
  ```
- Remember to configure firewalls to allow traffic to pass for each assignedport. For security, try to minimize ports.

6.6.49: Administering National Language Support

IBM WebSphere Application Server supports several Asian and Europeanlanguage locales, as described in the table below.

This article provides information aboutthe supported locales.

Supported locales

WebSphere Application Server supports the locales listed in the table below.The support is dependent on the availability of the TrueType fonts listed for therespective languages.

Language	Windows NT/2000	Solaris	AIX	HP-UX
Brazilian Portuguese	IBM-850	pt_BR	pt_BR.ISO8859-1	No Brazilian Portuguese locale
French	IBM-850 IBM-858	fr fr.ISO8859-15	fr_FR.ISO8859-1 fr_FR.ISO8859-1@euro	fr_FR.ISO8859-1 fr_FR.ISO8859-15@euro
German	IBM-850 IBM-858	de de.ISO8859-15	de_DE.ISO8859-1 de_DE.ISO8859-1@euro	de_DE.ISO8859-1 de_DE.ISO8859-15@euro
Italian	IBM-850 IBM-858	it it @ISO8859-15	it_IT.ISO8859-1 it_IT.ISO8859-1@euro	it_IT.ISO8859-1 it_IT.ISO8859-15@euro
Japanese	IBM-943	ja ja_PC.PCK	ja_JP.IBM.eucJP Ja_JP.IBM-942	ja_JP.SJIS *
Korean	IBM-1363	ko	ko_KR.IBM.eucKR	ko_KR.eucKR *
Simplified Chinese	IBM-1386	zh	zh_CN.IBM.eucCN	zh_CN.hp15CN *
Spanish	IBM-850 IBM-858	es es @ISO8859-15	es_ES.ISO8859-1 es_ES.ISO8859-1@euro	es_ES.ISO8859-1 es_ES.ISO8859-15@euro
Traditional Chinese	IBM-950	zh_TW.big5	zh_TW.big5	zh_TW.big5 *

6.6.51: Administering network configurations

This article discusses topics related to network configurations ina WebSphere environment. Network configurations can affect yourWebSphere configuration.

Multiple networks

WebSphere Application Server tolerates the presence of more than one network on a single platform as long as the environment used to run WebSphere Application Server is configured such that the following conditions are met:

- The primary hostname refers to the default - primary - network interface.
- The Domain Name Server's response to a query of the platform's hostname returns the address of the primary network interface.
- The IP for the platform resolves to the hostname and vice versa for both formats of hostname: fully qualified and not fully qualified.

Here is a good example of a multi-network interface card Windows NT platform that conforms to the above specifications.The hostnames, IP addresses, and adapter IDs have been genericized to *my.server.net*, patterns such as aaa.bbb.ccc.ddd, andvariables of the form *adapter_ID_n*, respectively:

```
d:\work>ipconfigWindows NT IP ConfigurationEthernet adapter adapter_ID_1:IP Address. . . . . . . . . : aaa.bbb.ccc.ddd (Primary IP)Subnet Mask . . . . . . . . : 255.255.255.0Default Gateway . . . . . . : 208.180.235.1Ethernet adapter adapter_ID_2:IP Address. . . . . . . . . : eee.fff.ggg.hhhSubnet Mask . . . . . . . . : 255.255.254.0Default Gateway . . . . . . : 129.17.32.1d:\work>nslookup aaa.bbb.ccc.dddServer: my.server.netAddress: mmm.nnn.ooo.ppp Name: cdm-235-122-pflu.cox-internet.com (Resolves to hostname)Address: aaa.bbb.ccc.dddd:\work>hostnamecdm-235-122-pflu (hostname is set correctly)d:\work>nslookup cdm-235-122-pflu (and resolves to the same)Server: my.server.netAddress: mmm.nnn.ooo.pppName: cdm-235-122-pflu.cox-internet.comAddress: aaa.bbb.ccc.ddd (IP when NOT fully qualified)d:\work>nslookup cdm-235-122-pflu.cox-internet.com (also resolves to the same)Server: my.server.netAddress: mmm.nnn.ooo.pppName: cdm-235-122-pflu.cox-internet.comAddress: aaa.bbb.ccc.ddd (IP when fully qualified)
```

6.6a: Starting and stopping servers

1. Starting administrative servers:
    - On UNIX systems
    - On Windows systems
2. Obtaining feedback that the administrative server started successfully
3. Configuring the administrative server startup settings
4. Starting the Java-based administrative console
5. Troubleshooting the administrative server startup
6. Starting and stopping application servers
7. Stopping administrative servers:
    - On UNIX systems
    - On Windows systems

Starting administrative servers on UNIX-based systems

To start the administrative server:

1. Ensure you are running as the root user.
2. Change to the directory containing thestartup script (as described above), from a command prompt and enter:

```
./startupServer.sh
```

Running as non-root on UNIX-based systems

Normally, the administrative server must run under the system root ID. To avoid this restriction, see the information about running the administrative server as non-root.

For information about running administrative and application servers as non-root on UNIX systems, see article 6.6.a.1 in the InfoCenter, available from the Library page of the product Web site.

Stopping servers on UNIX-based systems

To stop a WebSphere administrative server on UNIX-based systems, use an administrative client, such as the Java administrativeconsole. Using the kill command is **not** recommended.

Starting administrative servers on Windows-based systems

The administrative server is started and stoppedfrom the Control Panel Services menu.

1. From the Start menu, select **Settings** > **Control Panel**.
2. Open the Services icon.
3. From the Services panel, select **IBM WS AdminServer**.
4. Use the Start and Stop buttons. Stopping the administrative server also stops the application server.
5. To enable or disable automatic startup of the administrative serveron system reboot, click **Startup** and select **Automatic** or **Manual**.
6. If starting the administrative server, start your Web server. The WebSphere administrative server and the Web server run in separate processes. You can start them in any order.
7. Close Services.

Stopping servers on Windows-based systems

Stop the server, by stopping the corresponding Windows service.

Configuring the administrative server startup settings

The administrative server obtains its initialization parametersfrom the administrative configuration file.

Obtaining feedback

Obtain feedback in the console window from which you start the application server. You can also update the configuration to route the system output to a log file. A message indicates that theApplication Server is running successfully. A typical message is shown here.

```
[01.05.31 12:03:34:434 CDT] 16023bce Server  I WSVR0023I: Server __adminServer open for e-business
```

Troubleshooting

See the Problem Determination section for methodology and problem descriptions for debugging a server that does not start. Here are some highlights. If the server will not start ...

- ... and you encounter this error (in the Services panel on Windows NT):

  ```
  The IBM WebSphere AdminServer service returned service specific error 10
  ```

  Add a new line character at the end of the administrative server configuration file.
- ... when WebSphere security is enabled, on a Windows machine that is configured aspart of a Windows domain and not connected via the network to the domain server,configure the Windows NT/2000 machine separately from the Windows NT/2000 domain.

6.6.a.1: Running the product servers and consoles as non-root

Any application server or administrative server can be run using a non-root ID. The tradeoff is that you must use an LDAP directory for the authentication mechanism for WebSphere security. You can no longer use the local operating system.

By default, WebSphere servers use a root ID. Use the instructions below to change the ID. The Java administrative console can be accessed from a non-root ID, provided security permissions are configured appropriately.

- Running application servers as non-root
- Running administrative servers as non-root
- Running Java administrative consoles as non-root
- Running as non-root on Solaris: ndd
- Problems and symptoms based on running as non-root incorrectly

Running application servers as non-root

1. Start the WebSphere administrative server under the root ID.
2. Start the Java administrative console.
3. In the tree view, locate and click the application server to display its properties.
4. In the Advanced properties, modify the User ID and Group ID to be the user and groupfor the application server to "run as."
5. In the General properties, modify the standard output and standard error log pathsto refer to directories to which the "run as" identity has access.
6. Start the application server, using the new ID.

Running administrative servers as non-root

1. Change permissions to the product installation directories to allow access to the administrative server when it "runs as" a non-root ID.Do **one** of the following:
    - Change the ownership of all files and directories under the product_installation_root to the user and group that you want the administrative server to "run as."
    - Change the ownership of these specific files and directories to the user and group that you want the administrative server to "run as."
        - *product_installation_root*/bin/*
        - *product_installation_root*/config/*
        - *product_installation_root*/etc/*
        - *product_installation_root*/installedApps/*
        - *product_installation_root*/logs/*
        - *product_installation_root*/properties/*
        - *product_installation_root*/tranlog/*
        - *product_installation_root*/temp/*

      Make sure that the user has write and execute privileges.
2. Change the bootstrap port of the administrative server to a value greater than or equal to 1024:
    a. Open the administrative configuration file in a text editor.
    b. Add the following:

    ```
    com.ibm.ejs.sm.adminServer.bootstrapPort=2222
    ```

    where 2222 is just an example of a new port that you might use.

    Changing the bootstrap port affects the administrative clients that connect to the server. See port administration overview for details.

3. Start the administrative server, using the new ID.

Running Java administrative consoles as non-root

1. Change the ownership of the following directories and files to the user and group that you would like the console to "run as":

   `product_installation_root``/bin`*`product_installation_root`*`/properties/sas.client.props`
2. Make sure the user has permission to access the secured administrative account.

What you should know about running as non-root on Solaris: ndd

On Solaris, the "ndd" commands in the administrative server startupServer.sh script need to be commented out unless you are running as root.

If an "ndd" command is being executed by a non-root user, the following error message will be issued to stdout or stderr:

```
operation failed, Not owner
```

The 'ndd' command is for dynamically adjusting certain IP stack parameters. It attempts to operate on operating system level kernel device settings,which can only be performed by root. Thus the error message.

The workaround is to either run the administrative server as root or edit the startupServer.sh script, commenting out the ndd command.

It is still strongly recommended that the changes to the TCP parameters that the "ndd" command makes be made by root on all machines running the application server and Web server (in case they are not the same box).

Problems and symptoms based on running as non-root incorrectly

The following problems indicate the need to review the above instructions to ensure that your configuration is correct for running as non-root.

- The following error message is displayed when to start the administrativeserver as non-root on Solaris:

  ```
   $ ./startupServer.sh operation failed,Not owner
  ```
- The following error message is displayed when starting the server as non-root (bootstrapPort < 1024):

  ```
  NMSV0011E: Unable to start Bootstrap Server
  ```

  The most likely cause is that the bootstrap port is already in use. Ensure that no servers or other processes are already using the bootstrap server port.

InfoCenter Home >
6: Administer applications >
6.7: Tutorials

6.7: Tutorials

The tutorials help you learn about various aspects of application assembly, configuration, deployment, and special topics, such as SOAP support and debugging. The available tutorials are listed in the related information to the right.

The tutorials refer to example code that is shipped with the product or the documentation set, allowing you to practice the tasks firsthand.

Example code for "main" tutorials

The "main" set of tutorials, 6.7.1 - 6.7.6, comprise an end-to-end view of the administrative procedure. The main tutorials are not included in the **Back** and **Forward** button sequence. Instead, each one provides a link to the next tutorial in the sequence.

Save the following code to a directory of your choice:
tutorial.zip

For the tutorials, you will want to extract the files (preserving directory structure) so that you have the following directory:

`product_installation_root``/temp/tutorial/`

with subdirectories under "tutorial" matching the subdirectories in the tutorial.zip file (cmp, security, and assembly).

Tasks covered in "main" tutorials

The following table summarizes the tasks covered by the main set of tutorials.

Segment of main tutorial ->	6.7.1: Assembly	6.7.2: Deployment	6.7.3: Test	6.7.4: Security	6.7.5: Advanced (CMP)	6.7.6: Cleanup
Start assembly tool	✓			✓	✓	
Convert EJB 1.0 JAR to EJB 1.1 JAR	✓				✓	
Assemble EJB module (Session bean)	✓				✓	
Assemble Web module	✓					
Assemble application client module	✓				✓	

Back Forward

Related topics

Home (Getting started page)

Tutorials

6.7.1: Application assembly tutorial

6.7.2: Application deployment tutorial

6.7.3: Application testing tutorial

6.7.4: Application security tutorial

6.7.5: Advanced application assembly and deployment tutorial -- CMP bean application

6.7.6: Application cleanup and removal tutorial

6.7.soap: Deploying a Java class as a Web service, using SOAP

6.7.sq: StockQuote tutorial: Using Debugger and OLT

Assemble J2EE application	✓				✓	
Generate code for deployment	✓				✓	
Exit assembly tool	✓			✓	✓	
Start application server		✓		✓	✓	
Open administrative console		✓		✓	✓	
Deploy (install) J2EE application		✓		✓	✓	
Regenerate Web server plug-in		✓		✓	✓	
Stop and restart application server		✓		✓	✓	
Test Web client			✓			
Test Java client			✓		✓	
Enable security during assembly				✓		
Enable security in application server				✓		
Deploy secured application				✓		
Test secured Web client				✓		
Test secured Java client				✓		
Create database tables					✓	
Configure data source					✓	
Uninstall application						✓

6.7.hc: HitCount tutorial: Using Debugger and OLT

Peer topics

6.1: Quick reference for administration

6.2: Preparing to host applications

6.3: Assembling applications and generating deployment code

6.4: Installing applications and setting classpaths

6.5: Maintaining and updating applications

6.6: Tools and resources quick reference

6.6.a.1: Running the product servers and consoles as non-root

6.6a: Starting and stopping servers

6.10: Backing up and restoring administrative configurations

InfoCenter

To launch the full documentation set in a separate browser window, click:

Display InfoCenter

PDF library

To browse the PDF library for this product, containing this article and others, click:

PDF versions

Using this documentation

Become an InfoCenter super user! To find out more about navigation, numbering, search, downloads, and more, click:

Using this documentation

← Back Forward →

6.7.1: Application assembly tutorial

During this tutorial, you will assemble a J2EE application .ear file from an EJB JAR file, some Web application components, and a Java application client JAR file. Assembly is a prerequisite to deploying your application on the application server. Put simply, assembly involves identifying the various code artifacts as a single unit (application), and configuring the deployment descriptor of the application.

The application used for this tutorial is a Session bean application, but the assembly steps are similar for a BMP bean application. For information about CMP bean applications, see the Advanced tutorial, which assumes knowledge of this assembly tutorial and others.

Prerequisites

There are no prerequisites for performing this tutorial.

Overview of steps (requires 30 to 50 minutes)

1. Obtain the tutorial application
2. Start the Application Assembly Tool
3. Assemble an EJB module, converting EJB 1.0 JAR to EJB 1.1
4. Assemble a Web module
5. Assemble an application client module
6. Assemble a J2EE application
7. Generate code for deployment
8. Exit the Application Assembly Tool

Paths through the tutorial

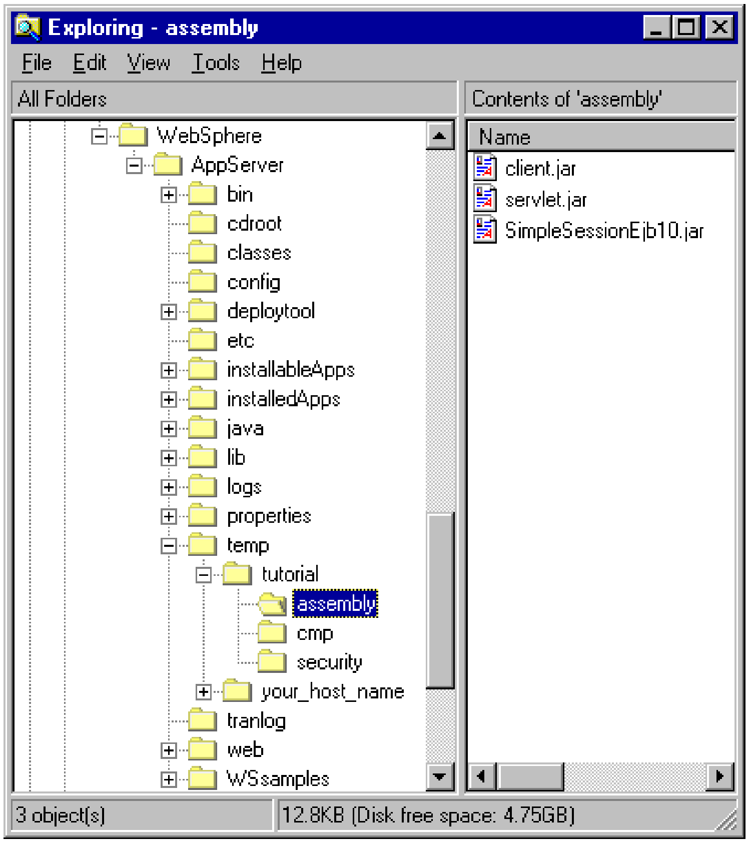
- **Quick path:** If your objective is to practice these steps as quickly as possible, skip the explanations marked with the ℹ graphic.
- **Maximum learning path:** If your objective is to understand and explore the product capabilities, read the ℹ notes and browse the links they provide to additional documentation.

Obtain the tutorial application

1. Create a directory named "tutorial" under the path:

   `product_installation_root/temp`
2. Click here to access the .zip file containing the tutorial application components.
3. Save the .zip file to

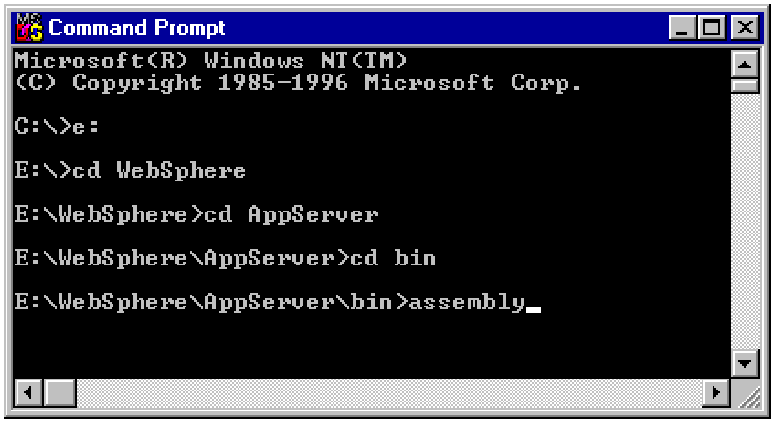
   `product_installation_root/temp/tutorial`
4. Use your favorite .zip or .jar utility to extract the tutorial.zip content into the tutorial directory.



Start the Application Assembly Tool

Start the Application Assembly Tool (AAT) by:

1. Open a system command prompt.
2. Change directory to *product_installation_root*/bin
3. Type assembly.



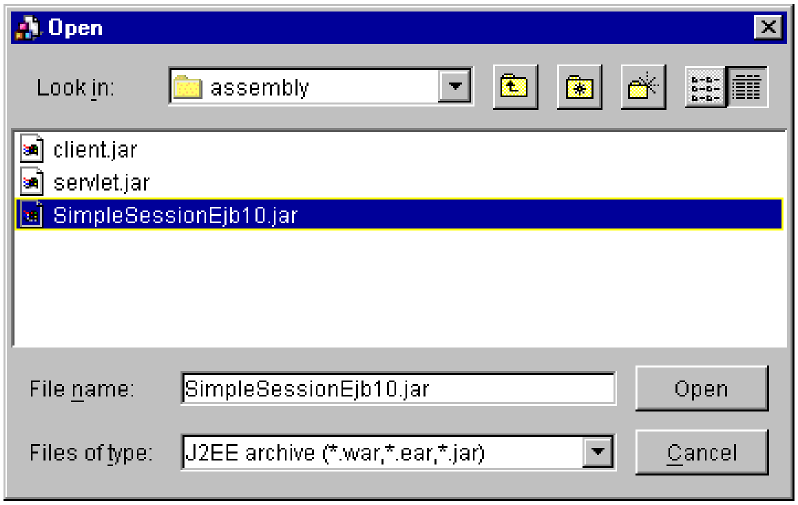
```
Microsoft(R) Windows NT(TM)
(C) Copyright 1985-1996 Microsoft Corp.

C:\>e:

E:\>cd WebSphere

E:\WebSphere>cd AppServer

E:\WebSphere\AppServer>cd bin

E:\WebSphere\AppServer\bin>assembly_
```

NT If using Windows NT or 2000, and you do not see the graphical interface of the tool right away, check for the minimized tool on the Task Bar.

Assemble an EJB module, converting EJB 1.0 JAR to EJB 1.1

1. Click **Cancel** at the **Welcome to Application Assembly Tool** panel.
2. Load the EJB jar file into the tool:
    a. Click **File** -> **Open**.
    b. Navigate to the **SimpleSessionEjb10.jar** and select **open**.
    c. Click **OK** at the "... please specify the dependent classpath ..." dialog.
3. [*Optional*] Edit the deployment descriptor.
    a. Select the jar (the top entry in the assembly tool tree view) and click it to display its properties.
    b. Enter the following:
        1. Display Name: **EJB11**
    c. Click **Apply**

4. Save your changes using **File**->**Save As**. Save the file as **Ejb11.jar** in the path:
   `product_installation_root/temp/tutorial/assembly/Ejb11.jar`

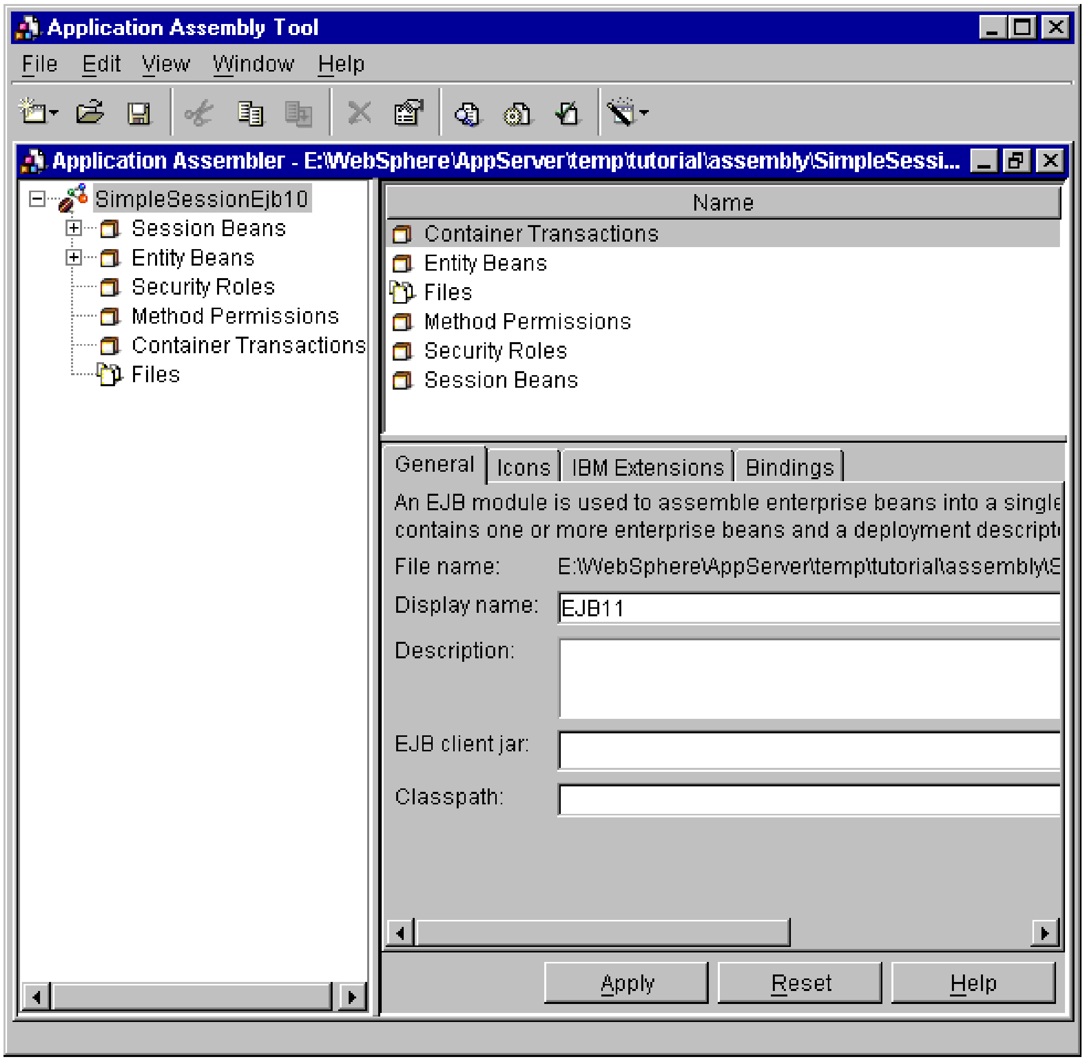


Look in: assembly

File name: SimpleSessionEjb10.jar

Files of type: J2EE archive (*.war,*.ear,*.jar)

You now have a view of the EJB module in the Application Assembly Tool tree view.



An EJB module is used to assemble enterprise beans into a single
contains one or more enterprise beans and a deployment descripti

File name: E:\WebSphere\AppServer\temp\tutorial\assembly\S

Display name: EJB11

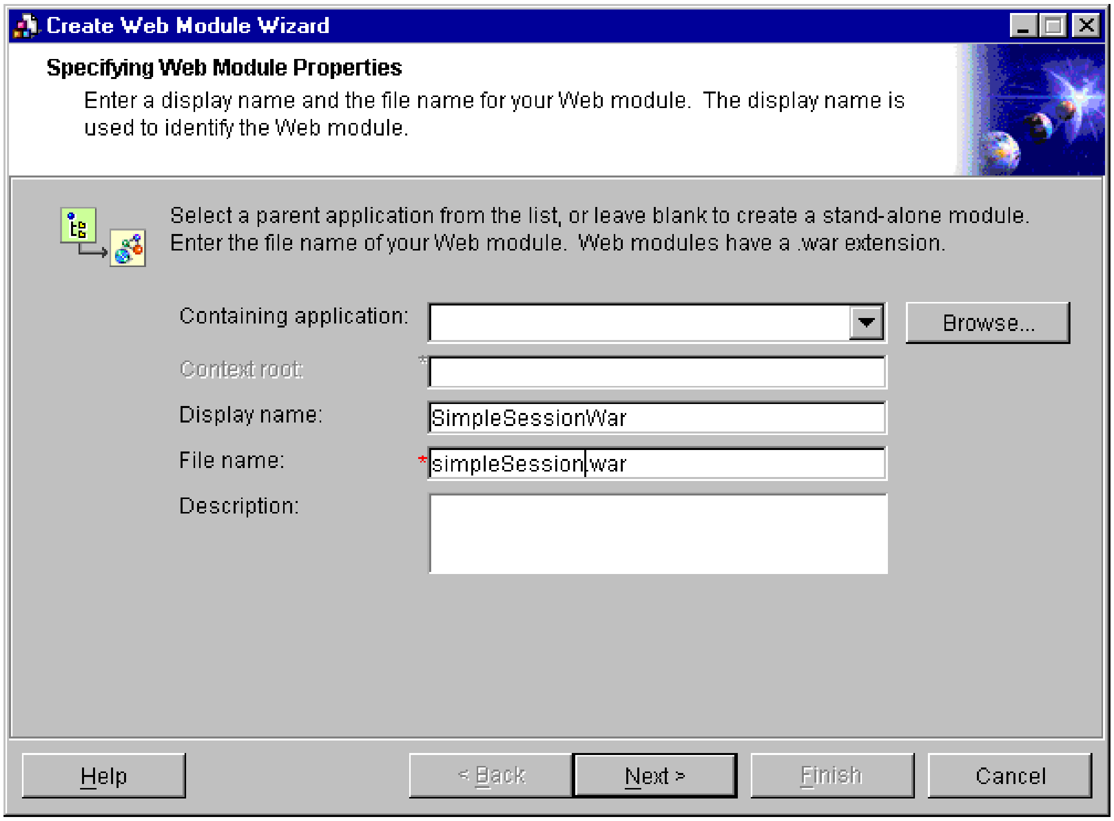
Description:

EJB client jar:

Classpath:

Assemble a Web Module

1. Create a new .war file by clicking **File** -> **Wizards** -> **Create Web Module Wizard**.
2. On the **Specifying Web Module Properties** panel, enter the following:
   a. Display Name: **SimpleSessionWar**
   b. File Name: **simpleSession.war**



Specifying Web Module Properties

Enter a display name and the file name for your Web module. The display name is used to identify the Web module.

Select a parent application from the list, or leave blank to create a stand-alone module. Enter the file name of your Web module. Web modules have a .war extension.

Containing application:

Context root: *

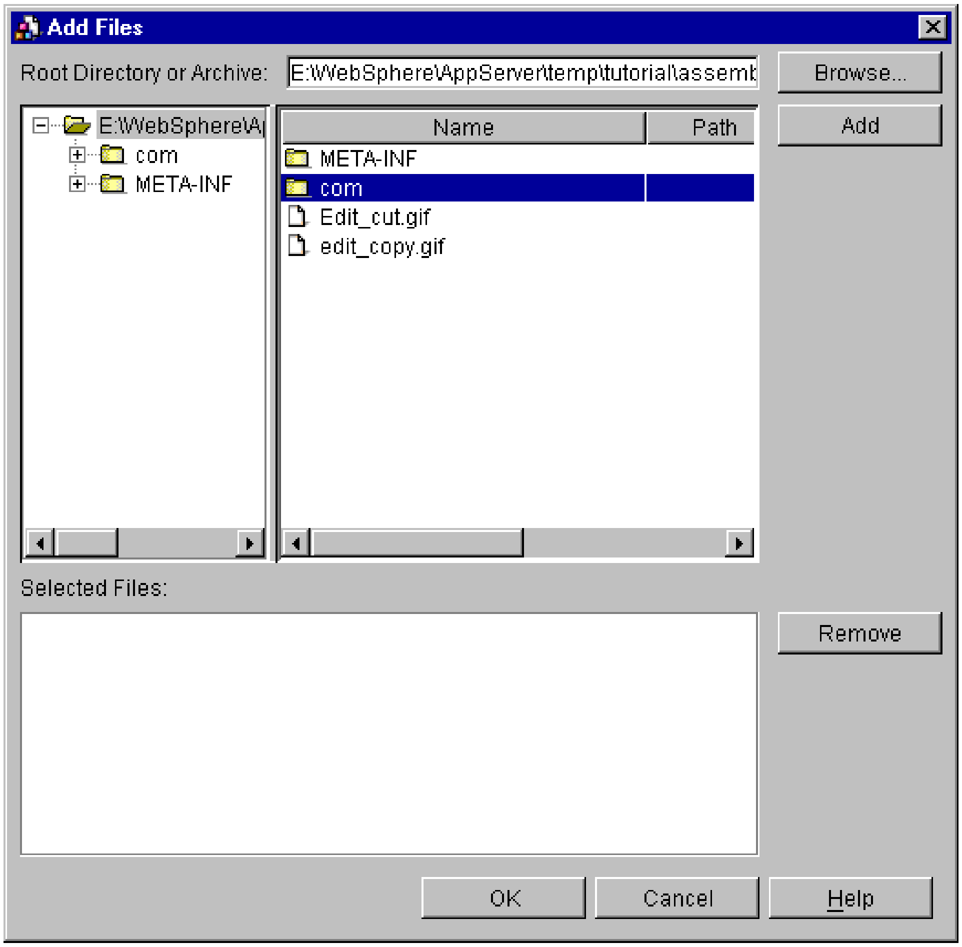
Display name: SimpleSessionWar

File name: * simpleSession.war

Description:

c. Click **Next.**

3. Click **Add Class Files...** on the **Adding Files** panel:
   a. Select **Browse**
   b. Navigate to the **servlet.jar** file, highlight the file and click **Select**.
   c. Select the **com** folder (in the right hand pane).



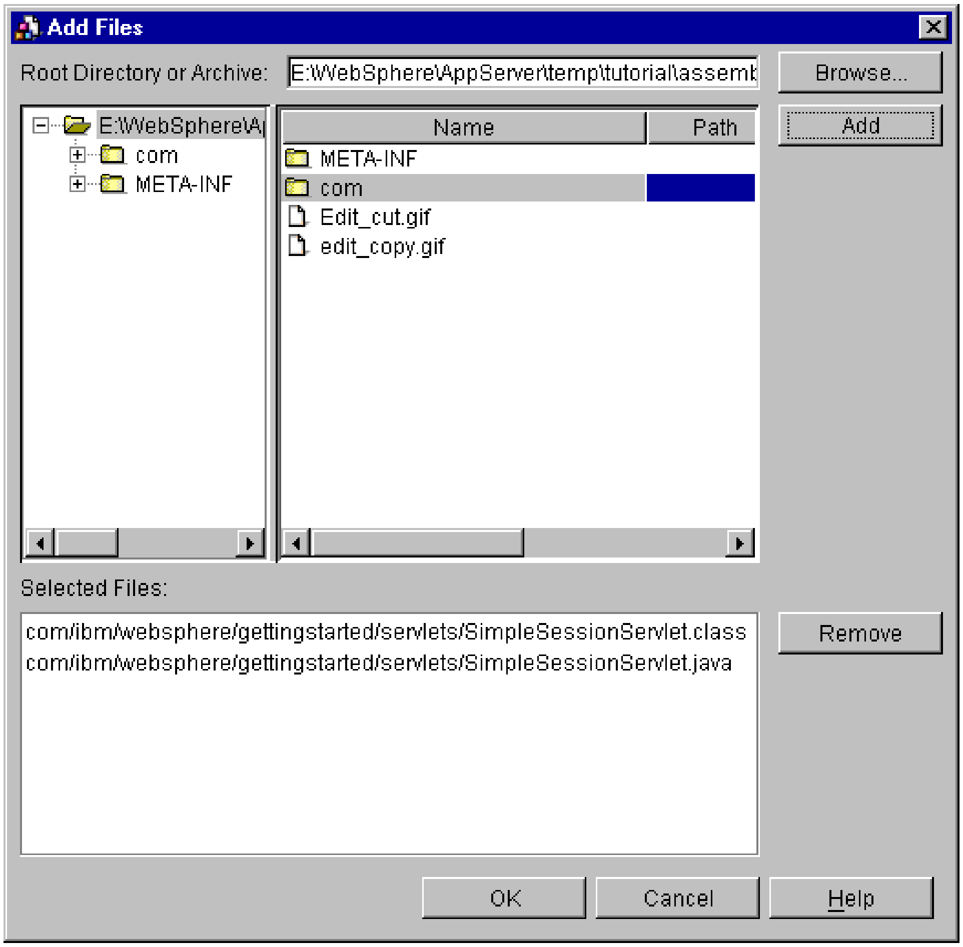
Root Directory or Archive: E:\WebSphere\AppServer\temp\tutorial\assemb

E:\WebSphere\A
com
META-INF

META-INF
com
Edit_cut.gif
edit_copy.gif

Selected Files:

d. Click **Add** to display the files (in the com folder) in the lower part of the dialog.



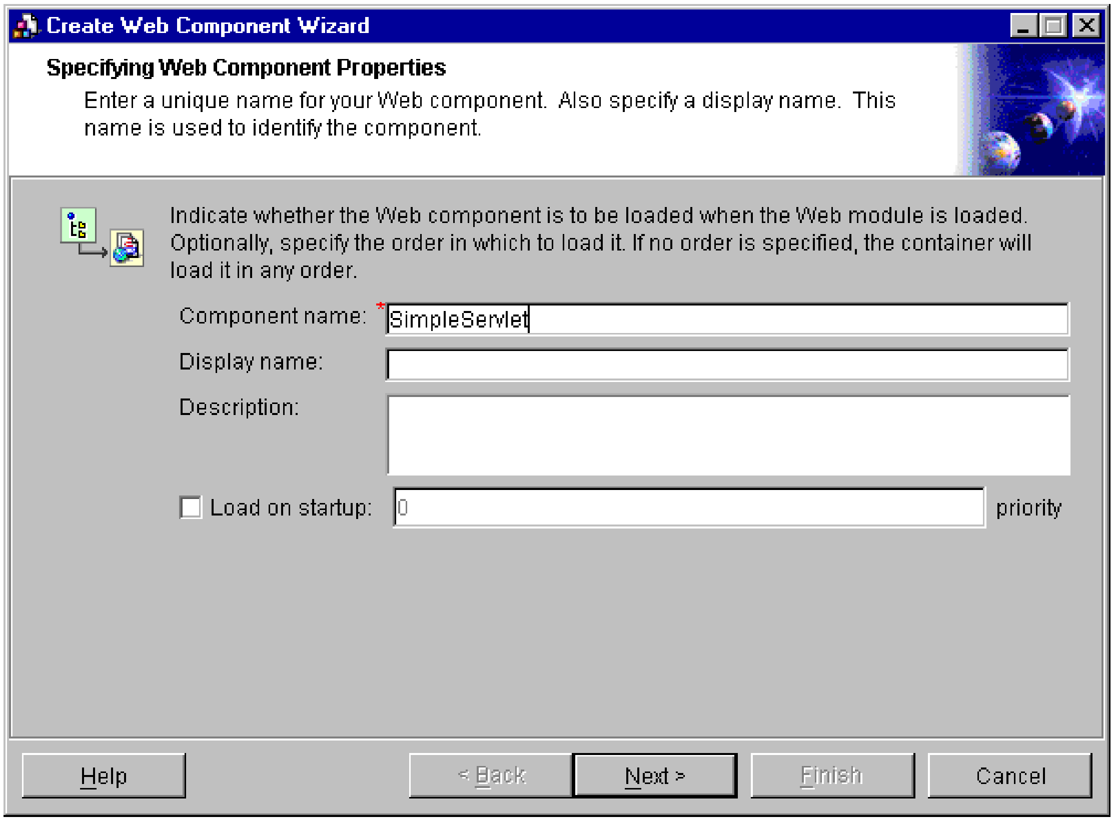
Root Directory or Archive: E:\WebSphere\AppServer\temp\tutorial\assem

Selected Files:

com/ibm/websphere/gettingstarted/servlets/SimpleSessionServlet.class
com/ibm/websphere/gettingstarted/servlets/SimpleSessionServlet.java

   e. Click **OK**.
   f. Click **Next**.

4. Click **Next** on the **Specifying Optional Web Module Properties** settings panel.
5. Click **Next** on the **Choosing Web Module Icons** settings panel.
6. Click **New...** on the **Adding Web Components** panel to start the **Create Web Component Wizard**:
   a. On the **Specifying Web Component Properties** panel:
      1. Enter the Component Name: **SimpleServlet**



Specifying Web Component Properties

Enter a unique name for your Web component. Also specify a display name. This name is used to identify the component.

Indicate whether the Web component is to be loaded when the Web module is loaded. Optionally, specify the order in which to load it. If no order is specified, the container will load it in any order.

Component name: * SimpleServlet

Display name:

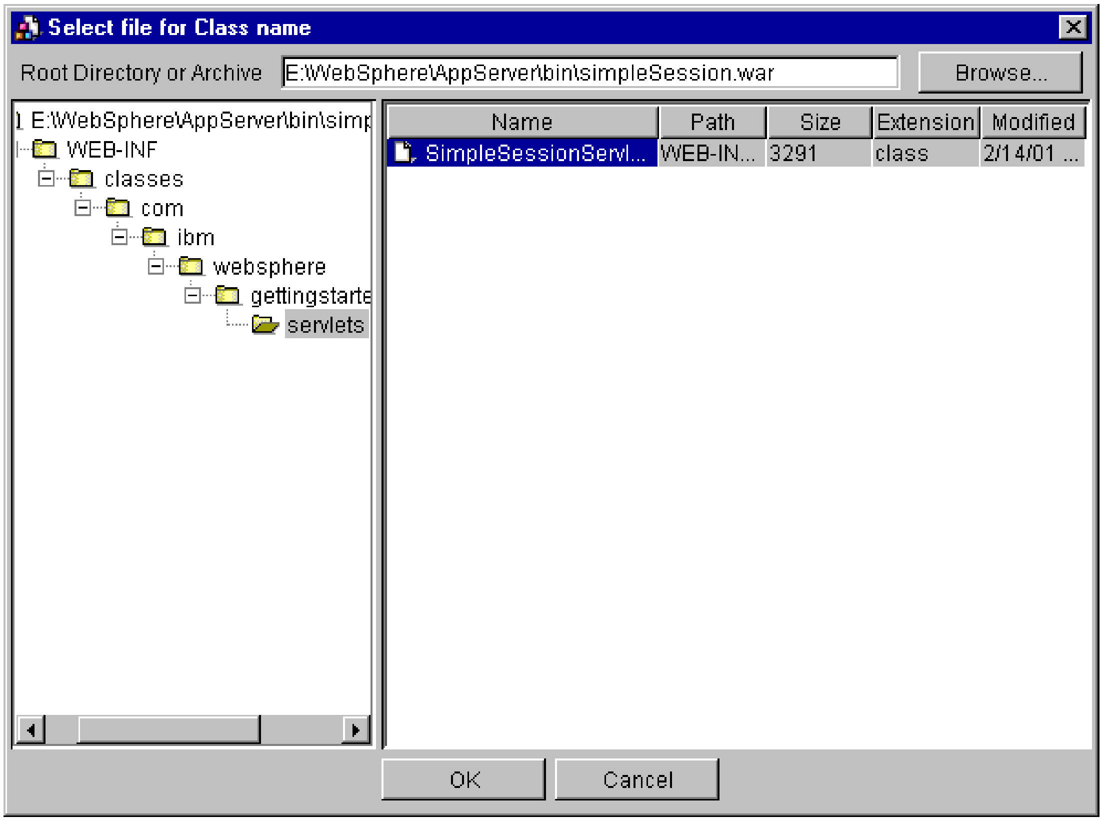
Description:

☐ Load on startup: 0 priority

b. Select **Next.**

c. On the **Specifying Web Component Type** panel:

1. Set the **Component Type** to Servlet.
2. Click **Browse**.
3. Expand the tree view as far as it will go and select **servlets**.
4. Select **SimpleSessionServlet.class** in the right pane.

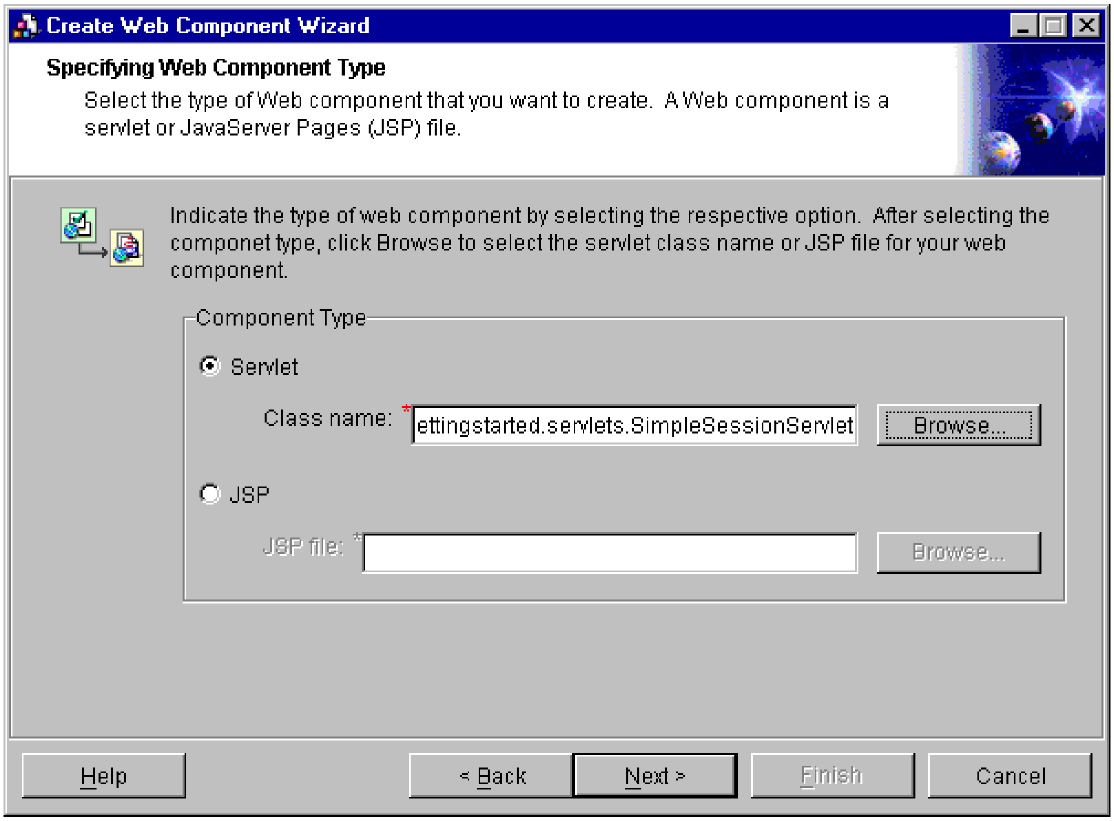


Root Directory or Archive E:\WebSphere\AppServer\bin\simpleSession.war

- E:\WebSphere\AppServer\bin\simp
  - WEB-INF
    - classes
      - com
        - ibm
          - websphere
            - gettingstarte
              - servlets

Name	Path	Size	Extension	Modified
SimpleSessionServl...	WEB-IN...	3291	class	2/14/01 ...

5. Click **OK**.



Create Web Component Wizard

Specifying Web Component Type

Select the type of Web component that you want to create. A Web component is a servlet or JavaServer Pages (JSP) file.

Indicate the type of web component by selecting the respective option. After selecting the component type, click Browse to select the servlet class name or JSP file for your web component.

Component Type

● Servlet

Class name: * ettingstarted.servlets.SimpleSessionServlet

○ JSP

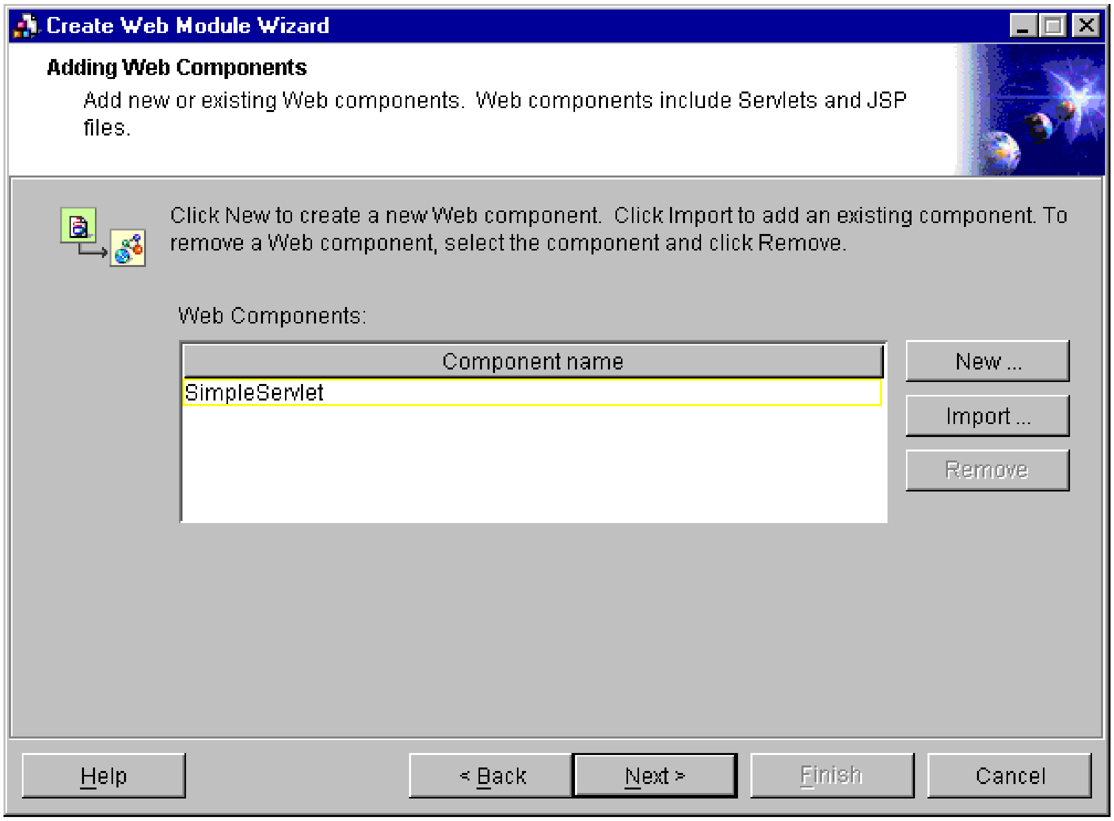
JSP file: *

6. Click **Next**

d. Click **Next** on the **Choosing Web Components Icon** panel.

e. Click **Next** on the **Adding Security Role References** panel.

f. Click **Finish** on the **Adding Initialization Parameters** panel. This will return you to the **Create Web Module** wizard.



Create Web Module Wizard

Adding Web Components

Add new or existing Web components. Web components include Servlets and JSP files.

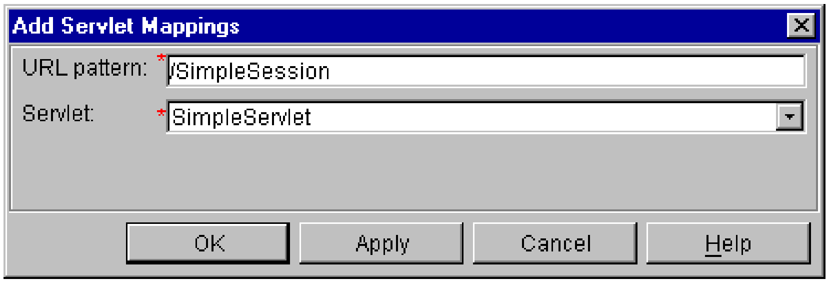
Click New to create a new Web component. Click Import to add an existing component. To remove a Web component, select the component and click Remove.

Web Components:

Component name
SimpleServlet

g. Click **Next**.

7. Click **Next** on the **Adding Security Roles** panel.
8. Click **Add** on the **Adding Servlet Mappings** panel.
    a. Specify a URL Pattern: **/SimpleSession**
    b. Ensure that **SimpleServlet** is selected as the **Servlet**.

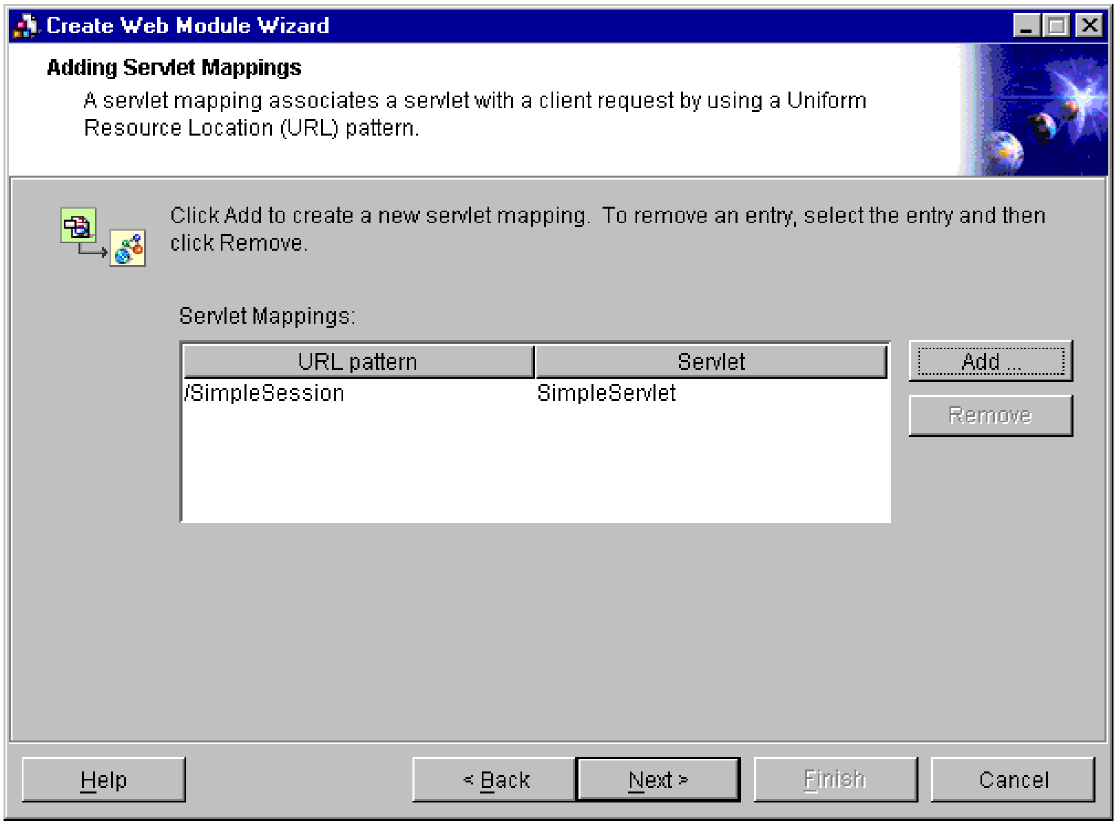


Add Servlet Mappings

URL pattern: /SimpleSession

Servlet: SimpleServlet

c. Click **OK**.



Create Web Module Wizard

Adding Servlet Mappings

A servlet mapping associates a servlet with a client request by using a Uniform Resource Location (URL) pattern.

Click Add to create a new servlet mapping. To remove an entry, select the entry and then click Remove.

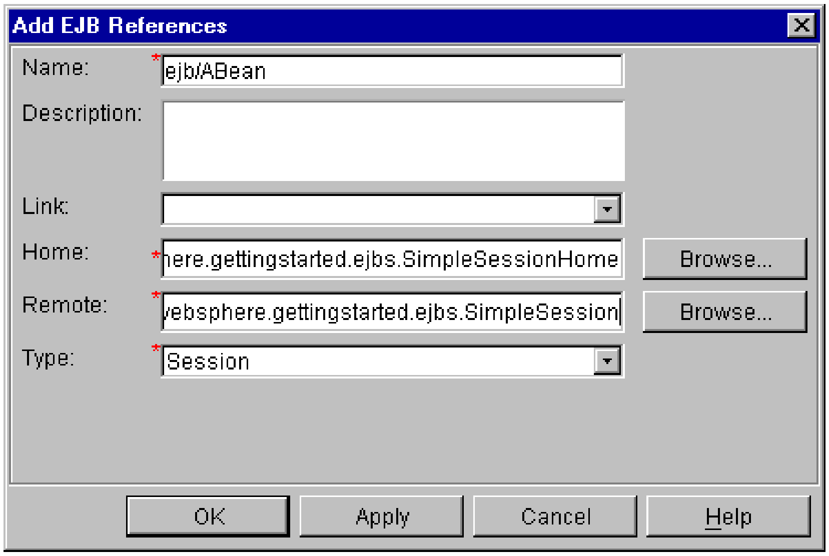
Servlet Mappings:

URL pattern	Servlet
/SimpleSession	SimpleServlet

d. Click **Next**.

9. Click **Next** at the **Adding Resource References** panel.
10. Click **Next** at the **Adding Context Parameters** panel.
11. Click **Next** at the **Adding Error Pages** panel.
12. Click **Next** at the **Adding MIME Mappings** panel.
13. Click **Next** at the **Adding Tag Libraries** panel.
14. Click **Next** at the **Adding Welcome Files** panel.
15. Click **Add** at the **Adding EJB References** panel:
    a. Enter the following:
        - Name: **ejb/ABean**
        - Home: **com.ibm.websphere.gettingstarted.ejbs.SimpleSessionHome**
        - Remote: **com.ibm.websphere.gettingstarted.ejbs.SimpleSession**

    b. Ensure **Session** is selected as the **Type**.



Name: ejb/ABean

Description:

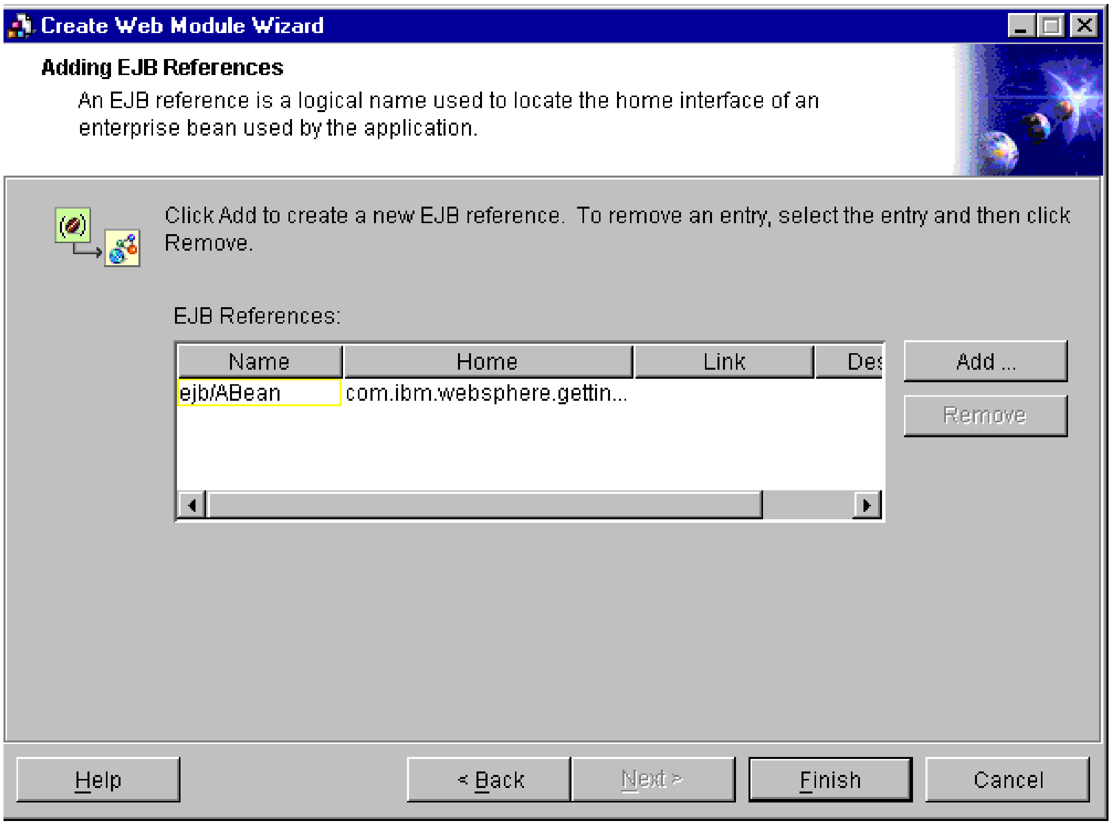
Link:

Home: here.gettingstarted.ejbs.SimpleSessionHome

Remote: ebsphere.gettingstarted.ejbs.SimpleSession

Type: Session

c. Click **OK**.



Adding EJB References

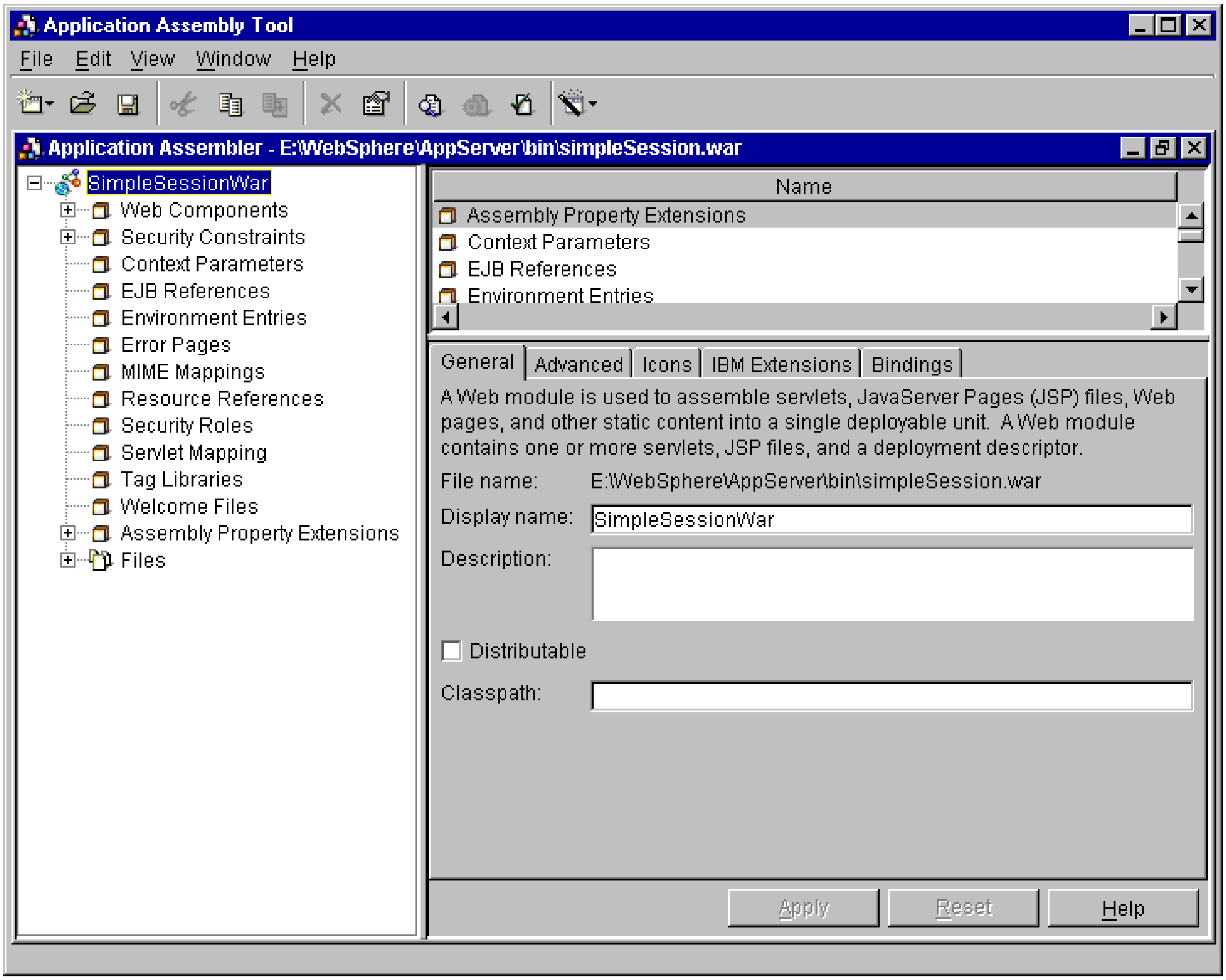
An EJB reference is a logical name used to locate the home interface of an enterprise bean used by the application.

Click Add to create a new EJB reference. To remove an entry, select the entry and then click Remove.

EJB References:

Name	Home	Link	Des
ejb/ABean	com.ibm.websphere.gettin...		

16. Click **Finish**. The Web module is displayed in the assembly tool interface.



A Web module is used to assemble servlets, JavaServer Pages (JSP) files, Web pages, and other static content into a single deployable unit. A Web module contains one or more servlets, JSP files, and a deployment descriptor.

File name: E:\WebSphere\AppServer\bin\simpleSession.war

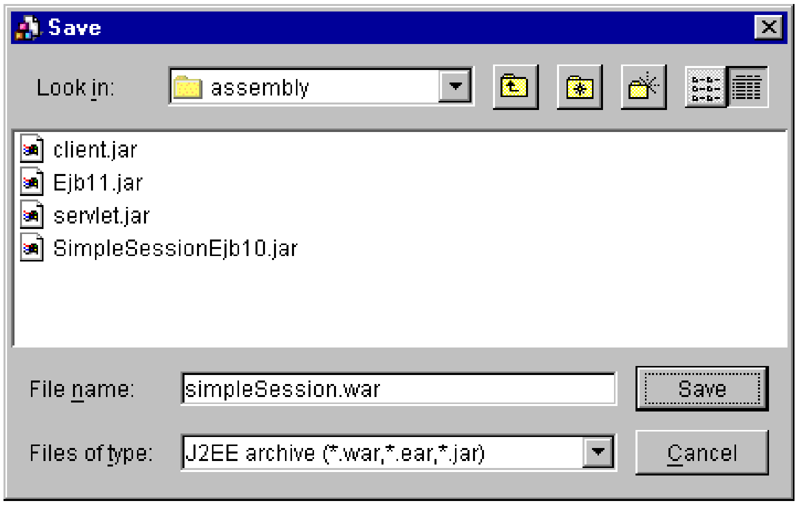
Display name: SimpleSessionWar

Description:

☐ Distributable

Classpath:

17. Save the .war file using **File -> Save As**. Name the file **simpleSession.war**.



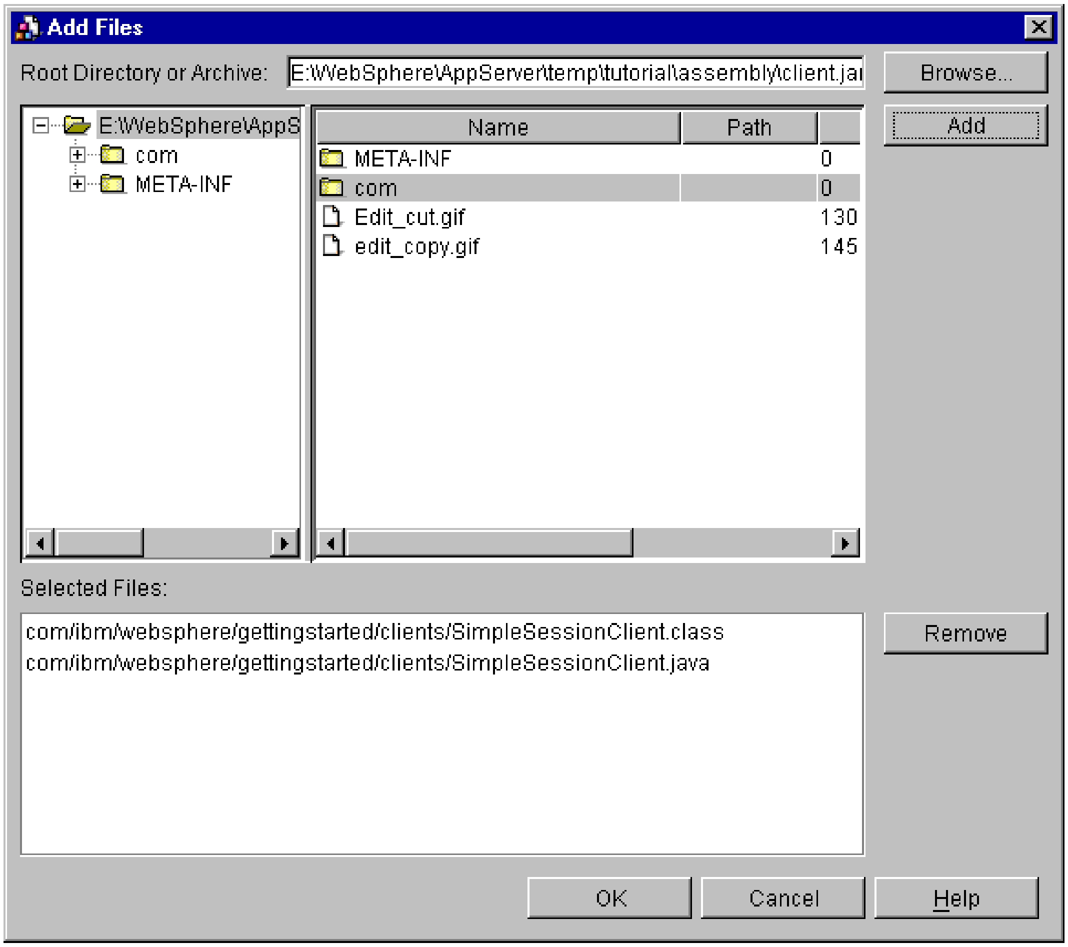
File name: simpleSession.war

Files of type: J2EE archive (*.war,*.ear,*.jar)

Assemble an application client module

1. Create a new application client using **File** -> **Wizards** -> **Create Application Client Wizard**.
2. On the **Specifying Application Client Module Properties** panel, enter the following:
    a. Display Name: **SimpleSessionClient**
    b. File Name: **simpleApp.jar**
    c. Click **Next**.
3. Click **Add...** on the **Adding Files** panel.
    a. Select **Browse.**
    b. Navigate to the **client.jar** file, highlight the file and click **Select**.

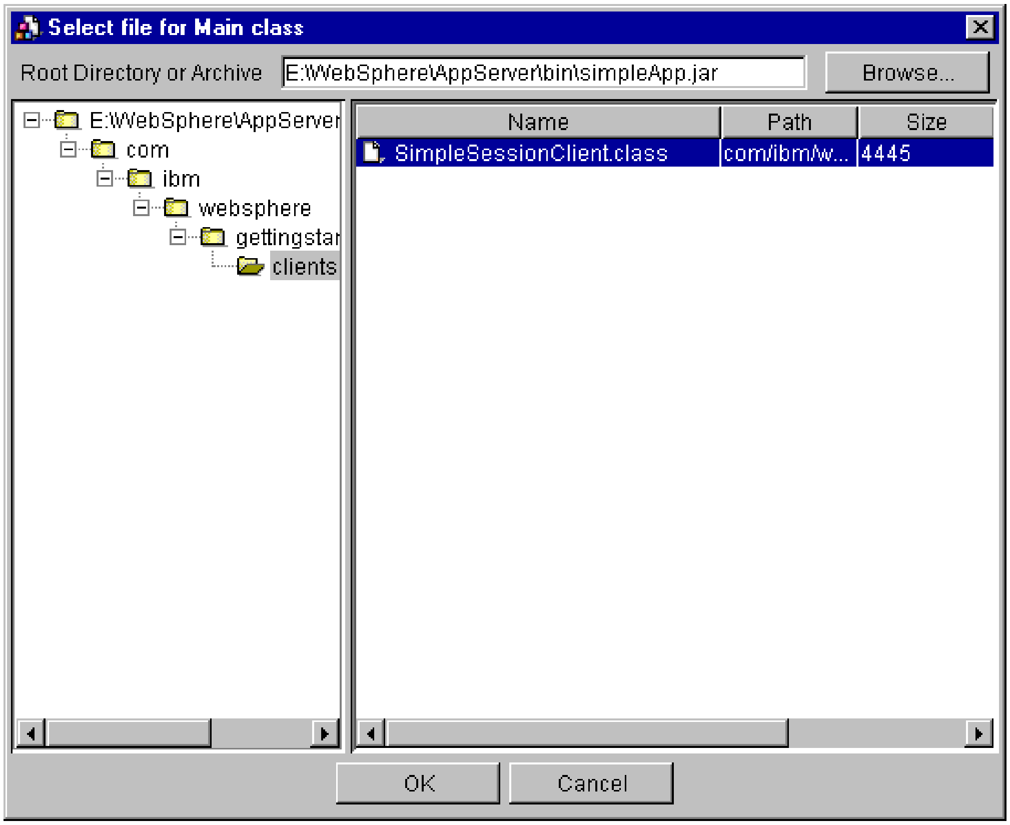
   c. Select the **com** folder (in the right hand pane).
   d. Click **Add**.



