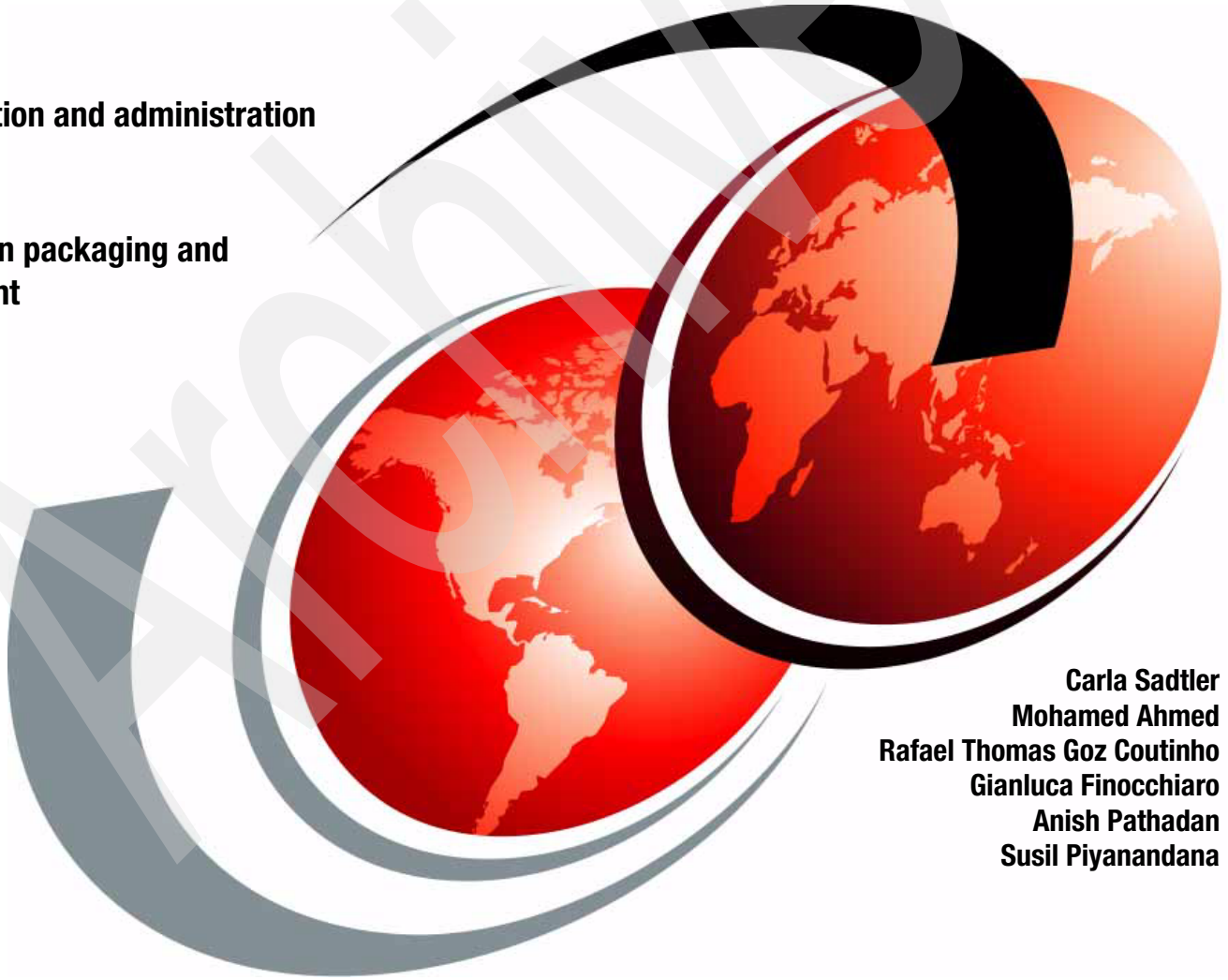
WebSphere. software

IBM

WebSphere Application Server Community Edition 2.0 User Guide

- Installation and migration
- Configuration and administration
- Application packaging and deployment



Carla Sadtler
Mohamed Ahmed
Rafael Thomas Goz Coutinho
Gianluca Finocchiaro
Anish Pathadan
Susil Piyanandana

Redbooks

ibm.com/redbooks

International Technical Support Organization

WebSphere Application Server Community Edition 2.0 User Guide

April 2008

SG24-7585-00

Note: Before using this information and the product it supports, read the information in “Notices” on page ix.

First Edition (April 2008)

This edition applies to WebSphere Application Server Community Edition version 2.0.0.0 and later.

Contents

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Trademarks

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

AIX®	IBM®	Redbooks®
Cloudscape®	i5/OS®	Redbooks (logo)  ®
DB2®	pSeries®	Tivoli®
developerWorks®	Rational®	WebSphere®

The following terms are trademarks of other companies:

Oracle, JD Edwards, PeopleSoft, Siebel, and TopLink are registered trademarks of Oracle Corporation and/or its affiliates.

AMD, the AMD Arrow logo, and combinations thereof, are trademarks of Advanced Micro Devices, Inc.

Adobe, and Portable Document Format (PDF) are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, other countries, or both.

EJB, Java, Java runtime environment, JavaMail, JDBC, JDK, JMX, JRE, JSP, JVM, J2EE, J2SE, MySQL, Solaris, Sun, Sun Java, and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Active Directory, Internet Explorer, Microsoft, SQL Server, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Intel, Intel logo, Intel Inside logo, and Intel Centrino logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

Preface

This IBM® Redbooks® publication takes you through the basics of using WebSphere® Application Server Community Edition V2 to run applications. It includes information about planning and installing the Community Edition server, configuring the environment, preparing applications for deployment, and managing deployed applications. This book also contains pointers to information that can help you with more advanced topics and to find updated information on using the Community Edition.

The target audience is primarily administrators of Community Edition, but this book is also useful for developers who are packaging and deploying applications to Community Edition.

The team that wrote this book

This book was produced by a team of specialists from around the world working at the International Technical Support Organization, Raleigh Center.

Carla Sadtler is an IT Specialist at the ITSO, Raleigh Center. She writes extensively about the WebSphere and Patterns for e-business areas. Before joining the ITSO in 1985, Carla worked in the Raleigh branch office as a Program Support Representative. She has a degree in Mathematics from the University of North Carolina at Greensboro.

Mohamed Ahmed is a Software Engineer in Egypt. He has four plus years of experience in the software development field. He graduated at faculty of Computers and Information, Cairo University in Cairo, in Egypt. He is interested in Middleware development, especially open source Middleware. In his personal time, he contributes to the Apache OpenEJB open source project, which is the Enterprise Java™ Bean (EJB™) container of the Community Edition and Geronimo.

Rafael Thomas Goz Coutinho is a Computer Engineer who graduated at Unicamp in Brazil. He has five years of experience in software development. He has worked at IBM for three years. His areas of expertise include Java enterprise applications development and architecture. He has written extensively on application development for Community Edition.

Gianluca Finocchiaro is an IT Specialist in Italy at Wireless Innovation Center of Catania in Sicily. He has worked at IBM for six years. He has more than eight years of experience in Java development. He has a degree in Computer Engineering from the University of Catania. His areas of expertise include wireless and field force automation solutions, JME, JEE, IBM WebSphere Application Server Community Edition, Standard Widget Toolkit, Eclipse, and Rational® Application Developer v7.0.

Anish Pathadan is a Software Engineer at the IBM India Software Labs. He has worked at IBM for the past year. He has more than three years of experience in Java technologies. He has a degree in Computer Science and Engineering from Calicut University. His areas of expertise include WebSphere Application Server Community Edition, ActiveMQ, Java EE, and messaging.

Susil Piyanandana is a Lead Integration Engineer in Australia and New Zealand Banking Group Limited in Australia. He has 23 years of experience in the data processing and information technology fields. He has degrees in B.Sc (Science), B.Sc. (Mathematics) from University of Peradeniya in Sri Lanka and M.Sc. (Computer Science) from the Asian Institute

of Technology in Bangkok, Thailand. His areas of expertise include Designing J2EE™ Infrastructures and building automation tools for WebSphere Application Server management.

Figure 0-1 Authors (left to right): Susil Piyanandana, Rafael Thomas Goz Coutinho, Carla Sadtler, Mohamed Ahmed, Gianluca Finocchiaro, Anish Pathadan

Thanks to the following people for their contributions to this project:

Ron Staerker
IBM US

Steven Calello
IBM US

Paul Craton
IBM UK

Become a published author

Join us for a two-to-six week residency program! Help write a book dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You will have the opportunity to team with IBM technical professionals, Business Partners, and Clients.

Your efforts will help increase product acceptance and customer satisfaction. As a bonus, you will develop a network of contacts in IBM development labs, and increase your productivity and marketability.

Find out more about the residency program, browse the residency index, and apply online at:

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

We want our books to be as helpful as possible. Send us your comments about this book or other IBM Redbooks in one of the following ways:

- Use the online **Contact us** review Redbooks form found at:

  ibm.com/redbooks

- Send your comments in an e-mail to:

  redbooks@us.ibm.com

- Mail your comments to:

  IBM Corporation, International Technical Support Organization
  Dept. HYTD Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400

1

Overview

In this chapter, we introduce you to WebSphere Application Server Community Edition V2.0 (referred to as Community Edition hereafter), which is a lightweight application server that is available as a free of charge downloadable product from IBM.

Our purpose, in this chapter, is to provide you with guidance so that you can assess whether Community Edition V2.0 is suitable for your solution.

We discuss the following topics in this chapter:

1.1 What is WebSphere Application Server Community Edition

WebSphere Application Server Community Edition is a lightweight application server that is built on Apache Geronimo (V2.0.1) technology, which is an open source application server that the Apache Software Foundation developed. Community Edition uses the best-of-breed open-source technologies, such as Apache Tomcat, OpenEJB, Apache ActiveMQ, and Apache Derby. Access to standard J2EE services, such as database resources, messaging, LDAP directory servers, and security are also supported in Community Edition.

The following Web site contains a complete list of open-source technologies (or projects) that are used in Community Edition V2.0.

http://publib.boulder.ibm.com/wasce/V2.0.0/en/open-source.html

1.2 Components of Community Edition

In this section, we describe the components that provide the runtime support in Community Edition. Table 1-1 lists the name of open-source technologies and the function that they perform in Community Edition V2.0.

Table 1-1 WebSphere Application Server Community Edition components

Component name	Technology used
Web container	Apache Tomcat
EJB container	OpenEJB
SOAP JAX-RPC, SAAJ support	Apache Axis2
JDBC™ support	TranQL JCA connectors
JMS support	Apache ActiveMQ
CORBA support	Apache Yoko
JMX™ support	MX4J

1.3 Community Edition architecture overview

Community Edition is built on Geronimo technology, which is a lightweight framework that has the Geronimo Bean (GBean) Kernel as its foundation. The GBean Kernel consists of kernel services, such as LifeCycle, Naming, Security, and Transaction, which provide the core functionality of the GBean Kernel. With Geronimo, you can add and manage optional plug-in packages using the GBean technology, which specifies how to build wrappers around the optional packages. These optional packages include open source technologies, such as Tomcat, OpenEJB, ActiveMQ, and so on. Figure 1-1 on page 3 illustrates an overview of the Geronimo architecture.

Community Edition does not contain all of the plug-ins that are found in the Geronimo Application Server, for example, Geronimo Application Server comes with Tomcat and Jetty as its Web containers; whereas, Community Edition comes with only Tomcat as its Web container. It is possible to extend the capabilities by deploying new plug-ins in Community Edition.

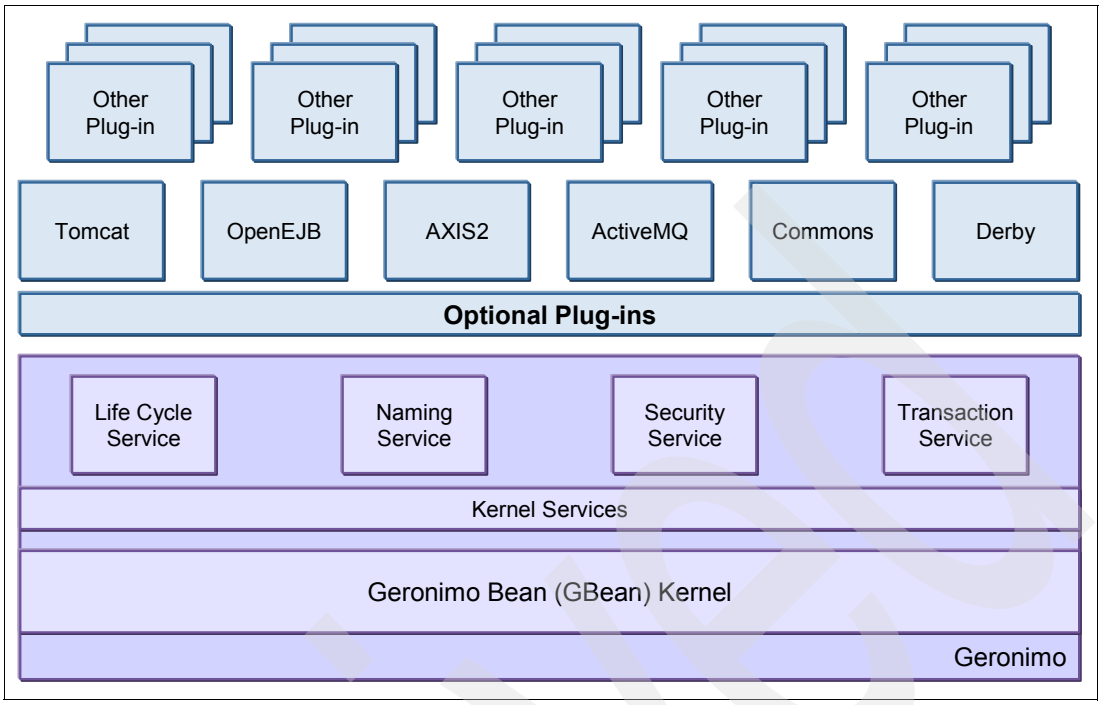


Figure 1-1 WebSphere Application Server Community Edition architecture

Refer to the following Web site for more information about the Geronimo architecture:

http://cwiki.apache.org/GMOxDEV/geronimo-architecture.html

1.4 Why choose Community Edition V2.0

Community Edition is a member of the WebSphere Application Server family. It provides you with a low acquisition-cost application server environment to deploy, test, and run Java-based solutions.

As a registered user, you have access to free technical support from IBM during the trial period, which starts after the product download. Information about the 30-Day Trial Support is located at the following Web site:

http://www.ibm.com/developerworks/downloads/ws/wasce/trialsupport.html

You can use Community Edition at no cost in any type of environment, which includes a production environment. You have access to optional IBM technical support for a fee. We discuss more details about the support options that are available to you as a Community Edition user in 1.6, “Support and training” on page 6.

Community Edition V2.0 is certified for Java Enterprise Edition 5 (Java EE 5), so you can take advantage of the new features that are available in Java EE 5 for your solutions. Given that Community Edition is built on Apache Geronimo, you can use services that are provided by a significant number of open source technologies in your solutions.

Community Edition provides an easy-to-setup stand-alone application server. You can deploy and test your Java-based applications in a short time. You can even migrate these applications to other WebSphere Application Servers if the business requirements for these applications, such as the number of users or high availability, changed. WebSphere

Extended Deployment customers can incorporate Community Edition servers into their operations' optimization environment to provide greater quality-of-service (QoS) and central management capabilities to Community Edition server environments.

You might also improve the reliability for Web applications that are hosted on Community Edition server environments by using the Tomcat Web-tier clustering technology. Details about how to implement this are in 4.1, "Community Edition application server clustering" on page 32.

You can incorporate Community Edition servers into integrated development environments (IDE), such as Eclipse and IBM Rational Application Developer, which allows Java-based solution developers to test the solutions using Community Edition inside IDEs.

1.5 What is new in V2.0

WebSphere Application Server Community Edition V2.0 is considered a major release. It is the first Community Edition product version to be certified as a Java Enterprise Edition Version 5 compliant application server. It provides the full support for applications that are developed according to the Java EE 5 specifications.

The next section provides a description of the new features that are available in Community Edition 2.0.

1.5.1 Support for Java Enterprise Edition V5

The updated Java EE 5 platform technologies in this release are:

- Web Applications:
  - Servlet 2.5
  - JSP™ 2.1
  - JSTL 1.2
  - JSF 1.2
- Enterprise Applications:
  - Common Annotations
  - EJB 3.0
  - JPA 1.0
  - JAF 1.1
  - JTA 1.1
  - JACC 1.1
  - JavaMail™ 1.4
- Web Services:
  - Web Services Metadata (POJO annotation)
  - JAXB 2.0
  - JAX-WS 2.0
  - SAAJ 1.1/1.3
  - SOAP 1.1/1.2
  - WSDL 1.1/2.0

Refer to the following Web site (for example, Sun™ Java™ EE) for details about the Java EE5 (JSR 244) contents and specifications:

http://java.sun.com/javaee/technologies/

The IBM software development kit (SDK), Java2 Technology Edition, was also updated to extend the support of 64-bit Linux® environments that run on Intel® and AMD™ processors.

1.5.2 Support for additional Apache open source projects

Because Community Edition V2.0 is based on Apache Geronimo V2.0, open source projects, such as ActiveMQ, Axis, Dojo, Log4J, and MyFaces are included in this release. Refer to the following Web site for a complete list of included open sources projects in this release:

http://publib.boulder.ibm.com/wasce/V2.0.0/en/open-source.html

1.5.3 Support for additional platforms

Installer and runtime support were extended to include new recommended and compatible platforms in Community Edition V2.0, which includes:

- New recommended platforms:
  - Asianux Server 3
  - Novell SUSE Linux Enterprise Desktop 10
  - Red Hat Enterprise Linux Version 5 Server
- New compatible platforms:
  - Red Hat Fedora 7
  - Ubuntu 6.06 LTS, 6.10 and 7.04

Refer to the following Web site for a full list of all supported platforms:

http://www-1.ibm.com/support/docview.wss?rs=2359&uid=swg27006834

1.5.4 Other enhancements

Additional enhancements include:

- The Community Edition V2.0 download Web site contains new and updated sample applications that demonstrate the new Java EE 5 features.
- Reorganized online documentation for Community Edition V2.0 based on administration and development tasks.
- Ability to use Web-tier (Tomcat) clustering for load balancing and fail-over.
- Eclipse plug-in with J2EE profiling capabilities for developing applications that can be ported to other WebSphere Application Server family servers.
- Integrated support for databases that are built on Apache Derby technology.
- New tools for administrators, for example:
  - JMX viewer
  - LDAP viewer
  - Classloader viewer
  - JNDI viewer
  - Dependency viewer

1.6 Support and training

In this section, we cover the support and training options that are available to you as a WebSphere Application Server Community Edition user.

1.6.1 Free trial support

As a registered user of Community Edition, you are entitled to limited online support from IBM during a 30-day trial period, at no charge. The Community Edition trial support program provides free online support to help you with problems or questions about the product. Refer to the following Web site for up-to-date information about the free trial support program.

http://www.ibm.com/developerworks/downloads/ws/wasce/trialsupport.html

1.6.2 Paid support

You have the option of getting fee-based IBM technical support for Community Edition as a subscription. The subscription is priced per server.

The following three tiered support offerings are available under this program:

- Entry Support

  Under this program, you can contact IBM for technical incidents that are related to the product use. You get access to the online documentation and you can also track fixes online. However, no Developer assistance is provided to you under this support program. Entry Support does not provide you with 24x7 support.

- Enhanced Support

  This support program provides you with access to the Developer assistance and all the support features in the Entry support program. Enhanced Support does not provide you with 24x7 support.

  Developer assistance allows you to call IBM support with questions that go beyond the typical traditional technical support, for example, you can receive assistance with specific questions regarding programming, best practice usage of the product, configuration, and tuning assistance.

- Elite Support

  The Elite support program is basically the same as the Enhanced support program; however, this tier provides you with 24x7 support, which is missing in the other two support programs.

Refer to the following Web site for up-to-date information about the tiered support offerings.

http://www-306.ibm.com/software/webservers/appserv/community/detail/table.html?S_TACT=105AD02W&S_CMP=campaign

1.6.3 Training

The Support tab at the following Web location contains links to information about Community Edition 2.0, which includes developerWorks® articles, the WebSphere technical library, and the WebSphere Redbook library.

http://www.ibm.com/developerworks/downloads/ws/wasce/support.html

1.7 Summary of references

The following list contains references:

- WebSphere Application Server Community Edition V2.0 documentation

  http://publib.boulder.ibm.com/wasce/V2.0.0/en/index.html

  - Included Open Source Projects

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/open-source.html

- Apache Geronimo architecture

  http://cwiki.apache.org/GMOxDEV/geronimo-architecture.html

- Download: WebSphere Application Server Community Edition 2.0 free trial support program

  http://www.ibm.com/developerworks/downloads/ws/wasce/trialsupport.html

- Sun Developer Network: Java EE Technologies at a Glance

  http://java.sun.com/javaee/technologies/

- WebSphere Application Server Community Edition detailed system requirements

  http://www-1.ibm.com/support/docview.wss?rs=2359&uid=swg27006834

- WebSphere Application Server Community Edition support offerings

  http://www-306.ibm.com/software/webservers/appserv/community/detail/table.html

- Download: WebSphere Application Server Community Edition 2.0 self-help and training resources

  http://www.ibm.com/developerworks/downloads/ws/wasce/support.html

2

Planning

In this chapter, we describe how to plan and design an infrastructure based on WebSphere Application Server Community Edition V2.0. It is important to understand what Community Edition V2.0 can offer you as an application server and what is required for it to function correctly before you plan for an infrastructure that is based on it. In this chapter, we intend to provide you with guidance for your planning activities.

We discuss the following topics in this chapter:

2.1 System requirements

Customers often want to use a broad set of configurations that include various combinations of platforms, operating systems, Java SDKs, database servers, LDAP servers, and so on, in an infrastructure that is based on Community Edition V2.0. Because Community Edition V2.0 might not have been tested in your specific configuration, it is important that you use the *recommended* versions of these infrastructure components in your design. However, there are also a number of *compatible configurations* that you can consider.

Recommended platforms: IBM recommends that you use these operating system levels and processor architectures because IBM tests WebSphere Application Server Community Edition with these platforms.

Compatible platforms: IBM makes no formal claim of how well these platforms actually perform and therefore cannot include them as Recommended Configurations at this point-in-time.

You might want to run WebSphere Application Server Community Edition on a platform that you are more familiar with but is not on the Recommended Configuration category. IBM expects that WebSphere Application Server Community Edition should function reliably on a given platform if:

- The application itself conforms to the J2EE/Java EE specification.
- The platform was validated by the Apache Geronimo community and is supported by IBM under an IBM Support for Apache Geronimo support contract.
- The platform has undergone a relatively minor level of testing by IBM to validate basic functionality (install, start the server, stop the server, and so on) on the platform.
- The IBM Business Partner (who owns, creates, maintains, publishes, sells, or delivers the platform) claims that WebSphere Application Server Community Edition works with their platform configuration.

A full, current list of recommended and compatible system requirements are at the following Web site:

http://www-1.ibm.com/support/docview.wss?rs=2359&uid=swg27006834

2.1.1 Recommended platforms

Community Edition is recommended for the following host operating systems:

- Host operating systems for platforms on Intel/AMD 32-bit processors:
  - Windows® 2003 Server with SP1 or SP2
  - Windows XP with SP2 (Supported for design, development, and testing only. No support is provided for production use.)
  - Red Hat Enterprise Linux V4 U4 or 5 AS, ES, WS
  - SUSE Linux Enterprise Server (SLES) V9 SP2 or SP3
  - SUSE Linux Enterprise Server (SLES) V10 SP1
  - SUSE Linux Enterprise Desktop (SLED) V10 SP1 (Supported for design, development, and testing only. No support is provided for production use.)
  - Asianux Server V3

- Host operating systems for platforms on Intel/AMD 64-bit processors:
  - Windows 2003 Server with SP1 or SP2
  - Windows 2003 Server x64 with SP1 or SP2
  - Windows XP with SP2 (Supported for design, development, and testing only. No support is provided for production use.)
  - Red Hat Enterprise Linux V4 U4 or 5 AS, ES, WS
  - SUSE Linux Enterprise Server (SLES) V9 SP2 or SP3
  - SUSE Linux Enterprise Server (SLES) V10 SP1
  - SUSE Linux Enterprise Desktop (SLED) V10 SP1 (Supported for design, development, and testing only. No support is provided for production use.)
  - Asianux Server V3
- Host operating systems for platforms on pSeries®:
  - Red Hat Enterprise Linux V4 U4 or 5 AS, ES, WS
  - SUSE Linux Enterprise Server (SLES) V9 SP2 or SP3
  - SUSE Linux Enterprise Server (SLES) V10 SP1
  - SUSE Linux Enterprise Desktop (SLED) V10 SP1 (Supported for design, development, and testing only. No support is provided for production use.)
  - Asianux Server V3
  - AIX® V5 U3 ML4 or later
- Host operating systems for platforms on SPARC
  - SUN Solaris™ V10

2.1.2 Recommended JVMs

IBM Java is the only SDK that is currently a recommended JVM™ for use with Community Edition. The recommended versions are:

- 32bit SE 5 SR5 + iFix IZ02455
- 64bit SE 5 SR5 + iFix IZ02455 (only on Linux/x86-64)

If you are in doubt as to the recommended JVM on your server, download the Community Edition version that includes the IBM JVM for your server. The Community Edition installer confirms that a valid JVM is being used for the install and warns the installer if the JVM is not a recommended version.

The Sun Java 32-bit SE 5 (1.5.0 Update 11 or later) is a compatible JVM that you can use in development and production use. However, we recommend that you use the IBM JVM in production.

2.1.3 Recommended database managers

Only the database managers that are in the following list are currently recommended for use with applications hosted on Community Edition servers. However, you can use any JDBC-based database with Community Edition. The administrative console Database Pools wizard provides simple steps for downloading and installing drivers for various databases.

The recommended database managers are:

- Apache Derby V10.2.2.0
- IBM DB2® Express-C V8.1 with FixPack 11 or 12, V8.2 with FixPack 4 or 5, V9, V9.1
- UDB V8.1 with FixPack 11 or 12, V8.2 with FixPack 4 or 5, V9, V9.1
- Microsoft® SQL Server® 2000 with SP 3a or 4
- Microsoft SQL Server 2005 with or without SP 1
- MySQL™ Community Edition V4.1.10a, V5.0.41
- Oracle® 10G V10.2.0.1.0

2.1.4 Recommended LDAP servers

Only the LDAP servers in the following list are currently recommended for use for application and administrative security implementations and directory server related services in the applications. However, almost any LDAP server can be used with Community Edition. The administrative console Security Realms wizard provides simple steps for creating and testing an LDAP based security realm. The recommended LDAP servers are:

- Apache Directory Server V1.0
- IBM Tivoli® Directory Server V5.1, V5.2 & V6.0
- Micro Soft Active Directory® 2000
- Open LDAP Server V2.2
- Sun One Directory Server V5.2

2.2 Things to consider

In this section, we cover what you need to consider before using Community Edition V2.0 as the application server in a development, test, or production environment.

2.2.1 Nature of the applications

It is important that you understand the non-functional requirements of the business application or the solution before you decide to use the Community Edition for it. Some of the main non-functional requirement items that you need to look at during the analysis process are:

- Application Uptime requirements

  You can use Community Edition with non-critical applications and critical applications that require fail over. However, critical applications that require high reliability, availability, and scalability requirements might be candidates for other WebSphere servers, such as WebSphere Application Server Network Deployment.

- Response time requirements

  Consider implementing Community Edition application server clustering based on Tomcat clustering technology to improve the response times by using multiple server instances and load balancing.

- Disaster recovery requirements

  Different business units in your organization can set different restoration times for their applications, for example, a business unit might require that the services of some applications be restored within four hours after the disaster. For some other applications, the restoration times can take as long as two days. Therefore, you need to decide how a Community Edition-based application server environment can be designed to meet these recovery requirements.

- Availability requirements

  You need to decide how a Community Edition-based application server can be designed to satisfy the availability requirements of your business applications. Consider using IP load balancers, Web servers, and clustering to set up a highly available application server environment for business applications.

2.2.2 Migration path to other WebSphere Application Servers

When you design and develop applications to host on a Community Edition V2.0 server, be aware of the migration paths that are available to you. If your applications implement Java EE 5 features that are not supported yet on WebSphere Application Server, you cannot migrate them to these more robust application servers. However, this is not an issue if the applications are designed with ease of migration considered.

Table 2-1 shows a comparison of Community Edition V2 to WebSphere Application Server V6.1.

Table 2-1 Comparison chart

Feature	Community Edition 2.0	WebSphere Application Server V6.1
Java EE certification	5	1.4
Servlet	2.5	2.4
JSP	2.1	2
JSF	1.2	1.1
EJB	3.0	2.1 (EJB 3.0 with Feature Pack for EJB 3.0)
Java Persistence API	1.0 (OpenJPA)	(with Feature Pack for EJB 3.0)
JMS	1.1	1.1
Web Service	2.0	2.0

For more information about migration strategies from Community Edition to WebSphere Application Server, refer to *Migrating from WebSphere Application Server Community Edition to WebSphere Application Server*, SG24-7433. The book is relative to Community Edition V1.1, but it is still valid because J2EE1.4 applications can be deployed *as-is* into Community Edition V2.0. The book is available at:

http://www.redbooks.ibm.com/abstracts/sg247433.html

More information about the differences between V1.1 and V2.0 of Community Edition is available in 13.1, “Migration from Community Edition v1.1 to Community Edition 2.0” on page 198.

2.2.3 Environment management

The following list contains issues for you to consider for ongoing management of a Community Edition environment:

- Software installation

  A WebSphere Application Server Community Edition installation can be propagated to multiple servers by zipping the *wasceHome* directory, copying it to the new server, and unzipping it, which makes it easy to install Community Edition application servers without going through the installation process each time.

  The location of the JVM (JAVA_HOME) must be the same on each server.

  If there are port conflicts on the new installation, you can change the ports using one of the methods that we outline in 5.4.1, “Changing port numbers” on page 53.

- Application deployment

  Although the hot deployment facility is useful in a development environment, it is not appropriate in production environments. The server processes each file within the /deploy directory at server startup, which causes slower startup and possible issues with deployed applications. We recommend that you use the console or deployer tool for production deployment environments.

- Physical server management

  Consider ways to ensure that platform requirements are met before Community Edition is installed on a server. It is possible that another team in you organization is responsible for managing the servers. Therefore, an agreement with the other team is necessary to ensure that installation of Community Edition software occurs without a problem. You can look at defining server pattern templates, which outlines the operating system, hard disk, memory, and security requirements for Community Edition environments.

- The Community Server administration

  There is no central management facility for multiple servers such as that found in WebSphere Application Server Network Deployment, which means that you must manage Community Edition environments independently of each other. To ensure the integrity of server environments, especially in production, you need to have processes in place that outline how the administration activities are carried out for these environments.

2.2.4 Standards and processes

It is important that you have a well defined and documented process for the usage of Community Edition in your organization. The following list contains some items that you should consider for this:

- Home directory name (referred to as *wasceHome* in the rest of this publication)

  Having a standard directory location to install Community Edition software allows you to develop a repeatable process for software installation, which also facilitates a standard security infrastructure implementation (for example, NTFS permission settings on Windows platform and file permission settings on Unix platform) for Community Edition directories. When you define standards for the home directory, you also need to consider how the future upgrades can be carried out without much disruption.

  Example 2-1 on page 15 is a suggested home directory name for the Windows platform.

Example 2-1 Suggested home directory name for the Windows platform

```
C:\WebSphere\CommunityEdition<versionno>
Examples:
   C:\WebSphere\CommunityEditionv10 (For V1.0)
C:\WebSphere\CommunityEditionV20 (For v2.0)
```

With the type of directory structure that is in Example 2-1:

- You can set the NTFS security at the C:\WebSphere directory level, thus avoiding the need to request security changes when ever a version upgrade is implemented.
- You can install and configure a new version of Community Edition while another version is operational. Also the migration (and back out, if needed) between old and new versions is easier.

- Directory names for repository locations when multiple repositories are used

  If you decide to use the multiple repository feature of Community Edition with a single service instance, it is a good idea to have simple and easily understood standards for naming these repositories. Having such a standard helps you with the administration and problem determination activities.

- Directory names for the var directory when multiple server instances are used

  If you decide to use the multiple server instance feature of Community Edition, it is a good idea to have simple standards for naming these server instances. Having such a standard helps you with the administration and problem determination activities.

- Port numbers for Community Edition servers

  It is possible that you might not want to use the default port numbers that Community Edition uses, for security reasons. If this is the case, reserve a set of port numbers at the network level for Community Edition server to use to avoid port conflicts with other applications that are running on the same server. It is a good idea to reserve port numbers if you decide to use the multiple server instance feature of Community Edition as the extension to the default port numbers, which might not be able to avoid port conflict situations.

- Server and application log file locations

  It is a good idea to store server and application log files in a designated directory for log files on a different drive to where the operating system and application binaries are stored, for example, you can create the logs file directory on the D: drive on a Windows platform and /var file system on UNIX systems. Having such a standard ensures that unexpected growths of log files do not cause unplanned outages to the servers. We discuss defining log file locations in 12.1.1, “Server logs” on page 188.

2.3 Summary of references

- *Migrating from WebSphere Application Server Community Edition to WebSphere Application Server*, SG24-7433.
- WebSphere Application Server Community Edition detailed system requirements

  http://www-1.ibm.com/support/docview.wss?rs=2359&uid=swg27006834

3

Community Edition installation

In this chapter, we describe what you need to know to install WebSphere Application Server Community Edition.

We discuss the following topics in this chapter:

We provide information about troubleshooting installation problems in Chapter 12, “Troubleshooting techniques” on page 187.

3.1 Downloading Community Edition V2.0

Community Edition is downloaded in an install bundle. A supported Java SDK environment (2.1, “System requirements” on page 10) must be available on the server or workstation where Community Edition is installed. If a supported Java SDK environment is not available, select the install bundle that contains both the application server and the IBM SDK that are relevant to your server or workstation type.

You can download the install bundles for Community Edition from the following Web site:

http://www.ibm.com/developerworks/downloads/ws/wasce/

Note: You need a valid universal IBM ID and password to download the install bundles. If you do not have a universal IBM ID, you can register for an ID using a link on the download sign in Web site.

3.1.1 Application server only

If you already have an IBM or SUN SDK or JRE™, you can download the install bundle that contains only the Community Edition application server software. The install bundle you download is dependent on the platform:

- For AIX, Linux and Solaris platforms

  wasce_setup-2.0.0.1-unix.bin (61MB)
- For Windows platforms

  wasce_setup-2.0.0.1-win.exe (62MB)

3.1.2 Application server and IBM SDK

If you are planning to install and use the IBM SDK to provide the JRE (Java Runtime Environment) for your Community Edition server environments, select one of the following install bundles:

- For AIX on pSeries server platform

  wasce_ibm150sdk_setup-2.0.0.1-ppc32aix.zip (139MB)
- For Linux on pSeries server platform

  wasce_ibm150sdk_setup-2.0.0.1-ppc32linux.tar.bz2 (130MB)
- For Linux on 32-bit Intel/AMD server platform

  wasce_ibm150sdk_setup-2.0.0.1-ia32linux.tar.bz2 (122MB)
- For Linux on 64-bit Intel/AMD server platform

  wasce_ibm150sdk_setup-2.0.0.1-x86_64linux.tar.bz2 (124MB)
- For SUN Solaris on SPARC server platform

  wasce_ibm150sdk_setup-2.0.0.1-sparc32solaris.zip (130MB)
- For Windows on 32-bit Intel/AMD server and desktop platforms

  wasce_ibm150sdk_setup-2.0.0.1-ia32win.zip (126MB)

3.1.3 Sample applications

A separate add-on install bundle that contains sample J2EE applications is also available for you to download. You can use the sample applications to evaluate the Community Edition or to practice some of the Community Edition administrative tasks. You can even use them as templates to build your own J2EE applications.

There are many sample applications included in the bundle. Table 3-1 shows some of the currently available samples.

Table 3-1 Sample applications

Application	Description
\calculator-stateless-pojo	Demonstrates the stateless EJB 3.0 support that OpenEJB provides.
\cviewer	Demonstrates a simple JSP Web application using a WAR file.
\daytrader	Demonstrates a sample EAR application that emulates a stock trading storyline. You can look at your portfolio, get quotes, and buy and sell stocks.
\dbdemo1	Demonstrates how to access user information that is stored in the embedded Apache Derby database.
\EMPdemo	Demonstrates how to access user information that is stored in the embedded Apache Derby, IBM Express-C and UDB, Microsoft SQL Server, MySQL, and Oracle database managers.
\file-realm-demo	Demonstrates a file properties security realm and an application that exploits this realm.
\geronimo-jsp-examples	Demonstrates Apache Tomcat JSPs.
\geronimo-servlet-examples	Demonstrates Apache Tomcat servlets.
\hello	Demonstrates a simple "hello world" JSP page.
\jaxws-calculator	Demonstrates a simple calculator using Axis2's JAX-WS.
\jpa	Demonstrates the Java Persistence API.
\jsf	Demonstrates Java Server Faces.
\ldap-realm-demo	Demonstrates an application that exploits LDAP realms that are provided by \ldap-realm.
\ldap-realm	Provides sample LDAP realm configuration plans for ldap-realm-demo. These prebuilt files require customization to use with your LDAP server.
\magicGball	Demonstrates using CORBA to allow an application client to ask a question of a mystical EJB that is deployed in the server over either a secure or unsecure transport.
\mdb	Demonstrates how to use a Message Driven Bean with OpenEJB to update a Customer Database using a Web application or stand-alone Java applications using ActiveMQ.
\PlantsByWebSphere	Demonstrates an online store that specializes in plants and gardening tools. You can open accounts, browse for items to purchase, view product details, and place orders.
\tomcat-cluster	Demonstrates Web-tier clustering, load balancer, and fail-over sample for the embedded Apache Tomcat Web container. This sample requires two physical servers and an Internet connection to download the Apache HTTP Server and mod-jk files.
\welcome	Demonstrates the default Welcome application that is supplied with WASCE.

The name of the sample application package is:

`wasce_samples-2.0.0.1.zip  (23MB)`

We provide an example of deploying a sample application in 10.3, “Deploying and undeploying applications” on page 135.

3.1.4 Community Edition WTP Server Adapter for Eclipse

If you want to use the Eclipse application development framework to develop and test applications using a Community Edition server, you must install the Community Edition WTP Server Adapter (formerly called the Eclipse plug-in) in Eclipse. It comes as a separate install bundle, which you can download from the following Web site:

`http://download.boulder.ibm.com/ibmdl/pub/software/websphere/wasce/updates/`

We provide information about installing and using the server adapter in Chapter 9, “Application development tools” on page 117.

3.1.5 Verifying the integrity of the downloads

The security policies of your organization might require that software that is downloaded from the Internet be verified for the integrity of the downloads. This might include the confirmation that the software actually came from IBM.

All of the Community Edition install bundles come with machine generated MD5 hashes, SHA-1 hashes, and PGP signatures. You can use MD5 and SHA-1 hashes to ensure that install bundles are not corrupted during the download. But these two cannot be used to confirm that install bundles actually came from IBM. For this, you can use PGP signatures.

For more information about installation and integrity checking, see the following Web site:

`http://publib.boulder.ibm.com/wasce/V2.0.0/en/choosing-an-installation-bundle.html`

3.2 Installing and uninstalling the IBM SDK V1.5

In this section, we discuss how to install and uninstall the IBM SDK 1.5 product in both attended (using install Wizard) and unattended (using the silent install facility) modes on a Windows platform.

Installation on all platforms: The following Web site contains information about installation on all platforms.

`http://publib.boulder.ibm.com/wasce/V2.0.0/en/installing-an-application-server.html`

You can skip this section if Java V5 (or V1.5) is already installed on your servers. However, most install issues are caused by incorrect JVMs. If you are in doubt or unsure of the JVM version, install the IBM JVM.

3.2.1 Attended mode installation on Windows

To install the IBM SDK in attended mode on a Windows system:

1. Download wasce_ibm150sdk_setup-2.0.0.0-ia32win.zip, and extract the files into a temporary folder using a utility, such as WinZip. You can delete this directory when the installation is complete to reclaim the disk space.
2. Open a command window, and issue the change directory command to the temporary directory that you used in the previous step.
3. Run ibm-java2-sdk-50-win-i386.exe to install the IBM SDK.

   Note: If the IBM SDK is already installed, you are asked if you want to uninstall it. You can cancel out of the uninstall or allow it to uninstall the SDK. After the SDK is uninstalled, execute ibm-java2-sdk-50-win-i386.exe to re-install the SDK.

4. Select the language for the installation, and click **OK**.
5. At the Welcome page, click **Next**.
6. Read the license agreement, and click **Yes** if you agree to the terms.
7. Select the destination folder for the installation, and click **Next**, for example:

   c:\IBM\Java50

8. Select the type of installation:
   - A typical installation installs the J2SE™ Runtime Environment (JRE), which includes the JVM, the Software Development Kit (SDK), and source code.
   - A compact installation installs only the JRE. Because many IDEs, such as Eclipse, use the SDK, use this selection only in a production environment.
   - A custom installation allows you to select from these components. At a minimum, the JRE is installed.

   For our purposes, we selected a typical installation.
9. You are given the option to install the JRE as the system JVM. If you want to use the IBM JRE as the default system JVM, make sure that this will not cause any issues for other Java applications that are running on the server.
10. On the next page, review your options before you start the installation. If your selections are OK, click **Next**.
11. From the list that is presented, select the browser you want to associate with the Java plug-in from, and click **Next**. Applets that are found in some browser-based applications are sensitive to the Java runtime environment that they run in. If you want to add IBM SDK as a Java plug-in in the Internet Explorer®, make sure that this will not cause any issues for other browser-based applications.
12. When the installation completes, click **Finish**.

3.2.2 Unattended mode (silent install) on Windows

The silent install facility provides you with a mechanism by which you can set up a repeatable process to install software products in unattended mode on Windows platform. The steps in this section outline how this is done for the IBM SDK.

To use the silent install facility, you will need a response file that contains installer specific codes for the responses that you enter in install wizard dialogs during an attended install of

the product. You can create a response file based on responses that are entered during an attended install.

Use these steps to create a response file on a Windows operating system. In this example, a test system is used because the product is installed (to generate a response file for install) and uninstalled (to generate a response file for uninstall):

1. Open a command window, and change the directory to the folder where the IBM SDK install program is located. Run the following command to install the SDK and generate a response file for use in silent installations, where `C:\<folder name>\install.iss` is the location where you want to store the response file.

   ```
   ibm-java2-sdk-50-win-i386.exe /r /f1"C:\<folder name>\install.iss"
   ```

2. Create a response file to uninstall the SDK by executing the following command, which uninstalls the product and creates the response file:

   ```
   ibm-java2-sdk-50-win-i386.exe /r /f1"C:\<folder name> \uninstall.iss"
   ```

3. Copy the response files and installation files to the system where you want to perform the silent install.

4. Change the directory to the folder where the IBM SDK install program is located, and execute the following command to use the response file to install the SDK:

   ```
   ibm-java2-sdk-50-win-i386.exe /s /f1"C:\<folder name>\install.iss"
   /f2"C:\<folder name>\install.log"
   ```

   In the command, the following applies:

   - `C:\<folder name>\install.iss` is the response file name
   - `C:\<folder name>\install.log` is where you want the installation log to be put

   Note: If the installation is successful, you should see the following entries in `"C:\<folder name>\install.log" file.`

   ```
   [ResponseResult]
   ResultCode=0
   ```

3.2.3 Verifying that IBM SDK successfully installed

To verify that IBM SDK successfully installed:

Open a command window, change the directory to <*JAVA_HOME*>\bin, and run the **`java -fullversion`** command.

If the installation was successful, you see output similar to Example 3-1. The `J2RE 1.5.0 IBM Windows 32` portion of the output remains constant. The rest of the output reflects the current maintenance level.

Example 3-1 Verifying the IBM SDK installation

```
C:\IBM\Java50\bin>java -fullversion

java full version "J2RE 1.5.0 IBM Windows 32 build pwi32devifx-20070810 (SR5 +
IZ02455)"
```

3.2.4 Uninstalling the IBM SDK

To uninstall the IBM SDK in attended mode on Windows systems, use the Windows Control Panel option **Add or Remove programs**. You can also run the ibm-java2-sdk-50-win-i386.exe install program from the installation package.

To uninstall the IBM JDK™ on platforms other than Windows, delete the folder where the IBM JVM was installed, and remove any path statement (for example, JAVA_HOME).

If you created an uninstall response file (step 2 in 3.2.2, “Unattended mode (silent install) on Windows” on page 21), you can uninstall the SDK in unattended mode:

1. Open a command window, and change the directory to the folder where the IBM SDK install program is located.
2. Enter the ibm-java2-sdk-50-*platform*.exe command, and specify the response file for example, on Windows, you would enter:

   ```
   ibm-java2-sdk-50-win-i386.exe /s /f1"C:\<folder name>\uninstall.iss"
   /f2"C:\<folder name>\uninstall.log"
   ```

   Note: If the installation is successful, you will see the following entries in `"C:\<folder name>\uninstall.log" file.`

   ```
   [ResponseResult]
   ResultCode=0
   ```

3.3 Installing and uninstalling Community Edition V2.0

In this section, we discuss how to install and uninstall the Community edition V2.0 in both attended (using install Wizard) and unattended (using Silent install facility) modes on a Windows platform.

Installing all platforms: Refer to the following Web site for installation information about all platforms.

http://publib.boulder.ibm.com/wasce/V2.0.0/en/installing-an-application-server.html

We assume that a supported Java SDK is already installed on the server where Community Edition is to be installed.

Important: You can run multiple Community Edition environments on a single server to effectively use the resources on a server. You can do this without installing multiple copies of Community Edition. See 4.3, “Configuring multiple server instances” on page 43 for more information.

3.3.1 Attended mode installation on Windows

The following steps describe how to install Community Edition using a Windows system as an example.

1. Open a command window. Change the directory that contains the install bundle, and execute Community Edition install program, for example:

   download_dir\wasce_setup-2.0.0.1-win.exe

   The installer looks for a compatible java.exe. If it finds one, the installation continues.

2. If you get a message indicating that the SDK cannot be found or that a suitable JVM cannot be found, ensure that you installed a compatible SDK, and use the **-is:javahome** option to start the installer.

   download_dir\wasce_setup-2.0.0.1-win.exe -is:javahome C:*JavaHome*

   Click **Next** to continue.

3. Accept the license agreement, and click **Next** to continue.

4. Specify a folder name to install the Community Edition software. If this installation occurs on a server, ensure that the name of the folder conforms to the standards in your organization. Click **Next**.

5. Review the install summary, and click **Install** to continue with the installation.

6. A typical install takes less than a minute. The final window indicates the success of the installation. Click **Finish** to complete the installation.

Figure 3-1 on page 25 shows what you should see if you use the file explorer to expand the Community Edition install folder.

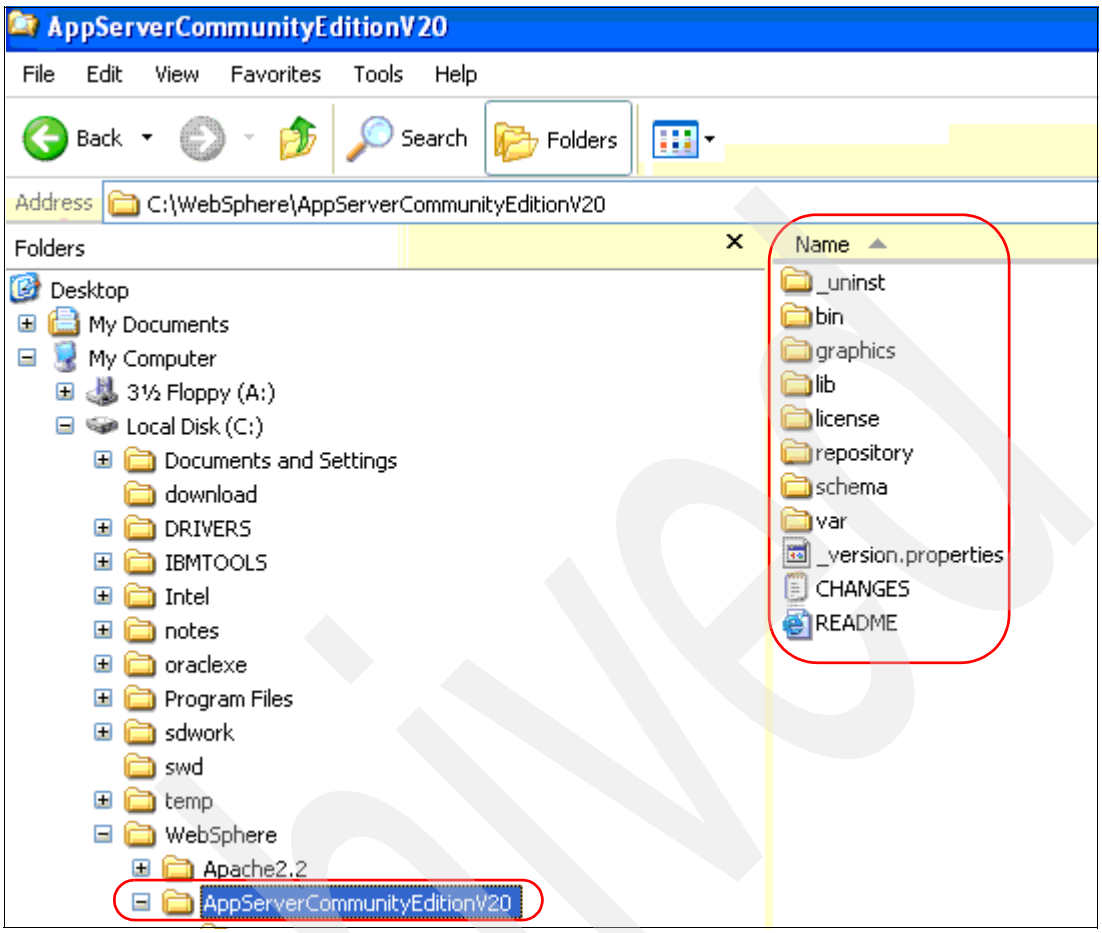


Figure 3-1 Community Edition file structure

Installing Community Edition on additional systems: To install additional instances of Community Edition, you can zip the *wasceHome* directory and copy it to a new location on the system or to new systems. The location of the JVM (JAVA_HOME) must be the same on each server. If there are port conflicts on the new installation, you can change the ports using one of the methods in 5.4.1, “Changing port numbers” on page 53.

3.3.2 Unattended (silent) install

Visit the following Web page to see how you can use the silent install facility to set up repeatable processes to install the Community Edition V2.0. Using this method you can automate the installation of the Community edition.

http://publib.boulder.ibm.com/wasce/V2.0.0/en/setup-reference.html

3.3.3 Attended mode uninstall

To uninstall the Community Edition in attended mode on Windows systems, use the Windows Control Panel option **Add or Remove programs**.

For Windows and other operating systems, such as Linux, you can uninstall by running the *wasceHome*_uninst\uninstaller.exe(sh).

On operating systems other than Windows, you can uninstall by deleting the *wasceHome* directory. Installing on Windows adds entries into the Program Menu, so we do not recommend this method for Windows installations.

3.3.4 Unattended mode uninstall

You can also use unattended mode uninstall to set up a repeatable process to uninstall Community Edition V2.0 on Windows-based servers. The following steps are an example of an uninstall on a Windows system:

1. Open a text editor, and create a batch command file with the following commands in it. Save it as WASCE-UnInstall.bat.

   ```
   SET JAVA_HOME=<fully qualified path to Java V5 home>
   SET WASCE_HOME=<fully qualified path for the Community Edition V2.0 home>
   call %WASCE_HOME%\_uninst\uninstaller -is:silent -is:javahome %JAVA_HOME%
   -silent
   ```

   Refer to the following Web site for more details about uninstaller program-specific parameters.

   http://publib.boulder.ibm.com/wasce/V2.0.0/en/setup-reference.html

   Notes about the install log file: The install log file contains entries that are produced during the install and uninstall. By default, it is created in the home directory of the user who executes the install command. If you are using an automated software install facility in your organization, you might find it difficult to locate the log file because the user ID of the automated installer might not be obvious to you. Therefore, it is a good idea to copy the install log file to a known location during the install. You can use the following commands to do this:

   ```
   SET WASCE_INSTALL_LOG=<fully qualified path where you want the installation
   log file to appear>
   copy /Y "%HOMEDRIVE%%HOMEPATH%\wasce_install.log" "%INSTALL_LOG%"
   ```

   Example 3-2 contains an example of the uninstall command file.

   Example 3-2 Uninstall command file

   ```
   SET JAVA_HOME=C:\IBM\Java50
   SET WASCE_HOME=C:\WebSphere\AppServerCommunityEditionV20
   SET INSTALL_LOG=C:\temp\WASCEV20_Install.log

   call %WASCE_HOME%\_uninst\uninstaller -is:silent -is:javahome %JAVA_HOME%
   -silent

   copy /Y "%HOMEDRIVE%%HOMEPATH%\wasce_install.log" "%INSTALL_LOG%"
   ```

2. Open a command window, and change the directory to the folder that contains WASCE-UnInstall.bat.
3. Run WASCE-UnInstall.bat.

3.4 Troubleshooting installation problems

If your installation was not successful, or you could not start the install wizard, see the following Web site for troubleshooting information.

http://publib.boulder.ibm.com/wasce/V2.0.0/en/setup-troubleshooting.html

We provide some general tips for diagnosing installation problems in the following sections.

3.4.1 Installation problems on Linux, AIX, and Solaris platforms

If you have trouble during installation, the first thing to check is that you have execute permission to the **wasce_setup_*version*.bin** installation bundle.

The Community Edition installer generates a log during installation that is useful for troubleshooting installation problems. If you logged on as root, this log is generated in /root/wasce_install.log; otherwise, it is generated in /home/*user*/wasce_install.log.

Use the following option to write the installation logs to a file of your choice.

```
./wasce_setup_version.bin -is:log launcher.log
```

Change *version* to correct the version of the Community Edition setup you downloaded. The log is generated in the directory from which the command is executed.

The installation wizard searches the registries and conventional locations on your platform to find the java run time. If you have a supported java runtime environment and the Community Edition installer cannot find it, you can specify the Java home location explicitly by using the following command, where *javaInstallDir* is the directory in which java is installed:

```
./wasce_setup_version.bin -is:javahome javaInstallDir
```

3.4.2 Installation problems on Windows platforms

The log file that is generated during installation contains information that is useful in troubleshooting. By default, the installation log file is generated in the following location:

```
C:\Documents and Settings\user\wasce_install.log
```

Use the following option to write the installation logs to a file of your choice, where *version* is the version of Community Edition that you downloaded.

```
wasce_setup_version.exe -is:log launcher.log
```

The log is generated in the directory from which the command is executed.

The installation wizard searches the registries and conventional locations on your platform to find the Java run time.If you have a supported Java runtime environment™ and the Community Edition installer cannot find it, you can specify the Java home location explicitly by using the following command, where *javaInstallDir* is the directory where Java is installed.

```
wasce_setup_version.exe -is:javahome javaInstallDir
```

3.5 Automatically starting a Community Edition server

If you start the Community Edition server using the **startup** command under your own user ID, the server ends when your logon session ends. Therefore, it is a good idea to have the Community Edition servers automatically started, especially on a production server. In the following sections, we describe the options that are available to you to make the Community Edition server an automatically started service on Windows and UNIX platforms.

3.5.1 Starting the server at system reboot time

You can use the **init** facility on Unix platforms or the Scheduled Tasks facility on Windows to have the Community Edition server started at server reboot time. Neither of these facilities tries to start the server process if it fails during its initialization.

On UNIX platforms

Edit the **/etc/inittab** file, and add a line similar to the following line to automatically start the Community Edition server at server reboot time:

```
wasceHome/bin/startup.sh> /dev/console 2>&1
```

Note: Add the following command in **/etc/rc.shutdown** file to have the Community Edition server automatically stopped during a server shutdown on UNIX platforms:

```
wasceHome/bin/shutdown.sh
exit 0
```

On Windows platforms

You can use the Windows Scheduled Tasks facility, which you can access from the Windows Control Panel, to add the Community Edition server as a scheduled task that gets started at system startup.

3.5.2 Running Community Edition Server as a managed service

If you want to run Community Edition server as a managed service, you must make it a daemon service on UNIX platforms or a Windows service on Windows platform. The Community Edition software does not include any tools to make the server a managed service. However, there are tools available on the Web, for example, the Java Service Wrapper, that you can use to configure Community Service as a managed service. The following Web sites contain information about the Java Wrapper Service and how to use it to configure Community Edition as a Windows service.

The following sites have information about how to configure Community Edition as a Windows Service:

- Apache Geronimo: Configuring Geronimo as a Windows Service

  http://cwiki.apache.org/GMOxDOC20/configuring-geronimo-as-a-windows-service.html
- KL-Patterns - Configuring WASCE 2.0 as Windows Service

  http://kl-patterns.com/articles/websphereas/configuring-wasce-2.0-as-windows-service.html

3.6 Summary of references

- Download: WebSphere Application Server Community Edition 2.0

  http://www.ibm.com/developerworks/downloads/ws/wasce/
- Eclipse Update site for IBM WebSphere Application Server Community Edition

  http://download.boulder.ibm.com/ibmdl/pub/software/websphere/wasce/updates/
- WebSphere Application Server Community Edition V2.0 documentation

  http://publib.boulder.ibm.com/wasce/V2.0.0/en/index.html
  - Choosing an installation bundle

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/choosing-an-installation-bundle.html
  - Installing an application server

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/installing-an-application-server.html
  - Setup reference

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/setup-reference.html
  - Setup troubleshooting

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/setup-troubleshooting.html
- Apache Geronimo: Configuring Geronimo as a Windows Service

  http://cwiki.apache.org/GMOxDOC20/configuring-geronimo-as-a-windows-service.html
- KL-Patterns - Configuring WASCE 2.0 as Windows Service

  http://kl-patterns.com/articles/websphereas/configuring-wasce-2.0-as-windows-service.html

4

Extending the topology

In this chapter, we discuss how the availability and scalability of WebSphere Application Server Community Edition application server environments can be improved using Tomcat Web-tier clustering. We also describe how to configure multiple server instances using a single installation of Community Edition V2.0.

We cover the following topics in this chapter:

4.1 Community Edition application server clustering

An application server cluster is a logical grouping of application servers that provide a single server view to the user while the application requests are served by any of the application servers in the cluster. Each application server in the cluster is a node. The process of configuring multiple servers into a cluster is called clustering. The Apache Tomcat Web-tier clustering provides the clustering in Community Edition V2.0.

Benefits of clustering

Clustering can improve the scalability of your applications, and as a result, you can increase the application hosting capacity by adding extra application server instances. Servers that are in a cluster continuously exchange messages to synchronize user session state data. Therefore, clustering can improve the availability of your application because any of the servers have the ability to take over a user session if the server that owns the session fails. You need to use a load balancer, such as Apache Web Server, with a cluster to a gain the benefits from clustering features.

Types of clustering

There are two types of clustering that are available for you to implement:

- Vertical clustering
- Horizontal clustering

Vertical clustering

Running multiple nodes on the same host is called vertical clustering, where you can effectively utilize the capacity of the host.

Note: It is important that you use a different set of port numbers with each server on the same host. Failure to do so results in port conflict errors.

Horizontal clustering

Running nodes of a cluster on multiple hosts is called horizontal clustering. You can improve the capacity of a horizontal cluster by adding new hosts. Horizontal clustering also provides improved availability as more than one host is used.

Limitations of clustering

The current limitations of Community Edition clustering are:

- Stateful session EJBs are not replicated. Therefore, your applications cannot use stateful EJBs.
- Dynamic JNDI updates are not replicated. You must add the JNDI names, which are required by your applications, to all of the application servers in the cluster.
- Application deployments are not replicated. You must deploy each application to each application server in the cluster. There might be some exception to this when a single repository is used with multiple instances of an application server on the same host. We provide more details about this in 4.3, “Configuring multiple server instances” on page 43.
- All of the application servers in a cluster must be on the same physical subnet because of multicast broadcasts requirements. The multicast broadcast is used to exchange user session data among servers.

- HTTP session data is replicated in all nodes and is persisted in memory. Each application server instance has the session data of all instances stored in its memory. Memory used by session data is dependent on the size of the session object and how long the session lasts. In large clusters, this could impact network traffic and storage utilization.
- All application servers in a cluster must have the same time zone and time.

Preparing an application for a clustered environment

An application must meet certain conditions for it to function correctly in a clustered environments. Those requirements are as follows.

- Every object that is placed in the HTTP session implements java.io.Serializable, which is required because the clustering implementation serializes the HTTP session objects when they get distributed to other servers in the cluster.
- The Web application deployment descriptor (web.xml) must have an entry to indicate that the Web application is distributable. Example 4-1 shows where this is specified in the web.xml file.

Example 4-1 Setting distributable in web.xml

```
<web-app>
    <display-name>... </display-name>
    <description>.... </description>
    <distributable/>
    ....
    ....
    ....
</web-app>
```

- Make sure that stateful EJBs are not used with your application.
- If the application requires access to database and JMS resources, make sure that they can be accessed from every server in the cluster. You must also make sure that the necessary resource adapters and drivers are correctly defined in every application server.
- Ensure that JNDI names that are used in your application are configured in all servers.

4.2 Clustering example using Apache HTTP Server

In this section, we describe how to configure a Community Edition application server to use the Tomcat Web-tier clustering facility. We also describe details about how to configure an Apache HTTP server to load balance among the server members of the cluster. The tomcat-cluster sample application is used to verify the load balancing and fail-over functions of the cluster.

Note: The sample configuration that we describe in this section is based on a test that was carried out using two instances of the Community Edition on a single host that ran on Windows. You should be able to use the same procedure to set up a cluster on a Linux and UNIX platform.

For more information about how to configuring clusters, see:

http://publib.boulder.ibm.com/wasce/V2.0.0/en/clustering.html

4.2.1 Installing and configuring two instances of Community Edition

The first step in using vertical clustering is to install multiple instances of Community Edition on one host, which involves the following tasks:

1. Install Community Edition V2.0 software twice using two different home directories, for example:
   - `C:\WebSphere\AppServerCommunityEditionV20_1`
   - `C:\WebSphere\AppServerCommunityEditionV20_2)`
2. Uncomment the PortOffset entry that is in *wasceHome*\var\config\config-substitutions.properties of the second server instance, and assign 10 to it. This ensures that the second instance of the Community Edition server uses a different set of port numbers as its listener ports.
3. Start both of the servers, and confirm that they start without any errors. Use the following URLs to logon to the administrative consoles of the each of the servers.
   - `http://localhost:8080/`
   - `http://localhost:8090/`
4. Shutdown both servers to continue with the remaining configuration activities.

4.2.2 Enabling session affinity in application servers

The session affinity, or sticky session, feature ensures that the load balancer sends Web requests that come from the same user session to the same application server until the server fails. Although the application servers continuously exchange user session data with each other, there is no guarantee that this will happen before subsequent Web requests come in from the same user session. Therefore, we recommend that the support for session affinity is enabled in all member servers.

Each application server must have a unique ID that the load balancer can use to maintain the session affinity. It is important that you perform the following change for all member servers of the cluster. In this example, make corresponding changes to config.xml in both Community Edition installs. Details about how to configure a unique ID for a server are as follows:

1. Use a text editor to open the *wasceHome*\var\config\config.xml file.
2. Find the <gbean> entry with `name="TomcatEngine"`. This entry configures the Tomcat engine of a Community Edition server.
3. Within the <gbean> element:
   a. Locate an attribute element with `name="initParms"`. The initParms element already has a parameter called `name=WASCE20`.
   b. Add a new parameter called `jvmRoute=`*`uniqueId`*, where uniqueId is the unique identifier that is assigned to the server.

   In this example, the new parameters are `jvmRoute=node1` for server instance 1 and `jvmRoute=node2` for server instance 2. Example 4-2 shows how the <gbean> element looks after the update.

Example 4-2 Tomcat <gbean> element with a unique ID configured

```
<gbean name="TomcatEngine">
            <attribute name="initParams">name=WASCE20 jvmRoute=node1</attribute>
</gbean>
```

4. Save the change and start the server.

4.2.3 Enabling clustering in the application server

Deploy the deployment plan in Example 4-3 in all application severs to make them members of a cluster.

Example 4-3 Deployment plan for a cluster

```
<?xml version="1.0" encoding="UTF-8"?>
<module xmlns="http://geronimo.apache.org/xml/ns/deployment-1.2">

  <environment>
    <moduleId>
      <groupId>org.apache.geronimo.configs</groupId>
      <artifactId>TomcatCluster</artifactId>
      <version>2.0</version>
      <type>car</type>
    </moduleId>
    <dependencies>
    <dependency>
      <groupId>org.apache.geronimo.configs</groupId>
      <artifactId>tomcat6</artifactId>
      <version>2.0.1</version>
    </dependency>
    </dependencies>
  </environment>

  <!-- Cluster -->
  <gbean
    class="org.apache.geronimo.tomcat.cluster.CatalinaClusterGBean"
    name="TomcatCluster">
    <attribute name="className">
      org.apache.catalina.ha.tcp.SimpleTcpCluster
    </attribute>
    <attribute name="initParams">
      managerClassName=org.apache.catalina.ha.session.DeltaManager
      expireSessionsOnShutdown=false
      useDirtyFlag=true
      notifyListenersOnReplication=true
    </attribute>
    <reference name="TomcatValveChain">
      <name>ReplicationValve</name>
    </reference>
    <reference name="ClusterListenerChain">
      <name>ClusterSessionListener</name>
    </reference>
    <reference name="Channel">
      <name>TomcatGroupChannel</name>
    </reference>
  </gbean>

  <!-- Cluster Channel -->
  <gbean
    class="org.apache.geronimo.tomcat.cluster.ChannelGBean"
    name="TomcatGroupChannel">
    <attribute name="className">
      org.apache.catalina.tribes.group.GroupChannel
```

```
  </attribute>
  <attribute name="initParams"/>
    <reference name="Membership">
      <name>TomcatMembership</name>
    </reference>
    <reference name="Receiver">
      <name>TomcatReceiver</name>
    </reference>
    <reference name="Sender">
      <name>TomcatSender</name>
    </reference>
    <reference name="ChannelInterceptor">
      <name>TomcatChannelInterceptor</name>
    </reference>
</gbean>

<!-- Cluster Membership -->
<gbean
  class="org.apache.geronimo.tomcat.cluster.MembershipServiceGBean"
  name="TomcatMembership">
  <attribute name="className">
    org.apache.catalina.tribes.membership.McastService
  </attribute>
  <attribute name="initParams">
    mcastAddr=228.0.0.4
    mcastPort=45564
    mcastFrequency=500
    mcastDropTime=3000
  </attribute>
  </gbean>

<!-- Cluster Receiver -->
<gbean
  class="org.apache.geronimo.tomcat.cluster.ReceiverGBean"
  name="TomcatReceiver">
  <attribute name="className">
    org.apache.catalina.tribes.transport.nio.NioReceiver
  </attribute>
  <attribute name="initParams">
    tcpListenAddress=localhost
    tcpListenPort=4001
    tcpSelectorTimeout=100
    tcpThreadCount=6
  </attribute>
</gbean>

<!-- Cluster Sender -->
<gbean
  class="org.apache.geronimo.tomcat.cluster.SenderGBean"
  name="TomcatSender">
  <attribute name="className">
    org.apache.catalina.tribes.transport.ReplicationTransmitter
  </attribute>
  <attribute name="initParams">
    replicationMode=pooled
```

```
      waitForAck=true
    </attribute>
  </gbean>

  <!-- Cluster Replication Valve -->
  <gbean
    class="org.apache.geronimo.tomcat.ValveGBean"
    name="ReplicationValve">
    <attribute name="className">
      org.apache.catalina.ha.tcp.ReplicationValve
    </attribute>
    <attribute name="initParams">

filter=.*\.gif;.*\.js;.*\.css;.*\.png;.*\.jpeg;.*\.jpg;.*\.htm;.*\.html;.*\.txt;
    </attribute>
    <reference name="NextValve">
      <name>JvmRouteBinderValve</name>
    </reference>
  </gbean>

  <!-- Cluster Route Binder -->
  <gbean
    class="org.apache.geronimo.tomcat.ValveGBean"
    name="JvmRouteBinderValve">
    <attribute name="className">
      org.apache.catalina.ha.session.JvmRouteBinderValve
    </attribute>
    <attribute name="initParams">
        enabled=true
    </attribute>
  </gbean>

  <!-- Cluster Listener -->
  <gbean
    class="org.apache.geronimo.tomcat.cluster.ClusterListenerGBean"
    name="ClusterSessionListener">
    <attribute name="className">
      org.apache.catalina.ha.session.ClusterSessionListener
    </attribute>
    <reference name="NextListener">
       <name>JvmRouteSessionIDBinderListener</name>
    </reference>
  </gbean>

  <!-- Cluster Binder Listener -->
  <gbean
    class="org.apache.geronimo.tomcat.cluster.ClusterListenerGBean"
    name="JvmRouteSessionIDBinderListener">
    <attribute name="className">
      org.apache.catalina.ha.session.JvmRouteSessionIDBinderListener
    </attribute>
  </gbean>

  <!-- Cluster Channel Interceptor -->
  <gbean
```

```
    class="org.apache.geronimo.tomcat.cluster.ChannelInterceptorGBean"
    name="TomcatChannelInterceptor">
    <attribute name="className">
      org.apache.catalina.tribes.group.interceptors.TcpFailureDetector
    </attribute>
  </gbean>

</module>
```

It is important that you change the following parameters, which are in the “Cluster Receiver” section of the plan, before you run the deploy tool to deploy them:

- tcpListenAddress

  This must be set to the DNS Name or the IP address of the host.

- tcpListenPort

  This must be set to the TCP listener port. This must be unique to an application server, for example, if two member application servers are running on the same host, remember to assign two different port numbers to this parameter.

- Tomcat_Server_Cluster_Name in the CatalinaClusterGBean name element.

  This is the name of the cluster, which is used in the application’s Web deployment plan to identify the name element as the cluster for the application.

Defining a cluster in the Web deployment plan (alternative)

An alternative to deploying a separate deployment plan and defining the cluster application to each application server, is to embed the cluster gbean definitions in the geronimo-web.xml for the application. The cluster is defined to the server when the application is deployed.

Example 4-4 shows a Web deployment plan with the cluster configuration. In Example 4-4, the following items are highlighted in bold for you to note:

- The <cluster> element identifies the cluster that the application is deployed to. This matches the cluster name, as defined in the name property in CatalinaClusterGBean.
- The embedded definition consists of the gbeans in Example 4-3 on page 35.

Example 4-4 Cluster gbean definition in the Web deployment plan for the application

```
<web-app xmlns="http://geronimo.apache.org/xml/ns/j2ee/web-2.0"
      xmlns:sec="http://geronimo.apache.org/xml/ns/security-2.0"
      xmlns:dep="http://geronimo.apache.org/xml/ns/deployment-1.2"
      xmlns:nam="http://geronimo.apache.org/xml/ns/naming-1.2">
   <dep:environment>
   <dep:moduleId>
      <dep:groupId>MyProjectName</dep:groupId>
      <dep:artifactId>MySimpleWebModule</dep:artifactId>
      <dep:version>2.0</dep:version>
      <dep:type>war</dep:type>
    </dep:moduleId>
    </dep:environment>
    <context-root>/mywebapp</context-root>
<cluster>Tomcat_Embedded_Cluster_Name</cluster>

<!-- imbed the gbeans for the cluster here ...-->
```

```
<gbean class="org.apache.geronimo.tomcat.cluster.CatalinaClusterGBean"
name="Tomcat_Embedded_Cluster_Name">
<attribute
name="className">org.apache.catalina.ha.tcp.SimpleTcpCluster</attribute>

...
<!-- end of gbeans for the cluster...-->

</web-app>
```

4.2.4 Deploying the sample application

The tomcat-cluster sample application comes with two WAR files called servlet-examples-cluster-server1.war and servlet-examples-cluster-server2.war. These two WAR files demonstrate the clustering features, such as load balancing and failover.

Complete the following instructions to deploy these applications in the two application servers:

Server 1

On Server1, do the following:

1. Make the changes provided in Example 4-5 in the Web module deployment plan.

Example 4-5 servlet-examples-cluster-plan.xml on server1

```
<dep:environment>
      <dep:moduleId>
        <dep:groupId>wasce-samples</dep:groupId>
        <dep:artifactId>servlet-examples-cluster-server1</dep:artifactId>
        <dep:version>2.0.0.1</dep:version>
        <dep:type>war</dep:type>
      </dep:moduleId>
      <dep:dependencies/>
      <dep:hidden-classes/>
      <dep:non-overridable-classes/>
</dep:environment>

  <context-root>/servlet-examples-cluster</context-root>

  <container-config>
   <tomcat>
      <cluster>TomcatCluster</cluster>
   </tomcat>
   </container-config>
...
</web-app>
```

   a. Modify the <dep:artifactId> element to reflect server 1.

   b. Ensure that the cluster name matches the cluster name that is used during the server configuration in 4.2.3, “Enabling clustering in the application server” on page 35.

2. Start the server. Deploy the application using the **deploy** command or the **Deploy New** portlet in the administrative console.

   In this example, we open a command window, and enter the commands shown in Example 4-6. Ensure that the **deploy** command is entered on a single line.

Example 4-6 Deploy the sample application on server1

```
SET SAMPLE_HOME=C:\WASCE20Samples\applications\tomcat-cluster
SET WASCE_HOME=C:\WebSphere\AppServerCommunityEditionV20_1

%WASCE_HOME%\bin\deploy.bat deploy
%SAMPLE_HOME%\servlet-examples-cluster-server1.war
%SAMPLE_HOME%\servlet-examples-cluster-plan.xml
```

3. Test the application using the following URL, and confirm that you see “Servlet Examples with Code - SERVER 1’ as the heading in the response Web page.

   `http://localhost:8080/servlet-examples-cluster/`

Server 2

On Server 2, do the following steps:

1. Make the changes provided in Example 4-7 in the Web module deployment plan.

Example 4-7 servlet-examples-cluster-plan.xml on server2

```
<dep:environment>
      <dep:moduleId>
        <dep:groupId>wasce-samples</dep:groupId>
        <dep:artifactId>servlet-examples-cluster-server2</dep:artifactId>
        <dep:version>2.0.0.1</dep:version>
        <dep:type>war</dep:type>
      </dep:moduleId>
      <dep:dependencies/>
      <dep:hidden-classes/>
      <dep:non-overridable-classes/>
</dep:environment>

  <context-root>/servlet-examples-cluster</context-root>

  <container-config>
   <tomcat>
      <cluster>TomcatCluster</cluster>
   </tomcat>
   </container-config>
...
</web-app>
```

   a. Modify the <dep:artifactId> element to reflect server 2, as shown in Example 4-7.

   b. Ensure that the cluster name matches the cluster name that you used during the server configuration in 4.2.3, “Enabling clustering in the application server” on page 35.

2. Start server instance 2. Deploy the application using the **deploy** command or the **Deploy New** portlet in the administrative console. In this example, we open a command window, and enter the commands shown in Example 4-8 on page 41. Ensure that the **deploy** command is entered on a single line.

Example 4-8 Deploy the sample application on server2

```
SET SAMPLE_HOME=C:\WASCE20Samples\applications\tomcat-cluster
SET WASCE_HOME=C:\WebSphere\AppServerCommunityEditionV20_2

%WASCE_HOME%\bin\deploy.bat --port 1109 deploy
%SAMPLE_HOME%\servlet-examples-cluster-server2.war
%SAMPLE_HOME%\servlet-examples-cluster-plan.xml
```

Attention: When you use the deploy tool on a host where there are more than one Community Edition application servers running, ensure that **--port <portno>** is included in the command line. Otherwise the deploy will use the default port number (port 1099) and will deploy the application in the wrong server. On server instance 2, the deployer listener port is 1109 (1099 + port offset 10).

3. Test the application using the following URL, and confirm that you see “Servlet Examples with Code - SERVER 2” as the heading in the response Web page.

   `http://localhost:8090/servlet-examples-cluster/`

4.2.5 Configuring the Apache HTTP Server

In this section, we describe how to configure an Apache HTTP Web server to work a Community Edition server cluster. It is important that you have an understanding about the components that are involved in providing the Web server to application server communication.

Apache HTTP Server

We use the Apache HTTP server as the Web server for this exercise. The Web server can run on another host provided that it can communicate with the application server hosts. If a firewall is located between the Web and application server hosts, you must open the ports that are used by AJP (Apache Jserv Protocol) connectors that are running on the application servers. Refer to “AJP Connectors” on page 41 for more details about the AJP connectors.

We assume that you already have a running Apache HTTP server; otherwise, use the following Web address to download and install the Apache HTTP server:

`http://httpd.apache.org/`

Apache Tomcat JK module

The Apache Tomcat JK module, called mod_jk, is an add-on module for the Apache HTTP Server that manages the communication between the Web server and the AJP connector on Tomcat Web containers. This module also manages the load balancing and fail over for Web requests that are directed to the application servers. The Community Edition software does not include the mod_jk module, so download it from the following Web site:

`http://tomcat.apache.org/connectors-doc/`

AJP Connectors

AJP connectors allow a Tomcat Web container to receive HTTP requests from other software components, such as Apache HTTP server, proxy servers, and AJP protocol aware load balancers. The AJP connectors run on the Tomcat Web container within a Community Edition application server. A default AJP connector gets created when the Community Edition server is installed. You also can use the administrative console to add, modify, and delete AJP

connectors. You can see the AJP connector in the console by selecting the **Web Server portlet**.

Configuration

Note: For this example, we recommend that you use the Apache HTTP V2.0.54 or later server and mod_jk 1.2.15 or later. You can also use the Apache HTTP Server Version 2.2.6 or above, which includes the Apache Tomcat JK modules, mod_proxy, mod_proxy_ajp, and mod_proxy_balancer, but we do not cover them in this example.

1. Copy module mod_jk-*<versionno>*.so to the *Apache_home*\modules folder.
2. Add the following text in the *Apache_home*\conf\httpd.conf file to the end of load module list:

   ```
   LoadModule jk_module modules/mod_jk-<versionno>.so
   ```

3. Add the lines in Figure 4-1 in *Apache_home*\conf\httpd.conf anywhere after the line:

   ```
   LoadModule jk_module modules/mod_jk-<versionno>.so
   ```

```
#
# The following can be added anywhere after the above LoadModule statement.
#
<IfModule mod_jk.c>
    JkWorkersFile       conf/workers.properties
    JkLogFile           logs/mod_jk.log
    JkLogLevel          info
    JkLogStampFormat    "[%a %b %d %H:%M:%S %Y] "
    JkOptions           +ForwardKeySize +ForwardURICompat -ForwardDirectories
    JkRequestLogFormat  "%w %V %T"

    <Location /*/WEB-INF/*>
        AllowOverride None
        deny from all
    </Location>

    <Location /*/META-INF/*>
        AllowOverride None
        deny from all
    </Location>

    # forward ALL web requests to our mod_jk loadbalancer workers
    JkMount /* loadbalancer

</IfModule>
```

Figure 4-1 *Apache_home*\conf\httpd.conf

4. Use a text editor to create a file called *workers.properties*, and save it in the *Apache_home*\conf folder. Specify the full path to this file in *Apache_home*\conf\httpd.conf using the `JkWorkersFile` parameter (See Figure 4-1). Module mod_jk uses the information that is in this file to direct Web requests for load balancing. Example 4-9 on page 43 provides an example of the workers.properties file.

Example 4-9 workers.properties

```
worker.list=loadbalancer,status
worker.maintain=60

worker.node1.type=ajp13
worker.node1.host=localhost
worker.node1.port=8009
worker.node1.socket_timeout=60
worker.node1.socket_keepalive=true
worker.node1.lbfactor=1

worker.node2.type=ajp13
worker.node2.host=localhost
worker.node2.port=8019
worker.node2.socket_timeout=60
worker.node2.socket_keepalive=true
worker.node2.lbfactor=1

worker.loadbalancer.type=lb
worker.loadbalancer.balanced_workers=node1,node2
worker.loadbalancer.sticky_session=1
worker.status.type=status
```

Note:

- The `worker.loadbalancer.balanced_workers=node1,node2` entry indicates that there are two servers, *node1* and *node2*, which are available for load balancing.
- Entries with worker.node1.* specify the details about *node1*, for example:
  - worker.node1.host specifies the DNS or IP address of the host on which application server1 (represented by *node1* in this file) is hosted.
  - worker.node1.port specifies the port number on which AJP connector on application server 1 listens. On server instance 2, the worker.node2 port is 8019 (8009 + port offset 10).

5. Restart the Apache HTTP server.

You can now use the following URL to access and test the applications. The following URLs do not contain the port numbers, which means that they use port 80, which is the port on which the Apache HTTP server listens.

```
http://localhost/servlet-examples-cluster
http://localhost/servlet-examples-cluster/servlet/SessionExample
```

4.3 Configuring multiple server instances

Community Edition V2.0 supports running multiple instances of application servers on the same host. In this section, we discuss the configuration options that are available to you.

It is important to understand how the Community Edition manages configuration data and deployed applications. This gives you an idea about the capabilities and limitations of the product when it comes to managing multiple instances of Community Edition application servers.

4.3.1 Configuration management

When the Community Edition V2.0 software is successfully installed, it creates the following directory structure on the server:

- /_uninst

  This directory contains information that the uninstaller program uses.

- /bin

  Contains command and jar files that are required to run the application server.

- /graphics

  Contains images that the administrative console uses.

- /lib

  Contains the kernel jar files that the kernel services use.

- /license

  Contains license and notices for the Community Edition.

- /repository

  Contains shared libraries, such as database, JMS drivers and connectors, and deployed applications.

- /schema

  Contains schema definitions for XML-based deployment plans and configuration files.

- /var

  Contains log files, transaction logs, and security files and configuration files that the server uses at run time.

Only the /var directory contains information that is specific to the application server runtime configuration. The /repository and other directories contain the kernel and supporting modules that are required for the server to function. The /repository directory also contains installed applications.

4.3.2 Multiple server instances using a single repository

You can run multiple instances of a server from a single installation directory on a host. Using a single repository for multiple servers can simplify application deployment. Deploying an application to one server, deploys it to all servers using the repository.

Advantages

- You can configure any number of physically independent application server environments depending on the hardware resources on the host. This means that you can reduce the cost on hardware and other software.
- Each server instance can operate independently.
- You can use clustering to improve scalability and availability to set up a managed application server environment to host production-like applications.
- Reduces the environment management overheads as the single installation of Community Edition is used to manage the underlying software.

Disadvantages

- Administrators and developers must be aware of the environment, for example, each server has its own administrative console. Therefore, when using the console, users need to be certain that the correct URL is used.

Setting up multiple server instances with a single repository

The following steps illustrate how to set up this environment:

1. Create a home directory for each new server instance in the *wasceHome* folder, as shown in Example 4-10.

   Example 4-10 Home directories for multiple server instances

   ```
   C:\WebSphere\CommunityEditionV20\DevServer
   C:\WebSphere\CommunityEditionV20\TestServer
   C:\WebSphere\CommunityEditionV20\QAServer
   ```

2. Copy *wasceHome*/var/* to *wasceHome**new_server_home*.
3. Update *wasceHome**new_server_home*\config\config-substitution.properties to uncomment the PortOffset parameter and assign a value to it (for example, 10). This change ensures that the port numbers that the new server uses does not conflict with the listener ports that the other application servers use on the host.
4. Use a text editor to create a command file called **startup.bat** (or **startup.sh** for UNIX), and enter the following lines in it.

   ```
   SET GERONIMO_OPTS=-Dorg.apache.geronimo.server.name=<new server name>
   ..\bin\startup.bat
   ```

   Save this file in the folder *wasceHome**new_server_home*. You can use this command file to start the new server, as shown in Example 4-11.

   Example 4-11 startup command for new server instance

   ```
   SET GERONIMO_OPTS=-Dorg.apache.geronimo.server.name=DevServer
   ..\bin\startup.bat
   ```

5. Use a text editor to create a command file called **shutdown.bat** (or **shutdown.sh** for UNIX), and enter the following lines in it:

   ```
   SET GERONIMO_OPTS=-Dorg.apache.geronimo.server.name=<new server name>
   ..\bin\shutdown.bat
   ```

   Save this file in folder *wasceHome**new_server_home*. You can use this command file to shutdown the new server, as shown in Example 4-12.

   Example 4-12 shutdown command for new server instance

   ```
   SET GERONIMO_OPTS=-Dorg.apache.geronimo.server.name=DevServer
   ..\bin\shutdown.bat --port <naming service port number>
   ```

6. Use the following URL to access the administrative console of the new server:

   ```
   http://localhost:(8080+value of(PortOffset))/console
   ```

   Note: It is also possible for you to have the new server automatically started. See 3.5.2, “Running Community Edition Server as a managed service” on page 28

Because all of the server instances use a single repository, you can use the deploy tool, with the default settings, to deploy applications and other assets.

4.4 Summary of references

- WebSphere Application Server Community Edition V2.0 documentation
  http://publib.boulder.ibm.com/wasce/V2.0.0/en/index.html
  - Clustering
    http://publib.boulder.ibm.com/wasce/V2.0.0/en/clustering.html
- Apache HTTP Server Project
  http://httpd.apache.org/
- The Apache Tomcat Connector
  http://tomcat.apache.org/connectors-doc/
- Apache Geronimo v2.0 documentation
  http://cwiki.apache.org/GMOxDOC20/documentation.html
  - Multiple repositories
    http://cwiki.apache.org/GMOxDOC20/multiple-repositories.html

5

Configuration and administration

WebSphere Application Server Community Edition provides a JSR-168 portlet-based administrative console that allows you (the administrator) to accomplish administration and configuration tasks easily. Community Edition also provides command scripts for the convenience of more experienced administrators or those of you who manage multiple systems.

For a full list of command scripts provided by Community Edition, see the following Web pages:

- http://publib.boulder.ibm.com/wasce/V2.0.0/en/commands.html
- http://cwiki.apache.org/GMOxDOC20/tools-and-commands.html

In this chapter, we describe how to use the administrative console and the command scripts to administer and configure different aspects of the Community Edition.

5.1 Starting and stopping the application server

You can manage Community Edition application servers through command scripts, and in the case of stopping the application server, through the administrative console.

Command script location: The command scripts for Community Edition are in the *wasceHome*/bin directory.

5.1.1 Starting an application server

For the administrative console to be available, the application server must be running. You can start the server using command scripts.

If Community Edition is installed on a Windows operating system, you can also use the Start menu, by clicking **Start** → **All programs** → **IBM WebSphere** → **Application Server Community Edition** → **Start the server**.

Using command scripts

There are two command scripts that are provided to start the application server. These scripts are located in *wasceHome*/bin:

- **`startup.bat`** or **`startup.sh`**

  This is the primary method of starting the application server. The server's output is written to *wasceHome*/var/log/server.log.

  For information about startup options, see:

  `http://publib.boulder.ibm.com/wasce/V2.0.0/en/startup.html`

- geronimo.bat or geronimo.sh

  These commands allows you to run the application server in the Java Debugger, to enable remote debugging using the Java Platform Debug Architecture, or to start the application server with more verbose information.

  To use this command, the administrator must have one of the supported Java SDKs installed, and not the JRE only, to run the previously mentioned debuggers.

If you have problems starting the application server, see 12.3, “Server startup problems” on page 192.

5.1.2 Stopping the application server

You can stop the application server using the console or using the Windows Start menu.

Using the console

1. From the menu on the left side of the console, select **Server** → **Shutdown**.
2. A portlet window is displayed warning you that shutting down the server disables the console. To proceed, click the **Shutdown** button.
3. A confirmation message box is displayed. To continue the shutdown, click **OK**. To stop the shutdown, click **Cancel**.

Using the console to stop the server is particularly useful for remote application servers because you can access the console from a Web browser.

Using the Windows Start menu

To perform a shutdown on the application server from a Windows Start menu:

1. Click **Start** → **All programs** → **IBM WebSphere** → **Application Server Community Edition** → → **Stop the server**.
2. A command prompt window is opened showing the status of the application server's shutdown. Provide a user name and password that has the authorization to do this task. The default user name and password values are **system** and **manager** respectively.

Using a command script

Use the **shutdown.bat** or **shutdown.sh** command script to shutdown the application server. If you do not provide the user name and password (for example, system / manager) you are prompted for them.

For more information, see:

http://publib.boulder.ibm.com/wasce/V2.0.0/en/shutdown.html

Using Ctrl+C

Another way to perform a shutdown is using the Ctrl+C key stroke:

1. Locate the window that opened when you issued the **startup** command script.
2. Open that window, and press Ctrl+C simultaneously.

   The following message is printed, which means the beginning of server shutdown.

   ```
   [] received stop signal
   ```

5.2 Using the administrative console

The administrative console provides a GUI interface for you to administer the application server.

After the application server initializes successfully, you can access the console from a Web browser.

Two URLs are available:

- You can access the *Welcome portal* using one of the following URLs:
  - http://*host_name*:8080/
  - https://*host_name*:8443/ (for SSL enabled)

  The port numbers are different if you are not using the defaults.

  The welcome page (Figure 5-1 on page 50) displays a welcome message that includes the technology supported by Community Edition V2. The left panel contains links to update sites, technical documentation, and other useful links, including a link to the administrative console.

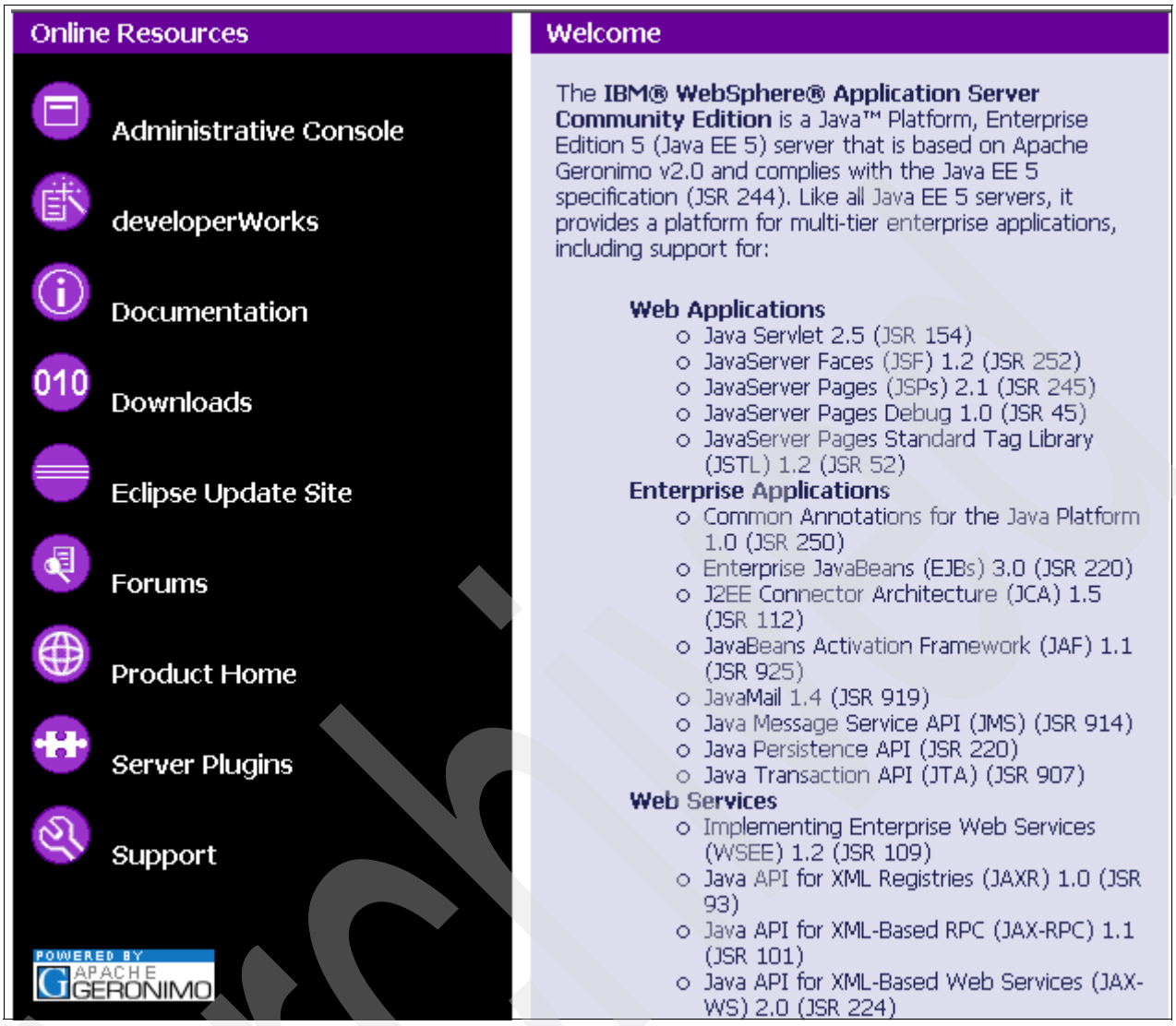


Figure 5-1 Community Edition Welcome page

- A direct link to the administrative console URL:
  - http://*host_name*:8080/console
  - https://*host_name*:8443/console (for SSL enabled)

Default login: The first page that is displayed when you access the administrative console is a login page. The default login is:

- username `system`
- password `manager`

Figure 5-2 on page 51 shows an example of the administrative console's welcome page.

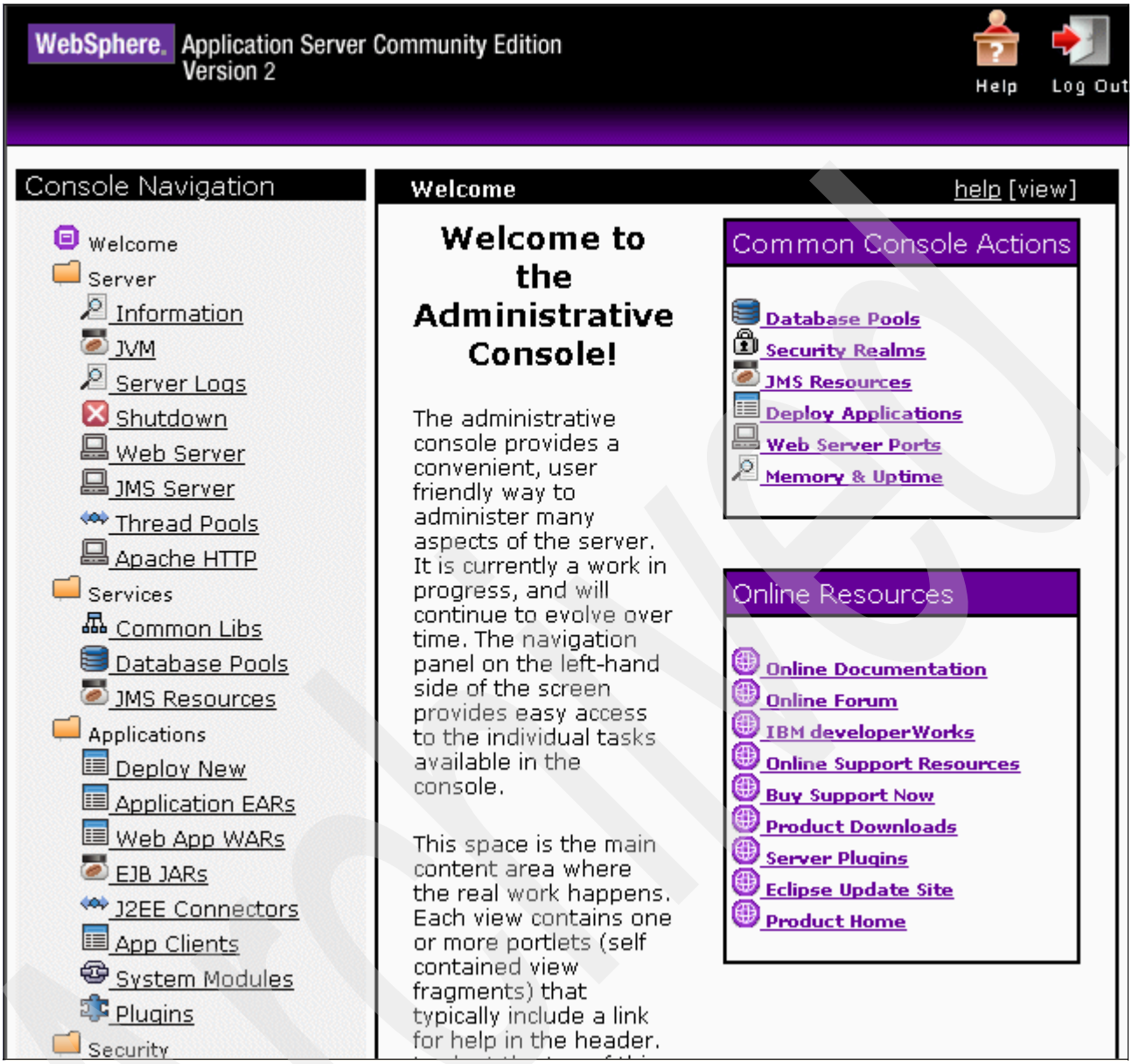


Figure 5-2 Administrative console

The console consists of three main areas:

- The banner area, at the top, contains links for support and to logout from the administrative console.
- The console navigation area, at the left, contains options (portlets) for administration of Community Edition. For more information about the portlets that are available in the console navigation area, see:

  `http://publib.boulder.ibm.com/wasce/V2.0.0/en/administrative-console.html`
- The work area contents change according to the selection you make in the console navigation area. Using the **Help** and **View** hyperlinks you can toggle between the two views for this page. View is the mode for administration and configuration. This is the default. Click **Help** to display information that aids you with using the portlet.

The console welcome page contains links to helpful sites, similar to the list that is displayed in the Community Edition Welcome page. Another group of links on this page provide shortcuts to common tasks that you can perform in the console.

5.2.1 Securing the administrative console

In the previous sections, we used the default user name and password to login to the administrative console. To configure users and groups who are authorized to work with the administrative console, click the **Console Realm** link in the console navigation area. This shows a list of defined users and groups, as shown in Figure 5-3.

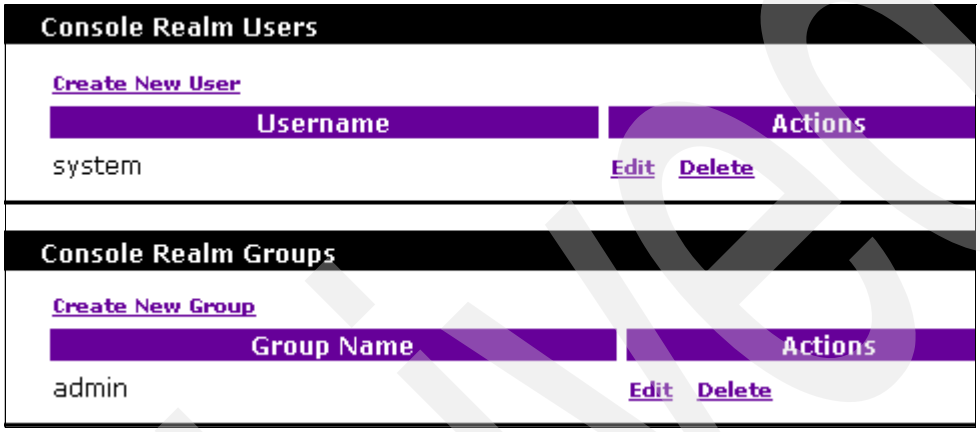


Figure 5-3 Managing the console realm

Links are available to:

- Create a new user.
- Create a new group.
- Edit a user. This allows you to change the password for the user.
- Edit a group. This allows you to add or remove users from a group and to edit the group.

  For a user to login to the administrative console or to execute administrative command scripts, the user must be in the *admin* group.

You can also manage administrative security by editing the following files:

- *wasceHome*/var/security/users.properties

  Add new user name and password pairs in the following format:

  ```
  username=password
  ```

- *wasceHome*/var/security/groups.properties

  Add the new user name to the admin group, and separate it from other users using a comma “,”.

Restart the application server so that the new password is encrypted and the new user addition can take effect.

5.3 Configuration using the config.xml file

An alternative to using the administrative console or command scripts for configuration is to manually edit the XML file that contains the configuration for the server. This method is best left to more experienced users. The configuration file is located in:

wasceHome/var/config/config.xml

To make configuration changes using the XML file:

1. Stop the application server.
2. Make a copy of the config.xml file as a backup.
3. Open the config.xml file in an editor, and make the required changes.
4. Start the application server so that the new configuration changes can take effect.

For more information about configuring Community Edition using the config.xml file, see:

http://publib.boulder.ibm.com/wasce/V2.0.0/en/xml-configuration.html

5.4 Configuring the Web container

The Web container in a Community Edition server is an integrated Tomcat server. The integration is done using GBeans that represent the component, container, connector, engine, host, valves, and realms.

There are some configuration differences between Community Edition and a standalone Tomcat server. The Tomcat configuration in Community Edition is driven by deployment plans as opposed to the standalone Tomcat configuration, which uses the /conf/server.xml file.

The Tomcat configuration is contained in a deployment plan in *wasceHome*/var/config/config.xml.

In this section, we discuss common configuration changes that you might want to make to the Web container.

5.4.1 Changing port numbers

The port configuration for the Web container is contained in the TomcatWebConnector and TomcatWebSSLConnector GBean configuration, which are located in *wasceHome*/var/config/config.xml. If you are running multiple servers on a system, or if these ports are in use by other applications, you can change the ports that Tomcat uses. Example 5-1 shows the Tomcat connector properties in config.xml.

Example 5-1 Tomcat connector properties in config.xml

```
<gbean name="TomcatResources" />
- <gbean name="TomcatWebConnector">
  <attribute name="host">${ServerHostname}</attribute>
  <attribute name="port">${HTTPPortPrimary + PortOffset}</attribute>
  <attribute name="redirectPort">${HTTPSPortPrimary + PortOffset}</attribute>
....
...
</gbean>
- <gbean name="TomcatWebSSLConnector">
```

```
  <attribute name="host">${ServerHostname}</attribute>
  <attribute name="port">${HTTPSPortPrimary + PortOffset}</attribute>
....
...
</gbean>
```

The default configuration uses properties defined in *wasceHome*/var/config/config-substitutions.properties to determine the values for the attributes. Example 5-2 shows the properties that are used.

Example 5-2 properties used

```
#PortOffset=0
ServerHostname=0.0.0.0
HTTPPortPrimary=8080
HTTPSPortPrimary=8443
```

The result is that 8080 is used for non-SSL communication. Port 8443 is used for SSL communication.

To change the ports that are used, you can use one of three methods:

- Uncomment PortOffset in the config-substitutions.properties file, and give it a value, for example, 10. This adds 10 to all of the ports that are defined in config.xml, which includes the Web connector ports. This is the recommended method if you have port conflicts with another Community Edition server.
- Change one or more port numbers by editing the config-substitutions.properties file.
- Use the administrative console to change port numbers. This assumes that the application server will start (and the console is available) and that the port to be changed is exposed by the console.

  In the Web Server portlet, use the **edit** link to change the port (for example, to change HTTP 8080 to 80). Stop and start the listener. If you are changing the HTTP port, login to the console using the HTTP's port (8443) because stopping the HTTP port disables the console, and you cannot restart the listener.

5.4.2 Configuring a connector

Community Edition provides pre-configured HTTP, HTTPS, and AJP connectors. Each connector listens on a specific port. You can add new connectors to listen on additional ports or modify connectors, for example to increase the number of threads that are used to handle incoming requests. You can configure connectors by editing the config.xml file or by using the administrative console.

Connectors are of one of the following types:

- Blocking I/O (BIO): This is the type of the server's initial configuration. This is appropriate if your server receives steady moderate traffic. It creates one thread per HTTP connection.
- Nonblocking I/O (NIO): This connector accepts multiple HTTP connections on a single thread. So in high volume traffic it does not require a large number of memory consuming threads, and there is no need to tune the thread pools.

- Apache Portable Runtime (APR): This connector uses the platform's native interfaces instead of Java interfaces. This is a more efficient connector but requires that the Apache Portable Runtime is installed.

To view, edit, or add the connectors from the console, select **Web Server** in the console navigation bar. The resulting page contains information about the existing connectors and has links for you to add new connectors, as shown in Figure 5-4.

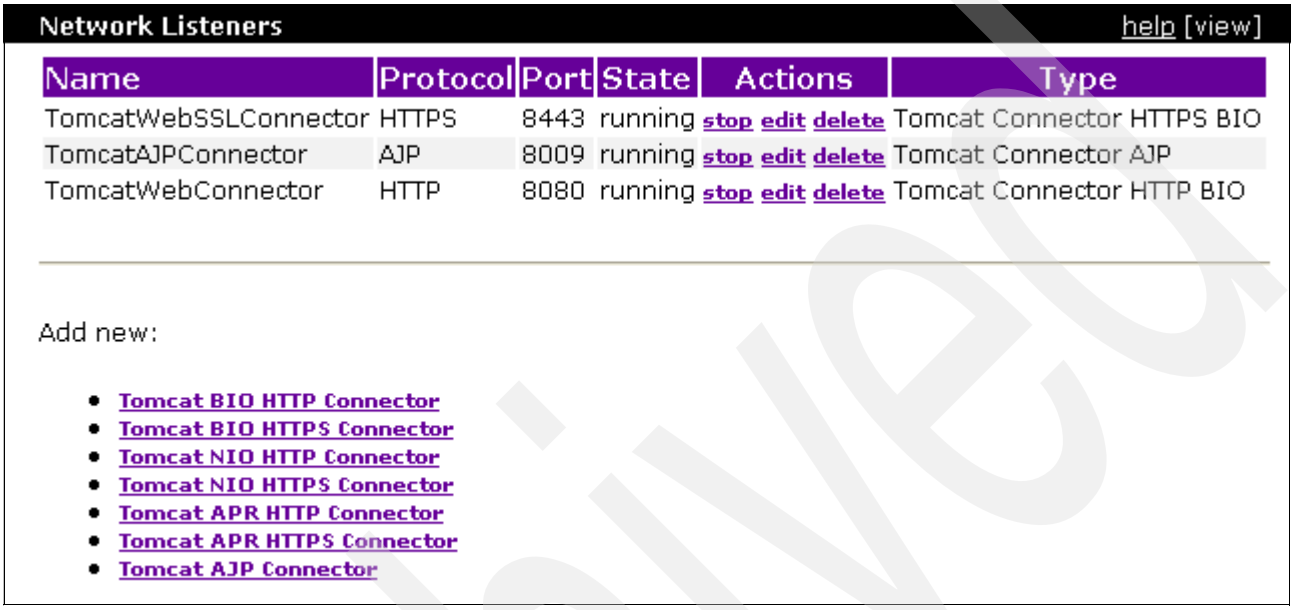


Figure 5-4 Connectors page on Web console

For a complete reference about each parameter of a connector, check the Tomcat configuration reference:

```
http://tomcat.apache.org/tomcat-6.0-doc/config/http.html
http://tomcat.apache.org/tomcat-6.0-doc/config/ajb.html
```

Check the following Web sites for more information about configuring HTTP, HTTPS, or AJP connectors:

```
http://publib.boulder.ibm.com/wasce/V2.0.0/en/http-connector.html
http://publib.boulder.ibm.com/wasce/V2.0.0/en/https-connector.html
http://publib.boulder.ibm.com/wasce/V2.0.0/en/ajb-connector.html
```

5.4.3 Configuring virtual hosts

Using virtual hosts allows you to host multiple Web sites, each with its own domain name on one server. To control which Web application is accessed on a given domain, specify the virtual host for the application in a <container-config> element in its deployment plan.