```
com/ibm/websphere/gettingstarted/clients/SimpleSessionClient.class
com/ibm/websphere/gettingstarted/clients/SimpleSessionClient.java
```

   e. Click **OK**.
   f. Click **Next**.

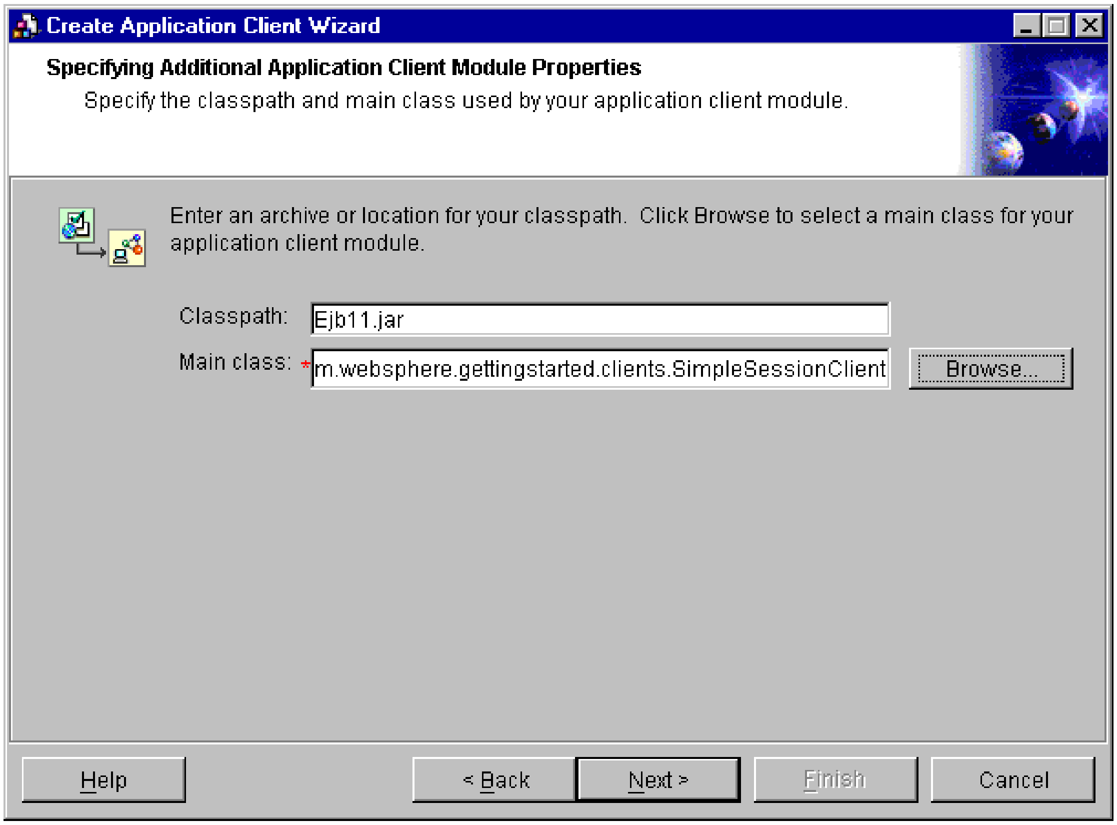
4. On the **Specifying Additional Application Client Modules Properties** panel:
   a. Enter **Ejb11.jar** for **Classpath**.
   b. Select **Browse** next to **Main Class** field.
   c. Expand the tree view in the left pane as far as it goes and select **clients**.
   d. Select **SimpleSessionClient.class** in the right panel.



Root Directory or Archive: E:\WebSphere\AppServer\bin\simpleApp.jar

Name	Path	Size
SimpleSessionClient.class	com/ibm/w...	4445

e. Click **OK.**



Specifying Additional Application Client Module Properties

Specify the classpath and main class used by your application client module.

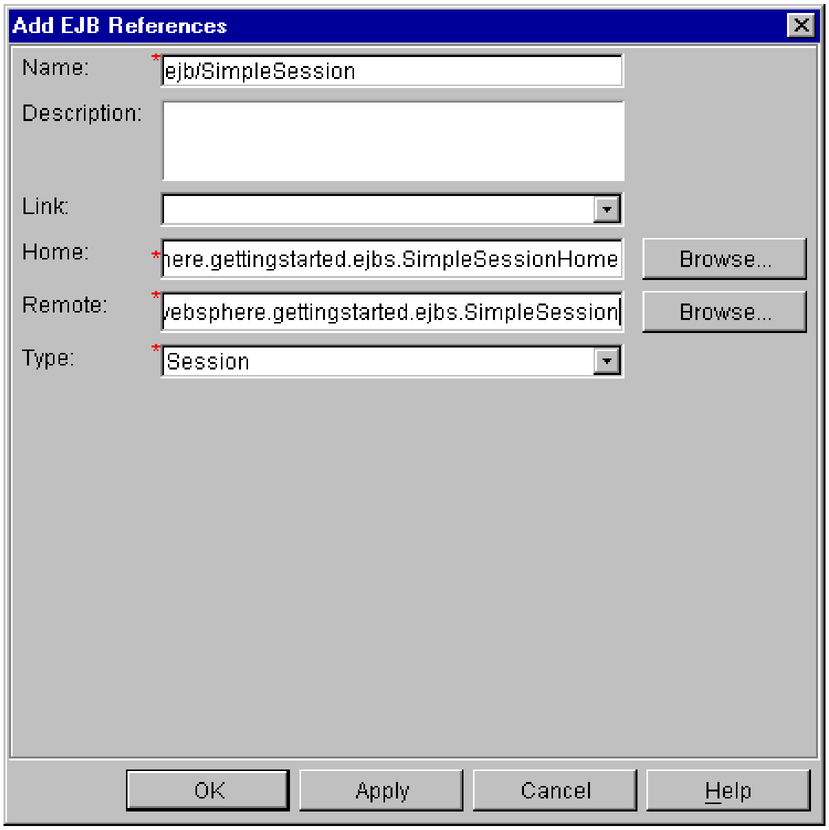
Enter an archive or location for your classpath. Click Browse to select a main class for your application client module.

Classpath: Ejb11.jar

Main class: * m.websphere.gettingstarted.clients.SimpleSessionClient

f. Click **Next.**

5. Click **Next** on the **Choosing Application Client ModulesIcons** panel.
6. Clcik **Add** on the **Adding EJB References** panel.
   a. Enter the following:
      - Name: **ejb/SimpleSession**
      - Home: **com.ibm.websphere.gettingstarted.ejbs.SimpleSessionHome**
      - Remote: **com.ibm.websphere.gettingstarted.ejbs.SimpleSession**



Name: ejb/SimpleSession

Description:

Link:

Home: ere.gettingstarted.ejbs.SimpleSessionHome

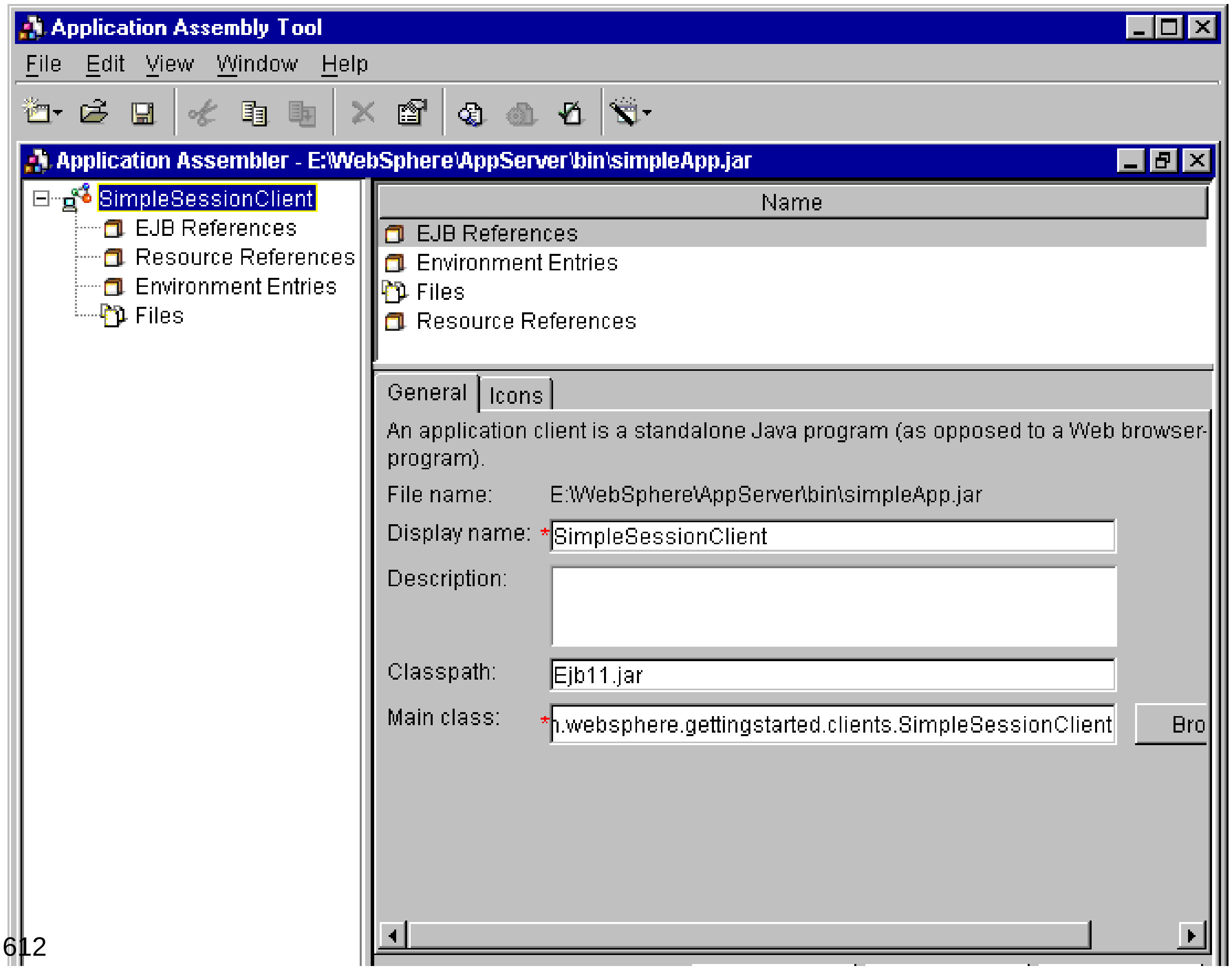
Remote: ebsphere.gettingstarted.ejbs.SimpleSession

Type: Session

b. Click **OK**.

c. Verify that your EJB reference is displayed in the EJB References list.

7. Click **Next**.
8. Click **Next** on the **Adding Resource References** panel.
9. Click the **Finish** on the **Specifying Environment Entries** panel. The application client module is displayed in the Application Assembly Tool.



An application client is a standalone Java program (as opposed to a Web browser program).

File name: E:\WebSphere\AppServer\bin\simpleApp.jar

Display name: SimpleSessionClient

Description:

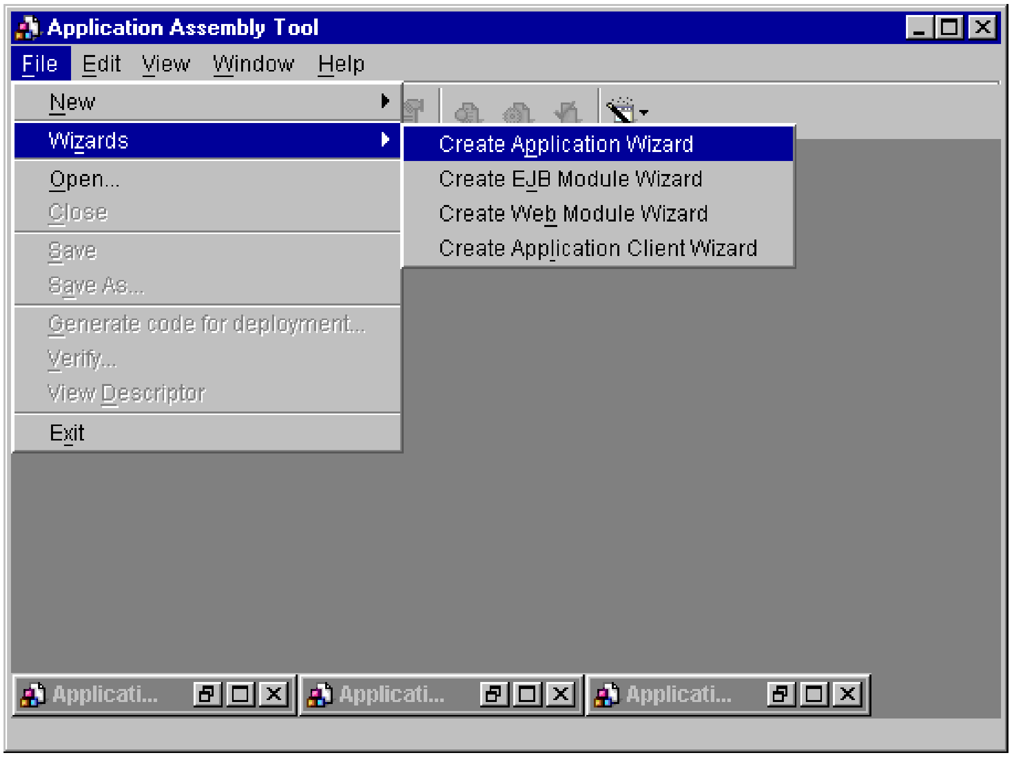
Classpath: Ejb11.jar

Main class: .websphere.gettingstarted.clients.SimpleSessionClient

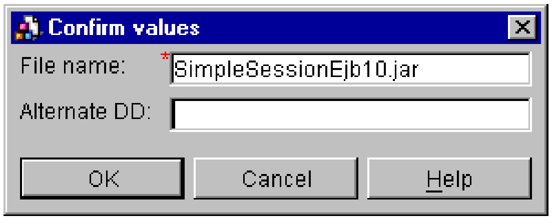
10. Save the application client jar file by clicking **File -> Save As**. Name it **simpleApp.jar**.

Assemble a J2EE application

1. Using AAT, create an .ear by clicking **File -> Wizards -> Create Application Wizard**.



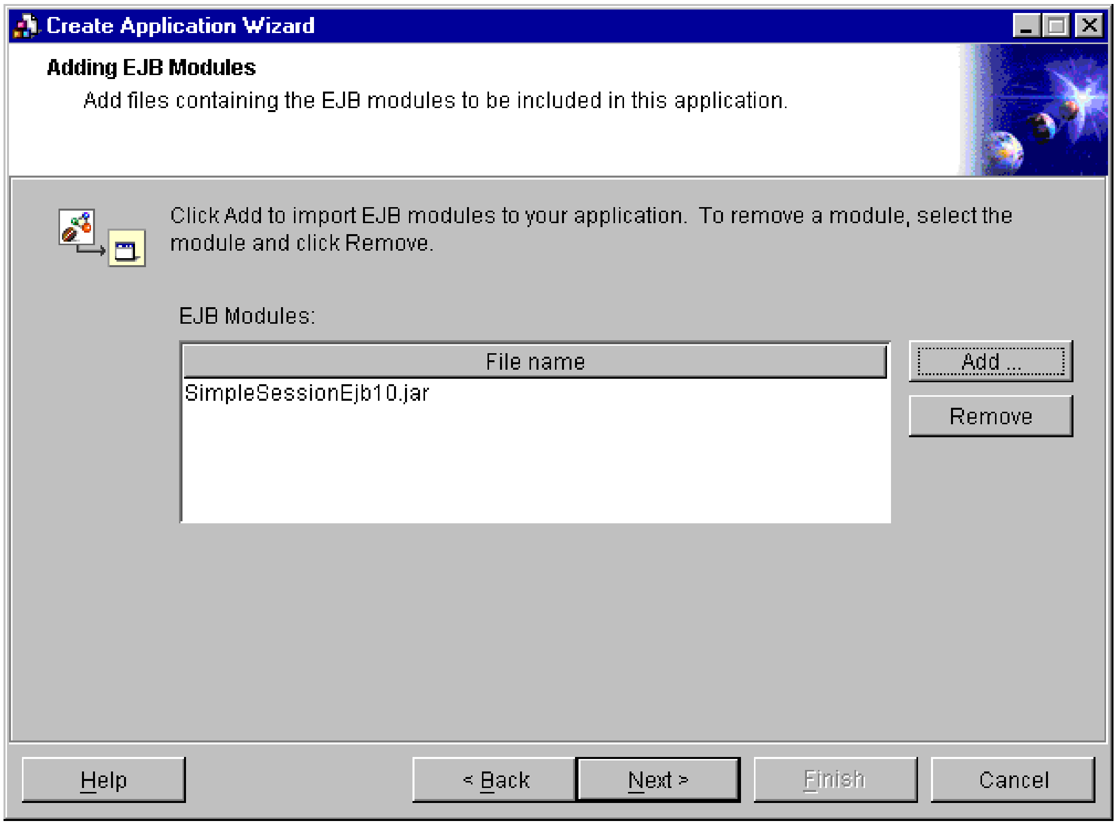
2. On the **Specifying Application Properties** panel:
    a. Enter the following:
        - Display Name: **SimpleSessionApp**
        - File Name: **simpleSession.ear**
    b. Click **Next**
3. Click Next on the **Adding Supplementary Files** panel.
4. Click Next on the **Choosing Application Icons** panel.
5. Click **Add...** on the **Adding EJB Modules** panel.
    a. Navigate to your **Ejb11.jar** file, highlight the file and click **Open**.
    b. Click **OK** on the resulting **Confirm values** dialog.



File name: *SimpleSessionEjb10.jar

Alternate DD:

The EJB module is now listed on the **Adding EJB Modules** panel.



Adding EJB Modules

Add files containing the EJB modules to be included in this application.

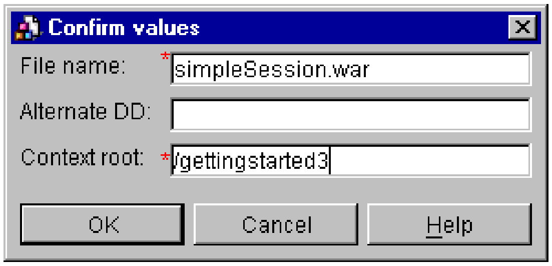
Click Add to import EJB modules to your application. To remove a module, select the module and click Remove.

EJB Modules:

File name
SimpleSessionEjb10.jar

   c. Click **Next.**

6. Click **Add...** on the **Adding Web Modules** panel.
   a. Navigate to your **simpleSession.war** file, highlight the file and click **Open**.
   b. In the resulting dialog, enter a Context Root: **/gettingstarted3**



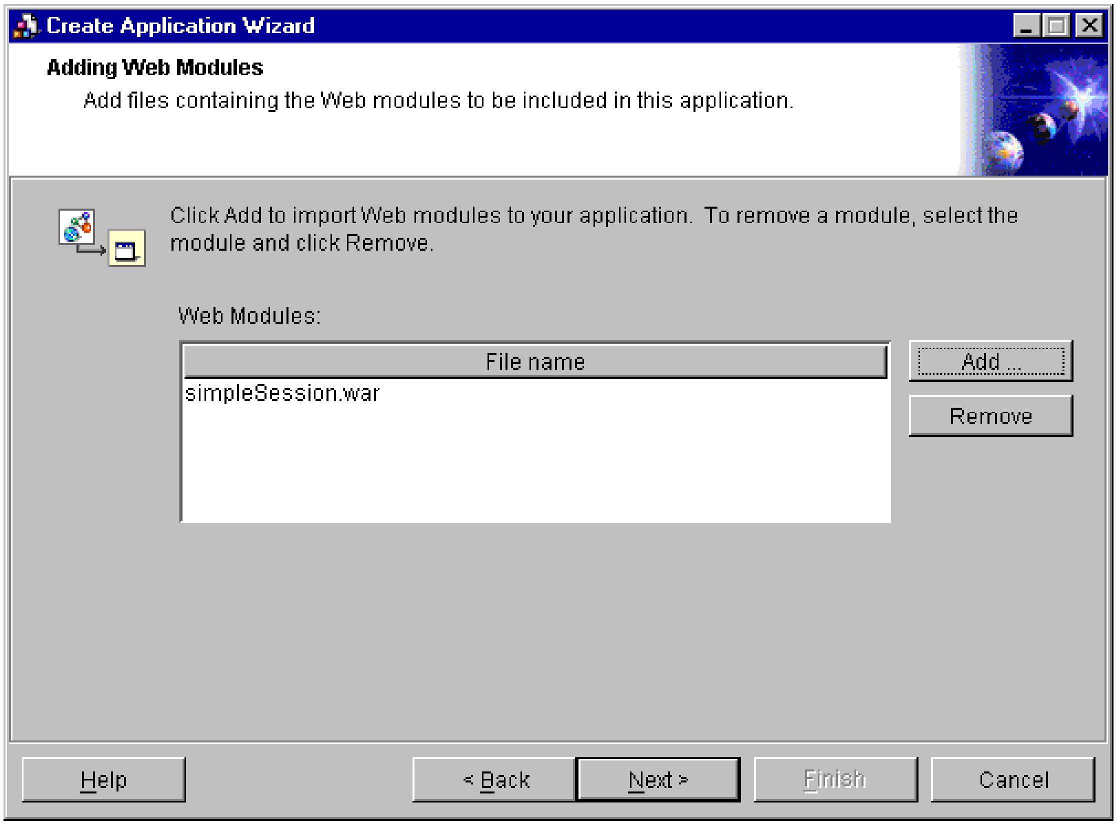
File name: * simpleSession.war

Alternate DD:

Context root: * /gettingstarted3

   c. Click **OK**.

   The Web module is now listed on the **Adding Web Modules** panel.



Adding Web Modules

Add files containing the Web modules to be included in this application.

Click Add to import Web modules to your application. To remove a module, select the module and click Remove.

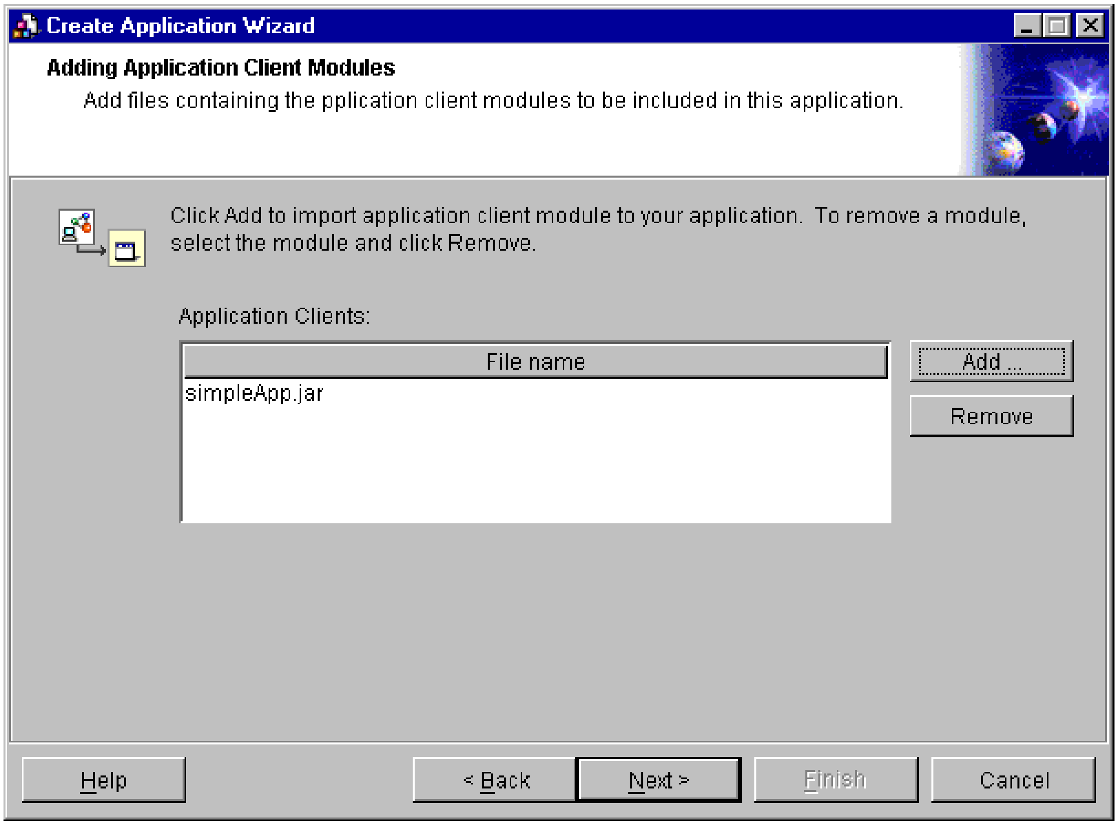
Web Modules:

File name
simpleSession.war

   d. Click **Next**.

7. Click **Add...** on the **Adding Application Client Modules** panel.
   a. Navigate to your **simpleApp.jar** file, highlight the file and click **Open**.
   b. Click **OK** on the resulting dialog.

   The application client module is now listed on the **Adding Application Client Modules** file.



Adding Application Client Modules

Add files containing the pplication client modules to be included in this application.

Click Add to import application client module to your application. To remove a module, select the module and click Remove.

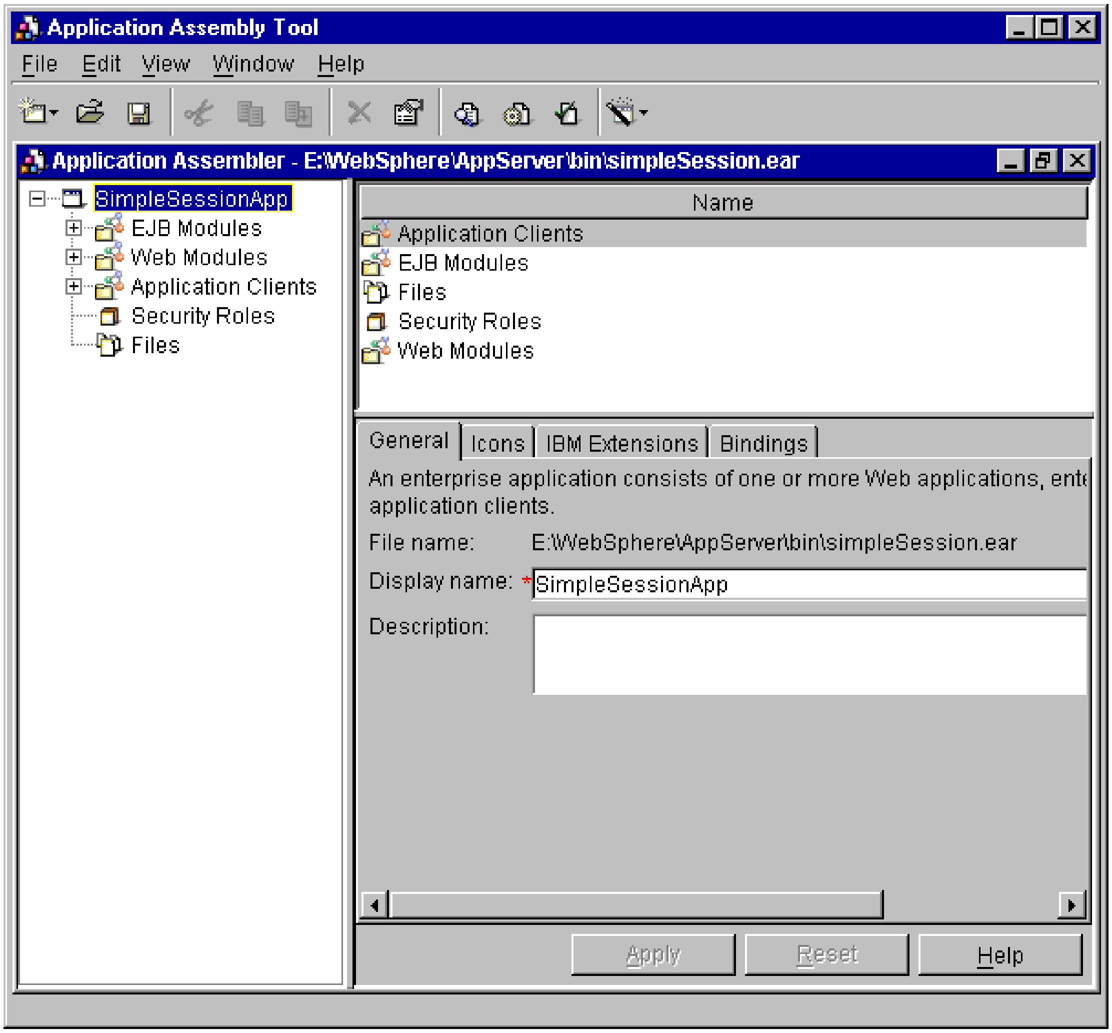
Application Clients:

File name
simpleApp.jar

   c. Click **Next.**

8. Click **Finish** on the **Adding Security Roles** panel.

   The application is now displayed in the Application Assembly Tool.



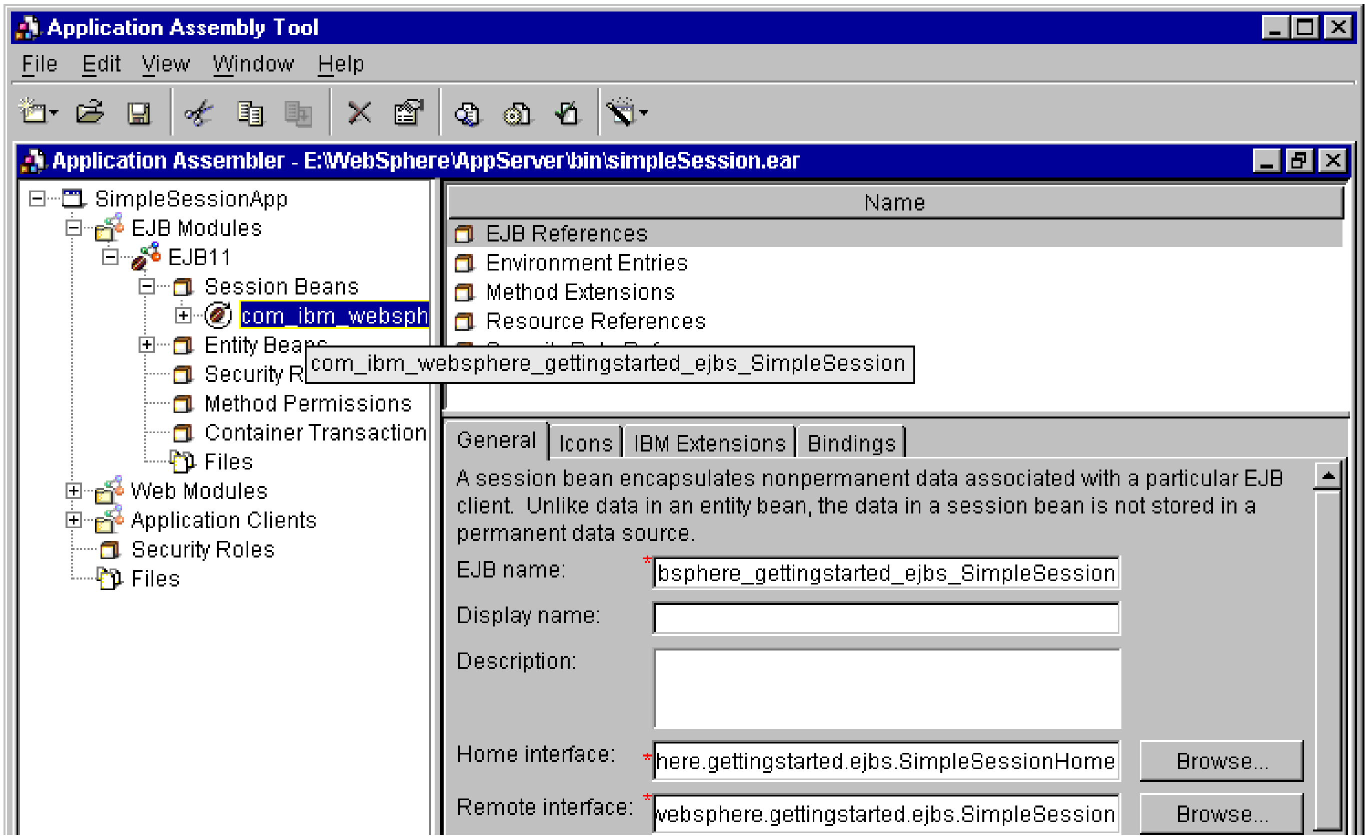
An enterprise application consists of one or more Web applications, ent
application clients.

File name: E:\WebSphere\AppServer\bin\simpleSession.ear

Display name: * SimpleSessionApp

Description:

9. Bind the EJB and references to it:
   a. In the assembly tool tree view, expand **EJB Modules -> EJB11 -> Session Beans** -> com_ibm...ejbs_SimpleSession.
   b. Click it to display its properties.



A session bean encapsulates nonpermanent data associated with a particular EJB client. Unlike data in an entity bean, the data in a session bean is not stored in a permanent data source.

EJB name: * bsphere_gettingstarted_ejbs_SimpleSession

Display name:

Description:

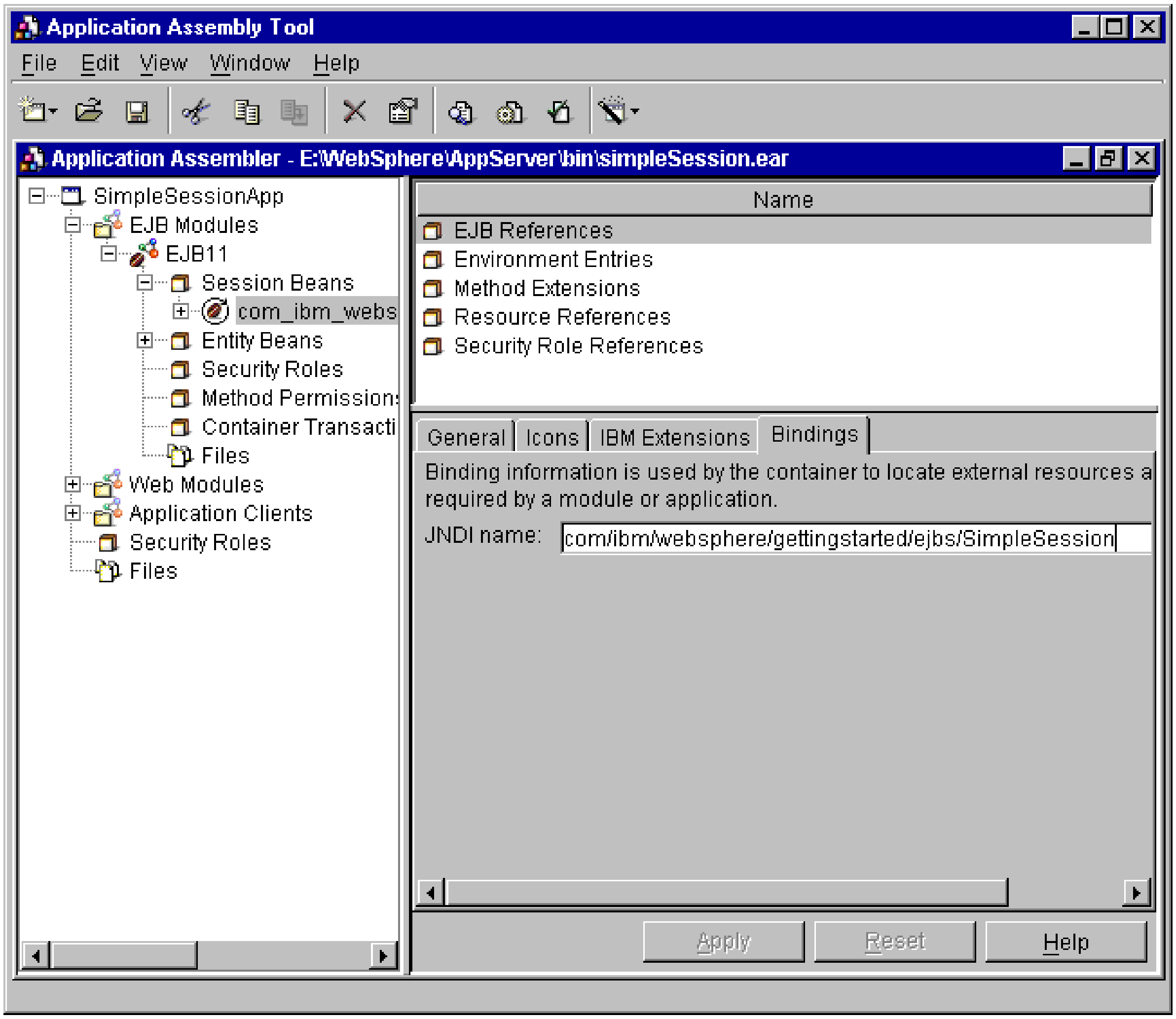
Home interface: * here.gettingstarted.ejbs.SimpleSessionHome

Remote interface: * websphere.gettingstarted.ejbs.SimpleSession

EJB class: phere.gettingstarted.ejbs.SimpleSessionBean

Session type: Stateless

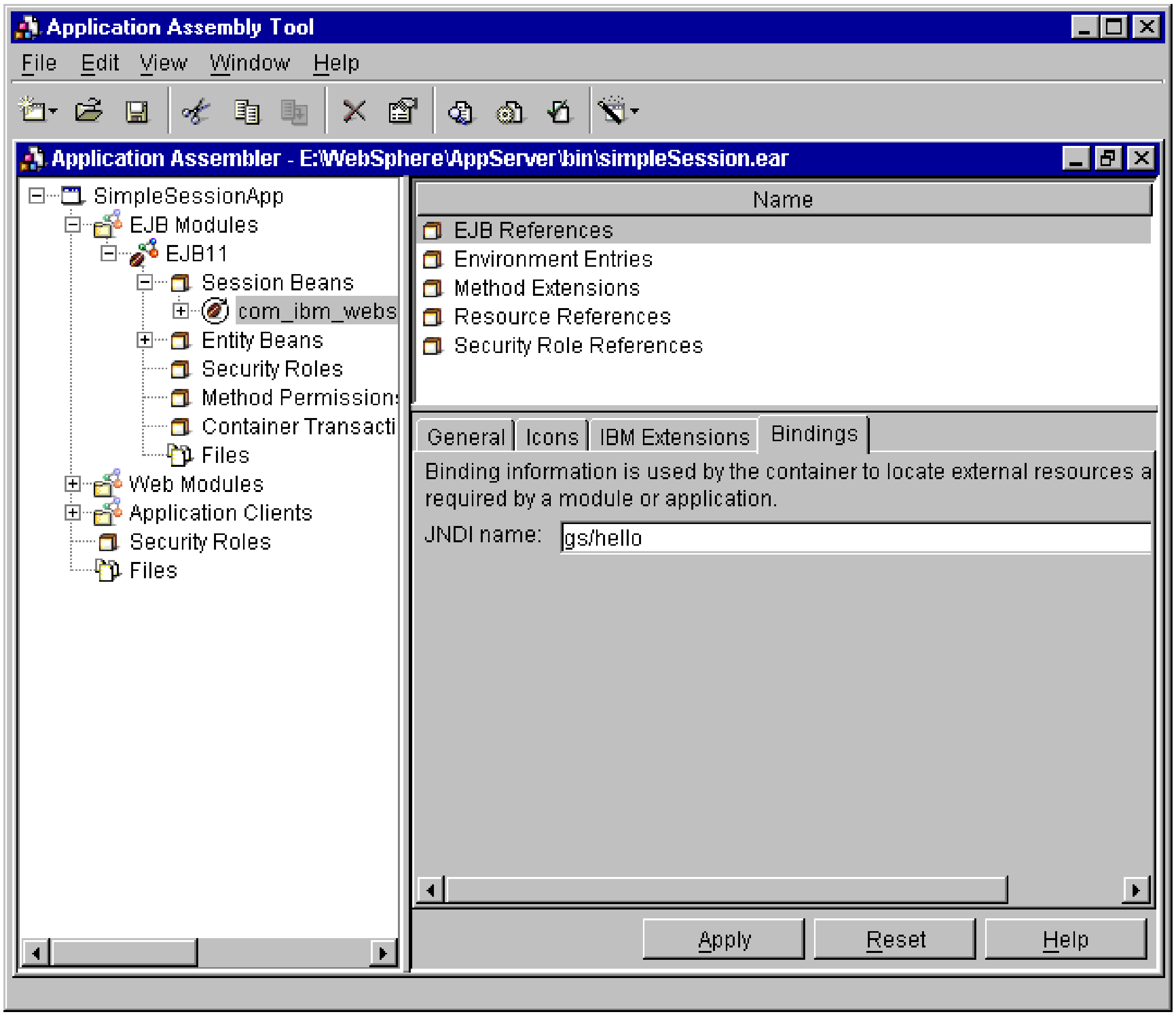
c. Select the **Bindings** tab.



Binding information is used by the container to locate external resources a required by a module or application.

JNDI name: com/ibm/websphere/gettingstarted/ejbs/SimpleSession

d. Replace the existing text string with `gs/hello` as the global JNDI name of the session bean.



Binding information is used by the container to locate external resources a required by a module or application.

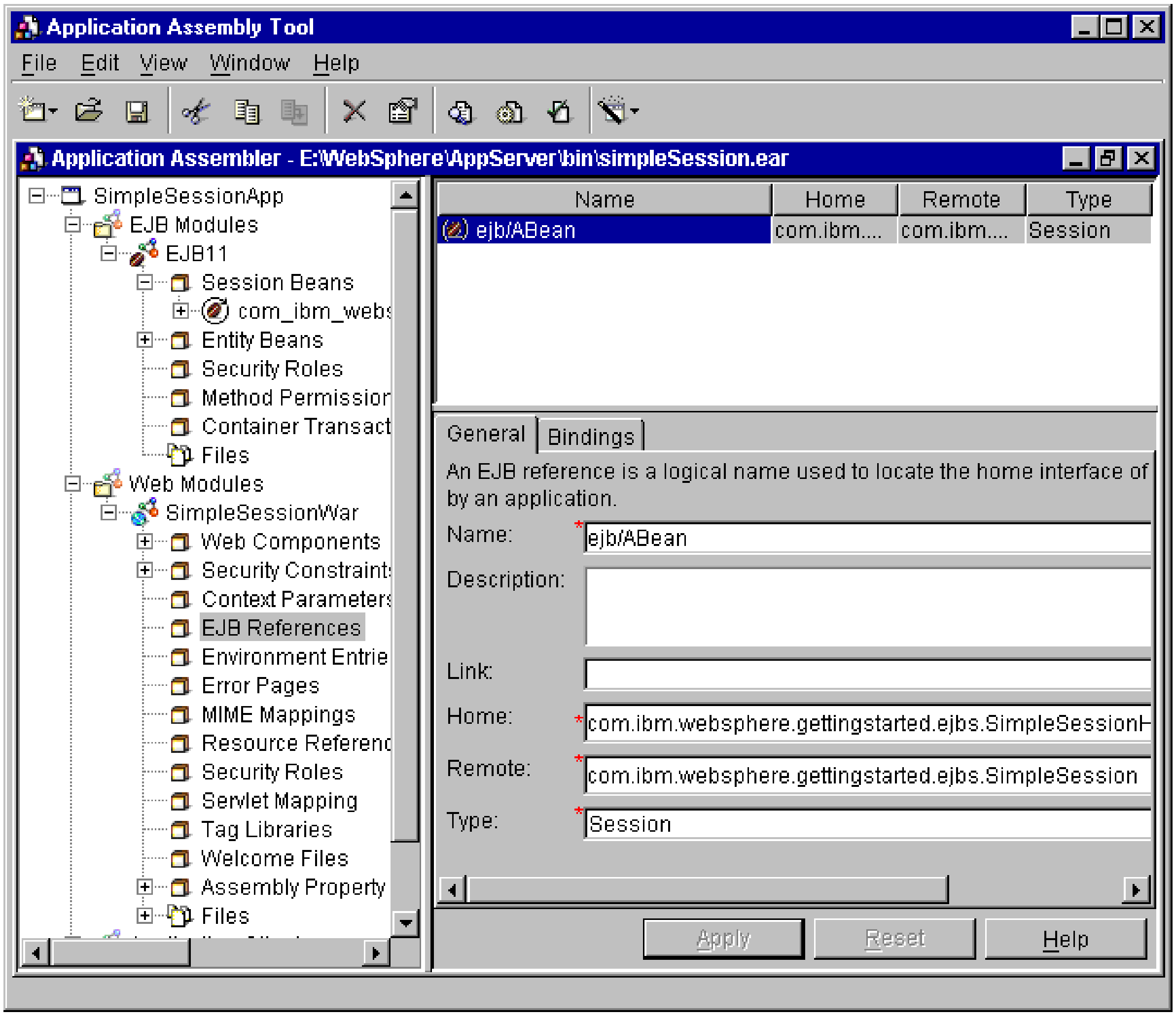
JNDI name: gs/hello

Although this example uses a short name for the session bean, consider using longer names in actual practice. A long name will help ensure that the name is unique in the global name space. An example of a suitable long name is:

```
com/ibm/websphere/gettingstarted/ejbs/SimpleSessionHome
```

e. Click **Apply**.

10. Bind the EJB reference (java:comp/env/ejb/ABean) to the global JNDI name of the bean:
    a. Expand **Web Modules -> SimpleSessionWar**.
    b. Select **EJB References**.
    c. Select in the right pane the particular reference called **ejb/ABean**.



Name	Home	Remote	Type
ejb/ABean	com.ibm....	com.ibm....	Session

An EJB reference is a logical name used to locate the home interface of by an application.

Name: ejb/ABean

Description:

Link:

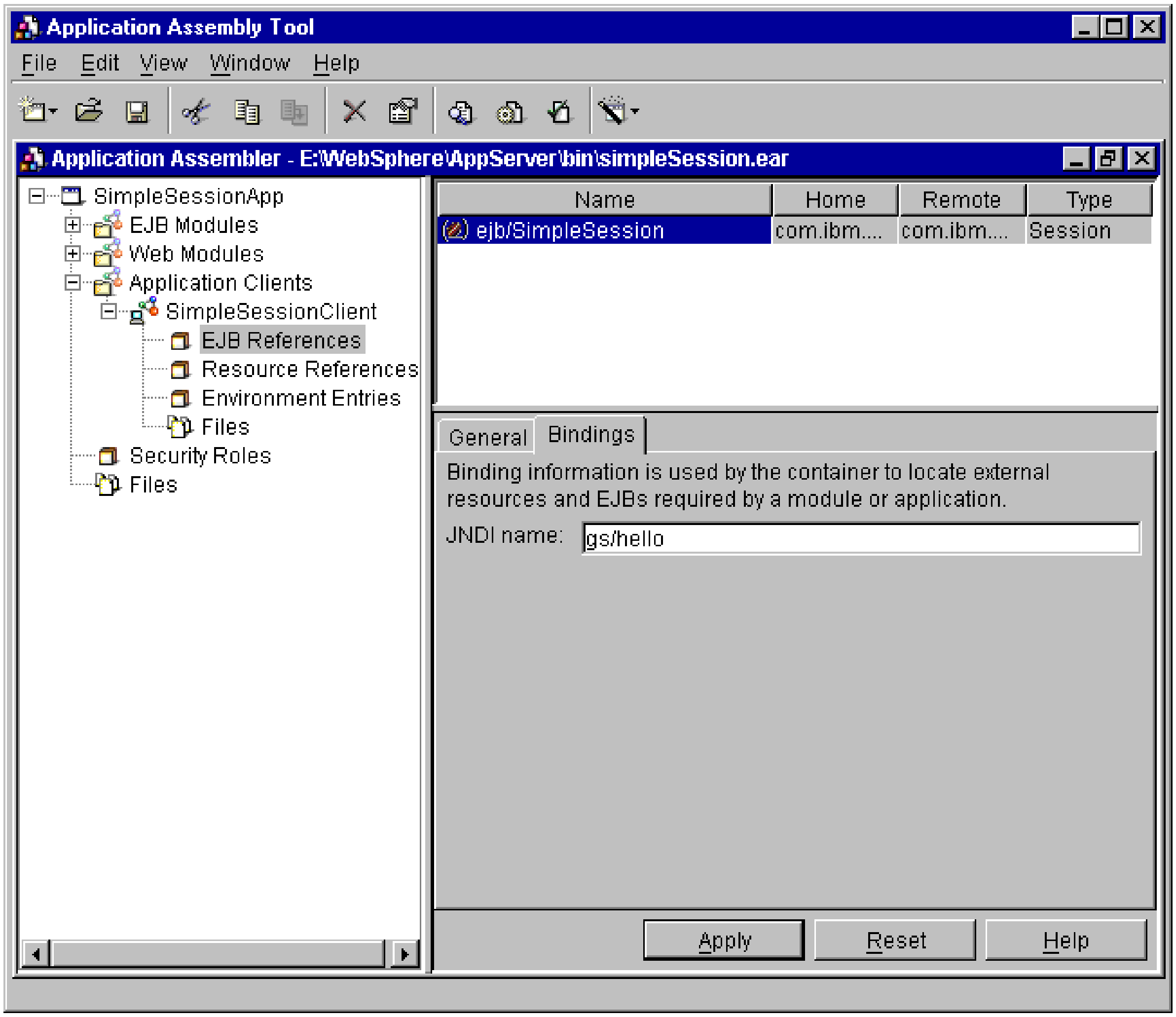
Home: com.ibm.websphere.gettingstarted.ejbs.SimpleSessionH

Remote: com.ibm.websphere.gettingstarted.ejbs.SimpleSession

Type: Session

   d. Select the **Bindings** tab.
   e. Replace the existing text string with **gs/hello**. This is the JNDI Name that you assigned to the enterprise bean in an earlier step.
   f. Click **Apply**.

11. Bind the EJB reference:
   a. Expand **Application Clients -> SimpleSessionClient**.
   b. Select **EJB References**.
   c. Select in the right pane the particular reference to your session bean: **ejb/SimpleSession**
   d. Select the **Bindings** tab.
   e. Replace the existing text string with **gs/hello**. This is the JNDI Name that you assigned to the enterprise bean in an earlier step.



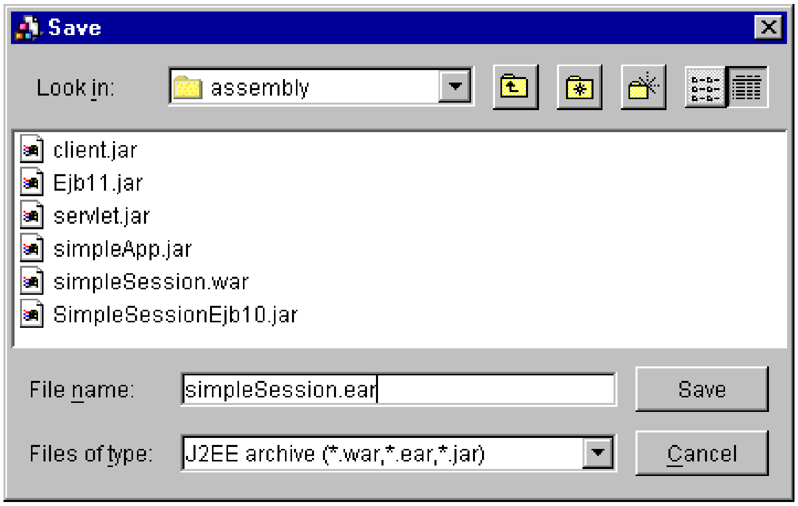
Name	Home	Remote	Type
ejb/SimpleSession	com.ibm....	com.ibm....	Session

Binding information is used by the container to locate external resources and EJBs required by a module or application.

JNDI name: gs/hello

f. Click **Apply**.

12. Save the .ear file by clicking **File -> Save As**. Name it **simpleSession.ear**.



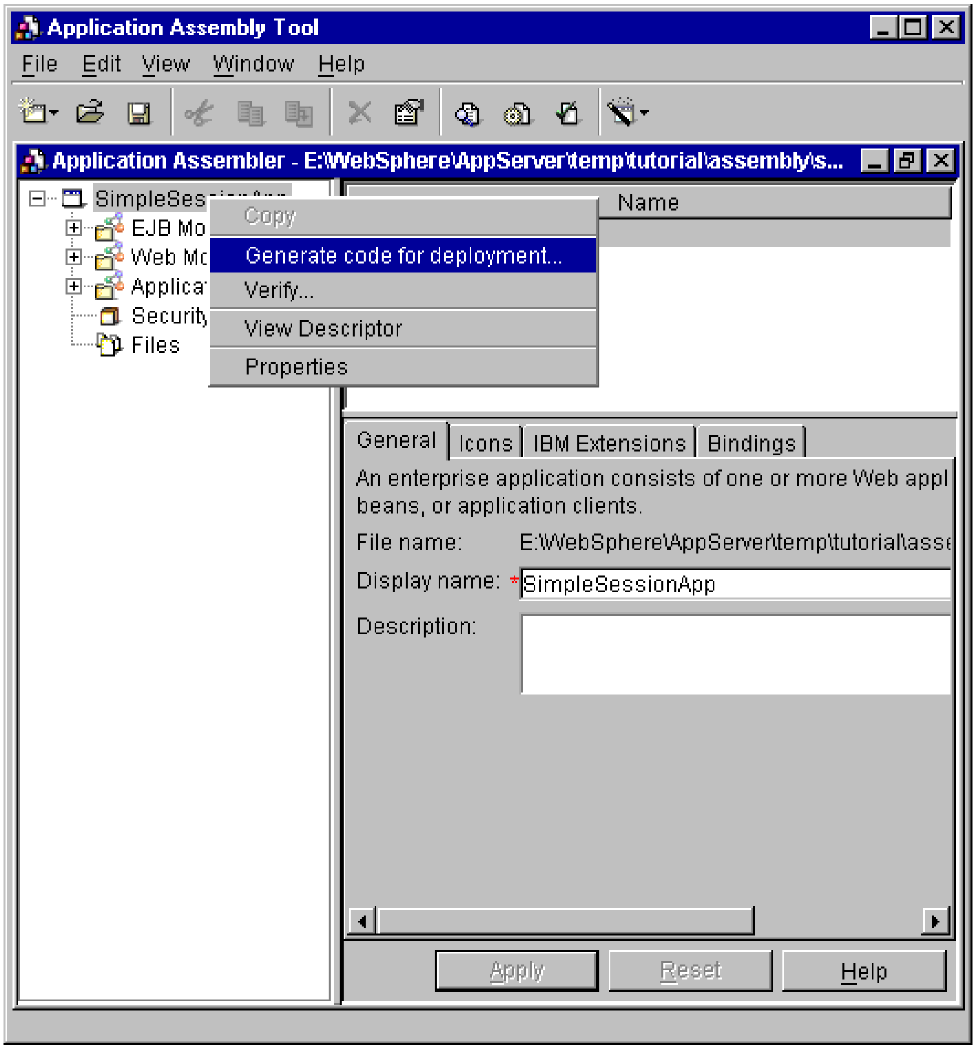
File name: simpleSession.ear

Files of type: J2EE archive (*.war,*.ear,*.jar)

Generate code for deployment

a. Right-click the .ear file by selecting the top entry in the tree view of the assembly tool.

b. Select **Generate Code for Deployment**.


Application Assembly Tool
File Edit View Window Help
Application Assembler - E:\WebSphere\AppServer\temp\tutorial\assembly\s...
SimpleSes
EJB Mo
Web Mo
Applica
Security
Files
Copy
Generate code for deployment...
Verify...
View Descriptor
Properties
Name
General
Icons
IBM Extensions
Bindings
Apply
Reset
Help

An enterprise application consists of one or more Web appl beans, or application clients.

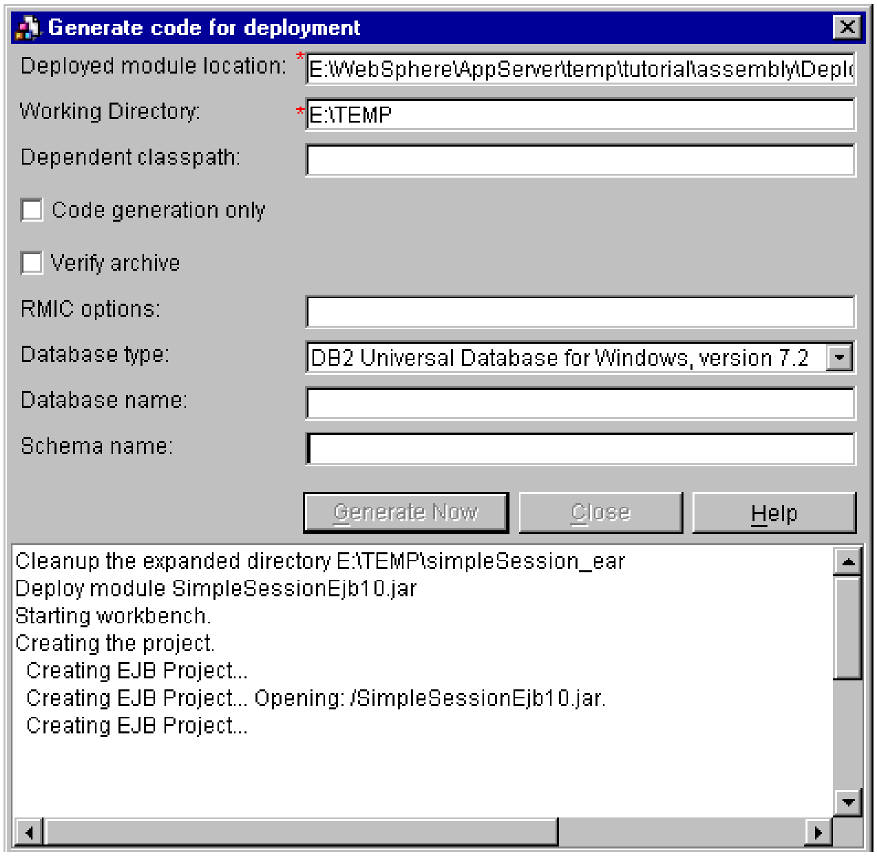
File name: E:\WebSphere\AppServer\temp\tutorial\asse

Display name: * SimpleSessionApp

Description:

c. Select appropriate **Database Type.**

d. Click **Generate Now**.



Deployed module location: * E:\WebSphere\AppServer\temp\tutorial\assembly\Deplo

Working Directory: * E:\TEMP

Dependent classpath:

☐ Code generation only

☐ Verify archive

RMIC options:

Database type: DB2 Universal Database for Windows, version 7.2

Database name:

Schema name:

```
Cleanup the expanded directory E:\TEMP\simpleSession_ear
Deploy module SimpleSessionEjb10.jar
Starting workbench.
Creating the project.
  Creating EJB Project...
  Creating EJB Project... Opening: /SimpleSessionEjb10.jar.
  Creating EJB Project...
```

e. When deployment is complete, close this window.

Another way to generate code for deployment is to use the Deployment Tool directly. The Application Assembly Tool calls the command line Deployment Tool for you when you use the **Generate Now** button and accompanying dialog.

Exit the Application Assembly Tool

Exit the assembly tool using the **File** -> **Exit** option.

What's next?

Now that you have your application assembled, you can deploy it on the application server, which includes "installing" it. The next steps are outlined by the Application deployment tutorial.

6.7.2: Application deployment tutorial

During this tutorial, you will deploy the J2EE application that you assembled during the Application assembly tutorial. Deployment involves configuring the server and supporting resources (you will use the existing default server), and installing the application. Installing the application means configuring the settings pertaining to this application in the server runtime. The runtime settings include choices such as whether to automatically load servlets, or to precompile JSP files.

Prerequisites

Before performing this tutorial, either:

- Complete the Application assembly tutorial
- Plan to use the already assembled "shortcut_simpleSession.ear" file described later in this tutorial

Overview of steps (requires 15 to 30 minutes)

1. Obtain the tutorial application
2. Start the administrative server
3. Open the administrative console
4. Prepare the application server and needed resources
5. Install the application
6. Regenerate the Web server plug-in
7. Modify Module Visibility, stop the application server, and start it again

Paths through the tutorial

- **Quick path:** If your objective is to practice these steps as quickly as possible, skip the explanations marked with the ℹ graphic.
- **Maximum learning path:** If your objective is to understand and explore the product capabilities, read the ℹ notes and browse the links they provide to additional documentation.

Obtain the tutorial application

If you completed the Application assembly tutorial, you simply need to know the location of the simpleSession.ear file that you assembled, including generating code for deployment. It should be located in the directory:

`product_installation_root``/temp/tutorial/assembly/Deployed_simpleSession.ear`

If you did not perform the assembly tutorial, and thus plan to use the already assembled .ear file:

1. Create a directory named "tutorial" under the path:

   `product_installation_root``/temp`
2. Click here to access the .zip file containing the tutorial application components.
3. Save the .zip file to

   `product_installation_root``/temp/tutorial`
4. Use your favorite .zip or .jar utility to extract the tutorial.zip content into the tutorial directory.
5. Copy the shortcut_Deployed_simpleSession.ear file. Name the copy Deployed_simpleSession.ear.

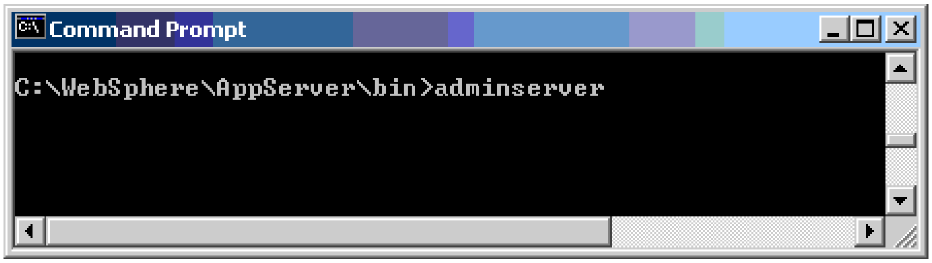
Start the administrative server

First, start the product administrative server.

1. Open a system command prompt.
2. Change directory to:

   `product_installation_root``/bin`
3. Enter: adminserver

Leave the command window open, as you will use it again later to start the server.



```
C:\WebSphere\AppServer\bin>adminserver
```

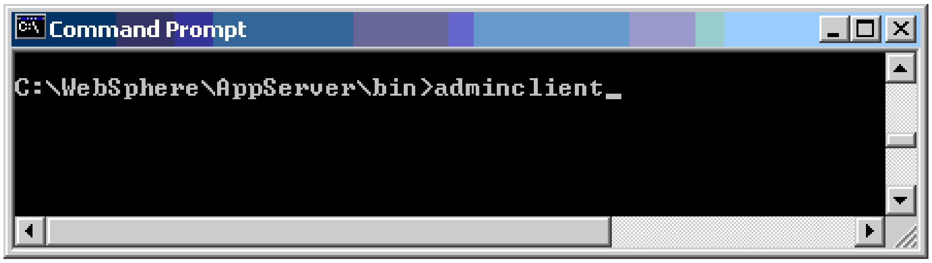
Open the administrative console

Now it is time to deploy (install) the application in the server runtime. To do so, you will use the administrative console.

1. Open a system command prompt.
2. Change directory to:

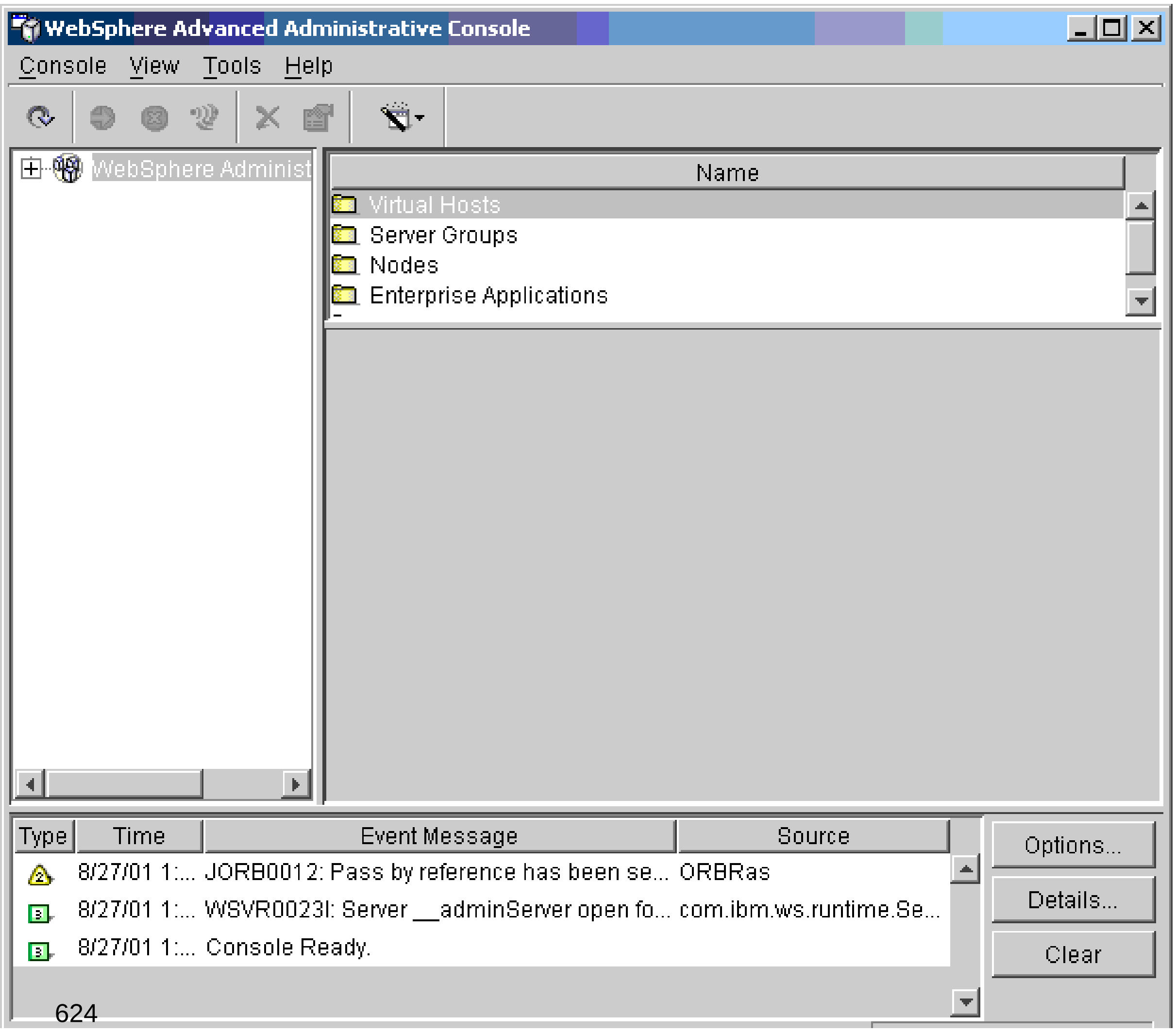
   `product_installation_root``/bin`

3. Enter: adminclient



```
C:\WebSphere\AppServer\bin>adminclient_
```

For more information about variations on starting and stopping the administrative server, see see the corresponding InfoCenter article.



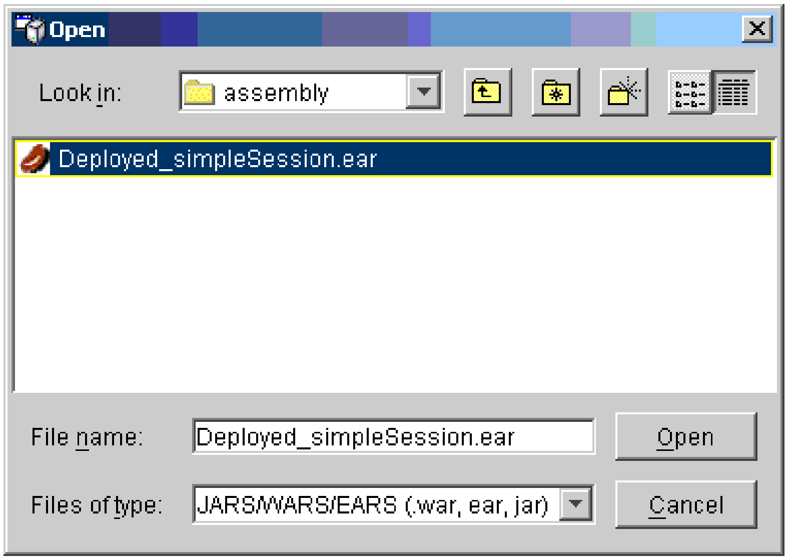
Type	Time	Event Message	Source
⚠	8/27/01 1:...	JORB0012: Pass by reference has been se...	ORBRas
ℹ	8/27/01 1:...	WSVR0023I: Server __adminServer open fo...	com.ibm.ws.runtime.Se...
ℹ	8/27/01 1:...	Console Ready.	

Prepare the application server and needed resources

For the purposes of the tutorial, use the WebSphere runtime environment "as is." The default application server and other resources are everything you need to deploy the simpleSession application.

Install the application

1. In the console tree view, right-click **WebSphere Administrative Domain -> Enterprise Applications**.
2. From the resulting menu, click **Install Enterprise Application** to launch the **Install Enterprise Application** wizard.
3. On the **Specifying the Application or Module** panel:
    1. Ensure that the **Browse for file on node** field is set to your current node.
    2. Ensure **Install Application** is selected.
    3. Click **Browse**, next to the **Path** field. Browse for the Deployed_simpleSession.ear file and select it.

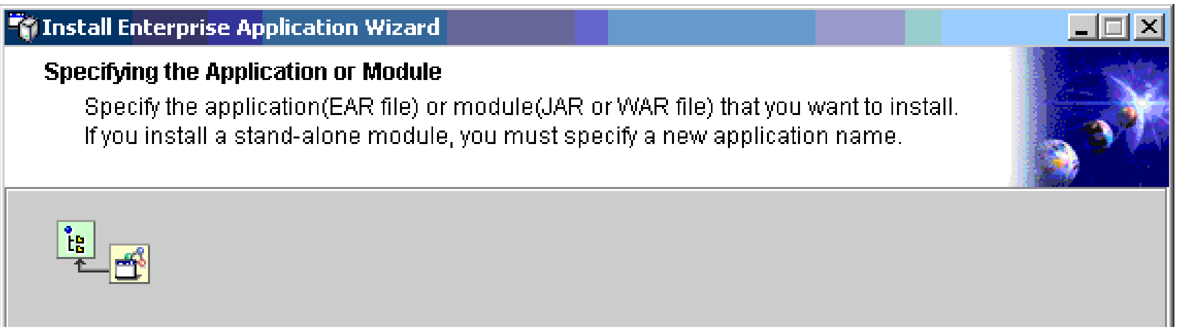
       Look in: assembly

       Deployed_simpleSession.ear

       File name: Deployed_simpleSession.ear

       Files of type: JARS/WARS/EARS (.war, ear, jar)

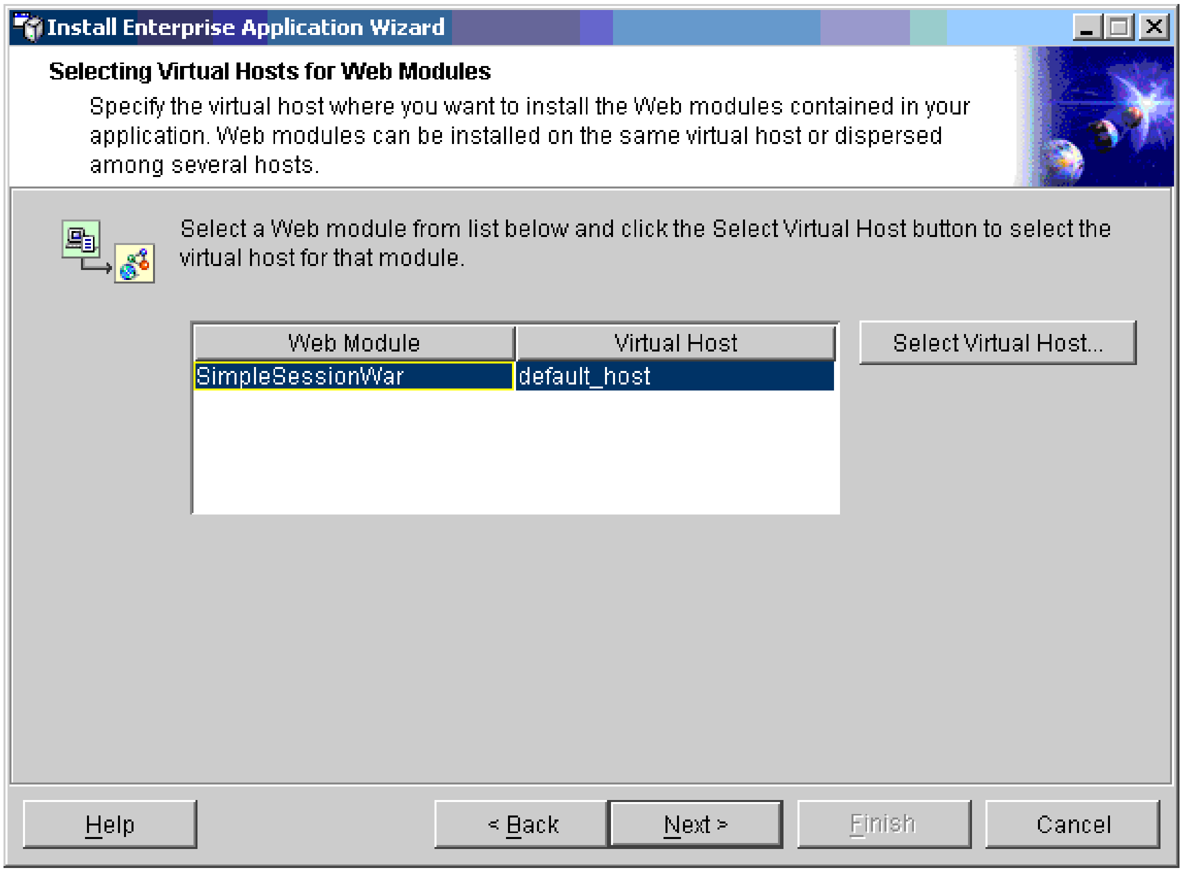
       Verify that its name is displayed in the **Path** field now.
    4. Specify SimpleSessionApp as the **Application name**.

       Install Enterprise Application Wizard

       Specifying the Application or Module

       Specify the application(EAR file) or module(JAR or WAR file) that you want to install.
       If you install a stand-alone module, you must specify a new application name.
    5. Click **Next**.
4. On the **Mapping Users to Roles** panel, click **Next**.
5. On the **Mapping EJB RunAs Roles to Users** panel, click **Next**.
6. On the **Binding Enterprise Beans to JNDI Names** panel, verify that the **JNDI Name** is set to gs/hello, and then click **Next**.
7. On the **Mapping EJB References to Enterprise Beans** panel, verify that the JNDI Name is set to gs/hello, and then click **Next**.
8. On the **Mapping Resource References to Resources** panel, click **Next**.
9. On the **Specifying the Default Datasources for EJB Modules** panel, click **Next**.
10. On the **Specifying Data Sources for Individual CMP Beans** panel, click **Next**.
11. On the **Selecting Virtual Hosts for Web Modules** panel, ensure that the **Virtual Host** is set to default_host, then click **Next**.



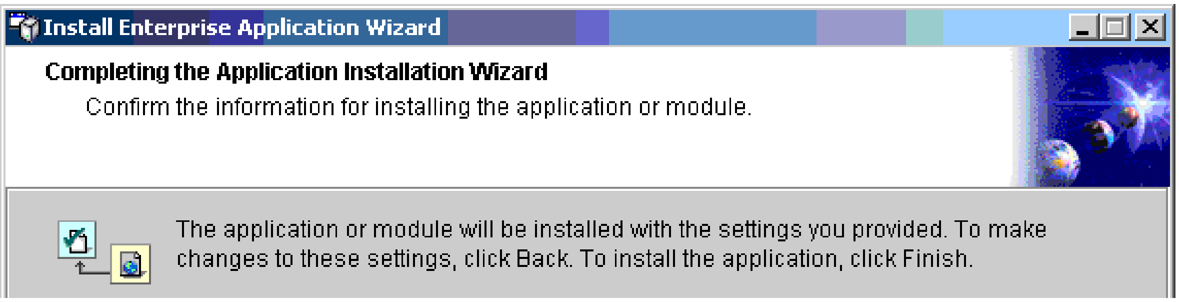
Selecting Virtual Hosts for Web Modules

Specify the virtual host where you want to install the Web modules contained in your application. Web modules can be installed on the same virtual host or dispersed among several hosts.

Select a Web module from list below and click the Select Virtual Host button to select the virtual host for that module.

Web Module	Virtual Host
SimpleSessionWar	default_host

12. On the **Selecting Application Server** panel, ensure that the EJB11 and SimpleSessionWar modules are going to reside on **Application Server** "Default Server," then click **Next**.
13. Click **Finish** on the **Completing the Application Installation Wizard** panel.

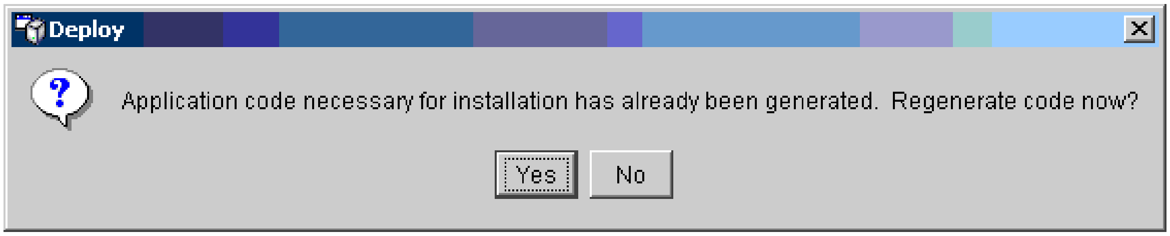


Completing the Application Installation Wizard

Confirm the information for installing the application or module.

The application or module will be installed with the settings you provided. To make changes to these settings, click Back. To install the application, click Finish.

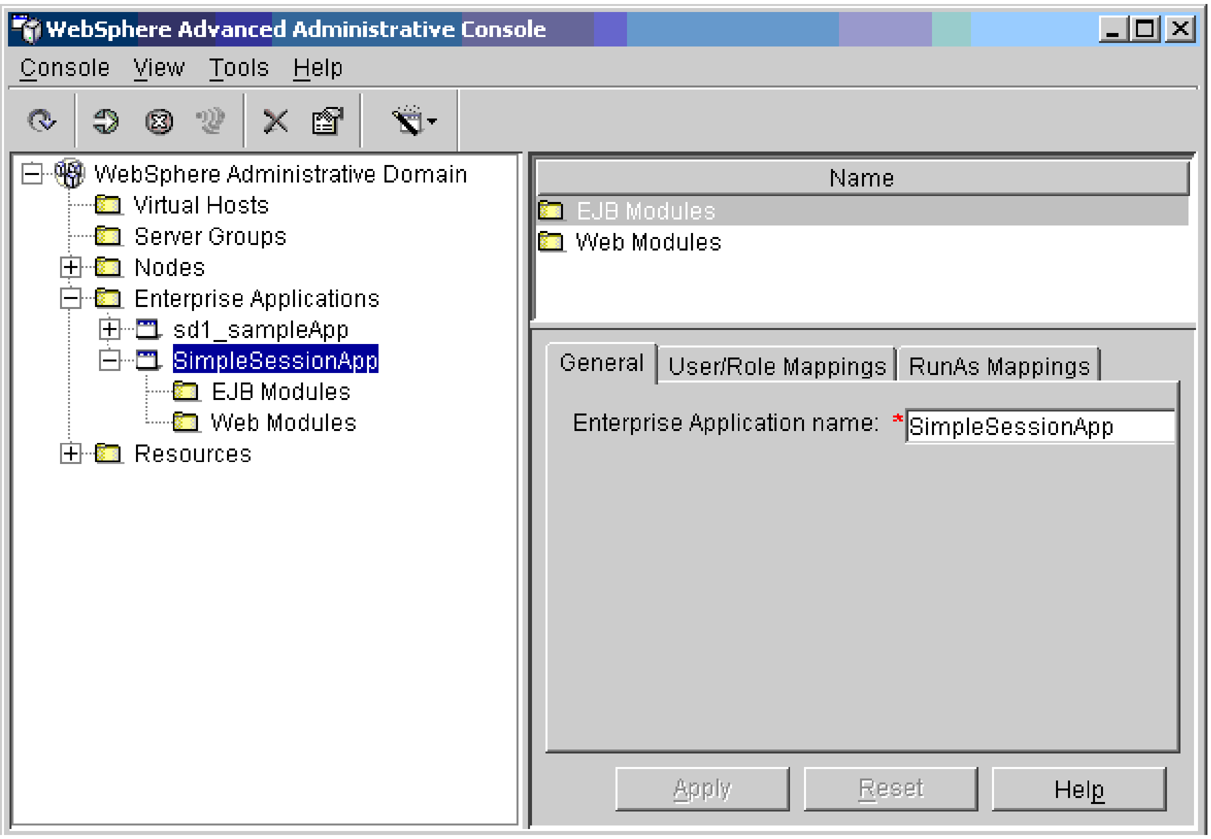
14. When prompted whether to regenerate code, click **No**.



Application code necessary for installation has already been generated. Regenerate code now?

15. Look for the message confirming successful installation of the application. It might be a minute before it is displayed.

You can now view the SimpleSessionApp in the console tree view.



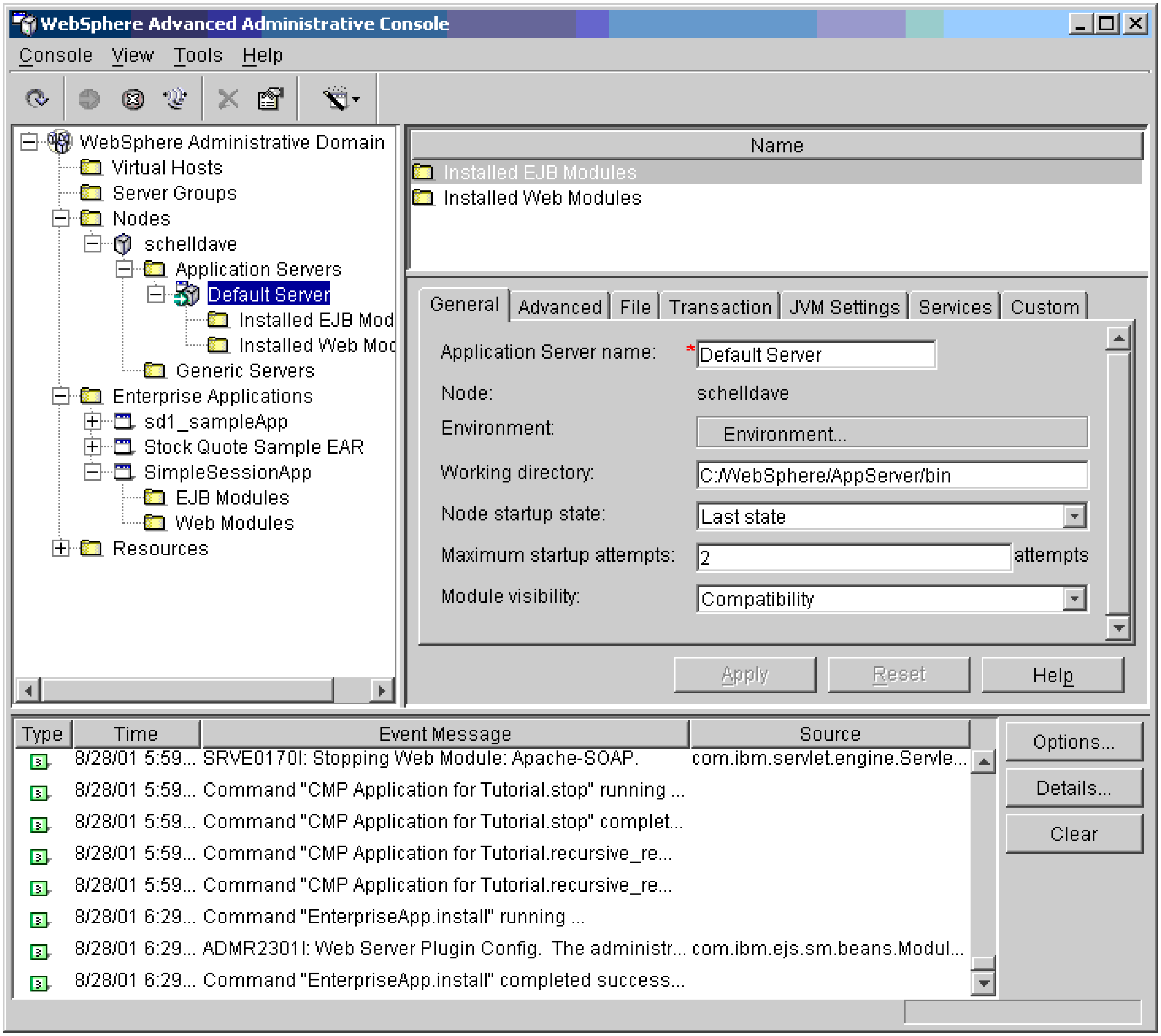
Enterprise Application name: * SimpleSessionApp

Regenerating the plug-in configuration

1. In the console tree view, right-click **WebSphere Administrative Domain** -> **Nodes** -> ***your_host_name***.
2. From the resulting menu, select **Regen Webserver Plugin**.
3. In the area at the bottom of the console, look for the **Event Message** saying that the Plugin regeneration has been completed. It might take a moment before it is displayed.

Modifying Module Visibility, stopping the application server, and starting it again

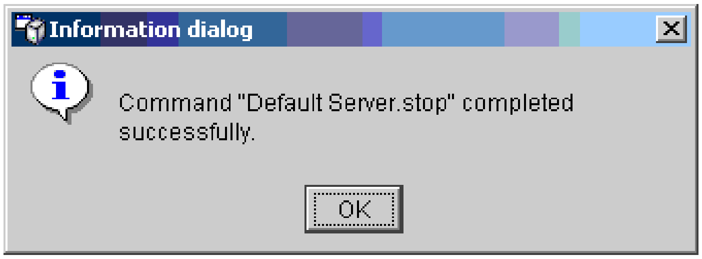
1. In the console tree view, click **WebSphere Administrative Domain** -> **Nodes** -> ***your_host_name*** -> **Application Servers** -> **Default Server**.
2. Change the **Module Visibility** setting of the **Default Server** to "Compatibility" and click **Apply**.



Application Server name: * Default Server
Node: schelldave
Environment:
Working directory: C:/WebSphere/AppServer/bin
Node startup state: Last state
Maximum startup attempts: 2 attempts
Module visibility: Compatibility

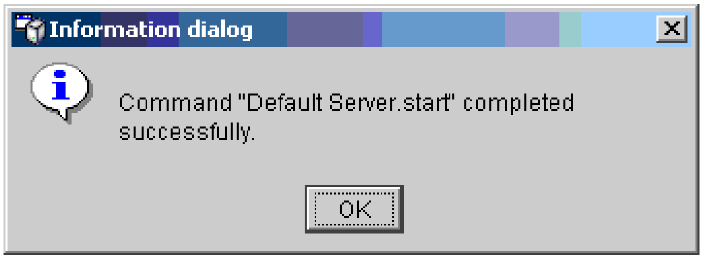
```
Type  Time          Event Message                                                Source
      8/28/01 5:59... SRVE0170I: Stopping Web Module: Apache-SOAP.               com.ibm.servlet.engine.Servle...
      8/28/01 5:59... Command "CMP Application for Tutorial.stop" running ...
      8/28/01 5:59... Command "CMP Application for Tutorial.stop" complet...
      8/28/01 5:59... Command "CMP Application for Tutorial.recursive_re...
      8/28/01 5:59... Command "CMP Application for Tutorial.recursive_re...
      8/28/01 6:29... Command "EnterpriseApp.install" running ...
      8/28/01 6:29... ADMR2301I: Web Server Plugin Config.  The administr...   com.ibm.ejs.sm.beans.Modul...
      8/28/01 6:29... Command "EnterpriseApp.install" completed success...
```

3. Now right-click the **Default Server**.
4. From the resulting menu, click **Stop**.
5. Look for the message that the server was stopped successfully.



Command "Default Server.stop" completed successfully.

6. Now right-click **Default Server** again and click **Start** on the resulting menu.
7. Again, wait for the confirmation message.



Command "Default Server.start" completed successfully.

Because adding a new application requires a change to the server configuration file, the server must be stopped and started again to pick up the change. However, there are some types of changes you can make to an installed application that do not require stopping the server and starting it again. See the dynamic reloading and hot deployment documentation for more information.

What's next?

Now that you have your application deployed, it is time to verify that users will be able to access it, using either a Web client or Java client. The next steps are described in the Application testing tutorial.

6.7.3: Application testing tutorial

During this tutorial, you will test the application that you deployed during the Application deployment tutorial. This involves trying the Web and Java clients of the application to ensure the application can be accessed by users.

Prerequisites

You need to have performed the Application deployment tutorial successfully so that you have a deployed application to test.

Overview of steps (requires 10 minutes)

1. Ensure that the application, application server, and Web server are running
2. Test the Web client
3. Test the Java application client

Paths through the tutorial

- **Quick path:** If your objective is to practice these steps as quickly as possible, skip the explanations marked with the ℹ graphic.
- **Maximum learning path:** If your objective is to understand and explore the product capabilities, read the ℹ notes and browse the links they provide to additional documentation.

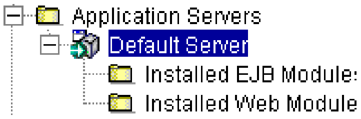
Ensure that the application, application server, and Web server are running

For successful use of the client:

- The application must be running
- The application server hosting the application must be running
- In the case of the Web client, the Web server must be running

To verify whether the application and application server are running:

1. Start the administrative server, or verify that it is running already.
2. Start the Default Server, or verify that it is running already. To verify its status:
   1. Open the administrative console.
   2. Expand the tree view to see **WebSphere Administrative Domain** -> **Nodes** -> ***your_host_name*** -> **Application Servers** -> **Default Server**.
   3. Verify that the icon next to the Default Server's name in the tree shows that it is running. The icon should feature a green circle with an arrow in it.

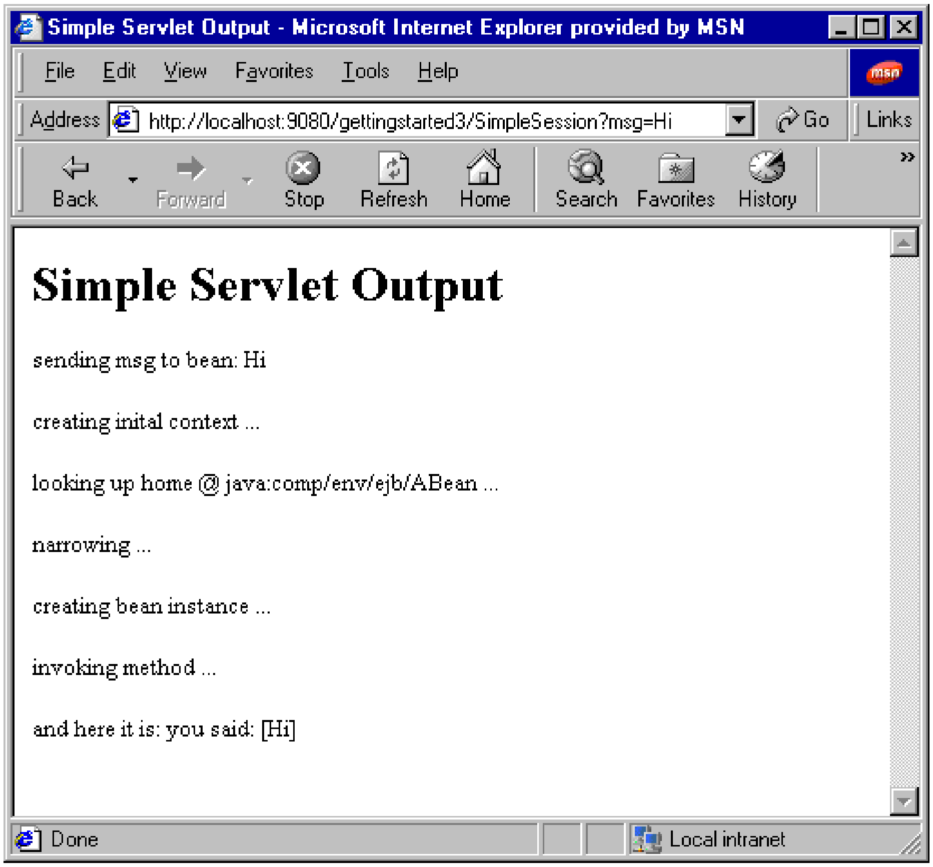


3. Verify that the SimpleSessionApp is running by right-clicking it in the console tree view under **WebSphere Administrative Domain** -> **Enterprise Applications** -> **SimpleSessionApp** and selecting **Show Status** from the resulting menu.
4. Ensure that your Web server is running.

Test the Web client

Test the Web client of simpleSession by browsing the URL:

`http://`*`hostname`*`:9080/gettingstarted3/SimpleSession?msg=Hi`


Simple Servlet Output - Microsoft Internet Explorer provided by MSN
File Edit View Favorites Tools Help
Address
http://localhost:9080/gettingstarted3/SimpleSession?msg=Hi
Go
Links
Back
Forward
Stop
Refresh
Home
Search
Favorites
History
Done
Local intranet

Simple Servlet Output

sending msg to bean: Hi

creating inital context ...

looking up home @ java:comp/env/ejb/ABean ...

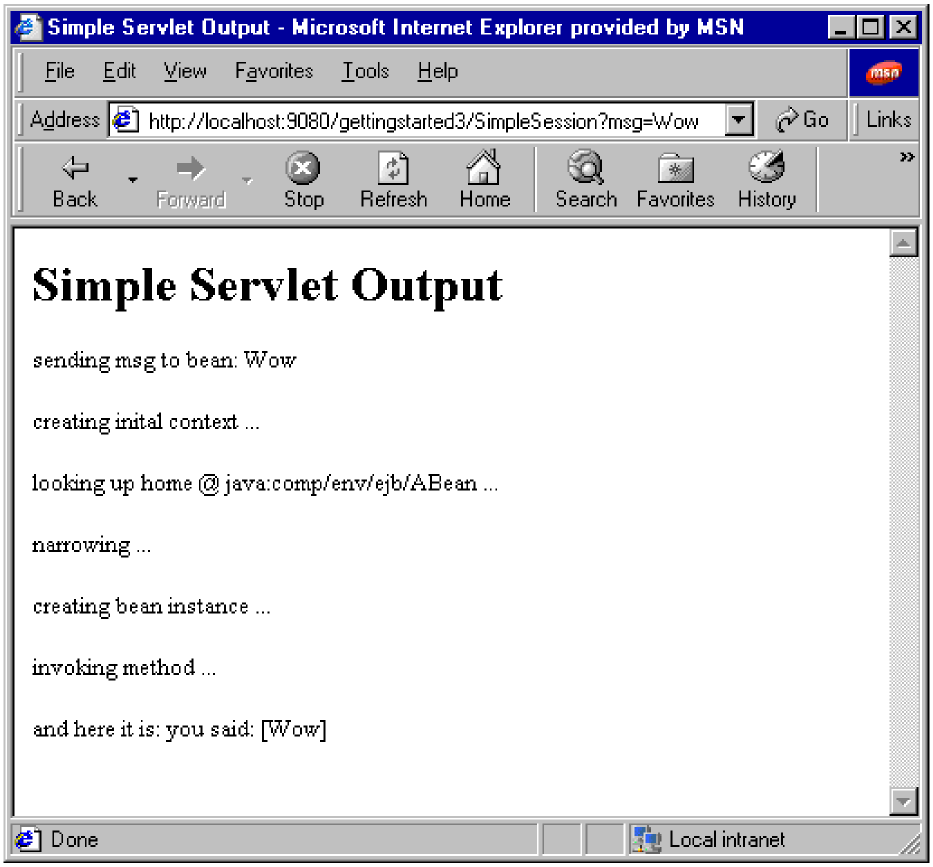
narrowing ...

creating bean instance ...

invoking method ...

and here it is: you said: [Hi]

The client will return your original message of "Hi" (notice the end of the URL: ?msg=Hi). You can send other messages and have them echoed back to you, such as ?msg=Wow


Simple Servlet Output - Microsoft Internet Explorer provided by MSN
File Edit View Favorites Tools Help
Address
http://localhost:9080/gettingstarted3/SimpleSession?msg=Wow
Go
Links
Back
Forward
Stop
Refresh
Home
Search
Favorites
History
Done
Local intranet

Simple Servlet Output

sending msg to bean: Wow

creating inital context ...

looking up home @ java:comp/env/ejb/ABean ...

narrowing ...

creating bean instance ...

invoking method ...

and here it is: you said: [Wow]

Test the Java application client

Test the Java application client by launching your client application in the client container.

1. Open a system command prompt.

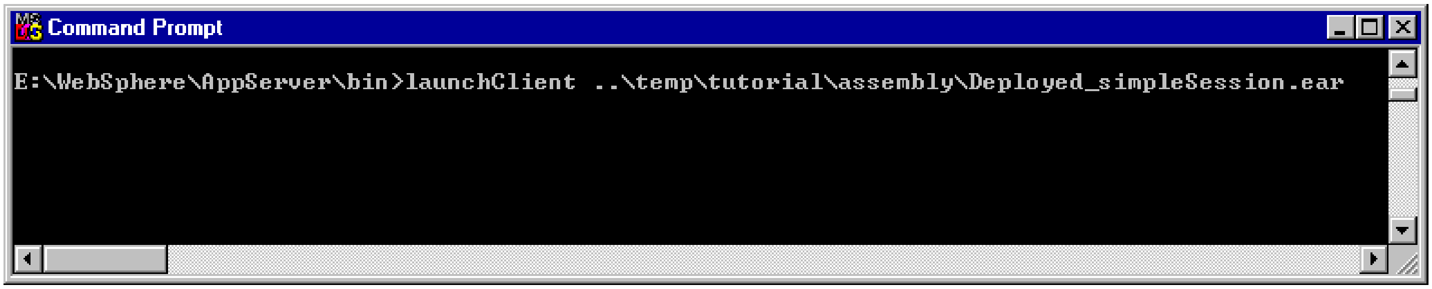
2. Change directory to:

   ```
   product_installation_root/bin
   ```

3. Enter the command:

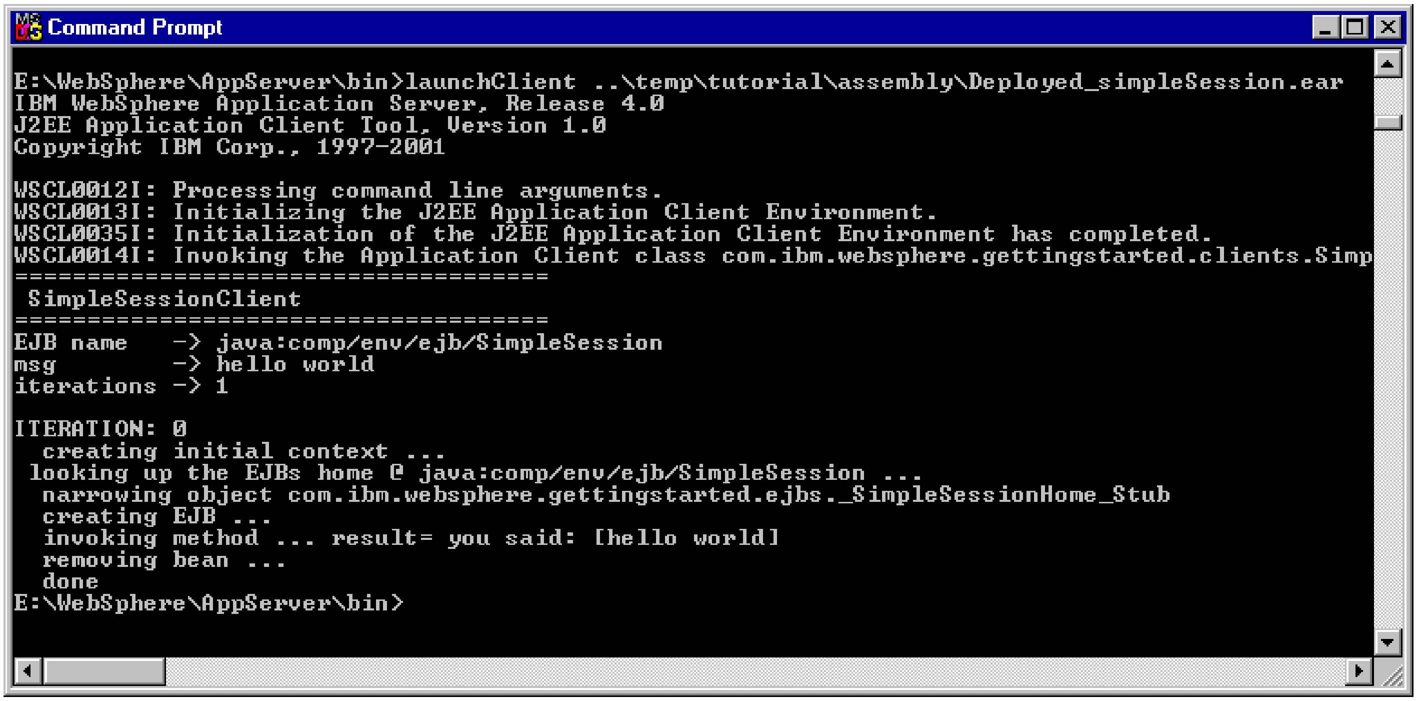
   ```
   launchClient ../temp/tutorial/assembly/Deployed_simpleSession.ear
   ```

   (Remember, on Windows systems, the forward slashes should be backslashes).



```
E:\WebSphere\AppServer\bin>launchClient ..\temp\tutorial\assembly\Deployed_simpleSession.ear
```

The client will contact the enterprise bean and return a "hello world" message.



```
E:\WebSphere\AppServer\bin>launchClient ..\temp\tutorial\assembly\Deployed_simpleSession.ear
IBM WebSphere Application Server, Release 4.0
J2EE Application Client Tool, Version 1.0
Copyright IBM Corp., 1997-2001

WSCL0012I: Processing command line arguments.
WSCL0013I: Initializing the J2EE Application Client Environment.
WSCL0035I: Initialization of the J2EE Application Client Environment has completed.
WSCL0014I: Invoking the Application Client class com.ibm.websphere.gettingstarted.clients.Simp
=====================================
 SimpleSessionClient
=====================================
EJB name   -> java:comp/env/ejb/SimpleSession
msg        -> hello world
iterations -> 1

ITERATION: 0
  creating initial context ...
 looking up the EJBs home @ java:comp/env/ejb/SimpleSession ...
  narrowing object com.ibm.websphere.gettingstarted.ejbs._SimpleSessionHome_Stub
  creating EJB ...
  invoking method ... result= you said: [hello world]
  removing bean ...
  done
E:\WebSphere\AppServer\bin>
```

The above command is issued from the bin directory because the argument providing the path to the Deployed_simpleSession.ear is relative to the bin directory. Actually, you should be able to issue the command from any directory, as long as you correctly specify the path to the .ear file whose application client you want to launch. More information is available about the launchClient command.

What's next?

Now that you have assembled and deployed an unsecured application, and tested your results, you might be interested in securing your application. The Application security tutorial describes how to do so, using local operating system authentication and login forms.

6.7.4: Application security tutorial

During this tutorial, you will enable security in all three containers of your simpleSession application:

- Web
- EJB
- Client

You will use the Application Assembly Tool (AAT) to declare and define J2EE security roles, as well as to control authorization on various J2EE modules. You will also enable security in the application server runtime and test your settings.

Authentication will be performed using the local operating system user registry. This example only uses declarative security. It does not illustrate any of the programmatic methods supported by the J2EE programming model.

Prerequisites

You need to have performed the Application deployment tutorial and Application testing tutorial successfully in order to perform this tutorial successfully. If you ran into trouble testing your application, you can still perform this tutorial to practice the steps involved, but the testing phase of this tutorial will be unsuccessful.

Overview of steps (requires 45 to 60 minutes)

1. Enable security in your application
2. Enable security in the application server runtime
3. Remove the unsecured version of the application
4. Install the secured application
5. Regenerate the Web server plug-in and save the server configuration
6. Stop the administrative server and start it again
7. Ensure the application, application server, and Web server are running
8. Test the Web client
9. Test the Java client
10. Disable security in the application server runtime

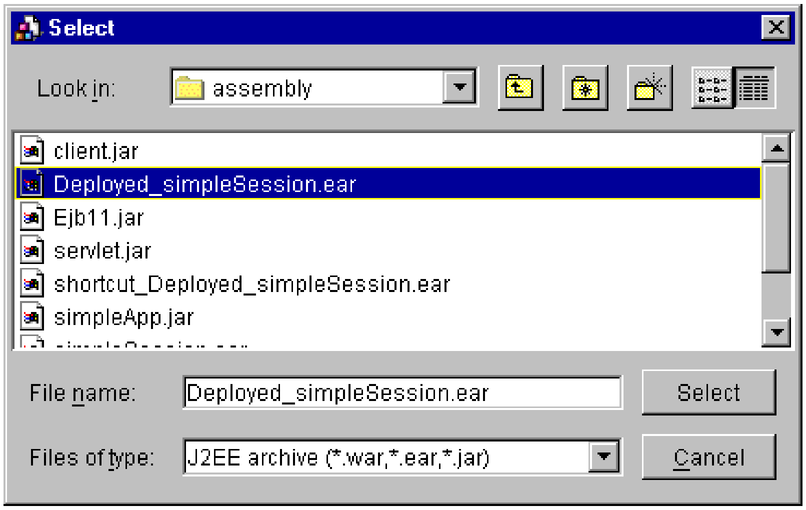
Paths through the tutorial

- **Quick path:** If your objective is to practice these steps as quickly as possible, skip the explanations marked with the ℹ graphic.
- **Maximum learning path:** If your objective is to understand and explore the product capabilities, read the ℹ notes and browse the links they provide to additional documentation.

Enable security in your application

1. Start the Application Assembly Tool, as discussed in the Application assembly tutorial.
2. Edit the Deployed_simpleSession.ear application.

   On the **Welcome to Application Assembly Tool** panel:
   1. Click the **Existing** tab.
   2. Click **Browse** next to the File name field.

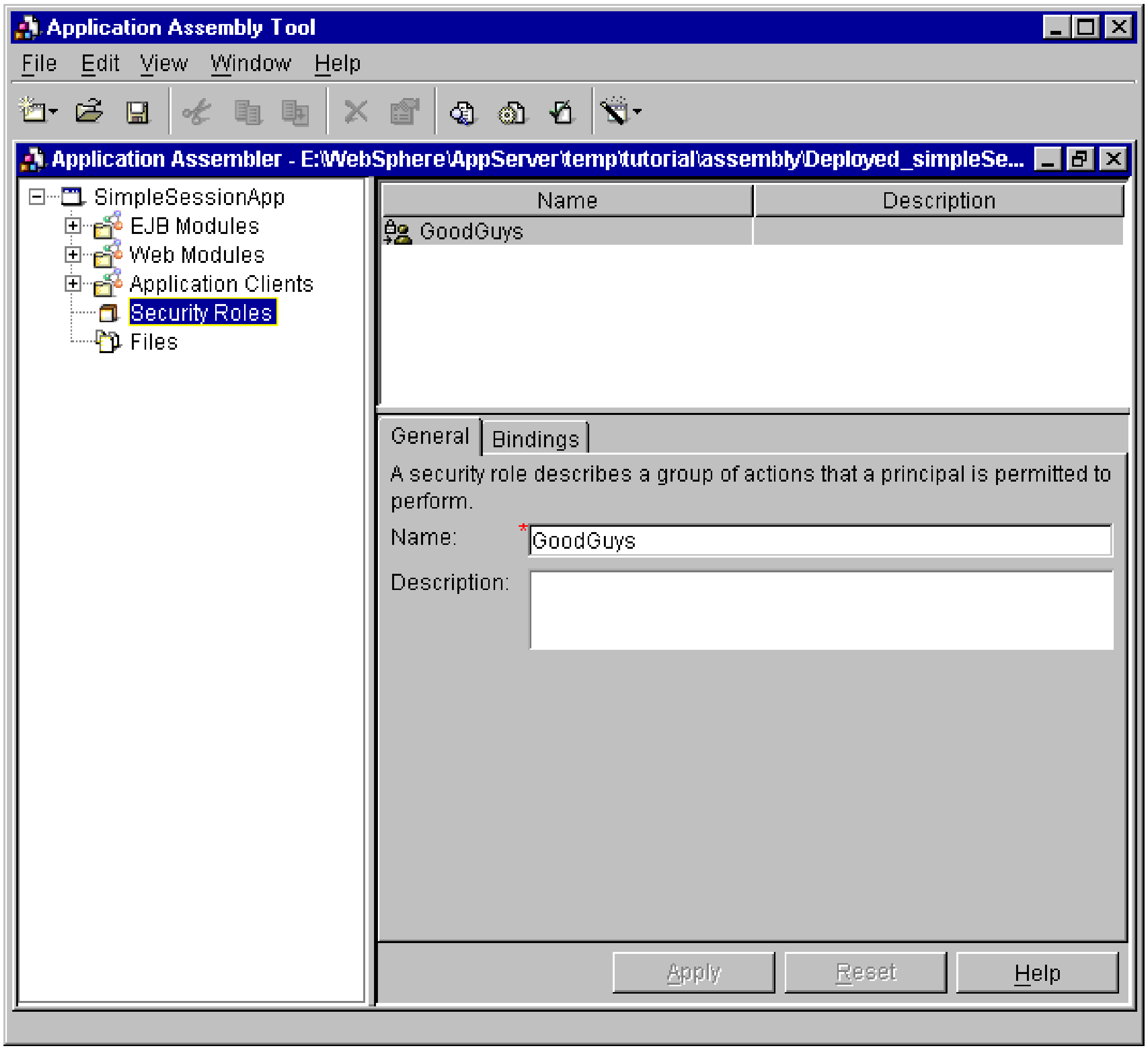
   

   File name: Deployed_simpleSession.ear

   Files of type: J2EE archive (*.war,*.ear,*.jar)

   3. Navigate to the **Deployed_simpleSession.ear** that was assembled in the Simple Session Bean tutorial and select **Select.**
   4. Select **OK.**
3. Create a security role to which you will be granting authorization to the session bean and servlet in the application:

a. In the .ear file, select the **Security Roles** folder.


Application Assembly Tool
File Edit View Window Help
Application Assembler - E:\WebSphere\AppServer\temp\tutorial\assembly\Deployed_simpleSe...
SimpleSessionApp
EJB Modules
Web Modules
Application Clients
Security Roles
Files
General
Bindings
Apply
Reset
Help

Name	Description
GoodGuys	

A security role describes a group of actions that a principal is permitted to perform.

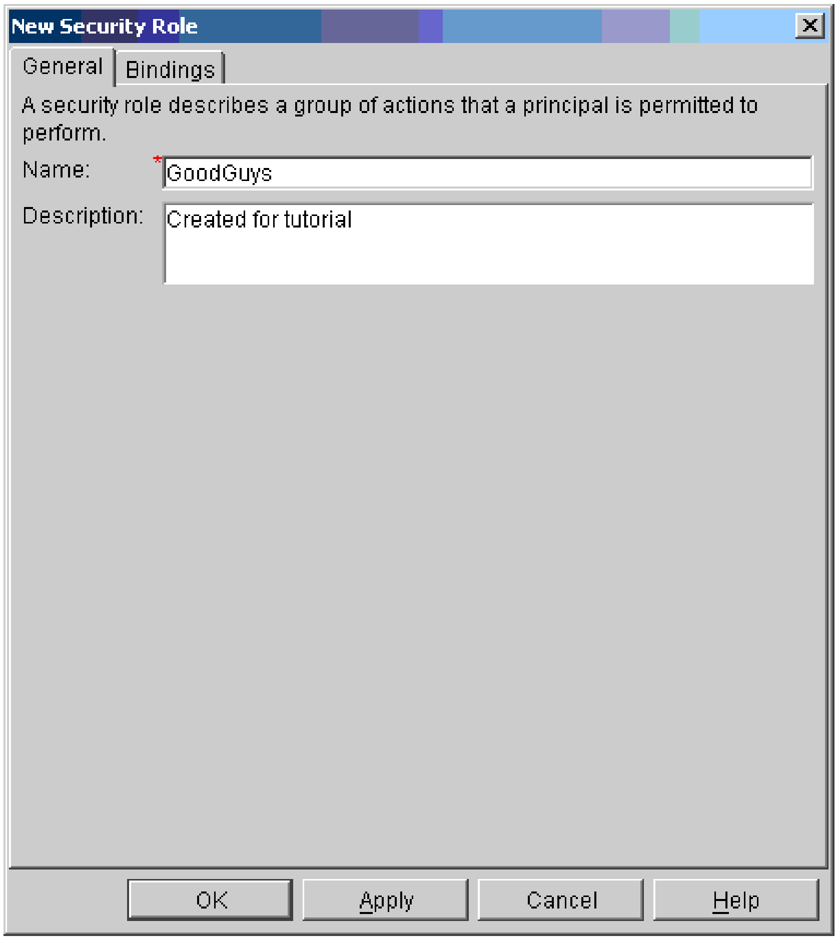
Name: * GoodGuys

Description:

b. Right-click it and select **New**.

c. On the **General** tab, enter **GoodGuys** for the **Name**.

d. Optionally, enter a **Description**.



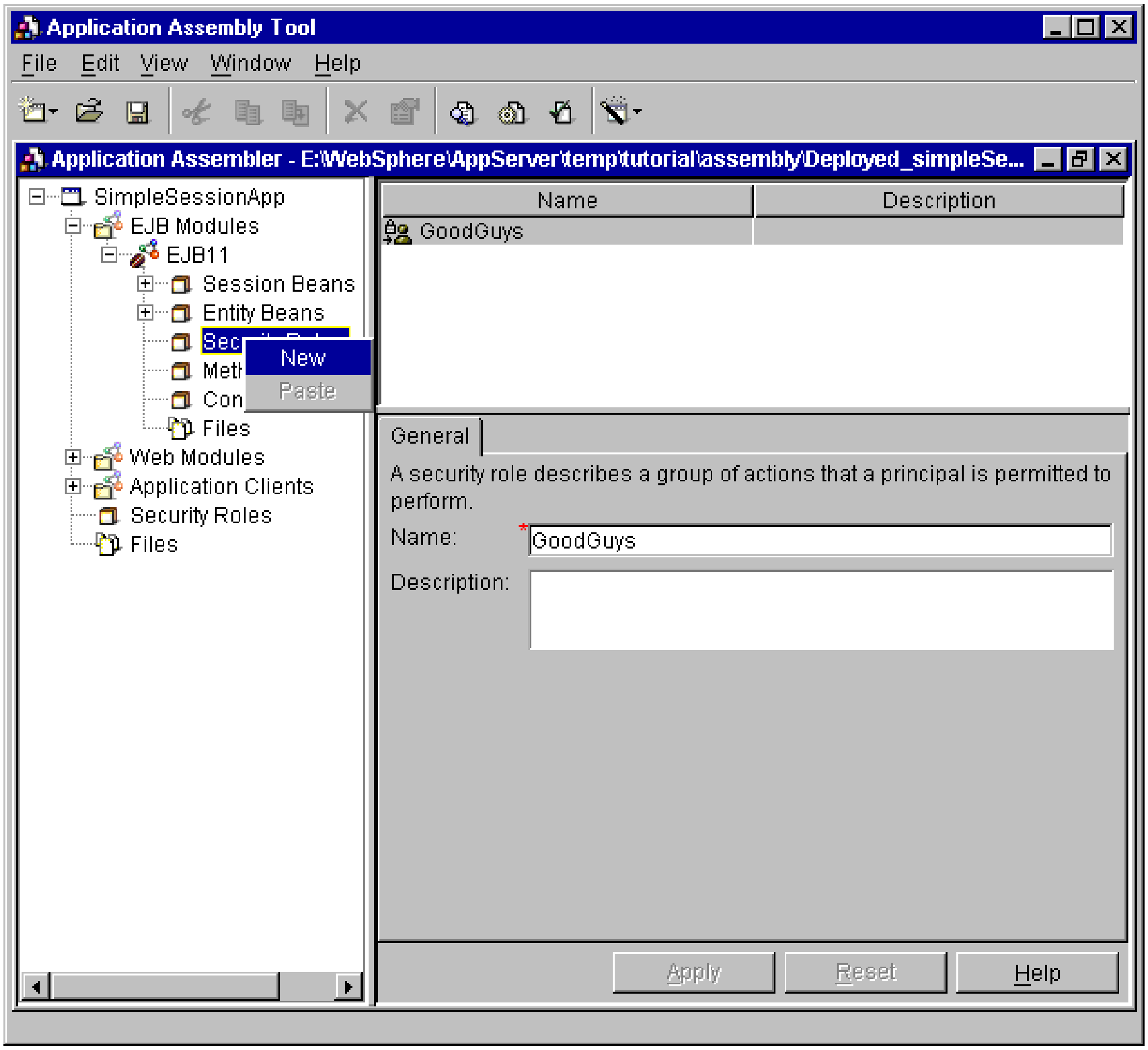
A security role describes a group of actions that a principal is permitted to perform.

Name: GoodGuys

Description: Created for tutorial

  e. Select the **Bindings** tab.
  f. Associate your (native) user ID with this security role:
    1. Click **Add...** (next to the **Users:** area).
    2. Specify your user ID in the **Name** field.
    3. Click **OK**.
  g. Click **OK.**

4. Authorize methods on the enterprise bean. You will authorize all home and remote methods to security role GoodGuys:
  a. In the tree view, expand **EJB Modules.**
  b. Expand the **EJB11**.
  c. Create a new security role:
    1. Select **Security Roles**.
    2. Right-click it and select **New**.

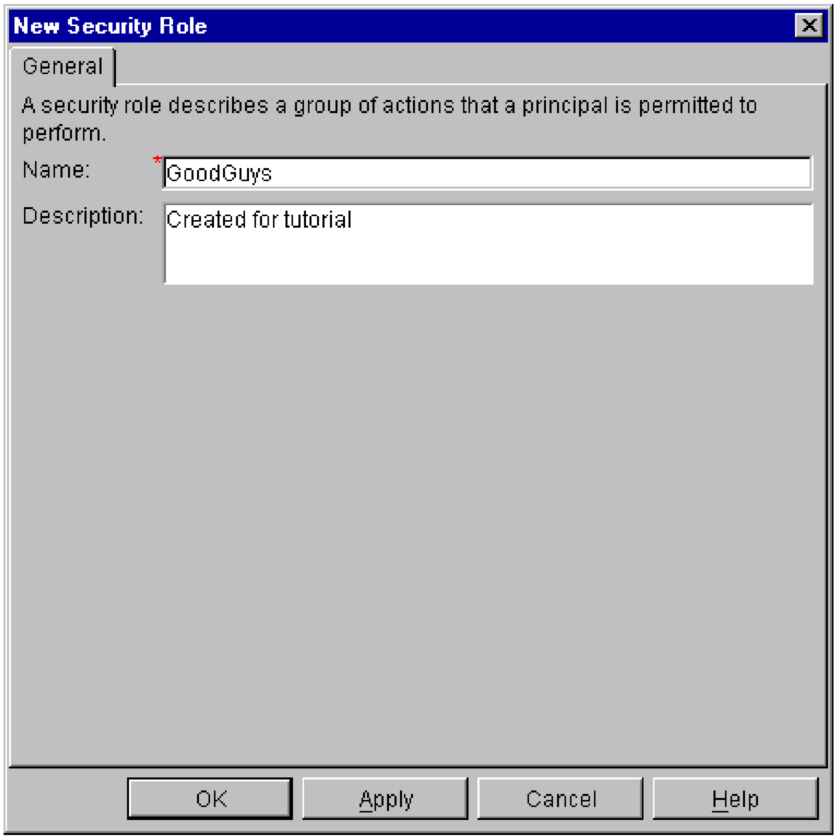


A security role describes a group of actions that a principal is permitted to perform.

Name: GoodGuys

Description:

3. Enter **GoodGuys** as the Name.
4. Optionally, enter a description.



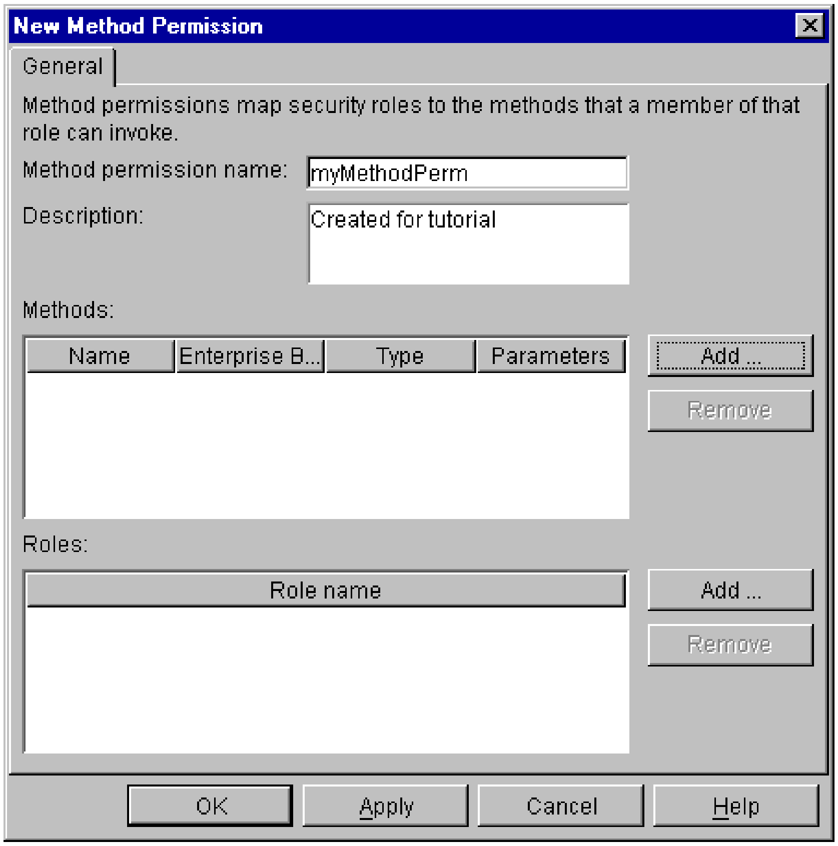
A security role describes a group of actions that a principal is permitted to perform.

Name: GoodGuys

Description: Created for tutorial

        5. Click **OK.**

5. Create a new method permission:
    a. In the tree view, select the **Method Permissions** under EJB11.
    b. Right-click it and select **New**.
    c. Add the methods:
        1. For the **Name**, specify myMethodPerm.
        2. In the **Methods** area, click **Add...**



Method permissions map security roles to the methods that a member of that role can invoke.

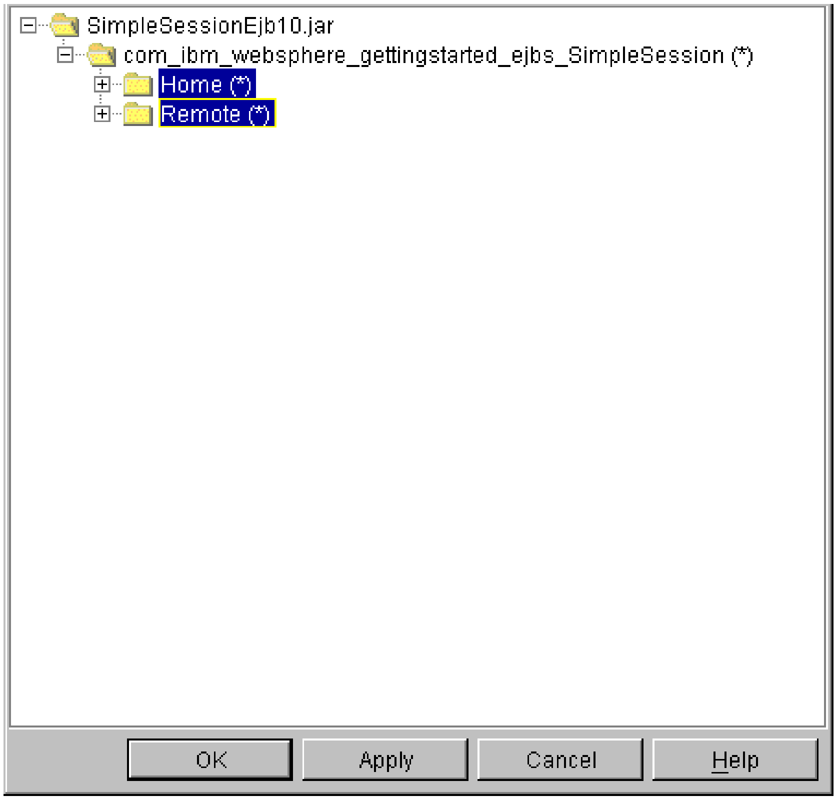
Method permission name: myMethodPerm

Description: Created for tutorial

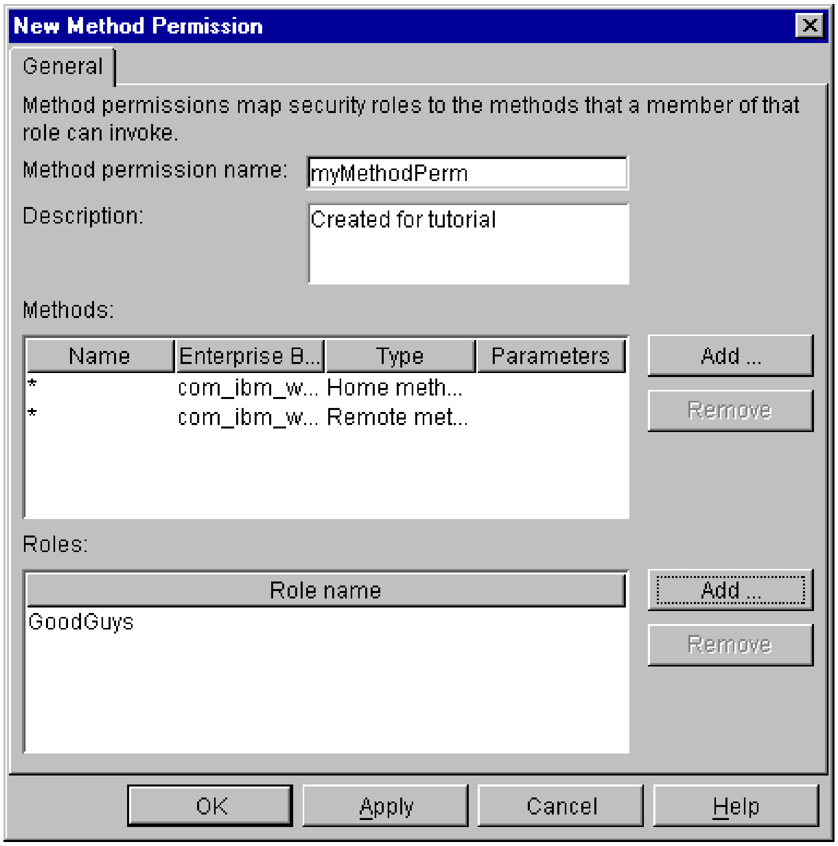
        3. In the **Add Methods** dialog, expand the Ejb11.jar and the beans path:

`com_ibm_websphere_gettingstarted_ejbs_SimpleSession`

4. Select the **Home** and **Remote** interface folders (hold down the Shift key to select both at once).



5. Click **OK**.
6. In the **Roles** area, click **Add...**
7. Select security role **GoodGuys**.
8. Click **OK.**



Method permissions map security roles to the methods that a member of that role can invoke.

Method permission name: myMethodPerm

Description: Created for tutorial

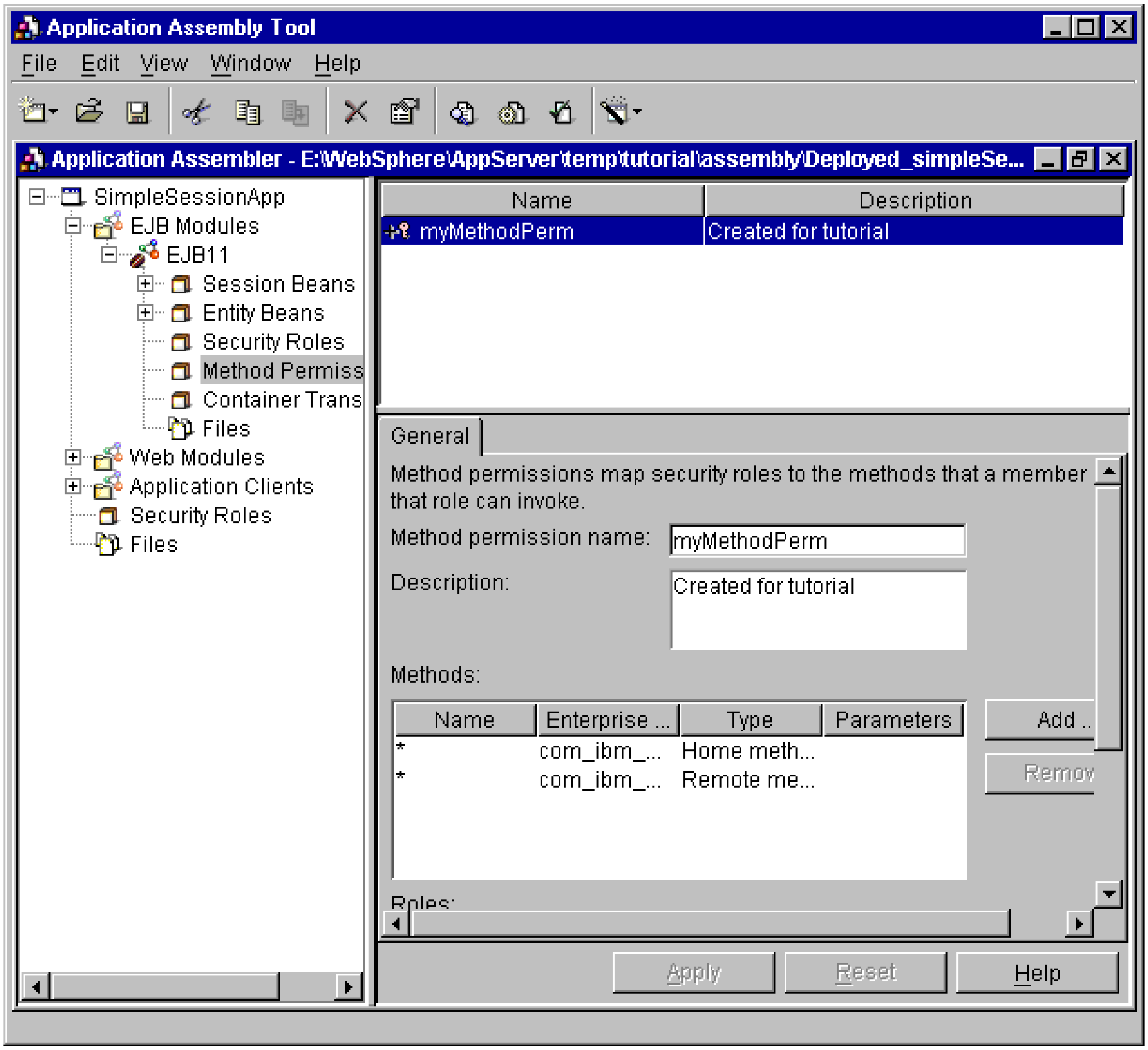
Methods:

Name	Enterprise B...	Type	Parameters
*	com_ibm_w...	Home meth...	
*	com_ibm_w...	Remote met...	

Roles:

Role name
GoodGuys

d. Click **OK** to close the New Method Permissions window.



Name	Description
myMethodPerm	Created for tutorial

Method permissions map security roles to the methods that a member that role can invoke.

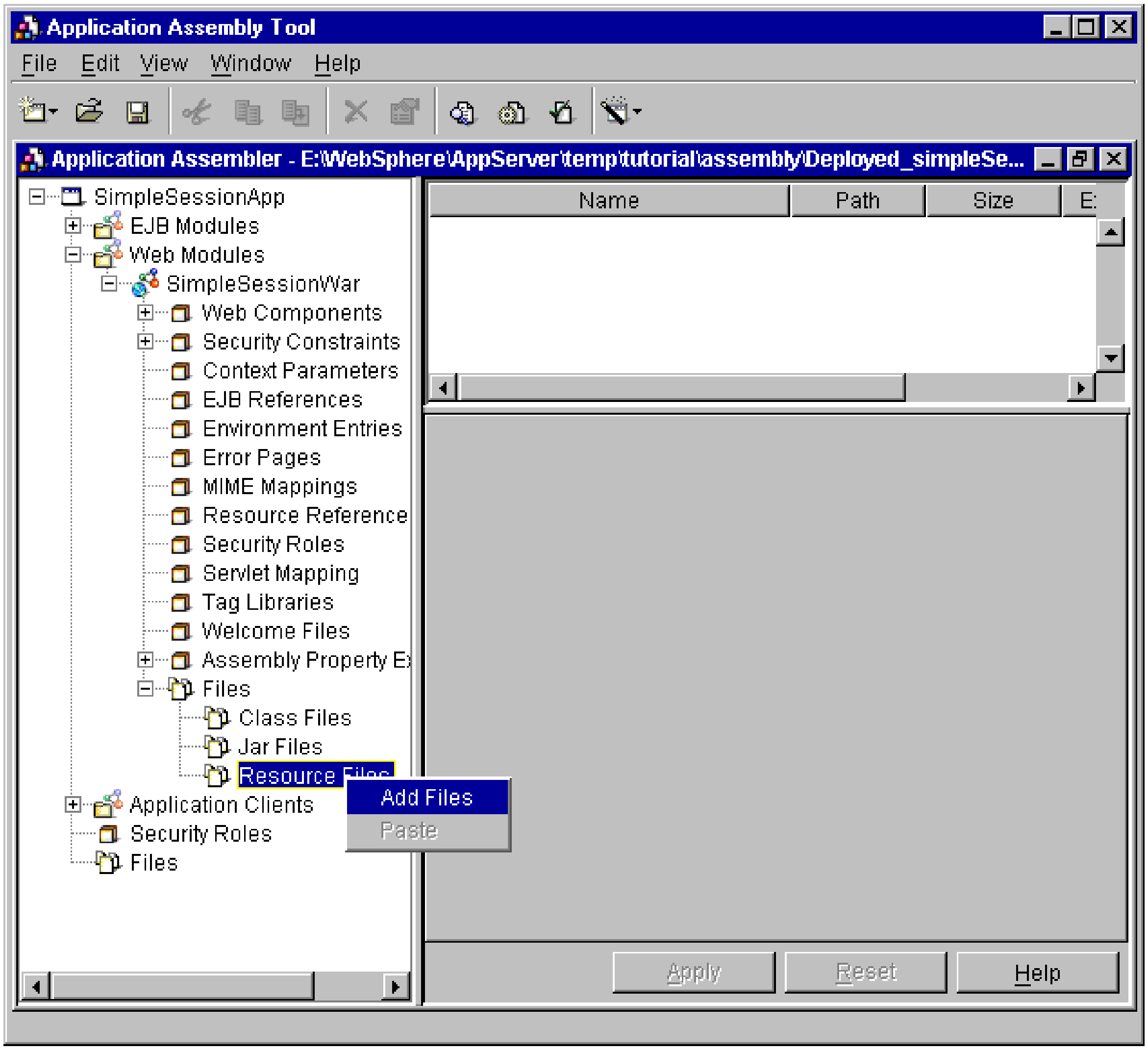
Method permission name: myMethodPerm

Description: Created for tutorial

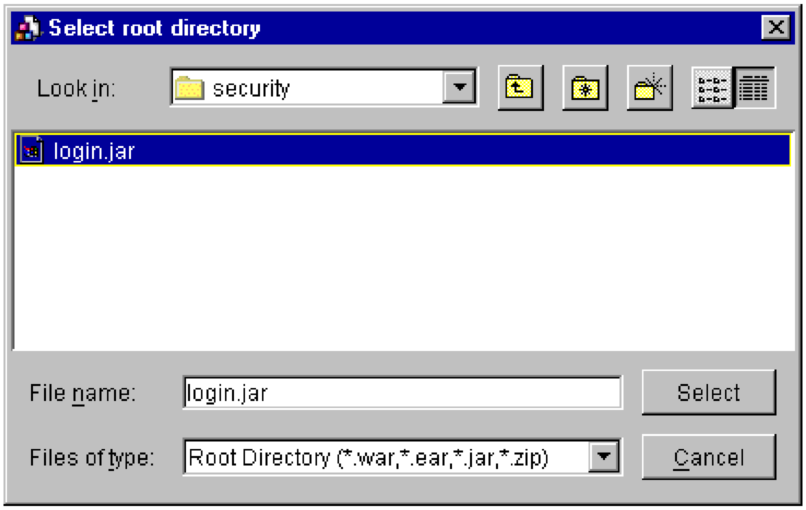
Methods:

Name	Enterprise ...	Type	Parameters
*	com_ibm_...	Home meth...	
*	com_ibm_...	Remote me...	

6. Authorize a servlet. Add a login form to be used for authentication:
    a. In the tree view, expand **Web Modules -> SimpleSessionWar -> Files**
    b. Select **Resource Files**.
    c. Right-click it and select **Add Files**.



d. Click **Browse** and navigate to the **login.jar** that you previously downloaded.



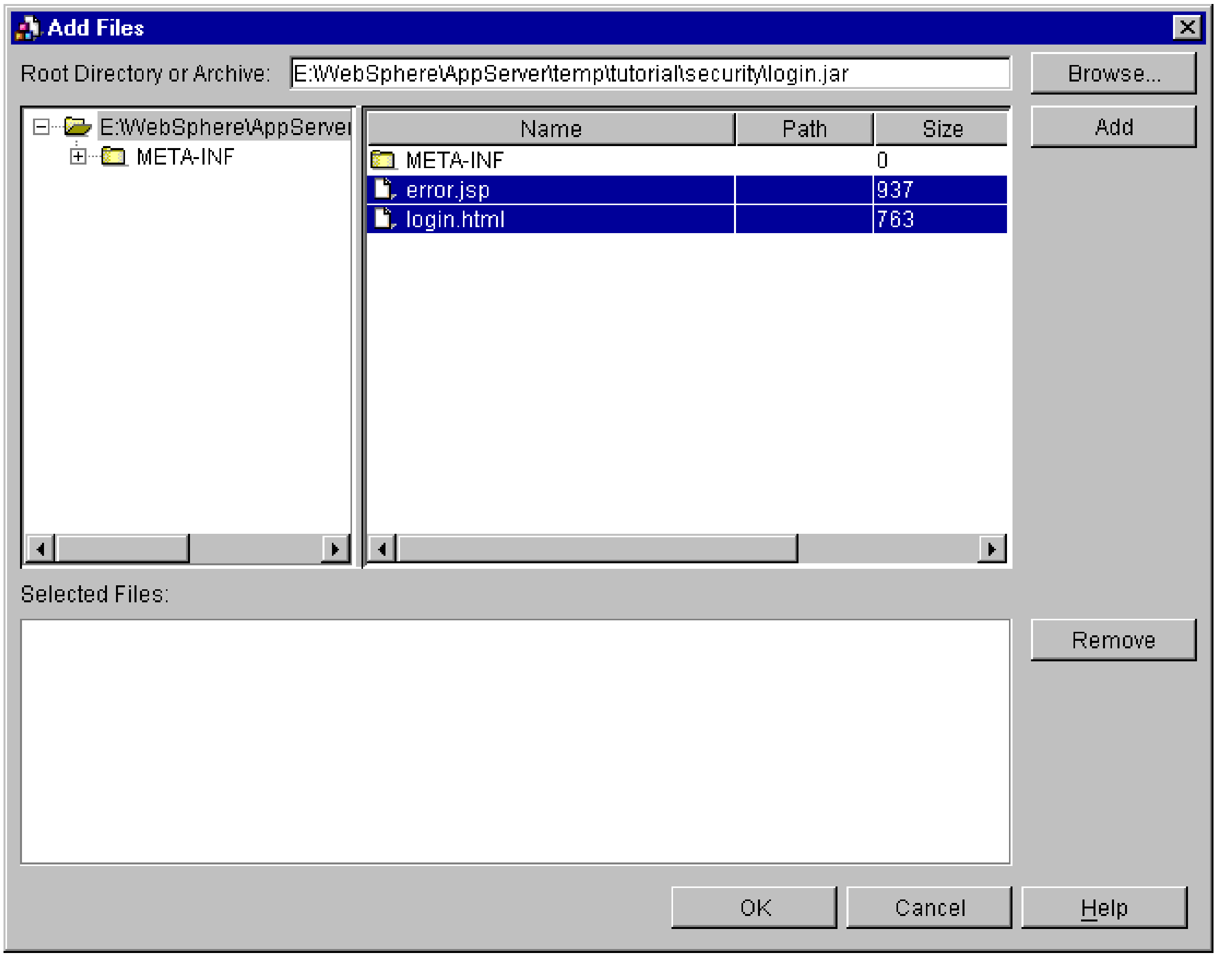
Look in: security

File name: login.jar

Files of type: Root Directory (*.war,*.ear,*.jar,*.zip)

e. Click **OK.**

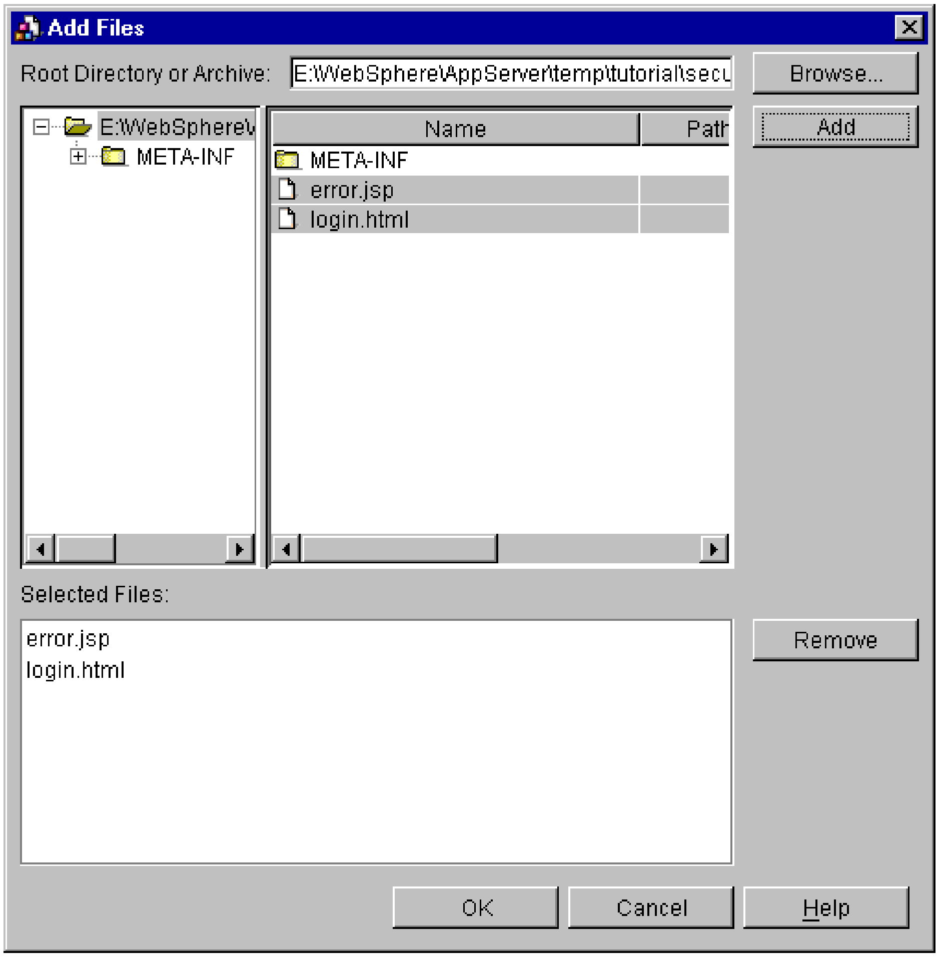
f. Select **login.html** and **error.jsp** (hold down the Shift key to select both at once).



Root Directory or Archive: E:\WebSphere\AppServer\temp\tutorial\security\login.jar

Name	Path	Size
META-INF		0
error.jsp		937
login.html		763

g. Click **Add**.



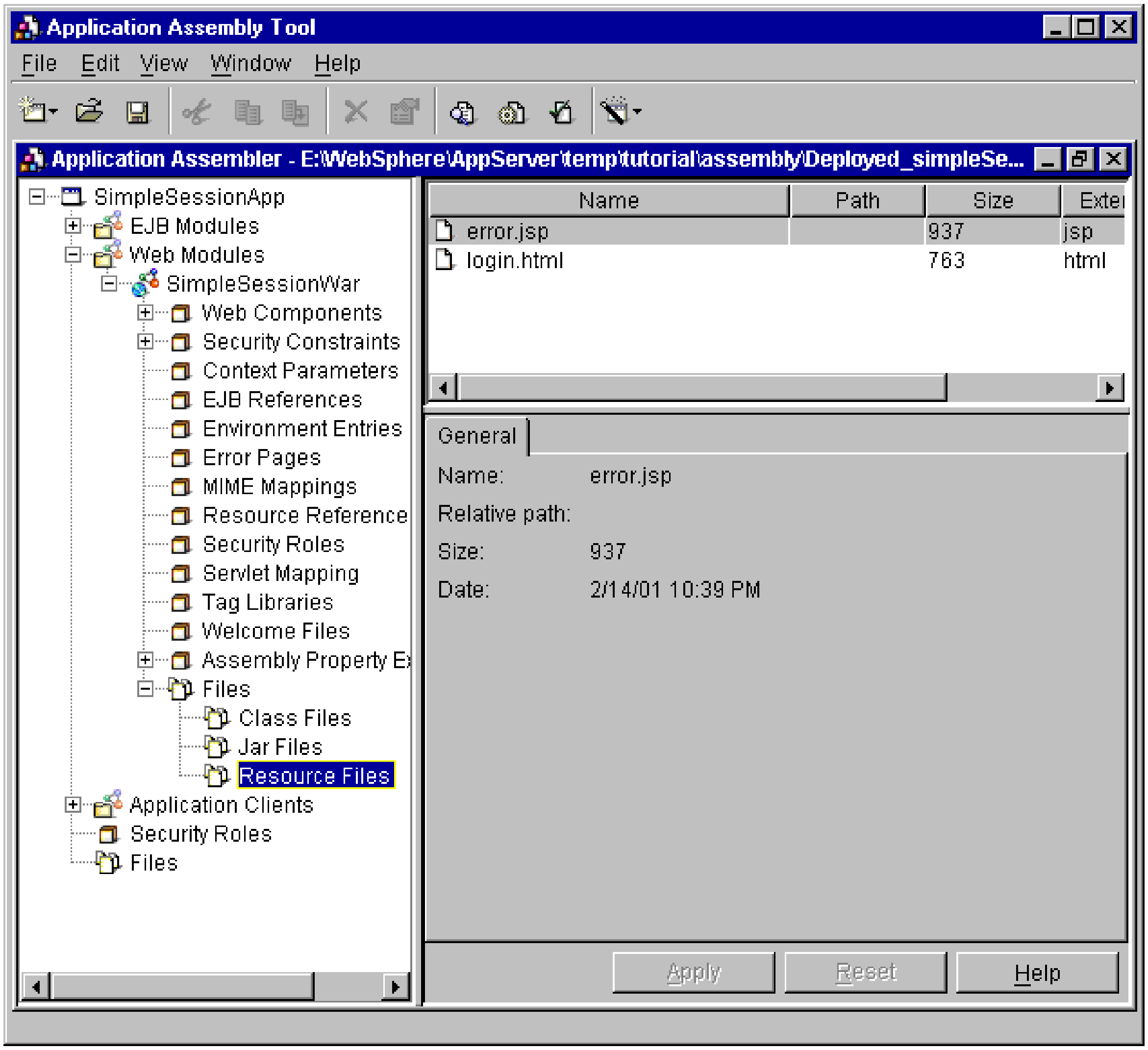
Root Directory or Archive: E:\WebSphere\AppServer\temp\tutorial\secu

Selected Files:

error.jsp
login.html

h. Click **OK.**

The files are displayed in the Application Assembly Tool, in the **Resource Files** folder of the Web module.



Name	Path	Size	Exte
error.jsp		937	jsp
login.html		763	html

Name: error.jsp

Relative path:

Size: 937

Date: 2/14/01 10:39 PM

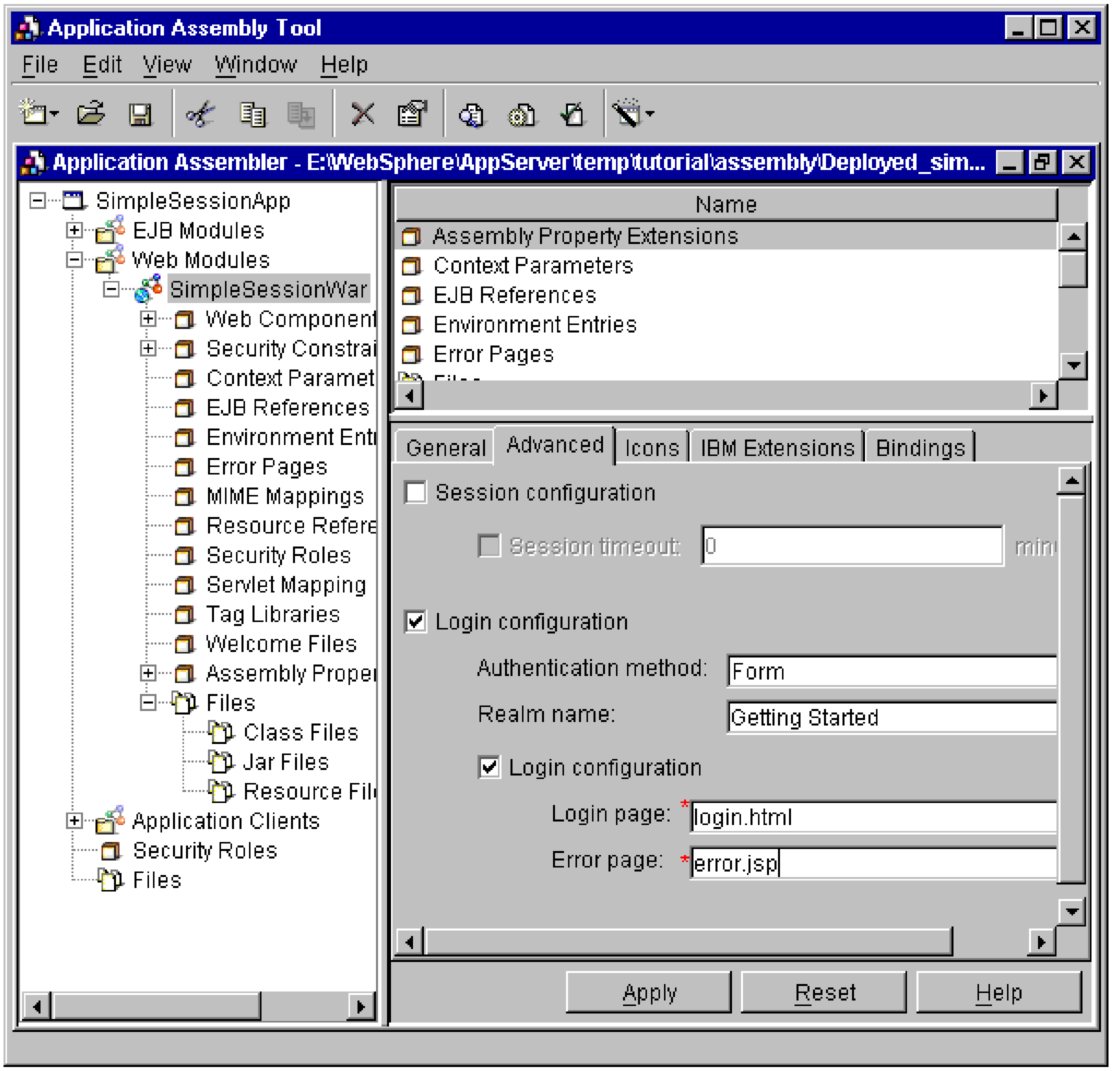
i. In the tree view, click the .war file named **SimpleSessionWar** to display its properties.

j. Click the **Advanced Tab.**

k. Check the **Login Configuration box**.

l. Do the following:

1. Set Authentication Method to **FORM**
2. Enter Realm name : **Getting Started**
3. Enter Login Page: **login.html**
4. Enter Error Page: **error.jsp**
5. Click **Apply** when finished.



☐ Session configuration

☐ Session timeout: 0

☑ Login configuration

Authentication method: Form

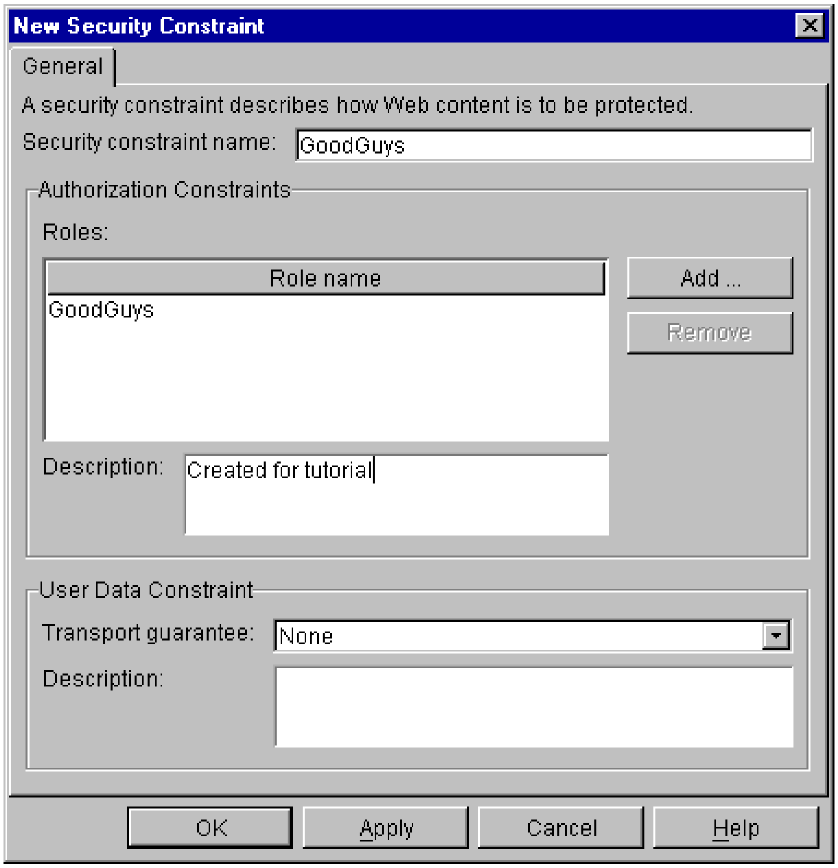
Realm name: Getting Started

☑ Login configuration

Login page: * login.html

Error page: * error.jsp

7. Add a security constraint:
    a. Expand **SimpleSessionWar**.
    b. Select **Security Constraints.**
    c. Right-click it.
    d. Click **New.**
    e. Add a new security constraint:
        1. Enter the Security Constraint Name: **GoodGuys**
        2. Click **Add...** next to the Roles area.
        3. Select the Security Role of **GoodGuys**.
        4. Select **OK.**
        5. Set **Transport Guarantee** to **NONE**.



A security constraint describes how Web content is to be protected.

Security constraint name: GoodGuys

Role name: GoodGuys

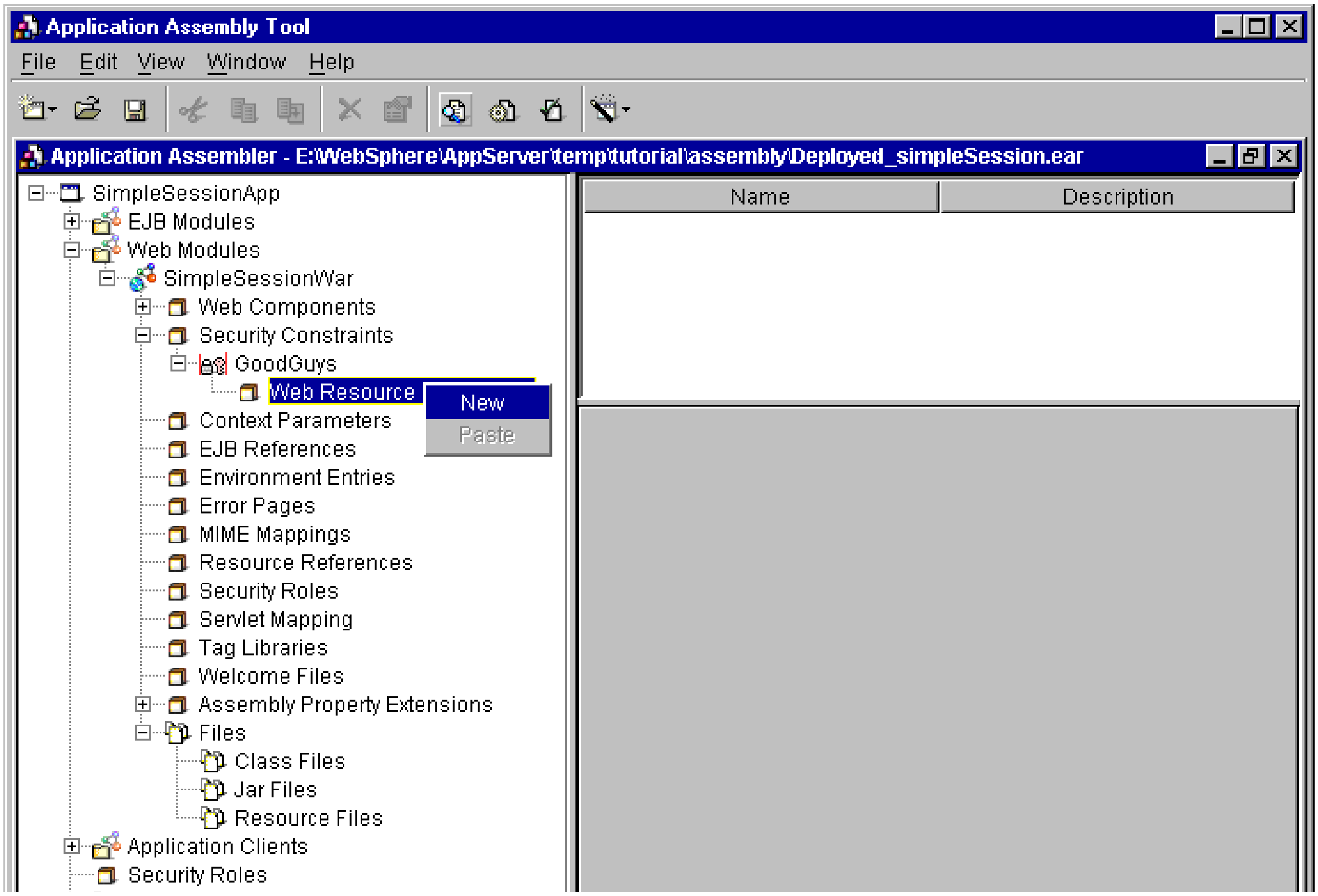
Description: Created for tutorial

Transport guarantee: None

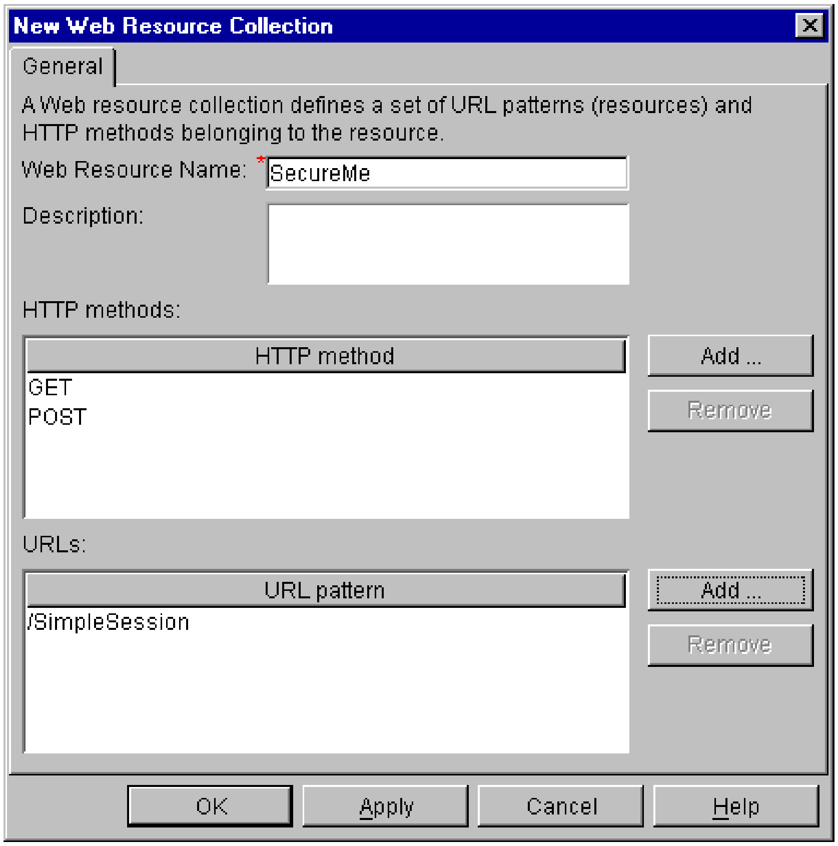
Description:

6. Click **OK**.

10. Add a resource collection to the constraint:
    a. Expand **SimpleSessionWar** -> **Security Constraints** -> **GoodGuys**
    b. Select **Web Resource Collections**.
    c. Right-click it and select **New.**



d. Add a new web resource collection:
   1. Enter the Web Resource Name: **SecureMe**.
   2. Click **Add...** in the **HTTP Methods** section.
   3. Change HTTP Method to **POST** .
   4. Click **OK.**
   5. Click **Add...** in the **HTTP Method** section**.**
   6. Change HTTP Method to **GET**.
   7. Click **OK.**
   8. Click **Add...** in the **URLs** section.
   9. Enter **/SimpleSession** for the **URL Pattern.**



A Web resource collection defines a set of URL patterns (resources) and HTTP methods belonging to the resource.

Web Resource Name: * SecureMe

Description:

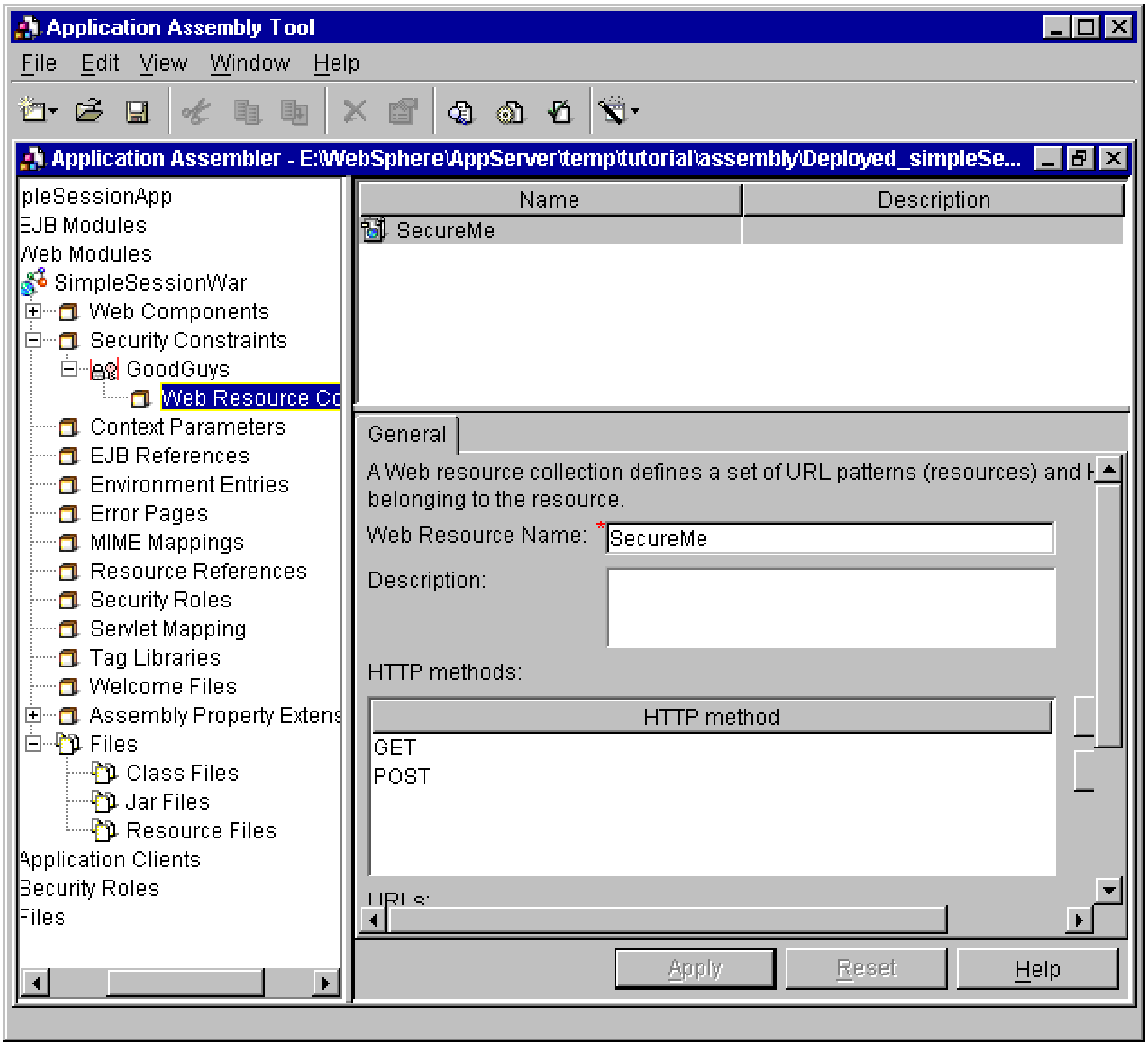
HTTP methods:

HTTP method
GET
POST

URLs:

URL pattern
/SimpleSession

   10. Click **OK.**



A Web resource collection defines a set of URL patterns (resources) and H belonging to the resource.

Web Resource Name: * SecureMe

Description:

HTTP methods:

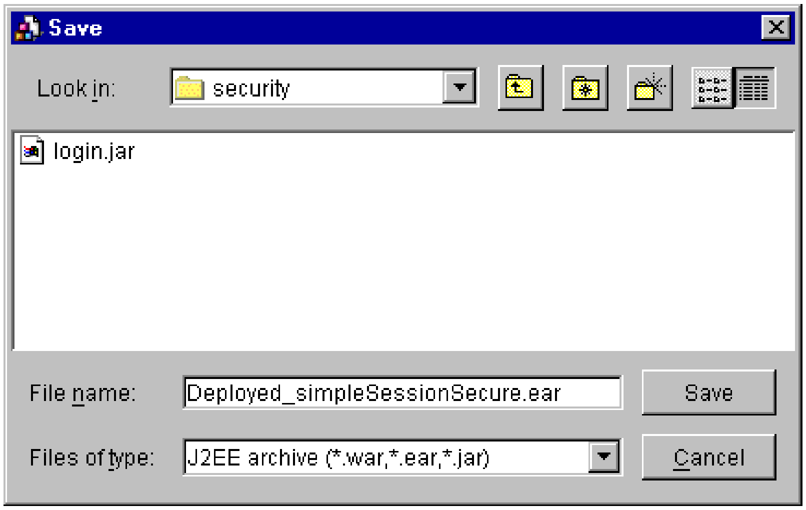
HTTP method
GET
POST

URLs:

e. Click **OK.**

11. Save your .ear file using **File** -> **Save As** and save as

product_installation_root`/temp/tutorial/security/Deployed_simpleSessionSecure.ear`



File name: Deployed_simpleSessionSecure.ear

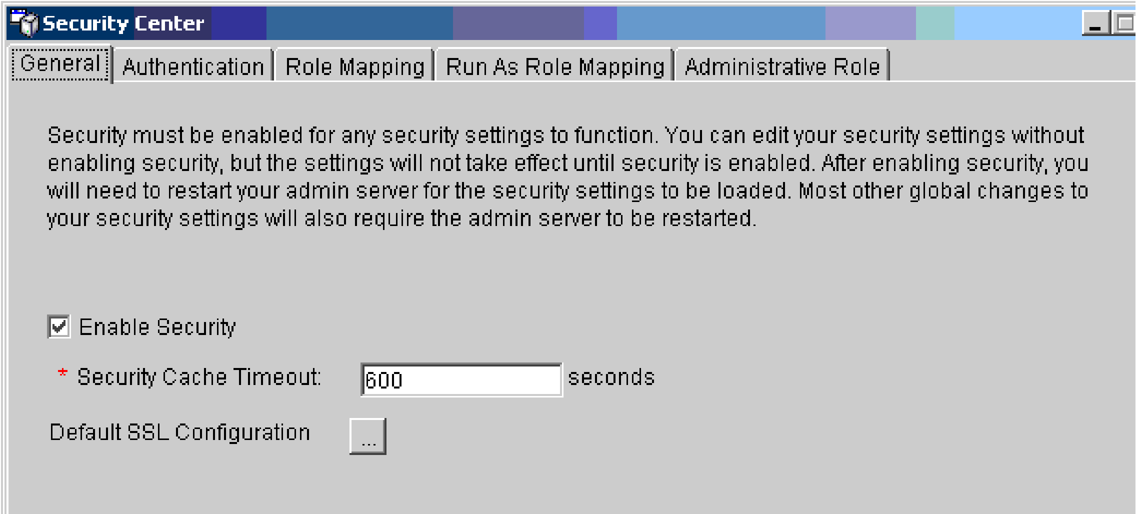
Files of type: J2EE archive (*.war,*.ear,*.jar)

12. Exit the AAT.

Enable security in the server runtime

1. Start the application server, as discussed in the Application deployment tutorial.
2. Open the administrative console
3. Enable security:
    1. Click **Console** -> **Security Center** on the console menu bar.

2. On the **General** tabbed page of the Security Center, click **Enable Security**.



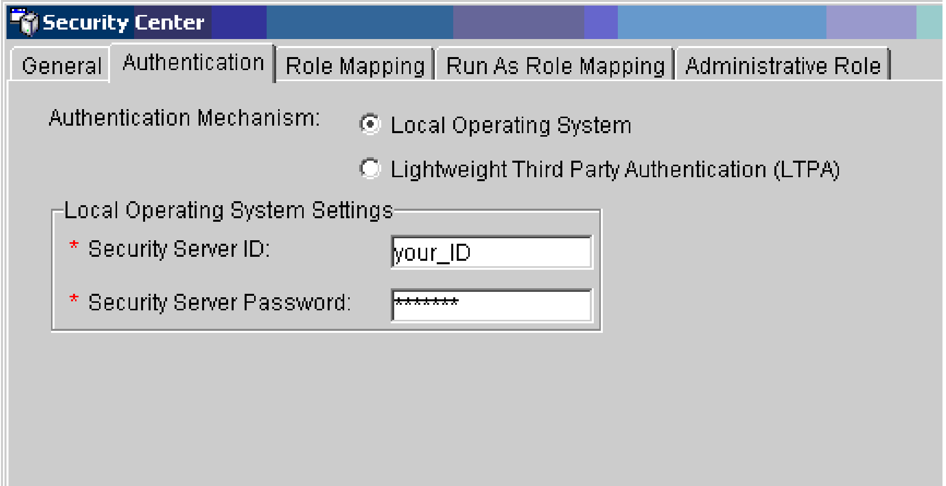
Security must be enabled for any security settings to function. You can edit your security settings without enabling security, but the settings will not take effect until security is enabled. After enabling security, you will need to restart your admin server for the security settings to be loaded. Most other global changes to your security settings will also require the admin server to be restarted.

☑ Enable Security

* Security Cache Timeout: 600 seconds

Default SSL Configuration ...

3. On the **Authentication** tabbed page, select **Local Operating System**.
4. Specify the **Security Server ID** and password, namely your native operating system



Authentication Mechanism: ● Local Operating System

○ Lightweight Third Party Authentication (LTPA)

Local Operating System Settings

* Security Server ID: your_ID

* Security Server Password: *******

5. Click **OK** to exit the Security Center.

Remove the unsecured version of the application

Remove the unsecured version of SimpleSessionApp as described in the Application removal and cleanup tutorial. Then return here to proceed with the security tutorial.

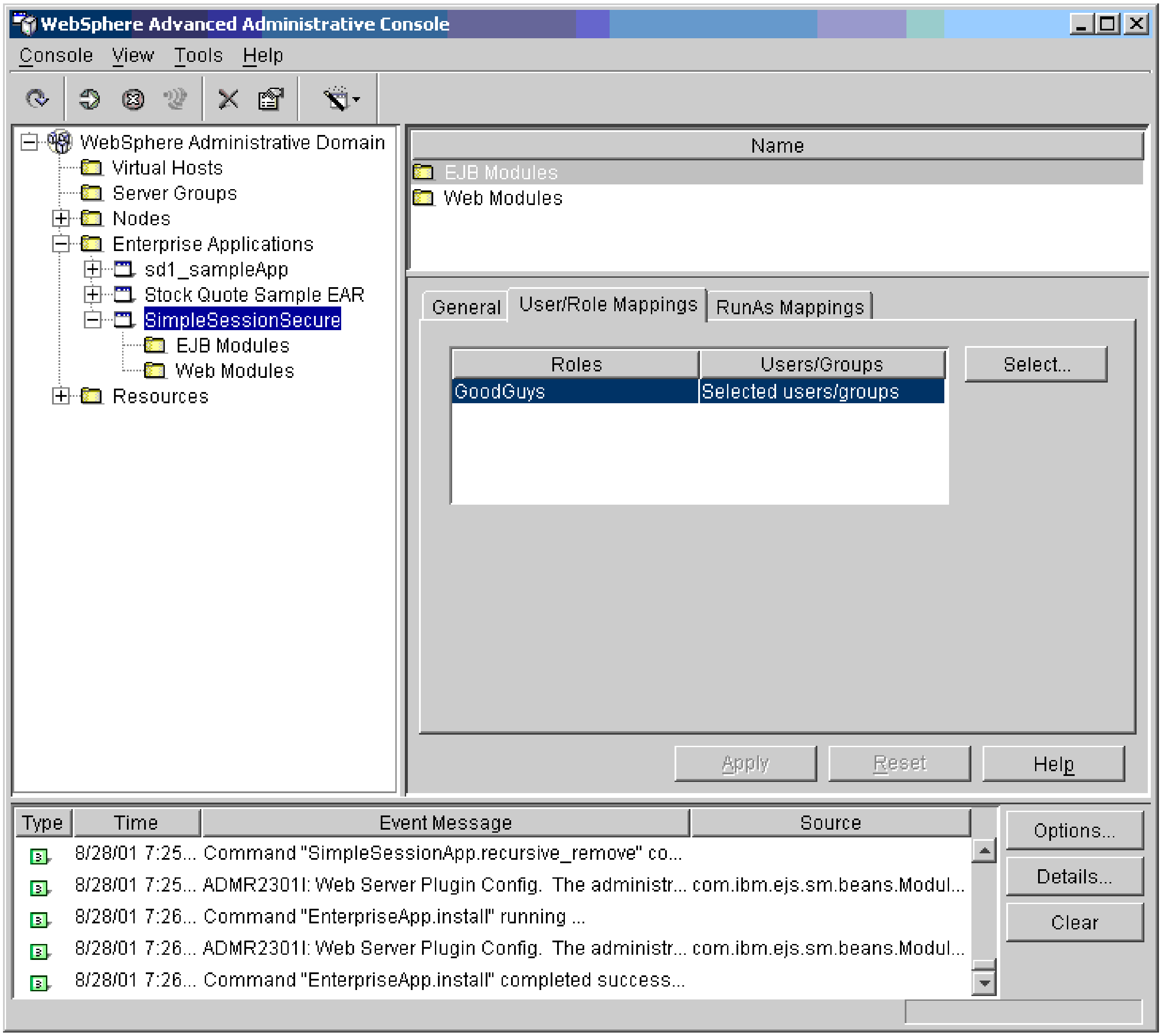
Install the secured application

1. In the console tree view, right-click **WebSphere Administrative Domain -> Enterprise Applications**.
2. From the resulting menu, click **Install Enterprise Application** to launch the **Install Enterprise Application** wizard.
3. On the **Specifying the Application or Module** panel:
    1. Ensure that the **Browse for file on node** field is set to your current node.
    2. Ensure **Install Application** is selected.
    3. Click **Browse**, next to the **Path** field. Browse for the Deployed_simpleSession.ear file and select it.

       Verify that its name is displayed in the **Path** field now.
    4. Specify SimpleSessionSecure as the **Application name**.
    5. Click **Next**.
4. Click **No** when prompted whether to deny access to unprotected methods.
5. On the **Mapping Users to Roles** panel, verify that the Goodguys role is mapped to your native user ID. Click **Select...** and make sure you can see your native ID listed in the **Selected Users/Groups** area of the resulting **Select Users/Groups** dialog. (To close the dialog after verification, click **OK**).
6. Click **Next**.
7. On the **Mapping EJB RunAs Roles to Users** panel, click **Next**.
8. On the **Binding Enterprise Beans to JNDI Names** panel, verify that the **JNDI Name** is set to gs/hello, and then click **Next**.
9. On the **Mapping EJB References to Enterprise Beans** panel, verify that the JNDI Name is set to gs/hello, and then click **Next**.
10. On the **Mapping Resource References to Resources** panel, click **Next**.
11. On the **Specifying the Default Datasources for EJB Modules** panel, click **Next**.

12. On the **Specifying Data Sources for Individual CMP Beans** panel, click **Next**.
13. On the **Selecting Virtual Hosts for Web Modules** panel, ensure that the **Virtual Host** is set to default_host, then click **Next**.
14. On the **Selecting Application Server** panel, ensure that the EJB11 and SimpleSessionWar modules are going to reside on **Application Server** "Default Server," then click **Next**.
15. Click **Finish** on the **Completing the Application Installation Wizard** panel.
16. When prompted whether to regenerate code, click **No**.
17. Look for the message confirming successful installation of the application. It might be a minute before it is displayed.

You can now view SimpleSessionSecure in the console tree view.



Roles	Users/Groups
GoodGuys	Selected users/groups