The Community Edition Tomcat configuration has a single virtual host named TomcatHost as its default host. All inbound requests are sent to that host. To modify that virtual host configuration or define new virtual hosts you must update the config.xml file, which is located in *installDir*/var/config/config.xml.

Information about using and configuring virtual hosts is at:

```
http://publib.boulder.ibm.com/wasce/V2.0.0/en/virtual-host.html
```

Configuring the default virtual host

The default virtual host, TomcatHost, is not included in the config.xml file. To override its configuration, add the GBean configuration, shown in Example 5-3, to the config.xml file.

Example 5-3 Overriding default virtual host configuration

```
<module name="org.apache.geronimo.configs/tomcat6/2.0.1/car">\
...
<gbean name="TomcatHost">
   <attribute name="initParams">
      name=HOST_NAME
   </attribute>
   <attribute name="aliases">LIST</attribute>
</gbean>
</module>
```

- HOSTNAME is the host name that is associated with the default virtual host (for example, www.ibm.com)
- LIST is a comma, separated list with no white spaces of the alias names of this host name (for example, www3.ibm.com,develperworks.ibm.com)

Adding an additional virtual host

Example 5-4 shows how to add a virtual host in config.xml.

Example 5-4 XML configuration to add a new virtual host

```
<gbean gbeanInfo="org.apache.geronimo.tomcat.HostBean"
name="org.apache.geronimo.configs./tomcat6/2.0.1/car,j2eeType=Host,name=BEANNAME">
   <attribute name="className">org.apache.catalina.core.StandardHost</attribute>
   <attribute name="initParams">
      name=HOSTNAME
   </attribute>
   <attribute name="aliases">LIST</attribute>
</gbean>
```

- BEANNAME is the name of this gbean (for example, VirtualHost01). This name must be unique among your gbeans.
- HOSTNAME is the host name that is associated with the default virtual host (for example, www.ibm.com).
- LIST is a command-separated list with no white spaces of the alias names of this host name (for example, www3.ibm.com,develperworks.ibm.com).

5.4.4 Valves management

Valves are Java classes that are inserted in the request processing pipeline of Tomcat. It is executed as a part of Tomcat's servlet container and is independent of any Web application. Valves are executed in a chain, where the last link in the chain is the Web application itself.

Information about valves is at:

http://publib.boulder.ibm.com/wasce/V2.0.0/en/valves.html

This site provides examples of the following common changes to a valve chain:

- Removing the Access Log valve
- Adding the Single Sign-on valve to the initial valve chain

- Replacing the Access Log valve
- Adding a valve chain to a Tomcat host

To configure a valve chain using config.xml, find the GBean configuration with name=”TomcatEngine”. Use the valve configuration template in Example 5-5 to update it.

Example 5-5 Valve chain configuration

```
<gbean name="TomcatEngine">
    <attribute name="initParams">name=WASCE20</attribute>
    <reference name="TomcatValveChain">
        <pattern>
            <name>NextValveInChain_0</name>
        </pattern>
    </reference>
</gbean>
<gbean gbeanInfo="org.apache.geronimo.tomcat.ValveGBean"
    name="org.apache.geronimo.configs/tomcat6/2.0.1/car?
    ServiceModule=org.apache.geronimo.configs/tomcat6/2.0.1/car,
    j2eeType=GBean,name=NextValveInChain_0">
    <attribute name="className">class</attribute>
    <attribute name="initParams">parms</attribute>
    <reference name="TomcatValveChain">
        <pattern>
            <name>NextValveInChain_1</name>
        </pattern>
    </reference>
</gbean>
...
<gbean gbeanInfo="org.apache.geronimo.tomcat.ValveGBean"
    name="org.apache.geronimo.configs/tomcat6/2.0.1/car?
    ServiceModule=org.apache.geronimo.configs/tomcat6/2.0.1/car,
    j2eeType=GBean,name=NextValveInChain_N">
    <attribute name="className">class</attribute>
    <attribute name="initParams">parms</attribute>
</gbean>
```

The reference element, where name="TomcatValveChain", has the link to the name of the next valve of the chain. The name of the valves are defined in the name attribute in the gbean element. The other parameters are:

- class: the class name of the valve.
- parms: is a white space, separated list of valve parameters, where each parameter has the form *key=value*.

5.4.5 Lifecycle listener management

Lifecycle listeners allow you to synchronize life cycle events between your server and other business systems that are outside the server, for example, you might want to automatically start and stop a database manager when the server starts and stops or clean up logs when a server stops.

You can associate lifecycle listeners with the following:

- Tomcat Web container
- Tomcat engine
- Tomcat host
- Web application context

You can add listeners to container life cycle events by creating a Java class that implements the lifecycle listener interface org.apache.catalina.LifecycleListener.

You can deploy a lifecycle listener on the server by configuring the config.xml file or by defining a lifecycle listener in the geronimo-web.xml file of your applications. For more information about lifecycle listeners see:

http://publib.boulder.ibm.com/wasce/V2.0.0/en/lifecycle-listeners.html

5.5 Configuring application security

Security is a key element in every enterprise application. In this section, we discuss the options for application security in Community Edition.

5.5.1 Security realm management

Security realms protect Web resources by defining security constraints and mapping them to security roles that are defined in the Web application. Community Edition provides different kinds of realms. We go through each realm in this section.

By default, you have a geronimo-admin properties file realm defined, which is initially set up for administrative console security but can be used by applications for security. However, most applications that require login use other security realms, such as an LDAP server or a database.

You can manage this realm by selecting the **Security Realms** portlet in the administrative console (Figure 5-5). You have the option to edit the realm and to view information about how to use the realm in an application. You also have the option to define a new security realm.

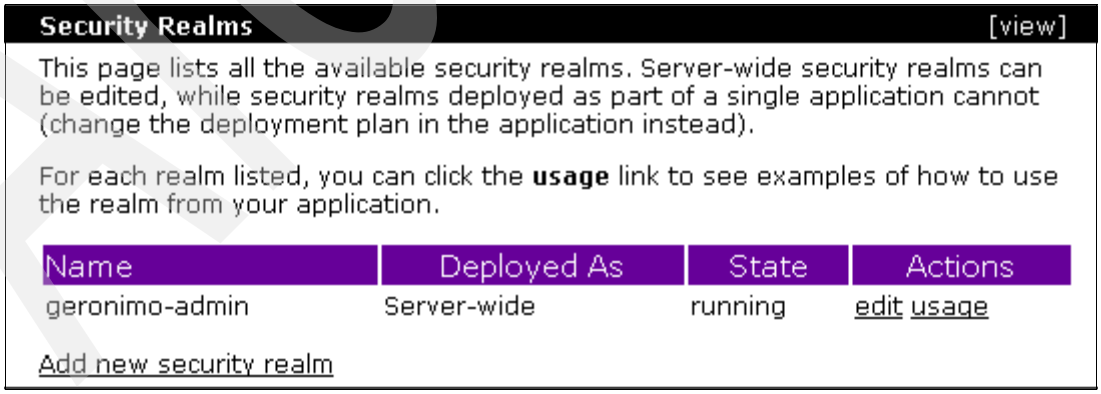


Figure 5-5 Default security realm

To remove a security realm, use the **deploy** command with the **undeploy** option. Make sure sure that you undeploy all the applications that depend on that realm first.

Properties file realms

Properties file realms are the most basic realms to use. They depend on two properties files, one that defines users and one that defines groups. The user, password, and group information is entered into the files whose locations are defined in the security realm definition.

To create a properties file realm:

1. Create two properties files and populate them with the initial user and group information, for example:
   - A file named *wasceHome*/var/security/my-users.properties. Entries in the file have the format username=*password*, for example:

     ```
     user1=password1
     user2=password2
     ```

   - A file named *wasceHome*/var/security/my-groups.properties. Entries in the file have the format group=*user,user,user*, for example:

     ```
     group1=user1,user2
     ```

2. In the console, select **Security Realms**.
3. Click **Add new security realm**.
4. Enter the a name for the realm, and select the **Properties File Realm** type, as shown in Figure 5-6. Click **Next**.

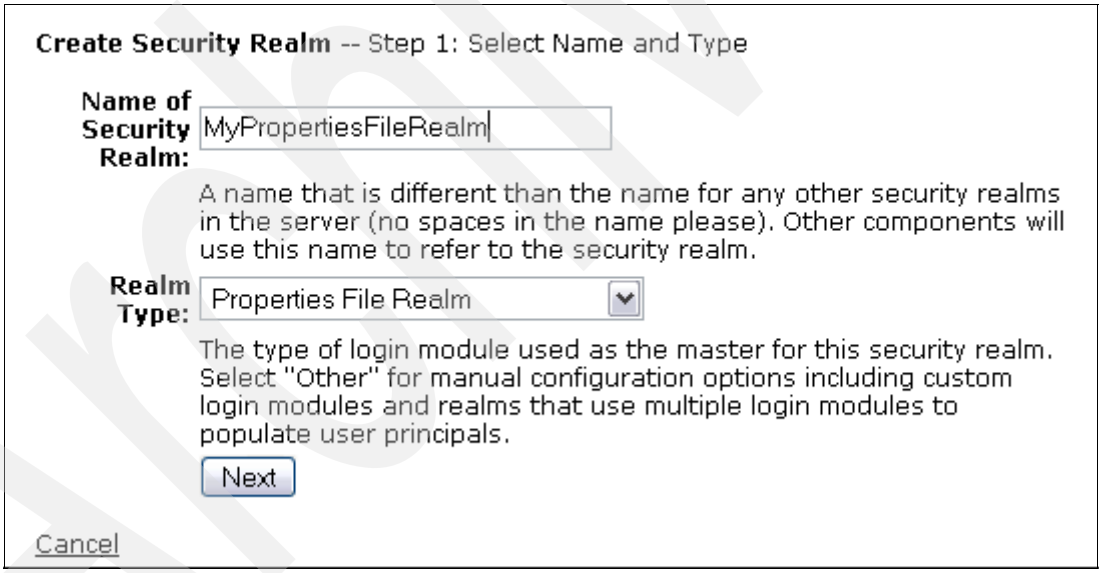


Figure 5-6 Create a new properties file realm

5. A new view is displayed, where you provide the location of the users and groups' properties files, digest algorithm and the digest encoding, as shown in Figure 5-7 on page 60. Click the **Next** button.

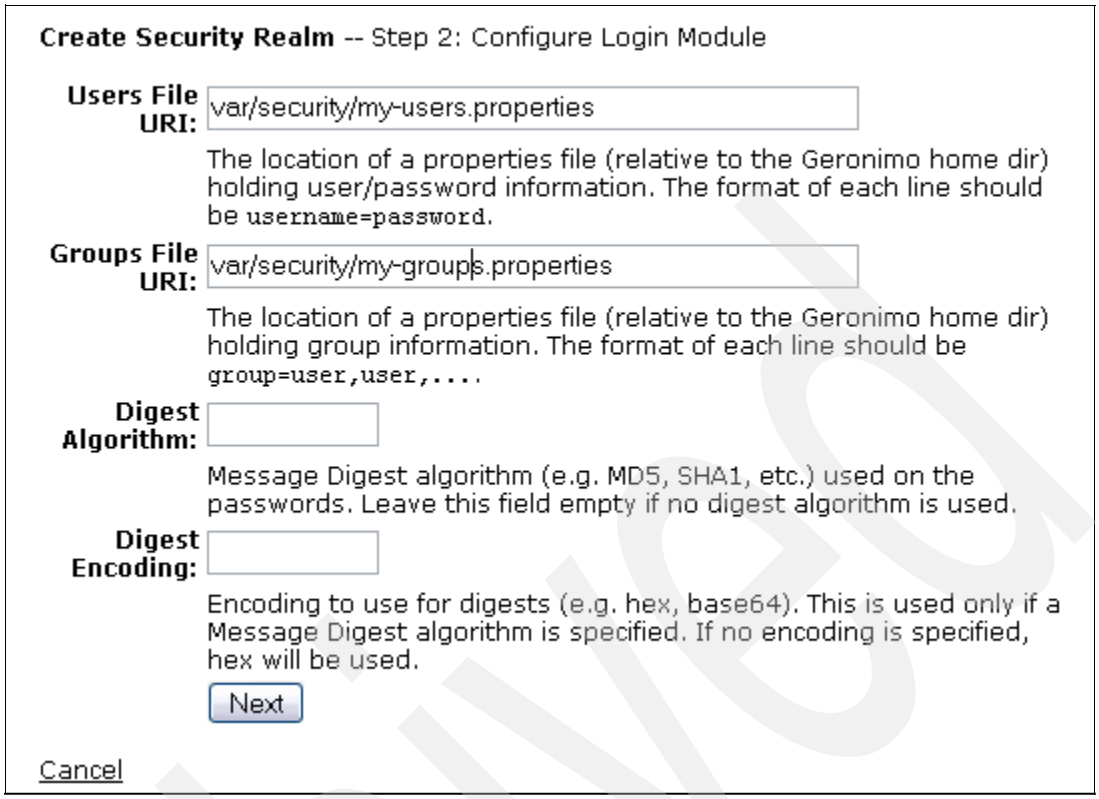


Figure 5-7 Defining the realm properties

6. The next page allows you to enable options for the realm. You can also test the login and view the realm's deployment plan, as shown in Figure 5-8 on page 61. Click **Skip Test and Deploy** to deploy the realm.

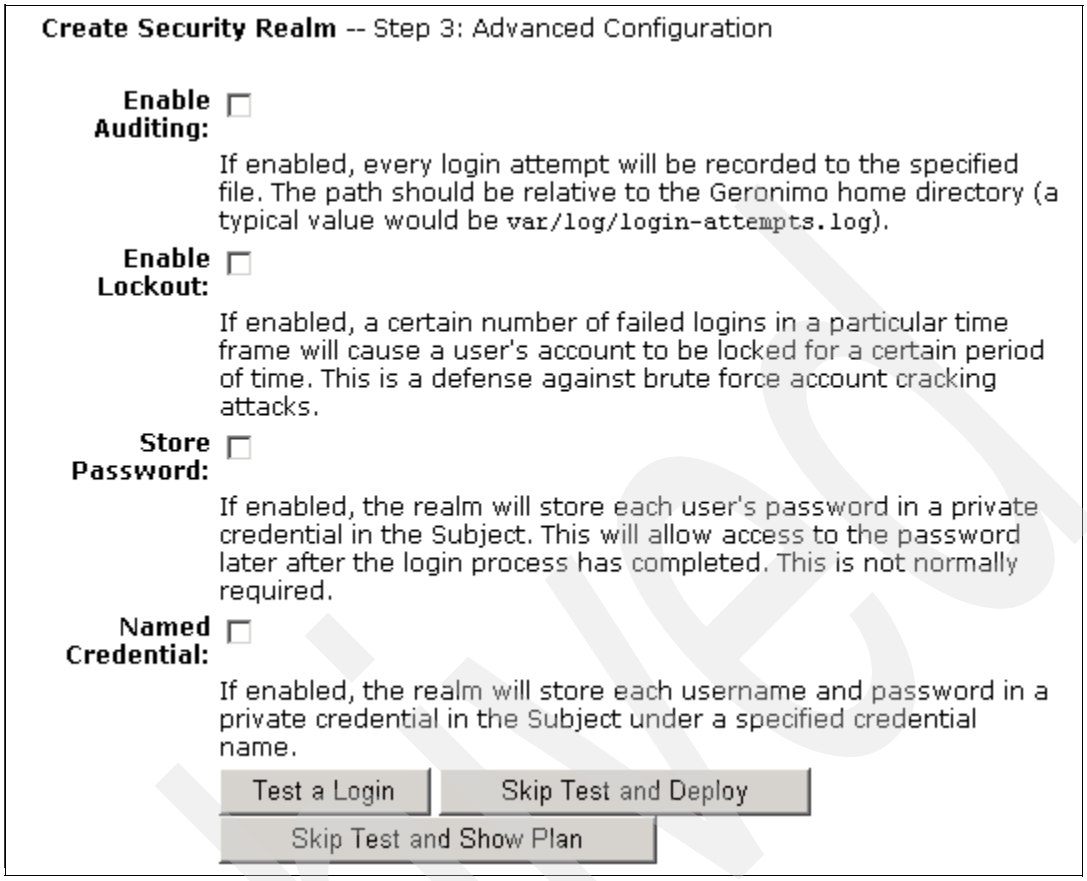


Figure 5-8 Defining the realm advanced properties

The realms listing view is displayed showing the new realm listed.

7. Click the **usage** link to see the entries that are required in the deployment plans to use this security realm.

Certificate properties file realms

The certificate properties file realm provides a trusted way to authenticate users. To use this kind of realm, you must first:

1. Set up a certificate trust on the server and client.
2. Configure an HTTPS connector with a trusted SSL certificate.

Follow the steps in “Properties file realms” on page 59 to create the realm, but select **Certificate Properties File Realm** as the realm type. We discuss certificate trusts and SSL in 5.5.2, “Secure communications using SSL” on page 62.

LDAP realms

LDAP servers store user information across the organization, so that multiple applications can use it in secure way. Configuring an LDAP server to work with Community Edition depends on the LDAP server itself. We recommend the following LDAP servers for use with Community Edition, although any LDAP server should work:

- Apache Directory Server (ApacheDS)
- IBM Tivoli Directory Server

- Microsoft Active Directory Server
- OpenLDAP server
- Sun ONE Directory Server

Database realms

With database realms, you can use a database for user name and password retrievals. To test a database realm, the database must exist before you create the realm.

Custom realms

You can make your own customized realm type. The custom realm must implement the javax.security.auth.spi.LoginModule.

5.5.2 Secure communications using SSL

SSL connectors can be used for secured communication. Encryption prevents the data from being observed or modified while it travels through the network.

A *keystore file* contains certificates that are used to encrypt the data that is exchanged using SSL. A certificate contains a public key and a private key, which are used to encrypt the data that is exchanged over the SSL connection. The certificate also contain a set of credentials that identify the server and the company that is associated with the server. Whether or not these credentials can be trusted depends on whether or not the client can trust the certificate authority that digitally signed the certificate.

You can administer keystores from the administrative console using the **Keystores** portlet. Manage certificate authorities using the **Certificate Authority** portlet.

Note: For information about using the security features that Community Edition provides, see:

- http://publib.boulder.ibm.com/wasce/V2.0.0/en/security-realms.html
- http://publib.boulder.ibm.com/wasce/V2.0.0/en/ssl-certificates.html
- http://publib.boulder.ibm.com/wasce/V2.0.0/en/trust.html
- http://cwiki.apache.org/GMOxDOC20/configuring-security.html
- http://www.ibm.com/developerworks/websphere/library/techarticles/0702_krishnasamy/0702_krishnasamy.html#sec5
- http://www.ibm.com/developerworks/websphere/library/techarticles/0702_krishnasamy/0702_krishnasamy.html#sec6
- http://www.ibm.com/developerworks/websphere/library/techarticles/0606_chillakuru/0606_chillakuru.html
- http://www.ibm.com/developerworks/websphere/library/techarticles/0710_vamsi/0710_vamsi.html?S_TACT=105AGX10&S_CMP=WASCE
- http://www.ibm.com/developerworks/websphere/library/techarticles/0709_vamsi/0709_vamsi.html?S_TACT=105AGX10&S_CMP=WASCE

5.6 Using proxies

If the Community Edition server resides behind a firewall that has a proxy to facilitate reaching resources outside of that firewall, configure Community Edition to use that proxy

before being able to go through it. See the following article for more information about the configuration options of using a proxy with the Community Edition:

http://publib.boulder.ibm.com/wasce/V2.0.0/en/proxy-configuration.html

5.7 Summary of references

- *Migrating from WebSphere Application Server Community Edition to WebSphere Application Server*, SG24-7433.
- WebSphere Application Server Community Edition V2.0 documentation

  http://publib.boulder.ibm.com/wasce/V2.0.0/en/index.html
  - Index of commands

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/commands.html
  - startup command

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/startup.html
  - shutdown command

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/shutdown.html
  - Administrative console

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/administrative-console.html
  - XML configuration

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/xml-configuration.html
  - Configuring an HTTP connector

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/http-connector.html
  - Configuring an HTTPS connector

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/https-connector.html
  - Configuring an AJP connector

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/ajp-connector.html
  - Configuring a virtual host

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/virtual-host.html
  - Managing valves

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/valves.html
  - Tomcat lifecycle listeners

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/lifecycle-listeners.html
  - Managing Security Realms

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/security-realms.html
  - Managing SSL certificates

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/ssl-certificates.html
  - Managing trust

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/trust.html
  - Proxy configuration

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/proxy-configuration.html

- Apache Geronimo v2.0 documentation

  http://cwiki.apache.org/GMOxDOC20/documentation.html

  - Tools and commands

    http://cwiki.apache.org/GMOxDOC20/tools-and-commands.html

  - Configuring security

    http://cwiki.apache.org/GMOxDOC20/configuring-security.html

- Apache Tomcat 6.0 documentation

  http://tomcat.apache.org/tomcat-6.0-doc/index.html

  - The HTTP connector

    http://tomcat.apache.org/tomcat-6.0-doc/config/http.html

  - The AJP connector

    http://tomcat.apache.org/tomcat-6.0-doc/config/ajp.html

- Advanced administration in WebSphere Application Server Community Edition: Part 1: Working with database realms and security elements

  http://www.ibm.com/developerworks/websphere/library/techarticles/0702_krishnasamy/0702_krishnasamy.html

- Client authentication using digital certificates in WebSphere Application Server Community Edition

  http://www.ibm.com/developerworks/websphere/library/techarticles/0606_chillakuru/0606_chillakuru.html

- Set up a public key infrastructure with WebSphere Application Server Community Edition V2.0

  http://www.ibm.com/developerworks/websphere/library/techarticles/0710_vamsi/0710_vamsi.html?S_TACT=105AGX10&S_CMP=WASCE

- Configuring Web application security in WebSphere Application Server Community Edition V2.0

  http://www.ibm.com/developerworks/websphere/library/techarticles/0709_vamsi/0709_vamsi.html?S_TACT=105AGX10&S_CMP=WASCE

6

Tuning

In this chapter, we provide an overview of tuning for Community Edition servers. We describe approaches to monitor Community Edition statistics and how to make changes to improve the performance of the whole system.

The following topics are included in this chapter:

6.1 Tuning overview

Tuning is a collection of techniques that you can apply to improve the system performance. Performance tuning usually involves the following iterative multi-step cycle:

1. Measure the performance before you apply any changes to the system by using the appropriate monitoring tools, called *profilers.*
2. Identify possible *bottlenecks,* where a single component leads to a performance deterioration of the whole system.
3. Make changes to remove the bottlenecks.
4. Return to step 1 to examine any relevant improvements.

The cycle ends when the system reaches an acceptable behavior, according to a series of numeric values that you established before you started the process.

In the following section, we describe the tuning activities that are available to improve the performance of the Community Edition server.

6.2 System tuning

The system tuning deals with all available sets of parameters and strategies to optimize the operating systems on which the Community Edition software components will run.

For a complete reference about system tuning, visit the following Web sites:

- For UNIX systems:

  `http://publib.boulder.ibm.com/wasce/V2.0.0/en/tuning-unix.html`
- For Windows systems:

  `http://publib.boulder.ibm.com/wasce/V2.0.0/en/tuning-windows.html`

6.3 Java Virtual Machine tuning

In this section, we discusses the common Java Virtual Machine (JVM) parameters that you can adjust to improve the performance of a Community Edition server.

6.3.1 Monitoring the memory usage

It is useful to monitor the current server memory usage before you apply any changes to the JVM parameters.

To analyze the memory, login to the administrative console, and select **Information** in the console navigation area, which opens the server information page. From this page, you can check the following relevant statistics:

- Current memory used
- Most memory used
- Total memory allocated

You can also view a real time graphic visualization of current server memory usage, as shown in Figure 6-1.

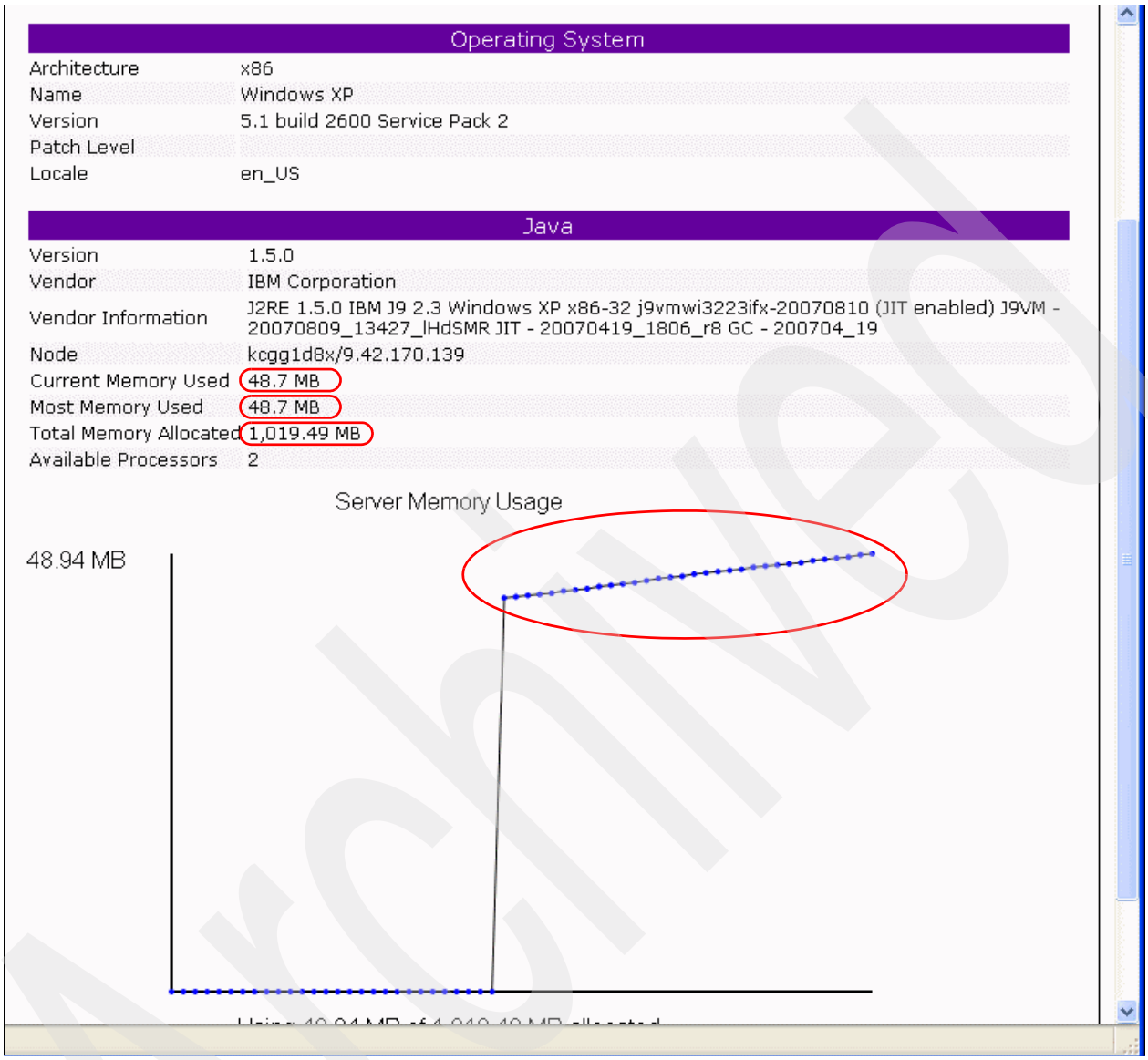


Figure 6-1 Server Memory Usage dynamic graphic

Note: This feature currently does not work in Internet Explorer in a Windows environment because of some interaction problems with the Adobe® SVG Viewer you should install. The feature works in the Mozilla Firefox Web browser. You can find more about the related JIRA issue at:

https://issues.apache.org/jira/browse/GERONIMO-1939

JConsole

Using the JConsole graphical tool, you can monitor the behavior of both the JVM and the Java applications. It provides useful information about performance and resources utilization of the JVM.

The JConsole can monitor:

- Local applications, running on the same machine as jconsole.
- Remote applications.

Note: JConsole is included in the IBM JVM SDK, but it is experimental and unsupported.

Information about the JConsole tool is at:

`http://java.sun.com/j2se/1.5.0/docs/guide/management/jconsole.html`

To monitor the Community Edition behavior, connect the JConsole to the running JVM, and perform the following steps:

1. Start the Community Edition server.
2. Open a prompt command. Make sure the *java_home*/bin directory of the IBM JVM is in the path. Start the tool by typing **`jconsole`**. The graphic console appears, as shown in Figure 6-2.

Figure 6-2 JConsole

3. Accept the default settings, and click **Connect**. JConsole starts with the Summary tab, as shown in Figure 6-3 on page 69.

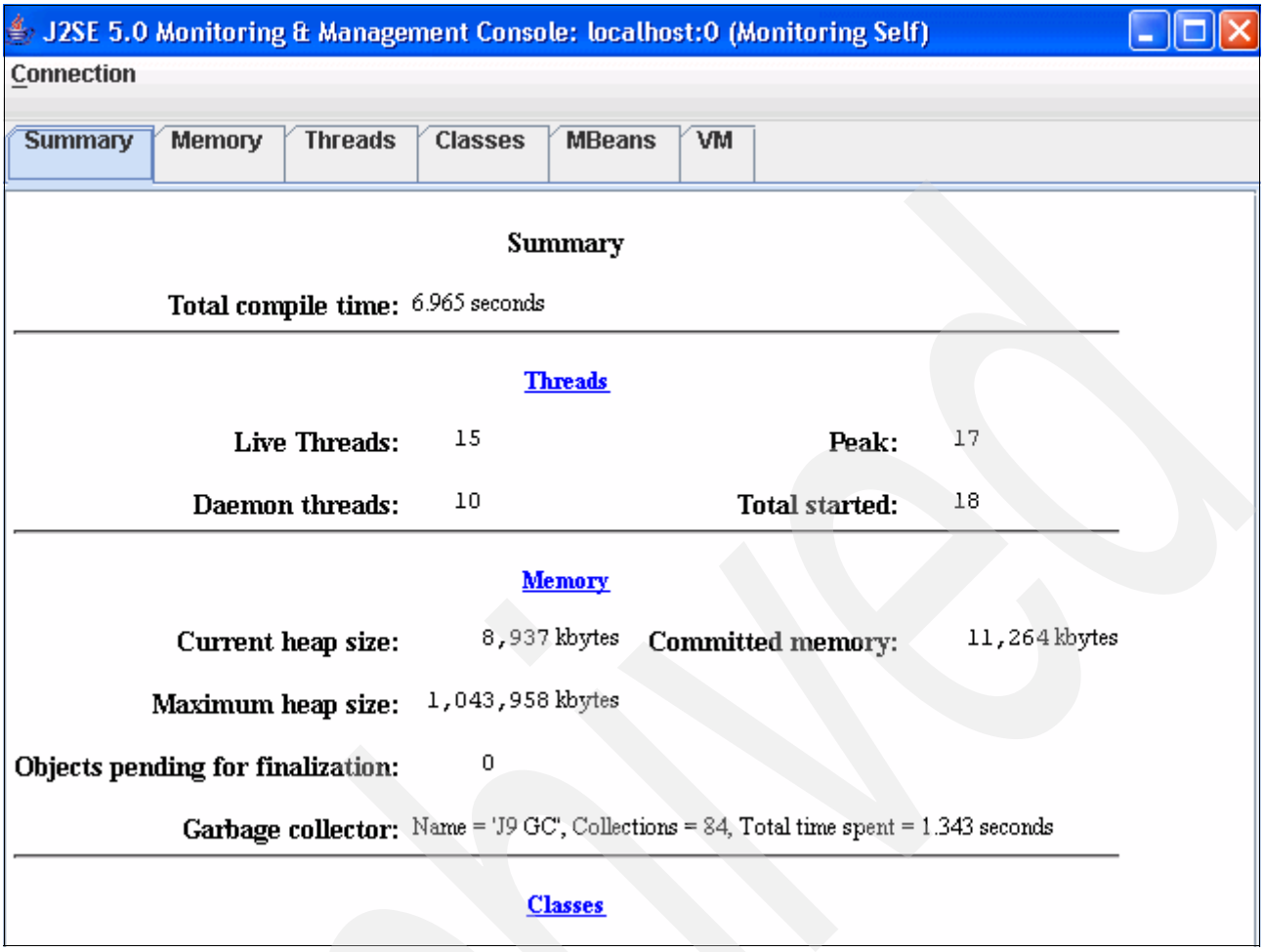


Figure 6-3 JConsole Summary tab

The Summary tab displays key information about the JVM, such as memory consumption, thread usage, and class loading.

More details are available by selecting the other tabs, for example, Figure 6-4 on page 70 shows the Memory tab.

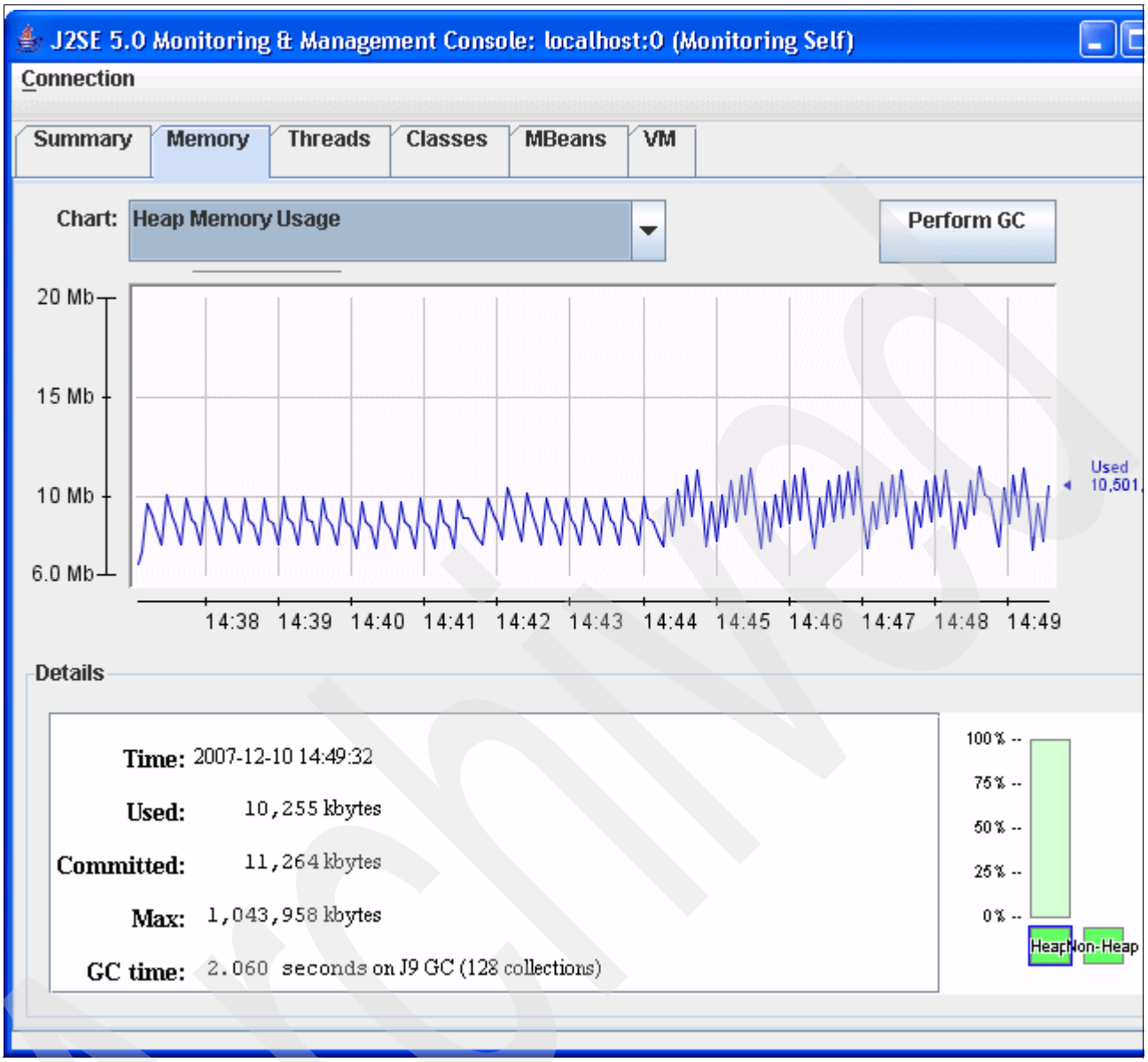


Figure 6-4 JConsole memory tab

The Memory tab shows a chart that displays information about the history of memory usage for different memory pools.

6.3.2 Tuning the Java heap size

The Java heap is the runtime space from which Java objects are allocated. The Garbage Collector (gc) starts to claim those objects that can no longer be referenced from any other living object. Because the mechanism is an extensive operation, it can slow application execution significantly.

The heap size determines the time and the frequency of the gc's activity. The goal of properly tuning the heap size is to minimize the overhead of gc to improve the server response and the throughput.

A proper balance must be found to avoid the following two opposite scenarios:

- A value too high results in high memory consumption and a low number of major slow garbage collection cycles.
- A value too low results in a high number of minor fast garbage collection cycles.

The IBM JDK allows you to change several parameters that are related to the heap size.

Initial heap size

The initial heap size specifies the initial amount of memory that is allocated in the heap when the JVM starts.

To set the initial heap size, include the following option in the JAVA_OPTS environment variable, where *mem* is the heap size in megabytes:

```
-Xms<mem>m
```

Note: For information about the JAVA_OPTS environment variable, see:

http://publib.boulder.ibm.com/wasce/V2.0.0/en/javaopts.html

You can add the JAVA_OPTS variable to the following files:

- *wasceHome*/bin/geronimo.bat(sh)

  The geronimo executable is called by startup.bat(sh)
- *wasceHome*/bin/setenv.bat(sh)

Maximum heap size

The maximum heap size specifies the maximum amount of memory that the JVM can use.

To set maximum heap size, include the following option in the JAVA_OPTS environment variable, where *mem* is maximum heap size in megabytes:

```
-Xms<mem>m
```

Default settings for heap size

Table 6-1 shows the IBM JVM default heap size values, both for Linux and for Windows runtime implementations.

Table 6-1 IBM JVM default heap size values

JVM setting	Linux	Windows
Initial Heap Size (-Xms)	4 MB	4 MB
Maximum Heap Size (-Xms)	Half the real memory with a minimum of 16 MB and a maximum of 512 MB	Half the real memory with a minimum of 16 MB and a maximum of 2 GB

Heap with large pages

It is also possible to configure the IBM JVM to allocate the heap using large pages. Use the following option:

```
-Xlp
```

For more information about these settings, see:

- For JVM tuning that is related to the Community Edition server:

  http://publib.boulder.ibm.com/wasce/V2.0.0/en/tuning-java.html

- For a complete reference about IBM JVM tuning and diagnostic tools:

  http://publib.boulder.ibm.com/infocenter/javasdk/v5r0/index.jsp

- For more information about JVM tuning issues, search the developerWorks Java technical library for performance related articles:

  http://www.ibm.com/developerworks/views/java/libraryview.jsp

6.4 Community Edition tuning

In this section, we describe parameters that you can monitor and modify to improve the performance of Community Edition.

6.4.1 Thread pool sizes

The thread pool size determines the processing capacity of Community Edition server, so its fine tuning is a requirement to achieve the best performance.

Community Edition allows the customization of two thread pool sizes:

- *DefaultThreadPool* size specifies the maximum number of threads possible in the server. Its default value is 500.
- *ConnectorThreadPool* size specifies the pool size of the connector threads in the server and is a subset of DefaultThreadPool. Its default value is 30.

6.4.2 Monitoring the thread pools

To choose the optimum value for the sizes, it is possible to monitor the current utilization of thread pools using the administrative console:

1. Select the **Thread Pools** portlet in the console to open the configuration page, which we show in Figure 6-5.

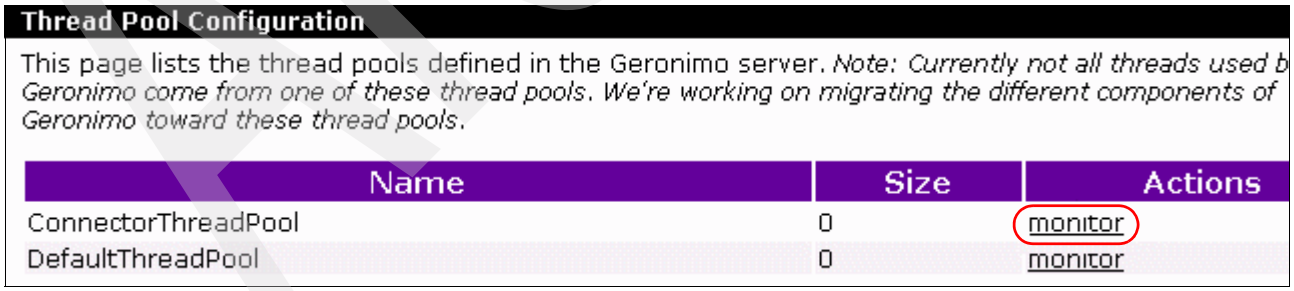


Figure 6-5 The thread pool configuration page

2. To monitor statistics that are related to a particular thread pool, click the **monitor** link corresponding to that pool in the Actions column. A page similar to Figure 6-6 on page 73 opens.

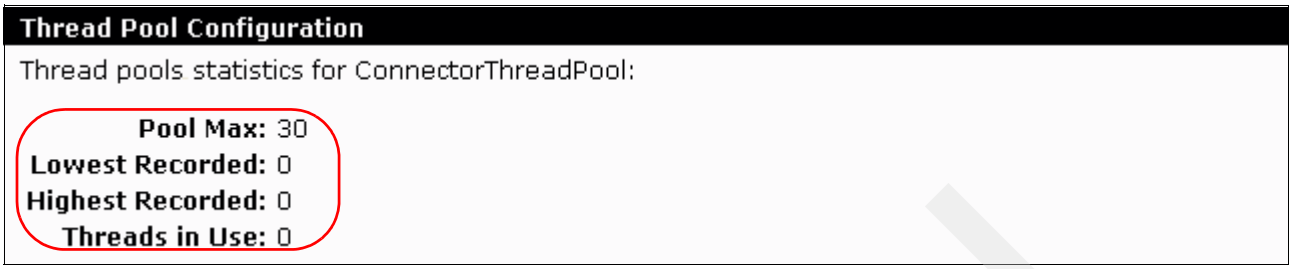


Figure 6-6 Thread pool statistics pane

You can monitor the followings statistics:

PoolMax	The maximum number of threads that Community Edition can instantiate in a thread pool.
Lowest Recorded	The lowest number of threads used up to now.
Highest Recorded	The highest number of threads used up to now.
Threads in Use	The number of threads currently in use.

6.4.3 Configuring the thread pool size

Choosing the optimum value for the thread pool size is important to sucessfully configure the Community Edition for best performance:

- Setting a too low value reduces the throughput because of an increased wait time for a request to be served.
- Setting a too high value reduces the CPU efficiency because of frequent context switching among the threads.

You must find the proper balance.

Modifying the configuration

You change thread pool sizes by updating the configuration files, which you cannot do using the adminitrative console.

To apply the required changes:

1. Stop the Community Edition server.
2. Make a backup of the *wasceHome*/var/config/config.xml file.
3. Open the configuration file with a text editor.
4. In the content, find the following line relative to the existing rmi-naming module configuration

   ```
   <module name="org.apache.geronimo.configs/rmi-naming/2.0.1/car"
   ```

5. Insert the following <gbean> element inside the <module> element, where *PoolSize* is the value you desire:

   ```
   <gbean name="DefaultThreadPool">
     <attribute name="keepAliveTime">5000</attribute>
     <attribute name="poolSize">PoolSize</attribute>
   </gbean>
   ```

Note: If the <module> element is a singleton element, you must remove the trailing "/" and add the closing </module> at the end.

6. Find the following line relative to the existing j2ee-server module configuration:

   ```
   <module name="org.apache.geronimo.configs/transaction/2.0.1/car"
   ```

7. Insert the following <gbean> element inside the <module> element>, where *PoolSize* is the value you desire:

   ```
   <gbean name="ConnectorThreadPool">
     <attribute name="keepAliveTime">5000</attribute>
     <attribute name="poolSize">PoolSize</attribute>
   </gbean>
   ```

8. Save the changes, and start the Community Edition server.
9. Verify your changes by opening the ConnectorThreadPool page and the DefaultThreadPool page as described in 6.4.2, “Monitoring the thread pools” on page 72.

For more information about the Community Edition tuning and thread pool size handling, see:

http://publib.boulder.ibm.com/wasce/V2.0.0/en/tuning-the-server.html

6.5 Summary of references

- WebSphere Application Server Community Edition V2.0 documentation

  http://publib.boulder.ibm.com/wasce/V2.0.0/en/index.html

  - Tuning UNIX

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/tuning-unix.html

  - Tuning Windows

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/tuning-windows.html

  - The JAVA_OPTS environment variable

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/javaopts.html

  - Tuning Java

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/tuning-java.html

  - Tuning the server

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/tuning-the-server.html

- Geronimo: Server Info portlet does not display the 'Server Memory Usage' live graph on Internet Explorer

  https://issues.apache.org/jira/browse/GERONIMO-1939

- Using jconsole

  http://java.sun.com/j2se/1.5.0/docs/guide/management/jconsole.html

- Java Diagnostics Guide 5.0

  http://publib.boulder.ibm.com/infocenter/javasdk/v5r0/index.jsp

- developerWorks Java technical library for performance related articles

  http://www.ibm.com/developerworks/views/java/libraryview.jsp

7

Messaging

Being a Java EE5 certified Application server, Community Edition fully supports JMS messaging. In this chapter, we describe how to configure different JMS providers for Community Edition.

This chapter contains the following topics:

7.1 JMS support in Community Edition

A *JMS provider* acts as an intermediary between message producers and consumers. You need a JMS provider for message-driven beans (MDBs) and for sending and receiving messages. A JMS provider is configured in Community Edition as a resource adapter module. Any JMS provider can be integrated with Community Edition by deploying the resource adapter for that JMS provider.

The J2EE Connector Architecture is a standard way of connecting J2EE applications and Enterprise Information Systems (EIS), such as WebSphere MQ or DB2. The architecture of Community Edition is such that you can configure any JMS provider for use with it by deploying the resource adapter for that message broker.

By default, Community Edition includes Apache ActiveMQ as the JMS provider. When you start a server, the ActiveMQ provider is automatically available. You can also configure a server to use other JMS providers, including WebSphere MQ.

A *JMS resource group* is a resource adapter module that binds together the related connection factories, queues, and topics. To create and access JMS resources, such as queues, topics, and connection factories in Community Edition, you have to create a JMS resource group.

7.1.1 Managing JMS resources in the administrative console

To view the available JMS providers and to manage connectors from the console, select the **JMS Server** portlet. You can also use this portlet to add new connectors.

Figure 7-1 shows the default JMS server configuration that is available when you install and start a Community Edition server.

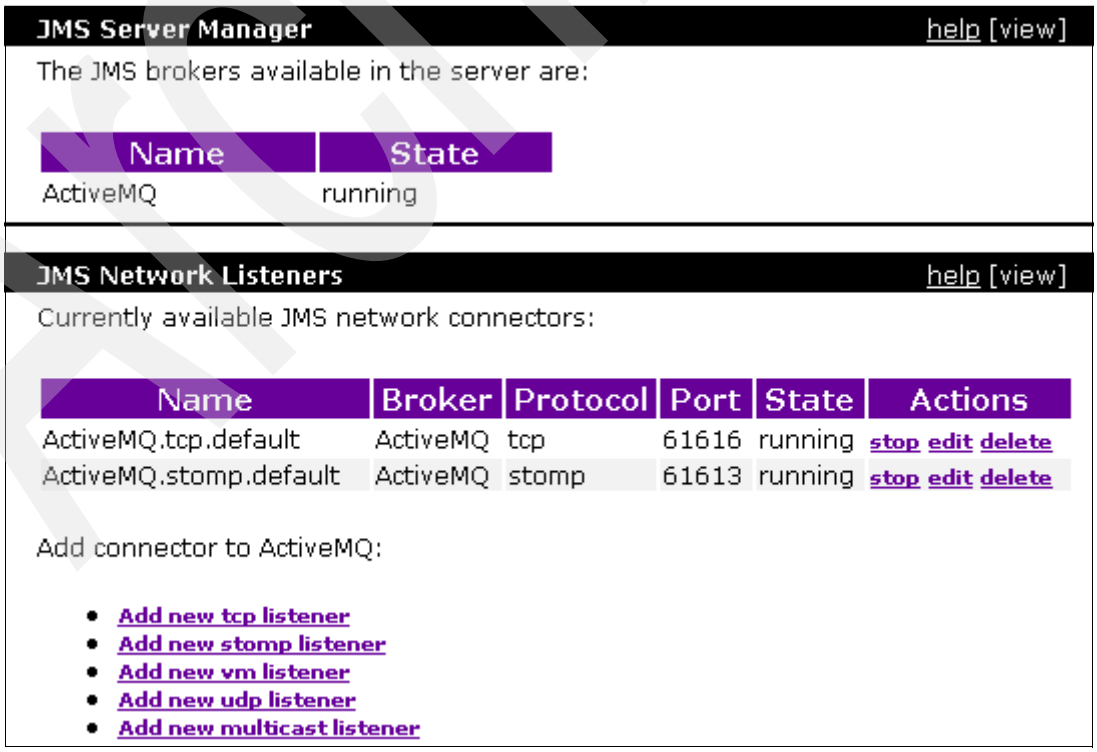


Figure 7-1 JMS brokers and network listeners

Select the **JMS Resources** portlet to view and create JMS resource groups. Figure 7-2 shows the JMS Resources page.

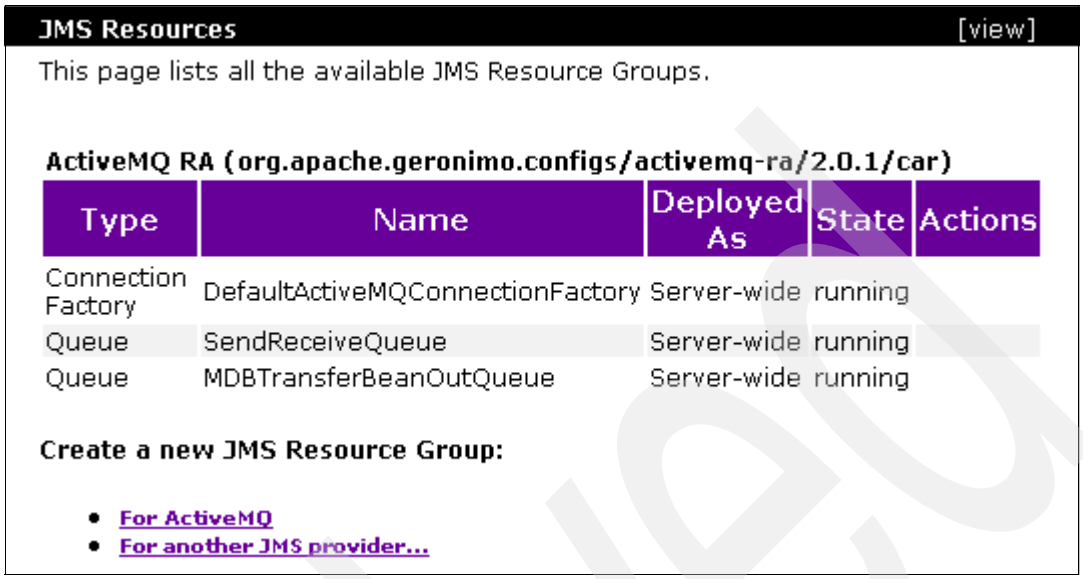


Figure 7-2 JMS Resources

7.2 ActiveMQ JMS provider

Apache ActiveMQ is an open source message broker that supports the JMS 1.1 specification. ActiveMQ supports both point-to-point and publish and subscribe messaging models. You can also use ActiveMQ from .NET, C, C++, Perl, Python, PHP, and Ruby using specific connectors. Community Edition includes ActiveMQ as the default JMS provider.

For more information about ActiveMQ, see the ActiveMQ project site at:

`http://activemq.apache.org/`

7.2.1 Default JMS resource group

By default, Community Edition comes with a pre-configured resource group called *ActiveMQ RA* for the ActiveMQ message broker, which allows you to develop applications without having to create a new resource group. The default configuration (Figure 7-2) contains the following connection factory and queues:

- DefaultActiveMQConnectionFactory
- SendReceiveQueue
- MDBTransferBeanOutQueue

The resource group is deployed with server scope, making it available to all applications that are deployed in Community Edition.

7.2.2 Adding a JMS resource group using the console

If an application requires queues or topics other than the default queues SendReceiveQueue and MDBTransferBeanOutQueue, you must add a new JMS resource group.

To create a queue called TestQueue and a queue connection factory called TestQCF for ActiveMQ:

1. To create a new JMS resource group for ActiveMQ using the administrative console, select **Services** → **JMS Resources**.
2. Click **For ActiveMQ**, which brings up a panel where you can enter the information that is required to connect to an ActiveMQ server. Figure 7-3 shows a partial view of the panel.

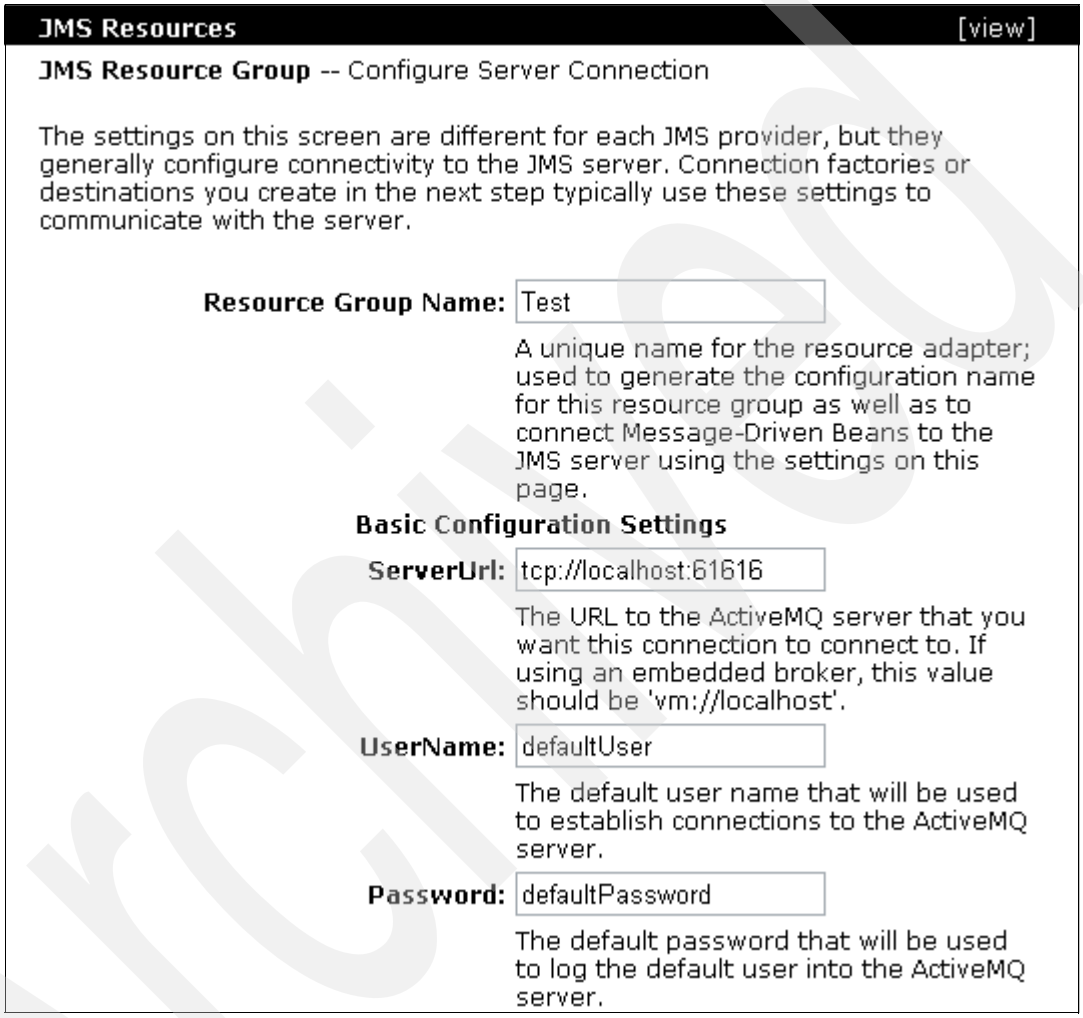


Figure 7-3 JMS resource group: configure the connection to ActiveMQ

On the JMS Resource Group-Configure Server Connection panel, perform the following actions, as shown in Figure 7-3:

a. Enter `Test` as the JMS resource group name.
b. 61616 is the default port the ActiveMQ server listens on. Change the port in the ServerUrl field, if you changed the ActiveMQ port.
c. You can leave the other fields unchanged, taking the default.
d. Click **Next**.

3. On the next panel (Figure 7-4), you can add connection factories and destinations (queues and topics) to the resource group.

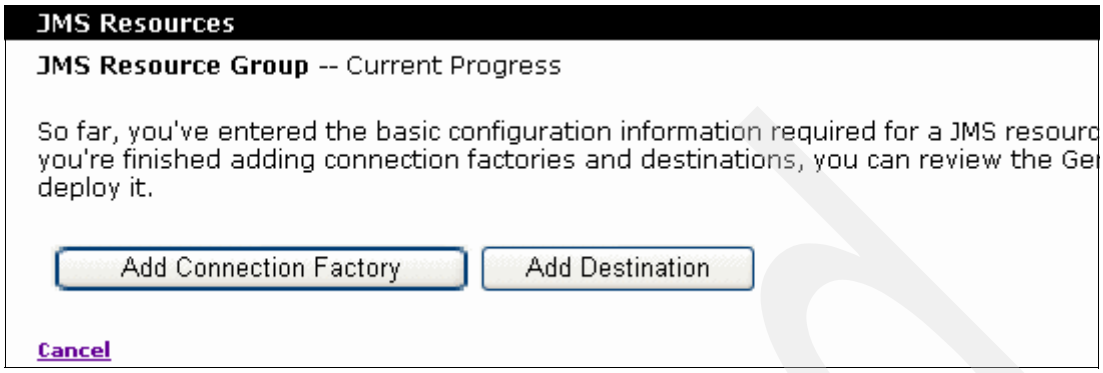


Figure 7-4 JMS resource group: Add connection factory or destination

4. Click **Add Connection Factory** to create the TestQCF connection factory (Figure 7-5).

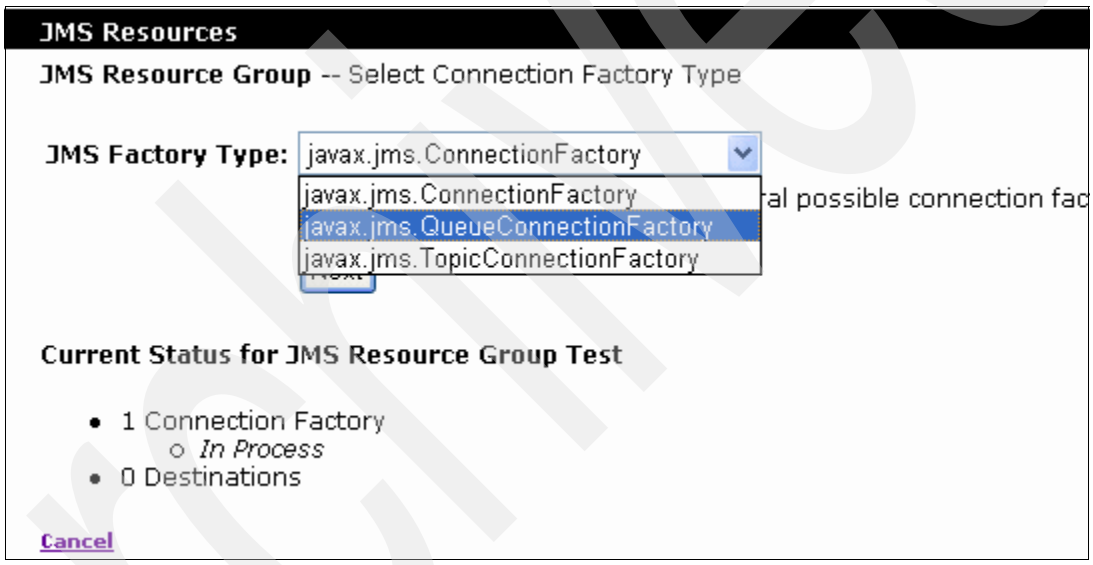


Figure 7-5 JMS resource group: Select the factory type

a. For a queue connection factory (as in this example), select **javax.jms.QueueConnectionFactory** from the list.

b. For a topic connection factory, select **javax.jms.TopicConnectonFactory**.

c. Click **Next**.

5. Enter the details for the connection factory, as shown in Figure 7-6.

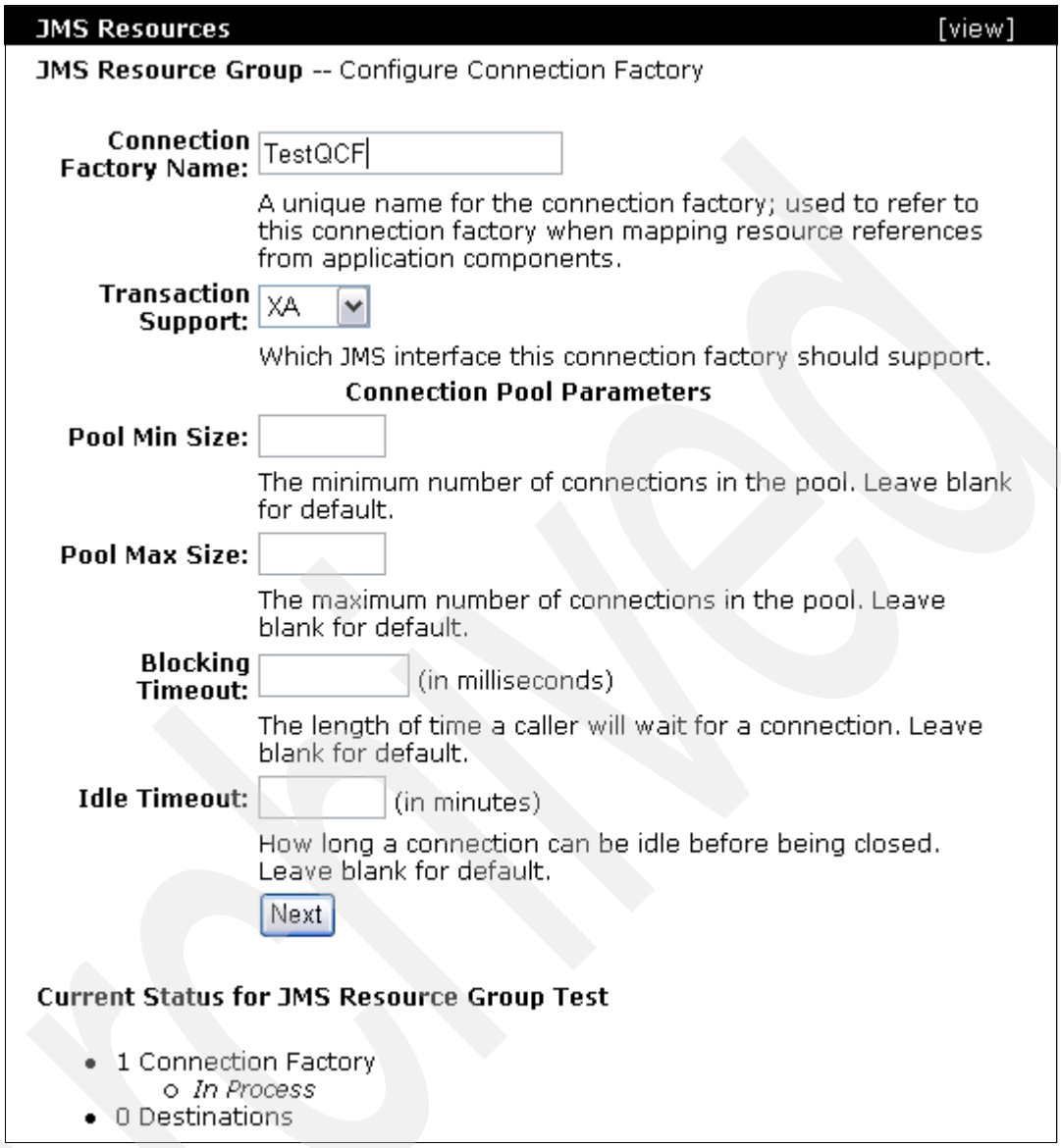


Figure 7-6 JMS resource group: configure a connection factory

   a. Enter the connection factory name, which is TestQCF for the example.

   b. If you do not want to use ActiveMQ within XA transactions, select the Transaction Support as Local or None. Otherwise leave it unchanged as XA.

   c. You can leave the other fields blank, and click **Next**.

6. The next panel (Figure 7-7 on page 81) shows the newly created connection factory. You can add more connection factories or destinations.

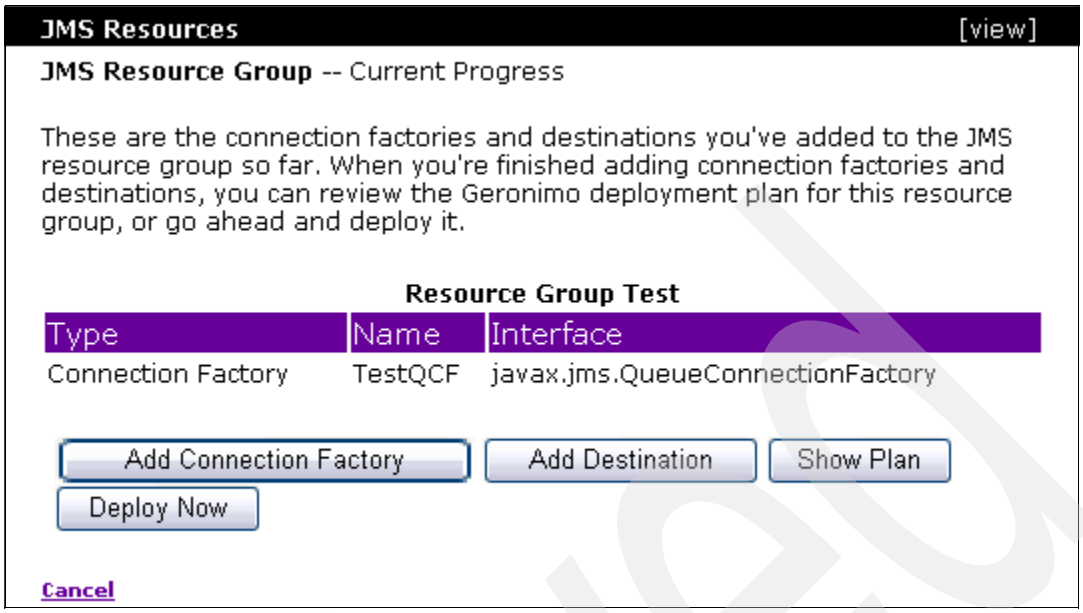


Figure 7-7 JMS resource group: Add connection factory results

7. Click **Add Destination** (Figure 7-7) to create the queue TestQueue.
8. To add a queue, as in this example, select **javax.jms.Queue** as the destination type, as shown in Figure 7-8. To add a new topic select java.jms.Topic. Click **Next**.

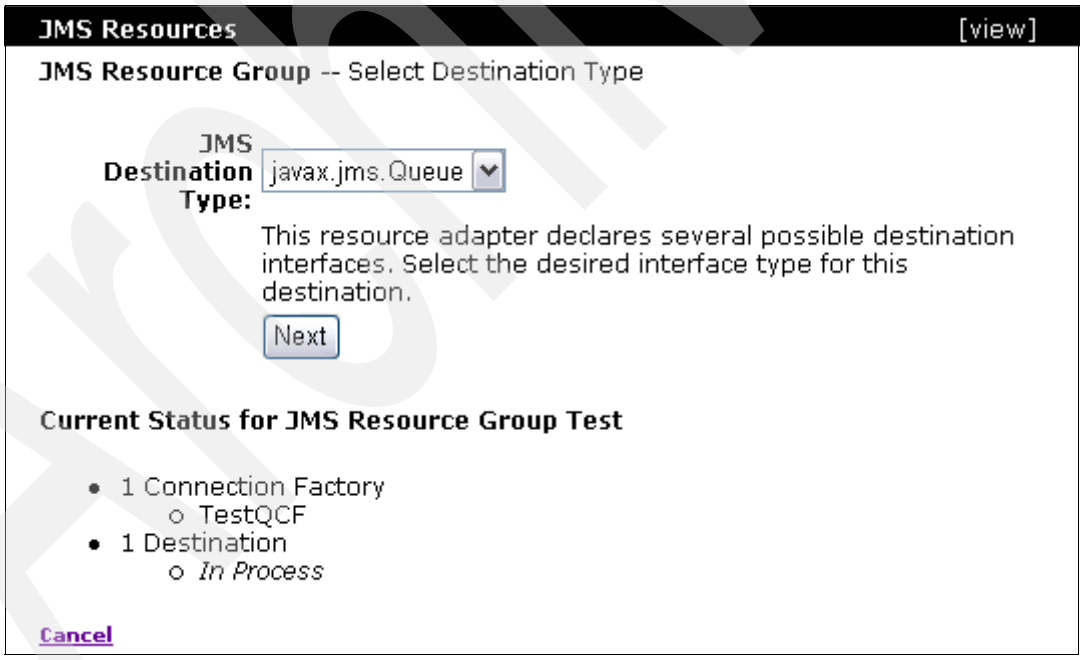


Figure 7-8 JMS resource group: Add destination

9. On the next panel (Figure 7-9), you can configure a destination name and the physical name to be given for the queue. Enter the destination and physical name. You use the destination name to refer to the actual physical queue from your deployment plans. Click **Next**.

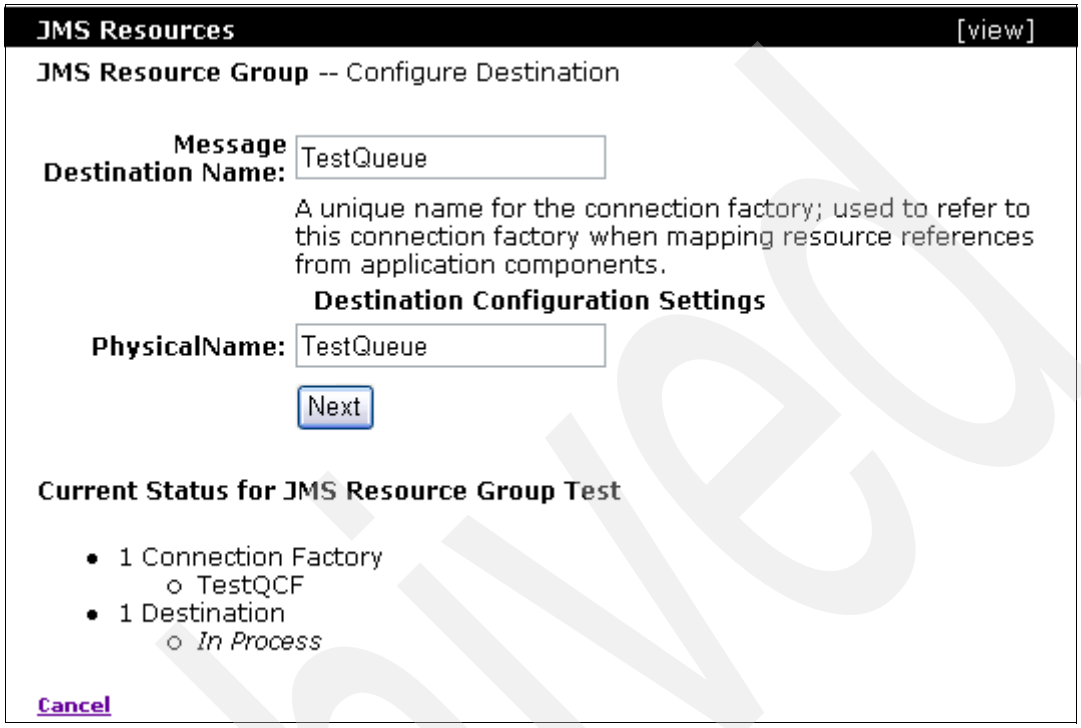


Figure 7-9 JMS resource group: configure the destination

The panel shown in Figure 7-10 lists all of the resources in the JMS resource group. You can add new destinations and connection factories from this panel.

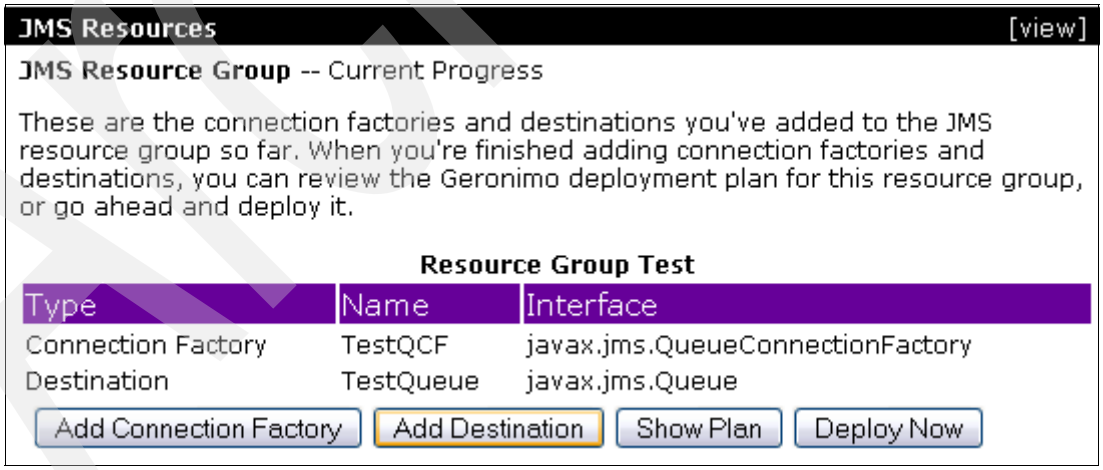


Figure 7-10 JMS resource group: add destination results

10. After you add all of the JMS resources, click **Deploy Now**. This deploys the JMS resource group and displays the JMS Resources panel (Figure 7-11 on page 83).

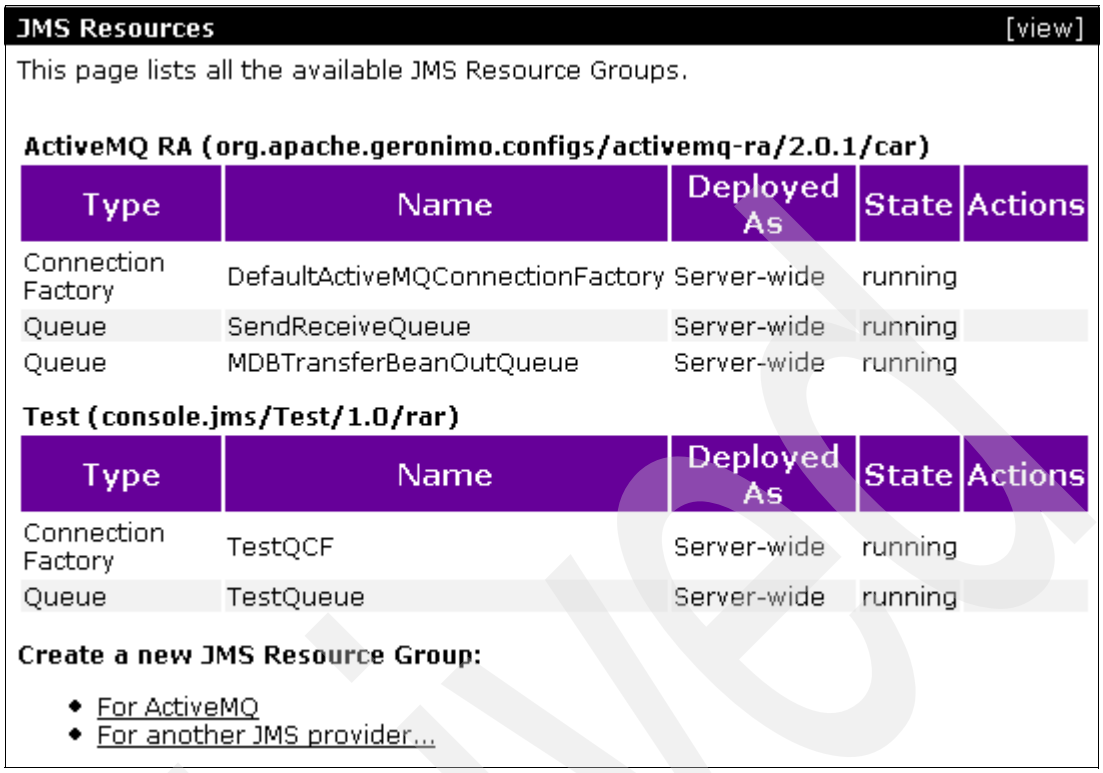


Figure 7-11 JMS resource group: deploy results

The JMS resource group Test is now deployed server wide and can be accessed from applications.

7.2.3 Adding a new JMS resource group using the deploy command

Because a resource adapter is a J2EE artifact, you can include it as part of your EAR file. When you use the console to create and deploy a JMS resource group, it is available for all deployed applications. When you deploy the resource adapter as part of an EAR, it is available only for that module.

You can also choose to deploy the resource adapter server wide using the **`deploy`** command. For more information, see:

`http://cwiki.apache.org/GMOxDOC20/jms-resources-deployment.html`

7.2.4 Removing a JMS resource group

If you no longer require a JMS resource group, you can remove it. To do this, select the **J2EE Connectors** portlet in the administrative console, and uninstall the resource adapter.

You cannot remove the default JMS resource group. The uninstall option for default JMS resource group is disabled.

Figure 7-12 on page 84 shows the connector list.

Component Name	State	Commands			Parent Components
console.jms/Test/1.0/rar	running	Stop	Restart	Uninstall	org.apache.geronimo.config broker/2.0.1/car org.apache.geronimo.config server/2.0.1/car org.apache.geronimo.config
org.apache.geronimo.configs/activemq-ra/2.0.1/car	running	Stop	Restart	Uninstall	org.apache.geronimo.config broker/2.0.1/car org.apache.geronimo.config server/2.0.1/car org.apache.geronimo.config
org.apache.geronimo.configs/system-database/2.0.1/car	running	Stop	Restart	Uninstall	org.apache.geronimo.config server/2.0.1/car org.apache.geronimo.config

Figure 7-12 Connector list

7.2.5 ActiveMQ connectors

Connectors allow ActiveMQ to send and receive messages using different protocols, for example, the tcp connector allows ActiveMQ to send and receive messages using the tcp protocol. By default, ActiveMQ in Community Edition comes with pre-configured tcp and Stomp connectors. The corresponding connector must be running to use a specific protocol. You can view and add connectors from the administrative console.

When creating a JMS resource group, you have to specify a serverUrl. The protocol for communication between ActiveMQ broker and Community Edition is decided based on the protocol that is specified in the serverUrl, for example, if the serverUrl is specified as `tcp://localhost:61616`, the tcp protocol is used for communication with ActiveMQ broker.

You can manage the connectors by selecting the **JMS Server** portlet in the console. Figure 7-13 on page 85 shows the ActiveMQ connectors in the console.

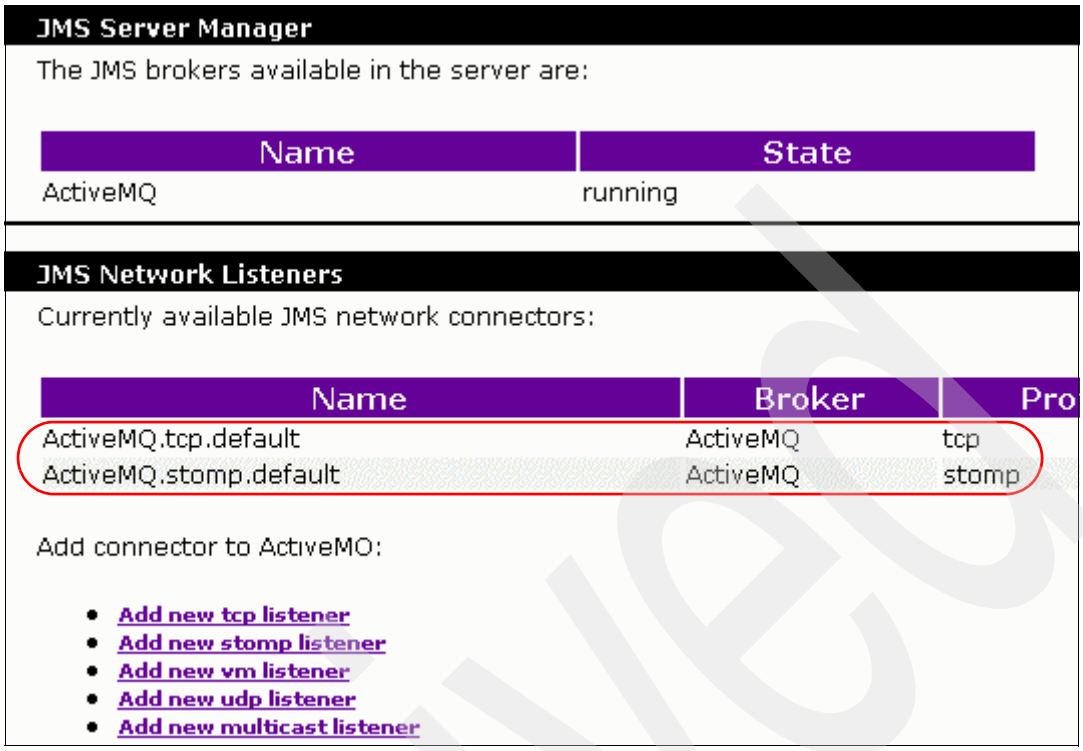


Figure 7-13 Viewing the ActiveMQ connectors in the console

Here are the ActiveMQ connectors:

- Tcp connector

  The tcp connector is used for communication with the ActiveMQ broker using the tcp protocol. To use the tcp connector, specify tcp in the serverUrl value when you configure the ActiveMQ server connection in the JMS resource group (see Figure 7-3 on page 78), for example: `tcp://localhost:61616`

  By default this connector is started.

- Stomp connector

  Ruby, Perl, Python, or PHP clients use the stomp connector for easier interaction with ActiveMQ. By default this connector is started. Refer to the following Web page for a detailed description of the stomp connector:

  `http://activemq.apache.org/stomp.html`

- VM Connector

  If ActiveMQ broker and Community Edition are running in the same JVM, you can use the VM protocol for communication with ActiveMQ. Using the VM protocol has a performance advantage because the message send and receive occurs using direct Java method calls. The tcp protocol carries the overhead of creating and maintaining tcp connections. The following instructions tell you how to use VM protocol for JMS messaging:

– Adding the VM Connector to ActiveMQ

  For adding a new connector, take the JMS Server tab, and click **Add new vm listener**, which opens a new portlet that will ask you to enter the name, host, and port. Specify the host as localhost and the port as any positive number. This creates a new vm connector.

– Configuring JMS resource group to use VM Connector

  After ActiveMQ is started with the VM connector, you can configure the JMS resource group to use the VM protocol. Specify the serverUrl value as:

  ```
  vm://localhost
  ```

7.3 WebSphere MQ JMS provider

To integrate WebSphere MQ with Community Edition you need to first get the WebSphere MQ resource adapter. The resource adapter is provided in WebSphere MQ 6.0.2.1 or later fix packs. You can also get the resource adapter with WebSphere MQ v6.0 client.

In the following sections, we describe how to configure a JMS resource group for use with WebSphere MQ. The sample creates a JMS resource group called MQRA that contains a queue connection factory called MyQueueConnectionFactory and a queue called TestQueue. The JMS resource group communicates with an existing queue manager (WASCE_Q) and queue (TestQueue) in WebSphere MQ. After the JMS resource group is configured, any JMS applications or MDBs running in the server can access the resources of queue manager WASCE_Q.

Refer to the WebSphere MQ V6.0 Information Center for more information about where to find the resource adapter file, the properties supported, and troubleshooting information:

http://publib.boulder.ibm.com/infocenter/wmqv6/v6r0/index.jsp?topic=/com.ibm.mq.csqzaw.doc/uj40010_.htm

7.3.1 Locate the MQ resource adapter

If you installed WebSphere MQ 6.0.2.1 or later, you can find the resource adapter wmq.jmsra.rar in the directory in Table 7-1.

Table 7-1 Directory for the wmq.jmsra.rar resource adapter

Platform	Directory
AIX	/usr/mqm/java/lib/jca
HP-UX, Linux and Solaris	/opt/mqm/java/lib/jca
i5/OS®	/QIBM/ProdData/mqm/java/lib/jca
Windows	*install_dir*\java\lib\jca

The WebSphere MQ V6 and V5.3 client is a free download from:

http://www-1.ibm.com/support/docview.wss?rs=171&uid=swg24009961&loc=en_US&cs=utf-8&lang=en

The client contains the WebSphere MQ resource adapter, which can connect to any V6 or V5.3 queue manager in client mode (for example, over TCP/IP). XA transactions are not

available by default in this configuration, so you must purchase the WebSphere MQ extended transactional client to use XA to a V5.3 queue manager.

7.3.2 Install the dependent jar files

To use WebSphere MQ in an XA transaction you need the WebSphere MQ Extended Transactional Client JAR file, com.ibm.mqetclient.jar**.** This JAR file provides extended transactional client support. In Windows, the file is located in the *MQ_xclient_install_dir*/java/lib folder.

The typical installation of WebSphere MQ 6.0 does not contain this JAR file. You have to select the extended transactional client support from the custom installation option. If your installation does not have this JAR file, you can purchase the extended transaction client from IBM.

After you have the com.ibm.mqetclient.jar file, you must define it to Community Edition as a common library. Later, you will have to give a dependency to this JAR file from the resource adapter deployment plan.

To install the JAR file:

1. Select the **Common Libs** portlet in the administrative console, and select the JAR file, as shown in Figure 7-14.
2. Provide the appropriate values for Group ID, Artifact, Version, and Type. Click **Install**.

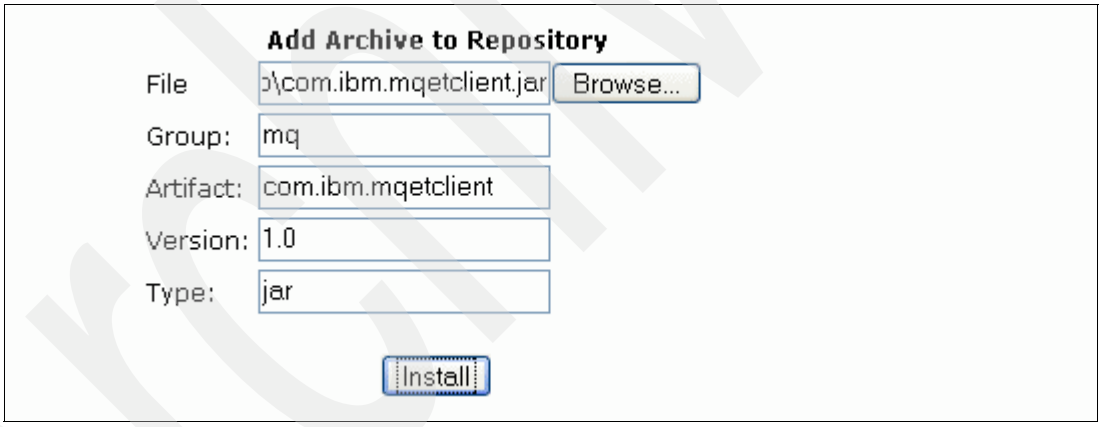


Figure 7-14 Define a common library

After you install the jar file, you can see it in the list of installed JAR files at the bottom of the same panel, as shown in Figure 7-15.

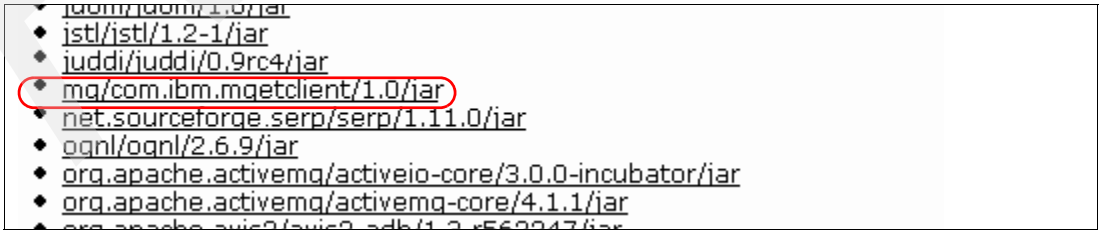


Figure 7-15 Installed resource adapter

7.3.3 Creating a Community Edition specific resource adapter deployment plan

To deploy the resource adapter you need a resource adapter deployment plan. The plan contains information that includes:

- Connection to the WebSphere MQ broker
- Queue manager name
- Mapping of queues and topics managed by the queue manager and Community Edition
- Connection pool size
- XA transaction support
- Logging details

Example 7-1 is a sample deployment plan that you can use as a reference.

Example 7-1 Resource adapter deployment plan

```
<?xml version="1.0" encoding="UTF-8"?>

<!-- This is Geronimo-specific descriptor -->

<connector xmlns="http://geronimo.apache.org/xml/ns/j2ee/connector-1.2">
   <dep:environment xmlns:dep="http://geronimo.apache.org/xml/ns/deployment-1.2">
      <moduleId>
         <groupId>mq</groupId>
         <artifactId>jmsra</artifactId>
         <version>1.0</version>
           <type>rar</type>
           </moduleId>
       <dep:dependencies>
      <dep:dependency>
         <dep:groupId>mq</dep:groupId>
         <dep:artifactId>com.ibm.mqetclient</dep:artifactId>
         <dep:version>1.0</dep:version>
         <dep:type>jar</dep:type>
       </dep:dependency>
        </dep:dependencies>
    </dep:environment>

    <resourceadapter>
      <resourceadapter-instance>
         <resourceadapter-name>MQRA</resourceadapter-name>
         <config-property-setting
name="traceEnabled">true</config-property-setting>
             <config-property-setting
name="traceLevel">10</config-property-setting>
         <config-property-setting
name="traceDestination">System.out</config-property-setting>
         <workmanager>
            <gbean-link>DefaultWorkManager</gbean-link>
         </workmanager>
      </resourceadapter-instance>

<outbound-resourceadapter>
          <connection-definition>
         <connectionfactory-interface>javax.jms.QueueConnectionFactory
```

```
            </connectionfactory-interface>
         <connectiondefinition-instance>
            <name>MyQueueConnectionFactory</name>
            <config-property-setting
name="queueManager">WASCE_Q</config-property-setting>
            <config-property-setting
name="channel">SYSTEM.DEF.SVRCONN</config-property-setting>
            <config-property-setting
name="transportType">CLIENT</config-property-setting>
                     <config-property-setting
name="hostName">localhost</config-property-setting>
            <config-property-setting name="port">1414</config-property-setting>
            <connectionmanager>
               <xa-transaction/>
                  <single-pool>
                     <max-size>50</max-size>
                     <min-size>20</min-size>

<blocking-timeout-milliseconds>5000</blocking-timeout-milliseconds>
                              <idle-timeout-minutes>2</idle-timeout-minutes>
                        <match-all />
                        </single-pool>
               </connectionmanager>
         </connectiondefinition-instance>
           </connection-definition>
      </outbound-resourceadapter>
   </resourceadapter>

   <adminobject>
      <adminobject-interface>javax.jms.Queue</adminobject-interface>

<adminobject-class>com.ibm.mq.connector.outbound.MQQueueProxy</adminobject-class>
      <adminobject-instance>
         <message-destination-name>TestQueue</message-destination-name>
         <config-property-setting
name="baseQueueName">TestQueue</config-property-setting>
         <config-property-setting
name="baseQueueManagerName">WASCE_Q</config-property-setting>
      </adminobject-instance>
   </adminobject>
</connector>
```

You can connect to other queue managers by adding more <connection-definition> instances. You can also add more queues or topics by adding new <adminobject> instances. The baseQueueName in <adminobject> specifies the queue name used in MQ, and message-destination-name specifies the name in Community Edition.

In this case, the `traceEnabled` property for the MQ resource adapter was set to true, and the traceLevel is set to 10. This logs all of the messages from the resource adapter. See the following Web site for more information:

http://publib.boulder.ibm.com/infocenter/wmqv6/v6r0/index.jsp?topic=/com.ibm.mq.csqzaw.doc/uj40010_.htm

Save this plan as wasce_mq.xml (You are free to choose any name).

7.3.4 Deploy the resource adapter

After you create the deployment plan and install the WebSphere MQ Extended Transactional Client jar file, you can deploy resource adapter:

1. In the console, select **Deploy New**.
2. Select the **wmq.jmsra.rar** RAR file as the archive and **wasce_mq.xml** as the plan.

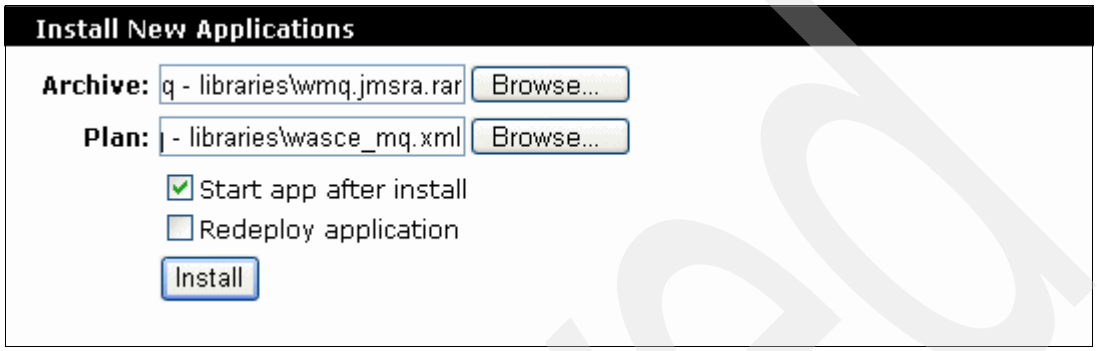


Figure 7-16 Deploy the WebSphere MQ resource adapter

After it is deployed, you can view the new JMS resource group for WebSphere MQ by selecting **JMS Resources** in the console.

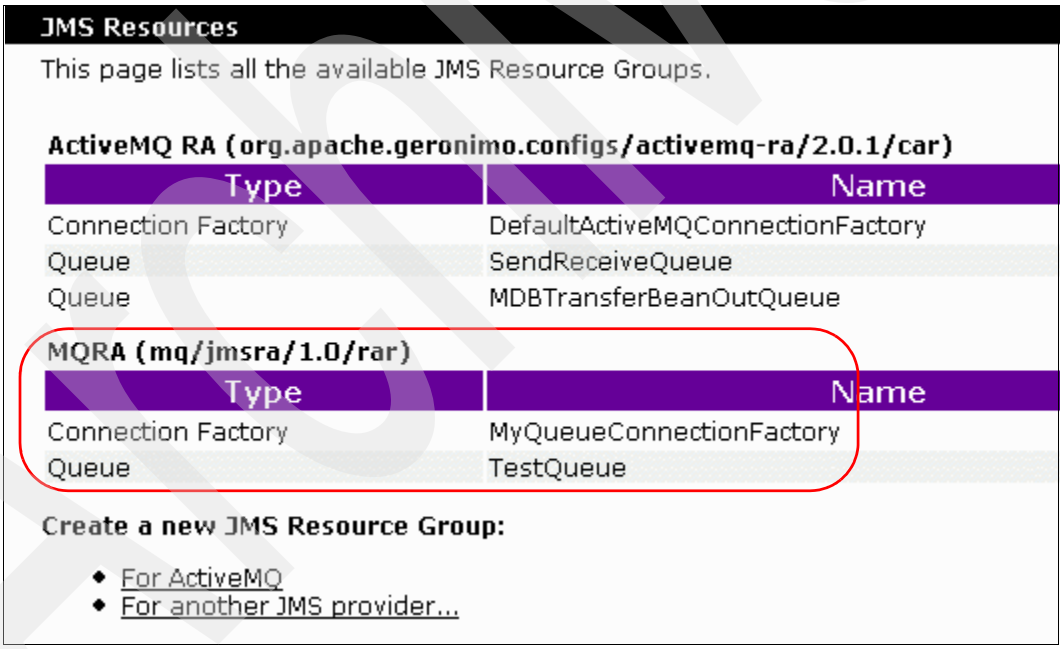


Figure 7-17 Viewing the new JMS resource group

7.3.5 Inbound resource adapters for MDBs

Inbound communication refers to receiving messages from WebSphere MQ. No inbound resource adapter is configured in wasce_mq.xml. You configure the inbound resource adapter when you configure MDBs using the activation-spec properties of the openejb-jar.xml deployment plan for the MDB. For an MDB to listen to TestQueue, you must specify the queue manager name, host, port, and channel. Example 7-2 on page 91 shows a sample openejb-jar.xml.

Example 7-2 openejb-jar.xml

```
<openejb-jar xmlns="http://www.openejb.org/xml/ns/openejb-jar-2.1"
             xmlns:nam="http://geronimo.apache.org/xml/ns/naming-1.2"
             xmlns:sec="http://geronimo.apache.org/xml/ns/security-1.2"
             xmlns:sys="http://geronimo.apache.org/xml/ns/deployment-1.2">

  <sys:environment>
    <sys:moduleId>
      <sys:groupId>com.ibm.wasce.samples</sys:groupId>
      <sys:artifactId>MDBDemo</sys:artifactId>
      <sys:version>2.0.0.1</sys:version>
      <sys:type>car</sys:type>
    </sys:moduleId>
    <sys:dependencies>
      <sys:dependency>
        <sys:groupId>mq</sys:groupId>
        <sys:artifactId>jmsra</sys:artifactId>
        <sys:version>1.0</sys:version>
        <sys:type>rar</sys:type>
      </sys:dependency>
    </sys:dependencies>
    <sys:hidden-classes/>
    <sys:non-overridable-classes/>
  </sys:environment>


  <enterprise-beans>
    <message-driven>
      <ejb-name>SampleMDB</ejb-name>
      <nam:resource-adapter>
        <nam:resource-link>MQRA</nam:resource-link>
      </nam:resource-adapter>
      <activation-config>
        <activation-config-property>

<activation-config-property-name>destination</activation-config-property-name>

<activation-config-property-value>TestQueue</activation-config-property-value>
        </activation-config-property>
        <activation-config-property>

<activation-config-property-name>destinationType</activation-config-property-name>

<activation-config-property-value>javax.jms.Queue</activation-config-property-valu
e>
        </activation-config-property>
<activation-config-property>

<activation-config-property-name>brokerQueueManager</activation-config-property-na
me>

<activation-config-property-value>WASCE_Q</activation-config-property-value>
        </activation-config-property>
        <activation-config-property>
```

```
<activation-config-property-name>hostName</activation-config-property-name>

<activation-config-property-value>localhost</activation-config-property-value>
        </activation-config-property>
    <activation-config-property>
          <activation-config-property-name>port</activation-config-property-name>
<activation-config-property-value>1414</activation-config-property-value>
        </activation-config-property>
   <activation-config-property>

<activation-config-property-name>channel</activation-config-property-name>

<activation-config-property-value>SYSTEM.DEF.SVRCONN</activation-config-property-v
alue>
        </activation-config-property>


      </activation-config>
    </message-driven>
  </enterprise-beans>

</openejb-jar>
```

7.4 Other JMS providers

Any third party JMS provider can be integrated with Community Edition by using the resource adapter for that message provider. See the following Web site for more details on integrating other JMS providers with Community Edition:

http://publib.boulder.ibm.com/wasce/V2.0.0/en/java-messaging-services.html

7.5 How to use JMS resources in an application

To use a JMS resource group you must add a dependency to it in your deployment plan, for example, if your Web application accesses a queue to send messages, you need to add a dependency in geronimo-web.xml to the queue's JMS resource group. Similarly, if you are writing an MDB, you need to add a dependency in openejb-jar.xml to the queue's JMS resource group.

If the application uses the default JMS resource group ACTIVEMQ RA, give the dependency in Example 7-3.

Example 7-3 Defining dependencies to the resource group in the application deployment plan

```
<sys:dependencies>
      <sys:dependency>
        <sys:groupId>org.apache.geronimo.configs</sys:groupId>
        <sys:artifactId>activemq-ra</sys:artifactId>
        <sys:type>car</sys:type>
      </sys:dependency>
</sys:dependencies>
```

7.5.1 Accessing queues or topics from an application

Developers specify resource references and resource environment references to the connection factories, queues, and topics used in their JSPs and servlets.

The web.xml deployment descriptor of a Web application that sends messages to the SendReceiveQueue defines the resource references. Example 7-4 shows web.xml with resource references.

Example 7-4 web.xml with resource references

```
<resource-ref>
        <description>jms broker</description>
        <res-ref-name>jms/broker</res-ref-name>
        <res-type>javax.jms.QueueConnectionFactory</res-type>
        <res-auth>Container</res-auth>
    </resource-ref>

    <resource-env-ref>
        <description>Predefined Queue</description>
        <resource-env-ref-name>jms/queue/SendReceiveQueue</resource-env-ref-name>
        <resource-env-ref-type>javax.jms.Queue</resource-env-ref-type>
    </resource-env-ref>
```

In Example 7-4, jms/broker and jms/queue/SendReceiveQueue are logical names used in the application.

Example 7-5 gives a code snippet for accessing the queue.

Example 7-5 Code snippet - accessing JMS resources

```
Context jndiContext = new InitialContext();
QueueConnectionFactory connectionFactory = (QueueConnectionFactory)
jndiContext.lookup("java:comp/env/jms/broker");
Queue destination =
(Queue)jndiContext.lookup("java:comp/env/jms/queue/SendReceiveQueue");
```

The deployer has to map the logical names used in programs to actual queues, topics, and connection factories defined on the server. To do this, first add a dependency to the queue's JMS resource group, and map the logical name to the actual physical name. Example 7-6 is a sample geronimo-web.xml accessing SendReceiveQueue.

Example 7-6 geronimo-web.xml - JMS resource mapping

```
<nam:resource-ref>
        <nam:ref-name>jms/broker</nam:ref-name>
        <nam:resource-link>DefaultActiveMQConnectionFactory</nam:resource-link>
    </nam:resource-ref>
    <nam:resource-env-ref>
        <nam:ref-name>jms/queue/SendReceiveQueue</nam:ref-name>

<nam:message-destination-link>SendReceiveQueue</nam:message-destination-link>
    </nam:resource-env-ref>
```

7.5.2 Message-driven beans

Whenever a message comes to the registered queue or topic the *onMessage* method of an MDB is triggered. For this to happen, the deployer has to define a dependency to the JMS resource group and specify the real name of the queue or topic in the openejb-jar.xml.

A sample <enterprise-beans> section of openejb-jar.xml for an MDB listening to SendReceiveQueue of default JMS resource group ACTIVEMQ RA is shown in Example 7-7.

Example 7-7 MDB openejb-jar.xml - dependency to a JMS resource group

```
<enterprise-beans>
    <message-driven>
      <ejb-name>SampleMDB</ejb-name>
      <nam:resource-adapter>
        <nam:resource-link>ActiveMQ RA</nam:resource-link>
      </nam:resource-adapter>
      <activation-config>
        <activation-config-property>

<activation-config-property-name>destination</activation-config-property-name>

<activation-config-property-value>SendReceiveQueue</activation-config-property-val
ue>
        </activation-config-property>
        <activation-config-property>

<activation-config-property-name>destinationType</activation-config-property-name>

<activation-config-property-value>javax.jms.Queue</activation-config-property-valu
e>
        </activation-config-property>
      </activation-config>
    </message-driven>
  </enterprise-beans>
```

7.5.3 Standalone application clients

JMS resources can be accessed from standalone Java clients using the message broker specific APIs, for example, ActiveMQ resources can be accessed by using ActiveMQConnectionFactory.

Example 7-8 gives a code snippet for accessing the resources.

Example 7-8 Code snippet - accessing the ActivMQConnectionFactory

```
ActiveMQConnectionFactory connectionFactory =
       new ActiveMQConnectionFactory("tcp://localhost:61616");
QueueConnection connection =
(QueueConnection)connectionFactory.createConnection();
Session session = connection.createQueueSession(false,Session.AUTO_ACKNOWLEDGE);
Queue queue = session.createQueue("SendReceiveQueue");
```

7.5.4 Sample application

Sample applications are in the following locations:

- An order processing example is at:

  http://cwiki.apache.org/GMOxDOC20/jms-and-mdb-sample-application.html
- A simple Chat MDB sample is included with the Community Edition samples package.