```
Type  Time           Event Message                                       Source
      8/28/01 7:25... Command "SimpleSessionApp.recursive_remove" co...
      8/28/01 7:25... ADMR2301I: Web Server Plugin Config.  The administr... com.ibm.ejs.sm.beans.Modul...
      8/28/01 7:26... Command "EnterpriseApp.install" running ...
      8/28/01 7:26... ADMR2301I: Web Server Plugin Config.  The administr... com.ibm.ejs.sm.beans.Modul...
      8/28/01 7:26... Command "EnterpriseApp.install" completed success...
```

Regenerating the plug-in configuration

1. In the console tree view, right-click **WebSphere Administrative Domain -> Nodes ->** ***your_host_name***.
2. From the resulting menu, select **Regen Webserver Plugin**.
3. In the area at the bottom of the console, look for the **Event Message** saying that the Plugin regeneration has been completed. It might take a moment before it is displayed.

Stopping the administrative server and start it again

1. To stop the administrative server, you can right-click *your_host_name* under **Nodes** in the administrative console and select **Restart** from the resulting menu. The console will close when you do so.
2. Open the administrative console again after the administrative server starts. This time, you will be asked to log in, because security is enabled.
3. In the console tree view, click **WebSphere Administrative Domain -> Nodes ->** ***your_host_name*** **-> Application Servers -> Default Server**.
4. Ensure that the **Module Visibility** setting of the **Default Server** is set to "Compatibility." Click **Apply** if you had to change it.

Ensure the application, application server, and Web server are running

Recall learning this step in the Application testing tutorial.

Test the Web client

1. Test the Web client the same way you tested the Web client for the unsecured simpleSession application, as discussed in the Application testing tutorial. That is, in a Web browser, type the URL:

   `http://`*`hostname`*`:9080/gettingstarted3/SimpleSession?msg=Hi`
2. When the login screen is displayed, enter your native operating system user ID and password.

   Notice that your login credentials flowed to the EJB container as well because the above procedure established authorization of the EJB methods, as well as of the servlet.

Test the Java client

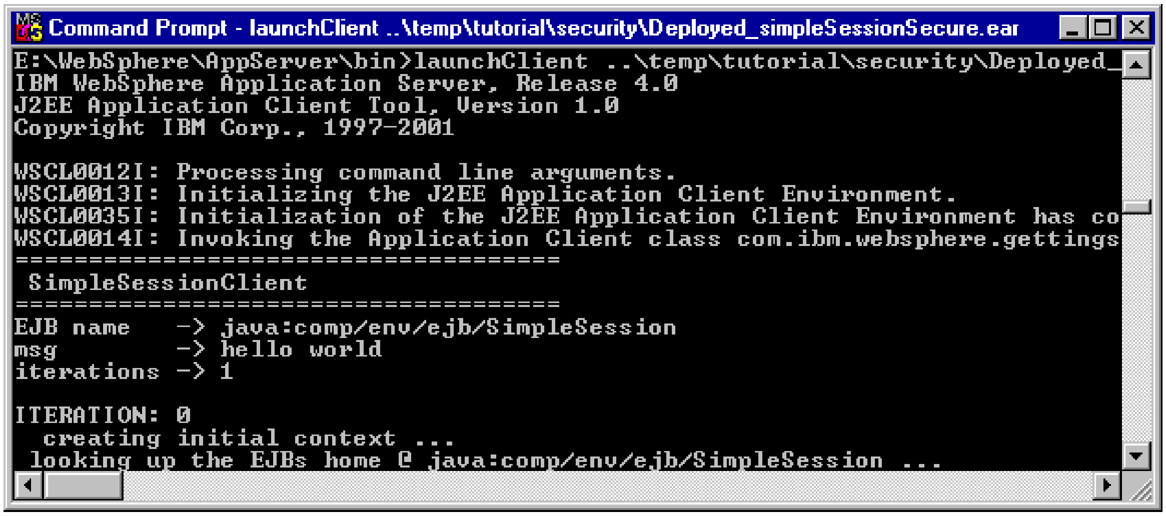
This, too, is similar to the way you did it in the Application testing tutorial, except you will reference the secured .ear file in the LaunchClient command.

1. Open a system command prompt.
2. Change directory to:

   `product_installation_root``/bin`
3. Enter the command:

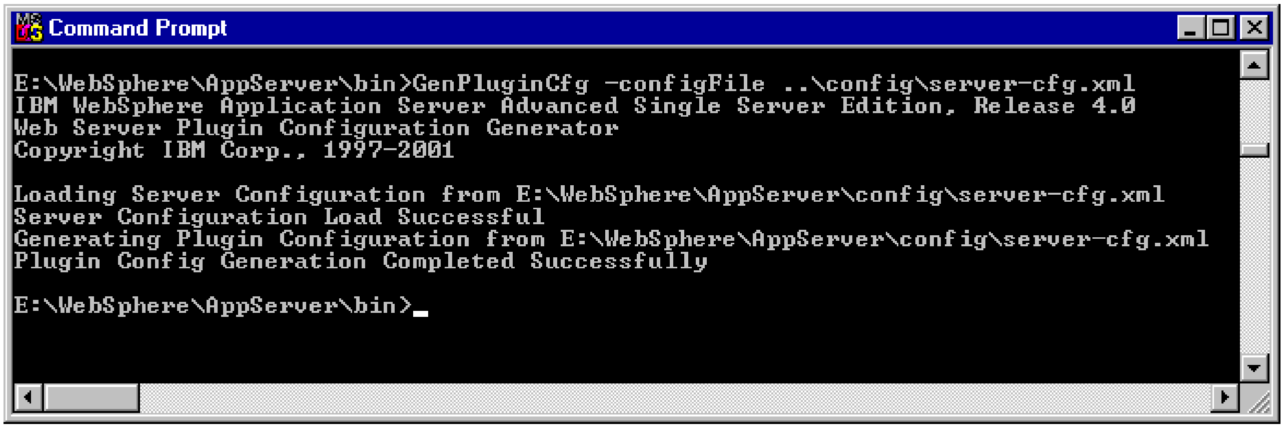
   ```
   launchClient ../temp/tutorial/security/Deployed_simpleSessionSecure.ear
   ```
   (Remember, on Windows systems, the forward slashes should be backslashes).

   The client will proceed to the point of looking the EJB home before it prompts the user to log in.


Command Prompt - launchClient ..\temp\tutorial\security\Deployed_simpleSessionSecure.ear

```
E:\WebSphere\AppServer\bin>launchClient ..\temp\tutorial\security\Deployed_
IBM WebSphere Application Server, Release 4.0
J2EE Application Client Tool, Version 1.0
Copyright IBM Corp., 1997-2001

WSCL0012I: Processing command line arguments.
WSCL0013I: Initializing the J2EE Application Client Environment.
WSCL0035I: Initialization of the J2EE Application Client Environment has co
WSCL0014I: Invoking the Application Client class com.ibm.websphere.gettings
=====================================
 SimpleSessionClient
=====================================
EJB name   -> java:comp/env/ejb/SimpleSession
msg        -> hello world
iterations -> 1

ITERATION: 0
  creating initial context ...
 looking up the EJBs home @ java:comp/env/ejb/SimpleSession ...
```

4. When the login pop-up is displayed, enter your native operating system user ID and password.



```

E:\WebSphere\AppServer\bin>GenPluginCfg -configFile ..\config\server-cfg.xml
IBM WebSphere Application Server Advanced Single Server Edition, Release 4.0
Web Server Plugin Configuration Generator
Copyright IBM Corp., 1997-2001

Loading Server Configuration from E:\WebSphere\AppServer\config\server-cfg.xml
Server Configuration Load Successful
Generating Plugin Configuration from E:\WebSphere\AppServer\config\server-cfg.xml
Plugin Config Generation Completed Successfully

E:\WebSphere\AppServer\bin>_
```

Disable security in the application server runtime

1. Start the application server, as discussed previously.
2. Open the administrative console, as discussed in the Application deployment tutorial.
3. Disable security.

    1. In the console tree view, select **Security.**
    2. Deselect the **Enabled** check box.
    3. Click **OK.**
4. Save the server configuration, using the dialog displayed by clicking the prompt "Configuration needs to be saved" at the top of the Security page.
5. Stop the application server and start it again, as discussed previously.

Do not forget to go back into the console and disable security **if** you are sharing the console with a colleague who will not know the correct ID and password the next time he or she tries to open the console!

What's next?

Now that you have assembled, deployed, and secured a Session bean application, you might want to try your hand at a CMP application. The Advanced application assembly and deployment tutorial describes how to do so.

6.7.5: Advanced application assembly and deployment tutorial -- CMP bean application

During this tutorial, you will assemble and deploy a CMP bean application. This simple J2EE application consists of a single CMP entity bean and a Java application client. The client creates an EJB, invokes a remote method, then destroys it. We will start with an EJB 1.0 jar file.

Prerequisites

This tutorial assumes knowledge of the Application assembly tutorial and Application deployment tutorial, which focused on a Session bean application, but are also applicable for BMP bean applications. Indeed, many steps of this tutorial are abbreviated, with links to the steps in the assembly and deployment tutorials in case you need to refresh your memory.

This tutorial puts more attention on the new steps for CMP bean support, namely, creating database tables and configuring data sources in the application server runtime.

You will need a **working, supported DB2 or Oracle database** to verify your results using the tutorial steps as they are written. Of course, you can always use this tutorial to practice the steps, knowing that your data access is not set up yet.

Overview of steps (60 minutes)

1. Obtain the example code
2. Start the Application Assembly Tool
3. Assemble an EJB module, converting from EJB 1.0 JAR to EJB 1.1
4. Assemble an application client module
5. Assemble a J2EE application
6. Exit the Application Assembly Tool
7. Start the server
8. Open the administrative console
9. Configure a data source
10. Install the application
11. Regenerate the Web server plug-in configuration
12. Stop the application server and start it again
13. Create database table or tablespace
14. Ensure that the application, application server, and Web server are running
15. Test the Java application client

Paths through the tutorial

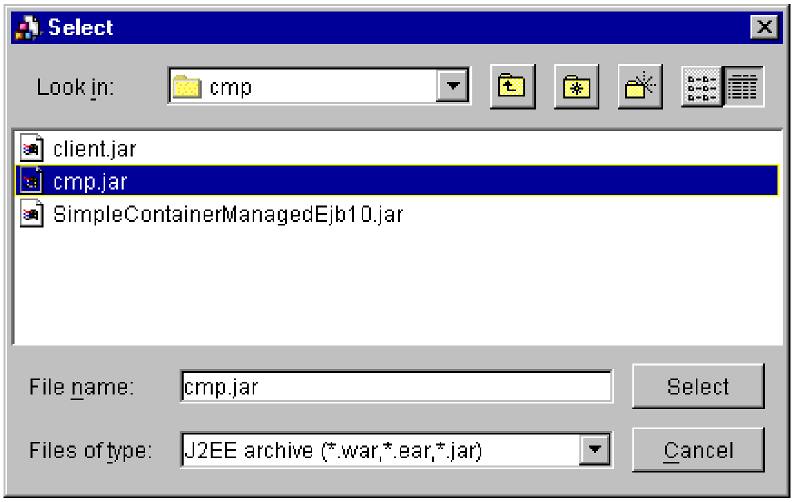
- **Quick path:** If your objective is to practice these steps as quickly as possible, skip the explanations marked with the ℹ graphic.
- **Maximum learning path:** If your objective is to understand and explore the product capabilities, read the ℹ notes and browse the links they provide to additional documentation.

Start the Application Assembly Tool

Start the Application Assembly Tool, as discussed in the Application assembly tutorial.

Assemble an EJB module, converting from EJB 1.0 JAR to EJB 1.1

1. On the **Welcome to Application Assembly Tool** panel:
    a. Click the **Existing** tab.
    b. Click **Browse** next to the File name field.
    c. Navigate to the **cmp.jar** file that you previously downloaded and select **Select.**


Select
Look in: cmp
client.jar
cmp.jar
SimpleContainerManagedEjb10.jar
File name: cmp.jar
Files of type: J2EE archive (*.war,*.ear,*.jar)
Select
Cancel

   d. Select **OK.**

2. Click **OK** at the "...please specify the dependent classpath..." dialog.

   This will convert the EJB 1.0 JAR file into an EJB 1.1 JAR file.

3. Set the global JNDI name of the bean, and bind the data source:
   a. In the topology tree, expand the **Entity Beans** folder.
   b. Select the bean labeled:

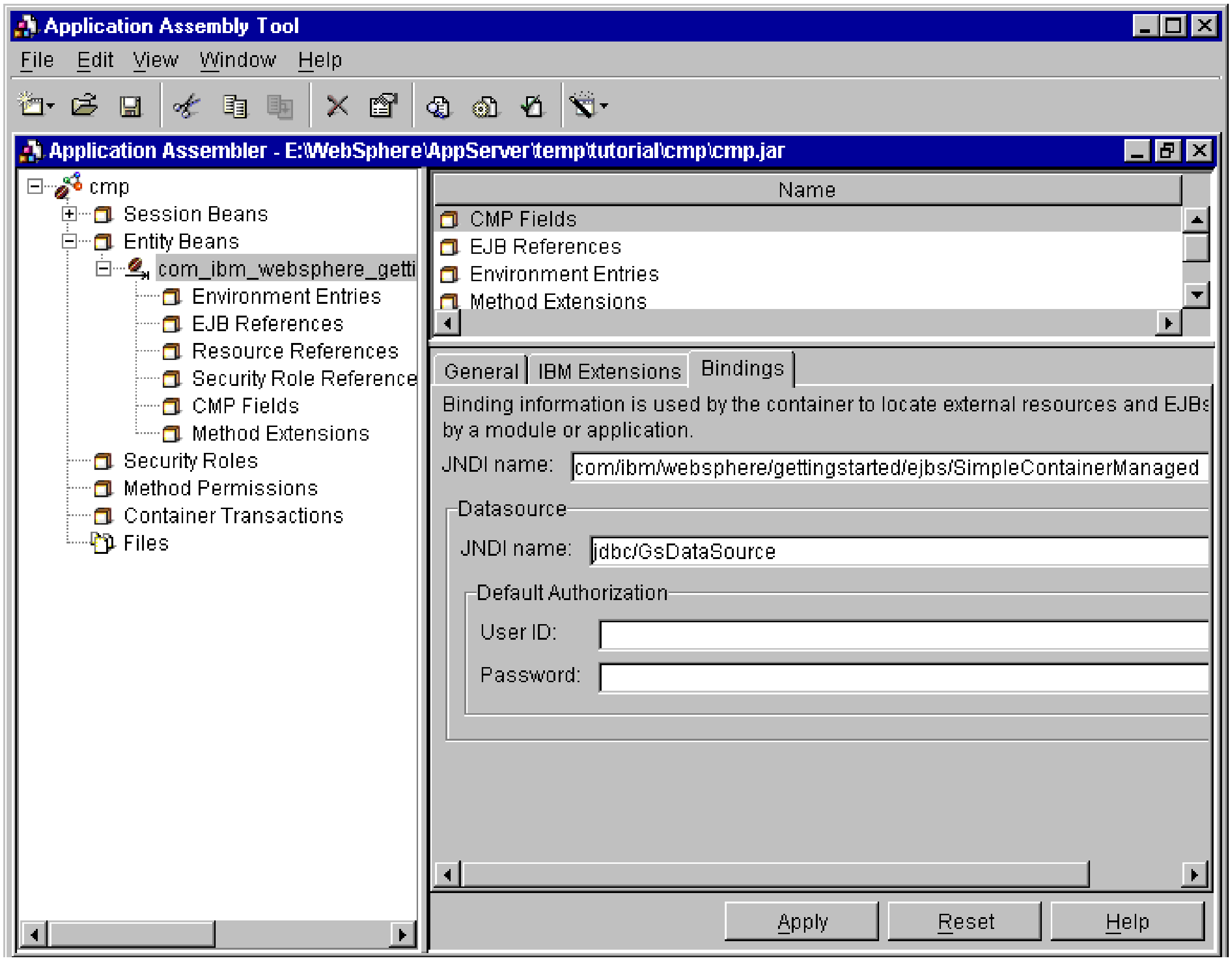
      com_ibm_websphere_gettingstarted_ejbs_SimpleContainerManaged

   c. Click it to display its properties.
   d. Select the **Bindings** tab.
      1. Set the **JNDI Name** to:

         com/ibm/websphere/gettingstarted/ejbs/SimpleContainerManagedHome

         You can use a shorter name as long as you guarantee that it is unique among all installed applications. Using the fully qualified package name of the bean will accomplish this, usually.

      2. Enter Datasource JNDI Name: **jdbc/GsDataSource**



Binding information is used by the container to locate external resources and EJBs by a module or application.

JNDI name: com/ibm/websphere/gettingstarted/ejbs/SimpleContainerManaged

Datasource

JNDI name: jdbc/GsDataSource

Default Authorization

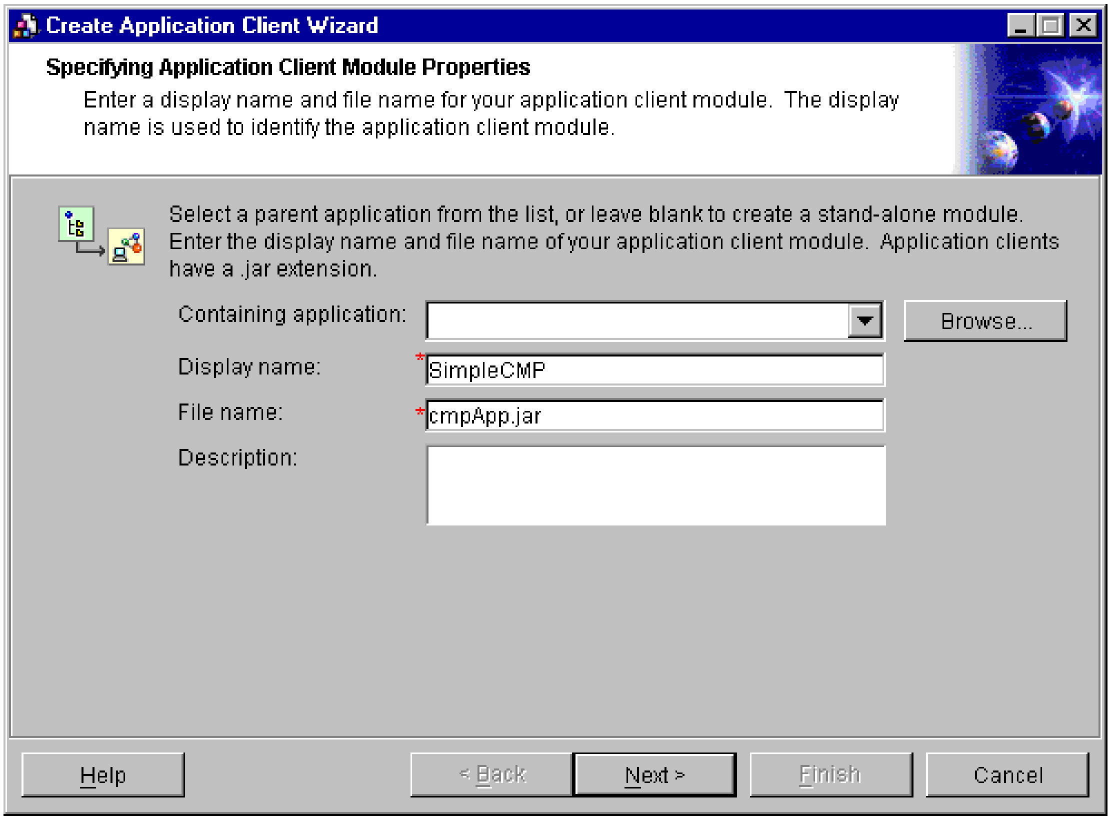
User ID:

Password:

   3. If using an Oracle database, enter the Oracle admin userid and password under **Default Authorizations** (this is not required for DB2).
   4. Click **Apply**.
5. Save the JAR file using **File** -> **Save As...** and name it **cmp11.jar**.

Assemble an application client module

1. Create a new application client using **File -> Wizards -> Create Application Client Wizard**.
2. On the **Specifying Application Client Module Properties** panel:
   a. Enter the following:
      - Display Name: **SimpleCMP**
      - File Name: **cmpApp.jar**



Create Application Client Wizard

Specifying Application Client Module Properties

Enter a display name and file name for your application client module. The display name is used to identify the application client module.

Select a parent application from the list, or leave blank to create a stand-alone module. Enter the display name and file name of your application client module. Application clients have a .jar extension.

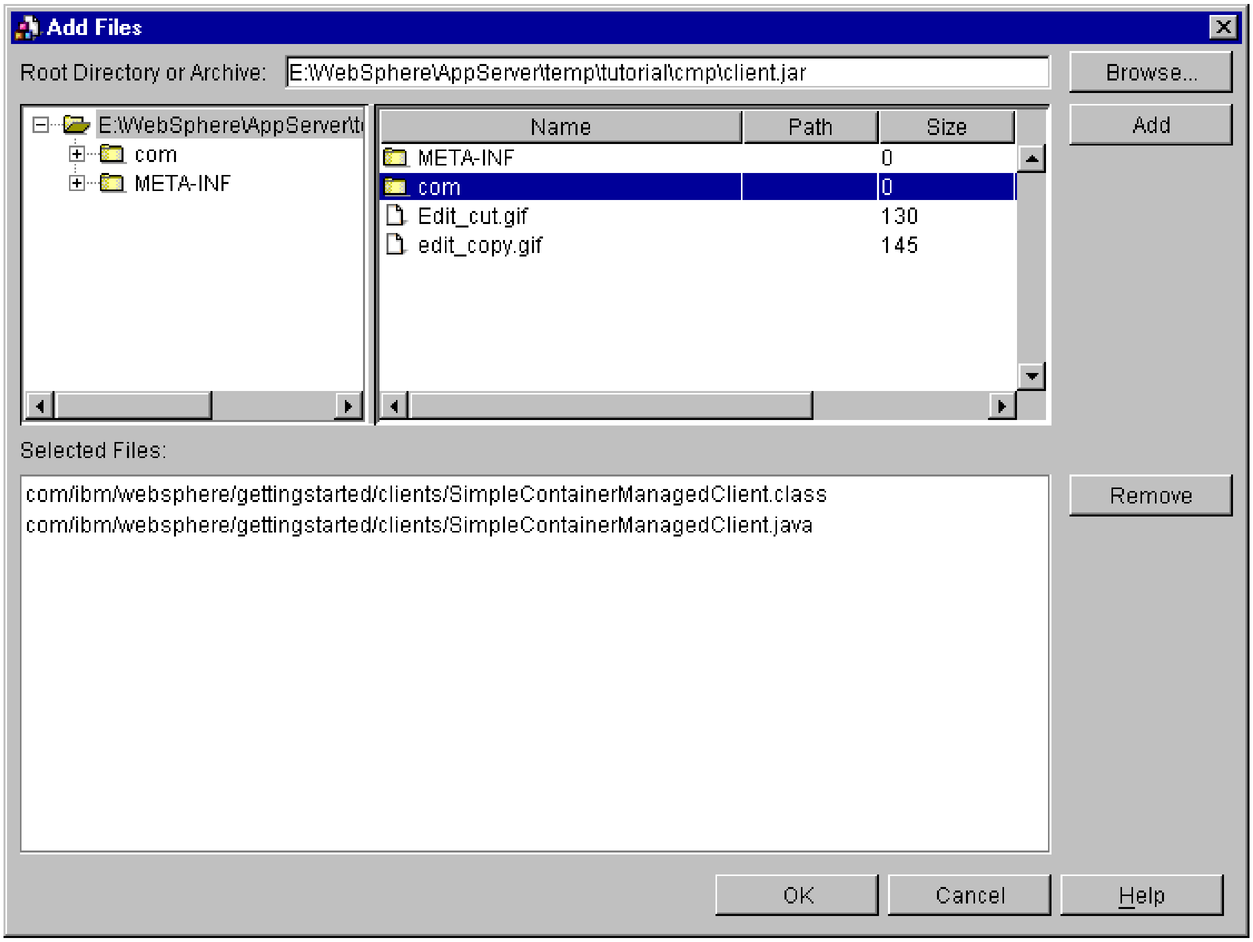
Containing application:

Display name: * SimpleCMP

File name: * cmpApp.jar

Description:

   b. Click **Next**.
3. Click **Add...** on the **Adding Files** panel.
   a. Click **Browse.**
   b. Navigate to the **client.jar** that was previously downloaded, highlight the file and click **Select.**
   c. Select the **com** folder (in the right hand pane).
   d. Click **Add**.



Root Directory or Archive: E:\WebSphere\AppServer\temp\tutorial\cmp\client.jar

E:\WebSphere\AppServer\t
- com
- META-INF

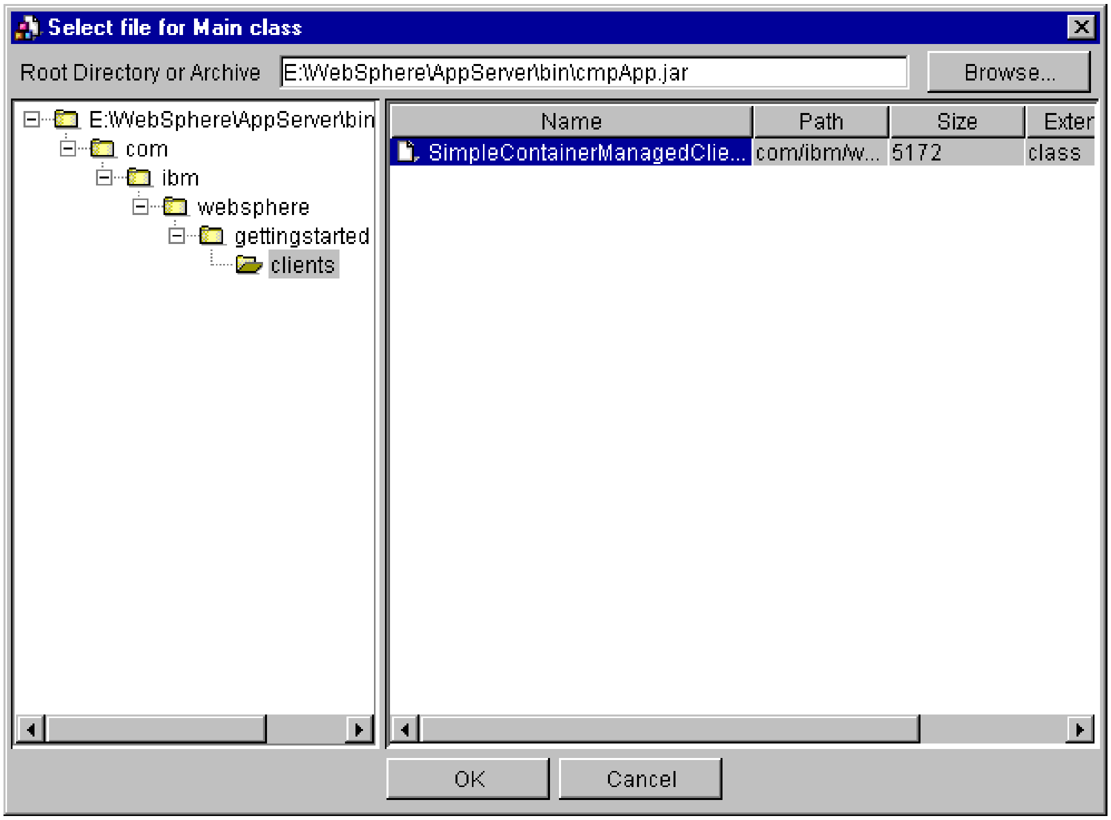
Name	Path	Size
META-INF		0
com		0
Edit_cut.gif		130
edit_copy.gif		145

Selected Files:

com/ibm/websphere/gettingstarted/clients/SimpleContainerManagedClient.class
com/ibm/websphere/gettingstarted/clients/SimpleContainerManagedClient.java

   e. Click **OK.**
   f. Click **Next.**

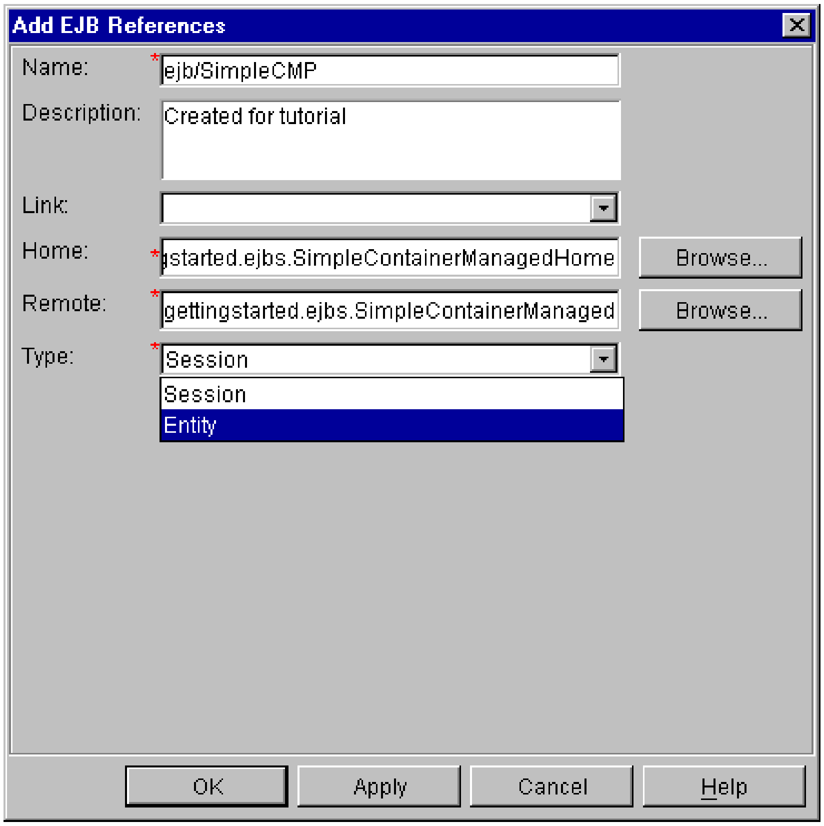
4. On the **Specifying Additional Application Client Module Properties** panel:
   a. Enter **cmp11.jar** for **Classpath.**
   b. Click **Browse** next to **Main Class** field.
   c. Expand the topology tree in the left pane as far as it goes and select **clients.**
   d. Select **SimpleContainerManagedClient.class** in the right pane.



Root Directory or Archive: E:\WebSphere\AppServer\bin\cmpApp.jar

Name	Path	Size	Exter
SimpleContainerManagedClie...	com/ibm/w...	5172	class

   e. Click **OK.**
   f. Click **Next.**
5. Click **Next** on the **Choosing Application Client Module Icons** panel.
6. Click **Add** on the **Adding EJB References** panel:
   a. Enter the following:
      - Name: **ejb/SimpleCMP**
      - Home: **com.ibm.websphere.gettingstarted.ejbs.SimpleContainerManagedHome**
      - Remote: **com.ibm.websphere.gettingstarted.ejbs.SimpleContainerManaged**
   b. Set Type: **Entity**



Name: ejb/SimpleCMP

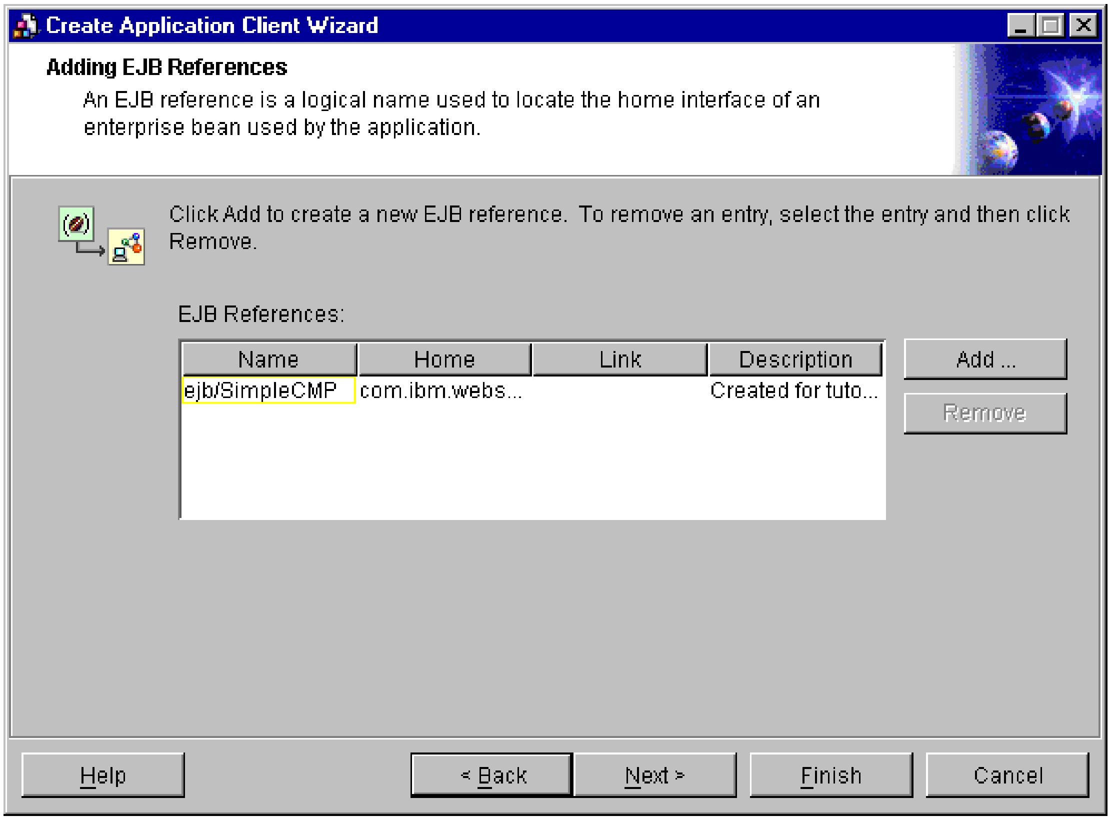
Description: Created for tutorial

Home: gstarted.ejbs.SimpleContainerManagedHome

Remote: gettingstarted.ejbs.SimpleContainerManaged

Type: Session

c. Click **OK**.
d. Verify that your EJB reference is displayed in the EJB References list.



Adding EJB References

An EJB reference is a logical name used to locate the home interface of an enterprise bean used by the application.

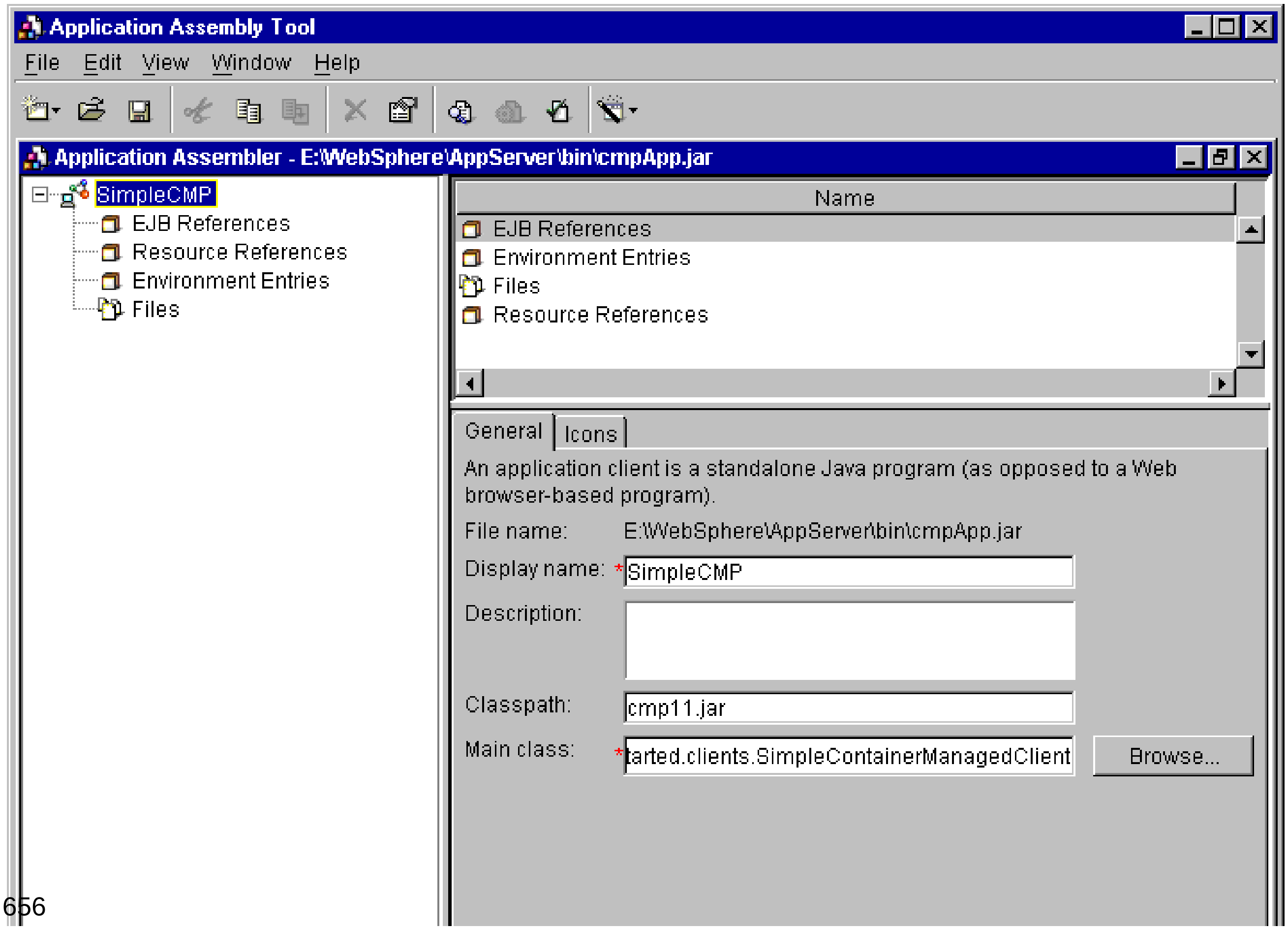
Click Add to create a new EJB reference. To remove an entry, select the entry and then click Remove.

EJB References:

Name	Home	Link	Description
ejb/SimpleCMP	com.ibm.webs...		Created for tuto...

e. Click **Next**.

7. Click **Next** on the **Adding Resource References** panel.
8. Click **Finish** on the **Adding Environment Entries** panel.
9. Save the application client jar file using **File -> Save As**. Name it **cmpApp.jar.**



An application client is a standalone Java program (as opposed to a Web browser-based program).

File name: E:\WebSphere\AppServer\bin\cmpApp.jar

Display name: * SimpleCMP

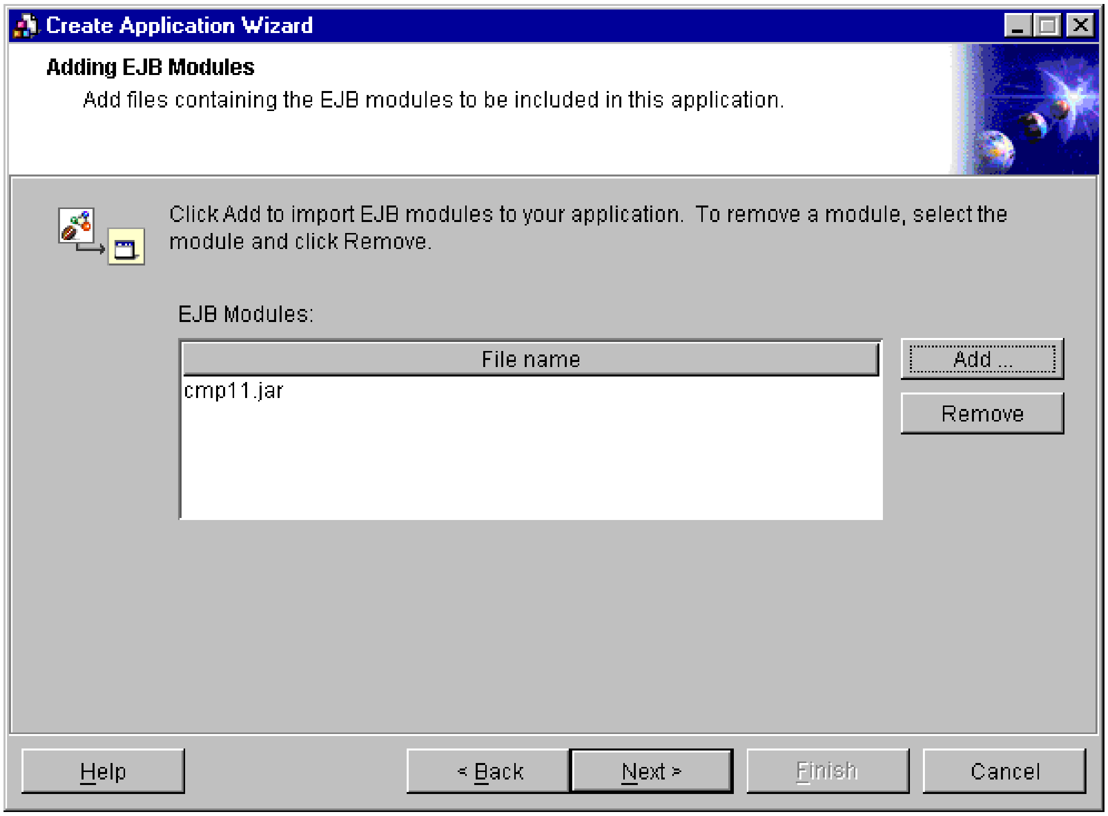
Description:

Classpath: cmp11.jar

Main class: * tarted.clients.SimpleContainerManagedClient

Assemble a J2EE application

1. Using AAT, create an .ear by clicking **File -> Wizards -> Create Application Wizard**.
2. On the **Specifying Application Properties** panel:
   a. Enter the following:
      - Display Name: **CmpApp**
      - File Name: **cmp.ear**
   b. Click **Next.**
3. Click **Next** on the **Adding Supplementary Files** panel.
4. Click **Next** on the **Choosing Application Icons** panel.
5. Click **Add...** on the **Adding EJB Modules** panel:
   a. Navigate to your **cmp11.jar** file, highlight the file and click **Open.**
   b. Click **OK** in the resulting dialog.



Create Application Wizard

Adding EJB Modules

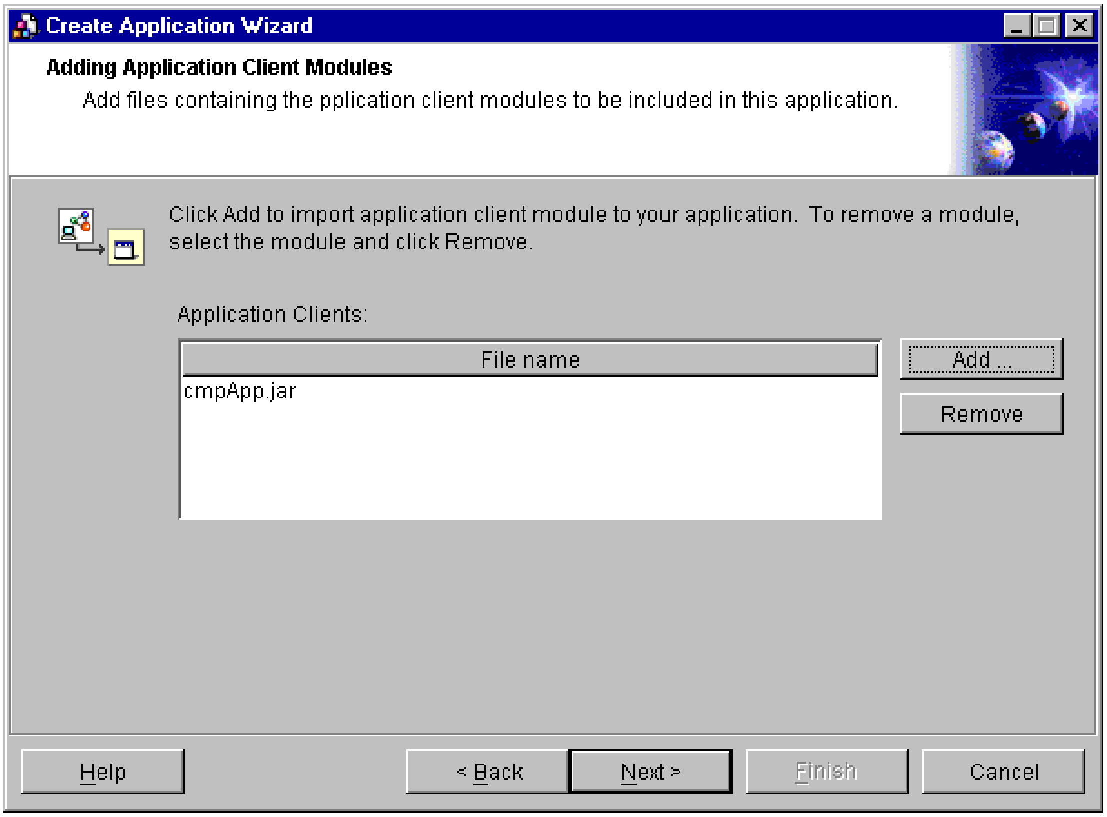
Add files containing the EJB modules to be included in this application.

Click Add to import EJB modules to your application. To remove a module, select the module and click Remove.

EJB Modules:

File name
cmp11.jar

   c. Click **Next.**
6. Click **Next** on the **Adding Web Modules** panel.
7. Click **Add...** on the **Adding Application Client Module** panel:
   a. Navigate to your **cmpApp.jar** file, hightlight the file and click **Open**.
   b. Click **OK** in the resulting dialog.



Adding Application Client Modules

Add files containing the pplication client modules to be included in this application.

Click Add to import application client module to your application. To remove a module, select the module and click Remove.

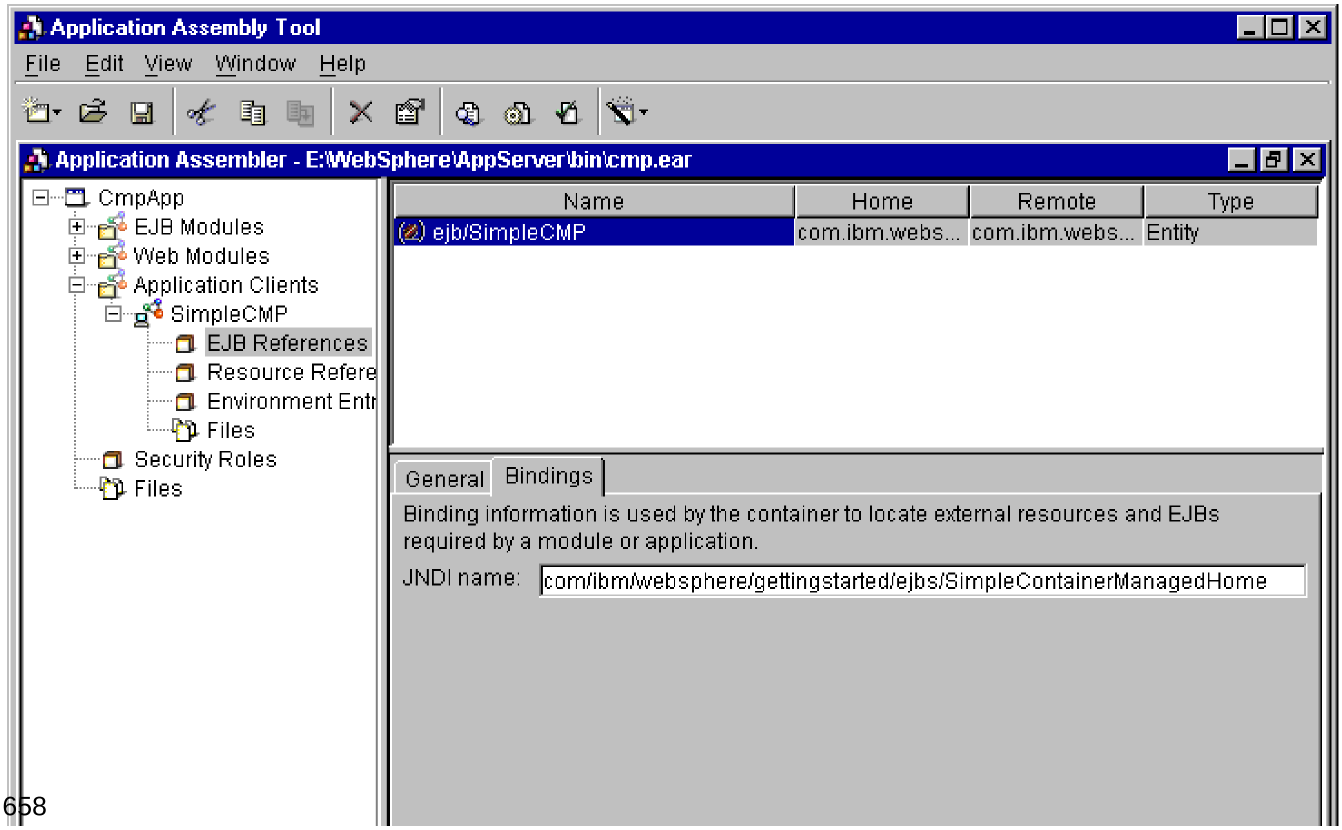
Application Clients:

File name
cmpApp.jar

c. Click **Next.**

8. Click **Finish** on the **Adding Security Roles** panel.
9. Bind the EJB reference:
   a. Expand **Application Clients**
   b. Expand **SimpleCMP**
   c. Select **EJB References.**
   d. Select in the right pane the particular reference to your session bean called **ejb/SimpleCMP.**
   e. Select the **Bindings** tab.
   f. Enter the following for the JNDI name:

   com/ibm/websphere/gettingstarted/ejbs/SimpleContainerManagedHome

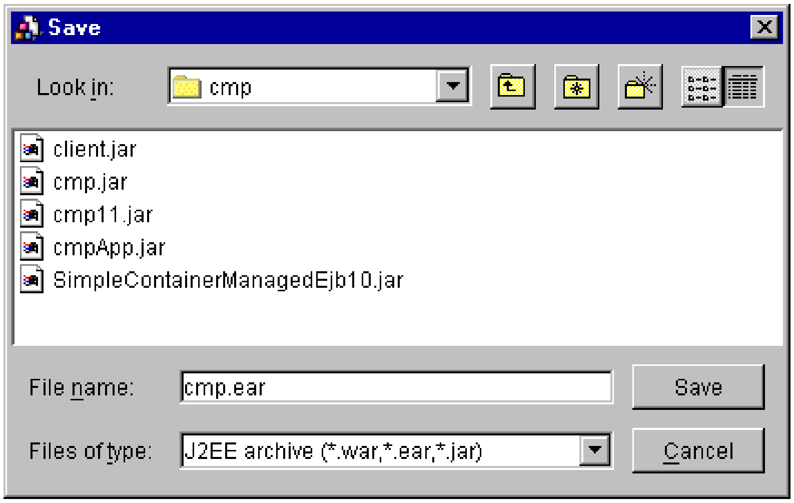


Name	Home	Remote	Type
ejb/SimpleCMP	com.ibm.webs...	com.ibm.webs...	Entity

Binding information is used by the container to locate external resources and EJBs required by a module or application.

JNDI name: com/ibm/websphere/gettingstarted/ejbs/SimpleContainerManagedHome

10. Click **Apply**.
11. Save the .ear file by clicking **File -> Save As**. Name it **cmp.ear**.



Look in: cmp

client.jar
cmp.jar
cmp11.jar
cmpApp.jar
SimpleContainerManagedEjb10.jar

File name: cmp.ear

Files of type: J2EE archive (*.war,*.ear,*.jar)

12. Generate code for deployment of the .ear file:
    a. Select the .ear file by selecting the top entry in the topology tree (named CmpApp).
    b. Right-click it.
    c. Select **Generate Code for Deployment**.
    d. In the Generate Code for Deployment panel:
        1. Accept the default for the Deployed module location.
        2. Select the appropriate database type from the list.
    e. Click **Generate Now**.
    f. When the deployment is complete, close this window.

    The file containing the code generated for deployment will be created called **Deployed_cmp.ear.**

It might take a while to generate the code for deployment. You can proceed to configure your data source, as described in the next few steps.

The terminology is potentially confusing here. The file containing the code generated for deployment on the application server is called the Deployed_*filename*, although it has not been deployed on the server yet. At this point in the application assembly and deployment procedure, the name "Deployed_*filename*" is more anticipatory than accurate. However, the name will be suitable shortly. After you deploy the "Deployed" file, the name will be useful for distinguishing the file from the pre-deployed version (the one for which you have neither generated code for deployment nor installed on the application server).

Exit the Application Assembly Tool

Recall this step from the Application assembly tutorial.

Start the server

Recall this step from the Application deployment tutorial.

Open the administrative console

Recall this step from the Application deployment tutorial.

Configure a data source

The instructions differ according to whether you are using a DB2 database or Oracle database.

DB2 Database:

1. In the administrative console, update the Sample DB Driver:
    a. In the console tree view, expand **Resources -> JDBC Providers.**
    b. Select **Sample DB Driver** to display its properties.
    c. Click the **Node** tabbed page.
    d. Check the **Class Path** to ensure it shows the location of the JAR file containing the DB2 JDBC driver classes (db2java.zip) on your system.

        - **NT or Windows 2000:** db2java.zip file is located in the **sqllib\java** directory
        - **Unix:** db2java.zip file is located in the **sqllib/java12** directory
    e. Click **Apply** if you changed the classpath.
2. Configure a new data source provider:
    1. In the tree view, expand **Resources -> JDBC Drivers -> Sample DB Driver**.
    2. Right-click **Data Sources** and select **New** from the resulting menu.
    3. Enter the following values. (If a field value is not specified, leave the field blank).
        - **Name**: GettingStartedDataSource
        - **JNDI Name**: jdbc/GsDataSource
        - **Database Name:** SAMPLEDB
    4. Click **OK**.

<u>Oracle Database:</u>

1. In the administrative console, create the OracleJdbcDriver:
    a. In the topology tree, expand **Resources**
    b. Select **JDBC Drivers.**
    c. Click **New.**
    d. Enter the following:
        - Server Class Path: **<oracle JDBC classes12.zip location>**
        - Name: **OracleJdbcDriver**
        - Description: **Oracle JDBC Driver**
        - Implementation Class: **oracle.jdbc.pool.OracleConnectionPoolDataSource**
    e. Click **OK.**
    f. Save the configuration by clicking **Save.**
2. In the administrative console, configure a new data source:
    a. In the topology tree, expand **Resources -> JDBC Drivers -> OracleJdbcDriver**.
    a. Select **Data Sources.**
    b. Click **New.**
    c. Enter the following values. (If a field value is not specified, leave the field blank).
        - Name: **GettingStartedDataSource**
        - JNDI Name: **jdbc/GsDataSource**
        - Default User ID: **<Oracle DB userid>**
        - Default Password: **<Oracle DB password>**
    d. Click **OK**.
    e. In the topology tree, expand **Resources -> JDBC Drivers -> OracleJdbcDriver -> DataSources.**
    f. Select **GettingStartedDataSource.**
    g. Select **Property Set** at bottom of screen.
    h. Select **Resource Properties.**
    i. Click **New.**
    j. Enter the following:
        - Name: **URL**
        - Type: **java.lang.String**
        - Value: **jdbc:oracle:thin:@<hostname>:1521:<your_Oracle_instance_name>**
    k. Click **OK**.
    l. Click **OK.**
    m. Click **OK.**
    n. Save the configuration by clicking **Save.** Save it under **server-cfg.xml** and click **OK.**

Install the application

Ensure you have finished generating code for deployment, before beginning this step.

1. In the console tree view, right-click **WebSphere Administrative Domain -> Enterprise Applications**.
2. From the resulting menu, click **Install Enterprise Application** to launch the **Install Enterprise Application** wizard.
3. On the **Specifying the Application or Module** panel:
    1. Ensure that the **Browse for file on node** field is set to your current node.
    2. Ensure **Install Application** is selected.
    3. Click **Browse**, next to the **Path** field. Browse for the Deployed_cmp.ear file and select it.

       Verify that its name is displayed in the **Path** field now.
    4. Specify CmpApp as the **Application name**.
    5. Click **Next**.
4. On the **Mapping Users to Roles** panel, click **Next**.

5. On the **Mapping EJB RunAs Roles to Users** panel, click **Next**.
6. On the **Binding Enterprise Beans to JNDI Names** panel, verify that the **JNDI Name** is set to com/ibm/websphere/gettingstarted/ejbs/SimpleContainerManagedHome, and then click **Next**.
7. On the **Mapping EJB References to Enterprise Beans** panel, click **Next**.
8. On the **Mapping Resource References to Resources** panel, click **Next**.
9. On the **Specifying the Default Datasource for EJB Modules** panel, verify that the **JNDI Name** is set to jdbc/GsDataSource, and click **Next**.
10. On the **Specifying Data Sources for Individual CMP Beans** panel,verify that the **JNDI Name** is set to jdbc/GsDataSource, and click **Next**.
11. On the **Selecting Virtual Hosts for Web Modules** panel, ensure that the **Virtual Host** is set to default_host, then click **Next**.
12. On the **Selecting Application Server** panel, ensure that the modules in CmpApp are going to reside on **Application Server** "Default Server," then click **Next**.
13. Click **Finish** on the **Completing the Application Installation Wizard** panel.
14. When prompted whether to regenerate code, click **No**.
15. Look for the message confirming successful installation of the application. It might be a minute before it is displayed.

You can now view the CmpApp in the console tree view.

Regenerate the plug-in configuration

Recall this step from the basic Application deployment tutorial.

Stop the server and start it again

Recall this step from the basic Application deployment tutorial.

Create database table or tablespace

The instructions differ according to whether you are using a DB2 database or an Oracle database.

DB2 Database:

1. Open a command window.
2. Type **db2cmd** (NT/Windows 2000) or **db2** (Unix) to start a DB2 command window.
3. **If** you do not already have a database named SAMPLEDB, you must create one:
    a. **db2 create database SAMPLEDB**
4. Extract the DDL from the deployed jar file:
    1. In the WebSphere administrative console tree view, select **Applications ->** ***CmpApp*** **-> EJB Modules** to display the list of EJB modules in the top right part of the console.
    2. Right-click the EJB module and select **Export Table DDL** from the resulting menu.
5. Create the table:
    a. **db2 connect to SAMPLEDB**
    b. **db2 -t -f META-INF\Table.ddl**
    A table named COM_IBM_WEBSPHERE_GETTINGSTAR1 will be added to the SAMPLEDB database.
6. Exit the db2 command window.

Oracle Database:

1. Open a command window.
2. Extract the DDL from the deployed jar file:
    a. **jar -xvf %was_home%\installedApps\Deployed_cmp.ear\cmp11.jar META-INF\Table.ddl**
3. Create the tablespaces:
    a. **svrmgrl**
    b. **connect <oracle db userid>/<oracle db password>**
    c. **@META-INF\Table.ddl**
    A tablespace named COM_IBM_WEBSPHERE_GETTINGSTAR1 will be added database.
4. Exit the oracle command window.

Ensure that the application, application server, and Web server are running

Recall learning this step in the Application testing tutorial.

Test the Java application client

1. Open a system command prompt.
2. Change directory to:

    ```
    product_installation_root/bin
    ```

3. Enter the command:

    ```
    launchClient ../temp/tutorial/cmp/Deployed_cmp.ear
    ```

    (Remember, on Windows systems, the forward slashes should be backslashes).

What's next?

Proceed to the Application removal and cleanup tutorial if you would like to uninstall the applications that you introduced into the server runtime during the tutorial.

6.7.6: Application removal and cleanup tutorial

During this tutorial, you will use the administrative console to uninstall (remove) the simpleSession application (or perhaps simpleSessionSecure application if you performed the Application security tutorial).

Prerequisites

This tutorial assumes that you have installed the simpleSession application (unsecured or secured) from a previous tutorial, and are okay with uninstalling it now. You can extrapolate the instructions to remove the applications pertaining to the SOAP tutorial and advanced CMP tutorial also.

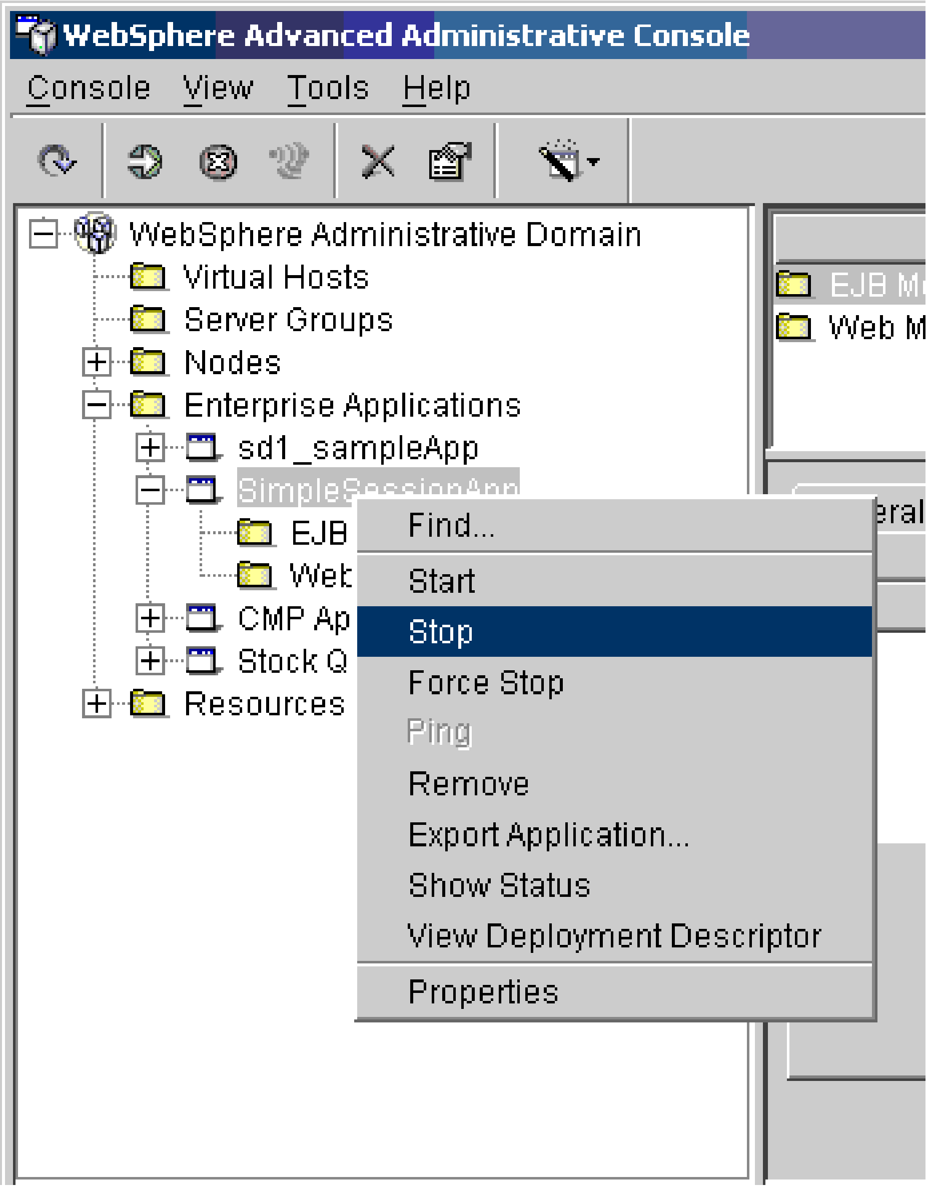
Overview of steps (5 minutes)

1. Start the administrative server.
2. Open the administrative console.
3. Stop the SimpleSessionApp.
4. Uninstall the application.

You learned most of the above steps already, in the Application deployment tutorial.

Uninstall the application

1. In the tree view of the console, right-click **WebSphere Administrative Domain -> Enterprise Applications -> SimpleSessionApp**.
2. From the resulting menu, click **Stop**.



3. Wait for the message confirming that the application was stopped successfully.
4. Right-click **SimpleSessionApp** again and click **Remove**.
5. For the purposes of the tutorial, click **No** when prompted about the application bindings.
6. Click **Yes** when prompted to confirm whether to remove the application.
7. Look for the message that the application was removed (uninstalled) successfully.

Tutorial -- Deploying a Java class as a Web service using SOAP

During this tutorial, you will deploy a Java class as a Web service in WebSphere Application Server. You will use the Stock Quote sample, which is one of the SOAP samples included with the product. Specifically, you will assemble a J2EE application .ear file containing the Stock Quote JavaBean component. Then you will SOAP-enable the .ear file and deploy it in the server runtime. Finally, you will test the enterprise application.

The value of Web services and SOAP technology is that you can take code components that were not previously Web applications -- such as enterprise beans or your Stock Quote sample -- and expose them as services available to HTTP requests (Apache SOAP Remote Procedure Call uses HTTP as its transport). You can do so without rewriting the code, in most cases. For exceptions, see "Creating Type Mappings" in the Apache SOAP 2.2 documentation on the Apache ' Organization Web site. In other words, SOAP can be applied as a transparent wrapper to Web-enable the components.

Note, SOAP message services *do* have to be aware of SOAP. An option is to write your own SOAP provider to get the message and translate it to match the method expected by the code component (object) that you are calling. For example, this practice could be used for communication between SOAP Messaging and the JMS layer. For more information, see the Apache site.

Paths through the tutorial

- **Quick path:** If your objective is to practice these steps as quickly as possible, skip the explanations marked with the ℹ graphic.
- **Maximum learning path:** If your objective is to understand and explore the product capabilities, read the ℹ notes and browse the links they provide to additional documentation.

Overview of steps (requires 30 - 60 minutes)

- Background

1. Expanding the enterprise application EAR file (soapsamples.ear)
2. Assembling the JavaBean into a JAR file
3. Starting the Application Assembly Tool
4. Assembling the EAR file containing the JavaBean JAR
5. Creating an Apache SOAP deployment descriptor
6. Exiting the Application Assembly Tool, saving your EAR file
7. SOAP-enabling the services in the EAR file
8. Starting the server
9. Starting the administrative console
10. Installing the EAR file into the server runtime
11. Regenerating the plug-in configuration
12. Stopping the application server and starting it again
13. Testing the Stock Quote service

Background

ℹ Use these links to expand your J2EE vocabulary. These terms are used throughout the tutorial.

- What are Web services?
- What are enterprise applications?
- What are Enterprise Archive (EAR) files?
- What are EJB modules?
- What are Web modules?
- What are Web Archive (WAR) files?
- What are application client modules?
- More concepts and terminology

See also the SOAP coverage in the InfoCenter.

ℹ You might notice small discrepancies between the graphics in this tutorial and the actual product screens as this documentation ages and product fix packs are issued.

Expanding the enterprise application EAR file (soapsamples.ear)

Use the following steps to obtain the SOAP sample server and client code from the soapsamples.ear file provided with the product. Expand the contents of the .ear file into a temporary directory as described here.

1. Create a soapsamples subdirectory in the path:

```
product_installation_root\temp
```

2. Change directory to:

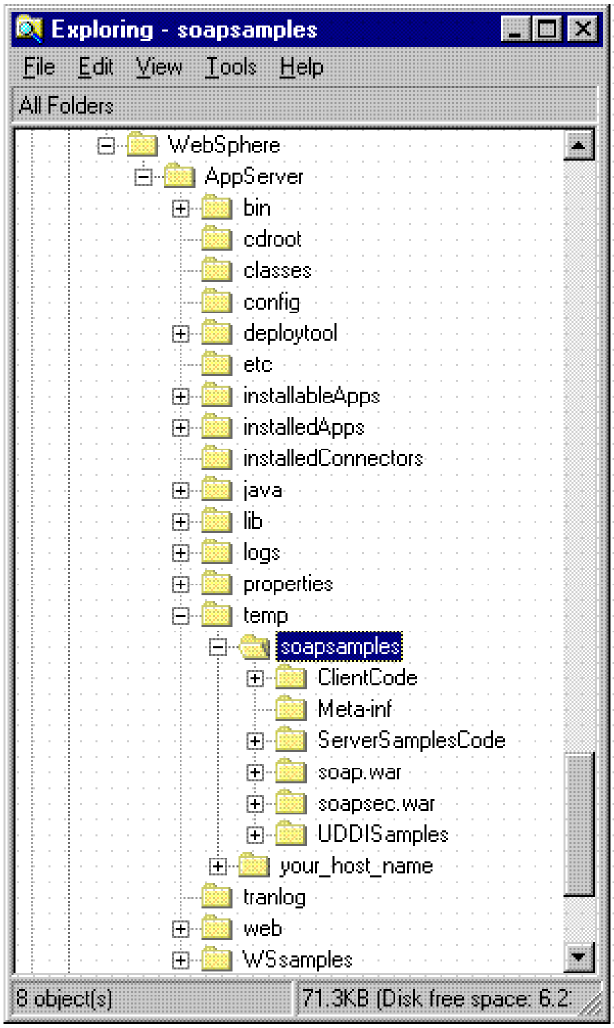
```
product_installation_root\bin
```

In this case, it is important to issue the command from bin because the arguments for the tool will be specified relative to the bin directory. On other occasions, you can use the tool from any directory, provided you specify the file paths in the arguments relative to that directory.

3. Expand the EAR file, using the EAR expander tool.
   1. Open a system command prompt.
   2. Issue the command (on a single line, using the appropriate forward or backward slashes for your system):

```
earexpander -ear
..\installableApps\soapsamples.ear -expandDir
..\temp\soapsamples -operation expand -expansionFlags war
```

When you finish, your temp directory will contain the contents of the expanded .ear file.



To learn more about the tool syntax, type earexpander at a command prompt, with no arguments. The -expansionFlags war argument is notable. This tells the tool to expand the .war files, but not the .jar files. If you see a directory named samples.jar in your temp/soapsamples directory, delete the soapsamples directory and issue the command again. Pay special attention to issuing the -expansionFlags war correctly.

Assembling the JavaBean into a JAR file

For the example used in the tutorial, you can skip this step.

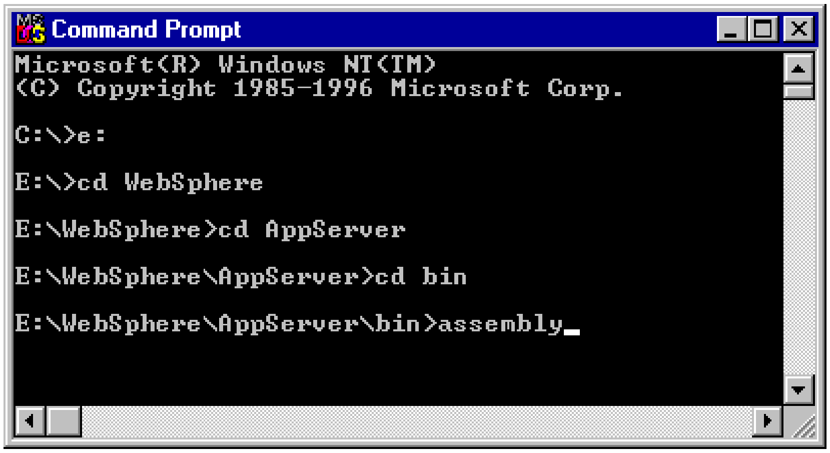
At this point, the code artifact (in this case, the Stock Quote JavaBean component) is typically compiled and then packaged into a JAR file. Because the Stock Quote sample is already packaged into samples.jar, you can skip this step. The samples.jar file is part of soapsamples.ear, as you will notice by viewing the contents of the expanded .ear file from the previous step.

Starting the Application Assembly Tool

The next step is assemble an EAR file (enterprise application) containing the JAR file for the Stock Quote JavaBean component. To do so, you must start the Application Assembly Tool.

1. Open a system command prompt.
2. Change directory to:

   product_installation_root/bin
3. Enter: assembly



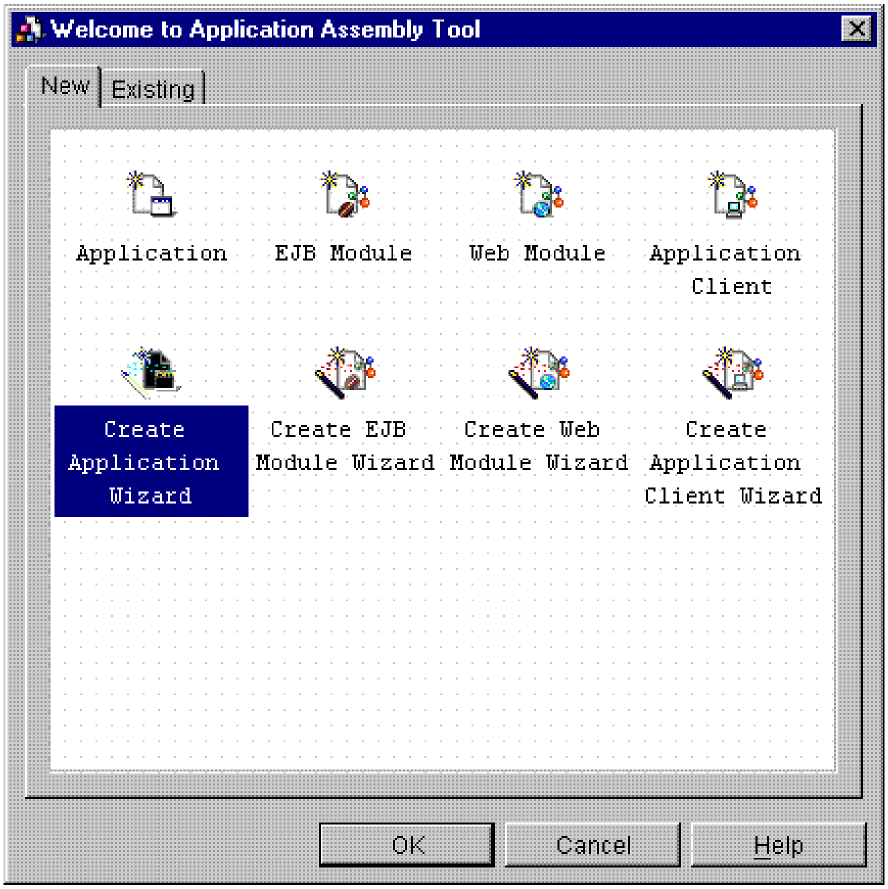
```
Microsoft(R) Windows NT(TM)
(C) Copyright 1985-1996 Microsoft Corp.

C:\>e:

E:\>cd WebSphere

E:\WebSphere>cd AppServer

E:\WebSphere\AppServer>cd bin

E:\WebSphere\AppServer\bin>assembly_
```

NT If using Windows NT or 2000, and you do not see the graphical interface of the tool right away, check for the minimized tool on the Task Bar.

Assembling the EAR file containing the JavaBean JAR

Now, create the enterprise application archive (EAR) with the Application Assembly Tool.

1. Create a new enterprise application archive (.ear file) containing the sample.jar.
   1. Select the **Create Application Wizard**.



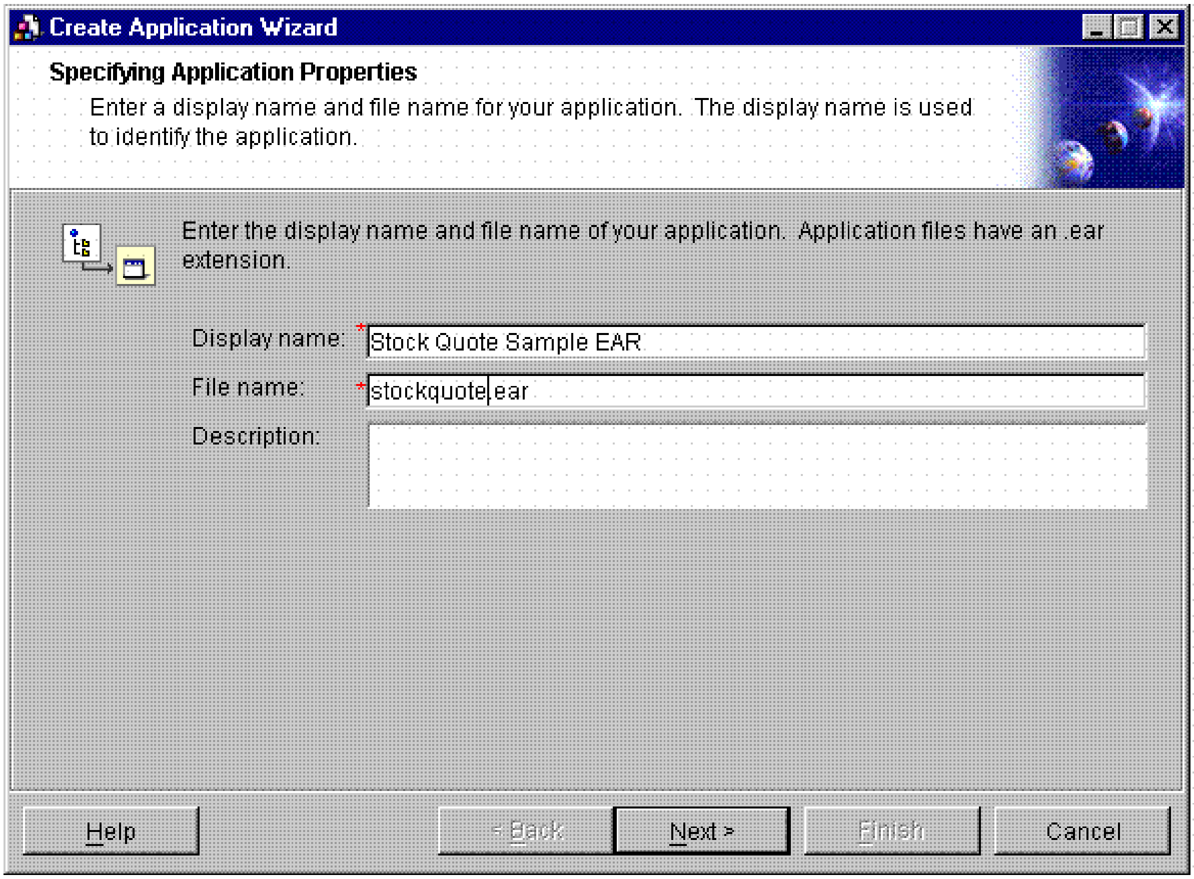
   2. In the **Specifying Application Properties** panel of the wizard, specify information:

      Display name

         Stock Quote Sample EAR

File name

stockquote.ear



Specifying Application Properties

Enter a display name and file name for your application. The display name is used to identify the application.

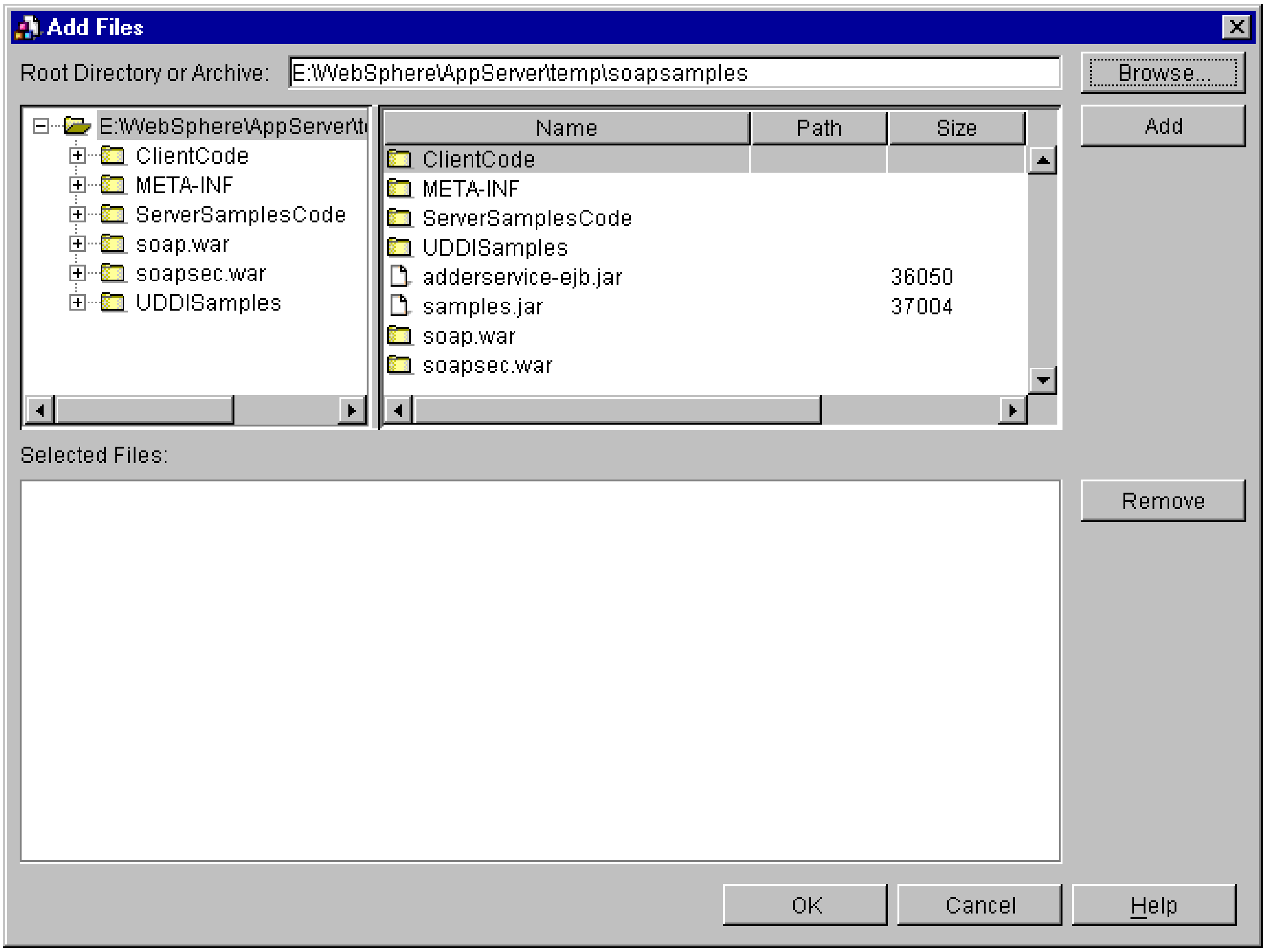
Enter the display name and file name of your application. Application files have an .ear extension.

Display name: * Stock Quote Sample EAR

File name: * stockquote.ear

Description:

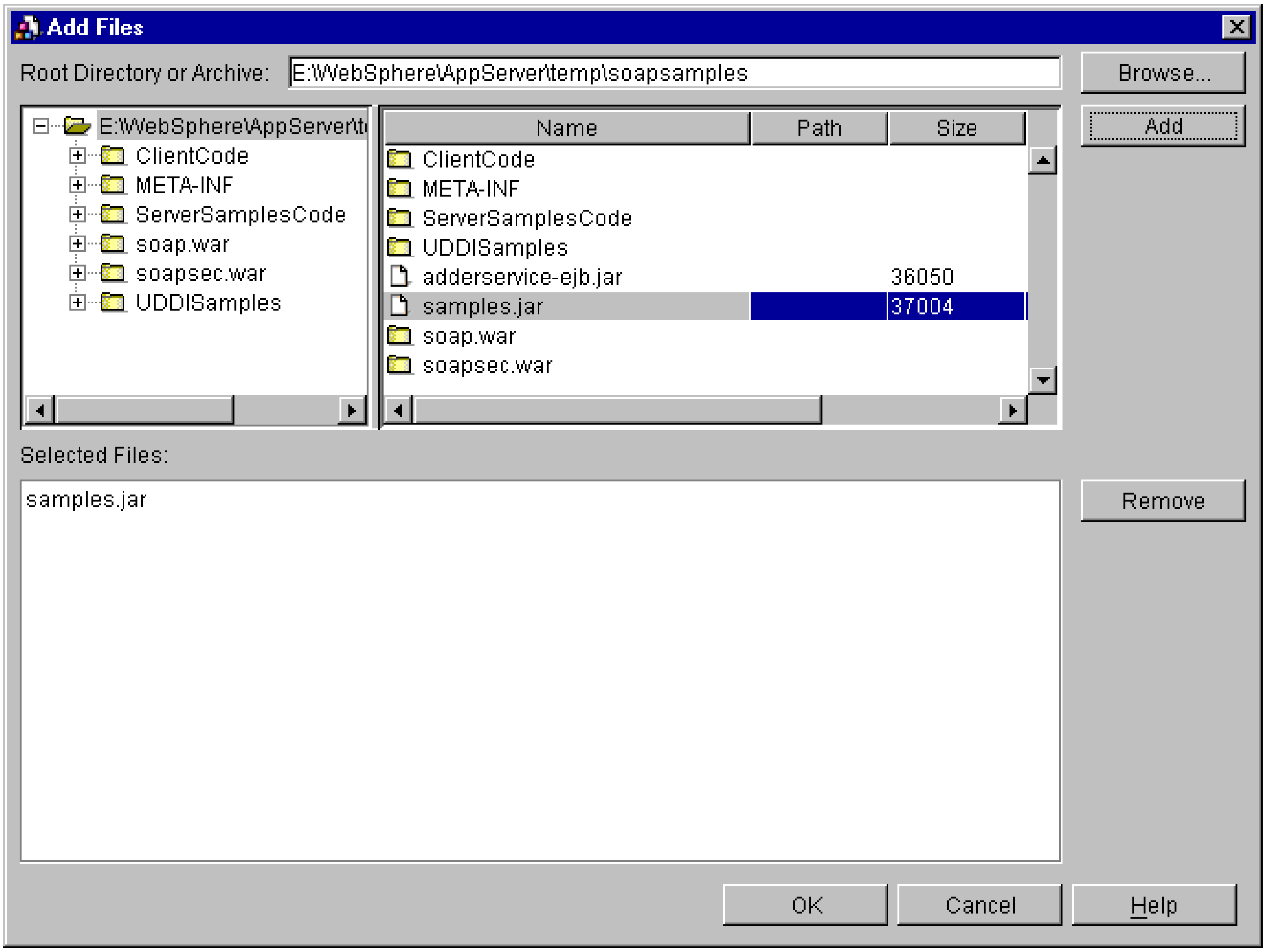
3. Click **Next**.
4. On the **Adding Supplementary Files** panel of the wizard, click **Add**.
5. Browse for and select **the directory** containing the samples.jar file -- that is, select the temp/soapsamples directory.



Root Directory or Archive: E:\WebSphere\AppServer\temp\soapsamples

Name	Path	Size
ClientCode		
META-INF		
ServerSamplesCode		
UDDISamples		
adderservice-ejb.jar		36050
samples.jar		37004
soap.war		
soapsec.war		

6. When you return to the **Add Files** dialog window:
   - Select the `samples.jar` file in the list in the upper right part of the screen.
   - Click **Add** to display the `samples.jar` in the lower half of the screen.

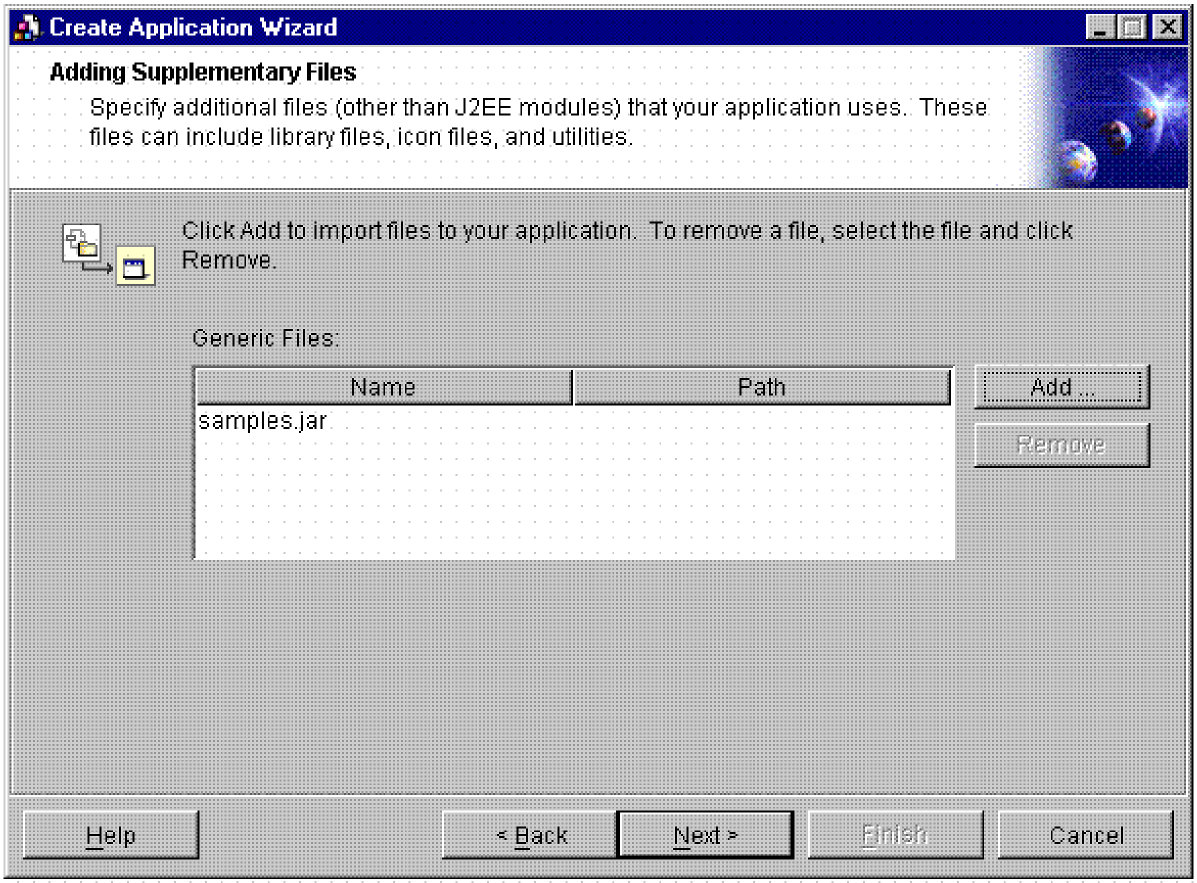


Root Directory or Archive: E:\WebSphere\AppServer\temp\soapsamples

Name	Path	Size
ClientCode		
META-INF		
ServerSamplesCode		
UDDISamples		
adderservice-ejb.jar		36050
samples.jar		37004
soap.war		
soapsec.war		

Selected Files: samples.jar

7. Click **OK** to exit the **Add Files** dialog window and return to the main wizard.



Adding Supplementary Files

Specify additional files (other than J2EE modules) that your application uses. These files can include library files, icon files, and utilities.

Click Add to import files to your application. To remove a file, select the file and click Remove.

Generic Files:

Name	Path
samples.jar	

    8. Click **Next** until you reach a panel on which you can click **Finish**. Do so, to exit the wizard.

       Advancing through each panel will give you a good idea of the types of application characteristics you can configure with this wizard. To learn more about the wizard, read the help file for the wizard.

2. As a formality, create a .war file for the enterprise application.

   The main assembly tool window will display a tree view containing the Stock Quote Sample EAR application.

    1. Expand the tree view until you can see the **Web Modules** folder.
    2. Right-click the folder and select **New** from the resulting menu.
    3. Specify the general Web module properties.

       File name

       stockquote.war

       Context root

       /stockquote

       Display name

       Stock Quote Sample

       Note, the context root value is arbitrary.
    4. Click **OK**.

   You will now be able to see the Stock Quote Sample if you expand the assembly tool tree view to show the contents of the **Web Modules** folder.

   Why create a .war file (Web module archive) if you are not enabling a servlet or JSP file? The Application Assembly Tool requires the EAR file to contain at least one J2EE archive, meaning a .war file or an EJB or application client .jar file. In this case, you will create an empty .war file (Web module) to satisfy the requirement.

Creating an Apache SOAP deployment descriptor

For the example used in the tutorial, you can skip this step.

Before running the SOAP EAR Enabler tool, typically you must create an Apache SOAP Deployment Descriptor for each service to be enabled. For the Stock Quote sample, the deployment descriptor was created for you. To view it:

1. Right-click the **Stock Quote Sample EAR** application in the assembly tool tree view.
2. Select **View Descriptor** from the resulting menu.
3. When finished viewing the descriptor, click **Close**.

See also the documentation about Apache SOAP deployment descriptors.

What are descriptors? Where do they originate? Currently, you can use a text editor to create the descriptor, based on IBM documentation of Apache SOAP descriptors. Basically, the descriptor is a service definition file containing the information the SOAP runtime needs to understand how to invoke the service. In a nutshell, it contains binding information such as:

- The name by which the service can be accessed, which is generally related to the JNDI name of an enterprise bean
- The methods available on the service for calling
- What 'class' implements the service
- What SOAP provider handles this service -- for example, the generic Java provider, an EJB provider, a Bean Scripting Framework (BSF) provider, and so on.
- Any special data type mappings so it can properly serialize/deserialize the objects a method receives and returns (In the case of a BSF script, the entire script can be embedded in the descriptor, hence the reason the tool asks you for classpath requirements. For a BSF script, it is possible to have no classpath requirements
- Anything else optional that the service may need. In the case of enterprise beans, the JNDI name of the bean and what the full home interface name

Exiting the Application Assembly Tool, saving your EAR file

Save your file and exit the assembly tool.

1. Click **File** -> **Save As** on the menu bar of the tool.
2. Save the stockquote.ear file to:

```
product_installation_root/temp
```

3. Click **File** -> **Close** on the menu bar of the tool.
4. If prompted, specify to save your modifications. Then click **OK** when you receive the message that the "Archive was saved successfully."
5. Click **File** -> **Exit** to exit the Application Assembly Tool.

SOAP-enabling the services in the EAR file

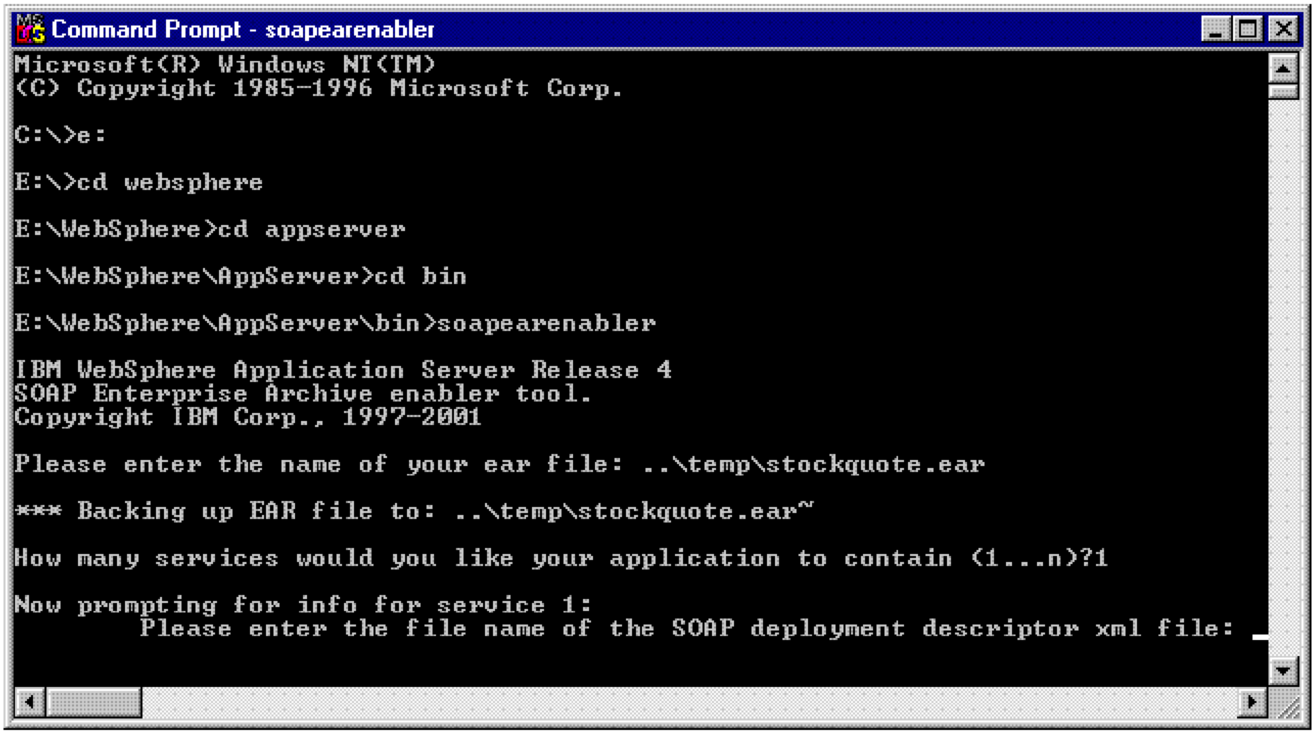
WebSphere Application Server provides a Java tool for enabling an interface in an EAR file. This Soap EAR Enabler tool takes as input the Apache SOAP Deployment Descriptor and the EAR file from the previous step.

1. Open a system command prompt.
2. Change directory to

   ```
   product_installation_root\bin
   ```

   In this case, it is important to issue the command from bin because the arguments for the tool will be specified relative to the bin directory. On other occasions, you can use the tool from any directory, provided you specify the file paths in the arguments relative to that directory.
3. Start the SOAP EAR Enabler tool by typing soapearenabler.
4. When the tool prompts you for the name of your .ear file, specify (using the correct forward or backward slashes for your system):

   ```
   ..\temp\stockquote.ear
   ```

   ```
   Microsoft(R) Windows NT(TM)
   (C) Copyright 1985-1996 Microsoft Corp.

   C:\>e:

   E:\>cd websphere

   E:\WebSphere>cd appserver

   E:\WebSphere\AppServer>cd bin

   E:\WebSphere\AppServer\bin>soapearenabler

   IBM WebSphere Application Server Release 4
   SOAP Enterprise Archive enabler tool.
   Copyright IBM Corp., 1997-2001

   Please enter the name of your ear file: ..\temp\stockquote.ear

   *** Backing up EAR file to: ..\temp\stockquote.ear~

   How many services would you like your application to contain (1...n)?1

   Now prompting for info for service 1:
           Please enter the file name of the SOAP deployment descriptor xml file: _
   ```

   What if the tool replies that you should specify a file that exists? Then your stockquote.ear is not in the temp directory. Exit the tool, move the file, and try again.
5. Answer the rest of the questions as you are prompted to do so. For readability, the following responses are shown on a separate line from the questions. When using the tool, you must enter your answer directly at the end of the question (on same line, with no spaces preceding your answer).

   How many services would you like your application to contain (1...n)?

   > 1

   Now prompting for info for service #1: Please enter the file name of the SOAP deployment descriptor xml file:

   > ..\temp\soapsamples\ServerSamplesCode\src\stockquote\DeploymentDescriptor\stockquote.xml

   Is this service an EJB (y= yes/n= no)?

   > n

   How many jar files are required for this service (0...n)?

   > 1

   Classpath requirement #1: Please choose a file ([1] samples.jar, [2] stockquote.war):

   > 1

   Should this service be secured (y/n)?

   > n

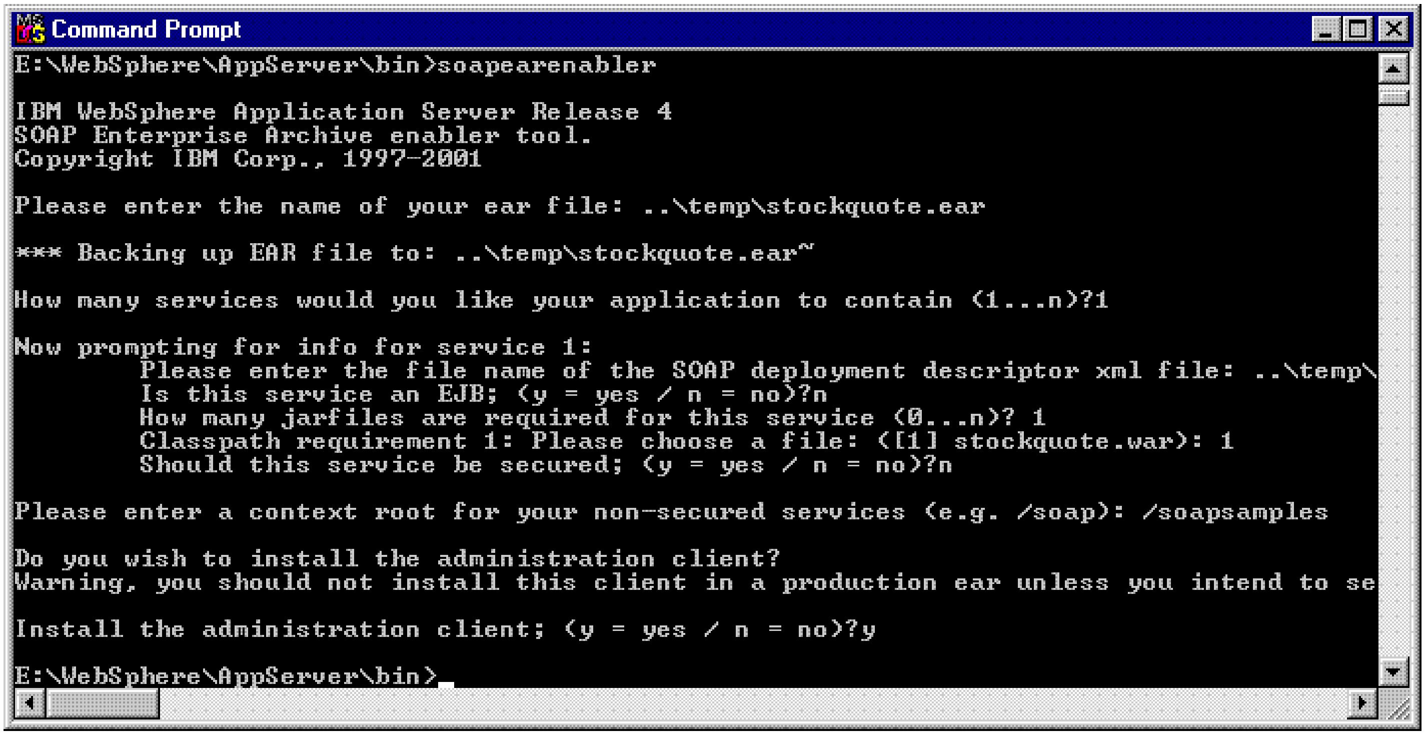
   Please enter a context root for your non-secured services (e.g. /soap):

   > /soapsamples
   >
   > The context root is important when invoking the router servlet from the client.

   Do you wish to install the administration client? Warning! You should not install this client in a production ear unless you intend to secure the URI to it. Install the administration client (y= yes/n= no)?

   > y

When the tool has received your answers for all of its questions, it will exit, and the system command prompt will return.