7.6 Summary of references

- Apache ActiveMQ

  http://activemq.apache.org/
- Apache Geronimo 2.0 Documentation: JMS Resources deployment

  http://cwiki.apache.org/GMOxDOC20/jms-resources-deployment.html
- Apache ActiveMQ Stomp support

  http://activemq.apache.org/stomp.html
- WebSphere MQ Information Center

  http://publib.boulder.ibm.com/infocenter/wmqv6/v6r0/index.jsp
  - The WebSphere MQ resource adapter

    http://publib.boulder.ibm.com/infocenter/wmqv6/v6r0/index.jsp?topic=/com.ibm.mq.csqzaw.doc/uj40010_.htm
- Apache Geronimo v2.0 documentation

  http://cwiki.apache.org/GMOxDOC20/documentation.html
  - JMS and MDB sample application

    http://cwiki.apache.org/GMOxDOC20/jms-and-mdb-sample-application.html
  - Java Messaging Services

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/java-messaging-services.html
- MQC6: WebSphere MQ V6.0 Clients

  http://www-1.ibm.com/support/docview.wss?rs=171&uid=swg24009961&loc=en_US&cs=utf-8&lang=en

8

Using databases

Community Edition servers use database pools to provide a mechanism for applications to access databases that are hosted on database managers. In this chapter, we provide you with an overview of database pools and related components and an example of how to configure a database pool.

We cover the following topics in this chapter:

8.1 Database pools

Database pools provide a mechanism for applications to access databases that are hosted on database managers without having to include the physical implementation details of the database in the application code. The database pools also relieve the application from managing things, such as connection pools and timeouts. There are two types of database pools that are available in Community Edition for applications:

- Server wide

  A server-wide database pool is available to all deployed applications. You can have more than one database pool on an application server that points to the same or different databases.

  You can create server-wide database pools using the administrative console or in a deployment plan. Use the administrative console to start and stop them.

  To use a resource reference to access this type of database pool in an application, declare a dependency on the corresponding database pool in the application deployment plan.

- Module scoped

  A module-scoped database pool is configured in an application deployment plan. It is deployed as part of the application and is started and stopped when the application that owns the database pool is started and stopped. You cannot create this type of database pool using the administrative console.

8.2 Resource adapters

Resource adapters serve as the point-of-contact between applications, application servers and Enterprise Information Systems (EIS), such as database managers, in the context of this chapter. The resource adapters manage JDBC connections to the databases and work with vendor-specific JDBC drivers to provide applications with access to the data that is stored in databases. Community Edition uses the resource adapters that are developed by open source project TranQL and comes with two types of resource adapters:

- Generic resource adapter

  This resource adapter can work with any JDBC database managers. But only the local transactions are supported by this resource adapter (for example, XA transaction support is not available).

- Vendor-specific resource adapters

  These resource adapters provide tighter integration with the corresponding (vendor) database manager. XA transaction support is available with these resource adapters.

See the following resource for more information about configuring vendor-specific resource adapters:

- Configuring a database connection pool

  http://publib.boulder.ibm.com/wasce/V2.0.0/en/database-pool.html

8.3 JDBC drivers

JDBC drivers provide vendor-specific implementation classes for resource adapters. Community Edition comes with the JDBC drivers for Apache Derby V10.2.2.0 and IBM DB2 V8.2 and 9.1. To use the JDBC drivers, you must download the drivers from the vendor Web site and add them to the Community Edition server's repository.

In the following section, we describe how to add new JDBC driver files in the repository, which we illustrate using the Microsoft SQL Server 2005 JDBC driver. You can use the same procedure to add JDBC drivers for other database managers.

8.3.1 Adding a JDBC driver to the repository

Before you begin the following instructions, store the downloaded JAR files of the JDBC driver in a folder that you can access from the administrative console.

To add a JDBC driver to the Community Edition server's repository:

1. Logon to the administrative console, and select the **Common Libs** portlet.
2. Enter the details of the library file, as shown in Figure 8-1.

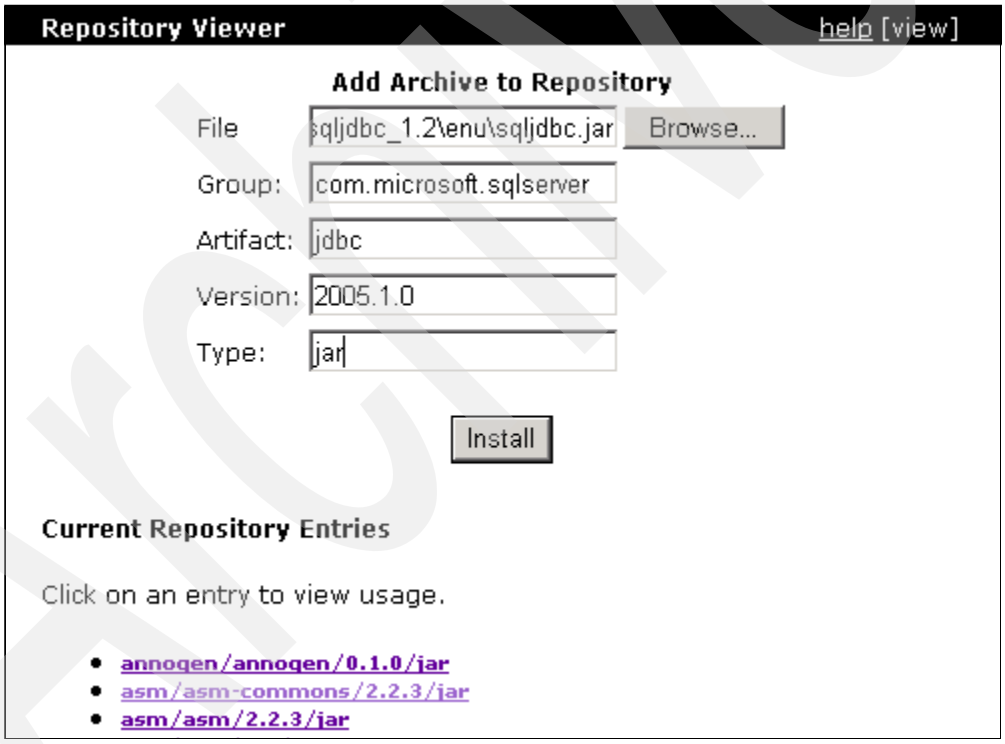


Figure 8-1 Add a JDBC driver archive to the repository

The information that you enter in the panel is used to deduce the name for the JDBC driver library that is being added. The Java libraries are stored in the repository using the following naming convention:

```
group/file/version/file-version.type
```

You are free to enter any values here. But it is a good idea to enter values relevant to the JDBC driver modules, for example:

– Group: The path to the JAR file in the package

– File: Use “jdbc” to indicate that this represents a JDBC driver module
– Version: A string that is based on version numbers used by the vendor
– Type: Use “jar” to indicate that this is a JAR file

Click **Install.**

If the driver has more than one JAR file, repeat this step for each JAR file.

3. As the JAR files are added, you see them in the Current Repository Entries list.

8.4 Sample configuration with a DB2 database

In this section, we describe how you can configure a server-wide database pool to work with an IBM DB2 database using the administrative console. Ensure that you have the following information before you begin the configuration activities:

- DNS or IP address of the DB2 server
- DB2 listener port number
- Database name
- User ID and password for the database

In the next section, we provide steps to configure a database pool for a DB2 database.

8.4.1 Creating a database pool

To create a database pool:

1. Logon to the administrative console, and select the **Database Pools** portlet.
2. Select **Using the Geronimo database pool wizard**, as shown in Figure 8-2.

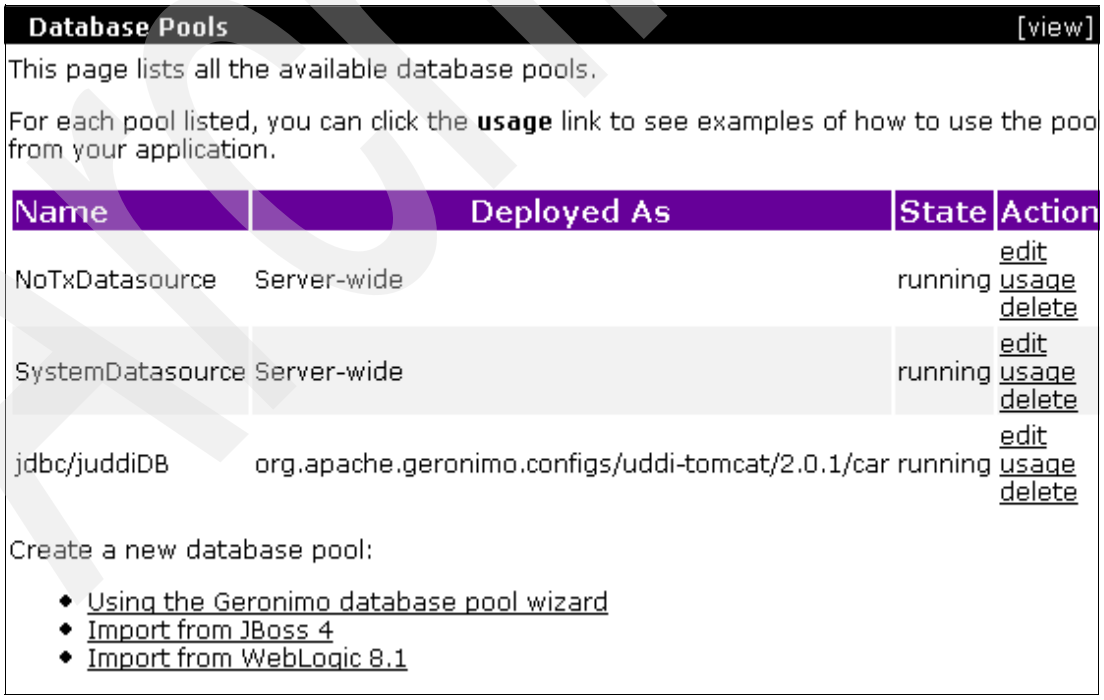


Figure 8-2 Database pool list

3. Enter a name for the database pool (for example, DB2-Pool-1), and select DB2 as the database type, as shown in Figure 8-3.

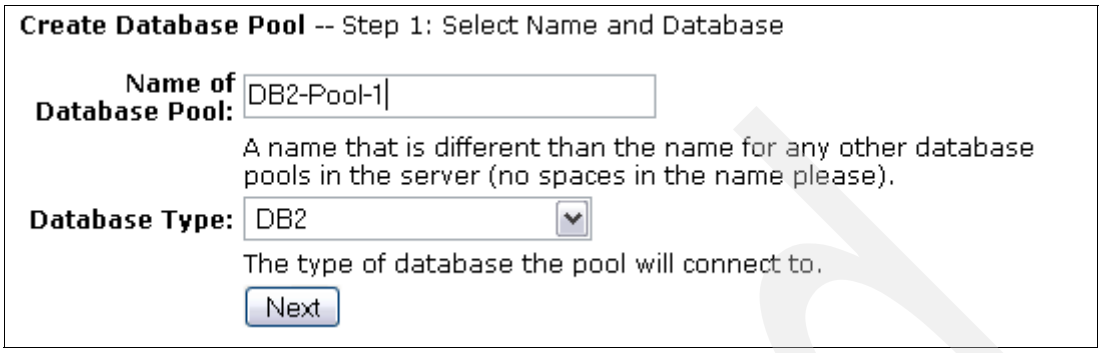


Figure 8-3 Create a database pool: select pool name and database type

Click **Next** to continue.

4. Complete the next page (Figure 8-4 on page 102) with the values that are required to define the driver and database connection:
   - In the JDBC Driver class field, enter `com.ibm.db2.jcc.DB2Driver`.
   - In the Driver JAR field, select **com.ibm.db2/db2jcc/<*version_no*>/jar** and **com.ibm.db2/db2jcc_cense_cu/<*version_no*>/jar**.

     Use CTRL-click to select more than one JAR file.
   - Enter the user ID and password required to connect to the database.
   - In the Port number field, enter the port number that is required to connect to the database manager.
   - In the Database field, enter the database name.
   - In the Host field, enter the DNS name or IP address of the DB2 server.

   Figure 8-4 on page 102 shows an administrative console panel for this DB2 V8.2 example.

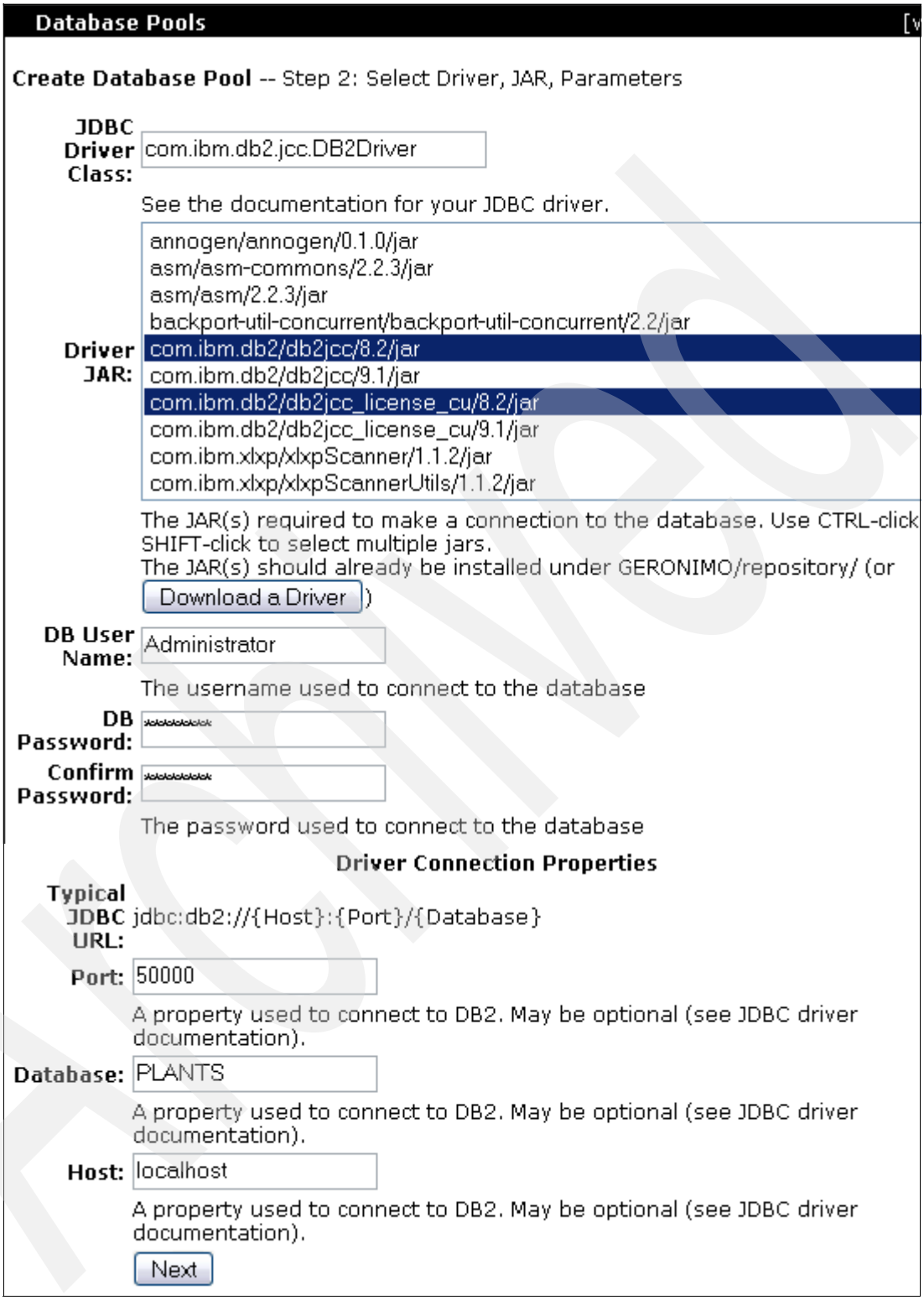


Figure 8-4 Create a database pool: select driver parameters

Click **Next** to continue.

5. Ensure that the JDBC Connect URL is populated with a value that is similar to:

   `jdbc:db2://<server name>:<port>/<database name>`

Here is a populated example:

```
jdbc:db2://db2.raleigh.itso.ibm.com:50000/PLANTS
```

If the database pool is correctly configured, you should see message **Loaded Successfully** in the Driver Status field.

6. You can leave the remaining fields blank to take the defaults.

   Figure 8-5 shows an administrative console panel for this configuration.

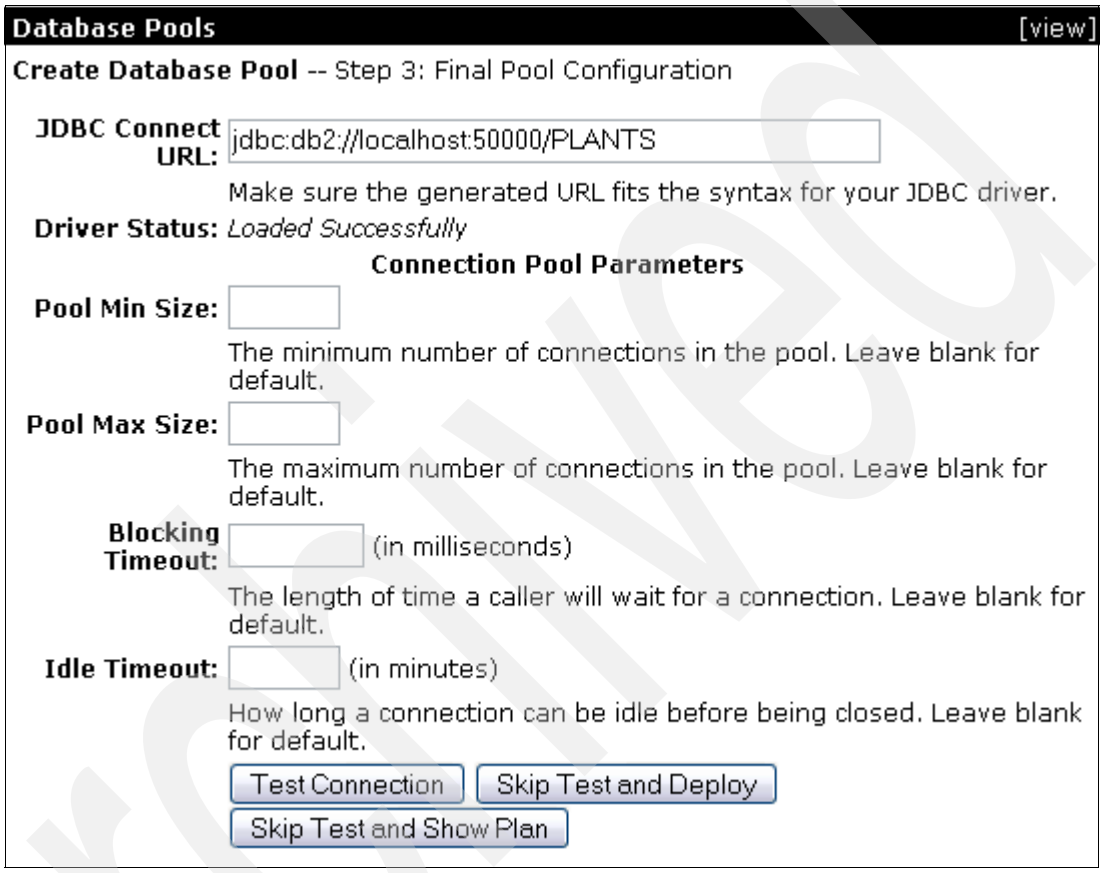


Figure 8-5 Create a database pool: final pool configuration

7. Click the **Test Connection** button (Figure 8-5) to test the connectivity with the DB2 database server. This step tries to connect to the database using the defined parameters:
   - If the connection works, a "Connected Successfully" message is displayed.
   - If the connection is not successful, a "Connection Error" and Test Error message is displayed. Review the error message to determine the connection problem, and use the **Edit Settings** button to modify the parameters (for example, DB User name or Password) to get the test connection to work.
8. Click **Deploy** to deploy a deployment plan, which contains a dependency element for the JDBC driver to be used in this database pool.

8.5 Sample configuration with Microsoft SQL Server 2005

In this section, we describe how you can configure a server-wide database pool to work with a Microsoft SQL Server 2005 database. Ensure that you have the following information before you begin the configuration activities:

- DNS or IP address of the SQL Server
- Database instance listener port number
- Database name
- User ID and password of the SQL login for the database

8.5.1 Creating a database pool

To create a database pool:

1. Logon to the administrative console, and from the left pane, select **Database Pools**.
2. Select **Using the Geronimo database pool wizard**.
3. Enter a name for the database pool (for example, MSSQL-Pool-1).
4. Select SQL Server as the database type.

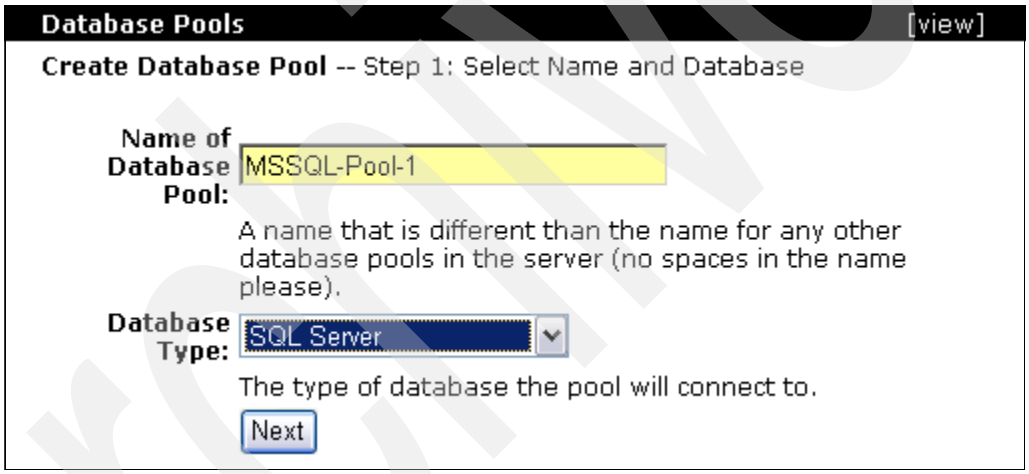


Figure 8-6 Create a database pool: select pool name and database type

5. Click **Next** to continue.
6. Complete the next page (Figure 8-7 on page 106) with the values that are required to define the driver and database connection:
   - In the JDBC Driver class field, enter **com.microsoft.sqlserver.jdbc.SQLServerDriver**.

     Note: The default JDBC driver class is valid for SQL servers prior to SQL Server 2005. If the target MS SQL server is 2005, you must change the JDBC Driver class.

   - Select **com.microsoft.sqlserver/jdbc/2005.1.0/jar** as the driver JAR.

     Note: This example uses the JDBC driver that we added in 8.3.1, “Adding a JDBC driver to the repository” on page 99, which is why we use the version number 2005.1.0 here.

– Enter the user ID and password that are required to connect to the database.
– In the Port number field, enter the port number that is required to connect to the database manager.
– In the Database field, enter the database name.
– In the Host field, enter the DNS name or IP address of the database server.

Figure 8-7 on page 106 below shows an administrative console dump for these configurations, which were done using a sample SQL Server Database.

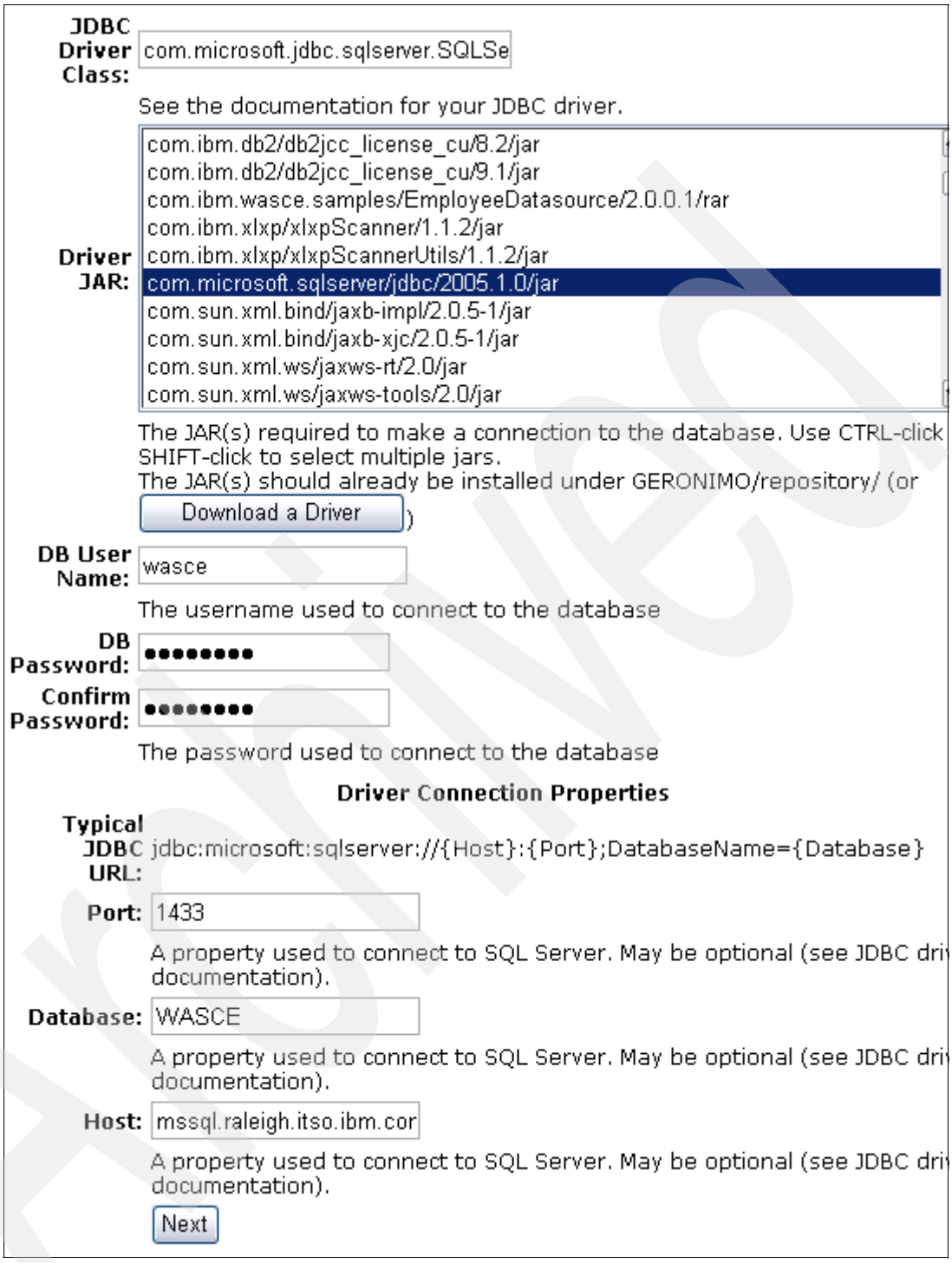


Figure 8-7 Create a database pool: select driver and parameters

Click **Next** to continue.

7. On the next page (Figure 8-8 on page 107), ensure that the JDBC Connect URL is set to:

   `jdbc:sqlserver://<server name>:<port>/DatabaseName=<database name>`

   Here is a populated example:

   `jdbc:sqlserver://mssql.raleigh.itso.ibm.com:1433;DatabaseName=WASCE`

Note: The default JDBC Connect URL is valid for SQL servers prior to SQL Server 2005. If the target MS SQL server is 2005, change the JDBC Connect URL as previously shown.

If the database pool is correctly configured, you should see message **Loaded Successfully** in the Driver Status field (Figure 8-8).

8. You can leave the remaining fields blanks to take the default values.

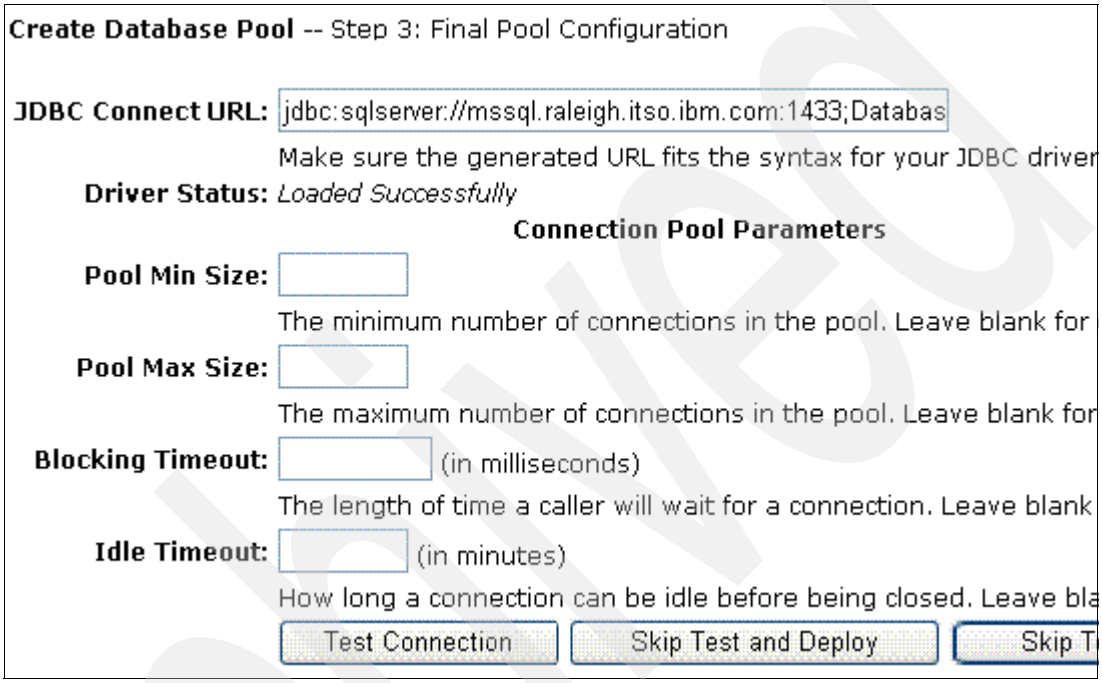


Figure 8-8 Create a database pool: final pool configuration

9. Click the **Test Connection** button to test the connectivity with the SQL Server 2005 database server. Figure 8-9 shows an administrative console panel with what you will see if the test connection works.

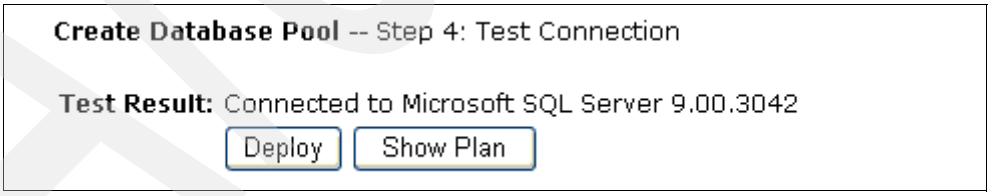


Figure 8-9 Create a database pool: test connection

10. Click **Deploy** to deploy a deployment plan that contains a dependency element for the JDBC driver that is used in this database pool. Figure 8-10 on page 108 shows an administrative console panel with what you will see if the deployment is successful. You will find that the new entry (for example MSSQL-Pool-1), which we just configured, is added as a server-wide database pool entry.

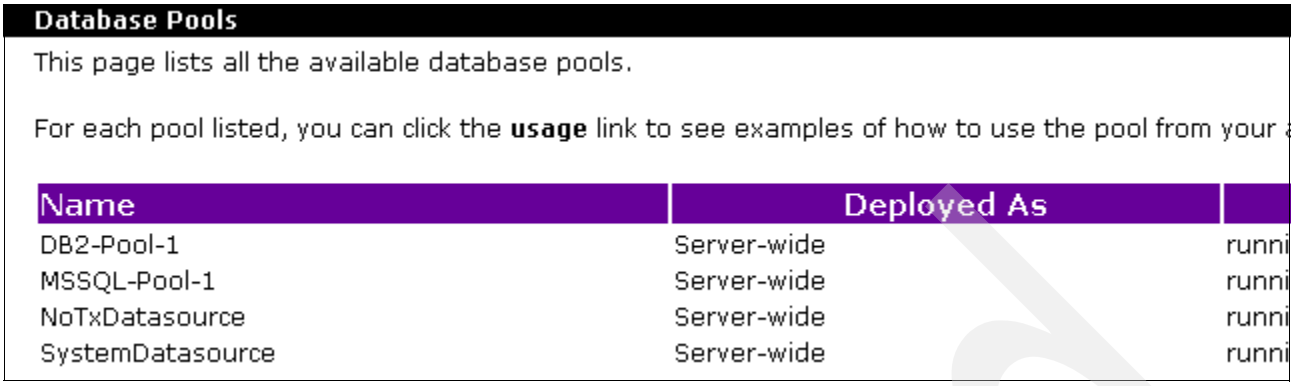


Figure 8-10 List of available database pools

8.6 Sample configuration with Derby (and deploy tool)

In the previous sections, we used the administrative console to configure database pools. In this section, we look at how you can configure a database pool using the Community Edition deploy tool. In this section, we also demonstrate how you can use the embedded database manager, Derby, to a test database application. Application EMPDemo, which is located in the Community Edition sample application package, is used for the demonstration purposes.

Apache Derby, which is based on the IBM Cloudscape® product, comes as a pre-configured database manager. Community Edition uses it to store system data. Derby runs inside the Community Edition JVM and is automatically started when Community Edition is started. You can use the administrative console to maintain user databases and to run SQL commands on them.

8.6.1 Creating a database

The following steps demonstrate how to create a database for the EMPDemo application using the administrative console:

1. Logon to the administrative console, and select the **DB Manager** portlet.
2. In the Create DB field, enter `Employee`, and click **Create**, as shown in Figure 8-11 on page 109.

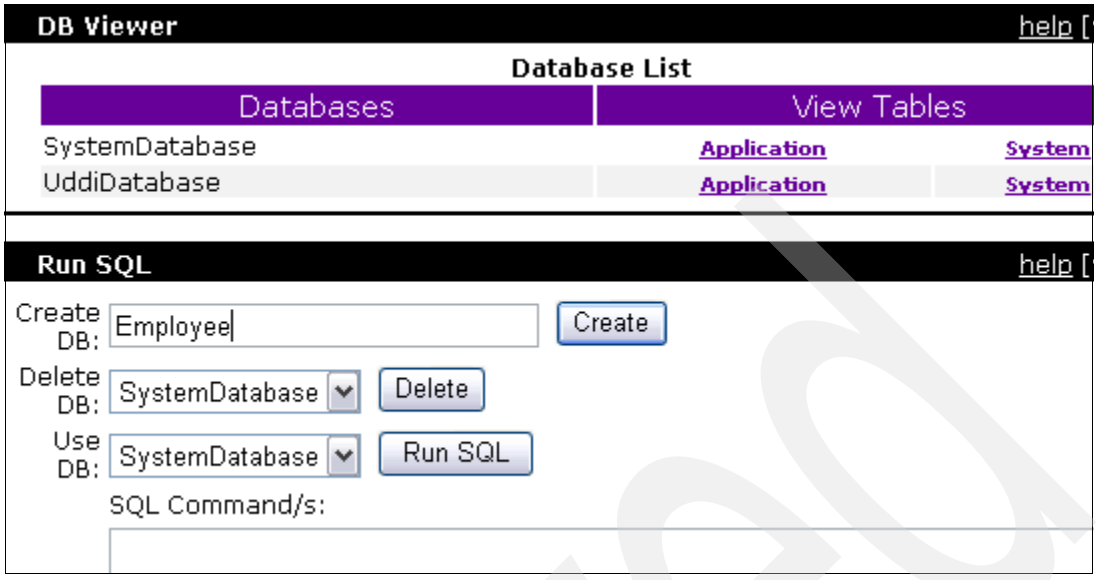


Figure 8-11 Create a Derby database

The new database is created and added to the Database List in the DB viewer, as shown in Figure 8-12.

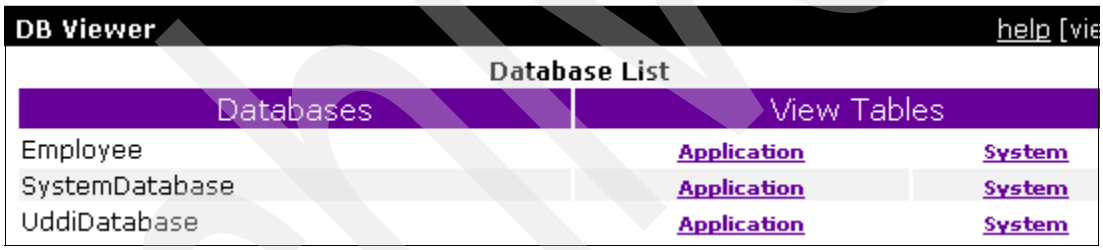


Figure 8-12 Employee database in the database list

3. Create a table called EMP in this database, and load some data into the table using SQL commands, as shown in Figure 8-13 on page 110. You can enter multiple SQL commands using a semicolon (;) as the delimiter:
   - Select **Employee** in the Use DB pull-down menu.
   - Enter the SQL commands to run, and click **Run SQL** to execute the command.
   - The SQL script is available in the EMPdemo sample application at *samples*\applications\EMPdemo\src\sql\emp_table_and_record_create.sql. Remove the DROP TABLE EMP; statement at the beginning of the script.

   See Figure 8-13 on page 110 for an example.

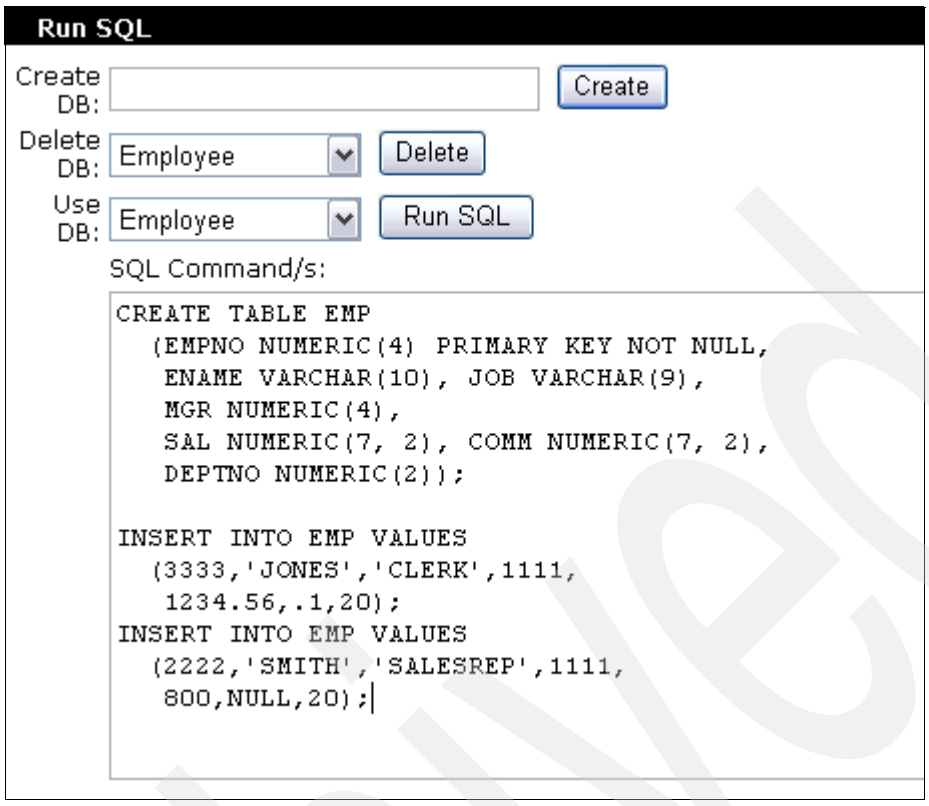


Figure 8-13 Run SQL to create tables and data in the Employee database

If the execution of the above command is successful, you should see the message **SQL command/s successful** under "Result:" found below the SQL Command/s box.

4. To check the content of the table that was just created, select the **Application** link, which is located next to the Employee database in the database list. Click the **View Contents** link, which is next to the EMP table on the tables list. The results look similar to Figure 8-14.

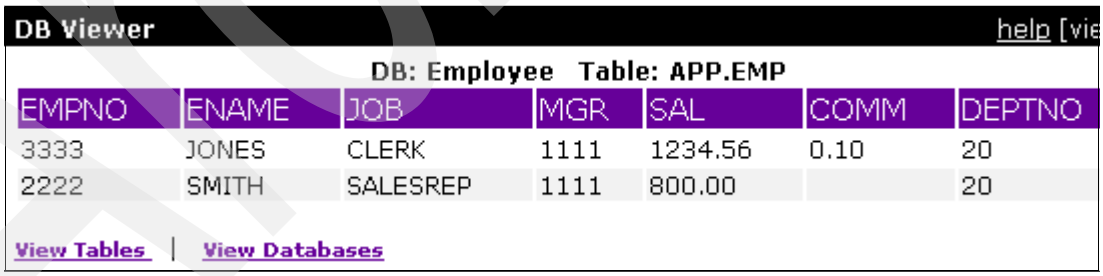


Figure 8-14 View database contents

8.6.2 Derby data source configuration

In this section, we describe how you can create a server-wide database pool using the deploy tool and a deployment plan. For this, you need a database plan that contains the information relating to the database. The EMPDemo sample application package has sample database plans for a number of database types. You can use the database plan for Derby, Cloudscape-db-plan.xml because a Derby database is already created to test the EMPDemo application.

The next section describes the components that you need to configure a data source for this exercise.

Database plan

Before you run the deploy tool, make sure that there is information in the database plan to create a data source for the database. The following section is a description of critical elements, which need to go in the database plan to define the Employee database.

moduleId

The <moduleId> element, in the database plan, is used to assign a unique name for this module, which is used to create a component name for the installed application or resource. It defines how the application is deployed in the repository.

Example 8-1 shows that the resource will be installed in directory *wasceHome*\repository\com\ibm\wasce\samples\EmployeeDatasource\2.0.0.1.

Example 8-1 Database plan <moduleID> element

```
<dep:moduleId>
            <dep:groupId>com.ibm.wasce.samples</dep:groupId>
            <dep:artifactId>EmployeeDatasource</dep:artifactId>
            <dep:version>2.0.0.1</dep:version>
            <dep:type>rar</dep:type>
</dep:moduleId>
```

dependency

Because the Employee database is created as a system database, this module has a dependency on the repository element that manages system databases. Therefore, the database plan must include a <dependency> element configured as shown in Example 8-2.

Example 8-2 Database plan <dependencies>

```
<dep:dependencies>
            <dep:dependency>
                <dep:groupId>org.apache.geronimo.configs</dep:groupId>
                <dep:artifactId>system-database</dep:artifactId>
                <dep:type>car</dep:type>
            </dep:dependency>
</dep:dependencies>
```

Resource adapter

The <resourceadapter> element of the database plan contains the information that describes the database pool.

The <connectiondefinition-instance> element configures a relationship between a JNDI name and physical implementation of the database, such as database name and connection pool sizes, and so on, as shown in Example 8-3.

Example 8-3 Database plan <connectiondefinition-instance>

```
<resourceadapter>
  <outbound-resourceadapter>
   <connection-definition>
   <connectionfactory-interface>javax.sql.DataSource</connectionfactory-interface>
       <connectiondefinition-instance>
         <name>jdbc/EmployeeDatasource</name>
```

```
        <config-property-setting name="UserName"/>
        <config-property-setting name="Password"/>
   <config-property-setting name="DatabaseName">Employee</config-property-setting>
   <config-property-setting name="CreateDatabase">true</config-property-setting>
        <connectionmanager>
           <xa-transaction>
             <transaction-caching/>
           </xa-transaction>
           <single-pool>
              <max-size>5</max-size>
              <min-size>0</min-size>
           <blocking-timeout-milliseconds>5000</blocking-timeout-milliseconds>
              <idle-timeout-minutes>15</idle-timeout-minutes>
              <match-one/>
            </single-pool>
          </connectionmanager>
        <!--global-jndi-name>EmployeeDatasource</global-jndi-name-->
        </connectiondefinition-instance>
      </connection-definition>
   </outbound-resourceadapter>
</resourceadapter>
```

Resource adapter

Community Edition uses TranQL resource adapters to manage JDBC resources. The XA version of the TranQL connector used for this example is:

wasceHome\repository\org\tranql\tranql-connector-derby-embed-xa\1.3\tranql-connector-derby-embed-xa-1.3.rar

Running the deploy tool

To deploy the data source, change the directory to the folder that contains **deploy.[bat | sh]**, and then you must run the following command in a command window:

```
deploy.[bat | sh] --user <name> ---password <password> deploy <path to database plan> <path to tranQL RAR>
```

Note: If the user ID and password is not present in the command, the deploy tool asks you to enter them on the command window.

Example 8-4 Deploy command and results

```
C:\WebSphere\CE\bin>deploy --user system --password manager deploy C:\Downloads\
WASCE_Samples\applications\EMPdemo\src\conf\Cloudscape-db-plan.xml C:\WebSphere\
CE\repository\org\tranql\tranql-connector-derby-embed-xa\1.3\tranql-connector-de
rby-embed-xa-1.3.rar

Using GERONIMO_BASE:   C:\WebSphere\CE
Using GERONIMO_HOME:   C:\WebSphere\CE
Using GERONIMO_TMPDIR: var\temp
Using JRE_HOME:        C:\Java50\jre

    Deployed com.ibm.wasce.samples/EmployeeDatasource/2.0.0.1/rar
```

As shown in Figure 8-15 on page 113, you will see the new database pool entry in the database pools list.

Name	Deployed As	State	Actions
DB2-Pool-1	Server-wide	running	edit usage delete
NoTxDatasource	Server-wide	running	edit usage delete
SystemDatasource	Server-wide	running	edit usage delete
jdbc/EmployeeDatasource	Server-wide	running	edit usage delete

Figure 8-15 New database pool definition

8.6.3 Deploying and testing the EMPDemo application

To deploy the EMPDemo application, you can either use the administrative console or the hot deployment facility.

To deploy the application using the administrative console:

1. Select the **Deploy New** portlet.
2. Browse to the location of the EMPdemo-2.0.0.1.war file in *wasce_samples*\applications\EMPdemo\target (Figure 8-16), and click **OK**.

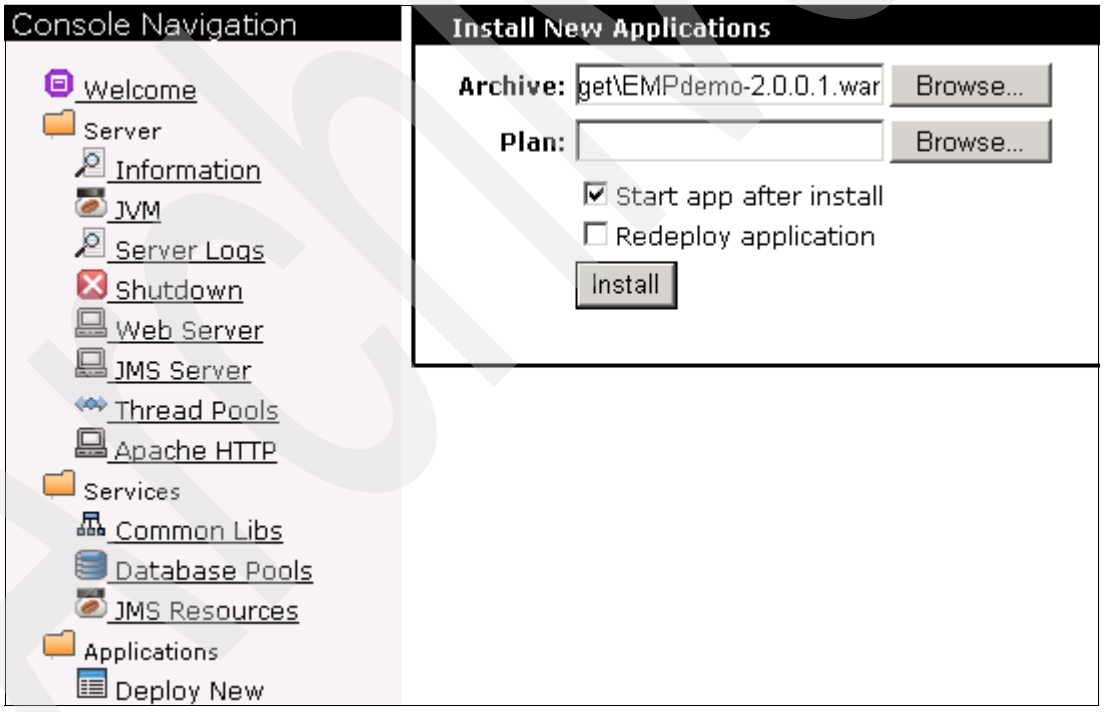


Figure 8-16 Install the sample application

3. Click **Install**.
4. Use the following URL to test the application.

   `http://localhost:8080/EMPdemo/jsp/EMPdemo.jsp`

   If the previous configurations were correctly done and the application is correctly deployed, you should see a response similar to Figure 8-17 on page 114.

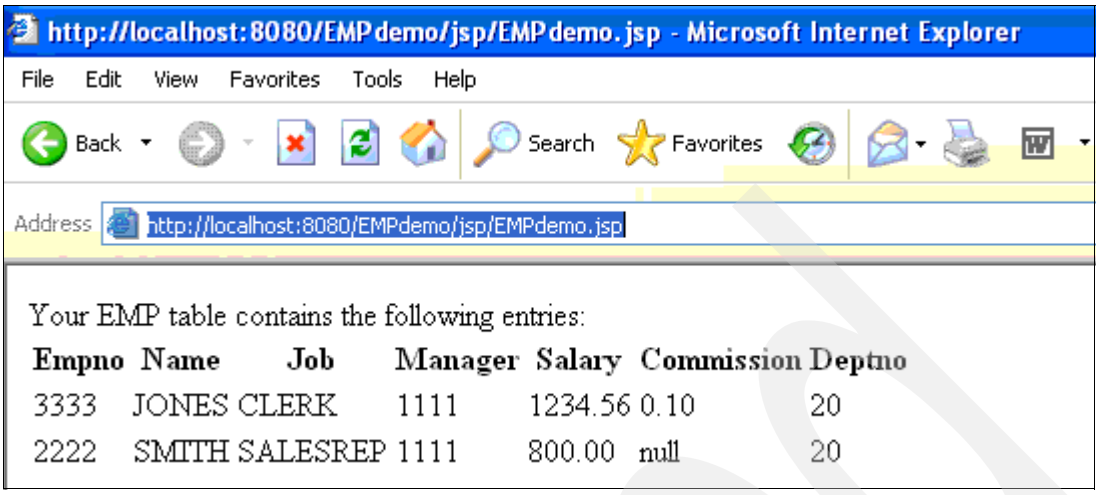


Figure 8-17 EMPdemo sample

8.7 Using a database pool in your application

In the following section, we describe how you can use a JNDI name in a Web module to access a database using a database pool.

Application code

If you use `jdbc/SampleDS` as the JNDI name for the data source pointing to the database pool, your application should contain lines of code similar to Example 8-5 to get a connection to the database pool.

Example 8-5 Code snippet referring to a database pool

```
try {
            InitialContext ctx = new InitialContext();
            DataSource ds = ctx.lookup("java:comp/env/jdbc/SampleDS");
            Connection con = ds.getConnection();
    } catch(NamingException e) {
            ...
    } catch(SQLException e) {
            ...
    }
```

WEB-INF/web.xml deployment descriptor file

You need a resource reference for the JNDI name to be present in the deployment descriptor of the module. Example 8-6 shows an example for the JNDI name jdbc/SampleDS.

Example 8-6 web.xml deployment descriptor with a database reference

```
<resource-ref>
   <res-ref-name>jdbc/SampleDS</res-ref-name>
   <res-type>javax.sql.DataSource</res-type>
   <res-auth>Container</res-auth>
   <res-sharing-scope>Shareable</res-sharing-scope>
</resource-ref>
```

WEB-INF/geronimo-web.xml

Add the entries in Example 8-7 in the deployment plan to relate the resource reference to the database pool. These entries create a mapping between the JNDI name that is used in your application and the database pool that is configured in Community Edition using the Database Pool's wizard. The wizard deploys the database pool connection in the console.dbpool GroupId. As you already saw, the database pool contains the physical implementation details of the database.

Example 8-7 geronimo-web.xml deployment plan with a database pool dependency

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app
    xmlns="http://geronimo.apache.org/xml/ns/j2ee/web-1.1">
    <environment>
        <moduleId>
            <artifactId>SampleWebApplication</artifactId>
        </moduleId>
        <dependencies>
            <dependency>
                <groupId>console.dbpool</groupId>
                <artifactId>pool_name</artifactId>
            </dependency>
        </dependencies>
    </environment>

    <context-root>/SampleWebApp</context-root>

    <!-- security settings, if any, go here -->

    <resource-ref>
          <ref-name>jdbc/SampleDS</ref-name>
          <resource-link>pool_name</resource-link>
    </resource-ref>
</web-app>
```

The *pool_name* in the <resource-link> element refers to the name given to the database pool, for example in Figure 8-3 on page 101 the name given to the pool was `DB2-Pool-1`. In Figure 8-6 on page 104, the name was `MSSQL-Pool-1`.

8.8 Connection parameters for database managers

Table 8-1 contains a list of JDBC driver classes and the format of the JDBC Connect URLs that you should be using to get a connection with some of the known database managers. Refer to vendor Web sites for details about the database managers that are not listed here.

Table 8-1 Database driver information

Name of database manager	JDBC driver class	JDBC Connect URL format
DB2	com.ibm.db2.jcc.DB2Driver	jdbc:db2://*hostname:portno/db name*
MySQL	com.mysql.jdbc.Driver	jdbc:mysql://*<hostname:portno /dbname*

Name of database manager	JDBC driver class	JDBC Connect URL format
Oracle (10g)	oracle.jdbc.driver.OracleDriver	jdbc:oracle:thin:@//*hostname:portno/dbname*
Oracle (9i)	oracle.jdbc.driver.OracleDriver	jdbc:oracle:thin:@<*hostname:portno/dbname*
SQL Server 2005 - using Microsoft JDBC driver	com.microsoft.sqlserver.jdbc.SQLServerDriver	jdbc:sqlserver://*hostname:portno*;databaseName=*dbname*
SQL Servers prior to 2005 using Microsoft JDBC driver	com.microsoft.jdbc.sqlserver.SQLServerDriver	jdbc:microsoft:sqlserver://*hostname:portno*;databaseName=*dbname*;SelectMethod=cursor
Derby Embedded	org.apache.derby.jdbc.EmbeddedDriver	jdbc:derby:*path to database*;create=true

8.9 Summary of references

- WebSphere Application Server Community Edition V2.0 documentation

  http://publib.boulder.ibm.com/wasce/V2.0.0/en/index.html

  - Configuring a database connection pool

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/database-pool.html

9

Application development tools

WebSphere Application Server Community Edition V2 provides a Web Tool Platform (WTP) Server Adapter with which you can control existing local or remote Community Edition servers, deploy applications, and analyze the code within the Eclipse IDE. This adapter supports version 1.1.0.x and 2.0.0.x and runs on Windows and Linux development environments running on Intel or AMD platforms.

In this chapter, we describe the Community Edition WTP Server Adapter version 2.0:

9.1 Installing the adapter

Installing the Community Edition WTP Server Adapter consists of the following process:

1. 9.1.1, “Installing the prerequisites” on page 118
2. 9.1.2, “Installing the WTP server adapter” on page 119
3. 9.1.3, “Defining a server” on page 122

9.1.1 Installing the prerequisites

Initial requirements:

A Community Edition installation is required on the system where you plan to use the WTP Server Adapter, even if you only plan to manage remote servers with the adapter. See Chapter 3, “Community Edition installation” on page 17 for installation instructions.

A Java Runtime Environment (JRE) is required for the WTP Server Adapter installation. You can install the recommended JRE with the IBM JDK 5.0, which you can download from:

`http://www.ibm.com/developerworks/java/jdk`

If you have Community Edition already installed locally, you can use the same JDK used by it for the WTP Server Adapter.

The WTP Server Adapter was created to run in Eclipse with the Web Tool Platform (WTP) project. The Eclipse prerequisites for the adapter are at:

`http://download.boulder.ibm.com/ibmdl/pub/software/websphere/wasce/updates/`

Table 9-1 lists the Eclipse prerequisites and their download location. You can download and install these packages separately, or you can use the WTP All in One package. If you download the packages separately, you must install the Eclipse-SDK package first.

WTP All in One package: The WTP All in One package includes all of the requirements, with the exception of the Community Edition server and the JRE.

You can download it from:

`http://download.eclipse.org/webtools/downloads/`

Download the latest *released* version (currently it is 2.0.1) of the All in One package. You install the package by unzipping the downloaded file.

To start Eclipse, execute *eclipseInstall*/eclipse/eclipse.exe in the Windows platform or *eclipseInstall*/eclipse/eclipse.sh in Linux.

Table 9-1 Separately downloadable prerequisites

Feature	Web site
IBM Java SDK 5.0 (also required by CE)	`http://www.ibm.com/developerworks/java/jdk`
Eclipse-SDK 3.3	`http://www.eclipse.org/downloads/moreinfo/classic.php`
Web Tool Platform (WTP) 2.0.1	`http://download.eclpse.org/webtools/downloads`

Feature	Web site
Eclipse Modeling Framework (EMF) 2.3.0	http://www.eclipse.org/modeling/emf/downloads or through Eclipse Update Manager
Graphical Editing Framework (GEF) 3.3	http://download.eclipse.org/tools/gef/downloads/index.php or through Eclipse Update Manager
Data Tools Platform (DTP) 1.5	http://www.eclipse.org/datatools/downloads.php or through Eclipse Update Manager
Test and Performance Tools Platform (TPTP) 4.3.0 *(Only needed if Server Profiling will be needed)	http://www.eclipse.org/tptp/home/downloads or through Eclipse Update Manager

9.1.2 Installing the WTP server adapter

Installation instructions are also at the following Web site:

http://publib.boulder.ibm.com/wasce/V2.0.0/en/eclipse.html#Eclipse-PrerequisiteSoftware

You have four options for installing the adapter:

- Using the “Download additional server adapters” link
- Using the Eclipse Update Manager
- Downloading and creating a local update site for Eclipse
- Downloading and unzipping the files directly into Eclipse directory

The simplest method is to use the Update Manager; however, this might not work if your network places restrictions on accessing the Internet. In this case, use one of the last two approaches in the list.

Using the Eclipse Update Manager to install

To use the Eclipse Update Manager to install the server adapter:

1. Open Eclipse Update Manager feature by starting Eclipse, and selecting **Help** → **Software Updates** → **Find and Install**.
2. Select **Search for new features to install**.
3. Add the following Web site as a remote update site:

   http://download.boulder.ibm.com/ibmdl/pub/software/websphere/wasce/updates/
4. Click **Finish**.
5. A list with the available features is displayed. Select the **Community Edition Runtime**, the **Server adapter** and the **Core features**.
6. Click **Finish** to start the installation.

Configuring the server runtime

After the server adapter is installed, configure Community Edition server runtime:

1. Select **Window** → **Preferences** → **Server** → **Installed Runtime**, and then click the **Add** button. A list of the available server runtimes is displayed.
2. Select **IBM WASCE v2.0**, as shown in Figure 9-1 on page 120. If you are working with your locally installed server, check the **Also create a new local server box** to define the local server to the adapter.

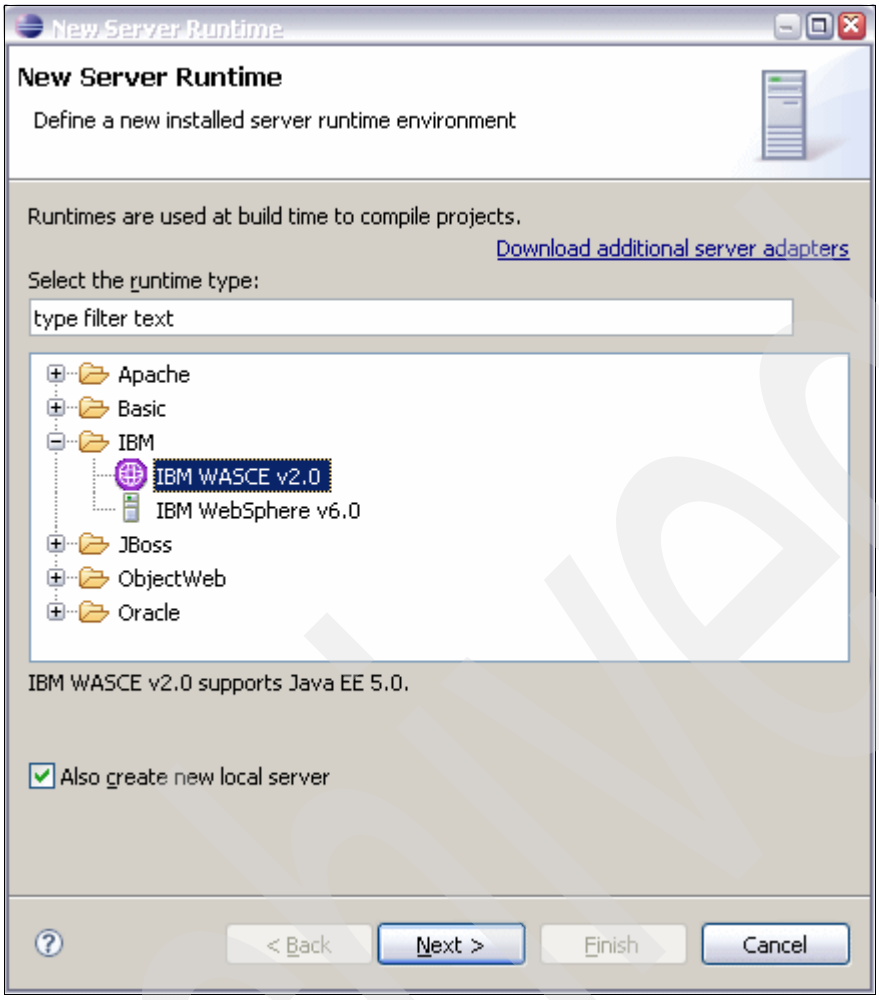


Figure 9-1 Selecting a server type

Click **Next**.

3. Select the IBM 5.0 JDK you installed (the name that is displayed might vary). In the Application Server Installation Directory field, insert the path where the Community Edition server is installed, as shown in Figure 9-2 on page 121. Do *not* use the **Download and Install** button. You must install the Community Edition Server first.

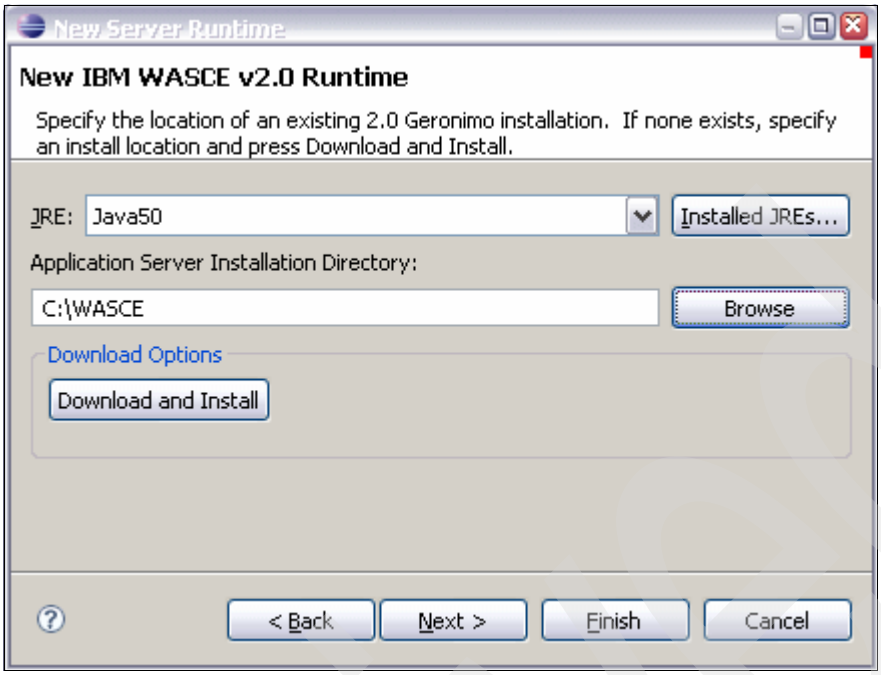


Figure 9-2 Setting server runtime

4. If the **Next** button is active, click **Next**. This takes you to the next panel where you can define the location, ports, and sign on information for the local server, as seen in Figure 9-3. This information must match the configuration for the existing server; otherwise, click **Finish**.

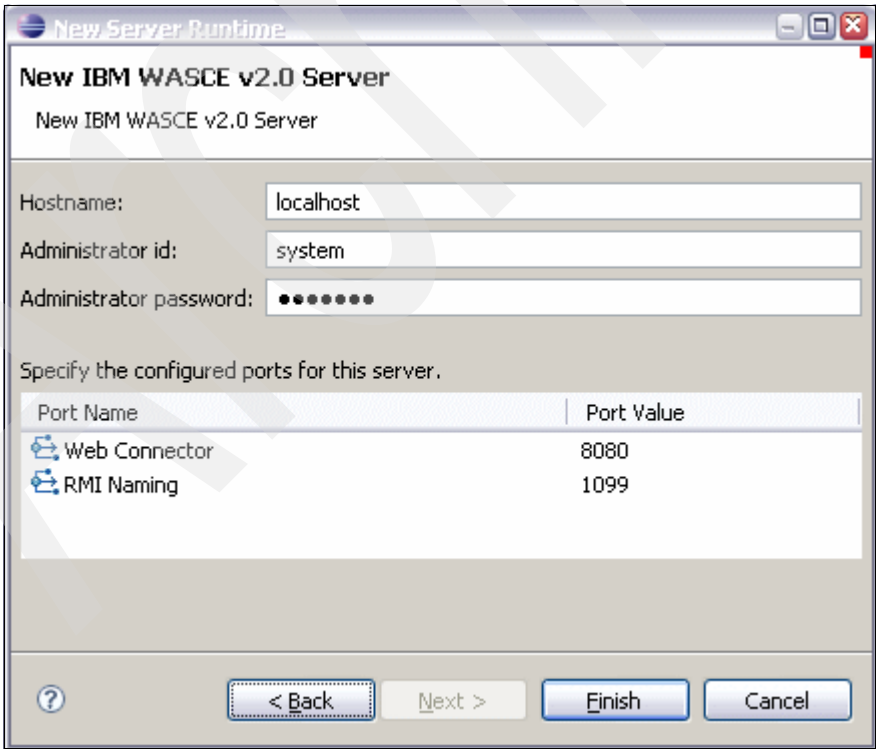


Figure 9-3 Define the parameters for the local host

9.1.3 Defining a server

You can define existing local and remote servers to the adapter, which allows you to manage the server (start, stop, restart, if the server is local). For local and remote servers, it allows you to deploy applications.

To define a server:

1. Open the Eclipse workspace, and select the Java EE perspective.
2. Select **File** → **New** → **Other** → **Server** → **Server** → **Define new Server.** Figure 9-4 is displayed.

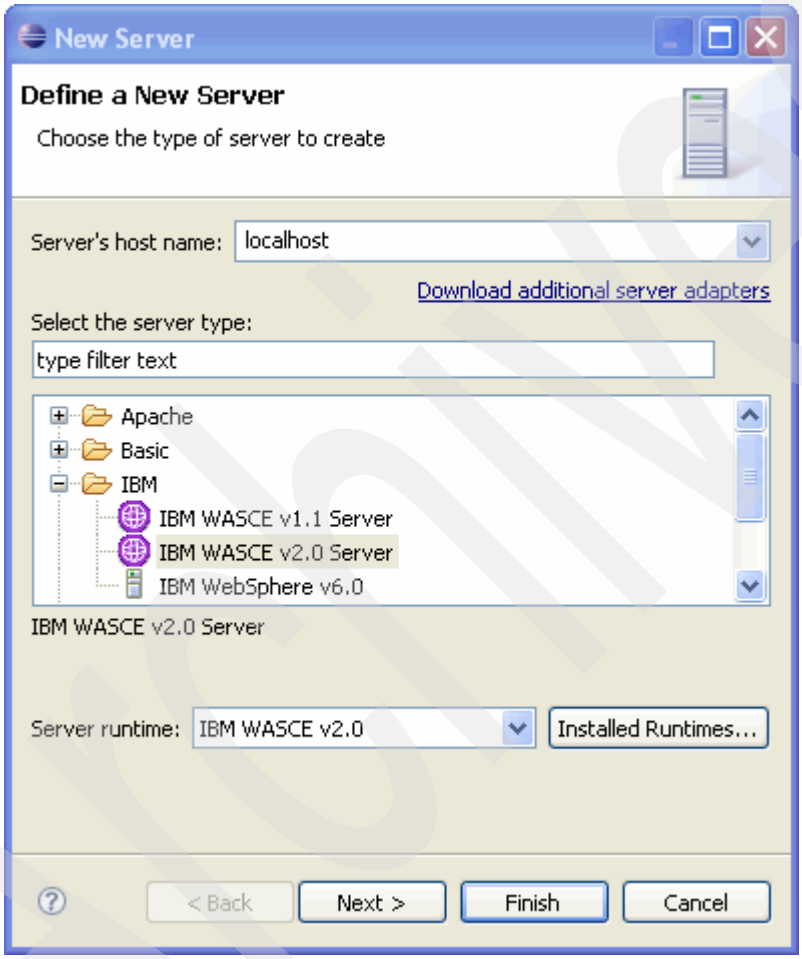


Figure 9-4 Defining a new server

3. In Figure 9-4, enter the server host name. If the server is local, use localhost. Select **IBM WASCE v2.0 Server**, and click **Next.**
4. Configure the host name, administrator ID, password, and the ports used for the Web and RMI Connector. These parameters must match with the server configuration, as shown in Figure 9-5 on page 123. For more information about these parameters, review 9.2.1, “Configuring the server” on page 124.

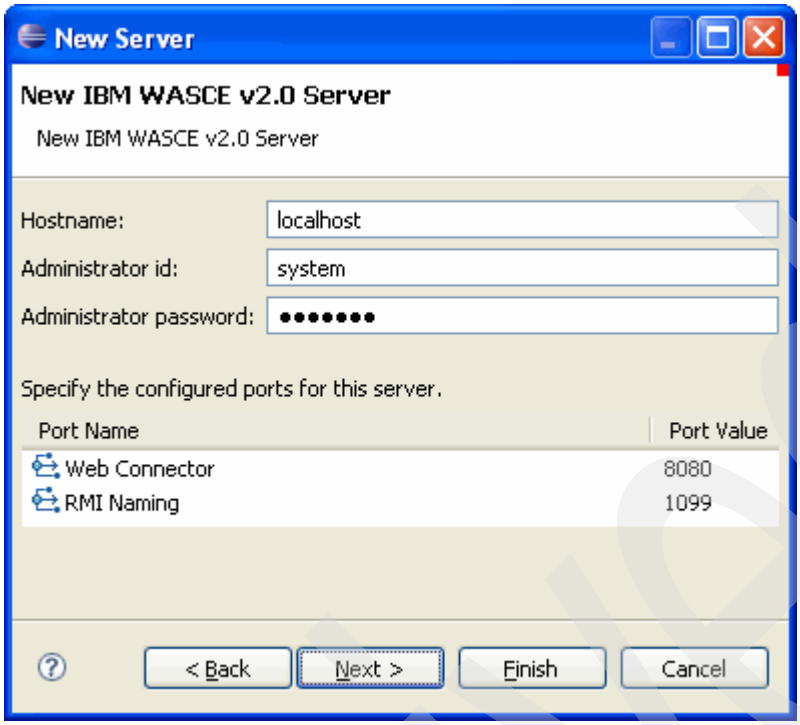


Figure 9-5 Server initial configuration

5. With this last step, you can add applications that exist in the workspace to the server. Open the Servers view to see the new server, as shown in Figure 9-6.

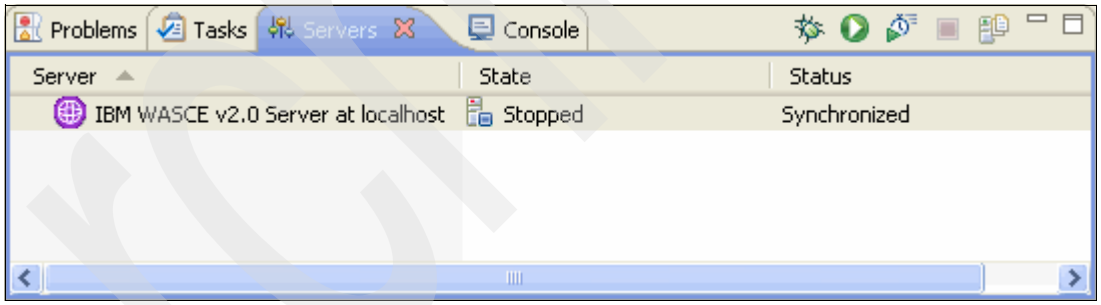


Figure 9-6 Servers view

6. If you defined a local server, you can start it by clicking the **Start the server** icon .

   If a server is already running using the same ports that you selected, you cannot start the new server. An error message is displayed that informs you that there is already a program using that port; however clicking on the error message connects you to the currently running server.

   If the server is a remote server, you must start (and stop) it from the remote machine that is running. After the remote server is configured, it takes some time to show the remote server status.

9.2 Using the adapter

With the 2.0 adapter, you can manage Version 2.0 and 1.1 Community Edition servers. Using the adapter, you can also directly deploy applications that are being developed in the Eclipse workspace, debug, and profile them while they are running on the server.

9.2.1 Configuring the server

You can configure the properties of a server by double clicking the server in the Servers view. It is possible to configure the following parameters:

- General Information (Figure 9-7) - Specify the host name and other common settings.
  - Server name - the name of the server.
  - Host name - the host name where the server is running, which allows remote servers to be configured.
  - Runtime - the server runtime used in this server. In this case, it would be a Community Edition Server runtime.

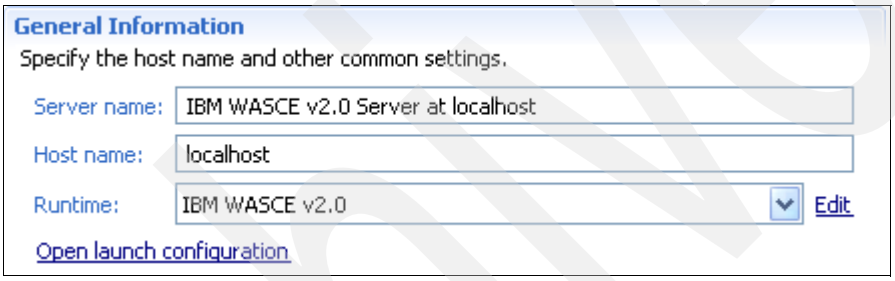


Figure 9-7 Server general configuration section

- Security (Figure 9-8) - Specify the security settings:
  - User ID - the user id used to protect the server administration.
  - User password - the password for that user ID.

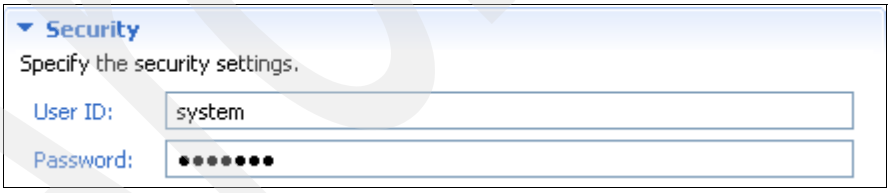


Figure 9-8 Server security configuration

- Enable in-place shared library support (Figure 9-9) - allows you to download jar files directly to the var/shared/lib folder on the Community Edition server, which makes the jar files part of the shared library.

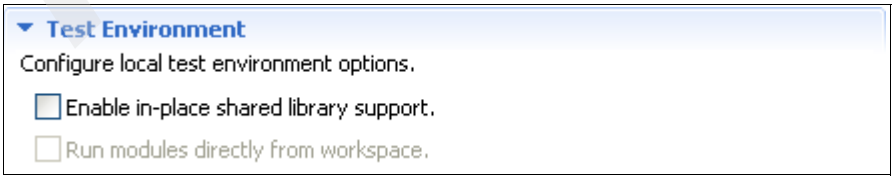


Figure 9-9 Shared library support configuration

- Automatic Publishing (Figure 9-10) - Override when the server is automatically published. The options are:
  - Use the default publishing settings - use the workspace configuration for publishing.
  - Never publish automatically - never publishes the workspace code automatically.
  - Override default settings - configures the publishing approach for this server only.

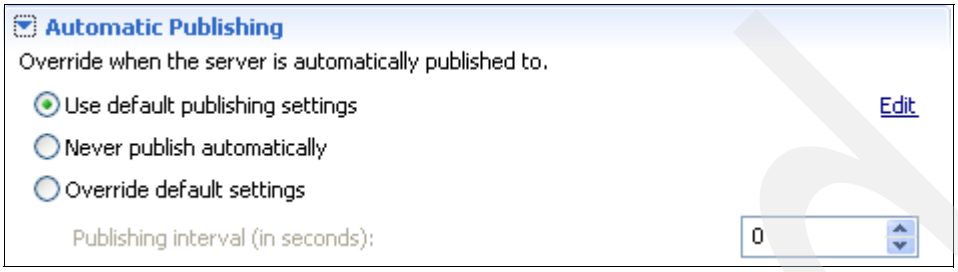


Figure 9-10 Automatic publishing configuration

- Port Configuration (Figure 9-11) - specify the ports for this server instance:
  - HTTP Port - the port the server will serve the HTTP protocol.
  - RMI Naming - the port the server will listen for the RMI protocol used by the deploy tool and server control.

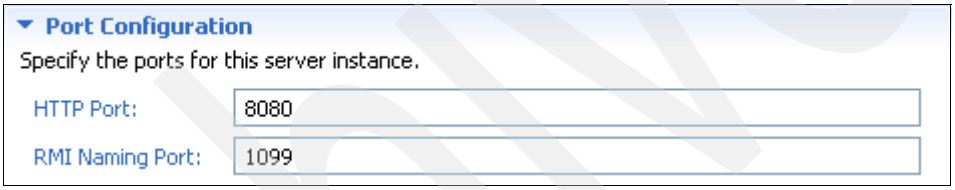


Figure 9-11 Port configuration settings

- Console Output (Figure 9-12) - set the server console output log level.

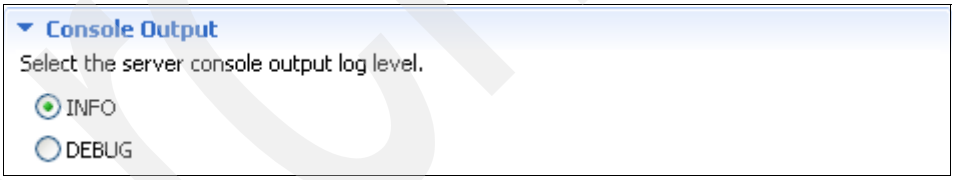


Figure 9-12 Log level settings

- Server Startup (Figure 9-13) - Specify the server startup constraints, which is used to track server status during initialization and administration. The fields are:
  - Ping Delay - set the timeout a ping to the server can have.
  - Ping Interval - what is the interval of each ping sent.
  - Maximum Pings - the maximum number of pings issued to a server.

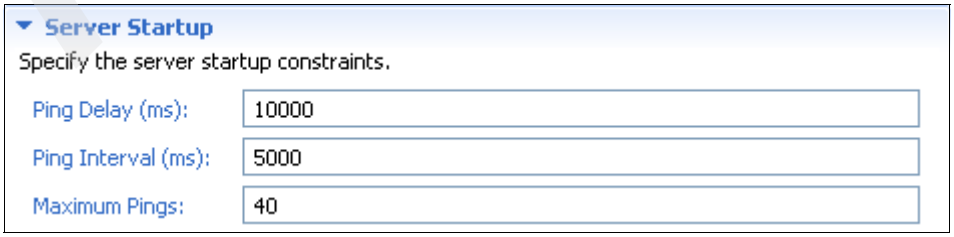


Figure 9-13 Server startup constraint settings

- Server Java VM arguments (Figure 9-14) - Specify the Java VM Arguments to be provided on server startup.

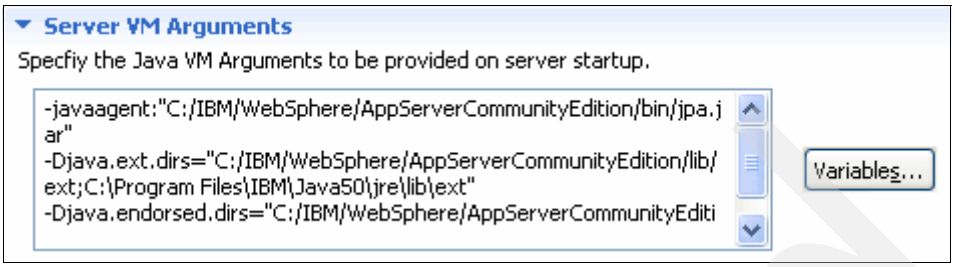


Figure 9-14 Virtual machine arguments

9.2.2 Basic server administration

For more information about basic server administration see:

http://publib.boulder.ibm.com/wasce/V2.0.0/en/using-a-server-in-eclipse.html

There are two ways to manage a server from the Servers view:

- Right-click over the server name to display a context menu, as shown in Figure 9-15.

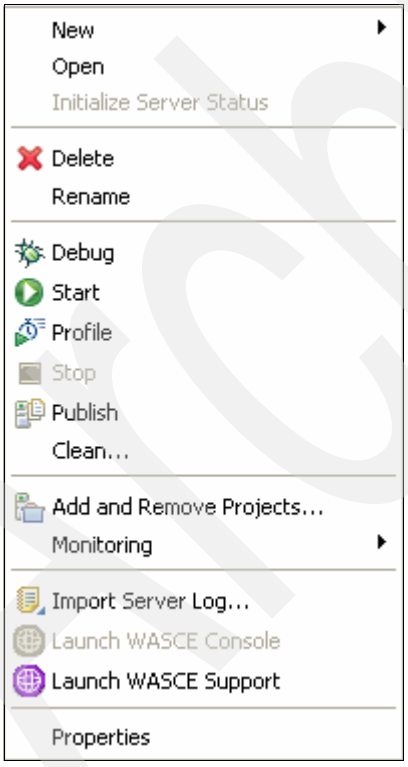


Figure 9-15 Server context menu

- The icons at the top of the view provide common control functions. Hover the mouse over a button to display its function, as shown in Figure 9-16 on page 127.

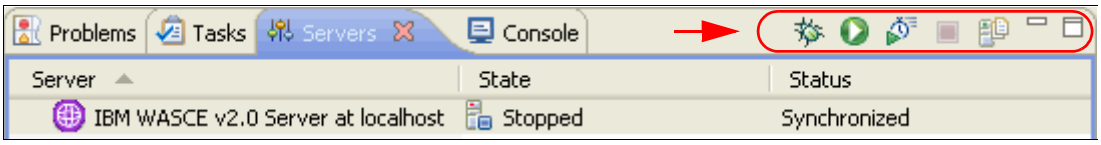


Figure 9-16 Servers view

The following list contains some of the most commonly used functions that are available from the icons in the view, the context menu, or both:

- Start the server

  You can only start local servers using this option. You must start remote servers on the remote system using the startup command. If the remote server is started, the adapter recognizes this and changes the status to Started.

- Stop the server

  You can use this option to stop both local and remote servers.

- Restart the server

  This option restarts local servers. It stops remote servers but does not restart them.

- Publish to the server

  The server re-publishes the code from the workspace to the server, which you can also do automatically with a setting in the server configuration (where). This option only re-publishes applications that are already installed. To add a new application, use the **Add and Remove Projects** option.

- Launch WASCE console

  The administrative console page opens in the Eclipse internal Web browser.

- Launch Support pages

  Launches the support page for Community Edition. You need an internet connection to use this feature.

9.2.3 Deploying applications to a server

To deploy an application to a local server using the Eclipse Adapter, right-click the application project, select **Run as** → **Run on Server**, then select the server and finish. The application is deployed and started into the server, and the Eclipse internal Web browser (in case of Web application).

An alternative method is to right-click the server in the Server view, and select **Add and Remove Projects**.

9.2.4 Debug mode

It is possible to debug an application that is running on Community Edition server using Eclipse Adapter 2.0. You must start the server in debug mode. This is only possible for local servers:

1. In the Servers view, right-click a stopped server, and select **Debug**.
2. Switch to Debug perspective to add breakpoints to your application. When you run the application, the debugger stops the program execution on that breakpoint.

   See Figure 9-17 on page 128.

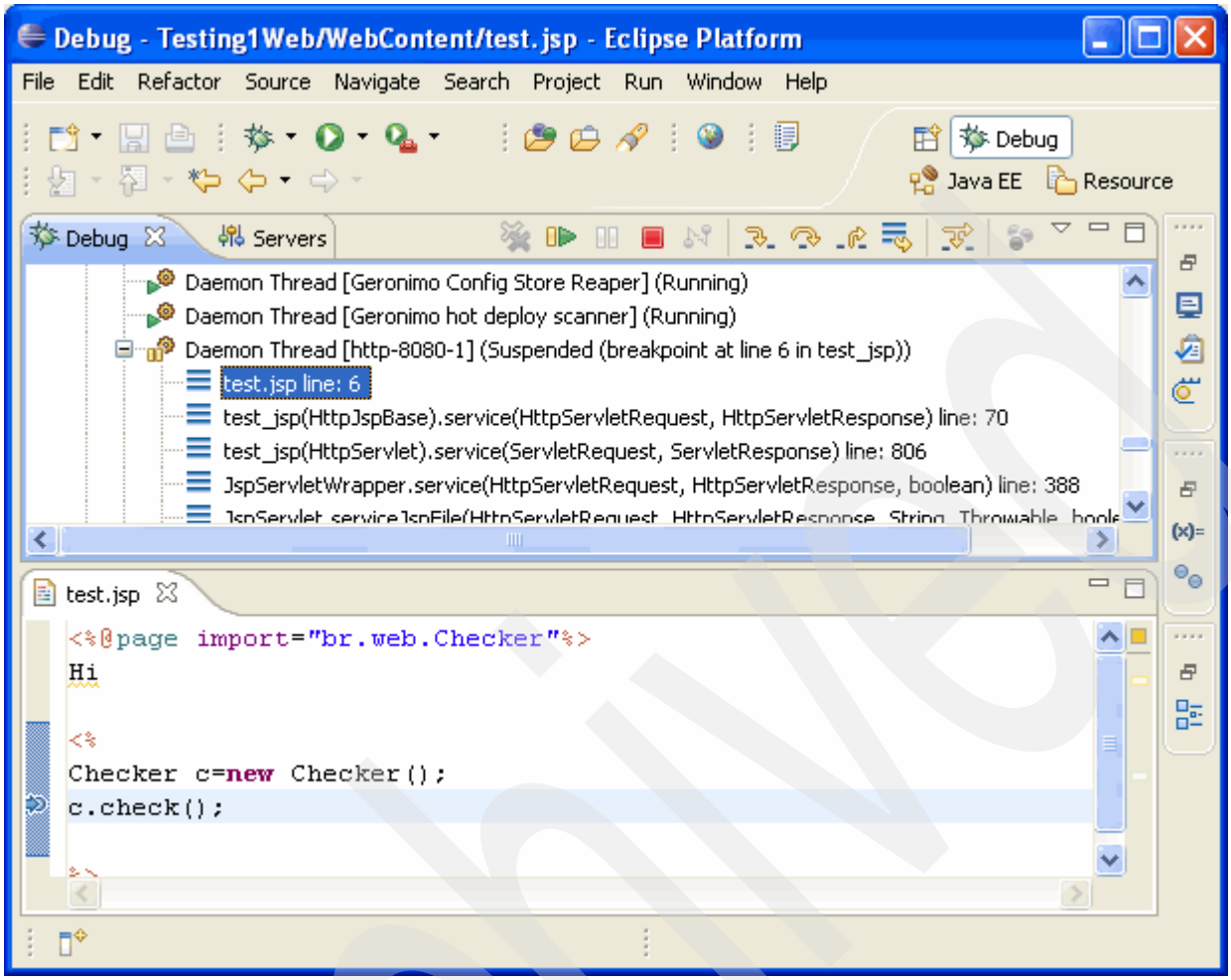


Figure 9-17 Debugging an JSP by using a break point

9.2.5 Code portability analyzer

The plug-in also provides the code portability analyzer. This tool analyzes code, checking whether it is portable or not to other servers. This is useful because Community Edition server runtime includes classes that are unique to that server. If an application uses these classes, it might impact the portability of the application to other servers. To enable the analyzer for Community Edition servers, you need the Test and Performance Tools Platform (TPTP) Eclipse plug-in.

After you install TPTP, you must extract the following file into your workspace: *eclipse_home*/features/om.ibm.wasce.feature_2.0.0/TPTrules.zip.

This file contains the rules to be checked against your code:

To execute the analyzer, use the Project Explorer view in the Java EE perspective:

1. Right-click your project, and select **Analysis** → **"Open Analysis Dialog"** and the Analysis dialog is displayed.
2. Right-click **Analysis**, and select **New**. A new analysis configuration is created.

3. You can define the scope of the analysis and the rules that are used in the analysis. Open the Rules tab to select the WASCE rules. Change the rule name to WASCE_Rules, as shown in Figure 9-18.

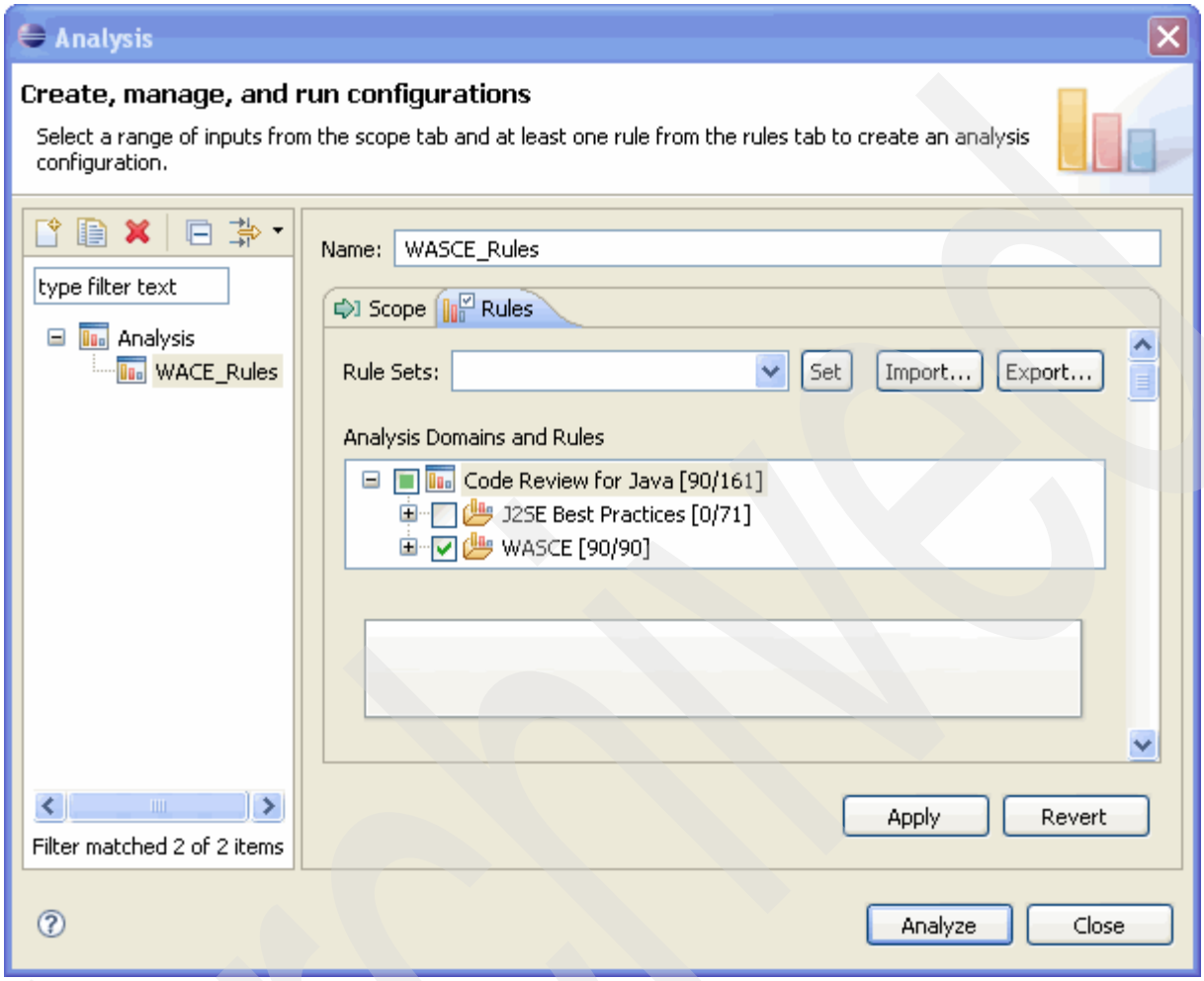


Figure 9-18 Analysis Dialog panel with the WASCE rules selected

4. To check the application, click **Analyze** in the Analysis window, or right-click the application, and select **Analyze** → **WASCE_Rules**. The code is analyzed using the selected rules.

   The results are displayed in the Analysis Results view. Possible portability issues are listed, as shown in Figure 9-19 on page 130.

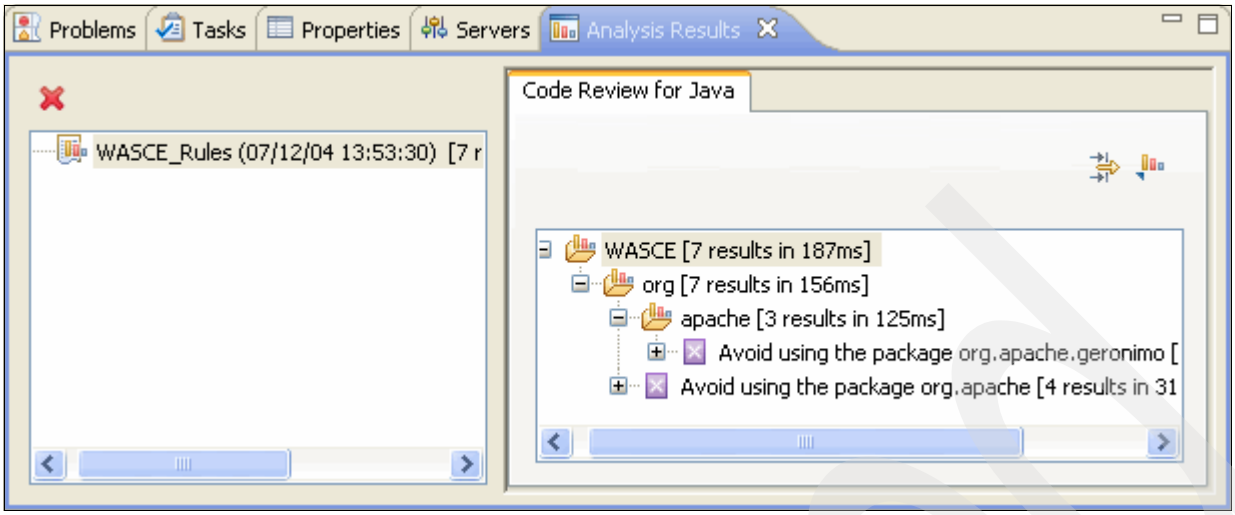


Figure 9-19 Analysis Results view with some portability issues listed

For more information see:
http://publib.boulder.ibm.com/wasce/V2.0.0/en/portability-using-eclipse.html

9.2.6 Troubleshooting the WTP server adapter 2.0

The following sections include some problems that were experienced with the adapter.

IP address change can orphan a server

If the IP of the host changes, Eclipse might not be able to send shutdown commands to the servers that are running on that machine. If this happens, stop the server's javaw process, which also impacts any server's unsaved configurations.

Unable to open the deployment plan editor

The WTP Server Adapter version 2.0 has a graphical editor for 1.1 deployment descriptors but not for 2.0. If you need to edit a 2.0 deployment descriptor, use the XML editor.

Unable to start or restart the server

There are several aspects that can cause the server not to be started or restarted using Eclipse. Some common situations are:

- The IP address of the server changed.
- You are trying to restart a remote server. You must start remote servers from the remote machine. When the remote server is started, the adapter recognizes this and changes the status to Started in the Servers view. If this does not happen, make sure the ports used are correct and are not blocked by a firewall. Also ensure that the user ID and password are correct in the server configuration.
- Wrong port configuration. Be sure the port configuration in the adapter matches the port configuration for the server.
- The JAVA_HOME environment variable is not defined correctly. During the installation of Community Edition, the install wizard searches for supported Java environments and writes the location to the *wasceHome*/bin/setenv.bat(sh) script. If the Java environment that is used is changed (moved, deleted, uninstalled or damaged), you need to update the **WASCE_JAVA_HOME** variable in setenv with the location of the supported Java environment.

For more information see:

http://publib.boulder.ibm.com/wasce/V2.0.0/en/eclipse.html#Eclipse-TroubleshootingWASCEWTPServerAdapter

9.3 Summary of references

- IBM developer kits

  http://www.ibm.com/developerworks/java/jdk
- Eclipse Update site for IBM WebSphere Application Server Community Edition

  http://download.boulder.ibm.com/ibmdl/pub/software/websphere/wasce/updates/
- Web Tools Platform downloads

  http://download.eclipse.org/webtools/downloads/
- Eclipse Classic

  http://www.eclipse.org/downloads/moreinfo/classic.php
- EMF download

  http://www.eclipse.org/modeling/emf/downloads
- Graphical Editing Framework (GEF) downloads

  http://download.eclipse.org/tools/gef/downloads/index.php
- Eclipse Data Tools Platform (DTP) Project Downloads

  http://www.eclipse.org/datatools/downloads.php
- Test and Performance Tools Platform (TPTP) Downloads

  http://www.eclipse.org/tptp/home/downloads
- Installing the WASCE WTP Server Adapter: prerequisite software

  http://publib.boulder.ibm.com/wasce/V2.0.0/en/eclipse.html#Eclipse-PrerequisiteSoftware
- WebSphere Application Server Community Edition V2.0 documentation

  http://publib.boulder.ibm.com/wasce/V2.0.0/en/index.html
  - Administrating the server in Eclipse

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/using-a-server-in-eclipse.html
  - Developing portable Java EE assets using Eclipse

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/portability-using-eclipse.html
  - Installing the WASCE WTP Server Adapter

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/eclipse.html#Eclipse-TroubleshootingtheWASCEWTPServerAdapter

10

Packaging, deploying, and managing applications

In this chapter, we describe how to package, deploy, and manage applications for Community Edition.

We include the following topics:

10.1 Introduction to application packaging

The Java Enterprise Edition platform specifications define standard deployment descriptors for application modules. The descriptors are XML files that describe the resources that are provided or needed by an application in a declarative way, for example, the deployment descriptor for a Java EE Web application is the web.xml file that is located in the WEB-INF folder of the Web archive (WAR) file. For Java EE enterprise applications, the deployment descriptor is the application.xml file that is located in the folder META-INF of the enterprise archive (EAR) file.

Although you must do some configuration in the standard Java EE deployment descriptors, the Java EE specifications also provide flexibility for implementation, which allows vendor or environment-specific deployment descriptors. In Community Edition, these files are usually called *deployment plans*, and they bind resources in the Java asset to resources that are configured in the server.

In the following sections, we describe the Community Edition-specific deployments plans and how to modify them.

Community Edition sample applications:

IBM provides a set of sample applications for Community Edition that are useful as references during development. Many of the examples in this chapter refer you to a specific sample application that illustrates the topic.

Download the Community Edition sample applications from:

`http://www.ibm.com/developerworks/downloads/ws/wasce`

For information about the sample applications, see:

`http://publib.boulder.ibm.com/wasce/V2.0.0/en/samples.html`

10.2 Deployment plan packaging options

In this section, we discuss the assets that you can deploy and the possible deployment plan packaging options.

10.2.1 Deployable assets

The assets that you can deploy in Community Edition using any of the deployment methods are:

- Web modules: the Java EE Web module is packaged in Web archives and uses the geronimo-web.xml deployment plan to describe its deployment.
- Enterprise modules: the Java EE enterprise module is packaged in enterprise archives and uses the geronimo-application.xml deployment plan to describe its deployment.
- EJB modules: the Java EE EJB module is packaged in Java archives (JAR) and uses the openejb-jar.xml deployment plan to describe its deployment.
- Enterprise client: the Java EE enterprise client module is packaged in Java archives and uses the geronimo-application-client.xml deployment plan to describe its deployment.

- Connector: the Java EE connector is packaged in Resource adapter archive (RAR) and uses the geronimo-ra.xml deployment plan.
- GBeans: GBeans can be deployed using the common deployment methods. They are packaged in Java Archive (JAR) files and it does not have a standard deployment plan file name.

10.2.2 No deployment plan

In simple applications where all bindings are done using the absolute reference to a resource, you can deploy the application without a deployment plan. The server uses default values for any required bindings, which is useful when you deploy sample applications, migrate simple assets, or while you develop applications.

However, it is always a good idea to have a simple deployment plan file to provide the moduleID and context-root for the application. You can add the separate deployment plan file using the command line deploy or console deploy. A simple deployment file example is in the "hello" sample in:

samples\applications\hello\src\main\webapp\WEB-INF\geronimo-web.xml

10.2.3 Using a deployment plan

For applications that require that you define bindings between server resources and Java EE assets, you need a deployment plan.

There are two packaging strategies for managing deployment plans:

- Packaging the deployment plan in the Java EE asset archive file. This option is good when you need to reduce the effort to distribute and deploy the asset. Only one file is managed and deployed. However if a deployment plan needs to be changed you must extract the archive contents, change the deployment descriptor, and repackage the asset.
- Having a separate deployment plan file from the Java EE asset archive file. This option is good when frequent changes are done on the bindings and other declarative configurations. You only need to change the deployment plan and deploy it again. Using this strategy, you manage two files, the archive and the deployment descriptor plan.

More information about the deployment descriptor packaging options are at:

`http://publib.boulder.ibm.com/wasce/V2.0.0/en/deployment-plans.html`

10.3 Deploying and undeploying applications

In this section, we describe the options for deploying and undeploying applications in Community Edition.

10.3.1 Command-line deploy

It is possible to deploy an application using the command line. The **`deploy`** command manages Java EE assets and is located in the *wasceHome*/bin/ directory.

The simplest use of the **deploy** command is:

```
deploy --user <user_name> --password <password> deploy <archive_file>
[deployment_plan]
```

In the command:

- *user_name* is the user authorized to manage the Community Edition server. The default is system.
- *password* is the user password. The default is manager.
- *archive_file* is the Java EE archive file path.
- *deployment_plan* is the deployment plan for this archive. This is an optional parameter.

Figure 10-1 shows an example of installing one of the sample applications using the **deploy** command. In this sample, the deployment plan geronimo-application.xml is included in the archive file.

```
C:\WebSphere\CE\bin>deploy --user system --password manager deploy
c:\WASCE_samples\applications\calculator-stateless-pojo/calculator-stateless-ea
r/target/calculator-stateless-ear-2.0.0.1.ear

Using GERONIMO_BASE:   C:\WebSphere\CE
Using GERONIMO_HOME:   C:\WebSphere\CE
Using GERONIMO_TMPDIR: var\temp
Using JRE_HOME:        C:\Java50\jre
    Deployed
    org.apache.geronimo.samples/calculator-stateless-ear/2.0.0.1/ear
      `-> calculator-stateless-war.war @ /calculator-stateless
      `-> calculator-stateless-ejb.jar
    Deployed org.apache.geronimo.configs/axis/2.0.1/car
    Deployed org.apache.geronimo.configs/axis2/2.0.1/car
C:\WebSphere\CE\bin>
```

Figure 10-1 Installation using the deploy command

Remote command-line deploy

The **deploy** command is also used for deploying assets to remote servers. To deploy to a remote server, the command arguments are:

```
deploy --host <host> --port <port> --user <user_name> --password <password>
<archive_file> [deployment_plan]
```

The two first arguments are the main difference comparing it with the local **deploy** syntax. The arguments are:

- *host* is the name of the host where the target server is running.
- *port* is the port number where the target server listens for RMI requests, typically 1099.
- *user_name* is the user who is authorized to manage the Community Edition server.
- *password* is the user password.
- *archive_file* is the Java EE archive file path.
- *deployment_plan* is the deployment plan for this archive. This is an optional parameter.

Redeploying an application

If you are redeploying an application, you can use the following command, where *configId* contains the configId attribute to identify a currently deployed application:

```
deploy --user <user_name> --password <password> redeploy <archive_file>
[deployment_plan] [configId]
```

Figure 10-2 shows an example of redeploying an application.