```
E:\WebSphere\AppServer\bin>soapearenabler

IBM WebSphere Application Server Release 4
SOAP Enterprise Archive enabler tool.
Copyright IBM Corp., 1997-2001

Please enter the name of your ear file: ..\temp\stockquote.ear

*** Backing up EAR file to: ..\temp\stockquote.ear~

How many services would you like your application to contain (1...n)?1

Now prompting for info for service 1:
        Please enter the file name of the SOAP deployment descriptor xml file: ..\temp\
        Is this service an EJB; (y = yes / n = no)?n
        How many jarfiles are required for this service (0...n)? 1
        Classpath requirement 1: Please choose a file: ([1] stockquote.war): 1
        Should this service be secured; (y = yes / n = no)?n

Please enter a context root for your non-secured services (e.g. /soap): /soapsamples

Do you wish to install the administration client?
Warning, you should not install this client in a production ear unless you intend to se

Install the administration client; (y = yes / n = no)?y

E:\WebSphere\AppServer\bin>_
```

A little bit about the classpath requirement prompt -- the reason for selecting samples.jar is to ensure that the file will be put on the classpath that the SOAP runtime will see. The samples.jar has to be at the root of the .ear file in order to show up in the classpath requirements list presented by the EAR Enabler. During the assembly phase, you might have wondered why, if you had to create an emtpy .war file to satisfy the assembly tool requirements, that you did not just put the samples.jar inside the .war file. The samples.jar has to be included outside of a Web module, EJB module, or application client module in the .ear file, in order to be added to the classpath for the SOAP runtime.

Starting the server

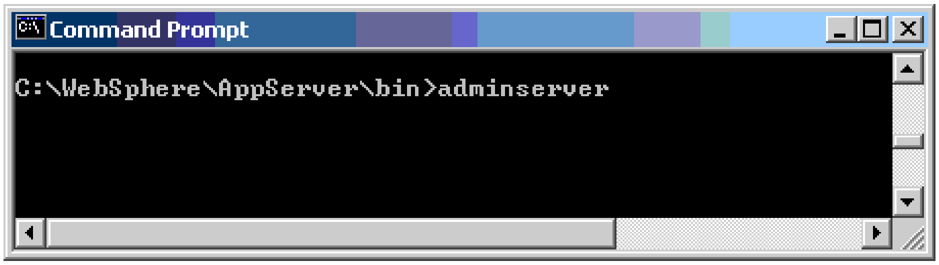
First, start the product administrative server.

1. Open a system command prompt.
2. Change directory to:

   `product_installation_root/bin`

3. Enter: adminserver

Leave the command window open, as you will use it again later to start the server.



```
C:\WebSphere\AppServer\bin>adminserver
```

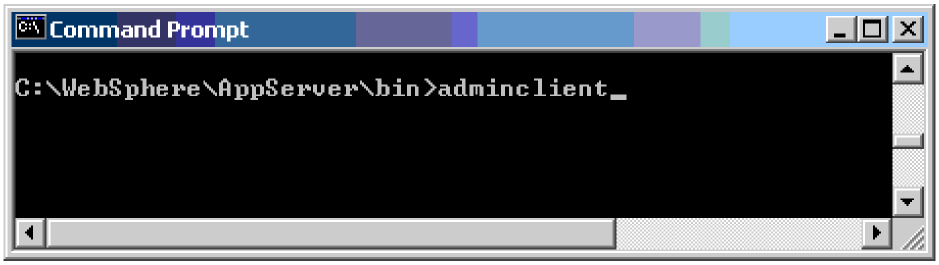
Starting the administrative console

Now it is time to deploy (install) the application in the server runtime. To do so, you will use the administrative console.

1. Open a system command prompt.
2. Change directory to:

   `product_installation_root/bin`

3. Enter: adminclient

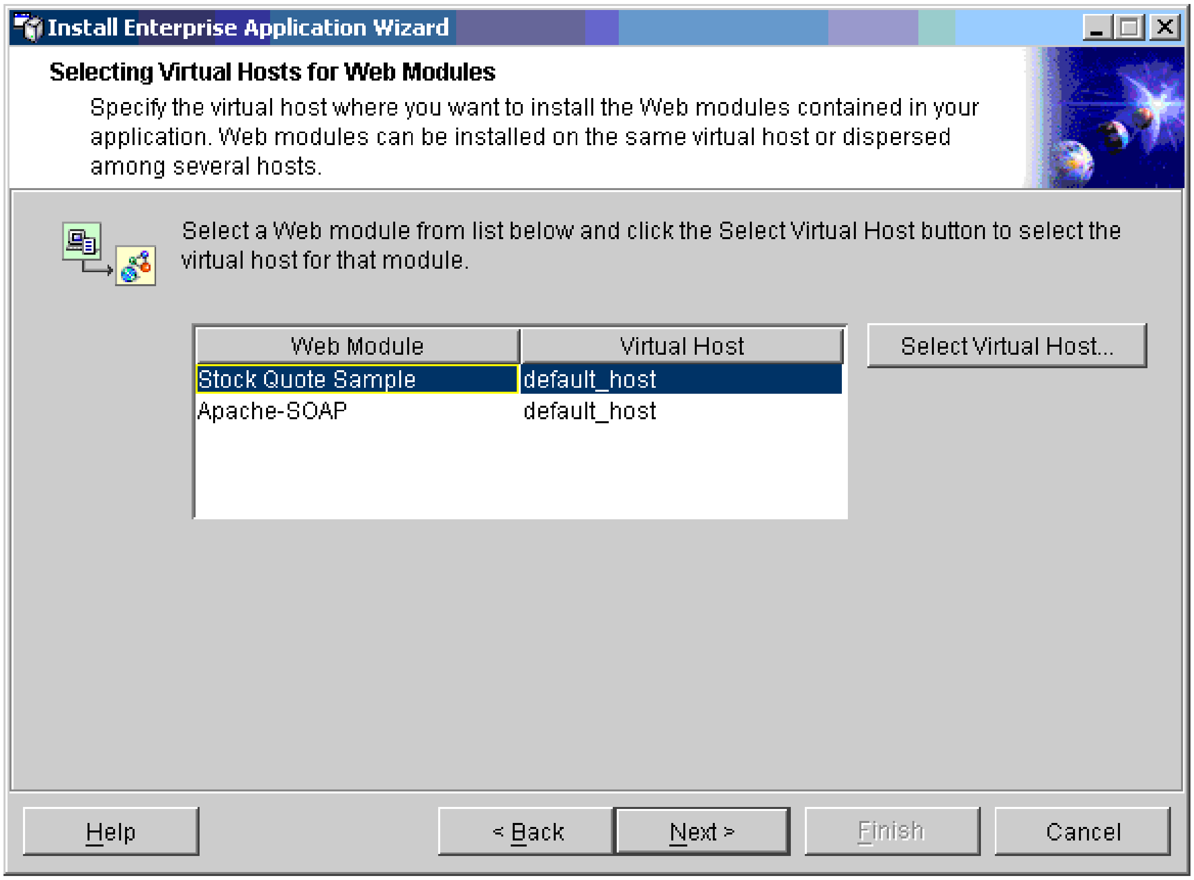


```
C:\WebSphere\AppServer\bin>adminclient_
```

For more information about variations on starting and stopping the administrative server, see see the corresponding InfoCenter article.

Installing the EAR file into the server runtime

1. Select **Console** -> **Wizards** -> **Install Enterprise Application** from the console menu bar to display the first panel of the wizard.
2. In the **Specifying the Application or Module** panel of the wizard:
    1. Ensure that the **Browse for file on node** field is set to your current node.
    2. Ensure **Install Application** is selected.
    3. Click **Browse**, next to the **Path** field. Browse for the stockquote.ear file and select it. Verify that its name is displayed in the **Path** field now.
    4. Click **Next**.
    5. Continue clicking **Next** to proceed through the panels until you reach the **Selecting Virtual Hosts for Web Modules** panel.
    6. On the **Selecting Virtual Hosts for Web Modules** panel, verify that the Web modules are using the "default_host."



Selecting Virtual Hosts for Web Modules

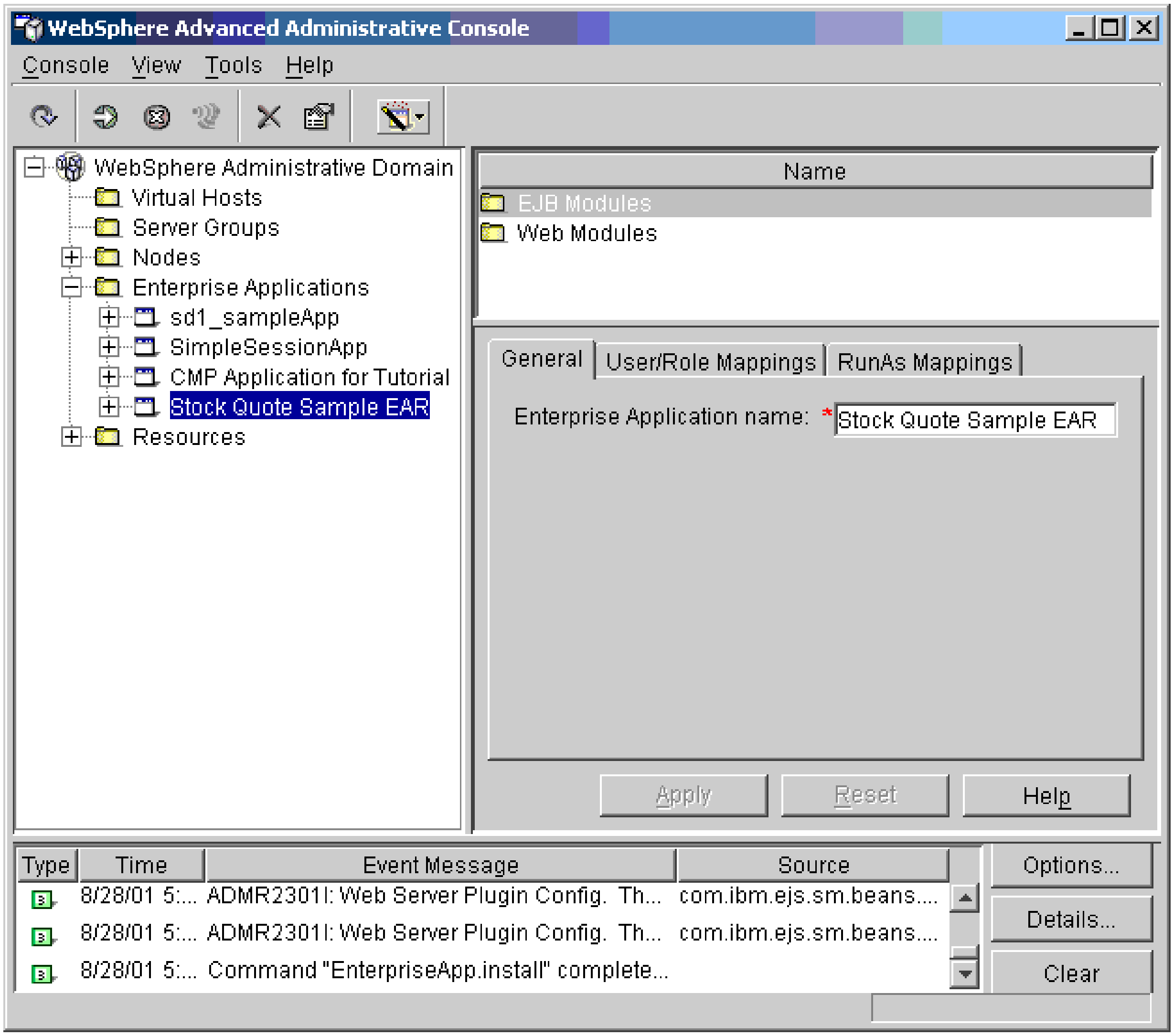
Specify the virtual host where you want to install the Web modules contained in your application. Web modules can be installed on the same virtual host or dispersed among several hosts.

Select a Web module from list below and click the Select Virtual Host button to select the virtual host for that module.

Web Module	Virtual Host
Stock Quote Sample	default_host
Apache-SOAP	default_host

3. Click **Next** to proceed until you are able to click **Finish** to exit the wizard.

The application will now be displayed in the console tree view:



Enterprise Application name: * Stock Quote Sample EAR

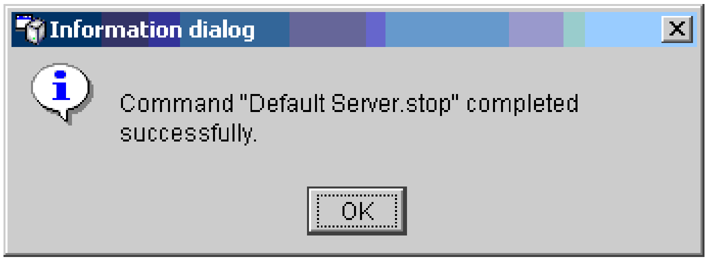
Type	Time	Event Message	Source
B	8/28/01 5:...	ADMR2301I: Web Server Plugin Config. Th...	com.ibm.ejs.sm.beans....
B	8/28/01 5:...	ADMR2301I: Web Server Plugin Config. Th...	com.ibm.ejs.sm.beans....
B	8/28/01 5:...	Command "EnterpriseApp.install" complete...	

Regenerating the plug-in configuration

1. In the console tree view, right-click **WebSphere Administrative Domain** -> **Nodes** -> ***your_host_name***.
2. From the resulting menu, select **Regen Webserver Plugin**.
3. In the area at the bottom of the console, look for the **Event Message** saying that the Plugin regeneration has been completed. It might take a moment before it is displayed.

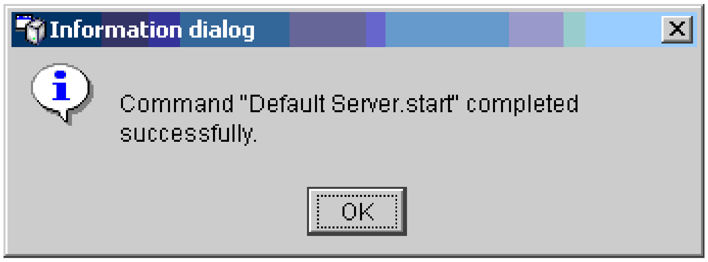
Stopping the application server and starting it again

1. In the console tree view, right-click **WebSphere Administrative Domain** -> **Nodes** -> ***your_host_name*** -> **Application Servers** -> **Default Server**.
2. From the resulting menu, click **Stop**.
3. Look for the message that the server was stopped successfully.



Command "Default Server.stop" completed successfully.

4. Now right-click **Default Server** again and click **Start** on the resulting menu.
5. Again, wait for the confirmation message.



Command "Default Server.start" completed successfully.

Because adding a new application requires a change to the server configuration, the server must be stopped and started again to pick up the change. However, there are some types of changes you can make to an installed application that do not require stopping the server and starting it again. See the dynamic reloading and hot deployment documentation for more information.

Testing the Stock Quote service

The SOAP samples come with a script to help you test the sample. The script is preconfigured with the URI of the SOAP router servlet. This URI is based on your specification of the context root when you ran the SOAP Ear Enabler tool (/soapsamples). If you specified a context root other than the default, then you must update the script.

1. Open a system command prompt.
2. Change directory to:

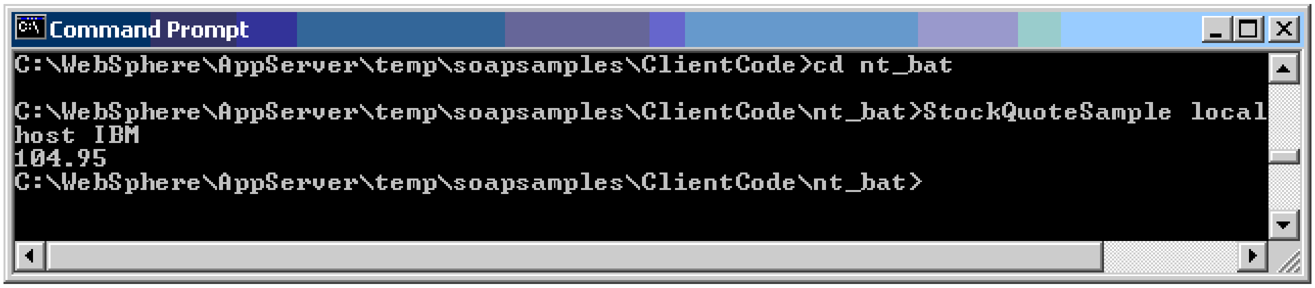
   ```
   product_installation_root\temp\soapsamples\ClientCode\
   ```
3. Change directory to where the scripts are located, for your operating system.
   - UNIX unix_scripts
   - NT nt_bat
4. Invoke the script:

   ```
   StockQuoteSample.bat|sh
   ```
   with the arguments "localhost IBM."

   For example, on Windows NT, this would be:

   ```
   StockQuoteSample localhost IBM
   ```

You should receive a current stock quote for IBM stock.



```
C:\WebSphere\AppServer\temp\soapsamples\ClientCode>cd nt_bat

C:\WebSphere\AppServer\temp\soapsamples\ClientCode\nt_bat>StockQuoteSample local
host IBM
104.95
C:\WebSphere\AppServer\temp\soapsamples\ClientCode\nt_bat>
```

If you receive a connection refused error or connection timed out error, then there could be a problem with your external access to the internet.

Workaround

If you experience a connection problem, it might have to do with your network's firewall. Try the following:

1. Start the product.
2. Open the administrative console.
3. Access the JVM command line settings.

   In the tree view, click **Nodes ->** ***your_host_name*** **-> Application Servers**. Click the **JVM** tabbed page of the application server properties.
4. At the *front* of the field for entering command line arguments, set the arguments for the socks or HTTP proxy hosts and ports. Enter the value in the **Generic Command Line Arguments** field.

   If using a socks proxy server, add:

   ```
   -DsocksProxyHost=The name of your socks proxy host
   ```
   and

```
-DsocksProxyPort=The number of your socks proxy port
```

If using an HTTP proxy server, add:

```
-DproxyHost=The name of your HTTP proxy host
```

and

```
-DproxyPort=The number of your HTTP proxy port
```

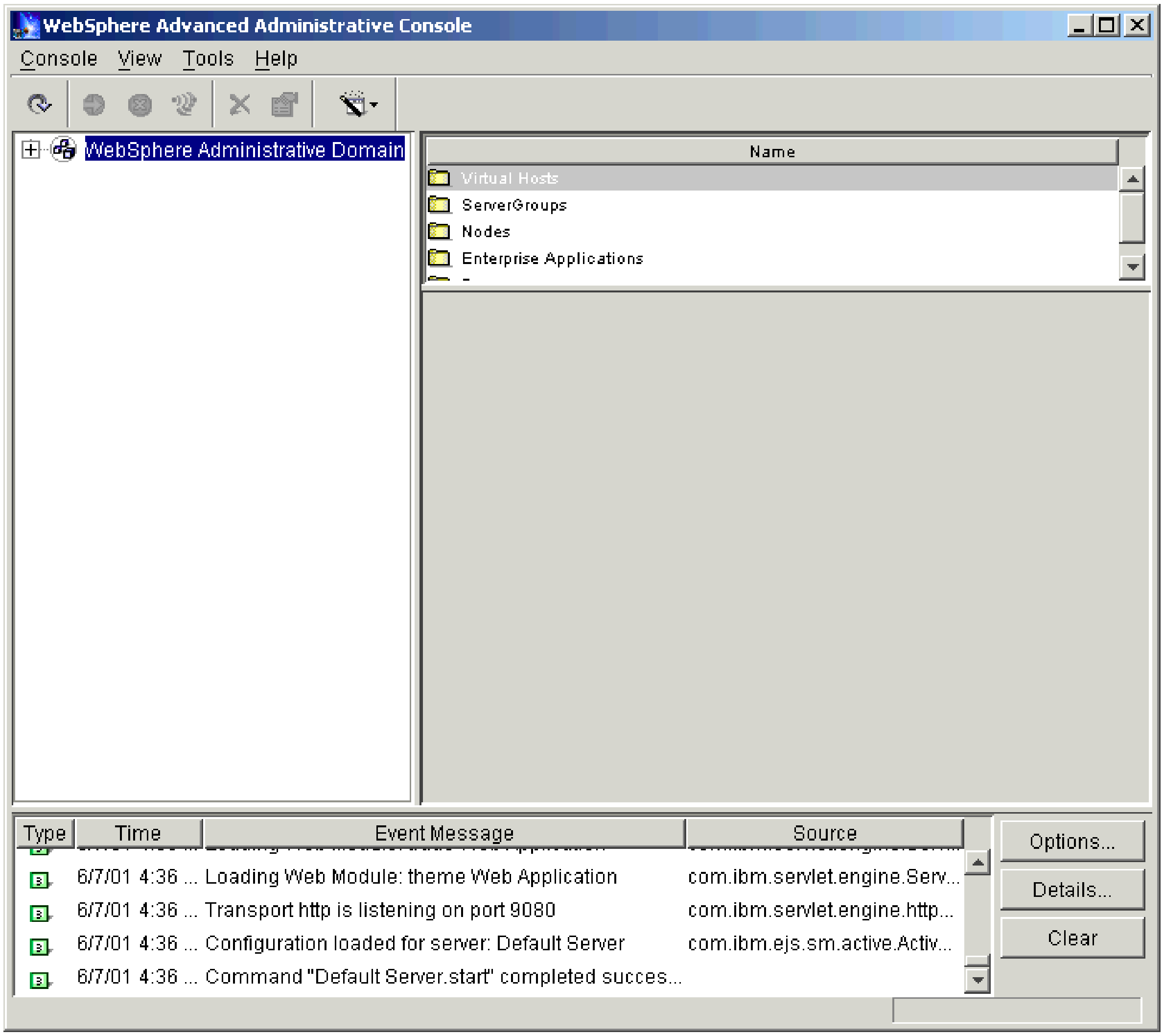
5. Stop the application server and start it again, as described previously in this tutorial.

HitCount tutorial: Using Debugger and OLT

HitCount is an example servlet that installs by default with IBM Websphere Application Server. This tutorial demonstrates how to use the IBM Distributed Debugger and Object Level Trace (OLT) on the HitCount servlet, referring to screen captures from the administrative console (WebSphere Administrative Console) used with the Application Server Advanced Single Server Edition.

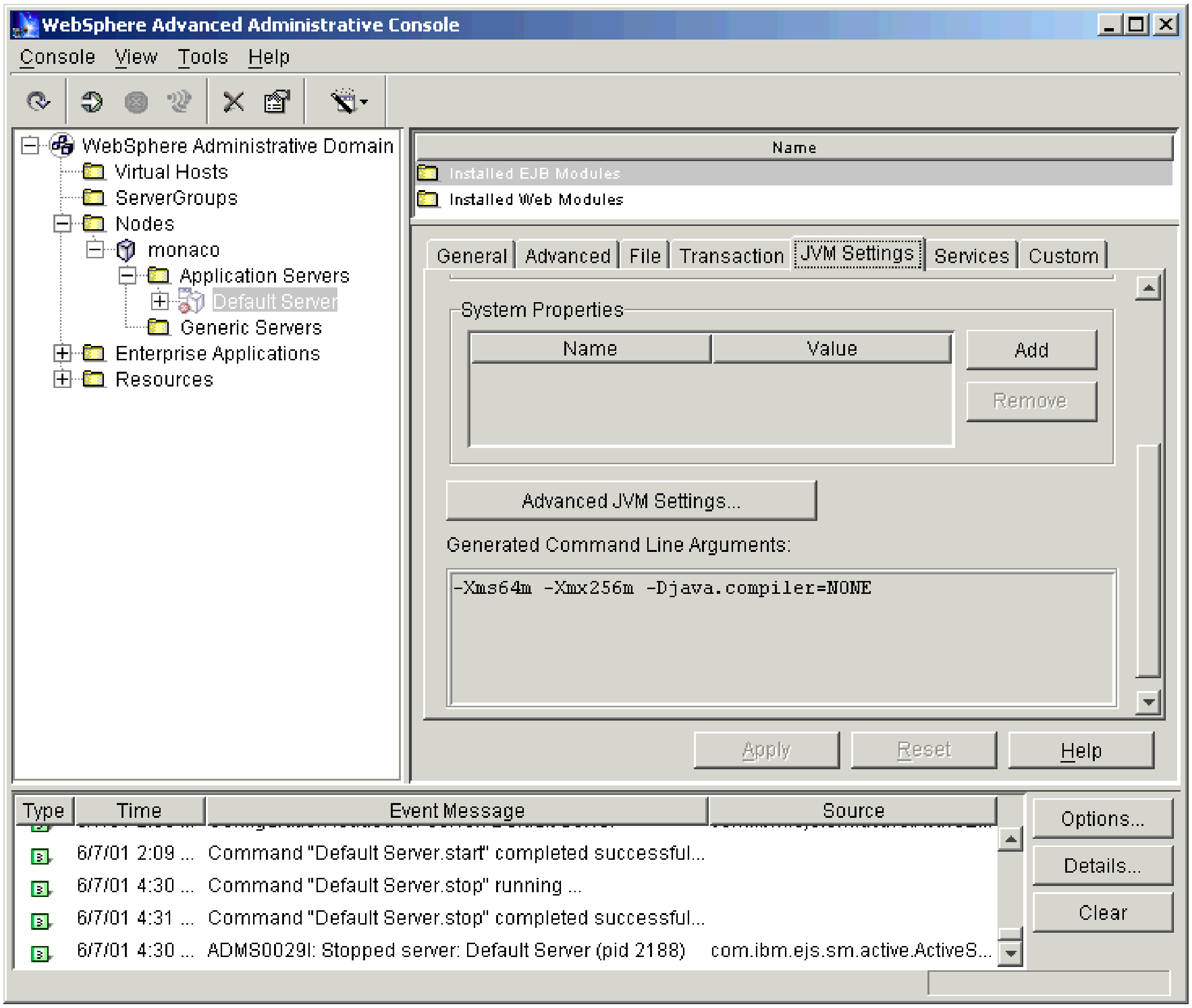
Let's begin.

1. Start the Administrative Console:



```
6/7/01 4:36 ... Loading Web Module: theme Web Application          com.ibm.servlet.engine.Serv...
6/7/01 4:36 ... Transport http is listening on port 9080           com.ibm.servlet.engine.http...
6/7/01 4:36 ... Configuration loaded for server: Default Server    com.ibm.ejs.sm.active.Activ...
6/7/01 4:36 ... Command "Default Server.start" completed succes...
```

2. Select the **Default Server** for your node in the topology view on the left side of the console, and then select the **JVM Settings** tab in the right side of the console:

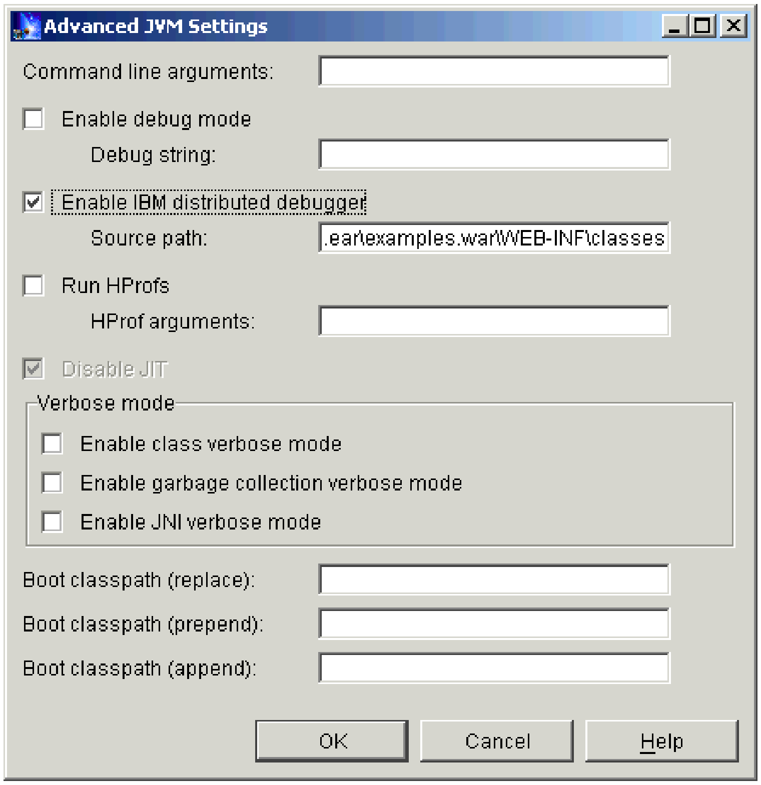


Generated Command Line Arguments:

```
-Xms64m -Xmx256m -Djava.compiler=NONE
```

Type	Time	Event Message	Source
B	6/7/01 2:09 ...	Command "Default Server.start" completed successful...	
B	6/7/01 4:30 ...	Command "Default Server.stop" running ...	
B	6/7/01 4:31 ...	Command "Default Server.stop" completed successful...	
B	6/7/01 4:30 ...	ADMS0029I: Stopped server: Default Server (pid 2188)	com.ibm.ejs.sm.active.ActiveS...

3. Click the **Advanced JVM Settings** pushbutton and then, in the **Advanced JVM Settings** dialog box, select the **Enable IBM Distributed Debugger** checkbox:



Command line arguments:

☐ Enable debug mode

Debug string:

☑ Enable IBM distributed debugger

Source path: .ear\examples.war\WEB-INF\classes

☐ Run HProfs

HProf arguments:

☑ Disable JIT

Verbose mode

☐ Enable class verbose mode

☐ Enable garbage collection verbose mode

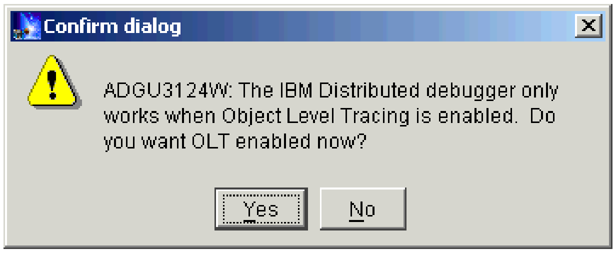
☐ Enable JNI verbose mode

Boot classpath (replace):

Boot classpath (prepend):

Boot classpath (append):

You will be prompted with a **Confirm dialog** message, asking you if you would like to enable OLT. Click **Yes** to enable OLT:



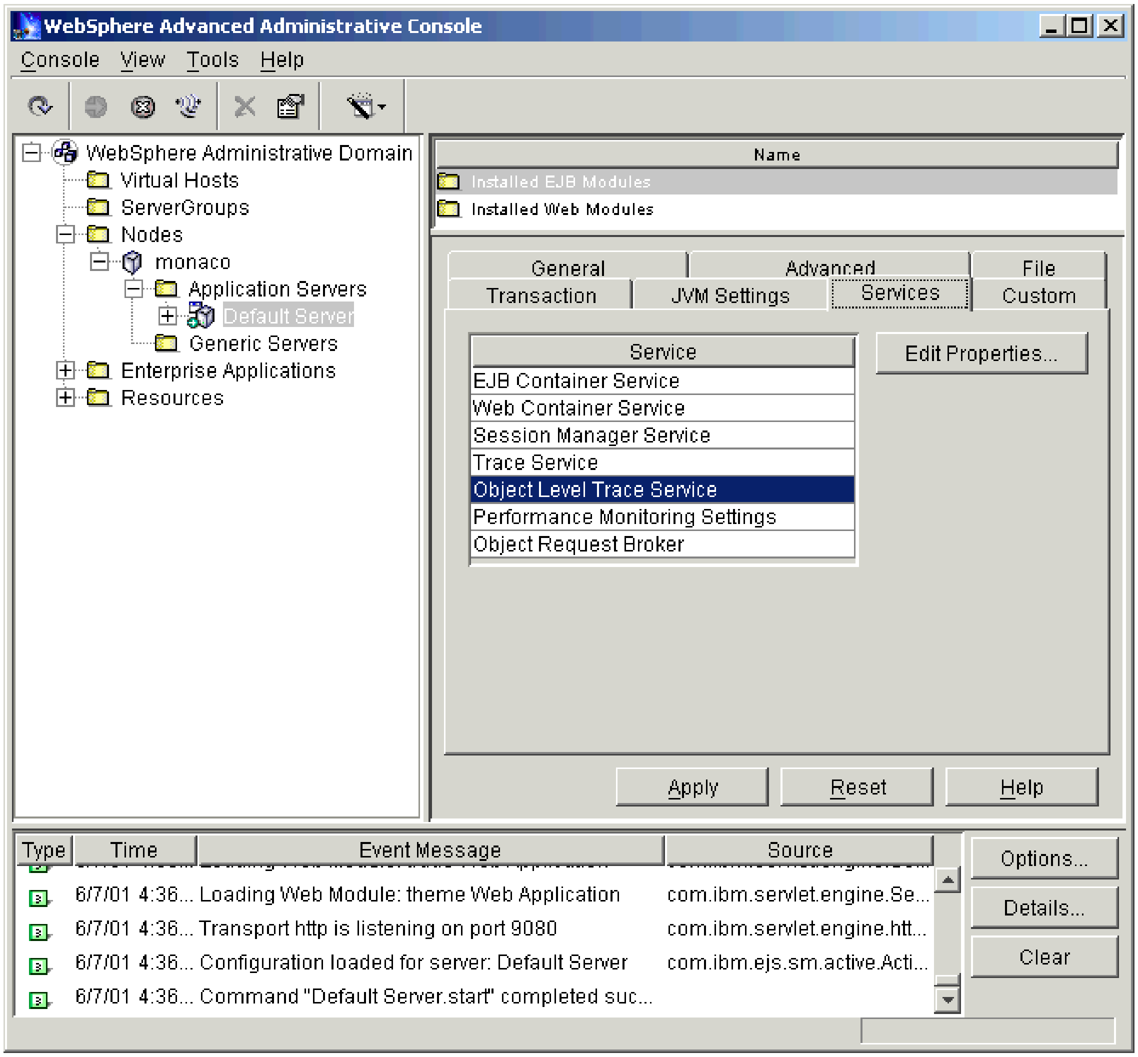
ADGU3124W: The IBM Distributed debugger only works when Object Level Tracing is enabled. Do you want OLT enabled now?

The entry in the **Advanced JVM Settings** dialog box **Source Path** field indicates to the Debugger where servlet source files, EJB source files, and JSP files can be found. Use semicolons to separate all entries in the Source Path. For this sample, add the following to the source path:

<WAS_ROOT>\installedApps\sampleApp.ear\examples.war;<WAS_ROOT>\installedApps\sampleApp.ear\examples.war\WEB-INF\classes

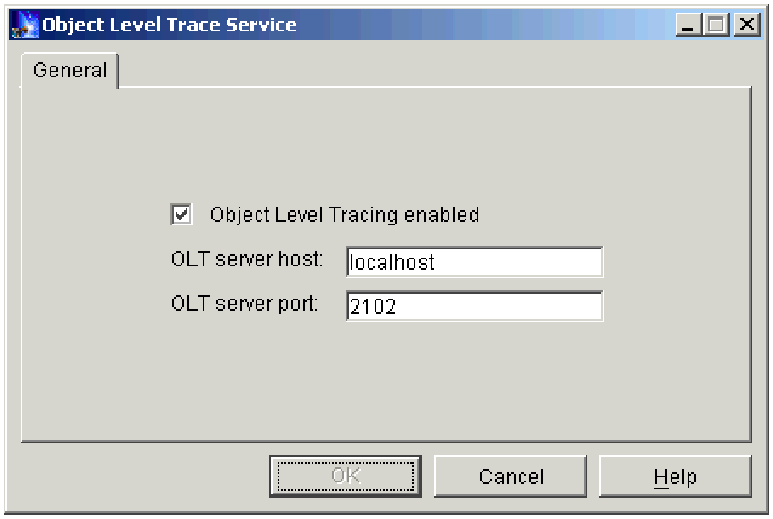
where WAS_ROOT is where WebSphere Application Server is installed (eg. c:\WebSphere\AppServer). Click **OK** when you are finished, and then click **Apply** in the **JVM Settings** tab to save your changes.

4. To see the OLT settings, select the **Services** tab:



```
6/7/01 4:36... Loading Web Module: theme Web Application     com.ibm.servlet.engine.Se...
6/7/01 4:36... Transport http is listening on port 9080      com.ibm.servlet.engine.htt...
6/7/01 4:36... Configuration loaded for server: Default Server  com.ibm.ejs.sm.active.Acti...
6/7/01 4:36... Command "Default Server.start" completed suc...
```

Select the Object Level Trace Service and then click **Edit Properties**.



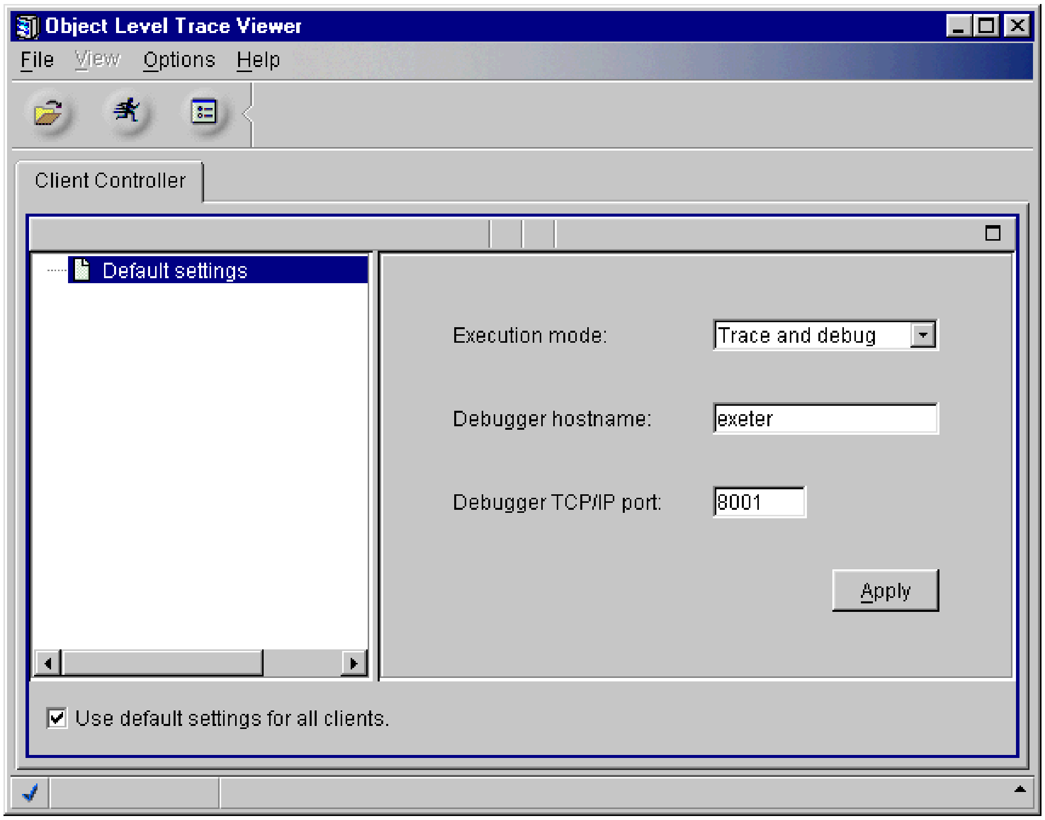
☑ Object Level Tracing enabled

OLT server host: localhost

OLT server port: 2102

In the **OLT server host** field, specify the OLT host by entering the name of the machine where you will be running the OLT tool. The default value is localhost. The **OLT server port** should not be changed from its default of 2102, unless there is a port conflict (see the Note below). If you do not need to change any of the OLT properties in the **Object Level Trace** dialog box, click **Cancel**. If you do change any of the OLT properties, click **OK** and then **Apply** to save your changes.

5. Start the **Default Server** by right-clicking on it and selecting **Start** from the pop-up menu.
6. Once you have started the application server, start OLT by issuing the olt command at a command prompt. You will then see the following screen:



Execution mode: Trace and debug

Debugger hostname: exeter

Debugger TCP/IP port: 8001

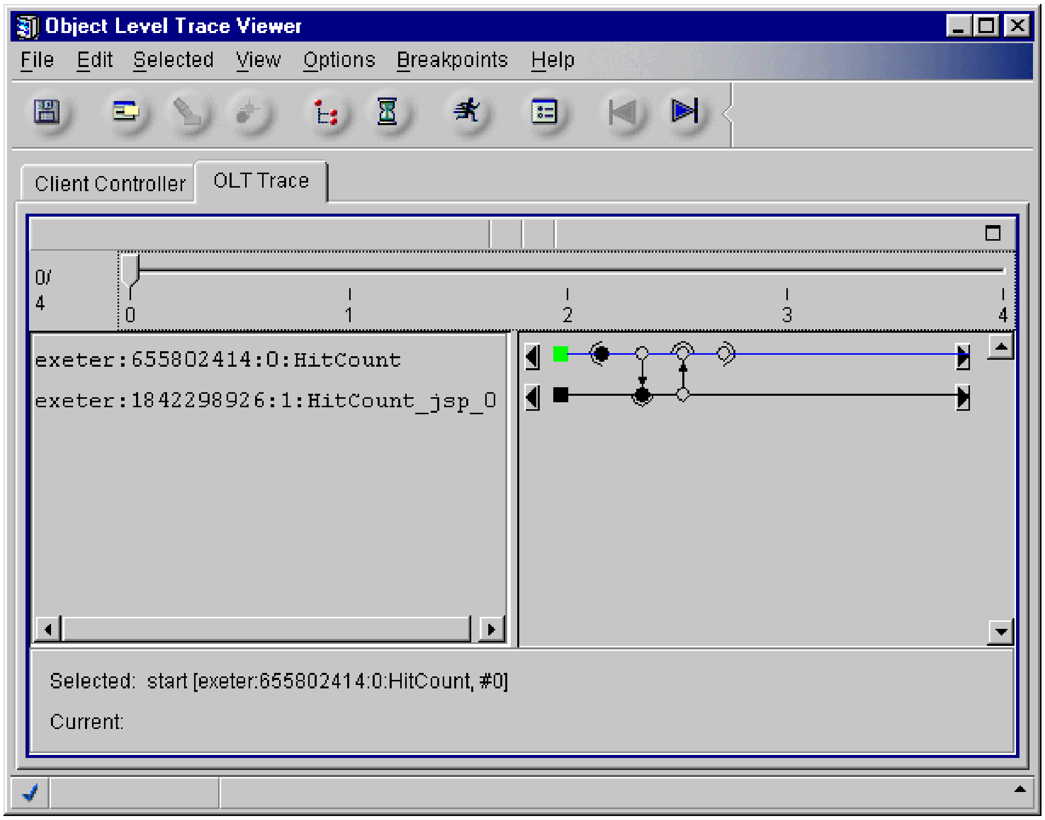
☑ Use default settings for all clients.

Ensure that the **Execution mode** is set to **Trace and debug**. If it is not, click the drop-down list and select it. In the **Debugger hostname** field, specify the name of the machine where the debugger tool is located. Assuming for this example that the Debugger is installed locally, enter the host name of the machine on which you are working. Keep the Debugger TCP/IP port set to 8001 (change it only if port 8001 is already in use on your machine). Click **Apply**.

7. Invoke the HitCount servlet by typing its URL in a Web browser:

   http://<your_machine>/webapp/examples/HitCount

   OLT will trace the initial calls to the HitCount servlet and JSP:


Object Level Trace Viewer
File Edit Selected View Options Breakpoints Help
Client Controller
OLT Trace
0/
4
0
1
2
3
4

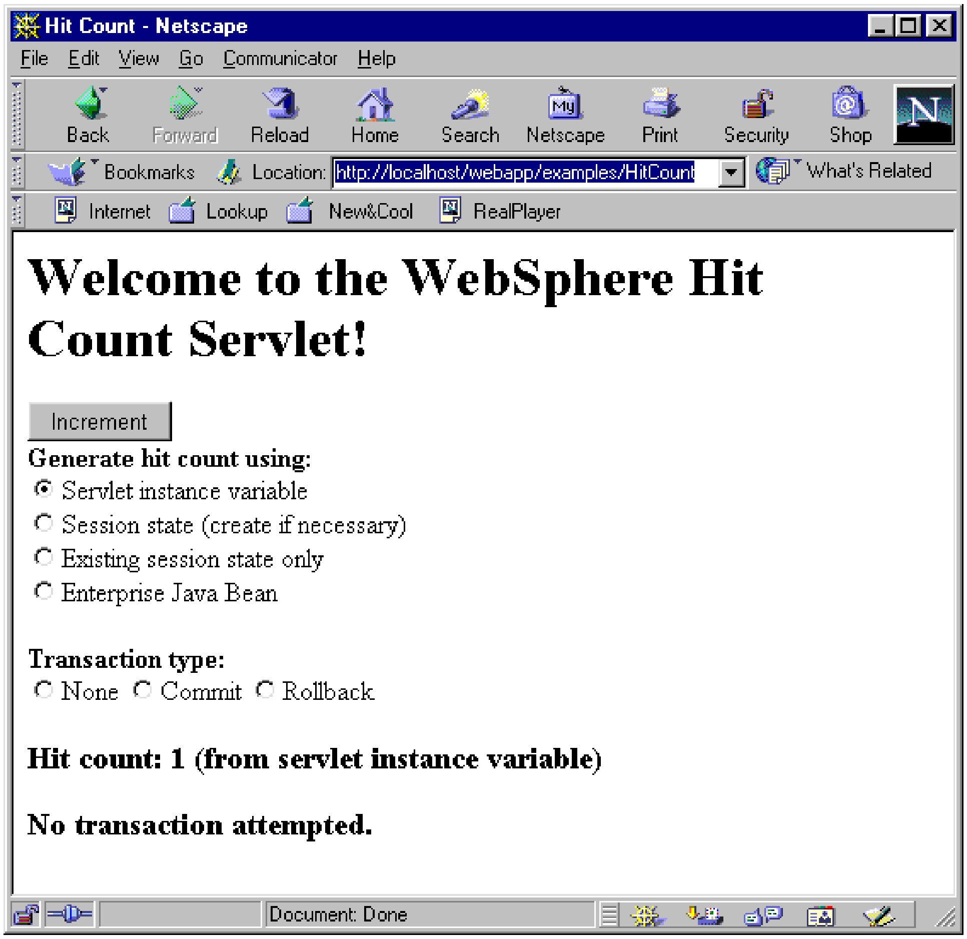
exeter:655802414:0:HitCount

exeter:1842298926:1:HitCount_jsp_0

Selected: start [exeter:655802414:0:HitCount, #0]

Current:

Your browser should now be displaying the following:



Location: http://localhost/webapp/examples/HitCount

Welcome to the WebSphere Hit Count Servlet!

Increment

Generate hit count using:

● Servlet instance variable
○ Session state (create if necessary)
○ Existing session state only
○ Enterprise Java Bean

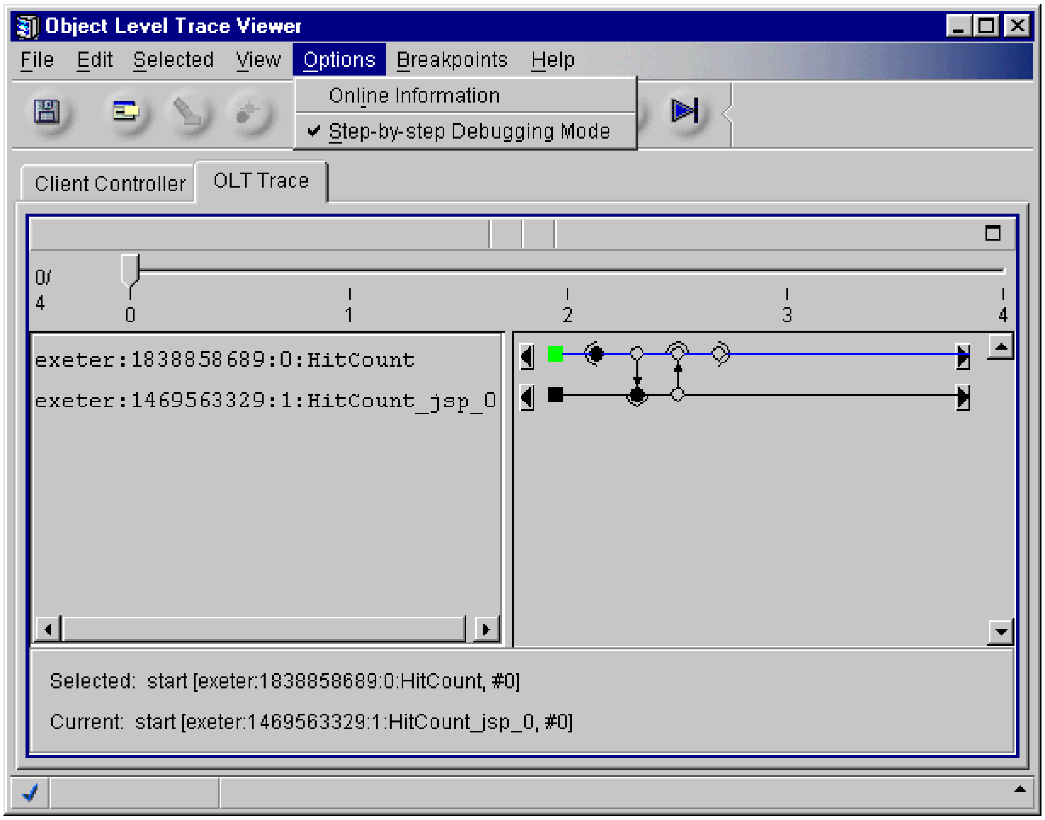
Transaction type:

○ None ○ Commit ○ Rollback

Hit count: 1 (from servlet instance variable)

No transaction attempted.

8. Let's enable step-by-step debugging now.


Object Level Trace Viewer
File Edit Selected View Options Breakpoints Help
Online Information
✓ Step-by-step Debugging Mode
Client Controller
OLT Trace
0/
4
0
1
2
3
4

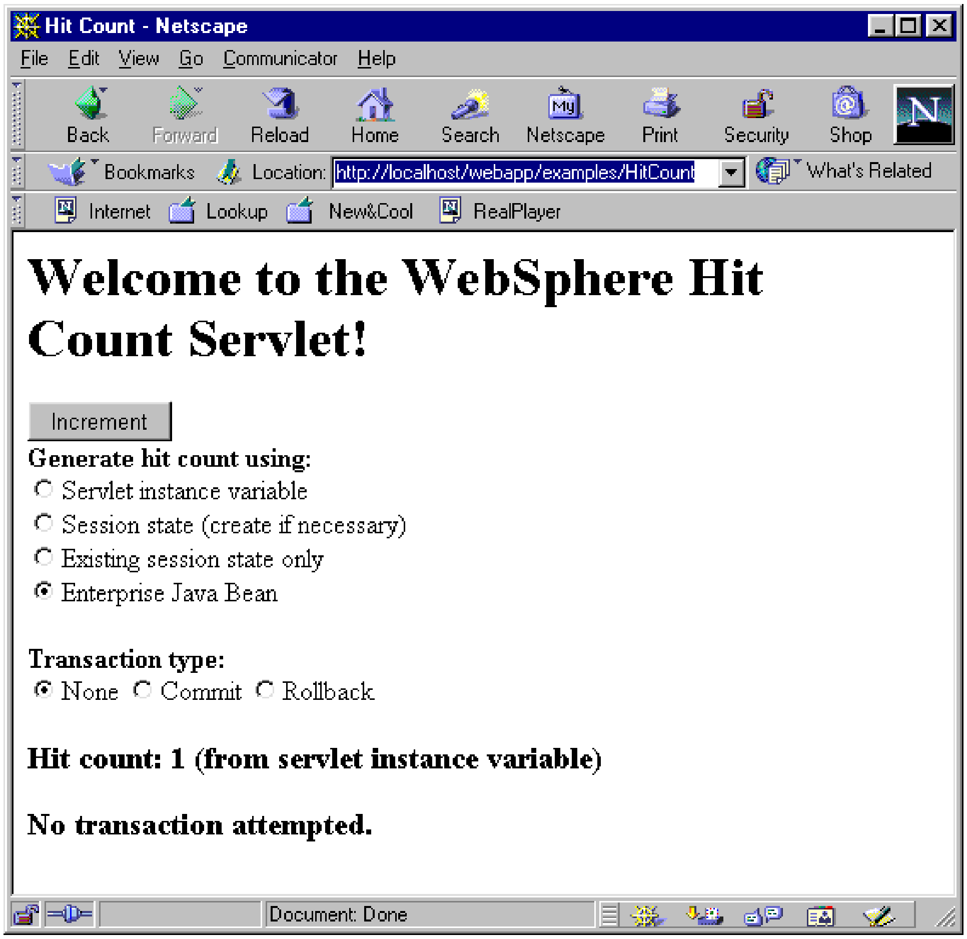
exeter:1838858689:0:HitCount

exeter:1469563329:1:HitCount_jsp_0

Selected: start [exeter:1838858689:0:HitCount, #0]

Current: start [exeter:1469563329:1:HitCount_jsp_0, #0]

9. In the browser, choose the **Enterprise Java Bean** radio button and Transaction Type **None**. Click the **Increment** button, and OLT will start tracing the events taking place:


Hit Count - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Shop
Bookmarks Location:
What's Related
Internet Lookup New&Cool RealPlayer
Document: Done

Location: http://localhost/webapp/examples/HitCount

Welcome to the WebSphere Hit Count Servlet!

Increment

Generate hit count using:

○ Servlet instance variable
○ Session state (create if necessary)
○ Existing session state only
● Enterprise Java Bean

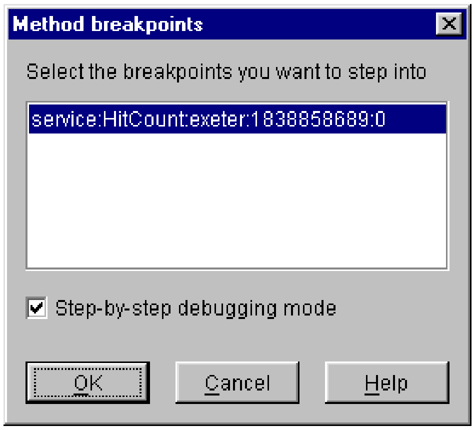
Transaction type:

● None ○ Commit ○ Rollback

Hit count: 1 (from servlet instance variable)

No transaction attempted.

10. Highlight the service method and Click **OK**. This will cause the debugger to attach and suspend in the first executable line of the HitCount service method:

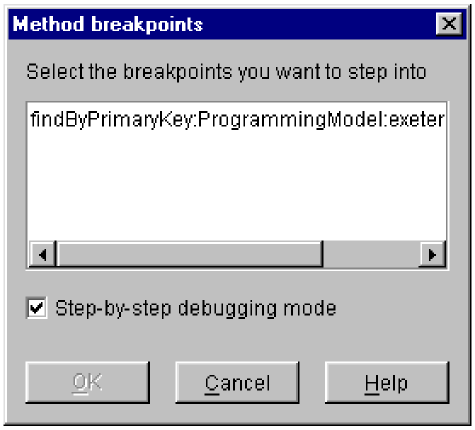


Select the breakpoints you want to step into

service:HitCount:exeter:1838858689:0

☑ Step-by-step debugging mode

11. You can now choose to step over methods, step into others, or run the program. Click **Run** in the Debugger toolbar. Because OLT is in Step-by-Step mode, when it encounters the next method that can be debugged, it will ask you again (via the **Method breakpoints** dialog box) if you would like to step into the method. Indicate that you would like to skip breaking into the enterprise bean. An enterprise bean is displayed as two distinct objects in the trace: the enterprise bean object itself, and an object called the ProgrammingModel. The ProgrammingModel is an EJB Factory instance that is beyond application logic and is not intended to be debugged.
12. To skip breaking into a method, click **Cancel** in the **Method breakpoints** dialog:

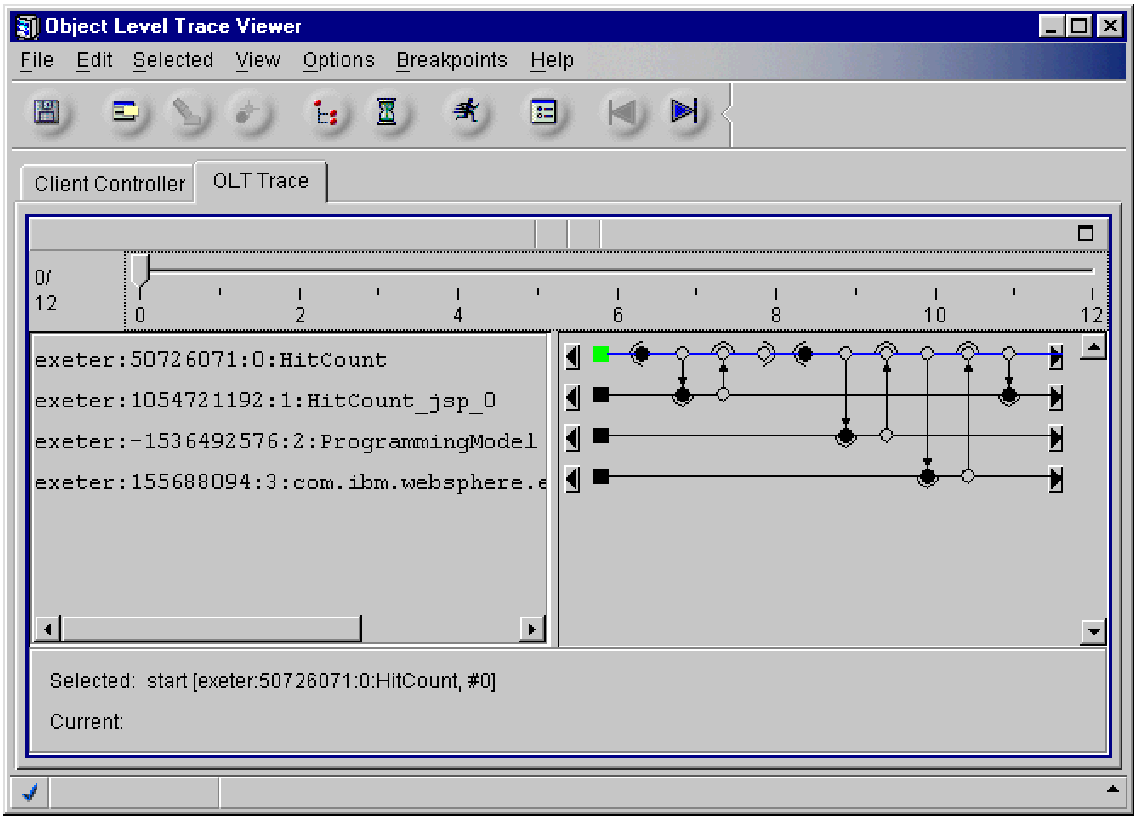


Select the breakpoints you want to step into

findByPrimaryKey:ProgrammingModel:exeter

☑ Step-by-step debugging mode

The EJB trace lines in the trace are IncBean and ProgrammingModel, as shown:

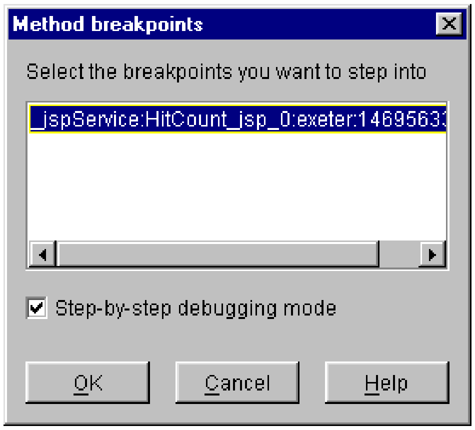


exeter:50726071:0:HitCount
exeter:1054721192:1:HitCount_jsp_0
exeter:-1536492576:2:ProgrammingModel
exeter:155688094:3:com.ibm.websphere.e

Selected: start [exeter:50726071:0:HitCount, #0]

Current:

13. To break into the JSP, step into the _jspService method:

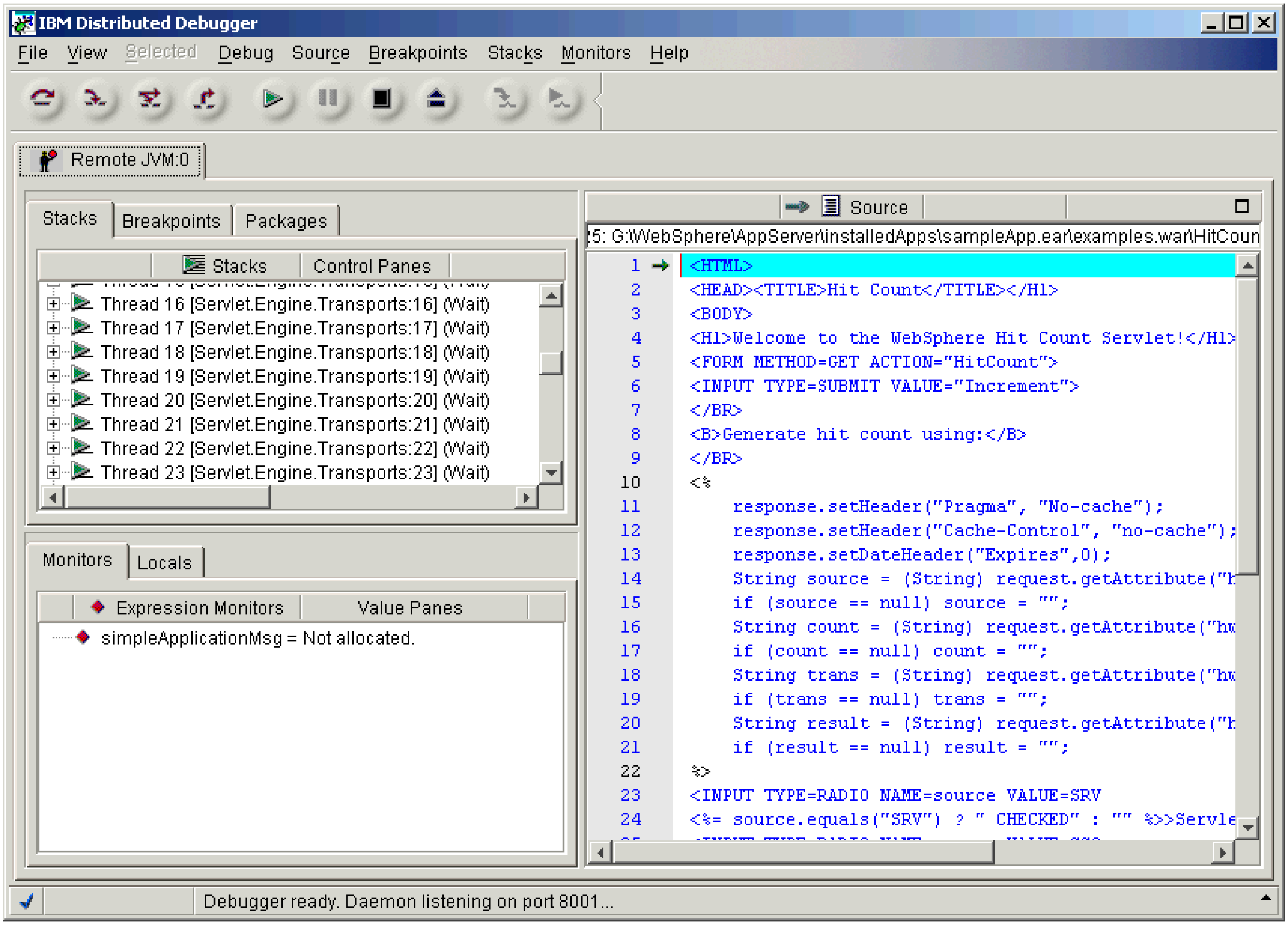


Select the breakpoints you want to step into

_jspService:HitCount_jsp_0:exeter:1469563

☑ Step-by-step debugging mode

The debugger will enter the JSP code, displaying the following:



```
Thread 16 [Servlet.Engine.Transports:16] (Wait)
Thread 17 [Servlet.Engine.Transports:17] (Wait)
Thread 18 [Servlet.Engine.Transports:18] (Wait)
Thread 19 [Servlet.Engine.Transports:19] (Wait)
Thread 20 [Servlet.Engine.Transports:20] (Wait)
Thread 21 [Servlet.Engine.Transports:21] (Wait)
Thread 22 [Servlet.Engine.Transports:22] (Wait)
Thread 23 [Servlet.Engine.Transports:23] (Wait)
```

```
simpleApplicationMsg = Not allocated.
```

```
:5: G:\WebSphere\AppServer\installedApps\sampleApp.ear\examples.war\HitCoun
```

```
<HTML>
<HEAD><TITLE>Hit Count</TITLE></H1>
<BODY>
<H1>Welcome to the WebSphere Hit Count Servlet!</H1>
<FORM METHOD=GET ACTION="HitCount">
<INPUT TYPE=SUBMIT VALUE="Increment">
</BR>
<B>Generate hit count using:</B>
</BR>
<%
    response.setHeader("Pragma", "No-cache");
    response.setHeader("Cache-Control", "no-cache");
    response.setDateHeader("Expires",0);
    String source = (String) request.getAttribute("h
    if (source == null) source = "";
    String count = (String) request.getAttribute("hw
    if (count == null) count = "";
    String trans = (String) request.getAttribute("hw
    if (trans == null) trans = "";
    String result = (String) request.getAttribute("h
    if (result == null) result = "";
%>
<INPUT TYPE=RADIO NAME=source VALUE=SRV
<%= source.equals("SRV") ? " CHECKED" : "" %>>Servle
```

```
Debugger ready. Daemon listening on port 8001...
```

14. You can now choose to step over, step debug, run, and so on. If you click **Run**, then HitCount should run to completion. Note that your browser may have timed out by this time. For information about increasing browser timeout, please see OLT Troubleshooting - Browsers in the OLT online documentation.

Note:

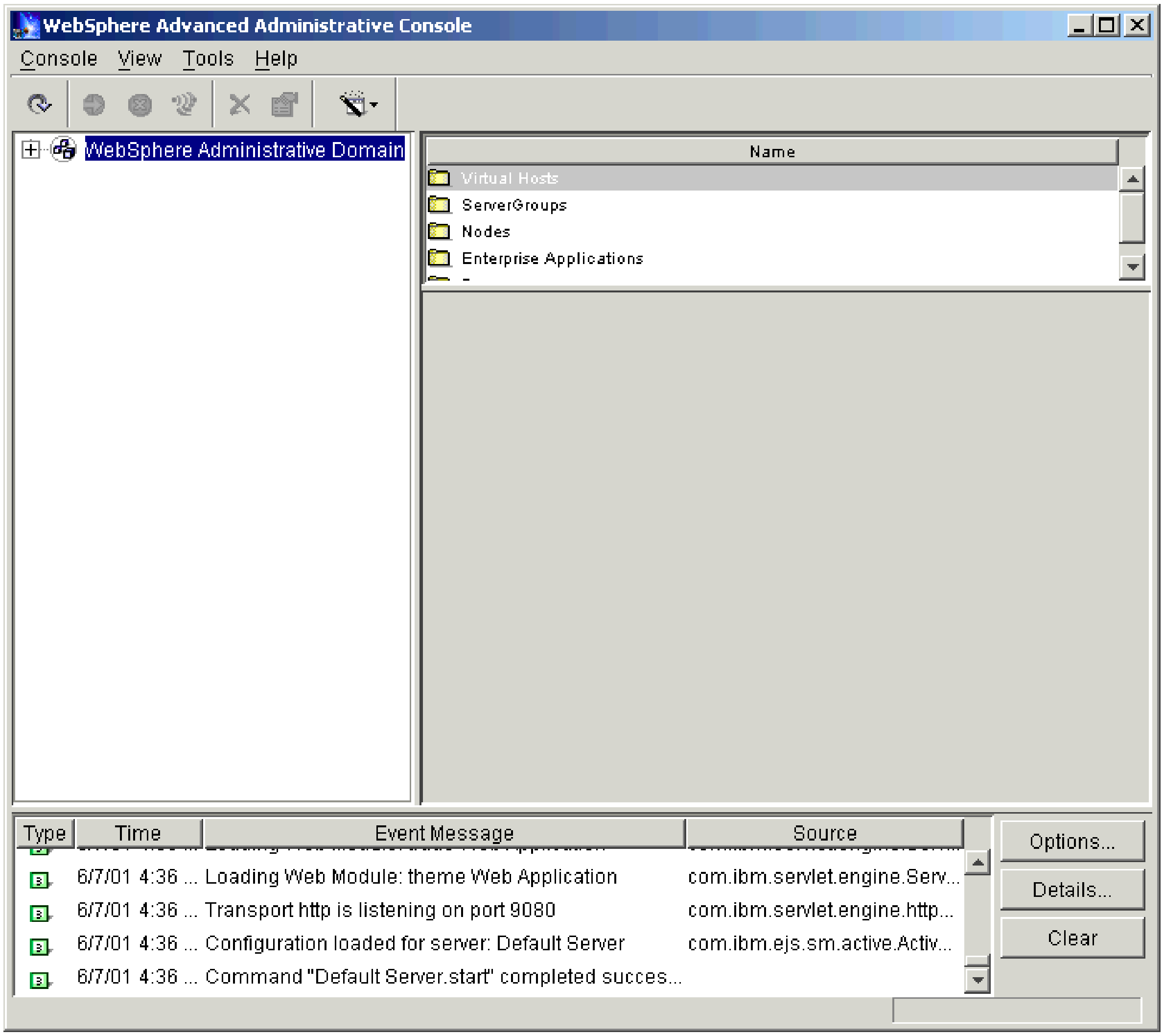
If you modify the Port in the **IBM Distributed Debugger and Object Level Trace settings** frame to a port other than 2102, you must start OLT and modify the **OLT Server TCP/IP port** accordingly. The OLT Server TCP/IP port is set in the **Browser Preferences** dialog box (accessed by selecting **File** > **Preferences** from the OLT menu bar and then selecting the **OLT** node). If there is a port conflict and you are unable to start OLT, go to the %userprofile%\DbgProf directory and modify the OLT_TRC_SRVAPP_PORT in the dertrenv.dat file accordingly.

StockQuote tutorial: Using Debugger and OLT

StockQuote is one of the samples that ships with IBM WebSphere Application Server. It is installed if, during WebSphere Application Server installation, you choose to install the Samples. This tutorial demonstrates how to use the IBM Distributed Debugger and Object Level Trace (OLT) on the StockQuote sample, referring to screen captures from the administrative console (WebSphere Administrative Console) used with the Application Server Advanced Single Server Edition.

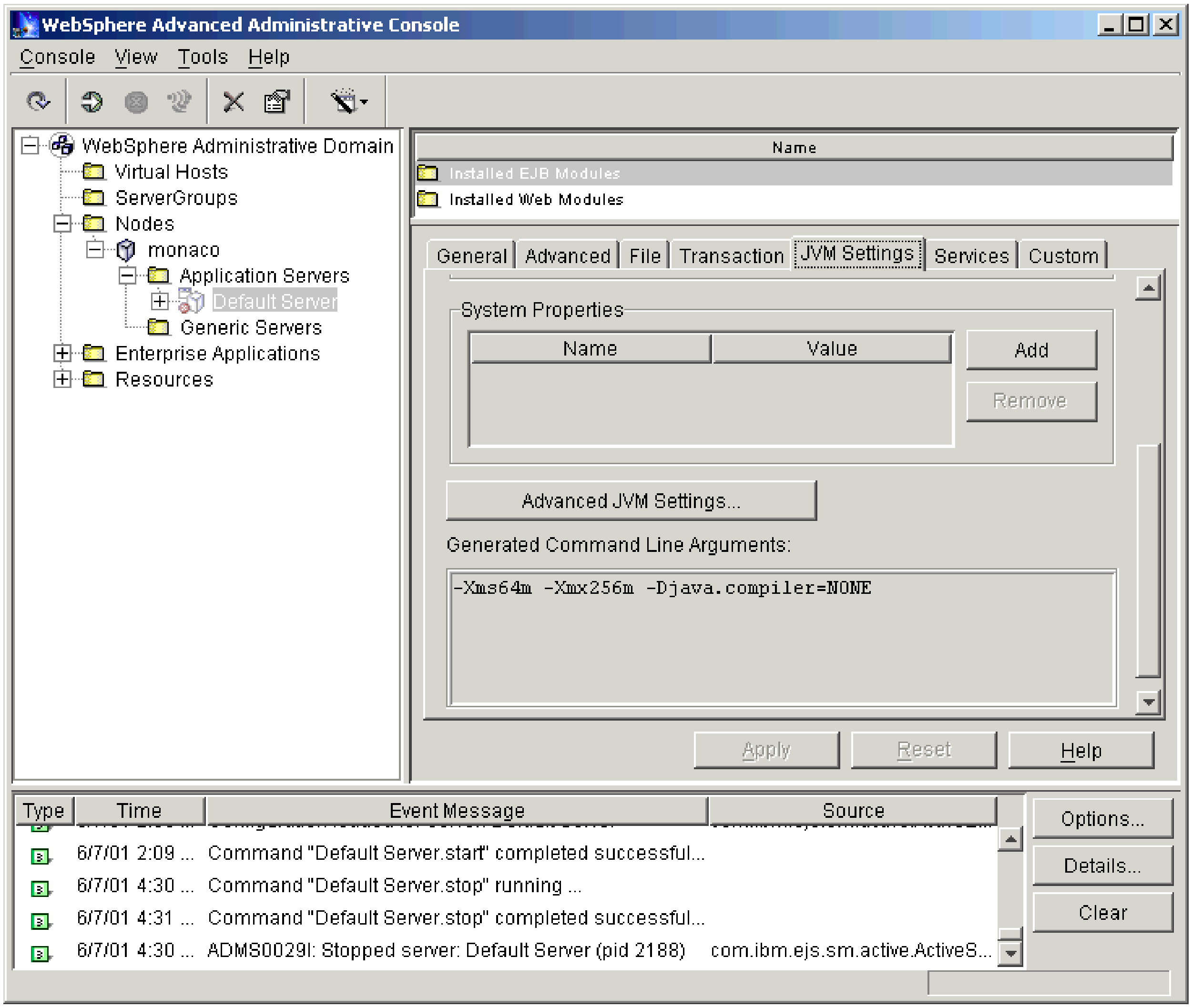
Let's begin.

1. Start the Administrative Console:



Type	Time	Event Message	Source
	6/7/01 4:36 ...	Loading Web Module: theme Web Application	com.ibm.servlet.engine.Serv...
	6/7/01 4:36 ...	Transport http is listening on port 9080	com.ibm.servlet.engine.http...
	6/7/01 4:36 ...	Configuration loaded for server: Default Server	com.ibm.ejs.sm.active.Activ...
	6/7/01 4:36 ...	Command "Default Server.start" completed succes...	

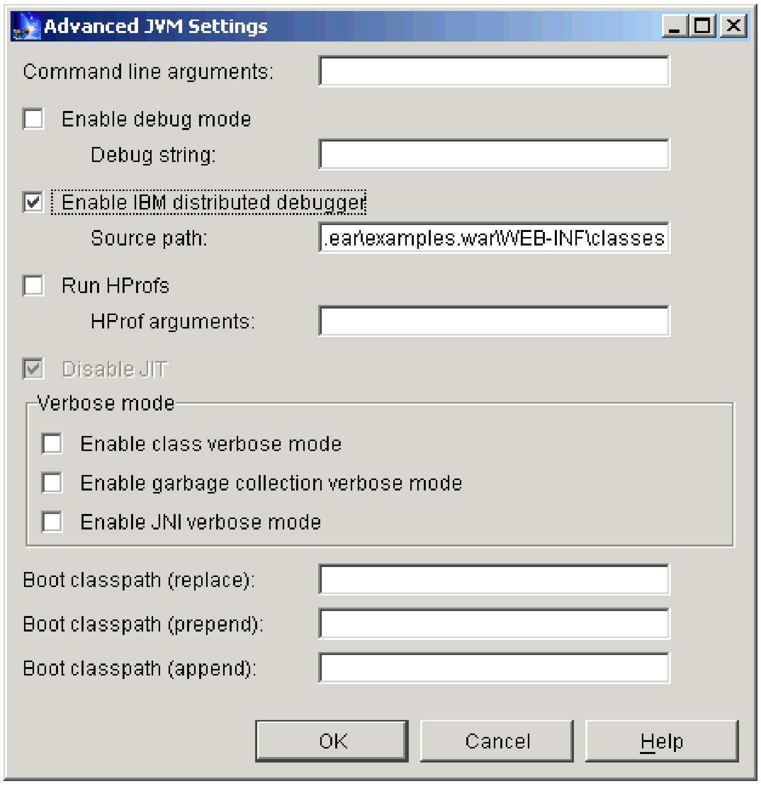
2. Select the **Default Server** for your node in the topology view on the left side of the console, and then select the **JVM Settings** tab in the right side of the console:



Generated Command Line Arguments: `-Xms64m -Xmx256m -Djava.compiler=NONE`

```
6/7/01 2:09 ...  Command "Default Server.start" completed successful...
6/7/01 4:30 ...  Command "Default Server.stop" running ...
6/7/01 4:31 ...  Command "Default Server.stop" completed successful...
6/7/01 4:30 ...  ADMS0029I: Stopped server: Default Server (pid 2188)   com.ibm.ejs.sm.active.ActiveS...
```

3. Click the **Advanced JVM Settings** pushbutton and then, in the **Advanced JVM Settings** dialog box, select the **Enable IBM Distributed Debugger** checkbox:



☐ Enable debug mode

☑ Enable IBM distributed debugger

Source path: .ear\examples.war\WEB-INF\classes

☐ Run HProfs

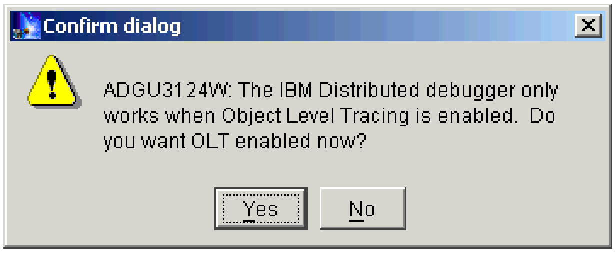
☑ Disable JIT

☐ Enable class verbose mode

☐ Enable garbage collection verbose mode

☐ Enable JNI verbose mode

You will be prompted with a **Confirm dialog** message, asking you if you would like to enable OLT. Click **Yes** to enable OLT:



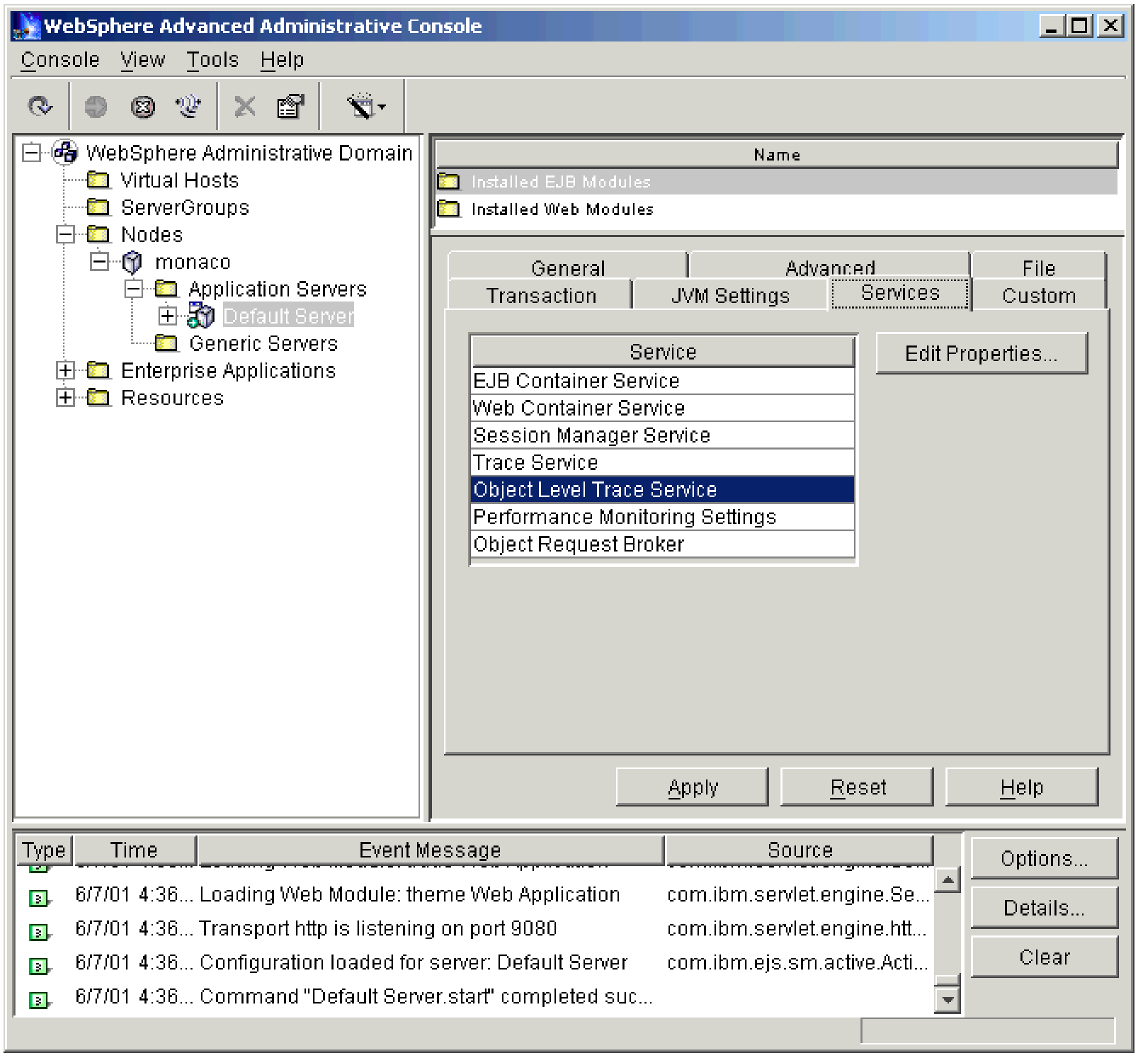
ADGU3124W: The IBM Distributed debugger only works when Object Level Tracing is enabled. Do you want OLT enabled now?

The entry in the **Advanced JVM Settings** dialog box **Source Path** field indicates to the Debugger where servlet source files, EJB source files, and JSP files can be found. Use semicolons to separate all entries in the Source Path. For this sample, add the following to the source path:

<WAS_ROOT>\installedApps\sampleApp.ear\examples.war;<WAS_ROOT>\installedApps\sampleApp.ear\examples.war\WEB-INF\classes

where WAS_ROOT is where WebSphere Application Server is installed (eg. c:\WebSphere\AppServer). Click **OK** when you are finished, and then click **Apply** in the **JVM Settings** tab to save your changes.

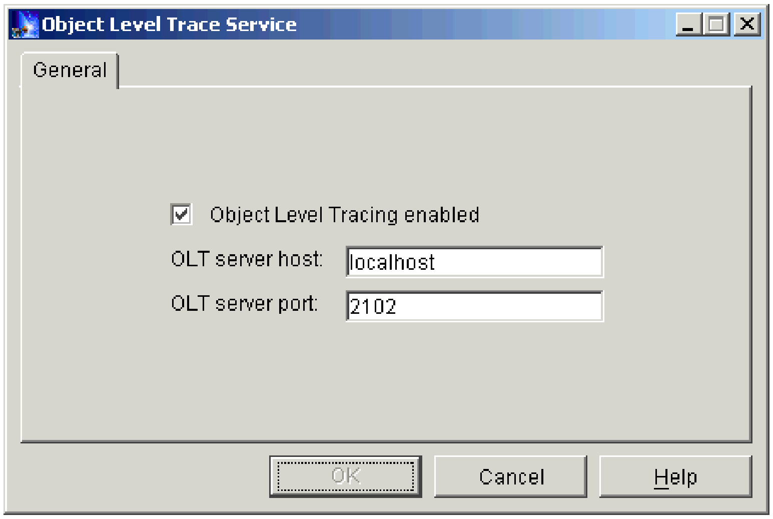
4. To see the OLT settings, select the **Services** tab:



Service
EJB Container Service
Web Container Service
Session Manager Service
Trace Service
Object Level Trace Service
Performance Monitoring Settings
Object Request Broker

```
6/7/01 4:36... Loading Web Module: theme Web Application    com.ibm.servlet.engine.Se...
6/7/01 4:36... Transport http is listening on port 9080     com.ibm.servlet.engine.htt...
6/7/01 4:36... Configuration loaded for server: Default Server    com.ibm.ejs.sm.active.Acti...
6/7/01 4:36... Command "Default Server.start" completed suc...
```

Select the Object Level Trace Service and then click **Edit Properties**.



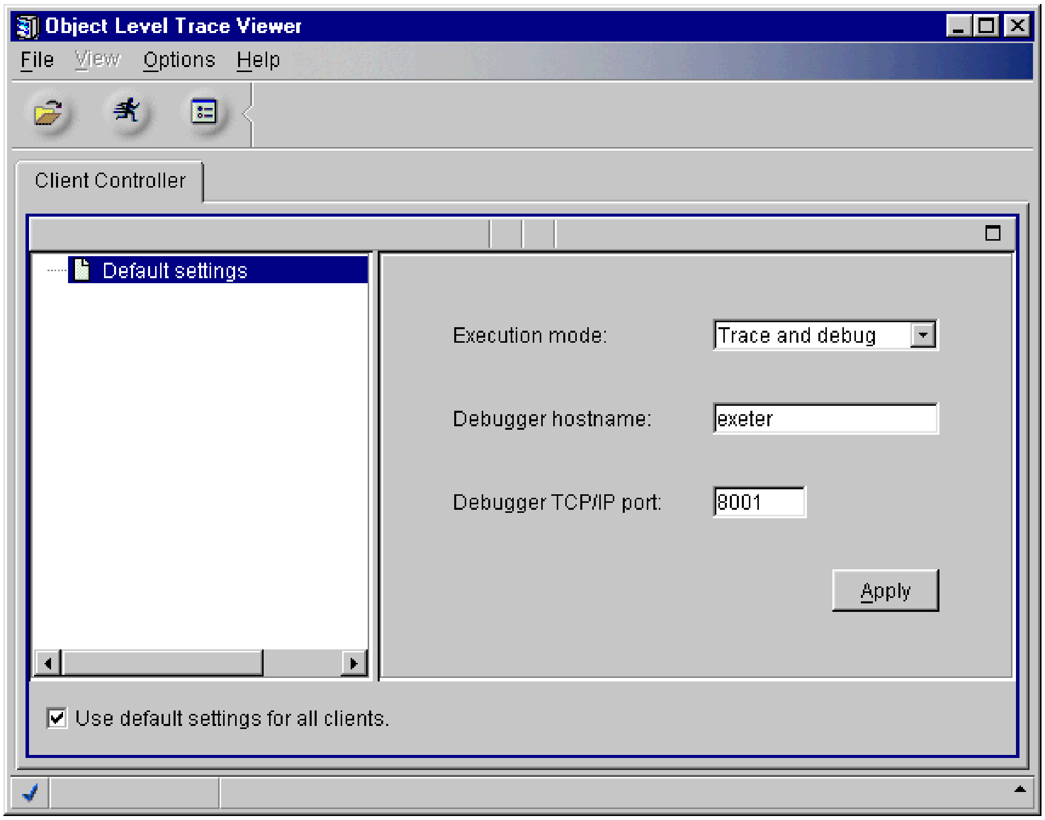
☑ Object Level Tracing enabled

OLT server host: localhost

OLT server port: 2102

In the **OLT server host** field, specify the OLT host by entering the name of the machine where you will be running the OLT tool. The default value is localhost. The **OLT server port** should not be changed from its default of 2102, unless there is a port conflict (see the Note below). If you do not need to change any of the OLT properties in the **Object Level Trace** dialog box, click **Cancel**. If you do change any of the OLT properties, click **OK** and then **Apply** to save your changes.

5. Start the **Default Server** by right-clicking on it and selecting **Start** from the pop-up menu.
6. Once you have started the application server, start OLT by issuing the olt command at a command prompt. You will then see the following screen:



Execution mode: Trace and debug

Debugger hostname: exeter

Debugger TCP/IP port: 8001

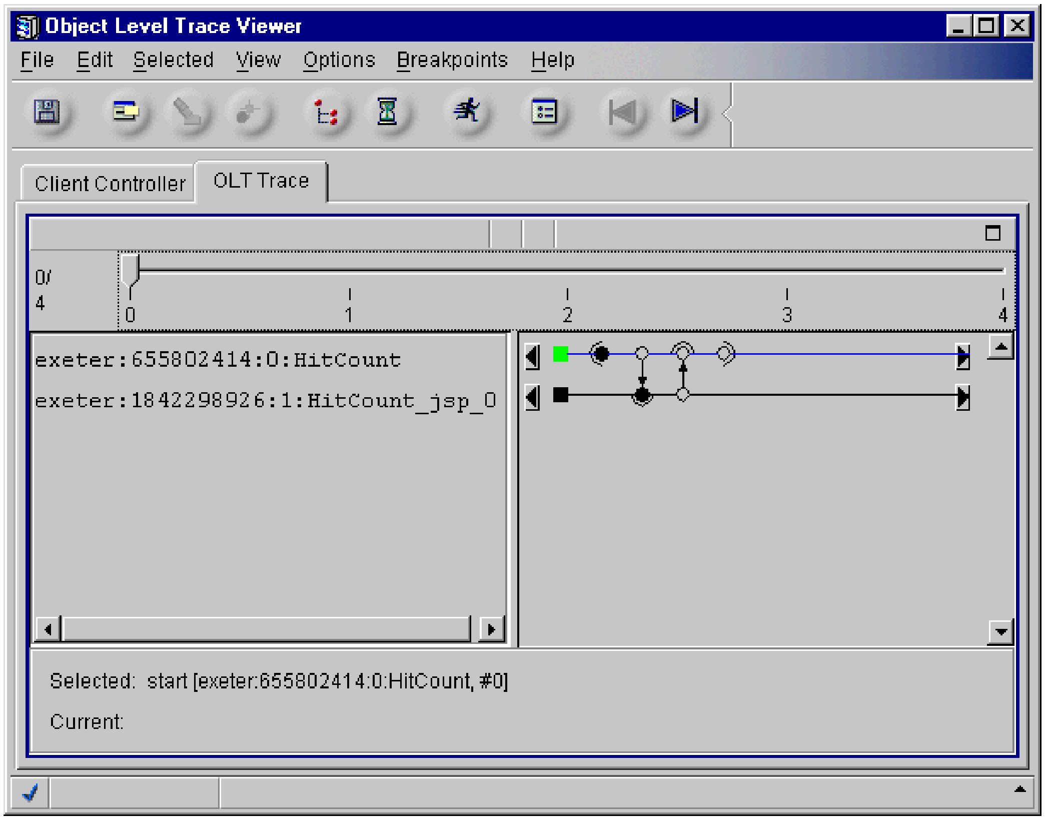
☑ Use default settings for all clients.

Ensure that the **Execution mode** is set to **Trace and debug**. If it is not, click the drop-down list and select it. In the **Debugger hostname** field, specify the name of the machine where the debugger tool is located. Assuming for this example that the Debugger is installed locally, enter the host name of the machine on which you are working. Keep the Debugger TCP/IP port set to 8001 (change it only if port 8001 is already in use on your machine). Click **Apply**.

7. Invoke the HitCount servlet by typing its URL in a Web browser:

http://<your_machine>/webapp/examples/HitCount

OLT will trace the initial calls to the HitCount servlet and JSP:


Object Level Trace Viewer
File Edit Selected View Options Breakpoints Help
Client Controller
OLT Trace
0/
4
0
1
2
3
4

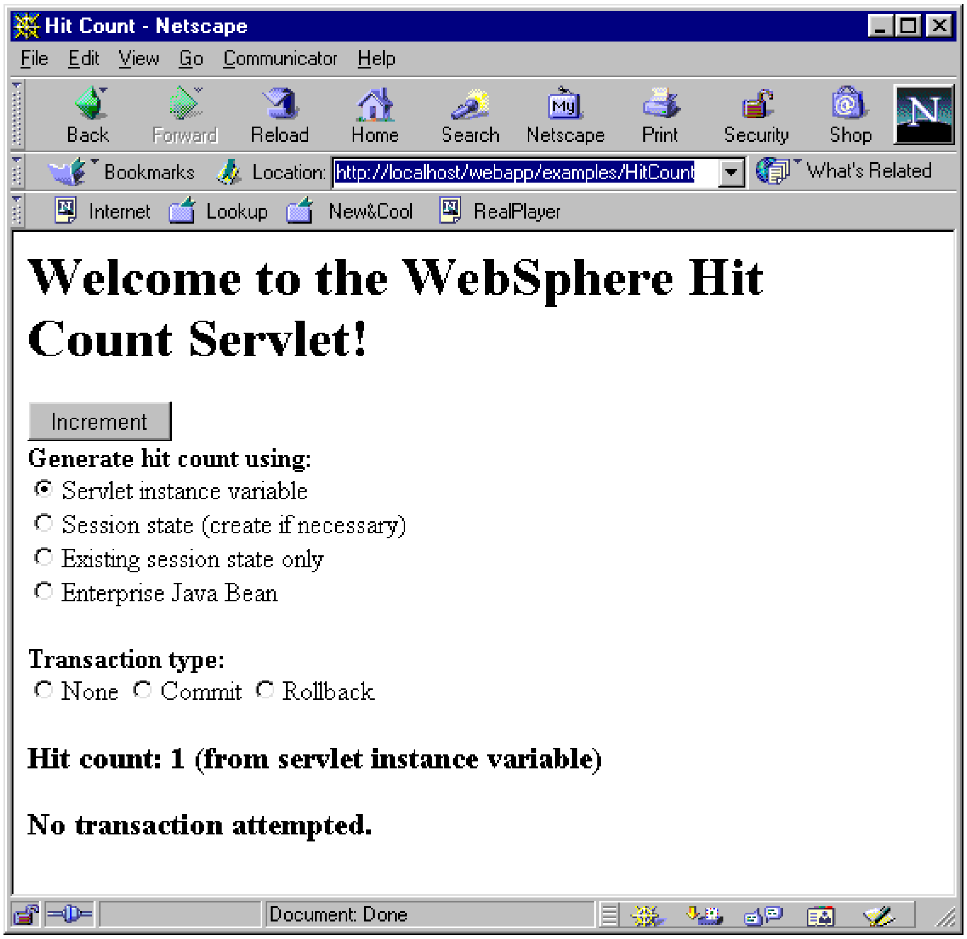
exeter:655802414:0:HitCount

exeter:1842298926:1:HitCount_jsp_0

Selected: start [exeter:655802414:0:HitCount, #0]

Current:

Your browser should now be displaying the following:



Location: http://localhost/webapp/examples/HitCount

Welcome to the WebSphere Hit Count Servlet!

Increment

Generate hit count using:

● Servlet instance variable
○ Session state (create if necessary)
○ Existing session state only
○ Enterprise Java Bean

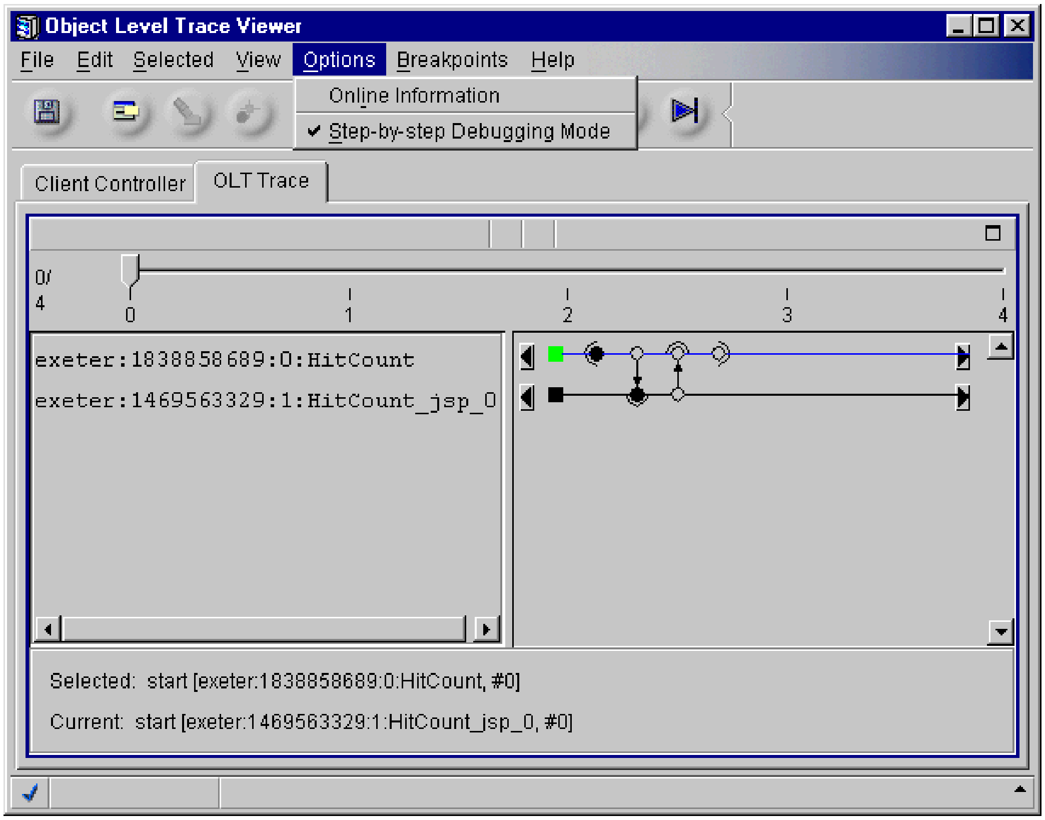
Transaction type:

○ None ○ Commit ○ Rollback

Hit count: 1 (from servlet instance variable)

No transaction attempted.

8. Let's enable step-by-step debugging now.


Object Level Trace Viewer
File Edit Selected View Options Breakpoints Help
Online Information
✔ Step-by-step Debugging Mode
Client Controller
OLT Trace
0/
4
0
1
2
3
4

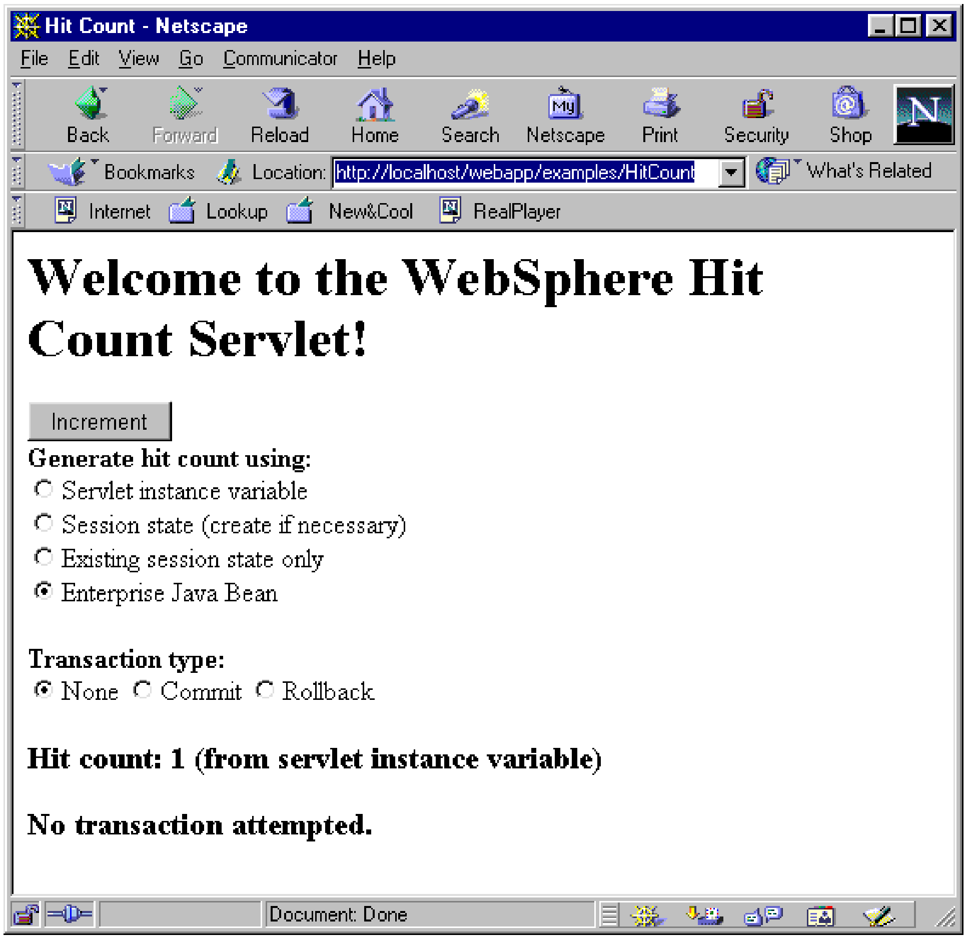
exeter:1838858689:0:HitCount

exeter:1469563329:1:HitCount_jsp_0

Selected: start [exeter:1838858689:0:HitCount, #0]

Current: start [exeter:1469563329:1:HitCount_jsp_0, #0]

9. In the browser, choose the **Enterprise Java Bean** radio button and Transaction Type **None**. Click the **Increment** button, and OLT will start tracing the events taking place:



Location: http://localhost/webapp/examples/HitCount

Welcome to the WebSphere Hit Count Servlet!

Generate hit count using:

○ Servlet instance variable
○ Session state (create if necessary)
○ Existing session state only
● Enterprise Java Bean

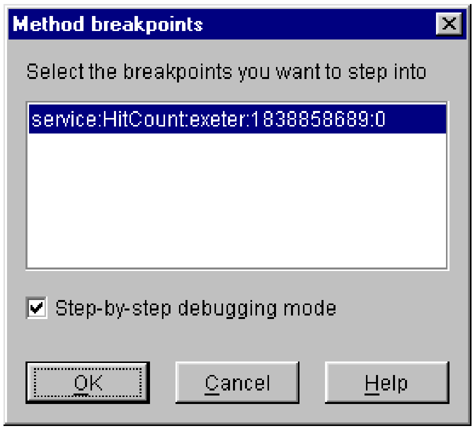
Transaction type:

● None ○ Commit ○ Rollback

Hit count: 1 (from servlet instance variable)

No transaction attempted.

10. Highlight the service method and Click **OK**. This will cause the debugger to attach and suspend in the first executable line of the HitCount service method:

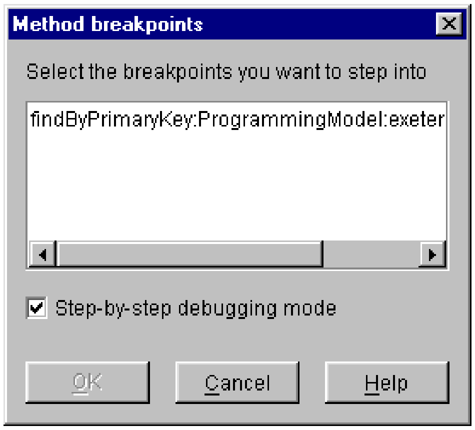


service:HitCount:exeter:1838858689:0

☑ Step-by-step debugging mode

11. You can now choose to step over methods, step into others, or run the program. Click **Run** in the Debugger toolbar. Because OLT is in Step-by-Step mode, when it encounters the next method that can be debugged, it will ask you again (via the **Method breakpoints** dialog box) if you would like to step into the method. Indicate that you would like to skip breaking into the enterprise bean. An enterprise bean is displayed as two distinct objects in the trace: the enterprise bean object itself, and an object called the ProgrammingModel. The ProgrammingModel is an EJB Factory instance that is beyond application logic and is not intended to be debugged.

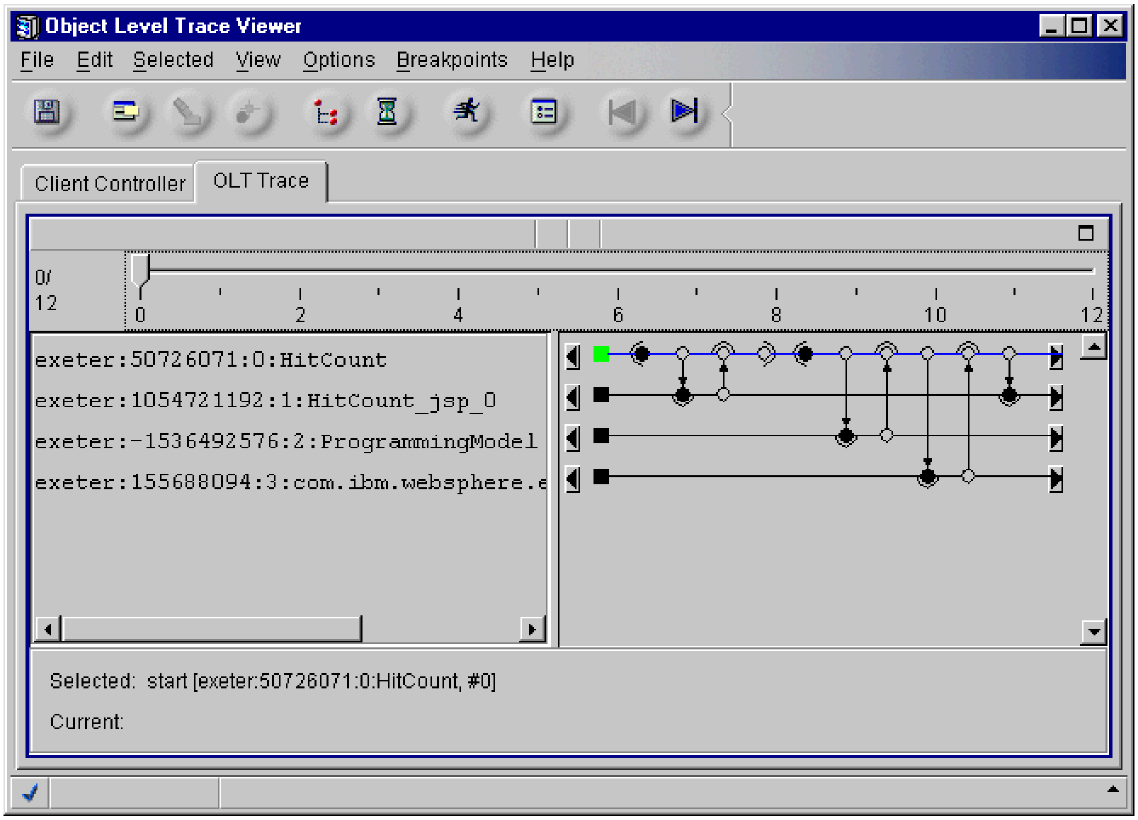
12. To skip breaking into a method, click **Cancel** in the **Method breakpoints** dialog:



findByPrimaryKey:ProgrammingModel:exeter

☑ Step-by-step debugging mode

The EJB trace lines in the trace are IncBean and ProgrammingModel, as shown:

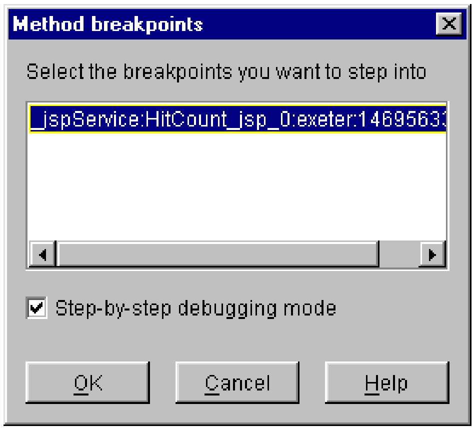


exeter:50726071:0:HitCount
exeter:1054721192:1:HitCount_jsp_0
exeter:-1536492576:2:ProgrammingModel
exeter:155688094:3:com.ibm.websphere.e

Selected: start [exeter:50726071:0:HitCount, #0]

Current:

13. To break into the JSP, step into the _jspService method:

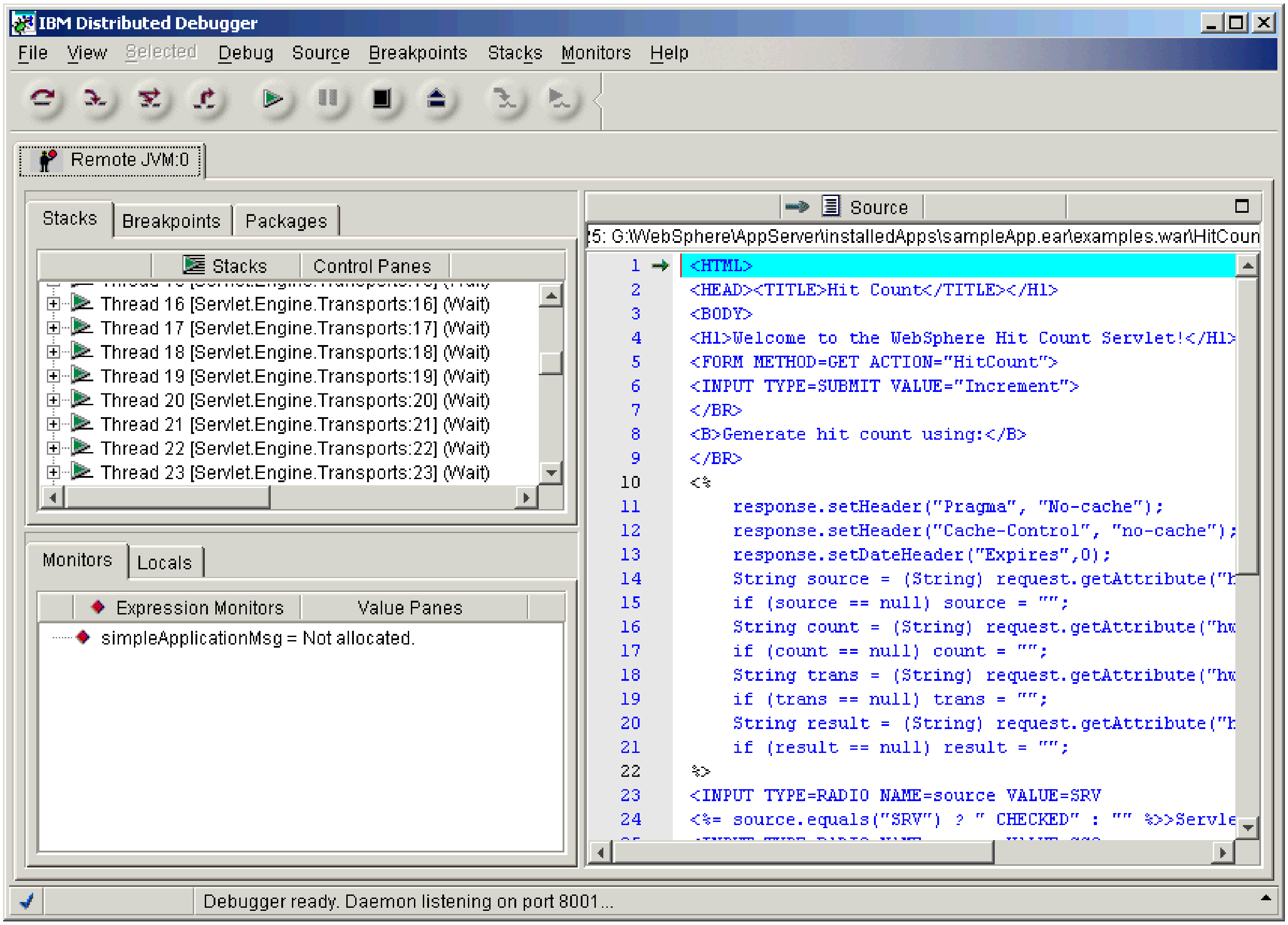


Select the breakpoints you want to step into

_jspService:HitCount_jsp_0:exeter:14695633

☑ Step-by-step debugging mode

The debugger will enter the JSP code, displaying the following:



Thread 16 [Servlet.Engine.Transports:16] (Wait)
Thread 17 [Servlet.Engine.Transports:17] (Wait)
Thread 18 [Servlet.Engine.Transports:18] (Wait)
Thread 19 [Servlet.Engine.Transports:19] (Wait)
Thread 20 [Servlet.Engine.Transports:20] (Wait)
Thread 21 [Servlet.Engine.Transports:21] (Wait)
Thread 22 [Servlet.Engine.Transports:22] (Wait)
Thread 23 [Servlet.Engine.Transports:23] (Wait)

simpleApplicationMsg = Not allocated.

5: G:\WebSphere\AppServer\installedApps\sampleApp.ear\examples.war\HitCoun

```
<HTML>
<HEAD><TITLE>Hit Count</TITLE></H1>
<BODY>
<H1>Welcome to the WebSphere Hit Count Servlet!</H1>
<FORM METHOD=GET ACTION="HitCount">
<INPUT TYPE=SUBMIT VALUE="Increment">
</BR>
<B>Generate hit count using:</B>
</BR>
<%
    response.setHeader("Pragma", "No-cache");
    response.setHeader("Cache-Control", "no-cache");
    response.setDateHeader("Expires",0);
    String source = (String) request.getAttribute("h
    if (source == null) source = "";
    String count = (String) request.getAttribute("hw
    if (count == null) count = "";
    String trans = (String) request.getAttribute("hw
    if (trans == null) trans = "";
    String result = (String) request.getAttribute("h
    if (result == null) result = "";
%>
<INPUT TYPE=RADIO NAME=source VALUE=SRV
<%= source.equals("SRV") ? " CHECKED" : "" %>>Servle
```

Debugger ready. Daemon listening on port 8001...

14. You can now choose to step over, step debug, run, and so on. If you click **Run**, then HitCount should run to completion. Note that your browser may have timed out by this time. For information about increasing browser timeout, please see OLT Troubleshooting - Browsers in the OLT online documentation.

Note:

If you modify the Port in the **IBM Distributed Debugger and Object Level Trace settings** frame to a port other than 2102, you must start OLT and modify the **OLT Server TCP/IP port** accordingly. The OLT Server TCP/IP port is set in the **Browser Preferences** dialog box (accessed by selecting **File** > **Preferences** from the OLT menu bar and then selecting the **OLT** node). If there is a port conflict and you are unable to start OLT, go to the %userprofile%\DbgProf directory and modify the OLT_TRC_SRVAPP_PORT in the dertrenv.dat file accordingly.

6.10: Backing up and restoring administrative configurations

WebSphere Application Server is capable of representing its administrativeconfigurations as XML files. The Java administrative console provides menu options for importing and exporting XML representations of the entire administrative domain. You can use it or the XMLConfig and WSCP administrative clientsto regularly export full configurations as a backup measure.

It also recommended that you keep backup copies of your applicationfiles deployed on IBM WebSphere Application Server, as well as backingup the administrative database itself.