```
C:\WebSphere\CE\bin>deploy --user system --password manager redeploy
c:\download
s\WASCE_samples\applications\calculator-stateless-pojo/calculator-stateless-ear
/
target/calculator-stateless-ear-2.0.0.1.ear
Using GERONIMO_BASE:   C:\WebSphere\CE
Using GERONIMO_HOME:   C:\WebSphere\CE
Using GERONIMO_TMPDIR: var\temp
Using JRE_HOME:        C:\Java50\jre
    No ModuleID or TargetModuleID provided.  Attempting to guess based
    on the content of the archive.
    Attempting to use ModuleID
    'org.apache.geronimo.samples/calculator-stateless-ear/2.0.0.1/ear'
    Stopped
    org.apache.geronimo.samples/calculator-stateless-ear/2.0.0.1/ear
    Unloaded
    org.apache.geronimo.samples/calculator-stateless-ear/2.0.0.1/ear
    Uninstalled
    org.apache.geronimo.samples/calculator-stateless-ear/2.0.0.1/ear
    Deployed
    org.apache.geronimo.samples/calculator-stateless-ear/2.0.0.1/ear
    Started
    org.apache.geronimo.samples/calculator-stateless-ear/2.0.0.1/ear
    Redeployed
    org.apache.geronimo.samples/calculator-stateless-ear/2.0.0.1/ear


C:\WebSphere\CE\bin>
```

Figure 10-2 Syntax for redeploying an application

If you are redeploying an application remotely, you can use the following command:

```
deploy --host <host> --port <port> --user <user_name> --password <password>
redeploy <archive_file> [deployment_plan] [configId]
```

Undeploying an application

To undeploy an application, use the following command:

```
deploy --user <user_name> --password <password> undeploy <configId>
```

To undeploy an application that is running on a remote server, use the following command:

```
deploy --host <host> --port <port> --user <user_name> --password <password>
undeploy <configId>
```

For more advanced information about the **deploy** command, visit:

http://publib.boulder.ibm.com/wasce/V2.0.0/en/deploy.html

Finding the configID of currently deployed applications

To get a list of the configIDs of the currently deployed applications, execute the following command:

```
deploy --host <host> --port <port> --user <user_name> --password <password>
list-modules
```

- *host* is the name of the host where the target server is running.
- *port* is the port number where the target server listens for RMI requests.
- *user_name* is the user who is authorized to manage the Community Edition server.
- *password* is the user password.

The configID parameter

The configID is a concatenation of the module identification parameters. A configID is created in the format:

groupId/artifactId/version/type

So, for example, a Web module with groupID called MyGroup, artifactID called MyApplication, version 2 would have the following configID:

MyGroup/MyApplication/2/war

For more information about module identification, see 10.6.2, “Module identity” on page 150.

10.3.2 Administrative console deploy

You can use the administrative console to deploy Java EE assets. This method is more user friendly because it provides a Web interface to deploy applications on local and remote servers.

To install a new asset:

1. Go to the console navigation area, and select **Deploy New**. Using the options for deploy, you can have a separate deployment plan.
2. Browse to the archive, and click **Install**, as shown in Figure 10-3. A message is displayed at the top of the panel indicating whether the deployment completed successfully or not.

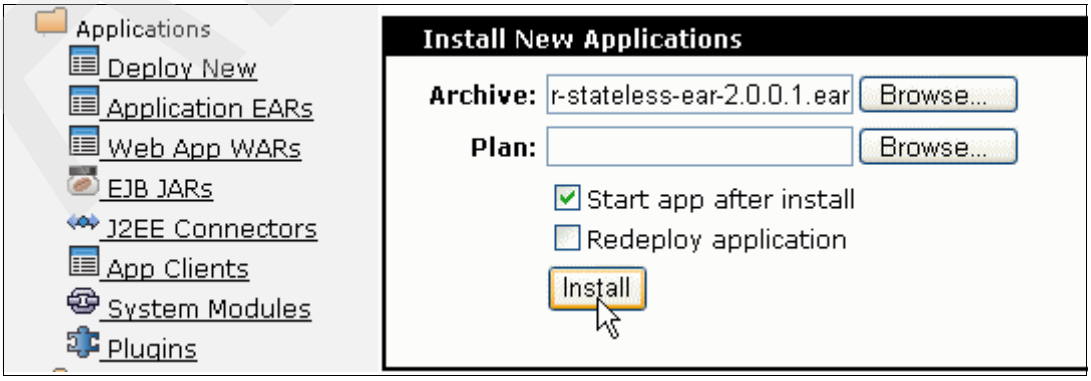


Figure 10-3 Application menu and the deployment form

Select the **Redeploy application** option, which is a shortcut to stop and undeploy an application, then redeploy it.

After you deploy the asset, you can view its status by selecting the asset type in the navigation panel. The application is identified by a component name, which is created as follows:

group_id/artifact_id/version/artifact_type

- *group_id* is the short name of the related group that is set in the deployment plan. If you do not provide a deployment plan that specifies the module identity, the default value for the group_id is `default`.
- *artifact_id* is the short name of the file that are set in the deployment plan. The default is the name of the archive file.
- *version* is the version that is set in the deployment plan. The default is the current time stamp.
- *artifact_type* is the Java artifact type (for example, war for Web artifact).

Undeploying an application using the console

To undeploy a Java EE application or module using the console:

1. Under the Applications section, click the link in the Console Navigation that corresponds to the module type you need to undeploy, for example, for Web modules, click **Web App WARs**, and for enterprise applications, click **Application EARs**.
2. Locate the application or module you want to undeploy.
3. Click **Uninstall**.
4. A confirmation message box is displayed. Click **OK** to complete the undeploy.

   A message is displayed at the bottom left corner of the portlet indicating the status of the operation.

10.3.3 Hot deploy

During a development cycle code is usually changed frequently. A quick deploy is needed to test the change. Hot deploy provides a convenient method of quickly deploying assets. You can deploy the Java EE assets manually or programmatically by copying the files into the hot deploy directory:

wasceHome/deploy

You can copy the Java archive files into the deploy directory, or you can keep the assets expanded in the deploy directory, and copy single files into it. When a new archive is put in the deploy directory, it is automatically deployed into the server. If files are updated, they are automatically refreshed on the server.

Hot deploy information is logged to the following file:

wasceHome`/var/log/server.log`

Note: If you use hot deploy for an application, you must continue to use hot deploy to update and redeploy the application. To undeploy, you can delete the application from the deploy directory, or you can use the other undeploy alternatives, which also delete the archive file from the /deploy directory.

To use hot deployment, consider the following:

- You must have access to the server file system. In order to copy the files to the deploy directory, you must have permission to write and read from that directory.
- The server must have been started at least once, so that the deploy directory is created.
- Deployment plans must be packaged with the Java EE asset. Hot deploy does not support external deployment plans.
- Server configuration archives (CARs) cannot be hot deployed.
- When the server is restarted all Java EE assets are redeployed. It increases the server start up time. Therefore, we do not recommend hot deploy for production servers. Use the deploy tool or console.
- If any other alternative is used to deploy a Java EE asset, it will not appear in the deploy directory.

Figure 10-4 shows an example of the log when an application archive is copied into the hot deployment directory.

```
10:55:34,078 INFO  [DirectoryHotDeployer] Deploying calculator-stateless-ear-2.0.0.1.ear
10:55:34,890 INFO  [config] Configuring Service(id=Default Stateless Container, type=Container,
provider-id=Default Stateless Container)
10:55:34,890 INFO  [config] Configuring Service(id=Default Stateful Container, type=Container, provider-id=Defa
Stateful Container)
10:55:34,890 INFO  [config] Configuring Service(id=Default BMP Container, type=Container, provider-id=Default B
Container)
10:55:34,890 INFO  [config] Configuring Service(id=Default CMP Container, type=Container, provider-id=Default C
Container)
10:55:34,890 INFO  [config] Configuring app: org.apache.geronimo.samples/calculator-stateless-ear/2.0.0.1/ear
10:55:35,078 INFO  [OpenEJB] Auto-deploying ejb Calculator:
EjbDeployment(deployment-id=calculator-stateless-ejb.jar/Calculator)
10:55:35,265 INFO  [config] Loaded Module: org.apache.geronimo.samples/calculator-stateless-ear/2.0.0.1/ear
10:55:37,546 INFO  [Enhance] You have enabled runtime enhancement, but have not specified the set of persistent
classes.  OpenJPA must look for metadata for every loaded class, which might increase class load times
significantly.
10:55:38,718 INFO  [startup] Assembling app:
C:\IBM\WebSphere\AppServerCommunityEdition\var\temp\geronimo-deploymentUtil32016.jar
10:55:39,046 INFO  [startup] Jndi(name=CalculatorLocal) -->
Ejb(deployment-id=calculator-stateless-ejb.jar/Calculator)
10:55:39,046 INFO  [startup] Jndi(name=CalculatorRemote) -->
Ejb(deployment-id=calculator-stateless-ejb.jar/Calculator)
10:55:39,046 INFO  [startup] Created Ejb(deployment-id=calculator-stateless-ejb.jar/Calculator,
ejb-name=Calculator, container=Default Stateless Container)
10:55:39,046 INFO  [startup] Deployed
Application(path=C:\IBM\WebSphere\AppServerCommunityEdition\var\temp\geronimo-deploymentUtil32016.jar)
10:55:40,406 INFO  [OpenEJB] invoking method create on calculator-stateless-ejb.jar/Calculator
10:55:40,484 INFO  [OpenEJB] finished invoking method create
10:55:40,593 INFO  [DirectoryHotDeployer]     Deployed
org.apache.geronimo.samples/calculator-stateless-ear/2.0.0.1/ear
10:55:40,593 INFO  [DirectoryHotDeployer]       `-> calculator-stateless-war.war @ /calculator-stateless
10:55:40,593 INFO  [DirectoryHotDeployer]       `-> calculator-stateless-ejb.jar
10:55:40,593 INFO  [DirectoryHotDeployer]     Deployed org.apache.geronimo.configs/axis/2.0.1/car
10:55:40,593 INFO  [DirectoryHotDeployer]     Deployed org.apache.geronimo.configs/axis2/2.0.1/car
```

Figure 10-4 Log file of an application deployed using hot deploy

Figure 10-5 on page 141 shows an example of the log output when a new copy of an application is placed in the deploy file, in effect, redeploying the application.

```
11:19:04,593 INFO  [DirectoryHotDeployer] Undeploying calculator-stateless-ear-2.0.0.1.ear
11:19:05,203 INFO  [JDBC] OpenJPA will now connect to the database to attempt to determine what type of database
dictionary to use.  To prevent this connection in the future, set your openjpa.jdbc.DBDictionary configuration
property to the appropriate value for your database (see the documentation for available values).
11:19:05,218 INFO  [JDBC] Using dictionary class "org.apache.openjpa.jdbc.sql.DerbyDictionary" (Apache Derby
10.2.2.0 - (485682) ,Apache Derby Embedded JDBC Driver 10.2.2.0 - (485682)).
11:19:05,218 INFO  [startup] Undeploying app:
C:\IBM\WebSphere\AppServerCommunityEdition\var\temp\geronimo-deploymentUtil32016.jar
11:19:05,484 INFO  [DirectoryMonitor] Hot deployer notified that an artifact was removed:
org.apache.geronimo.samples/calculator-stateless-ear/2.0.0.1/ear
11:19:05,515 INFO  [DirectoryHotDeployer]     Undeployed
org.apache.geronimo.samples/calculator-stateless-ear/2.0.0.1/ear
11:19:08,468 INFO  [DirectoryHotDeployer] Deploying calculator.ear
11:19:09,593 INFO  [config] Configuring Service(id=Default Stateless Container, type=Container,
provider-id=Default Stateless Container)
11:19:09,593 INFO  [config] Configuring Service(id=Default Stateful Container, type=Container, provider-id=Default
Stateful Container)
11:19:09,593 INFO  [config] Configuring Service(id=Default BMP Container, type=Container, provider-id=Default BMP
Container)
11:19:09,593 INFO  [config] Configuring Service(id=Default CMP Container, type=Container, provider-id=Default CMP
Container)
11:19:09,593 INFO  [config] Configuring app: org.apache.geronimo.samples/calculator-stateless-ear/2.0.0.1/ear
11:19:09,796 INFO  [OpenEJB] Auto-deploying ejb Calculator:
EjbDeployment(deployment-id=calculator-stateless-ejb.jar/Calculator)
11:19:10,218 INFO  [config] Loaded Module: org.apache.geronimo.samples/calculator-stateless-ear/2.0.0.1/ear
11:19:12,640 INFO  [Enhance] You have enabled runtime enhancement, but have not specified the set of persistent
classes.  OpenJPA must look for metadata for every loaded class, which might increase class load times
significantly.
11:19:13,250 INFO  [startup] Assembling app:
C:\IBM\WebSphere\AppServerCommunityEdition\var\temp\geronimo-deploymentUtil32021.jar
11:19:13,437 INFO  [startup] Jndi(name=CalculatorLocal) -->
Ejb(deployment-id=calculator-stateless-ejb.jar/Calculator)
11:19:13,437 INFO  [startup] Jndi(name=CalculatorRemote) -->
Ejb(deployment-id=calculator-stateless-ejb.jar/Calculator)
11:19:13,437 INFO  [startup] Created Ejb(deployment-id=calculator-stateless-ejb.jar/Calculator,
ejb-name=Calculator, container=Default Stateless Container)
11:19:13,437 INFO  [startup] Deployed
Application(path=C:\IBM\WebSphere\AppServerCommunityEdition\var\temp\geronimo-deploymentUtil32021.jar)
```

Figure 10-5 Log file of an application redeployed using hot deploy

Figure 10-6 shows the log file that results when an application is deleted from the folder, which undeploys the application.

```
13:49:47,437 INFO  [DirectoryHotDeployer] Undeploying calculator-stateless-ear-2.0.0.1.ear
13:49:48,015 INFO  [JDBC] OpenJPA will now connect to the database to attempt to determine what type of database
dictionary to use.  To prevent this connection in the future, set your openjpa.jdbc.DBDictionary configuration
property to the appropriate value for your database (see the documentation for available values).
13:49:48,015 INFO  [JDBC] Using dictionary class "org.apache.openjpa.jdbc.sql.DerbyDictionary" (Apache Derby
10.2.2.0 - (485682) ,Apache Derby Embedded JDBC Driver 10.2.2.0 - (485682)).
13:49:48,015 INFO  [startup] Undeploying app:
C:\IBM\WebSphere\AppServerCommunityEdition\var\temp\geronimo-deploymentUtil32033.jar
13:49:48,234 INFO  [DirectoryMonitor] Hot deployer notified that an artifact was removed:
org.apache.geronimo.samples/calculator-stateless-ear/2.0.0.1/ear
13:49:48,328 INFO  [DirectoryHotDeployer]     Undeployed
org.apache.geronimo.samples/calculator-stateless-ear/2.0.0.1/ear
```

Figure 10-6 Log file of an application undeployed using hot deploy

For more information about deployment, see:

- http://publib.boulder.ibm.com/wasce/V2.0.0/en/deploy-assets.html
- http://cwiki.apache.org/GMOxDOC20/installing-and-removing-applications.html

Maven hot deploy

Using the Maven tool, you can build and manage any Java project. The Maven tool is available at the Apache Foundation:

http://maven.apache.org/

You can use Maven with the Community Edition hot deploy feature to quickly and automatically deploy applications.

Configure Maven to deploy the assets into *wasceHome*/deploy as the target directory. After Maven builds the Java asset, the hot deploy feature automatically installs a new application or redeploys a running application.

The advantages of this approach are that you can check out the latest version of a project from a source code repository, such as CVS or Subversion, and automatically deploy it to a server. This is useful when several developers are updating the code and a daily build to a central development server is needed.

10.4 Managing applications and modules

In this section, we discuss how to administer applications and modules in terms of how to start, restart, and stop them.

10.4.1 Using the administrative console

To administer Java EE applications and modules from the administrative console:

1. From the Console Navigation area, click **Application EARs**. The installed applications are listed with their state and other information, as shown in Figure 10-7.

Installed Application EARs [view]

☐ Expert User (enable all actions on Geronimo Provided Components)

Component Name	State	Commands			Parent Components	Child Components
org.apache.geronimo.configs/uddi-tomcat/2.0.1/car	stopped	Start		Uninstall	org.apache.geronimo.configs/axis/2.0.1/car org.apache.geronimo.configs/j2ee-server/2.0.1/car org.apache.geronimo.configs/jasper/2.0.1/car org.apache.geronimo.configs/system-database/2.0.1/car org.apache.geronimo.configs/tomcat6/2.0.1/car org.apache.geronimo.configs/transaction/2.0.1/car	
org.apache.geronimo.configs/webconsole-tomcat/2.0.1/car	running	Stop	Restart	Uninstall	org.apache.geronimo.configs/connector-deployer/2.0.1/car org.apache.geronimo.configs/dojo-tomcat/2.0.1/car org.apache.geronimo.configs/j2ee-corba-yoko/2.0.1/car org.apache.geronimo.configs/j2ee-server/2.0.1/car org.apache.geronimo.configs/jasper/2.0.1/car org.apache.geronimo.configs/system-database/2.0.1/car org.apache.geronimo.configs/tomcat6/2.0.1/car org.apache.geronimo.configs/tomcat6-deployer/2.0.1/car	
org.apache.geronimo.samples/calculator-stateless-ear/2.0.0.1/ear	running	Stop	Restart	Uninstall	org.apache.geronimo.configs/axis/2.0.1/car org.apache.geronimo.configs/axis2/2.0.1/car org.apache.geronimo.configs/j2ee-corba-yoko/2.0.1/car org.apache.geronimo.configs/j2ee-server/2.0.1/car org.apache.geronimo.configs/jasper/2.0.1/car org.apache.geronimo.configs/openejb/2.0.1/car org.apache.geronimo.configs/openjpa/2.0.1/car org.apache.geronimo.configs/system-database/2.0.1/car org.apache.geronimo.configs/tomcat6/2.0.1/car	

Figure 10-7 List of installed applications and management options

The options you have for managing the applications are listed in the Commands column. The **Expert User** box enables access to commands for system components. We do not recommend that you stop and uninstall system components unless you understand how the system is affected (as in, Expert User).

2. Click the command you want to execute. The status of the command is displayed in the lower-left corner of the portlet.

10.4.2 Using the command scripts

You can use the **deploy** command to manage applications and modules.

Listing deployed modules

Use the following command to get a list of deployed modules.

```
deploy --user username --password password list-modules
```

Figure 10-8 shows an excerpt from the output of the command, which shows that the calculator sample application is installed.

```
C:\WebSphere\CE\bin>deploy --user system --password manager list-modules
Using GERONIMO_BASE:   C:\WebSphere\CE
Using GERONIMO_HOME:   C:\WebSphere\CE
Using GERONIMO_TMPDIR: var\temp
Using JRE_HOME:        C:\Java50\jre
Found 61 modules
...
...
+ org.apache.geronimo.samples/calculator-stateless-ear/2.0.0.1/ear
      `-> calculator-stateless-war.war
      `-> calculator-stateless-ejb.jar
...
...
C:\WebSphere\CE\bin>
```

Figure 10-8 List of installed modules

The configID of the module is the name listed, for example, the configID of the sample application is org.apache.geronimo.samples/calculator-stateless-ear/2.0.0.1/ear.

Modules that have the **+** sign in front of them are running. In this example, the calculator application is running.

Starting a module

To start a module, use the following command:

```
deploy --user username --password password start config_id
```

Figure 10-9 on page 144 shows an example of starting an application using the **deploy** command.

```
C:\WebSphere\CE\bin>deploy --user system --password manager start
org.apache.geronimo.samples/calculator-stateless-ear/2.0.0.1/ear
Using GERONIMO_BASE:   C:\WebSphere\CE
Using GERONIMO_HOME:   C:\WebSphere\CE
Using GERONIMO_TMPDIR: var\temp
Using JRE_HOME:        C:\Java50\jre

    Started
    org.apache.geronimo.samples/calculator-stateless-ear/2.0.0.1/ear
      `-> calculator-stateless-war.war @ /calculator-stateless
      `-> calculator-stateless-ejb.jar

    Started org.apache.geronimo.configs/axis2/2.0.1/car

C:\WebSphere\CE\bin>
```

Figure 10-9 Start an application using the deploy command

Restarting modules

To restart an application, use this command script, as shown in Figure 10-10:

```
deploy --user username --password password restart config_id
```

```
C:\WebSphere\CE\bin>deploy --user system --password manager restart
org.apache.geronimo.samples/calculator-stateless-ear/2.0.0.1/ear
Using GERONIMO_BASE:   C:\WebSphere\CE
Using GERONIMO_HOME:   C:\WebSphere\CE
Using GERONIMO_TMPDIR: var\temp
Using JRE_HOME:        C:\Java50\jre

    Restarted
    org.apache.geronimo.samples/calculator-stateless-ear/2.0.0.1/ear
      `-> calculator-stateless-war.war @ /calculator-stateless
      `-> calculator-stateless-ejb.jar

    Restarted org.apache.geronimo.configs/axis2/2.0.1/car
```

Figure 10-10 Restart an application using the deploy command

Stopping an application

To stop an application, use this command script, which Figure 10-11 on page 145 illustrates:

```
deploy --user username --password password stop config_id
```

```
C:\WebSphere\CE\bin>deploy --user system --password manager stop
org.apache.geronimo.samples/calculator-stateless-ear/2.0.0.1/ear
Using GERONIMO_BASE:   C:\WebSphere\CE
Using GERONIMO_HOME:   C:\WebSphere\CE
Using GERONIMO_TMPDIR: var\temp
Using JRE_HOME:        C:\Java50\jre

    Stopped
    org.apache.geronimo.samples/calculator-stateless-ear/2.0.0.1/ear
      `-> calculator-stateless-war.war
      `-> calculator-stateless-ejb.jar

    Stopped org.apache.geronimo.configs/axis2/2.0.1/car

C:\WebSphere\CE\bin>
```

Figure 10-11 Stop an application using the deploy command

For more information, see:

http://publib.boulder.ibm.com/wasce/V2.0.0/en/control-applications.html
http://cwiki.apache.org/GMOxDOC20/starting-and-stopping-application-modules.html

10.5 Using shared libraries

Applications and modules that are deployed on an application server can reference the Java libraries and framework that are included in a common shared repository, the *wasceHome*/repository directory.

The shared repository is a convenient feature because when you want a specific library to be available for different applications, you do not have to package the same JAR in each artifact; instead, you just define a dependency to that library, which eliminates redundancies, simplifies the packaging, and allows a centralized management of the libraries (bug fixing, upgrades, and so on).

To add a shared library to the application server:

1. In the console, select the **Common Libs** portlet to open the page shown in Figure 10-12 on page 146. This page shows you a list of current JARs in the repository.

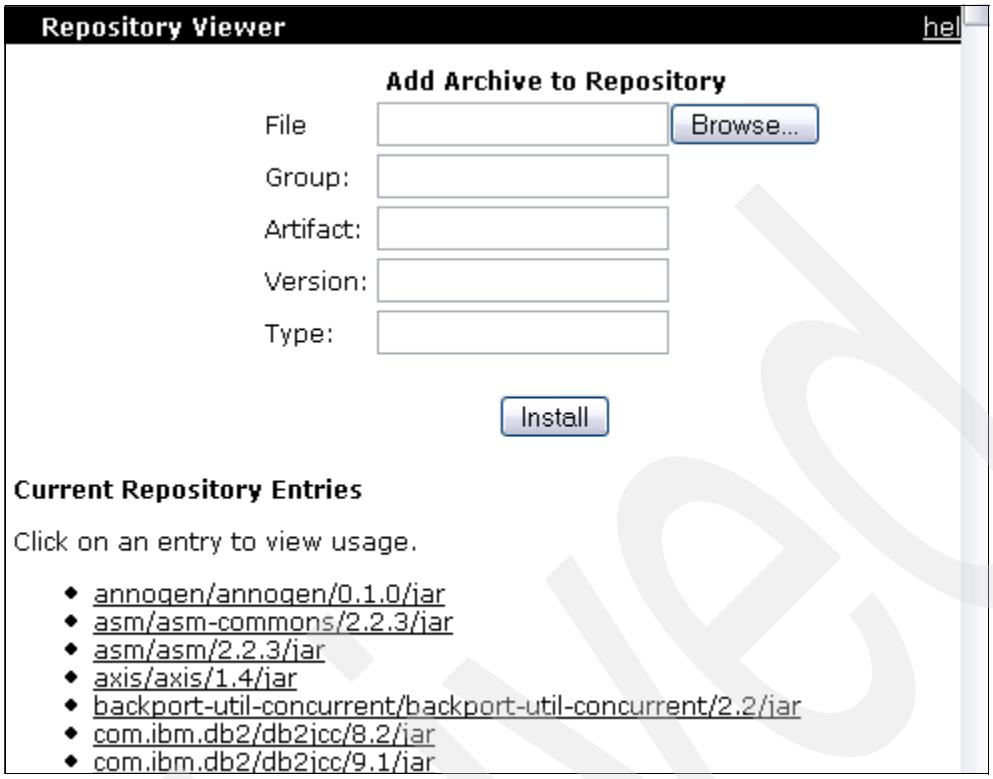


Figure 10-12 The Repository Viewer page

2. Click an entry to see information about how to add the JAR file as a dependency in an application or module, as shown in Figure 10-13.

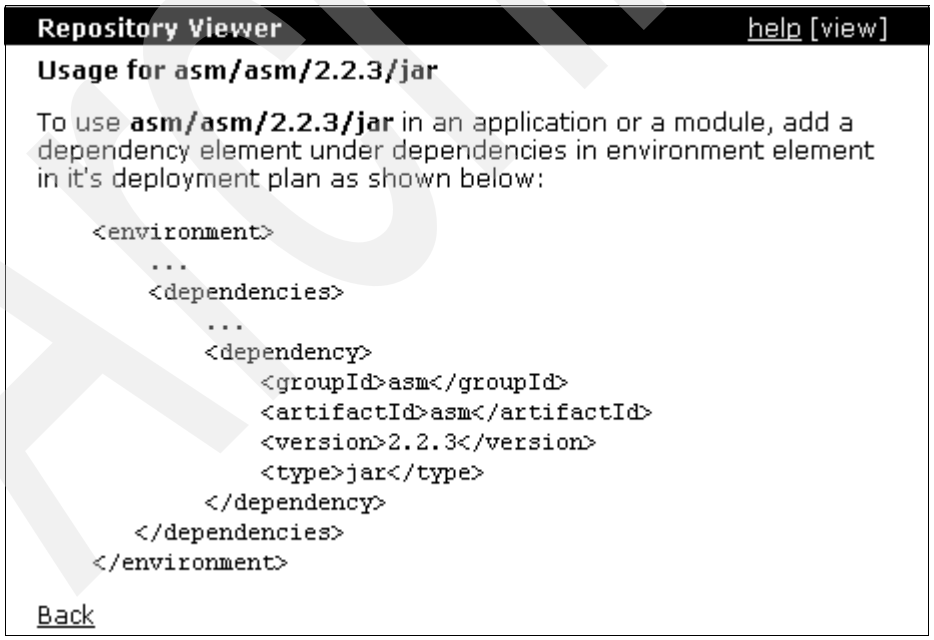


Figure 10-13 Entry usage

3. To add a new library, locate the main repository list, and click the **Browse** button (Figure 10-12), and find the JAR file you want to add.

4. Complete the fields, as shown in Figure 10-14:
   - Group: Identifies the name of the project or the directory tree that corresponds to Java package prefix, for example *com.ibm.itso.*
   - Artifact: The name of the artifact.
   - Version: The version number, for example *1.0.0.*
   - Type: Type of library, typically *jar.*

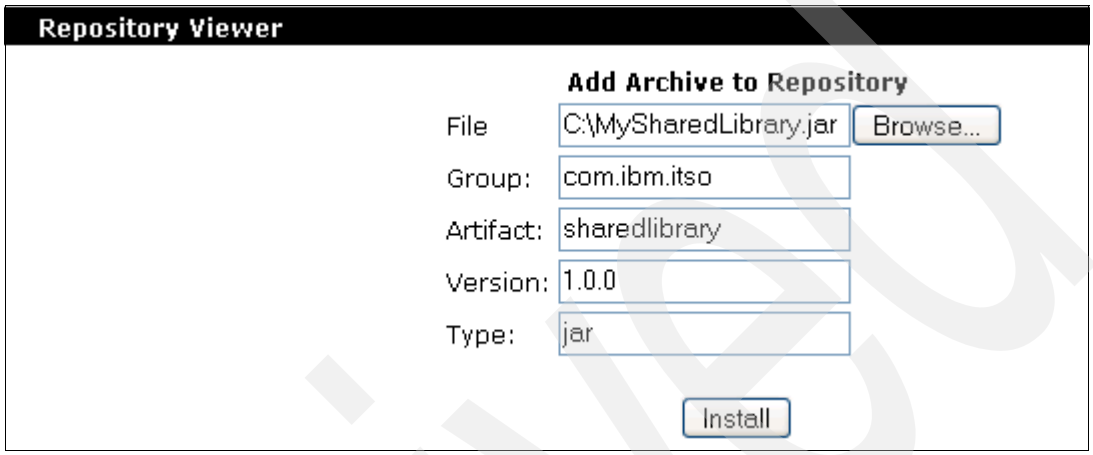


Figure 10-14 Shared library form

5. Click **Install** to install the library on the application server.
6. After the library is installed, you can check the library in the list of all installed shared libraries. You can also check the archive inside the repository directory (Figure 10-15).

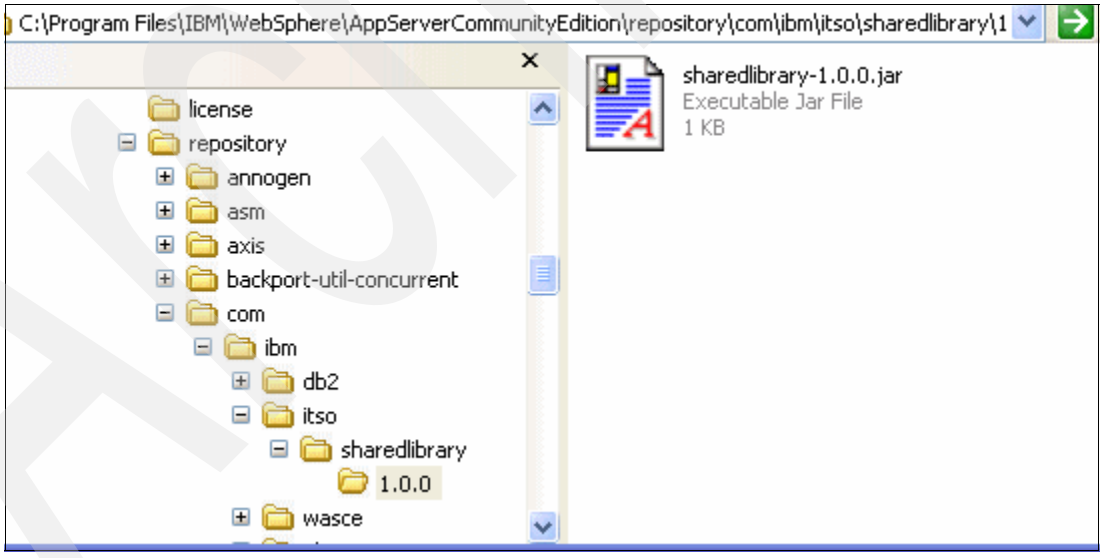


Figure 10-15 The library inside the repository directory

Note the convention adopted by Community Edition to store the library into the repository:

`group/artifact/version/artifact-version.type`

So the library in this example is installed as:

`com/ibm/itso/sharedlibrary/1.0.0/sharedlibrary-1.0.0.jar`

10.5.1 Using shared libraries

To use the library from your application, you must define a dependency in the corresponding deployment plan, as shown in Example 10-1.

Example 10-1 Depends from a Web application

```
<?xml version="1.0" encoding="UTF-8"?>

<web-app xmlns="http://geronimo.apache.org/xml/ns/j2ee/web-2.0"
  xmlns:dep="http://geronimo.apache.org/xml/ns/deployment-1.2"
  xmlns:naming="http://geronimo.apache.org/xml/ns/naming-1.2"
  xmlns:security="http://geronimo.apache.org/xml/ns/security-2.0">

<dep:environment>
   <dep:moduleId>
      <dep:groupId>com.ibm.itso</dep:groupId>
      <dep:artifactId>SimpleWebApp</dep:artifactId>
      <dep:version>1.0.0</dep:version>
      <dep:type>war</dep:type>
   </dep:moduleId>
   <dep:dependencies>
      <dep:dependency>
         <groupId>com.ibm.itso<groupId>
         <artifactId>sharedlibrary<artifactId>
         <version>1.0.0<version>
         <type>jar<type>
      </dependency>
   </dependencies>
   <dep:hidden-classes/>
   <dep:non-overridable-classes/>
</dep:environment>
```

In Example 10-1, the defined dependency (highlighted in bold) allows the previously deployed sharedlibrary jar to be included in the classpath of the SimpleWebApp application.

For more details about the shared repository, look at:

http://publib.boulder.ibm.com/wasce/V2.0.0/en/adding-java-libraries.html

10.5.2 JAX-WS tools

Community Edition provides the **jaxws-tools** command that you can use to generate the required artifacts for Web services deployment and invocation. You can invoke it when either the server is started or stopped.

The command is located in *wasceHome*/bin.

For more information about using jaxws-tools see:

http://publib.boulder.ibm.com/wasce/V2.0.0/en/using-jax-ws-tools.html

10.6 Community Edition-specific deployment descriptors

Using the Java EE specifications, you can provide configuration information outside of the standard application deployment descriptors, which gives the implementation flexibility when configuring the application, for example, the specification does not specify how to map an ejb-name to a home interface.

This additional configuration information is provided in XML documents that contain deployment information to bind application resources to the server resources. These documents are called *vendor specific deployment descriptors* or *deployment plans*.

In this section, we describe the Community Edition Java EE module deployment plans.

For more information about deployment plans see the following:

http://publib.boulder.ibm.com/wasce/V2.0.0/en/developing-deployment-plans.html
http://cwiki.apache.org/GMOxDOC12/deployment-plans.html

10.6.1 Common namespaces

The deployment plan XML documents in Community Edition use the schema and namespaces in Example 10-1 on page 148.

Table 10-1 Schema and namespaces that deployment plan XML documents in Community Edition use

File name	Schema namespace	Description
attributes-1.2.xsd	http://geronimo.apache.org/xml/ns/attributes-1.2	Definition for storing manageable attribute values.
geronimo-application-2.0.xsd	http://geronimo.apache.org/xml/ns/j2ee/application-2.0	Defines Geronimo enterprise application deployment plan.
geronimo-connector-1.2.xsd	http://geronimo.apache.org/xml/ns/j2ee/connector-1.2	Defines Geronimo resource adapter deployment plan.
geronimo-application-client-1.2.xsd	http://geronimo.apache.org/xml/ns/j2ee/application-client-2.0	Defines Geronimo Java application client deployment plan.
geronimo-login-config-2.0.xsd	http://geronimo.apache.org/xml/ns/loginconfig-2.0	Defines login module configuration to use a particular security module.
geronimo-module-1.2.xsd	http://geronimo.apache.org/xml/ns/deployment-1.2	Defines Geronimo service deployment plan.
geronimo-naming-1.2.xsd	http://geronimo.apache.org/xml/ns/naming-1.2	Defines common naming elements for resolving Java assets.
geronimo-security-2.0.xsd	http://geronimo.apache.org/xml/ns/security-2.0	Defines Web and EJB access security configurations.
geronimo-web-2.0.xsd	http://geronimo.apache.org/xml/ns/j2ee/web-2.0	Defines Web application deployment plan.
openejb-jar-2.1.xsd	http://www.openejb.org/xml/ns/openejb-jar-2.1	Defines the EJB deployment plan.

10.6.2 Module identity

Deployment plans contain an environment configuration that uniquely identifies this module, define its dependencies, and specifies class loader configurations. The identity of a module is defined in the deployment namespace. The module identity elements are:

- <moduleId>

  Uniquely identifies this module in the server. It is composed of the following elements:

  - GroupId: Defines a group of modules. Modules can be grouped if they are related, which might be the project name, company name, and so on.
  - ArtifactId: Identifies a specific module in a group. This must be unique in a group. If no artifactID is defined, the default value is the Java archive file name.
  - Version: Defines the version number of this deployment.
  - Type: the type of this module. The possible values are CAR, if it is a system module, or the Java archive type (EAR, WAR, JAR).

- <dependencies>

  The dependencies element specifies other modules (that exists in the server's repository) that your application depends on. Using this element, the server manages to control the classpath of the application.

  For more information about the dependencies configuration see:

  http://publib.boulder.ibm.com/wasce/V2.0.0/en/managing-the-classpath.html

- <hidden-classes>

  This element lists packages or classes that, if present in a parent class loader, should not be exposed to the application. This element is useful if you want to use a specific library version and avoid using a shared lib to override it.

- <non-overridable-classes>

  Does the opposite of the <hidden-classes> element. If a class or package listed in here exists in a parent class loader, the parent class should be used.

- <inverse-class-loading>

  If this element is present, the module class loader is always checked for a class first before checking the parent class loader.

Example 10-2 shows an example of the <environment> element.

Example 10-2 Environment element

```
<environment>
    <moduleId>
      <groupId>MyProjectName</groupId>
      <artifactId>MySimpleWebModule</artifactId>
      <version>2.0</version>
      <type>war</type>
    </moduleId>
    <dependencies>
    <dependency>
       <groupId>MyProjectName</groupId>
       <artifactId>MyBankEjbModule</artifactId>
       <type>jar</type>
    </dependency>
    </dependencies>
```

```
    <hidden-classes>
        org.apache.commons.logging,javax.servlet
    </hidden-classes>
    <non-overridable-classes>
        javax.ejb,javax.activation
    </non-overridable-classes>
</environment>
```

10.6.3 Security configuration

The Java EE specification defines that a standard Web module deployment can have defined security roles and those are mapped to groups in the security realm. The security configuration in the deployment plan has two objectives, one is to select the realm that handles the authentication and the other is to map the application user roles to its principals. Example 10-3 shows the development plan's role mapping.

Example 10-3 Deployment plan role mapping

```
<security-realm-name>geronimo-admin</security-realm-name>

<security>
   <role-mapping>
      <role role-name="administrator">
         <principal name="admin" class="com.ldap.Group"/>
      </role>
      <role role-name="webmaster">
         <principal name="Rafael" class="com.ldap.User"/>
      </role>
   </role-mapping>
</security>
```

The <security-realm-name> entry specifies the realm that the application uses to authenticate users. This realm must match the name that is specified for the security realm in the Community Edition configuration. Select the **Security Realms** portlet (Figure 10-16) to display, in the administrative console, the security realms that are available in the server.

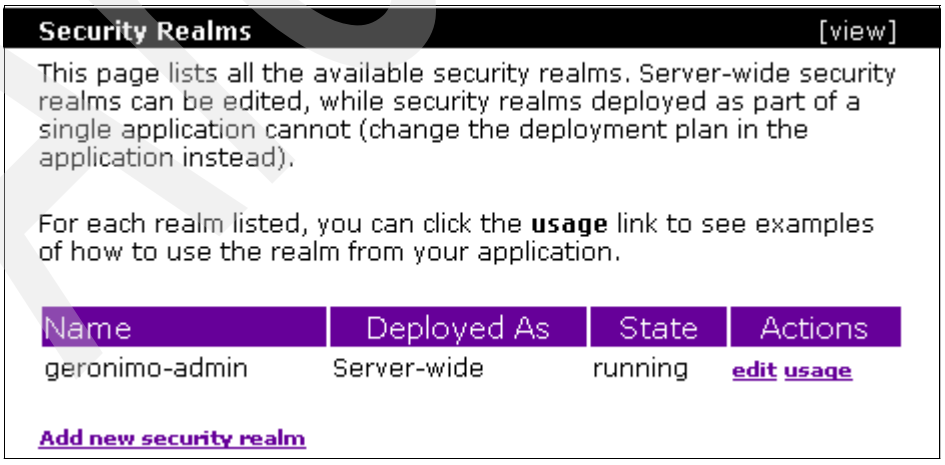


Figure 10-16 Security realms list in administrative console

The <principal> entries identify the class used and the principal name. Depending on the class, it identifies a group or a single user.

We discuss security administration in 5.5, “Configuring application security” on page 58. For more information about security group name and mapping, see:

http://publib.boulder.ibm.com/wasce/V2.0.0/en/security-group-and-name-mapping.html

In the Community Edition sample application package, you can look at the following applications for examples of security configuration:

- file-realm-demo: A Web application that shows how to use a file as a realm.
- ldap-realm-demo: A Web application that shows how to use an LDAP as a realm.

10.6.4 Binding common resources

In this section, we describe how to bind a resource that is configured on the server with its resource reference in the application. The binding elements are defined by the namespace of the schema geronimo-naming-1.2.xsd.

Binding data sources

To bind a data source reference in an application to a data source that is configured in the Community Edition server, you must use the <resource-ref> element. This element contains the name that is used in the reference and the name it is defined as on the server. Example 10-4 provides an example.

Example 10-4 Data source binding in the geronimo-web.xml deployment plan file

```
<resource-ref>
   <ref-name>jdbc/MyAppNameOfTheDatasource</ref-name>
   <resource-link>DatasourceNameInServer</resorce-link>
</resource-ref>
```

To use it in a Web application Example 10-5 shows what is needed in the standard Java EE module deployment descriptor.

Example 10-5 Data source reference name exposed in the web.xml deployment descriptor file

```
<resource-ref>
   <res-ref-name>jdbc/MyAppNameOfTheDatasource</res-ref-name>
   <res-type>javax.sql.DataSource</res-type>
   <res-auth>Container</res-auth>
   <res-sharing-scope>Sharable</res-sharing-scope>
</resource-ref>
```

As an example, you can look at the dbdemo1 sample application in the \src\main\webapp\WEB-INF folder.

Binding EJBs

There are two distinct situations for EJB binding in Community Edition. One is when the EJB is located in the same application as the Web module. In this case, the binding is done by using the <ejb-link> element in the deployment plan. When the EJB is not in the same application where the Web module is packaged, the <ejb-ref> and <ejb-local-ref> are also needed, and the Web deployment plan must list the dependency of the EJB module.

EJB reference in the same application

This section contains examples of how to reference an EJB that is deployed in the same enterprise application as the Web module. Example 10-6 on page 153 shows the web.xml file with the EJB reference tags.

Example 10-6 EJB reference in web.xml

```
<ejb-ref>
   <ejb-ref-name>MySessionEJB</ejb-ref-name>
   <ejb-ref-type>Session</ejb-ref-type>
   <home>com.ibm.samples.TestSessionHome</home>
   <remote>com.ibm.samples.TestSession</remote>
</ejb-ref>
```

The reference name used was MySessionEJB, and it must be linked with the deployment plan configuration shown in Example 10-7.

Example 10-7 EJB binding in geronimo-web.xml

```
<ejb-ref>
   <ref-name>MySessionEJB</ref-name>
   <ejb-link>RealSessionEJBName</ejb-link>
</ejb-ref>
```

EJB reference in different application

If the referenced EJB is in a different application, the EJB module must be listed in the dependencies of the other module (another EJB module or a Web module). One consequence of mapping the EJB module to another module is that the EJB interface classes are automatically added to that module's classpath, which eliminates the need to create the EJB client jar files manually.

The deployment descriptor used is shown in Example 10-8.

Example 10-8 EJB reference in the standard deployment descriptor

```
<ejb-ref>
   <ejb-ref-name>MySessionEJB</ejb-ref-name>
   <ejb-ref-type>Session</ejb-ref-type>
   <home>com.ibm.samples.TestSessionHome</home>
   <remote>com.ibm.samples.TestSession</remote>
</ejb-ref>
```

The deployment plan must have the dependency and all of the binding elements configured, as shown in Example 10-9.

Example 10-9 EJB binding in deployment plan

```
...
<environment>
   <moduleId>
      ...
   </moduleId>
   <dependencies>
      <dependency>
         <groupId>MyCompanyGroup</groupId>
         <artifactId>MyReferencedEJBModule</artifactId>
         <type>jar</type>
      </dependency>
   </dependencies>
</environment>
<ejb-ref>
   <ref-name>MySessionEJB</ref-name>
```

```
   <pattern>
      <groupId>MyCompanyGroup</groupId>
      <artifactId>MyReferencedEJBModule</artifactId>
      <name>RealSessionEJBName</name>
   </pattern>
</ejb-ref>
```

To access the EJB from a Java class, use the code shown in Example 10-10.

Example 10-10 Traditional JNDI lookup

```
public class TestServlet extends HttpServlet{
public void init(){
   try{
      InitialContext ctx = new InitialContext();
      TestSessionHome home =
                  (TestSessionHome) ctx.lookup("java:comp/env/MySessionEJB");
      home.create();
   }catch(Exception e){
      ...
   }
}
}
```

Example 10-11 shows a reference between two EJB modules using annotations.

Example 10-11 EJB 3.0 annotated reference

```
@Stateless(name="AppService")
public class AppService {
   @EJB(name="controller")
   private ControllerRemote control = null;
...
```

In the EJB module deployment plan (ejb-jar.xml), we bind the reference to an existing EJB, as shown in Example 10-12.

Example 10-12 EJB module deployment plan with reference binding

```
<openejb-jar xmlns="http://www.openejb.org/xml/ns/openejb-jar-2.1"
             xmlns:dep="http://geronimo.apache.org/xml/ns/deployment-1.2"
             xmlns:naming="http://geronimo.apache.org/xml/ns/naming-1.2">
<dep:environment>
   <dep:moduleId>
      ...
   </dep:moduleId>
   <dep:dependencies>
      <dep:dependency>
         <dep:groupId>ControllerAppGroup</dep:groupId>
         <dep:artifactId>ControllerEJBModule</dep:artifactId>
         <dep:type>jar</dep:type>
      </dep:dependency>
   </dep:dependencies>
</dep:environmen>

<enterprise-beans>
```

```
   <session>
      <ejb-name>AppService</ejb-name>
      <ejb-ref>
         <ref-name>controller</ref-name>
         <naming:pattern>
            <naming:artifactId>ControllerEJBModule</naming:artifactId>
            <naming:name>ControllerImplementation</naming:name>
         </naming:pattern>
      </ejb-ref>

   </session>
</openejb-jar>
```

Note that the dependencies were set because the application needs to import the called EJB module.

For more details about how to map annotated EJBs, see:

http://cwiki.apache.org/GMOxDOC20/jar-to-jar-ejb-references-no-ear.html

The calculator-stateless-pojo sample application can be used as an example.

Binding JMS resources

There are two bindings that can be done between Java EE modules and JMS resources: the queue connection factory and the queue or topic. The web.xml configuration in Example 10-13 shows these binding types.

Example 10-13 JMS references in web.xml

```
<web-app xmlns="http://java.sun.com/xml/ns/j2ee">
   ...
   <resource-ref>
      <res-ref-name>jms/CommonConnectionFactory</res-ref-name>
      <res-type>javax.jms.QueueConnectionFactory</res-type>
      <res-auth>Container</res-auth>
      <res-sharing-scope>Shareable</res-sharing-scope>
   </resource-ref>

   <message-destination-ref>
      <message-destination-ref-name>jms/OrderQueue</message-destination-ref-name>
      <message-destination-type>javax.jms.Queue</message-destination-type>
      <message-destination-usage>Produces</message-destination-usage>
      <message-destination-link>OrderQueue</message-destination-link>
   </message-destination-ref>

</web-app>
```

The deployment plan must bind the references and configure the dependencies for the queue factory Java EE connector and the Community Edition active-mq broker module, as shown in Example 10-14.

Example 10-14 JMS binding in module deployment plan

```
   <environment>
      <moduleId>
         ...
      </moduleId>
```

```
    <dependencies>
       <dependency>
          <groupId>org.apache.geronimo.config</groupId>
          <artifactId>activemq-broker</artifactId>
          <type>car</type>
       </dependency>
       <dependency>
          <groupId>org.apache.geronimo.samples</groupId>
          <artifactId>jms-resources</artifactId>
       </dependency>
    </dependencies>
 </environment>

 <resource-ref>
    <ref-name>jms/CommonConnectionFactory</ref-name>
    <resource-link>SampleConnectionFactory</resource-link>
 </resource-ref>

 <resource-env-ref>
    <ref-name>jms/OrderQueue</ref-name>
    <admin-object-link>RealOrderQueueName</admin-object-link>
 </resource-env-ref>
```

This deployment plan is the same for any type of Java EE module. The mdb sample application can be used as an example.

More information about configuring JMS resources is in Chapter 7, “Messaging” on page 75.

GBeans binding

If the Java module needs to reference Community Edition specific services, and not a Java EE standard, it needs to create references to the server GBeans. The QuartzScheduler GBean is an example of specific services that Java modules can reference.

A Community Edition specific resource is only configured in the deployment plan, but it is accessed using the default application private JNDI space (i.e. java:comp/env/). Example 10-15 shows the Java asset deployment plan with a GBean reference.

Example 10-15 Java asset deployment plan with a GBean reference

```
...
 <gbean-ref>
    <ref-name>gbeans/ServerInfo</ref-name>
    <ref-type>
       org.apache.geronimo.system.serverinfo.ServerInfo
    </ref-type>
    <pattern>
       <name>ServerInfo</name>
    </pattern>
 </gbean-ref>
```

Example 10-15 binds the JNDI name gbeans/ServerInfo to the GBean that contains the current server information. Example 10-16 on page 157 is a Java code snippet that shows how to access it.

Example 10-16 Java code snippet of how use a GBean reference

```
...
Context ctx = new InitialContext();
ServerInfo serverInfo =
           (ServerInfo) ctx.lookup("java:comp/env/gbeans/ServerInfo");
System.out.print("Server Version:");
System.out.print(serverInfo.getVersion());
...
```

See the following Web site for more references about GBean binding:

http://publib.boulder.ibm.com/wasce/V2.0.0/en/configuring-resources-in-the-asset-scope.html

10.6.5 Web module deployment plan

The Community Edition Web module deployment plan is geronimo-web.xml. The schema used to define this XML is geronimo-web-2.0.xsd and is in *wasceHome*/schema/.

A typical geronimo-web.xml file imports naming, security, Web, and deployment namespaces with the attributes in Example 10-17.

Example 10-17 Typical namespace of a Web deployment plan.

```
<web-app xmlns="http://geronimo.apache.org/xml/ns/j2ee/web-2.0"
      xmlns:sec="http://geronimo.apache.org/xml/ns/security-2.0"
      xmlns:dep="http://geronimo.apache.org/xml/ns/deployment-1.2"
      xmlns:nam="http://geronimo.apache.org/xml/ns/naming-1.2">
...
</web-app>
```

This deployment plan can be packaged into the Java Web archive in the WEB-INF folder.

The following sections describe the elements for geronimo-web.xml deployment plan.

Specific Community Edition Web deployment elements

In this subsection, we describe the specific deployment elements for Community Edition. The namespace for these elements are defined in the geronimo-web-2.0.xsd schema.

Context root element

The <context-root> element defines the Web application context root, as shown in Example 10-18. It is the path name to be used in the URL to access this Web module.

Example 10-18 Context root example

```
<context-root>/mywebapp</context-root>
```

Host element

The deployment plan can specify which virtual host the Web application is to be deployed to using the <host> element, as shown in Example 10-19 on page 158. This element is defined in the Web namespace inside the <container-config> element.

Example 10-19 Virtual host element.

```
<container-config>
  <tomcat>
   <host>TomcatCluster</host>
  </tomcat>
</container-config>
```

Binding Web containers

The <web-container> element identifies which Web container the application is deployed to, as shown in Example 10-20. This element is in the naming namespace.

Example 10-20 The Web container element.

```
<web-container>
   <!-- the name of a configured web container-->
   <gbean-link>TomcatWebContainer2</gbean-link>
</web-container>
```

Configuring Web container specifics

You can do some Web container configuration in the Web deployment plan, for example, to identify the cluster this Web application is deployed to. Example 10-21 shows the clustering configuration in geronimo-web.xml.

Example 10-21 Clustering configuration in geronimo-web.xml

```
<container-config>
   <tomcat>
      <cluster>TomcatClusterName</cluster>
   </tomcat>
</container-config>
```

Web security configuration

We discuss security in 10.6.3, “Security configuration” on page 151. Example 10-22 shows a standard deployment descriptor that can be used in the deployment plan that we described in that section.

Example 10-22 Web.xml security declaration.

```
<web-app xmlns="http://java.sun.com/xml/ns/j2ee">
...
<security-constraint>
   <web-resource-collection>
      <web-resource-name>Admin Role</web-resource-name>
      <url-pattern>/admin/*</url-pattern>
   </web-resource-collection>
   <auth-constraint>
      <role-name>administrator</role-name>
   </auth-constraint>
</security-constraint>

<security-constraint>
   <web-resource-collection>
      <web-resource-name>Webmaster Role</web-resource-name>
      <url-pattern>/protected/*</url-pattern>
```

```
   </web-resource-collection>
   <auth-constraint>
      <role-name>webmaster</role-name>
   </auth-constraint>
</security-constraint>

<login-config>
   <auth-method>FORM</auth-method>
   <realm-name>MyAppRealm</realm-name>
   <form-login-config>
      <form-login-page>/logon.html</form-login-page>
      <form-error-page>/logonError.html</form-error-page>
   </form-login-config>
</login-config>

<security-role>
   <role-name>administrator</role-name>
</security-role>
<security-role>
   <role-name>webmaster</role-name>
</security-role>
...
</web-app>
```

The Web deployment descriptor protects the resources in the /protected/ path, and allows only the user with the Webmaster role to access it. So, in the deployment plan that we describe in the 10.6.3, “Security configuration” on page 151, there is a mapping of that role with their principal.

Another way to specify the roles needed is using annotations.

Example 10-23 Role specification using annotations

```
@DeclareRoles(“webmaster“)
public class ConfigAppServlet extends HttpServlet{
...
}
```

For more information about Web archive deployment plans see:

http://publib.boulder.ibm.com/wasce/V2.0.0/en/war.html

10.6.6 Enterprise module deployment plan

The Community Edition enterprise module deployment plan is geronimo-application.xml. The schema that is used to define this XML is geronimo-application-2.0.xsd and is in *wasceHome*/schema/.

A typical geronimo-application.xml file imports naming, security, Web, and deployment namespaces with the attributes in Example 10-24.

Example 10-24 Typical namespace of an application deployment plan.

```
<application xmlns="http://geronimo.apache.org/xml/ns/j2ee/application-2.0"
      xmlns:sec="http://geronimo.apache.org/xml/ns/security-2.0"
      xmlns:dep="http://geronimo.apache.org/xml/ns/deployment-1.2"
```

```
      xmlns:nam="http://geronimo.apache.org/xml/ns/naming-1.2">
...
</application>
```

This deployment plan can be packaged into the Java enterprise archive into the folder META-INF.

In the following sections, we describe the elements for geronimo-application.xml deployment plan.

Specific Community Edition enterprise deployment elements

In this section, we describe the specific deployment elements for Community Edition. The namespace for these elements are defined in the geronimo-application-2.0.xsd schema.

Module element

The <module> element maps the modules that are packaged in this enterprise archive. There is one module configuration for each Java EE asset type, and each has the mapping to the module archive file and the path to their deployment plan. Example 10-25 provides an example.

Example 10-25 Module examples in geronimo-application.xml

```
<!-- Web module -->
<module>
   <web>warFile</web>
   <alt-dd>deployment_plan_path</alt-dd>
</module>

<!-- EJB module -->
<module>
   <ejb>ejb-jarFile</ejb>
   <alt-dd>deployment_plan_path</alt-dd>
</module>

<!-- Connector module -->
<module>
   <connector>rarFile</connector>
   <alt-dd>deployment_plan_path</alt-dd>
</module>

<!-- Client module -->
<module>
   <java>jarFile</java>
   <alt-dd>deployment_plan_path</alt-dd>
</module>
```

Optionally, you can replace the *alt-dd* tag by the actual deployment plan directly in the enterprise deployment plan, as shown in Example 10-26.

Example 10-26 Including the module deployment plan directly in the geronimo-application.xml

```
<module>
   <web>warFile</web>
   <web-app xmlns=”xmlns="http://geronimo.apache.org/xml/ns/j2ee/web-2.0”>
```

```
      <!-- all the web module deployment descriptor can be embedded here -->
   </web-app>
</module>
```

External module element

Using the <ext-module> element you can include other modules that are not defined in the application.xml deployment descriptor. The syntax is similar to the <module> element. The internal path is the URI to the standard repository compound by *groupId/artifactId/version/type* (e.g. tranql/tranql-connector/1.1/rar). Example 10-27 shows the external module element for geronimo-application.xml.

Example 10-27 External module element for geronimo-application.xml

```
<!-- External Web module -->
<ext-module>
   <web>moduleId</web>
   <external-path>URI_of_the_module_repository</external_path>
<!-- if the module does not have an internal deployment plan it can be added
here-->
</ext-module>

<!-- External EJB module -->
<ext-module>
   <ejb>moduleId</ejb>
   <external-path>URI_of_the_module_repository</external_path>
<!-- if the module does not have an internal deployment plan it can be added
here-->
</ext-module>

<!-- External Connector module -->
<ext-module>
   <connector>moduleId</connector>
   <external-path>URI_of_the_module_repository</external_path>
<!-- if the module does not have an internal deployment plan it can be added
here-->
</ext-module>

<!-- External Client module -->
<ext-module>
   <java>moduleId</java>
   <external-path>URI_of_the_module_repository</external_path>
<!-- if the module does not have an internal deployment plan it can be added
here-->
</ext-module>
```

You can replace the <external-path> with <internal-path>, and point to the Java EE archive file, if it is included in the EAR.

Security role element

Using the <security-role> element you can configure application security. Example 10-28 on page 162 shows an application.xml file that would be used with the security role mapping that we defined in 10.6.3, “Security configuration” on page 151.

Example 10-28 Application.xml declaration of security roles

```
<application id="Application_ID" version="5"
              xmlns="http://java.sun.com/xml/ns/javaee" >
...
<security-role id=”sec01”>
   <role-name>administrator</role-name>
   <description>The application wide role created for administrators</description>
</security-role>

<security-role id=”sec01”>
   <role-name>webmaster</role-name>
   <description>The application wide role created for webmaster</description>
</security-role>
</application>
```

For more information see:

http://publib.boulder.ibm.com/wasce/V2.0.0/en/ear.html

10.6.7 EJB module deployment plan

The Community Edition EJB module deployment plan is openejb-jar.xml. The schema used to define this XML is geronimo-application-2.0.xsd and is in *wasceHome*/schema/.

A typical geronimo-ejb.xml file imports naming, security, Web, and deployment namespaces with the attributes in Example 10-29.

Example 10-29 Typical namespace of an application deployment plan

```
<openejb-jar xmlns="http://www.openejb.org/xml/ns/openejb-jar-2.1">
...
</openejb-jar>
```

This deployment plan can be packaged into the Java EJB archive into the folder META-INF.

In the following sections, we describe the elements for openejb-jar.xml deployment plan.

Container-managed persistence configurations

To configure container-managed persistence (CMP), set the elements in the following sections.

Database pool

To configure a database pool to be used by the CMP, you must set the tag <cmp-connection-factory>, as shown in Example 10-30, where databasePool is replaced by the database connection pool name.

Example 10-30 Setting a connection pool to be used by the container

```
<cmp-connection-factory>
   <resource-link>databasePool</resource-link>
</cmp-connection-factory>
```

EJB query language

The <ejb-ql-compiler-factory> element defines the compiler class that compiles the EJB query language. It only needs to be specified if EJBQL is being used. Example 10-31 shows how to set the EJB QL compiler class.

Example 10-31 Setting the EJB QL compiler class

```
<ejb-ql-compiler-factory>
   compilerClassName
</ejb-ql-compiler-factory>
```

Database factory

To define a customized SQL statement processor, you can use the <db-syntax-factory> element, as shown in Example 10-32.

Example 10-32 Setting a class to process specific SQL statements

```
<db-syntax-factory>
   className
</db-syntax-factory>
```

Foreign key constraint

If the container must order the insert, update, and delete statements to be compliant with a defined foreign key constraint, you must include the <enforce-foreign-key-constraint/> element in the deployment plan.

Enterprise beans configuration

In this section, we describe the configurations for the three types of enterprise beans: session beans, entity beans, and message driven beans. All of them must be in the <enterprise-beans> element.

Session beans

The <session> element is used to describe session beans, with:

- <ejb-name>: name that the EJB uses. It is specified in the ejb-jar.xml file.
- <jndi-name>: the JNDI name that is used to access the remote interface of the EJB.
- <local-jndi-name>: the JNDI name to access the local interface of the EJB.

Example 10-33 shows a sample of a session bean configuration.

Example 10-33 Sample of a session bean configuration

```
...
<enterprise-beans>
   <session>
      <ejb-name>TestSession</ejb-name>
      <jndi-name>ejb/TestSession</jndi-name>
      <local-jndi-name>ejb/TestSessionLocal</local-jndi-name>
   </session>
</enterprise-beans>
...
```

Entity beans

The <entity> element describes the entity beans, with:

- <ejb-name>: Name that the EJB uses. It is specified in the ejb-jar.xml file.
- <jndi-name>: The JNDI name used to access the remote interface of the EJB.
- <local-jndi-name>: The JNDI name to access the local interface of the EJB.
- <table-name>: The table to which the entity is mapped.
- <cmp-field-mapping>: This field maps each field to a column. Set the field name in <cmp-field-name> and the table column in the <table-column> element. The <cmp-field-mapping> element can be repeated as many times as needed.

Example 10-34 shows an entity bean description.

Example 10-34 Entity bean description

```
...
<enterprise-beans>
   <entity>
      <ejb-name>Bill</ejb-name>
      <jndi-name>ejb/Bill</jndi-name>
      <local-jndi-name>ejb/BillLocal</local-jndi-name>
      <table-name>TelephoneBill</table-name>

      <cmp-field-mapping>
         <cmp-field-name>amount</cmp-field-name>
         <table-column>total_amount</table-column>
      </cmp-field-mapping>

      <cmp-field-mapping>
         <cmp-field-name>status</cmp-field-name>
         <table-column>bill_status</table-column>
      </cmp-field-mapping>

   </entity>
</enterprise-beans>
...
```

Message driven beans

Message driven beans (MDB) are configured using the <message-driven> element in the <enterprise-beans> tag. The elements of an MDB configuration are:

- <ejb-name>: Name that the EJB uses. It is specified in the ejb-jar.xml file.
- <resource-adapter>: Sets the Java Message Service (JMS) connection pool.
- <activation-config>: Sets the parameters that are used to select a specific queue or topic where the bean receives its messages. Its value is vendor specific.

Example 10-35 shows a message driven bean configuration.

Example 10-35 Message driven bean configuration

```
...
<enterprise-beans>
...
   <message-driven>
      <ejb-name>MyListener</ejb-name>
```

```
      <resource-adapter>
         <resource-link>ActiveMQ</resource-link>
      </resource-adapter>
      <activation-config>
         <activation-config-property>
            <activation-config-property-name>
                  destinationType
            </activation-config-property-name>
            <activation-config-property-value>
                  javax.jms.Topic
            </activation-config-property-value>
         </activation-config-property>
         <activation-config-property>
            <activation-config-property-name>
                  destination
            </activation-config-property-name>
            <activation-config-property-value>
                     MyEventTopic
            </activation-config-property-value>
         </activation-config-property>
      </activation-config>
   </message-driven>
</enterprise-bean>
```

Use the daytrader sample application as an example of how to reference MDBs.

EJB relationships

Community Edition is a Java EE 5 certified server, and as such, you can configure EJB relationships, which you can do using the <relationships> element. Within that element there is one <ejb-relation> element for each mapped EJB relationship.

The relationship configuration is initially configured in the standard Java EE deployment descriptor; however, specific database elements are defined in the deployment plan. The sub-elements of <ejb-relation> element are:

- <ejb-relationship-role-name>: Used for XML documentation purposes. It is not used by the EJB container.
- <relationship-role-source>: Specifies the EJB that is associated with this relationship role. The <cmr-field> element uniquely identifies the relationship role to which these configurations are related.
- <ejb-name>: The name of the EJB that forms this relationship role. This must match with the <relationship-role-source>/<ejb-name> from the ejb-jar.xml.
- <cmr-field>: If the ejb-jar.xml relationship configuration defines a cmr-field, the same must be done in this deployment plan.
- <cmr-field-name>: The name of the <cmr-field> on the EJB. This must match the same configuration in the ejb-jar.xml.

- <foreign-key-column-on-source>: If it is an 1x1 or 1xn relationship, the foreign key can be included in the source table. If so, this attribute is needed to indicate it.
- <cmr-field-mapping>: Holds a mapping between the primary key and the foreign key:
  - <key-column>: The column in the source EJB table that contains the key to the other EJB.
  - <foreign-key-column>: The column in the destination EJB's table that contains the key.

Example 10-36 maps the Person EJB to its Address EJB. In the database, the Address table has a foreign key to Person table.

Example 10-36 Mapping relationship between EJBs

```
<relationships>
   <ejb-relation>
   <ejb-relation-name>Person-Address</ejb-relation-name>
   <ejb-relationship-role>
      <ejb-relationship-role-name>Address-to-Person</ejb-relationship-role-name>
         <relationship-role-source>
            <ejb-name>Address</ejb-name>
         </relationship-role-source>
         <cmr-field>
            <cmr-field-name>person</cmr-field-name>
         </cmr-field>
         <foreign-key-column-on-source/>
         <role-mapping>
            <cmr-field-mapping>
               <key-column>PERSON_ID</key-column>
               <foreign-key-column>PERSON_FK</foreign-key-column>
            </cmr-field-mapping>
         </role-mapping>
   </ejb-relationship-role>
   </ejb-relation>
</relationships>
```

For more references about the EJB deployment plan, see:

http://publib.boulder.ibm.com/wasce/V2.0.0/en/ejb.html
http://cwiki.apache.org/GMOxDOC12/openejb-jarxml.html

10.6.8 Resource adapter deployment plan

The Community Edition connector resource adapter module deployment plan is geronimo-ra.xml. The Java Connector Architecture version that is supported is 1.5. The schema that we used to define this XML is geronimo-connector-1.2.xsd, and is located in *wasceHome*/schema/.

Example 10-37 shows what a typical geronimo-ra.xml file looks like.

Example 10-37 Typical namespace of a connector deployment plan.

```
<connector xmlns="http://geronimo.apache.org/xml/ns/j2ee/connector-1.1"
        version="1.5" >
...
```

```
</connector>
```

You can package this deployment plan into the Java resource adapter archive into the folder META-INF.

In the following sections, we describe the elements for the geronimo-ra.xml deployment plan, whose main element is <resource-adapter>.

Inbound resource adapter configuration

To receive asynchronous messages from external systems, the <resourceadapter-instance> element is required. It defines the name, the configuration properties, and a work manager.

- <resourceadapter-instance>: Contains the other elements.
- <resourceadapter-name>: Defines a name for the inbound resource adapter.
- <config-property-setting>: Adds vender-specific properties. Its attribute name specifies the property name, and its content is the configuration value.
- <workmanager>: Allows you to specify the work manager in the initial server configuration. The work manager is defined using a GBean reference.

Example 10-38 shows an inbound resource adapter configuration.

Example 10-38 Inbound resource adapter configuration

```
<connector xmlns="http://geronimo.apache.org/xml/ns/j2ee/connector-1.1"
        version=”1.5” >
...
<resource-adapter>
   <resourceadapter-instance>
      <resourceadapter-name>MyInboundResourceAdapter</resourceadapter-name>
      <!-- for each configuration property repeat the next tag -->
      <config-property-setting name=”password”>
         mysecretpassword
      </config-property-setting name>
      <workmanager>
         <gbean-link>DefaultWorkManager</gbean-link>
      </workmanager>
   </resourceadapter-instance>
</resource-adapter>
...
</connector>
```

Outbound resource adapter configuration

If the adapter sends messages to an external system and receives synchronous replies, an outbound resource adapter configuration is needed.

The following elements are the most common configuration properties:

- <connection-definition>
  - <connectionfactory-interface>: The class name of the connection factory’s interface that is defined in the resource adapter ra.xml.
  - <connectiondefinition-instance>: This element groups the following connection elements:
    - <name>: The name of this configuration.

- <implemented-interface>: An additional interface class name that is exposed by the connection factory.
- <config-property-setting>: Specific properties are configured using this element.

– <connectionmanager>: Contains the connection manager configuration:
- <connection-managed-security>: If this element is present, the container provides the authentication information; otherwise, the adapter handles the authentication.

– Transaction Settings
- <no-transaction>: If this tag is present, the adapter does not support transactions.
- <xa-transaction>: If this tag is present, the adapter will support XA transactions. This element might contain other configurations, such as <transaction-caching> and <transaction-log>.

For a complete reference of the resource adapter configuration parameters, visit:

http://cwiki.apache.org/GMOxDOC11/geronimo-raxml.html

Example 10-39 is an example of an outbound resource adapter deployment plan.

Example 10-39 Outbound resource adapter deployment plan

```
<connector xmlns="http://geronimo.apache.org/xml/ns/j2ee/connector-1.1"
        version="1.5" >
...
<resource-adapter>
   <outbound-resourceadapter>
      <connection-definition>
         <connectionfactory-interface>
            factoryClassName
         </connectionfactory-interface>
         <implemented-interface>
            interfaceClassname
         </implemented-interface>
         <config-property-setting name="propertyName">
            value
         <config-property-setting>
         <connectionmanager>
            <container-managed-security />
            <no-transaction/>
         </connectionmanager>
      </connection-definition>
   </outbound-resourceadapter>
</resource-adapter>
...
</connector>
```

Admin object configuration

In addition to inbound and outbound resource adapters, a connector can also provide administered objects. A common example is the JMS connector that exposes its destination queues or topics.

The elements in the <adminobject> element are:

- <adminobject-interface>: The interface of the object to be administered.
- <adminobject-class>: The class that implements the interface.
- <adminobject-instance>: Contains the message destination name and allows you to configure properties for it.

Example 10-40 is the ActiveMQ resource adapter that is included in Community Edition.

Example 10-40 Administered object configuration for the ActiveMQ resource adapter

```
<adminobject>
  <adminobject-interface>javax.jms.Queue</adminobject-interface>
 <adminobject-class>org.apache.activemq.command.ActiveMQQueue</adminobject-class>
<config-property>
 <config-property-name>PhysicalName</config-property-name>
 <config-property-type>java.lang.String</config-property-type>
 </config-property>
 </adminobject>
<adminobject>
 <adminobject-interface>javax.jms.Topic</adminobject-interface>
<adminobject-class>org.apache.activemq.command.ActiveMQTopic</adminobject-class>
<config-property>
 <config-property-name>PhysicalName</config-property-name>
 <config-property-type>java.lang.String</config-property-type>
</config-property>
</adminobject>
```

For more information see the following Web sites:

http://cwiki.apache.org/GMOxDOC12/geronimo-raxml.html
http://publib.boulder.ibm.com/wasce/V2.0.0/en/rar.html

10.6.9 Application client deployment plan

The Community Edition client application module deployment plan is geronimo-application-jar.xml. The schema that defines this XML is geronimo-application-2.0.xsd and is located in *wasceHome*/schema/.

Example 10-41 shows what a typical geronimo-ra.xml file looks like.

Example 10-41 Typical namespace of an application deployment plan

```
<application-client
xmlns="http://geronimo.apache.org/xml/ns/j2ee/application-client-2.0">
...
</application-client>
```

This deployment plan can be packaged in the JAR file in the META-INF folder.

Application identification

The application client module must be uniquely identified in the server environment and in the client environment. So, the environment element is split into two elements:

- <server-environment>: Identifies the module in the server.
- <client-environment>: Identifies the module in the client.

Both elements have the same elements as the <Environment> element, which we described in 10.6.2, “Module identity” on page 150.

Example 10-42 shows a sample environment configuration in geronimo-application-jar.xml.

Example 10-42 Sample environment configuration in geronimo-application-jar.xml

```
<application-client
xmlns="http://geronimo.apache.org/xml/ns/j2ee/application-client-2.0"
xmlns:dep="http://geronimo.apache.org/xml/ns/deployment-1.2"
>
<dep:client-environment>
   <dep:moduleId>
      <dep:groupId>myClientGroup</dep:groupId>
      <dep:artifactId>MyClientArtifactName</dep:artifactId>
      <dep:version>1.0</dep:version>
      <dep:type>jar</dep:type>
   </dep:moduleId>
   <dep:dependencies/>
   <dep:hidden-classes/>
   <dep:non-overridable-classes/>
</dep:client-environment>

<dep:server-environment>
   <dep:moduleId>
      <dep:groupId>MyServerGroup</dep:groupId>
      <dep:artifactId>MyServerArtifactGroup</dep:artifactId>
      <dep:version>1.0</dep:version>
      <dep:type>jar</dep:type>
   </dep:moduleId>
   <dep:dependencies/>
   <dep:hidden-classes/>
   <dep:non-overridable-classes/>
</dep:server-environment>
</application-client>
```

Security

The following security configuration is applicable for application clients:

- Realm configuration: The <realm-name> element names the default security realm to be used to authenticate users. You must also configure the <callback-handler> element, which defines a class that the application client will use to obtain the JAAS authentication information. This class must implement the javax.security.auth.callback.CallbackHandler interface.
- Default subject: If you want to use a default user to run the application client, the <default-subject> element sets the default used realm and a default user.

Example 10-43 shows the security configurations in geronimo-application-jar.xml.

Example 10-43 Security configurations in geronimo-application-jar.xml

```
<application-client
xmlns="http://geronimo.apache.org/xml/ns/j2ee/application-client-2.0"
xmlns:security="http://geronimo.apache.org/xml/ns/security-2.0"
>
```

```
...
<!-- To use a default userid -->
<security:default-subject>
   <realm>DefaultRealmToBeUsed</realm>
   <id>DefaultUserId</id>
</security:default-subject>

<!-- Defining a security realm-->
<realm-name>VerySecureRealm</realm-name>
<callback-handler></callback-handler>

...
</application-client>
```

Resource element

The <resource> element is required for Java EE assets that are limited to the application client scope.

A resource can be:

- Internal: if it is packaged into the archive file. In this case, the <internal-rar> element is used, and it must contain the resource file name.
- External: if it is not packaged within the archive file. In this case, the <external-rar> element is used, and it must contain the resource URI of the form *group/artifact/version*.

For both types of resources the connector element must be specified to define the Java EE Connector to which it refers. Example 10-44 shows the resource configurations in geronimo-application-jar.xml.

Example 10-44 Resource configurations in geronimo-application-jar.xml

```
<application-client
xmlns="http://geronimo.apache.org/xml/ns/j2ee/application-client-2.0"
>
<resource>
   <internal-rar>filename</internal-rar>
   <connector xmlns=”http://geronimo.apache.org/xml/ns/j2ee/connector-1.2”>
      <!-- Connector deployment plan for the connector -->
      ...
   </connector>
</resource>

<resource>
   <external-rar>groupid/artifactid/version</external-rar>
   <connector xmlns=”http://geronimo.apache.org/xml/ns/j2ee/connector-1.2”>
      <!-- Connector deployment plan for the connector -->
      ...
   </connector>
</resource>
...
</application-client>
```

For more information about application client deployment descriptors, check the following Web site:

http://publib.boulder.ibm.com/wasce/V2.0.0/en/jar.html

10.7 Summary of references

- WebSphere Application Server Community Edition V2.0 documentation

  http://publib.boulder.ibm.com/wasce/V2.0.0/en/index.html

  - Sample Java EE assets

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/samples.html
  - Choosing deployment plans and deployment options

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/deployment-plans.html
  - Deploy

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/deploy.html
  - Developing a deployment plan for a Web ARchive (WAR)

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/war.html
  - Developing a deployment plan for an Enterprise ARchive (EAR)

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/ear.html
  - Developing a deployment plan for an Enterprise Java Bean (EJB) archive

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/ejb.html
  - Adding Java libraries to the server's repository

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/adding-java-libraries.html
  - Using JAX-WS Tools

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/using-jax-ws-tools.html
  - Developing deployment plans

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/developing-deployment-plans.html
  - Security group and name mapping

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/security-group-and-name-mapping.html
  - Configuring resources in the asset scope

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/configuring-resources-in-the-asset-scope.html
  - Managing the classpath

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/managing-the-classpath.html
  - Deploying Java EE assets

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/deploy-assets.html
  - Developing a deployment plan for a connector Resource Adapter Archive

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/rar.html

- Developing a deployment plan for an application client Java Archive

  http://publib.boulder.ibm.com/wasce/V2.0.0/en/jar.html
- Controlling application modules

  http://publib.boulder.ibm.com/wasce/V2.0.0/en/control-applications.html

- Download: WebSphere Application Server Community Edition 2.0

  http://www.ibm.com/developerworks/downloads/ws/wasce
- Apache Geronim0 v2.0 documentation:

  http://cwiki.apache.org/GMOxDOC20/documentation.html
- Installing and removing applications

  http://cwiki.apache.org/GMOxDOC20/installing-and-removing-applications.html
- Starting and stopping application modules

  http://cwiki.apache.org/GMOxDOC20/starting-and-stopping-application-modules.html
- Jar to Jar EJB references (no ear)

  http://cwiki.apache.org/GMOxDOC20/jar-to-jar-ejb-references-no-ear.html
- Apache Geronim0 v1.1 documentation:

  http://cwiki.apache.org/GMOxDOC11/documentation.html
- geronimo-ra.xml

  http://cwiki.apache.org/GMOxDOC11/geronimo-raxml.html
- Apache Geronim0 v1.2 documentation:
- openejb-jar.xml

  http://cwiki.apache.org/GMOxDOC12/openejb-jarxml.html
- Deployment plans

  http://cwiki.apache.org/GMOxDOC12/deployment-plans.html
- geronimo-ra.xml

  http://cwiki.apache.org/GMOxDOC12/geronimo-raxml.html
- Apache Maven Project

  http://maven.apache.org/
- Index of /xml/ns/j2ee

  http://geronimo.apache.org/xml/ns/j2ee/

11

Plug-ins

In this chapter, we introduce the Geronimo plug-in feature of Community Edition. It describes how to use both the command-line tools and the administrative console to search, install, and create a plug-in. It also provides a short overview of the Dojo plug-in that can be used to incorporate Web 2.0 features in your Web applications.

This chapter includes the following topics:

11.1 Geronimo plug-ins

A *Geronimo plug-in* is a pre-configured Geronimo module that you can install into a Community Edition server with no configuration or XML required. A plug-in can be either a complete J2EE application or an extension of the Community Edition server along with all needed configuration files.

You can install a plug-in at runtime using the deploy command or the administrative console. The installation process automatically downloads and installs all required dependencies (JARs or other plug-ins) specified in the metadata information before starting the plug-in itself.

Plugins are also a convenient way to get server upgrades, to integrate other new features, products and services, such as LDAP Server, scheduler, drivers, and so on, which allows a Community Edition instance to be highly customized to specific requirements. You can also package an existing module from a working Community Edition server as a plug-in. This is useful in distributed environments, allowing you to clone features from server to server without dealing with repetitive and error-prone manual configurations.

There are several public plug-in repositories hosting both open source and commercial plug-ins that you can download and install in your Community Edition server:

- **GeronimoPluginCentral.org**

  `http://geronimoplugincentral.org`

  The following is a list of some of the plug-ins you can find at this site:

  - Administrative console
  - Welcome application
  - JSP examples
  - Servlet examples
  - LDAP examples
  - Apache Directory Server
  - OpenJPA
  - JSF plugin

- **geronimoplugins.com**

  `http://www.geronimoplugins.com`

  The following is a list of some of the plug-ins you can find at this site:

  - Oracle XA driver for console
  - Quartz Job Deployer
  - Quartz Scheduler Integration

11.1.1 Installing a plug-in

You can install a plug-in using one of the two following alternatives:

- Using the **`deploy`** command.
- Using the administrative console.

Using the deploy command

Using the **`deploy`** command line tool you can install a plug-in from a remote repository or from the local file system.

Remote repository

Use the following steps to install a plug-in from a remote repository using the deploy tool:

1. Start the Community Edition server.
2. Open a command prompt, and change the directory to *wasceHome*/bin.
3. Type the following command to get a list of all the available server plug-ins from the remote repository:

   ```
   deploy search-plugins <remote_repository_url>
   ```

   Tip: The default remote repository for Community Edition is:

   http://download.boulder.ibm.com/ibmdl/pub/software/websphere/wasce/plugins-2.0/

   The command displays a list of all available plug-ins, as shown in Figure 11-1.

```
C:\Program Files\IBM\WebSphere\AppServerCommunityEdition\bin>deploy search-plugins h
ttp://download.boulder.ibm.com/ibmdl/pub/software/websphere/wasce/plugins-2.0/
Using GERONIMO_BASE:   C:\Program Files\IBM\WebSphere\AppServerCommunityEdition
Using GERONIMO_HOME:   C:\Program Files\IBM\WebSphere\AppServerCommunityEdition
Using GERONIMO_TMPDIR: var\temp
Using JRE_HOME:        C:\Program Files\IBM\Java50\jre
Username: system
Password:


    Samples
        1 : Class Viewer Example (2.0)
        2 : JSF Example (2.0)
        3 : JSP Examples (2.0)
        4 : Servlets Examples (2.0)
        5 : Welcome Application (2.0)

Install Service [enter number or 'q' to quit]:
```

Figure 11-1 List of the available plug-ins from the deploy command

4. Choose the number that corresponds to the plug-in that you want to install. The tool completes the installation and starts the plug-in, as shown in Figure 11-2.

```
Checking for status every 1000ms:
Downloading org.apache.geronimo.configs/welcome-tomcat/2.0/car... (0%)
Downloading org.apache.geronimo.configs/welcome-tomcat/2.0/car... (17%)
Finished downloading org.apache.geronimo.configs/welcome-tomcat/2.0/car (121 kB)
 (100%)

    **** Installation Complete!
    Used existing: org.apache.geronimo.configs/jasper//car
    Used existing: org.apache.geronimo.configs/tomcat6//car

    Downloaded 208 kB in 7s (29 kB/s)
    Now starting org.apache.geronimo.configs/welcome-tomcat/2.0/car...

    Started org.apache.geronimo.configs/welcome-tomcat/2.0/car @ /

C:\Program Files\IBM\WebSphere\AppServerCommunityEdition\bin>
```

Figure 11-2 Download and installation of a plug-in

Standalone installation

It is possible to install a plug-in as a standalone bundle, which is useful when the plug-ins are not available over the Internet but distributed by other means of support (such as CD, DVD, and so on). The syntax is:

```
deploy --user username --password password install-plugin Plugin.zip
```

If the plug-in has dependencies, they are downloaded and installed as well. Refer to Figure 11-3.

```
Checking for status every 1000ms:
Finished downloading default/SimpleWeb/1.0/car (18 kB) (100%)

    **** Installation Complete!
    Used existing: org.apache.geronimo.configs/tomcat6//car
    Used existing: org.apache.geronimo.configs/j2ee-corba-yoko//car
    Used existing: org.apache.geronimo.configs/openjpa//car
    Used existing: org.apache.openejb/openejb-core//jar
    Used existing: org.apache.geronimo.modules/geronimo-openejb//jar
    Used existing: org.apache.openejb/openejb-loader//jar
    Used existing: org.apache.openejb/openejb-ejbd//jar
    Used existing: org.apache.openejb/openejb-server//jar
    Used existing: org.apache.openejb/openejb-client//jar
    Used existing: org.apache.openejb/openejb-javaagent//jar
    Used existing: org.apache.openejb/openejb-jee//jar
    Used existing: org.apache.xbean/xbean-reflect//jar
    Used existing: org.codehaus.swizzle/swizzle-stream//jar
    Used existing: asm/asm//jar
    Used existing: asm/asm-commons//jar
    Used existing: org.apache.geronimo.configs/axis//car
    Used existing: org.apache.geronimo.configs/axis2//car
    Used existing: org.apache.geronimo.configs/jasper//car

    Downloaded 18 kB in 1s (18 kB/s)
    Now starting default/SimpleWeb/1.0/car...

    Started default/SimpleWeb/1.0/car @ /SimpleWeb
```

Figure 11-3 Local installation of a plug-in

Using the administrative console

To install a plugin from the administrative console:

1. Select the **Plugins** portlet to open the Web page, as shown in Figure 11-4 on page 179.

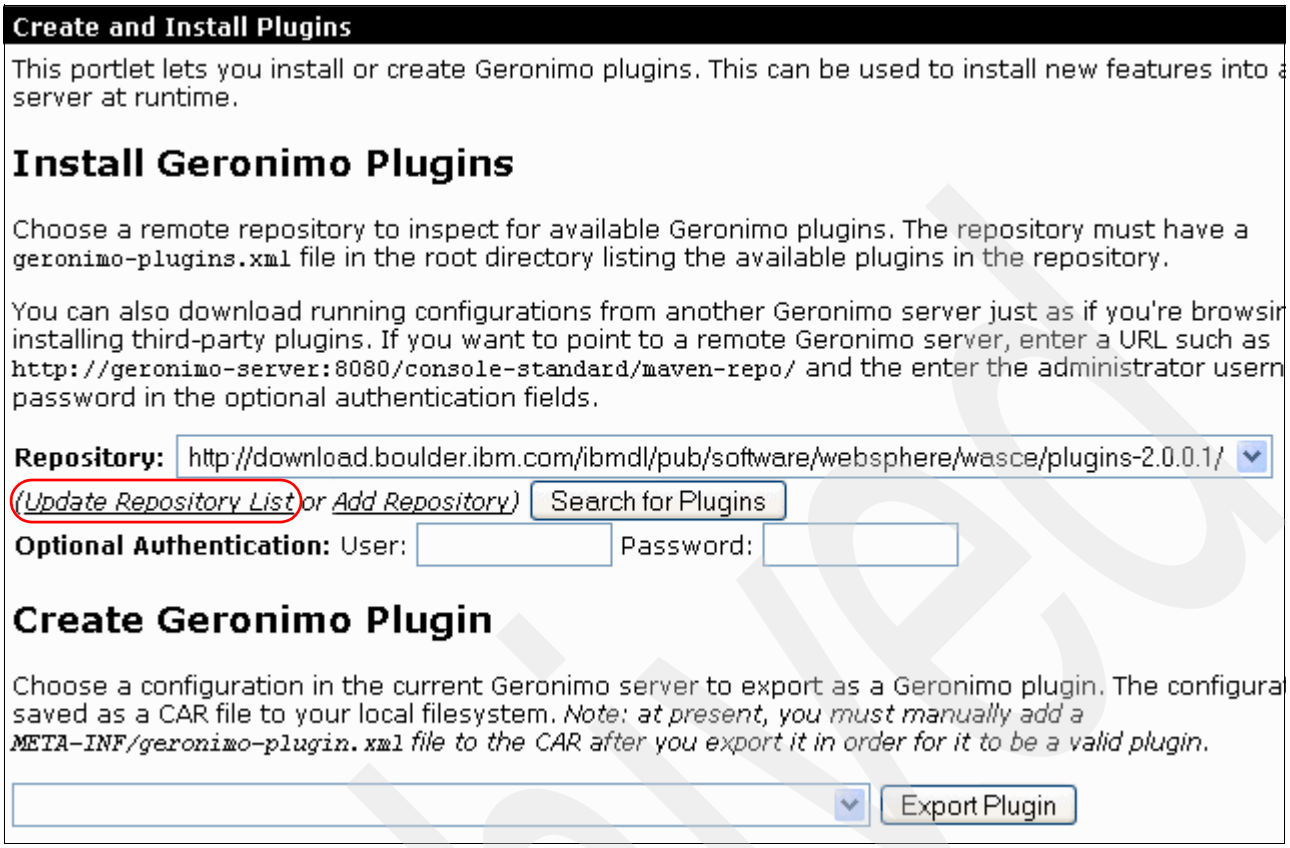


Figure 11-4 The Create and Install Plugin page

2. If the Repository field is empty, click the **Update Repository List** link to populate the pull-down list.
3. From the list, choose the default Community Edition remote plug-ins repository:

   `http://download.boulder.ibm.com/ibmdl/pub/software/websphere/wasce/plugins-2.0/`

   Tip: It is also possible to point to another Community Edition server to install running plug-ins. Click **Add repository**, enter the URL of the repository, for example, http://geronimo-server:8080/console-standard/maven-repo/, and then fill the optional authentication fields with the administrator user name and password.

4. Click the **Search for Plugin** button to get the list of all available plug-ins from that repository, as shown if Figure 11-5 on page 180.

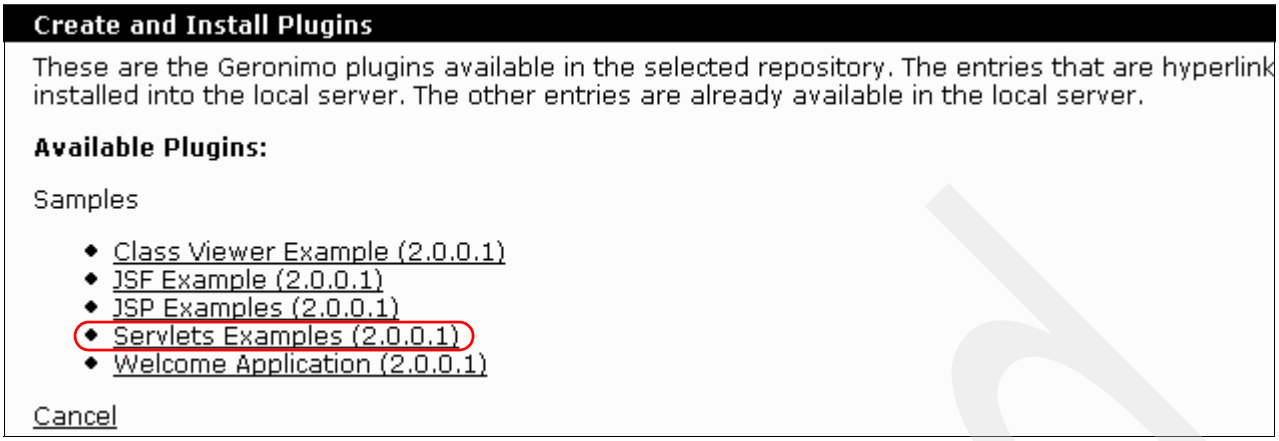


Figure 11-5 Select a plug-in to install

5. Click the link that corresponds to the plug-in that you want to install, which opens a summary page with the plug-in information, as shown in to Figure 11-6.

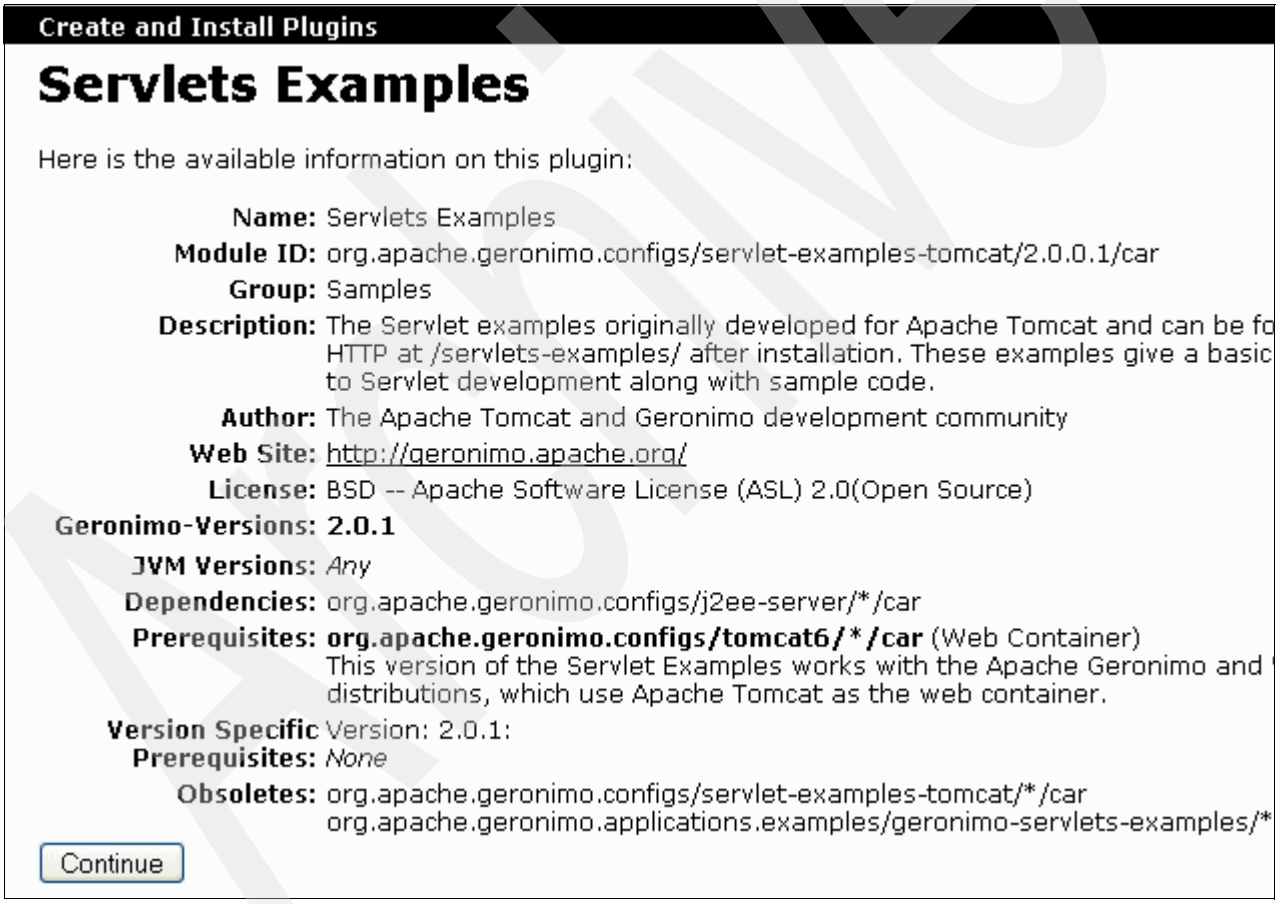


Figure 11-6 Summary of plug-in information

6. Click **Continue**. If the plug-in has dependencies, a message informs you that the dependencies are automatically downloaded and installed as well, as shown in Figure 11-7 on page 181.

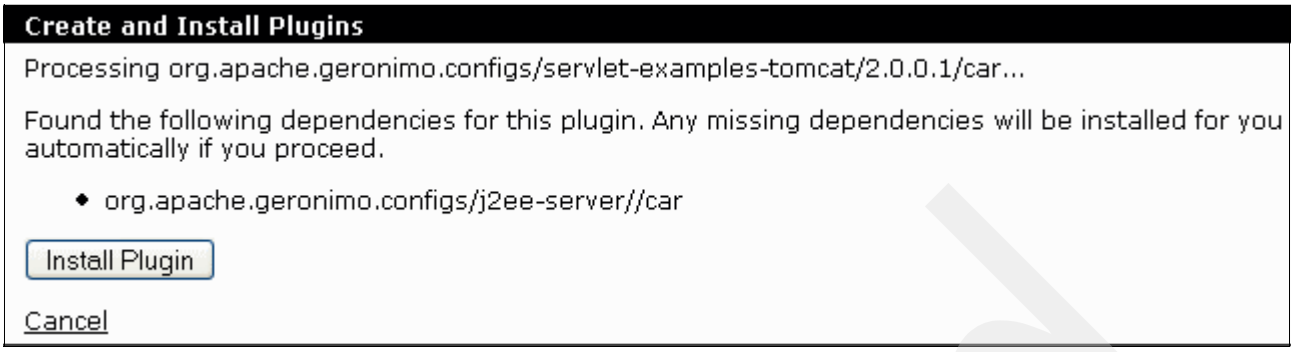


Figure 11-7 Community Edition process any missing required dependencies

7. Click **Install Plugin** to start downloading the plug-in and dependencies. A dynamic page shows the download process.

   A page (Figure 11-8) displays the installation result along with the list of all processed dependencies.

Figure 11-8 Start the plug-in

8. Click **Start *plugin_name*** to start the installation.

   A page (Figure 11-9) displays the list of all plug-ins that are available from the remote repository, and indicates which plug-in is currently installed on Community Edition.

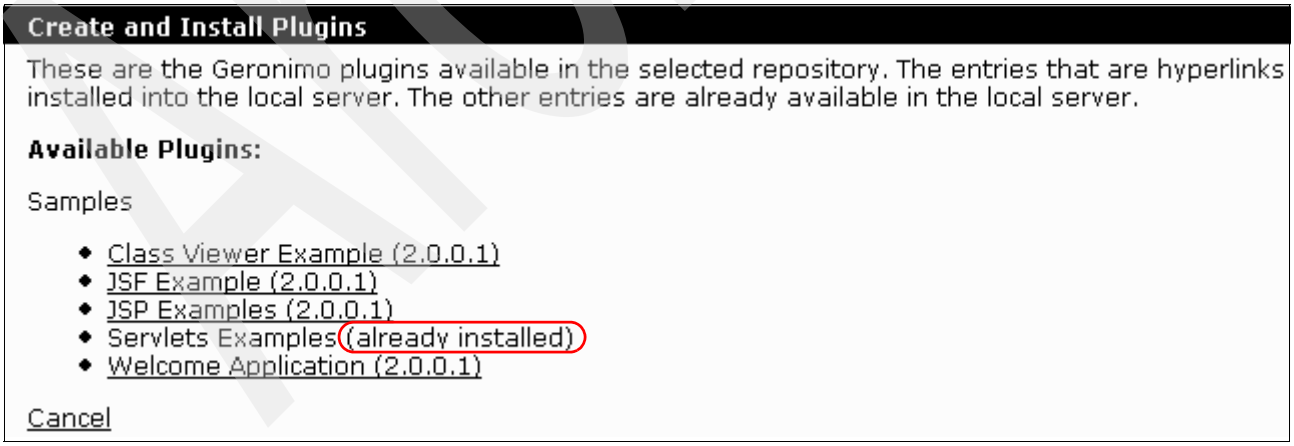


Figure 11-9 The plug-in is installed

After it is installed, the plug-in is a new active Geronimo module that is deployed on Community Edition with no configuration required. The Servlets Example application can be accessed at http://localhost:8080/servlets-examples/.

11.1.2 Exporting a module as a plug-in

After a Community Edition module is configured and installed on a server, you can export it as a Geronimo plug-in to be easily distributed and deployed on any other Community Edition servers.

To export a Community Edition module as a plugin from the administrative console:

1. Select the **Plugins** portlet.
2. To select the module you want to export as a plugin, click the drop-down selection in the Create Geronimo Plugins section, as shown in Figure 11-10.

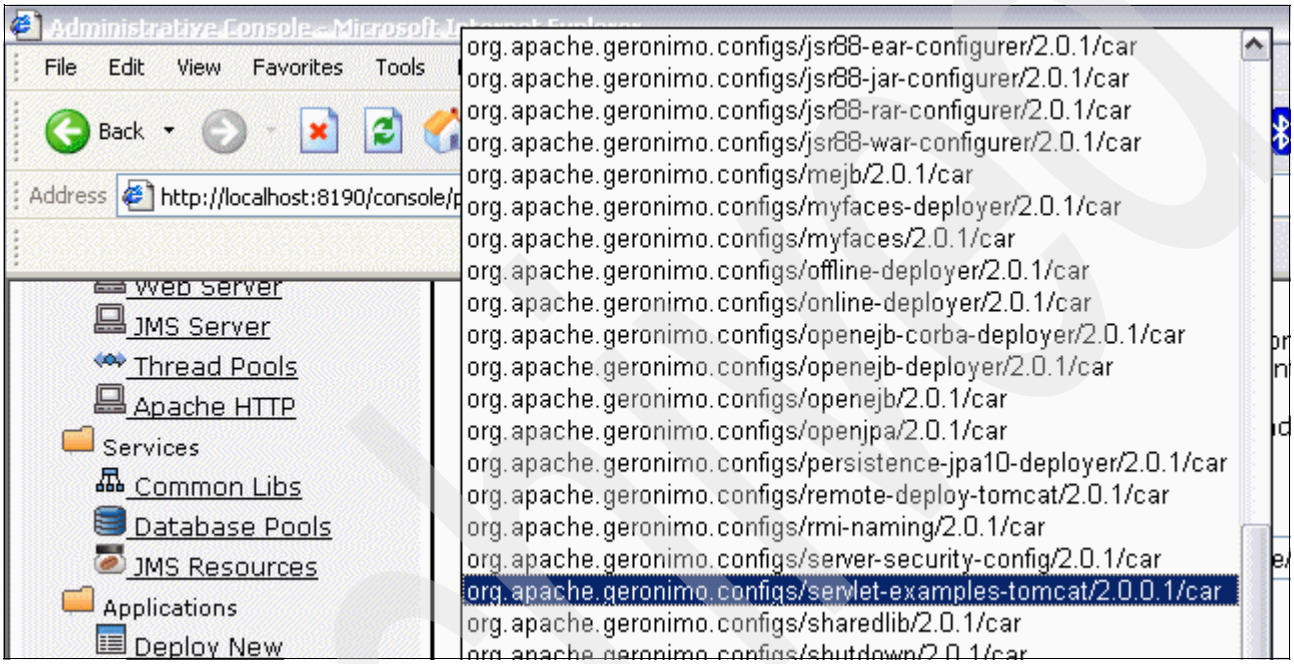


Figure 11-10 Select the module to export

3. Click **Export Plugin** to view the data about the module you are going to export, as shown in Figure 11-11 on page 183.

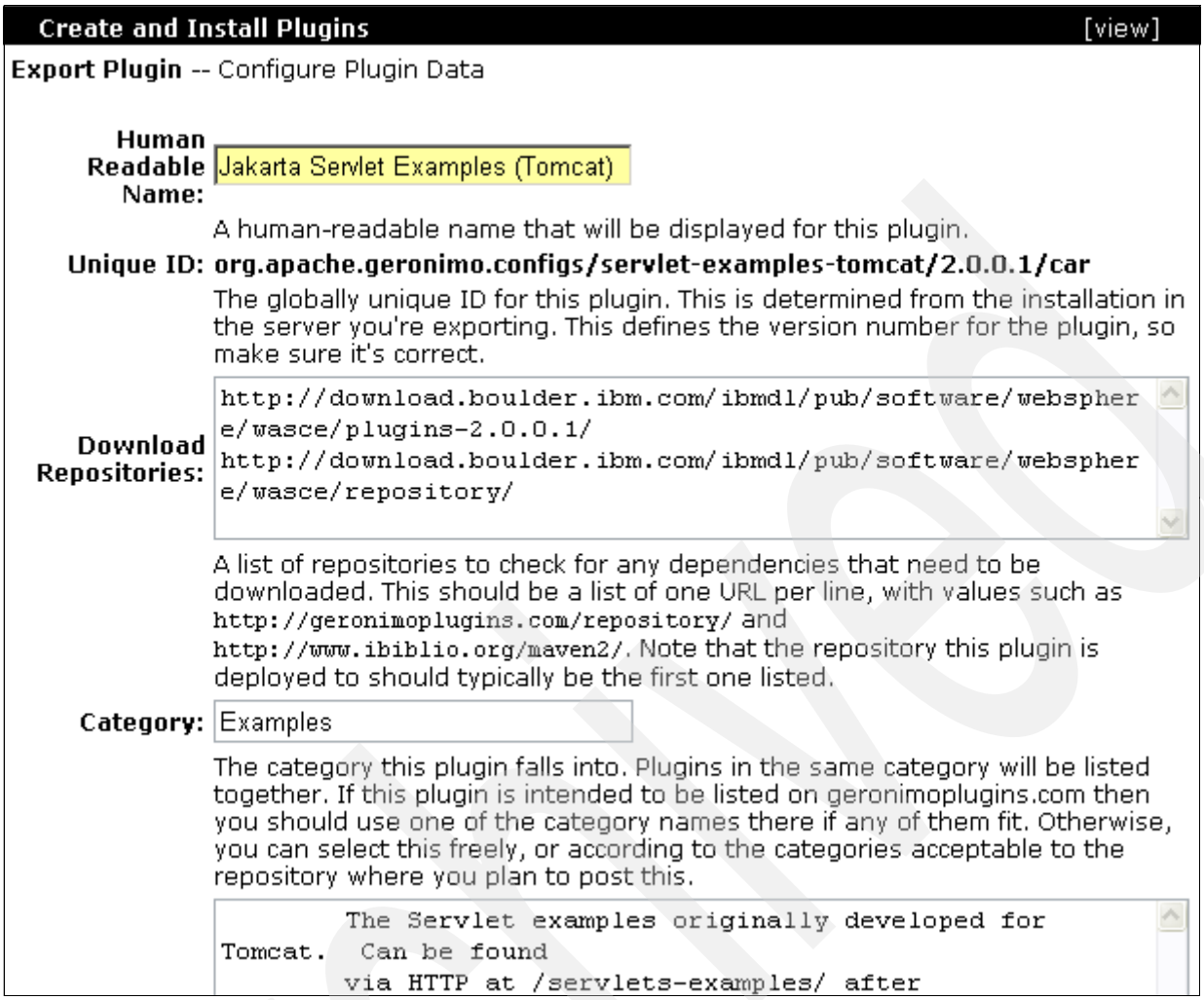


Figure 11-11 Data for the plug-in to be exported

Complete the fields with any additional information. Each field is followed by a description of what you can enter.

Click **Save Plugin Data** to open the final page.

4. Click **Export Plugin**, and then click **OK** on the message to export the plug-in and save it to disk, as shown in Figure 11-12 on page 184.

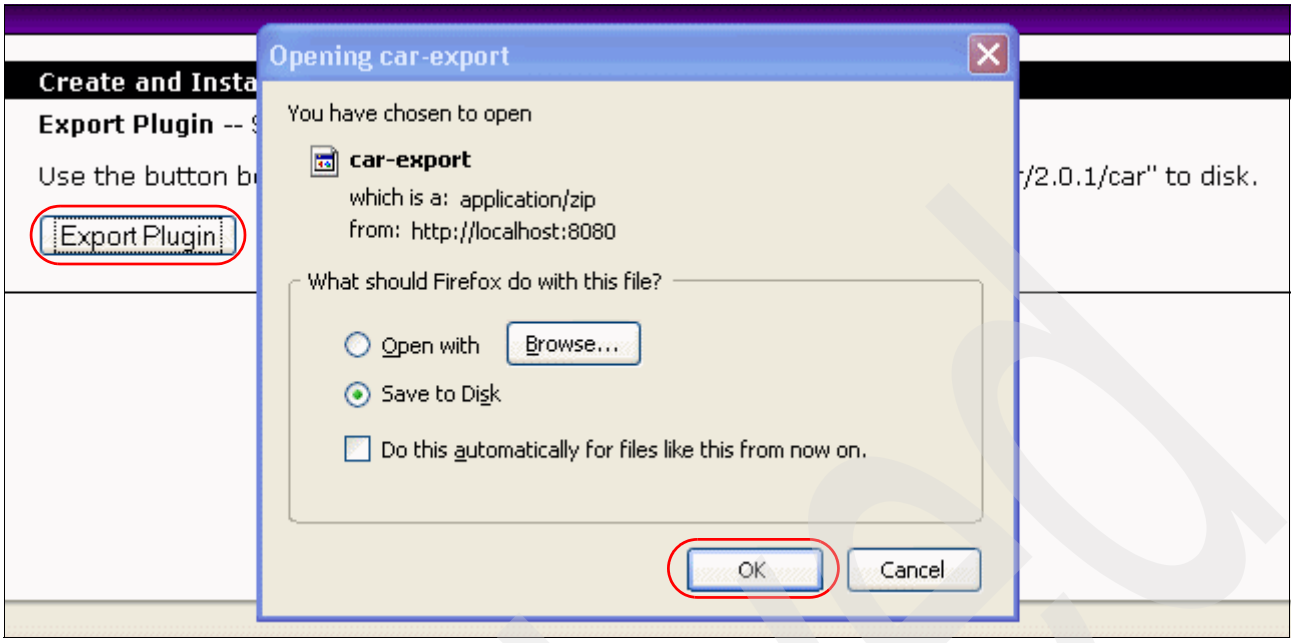


Figure 11-12 Save the plug-in to the disk

11.2 The Dojo plug-in

Dojo is a popular and powerful open source javascript library. Dojo provides a lot of useful features that allow you to build highly interactive Web 2.0 applications, including:

- Ajax facilities
- Widget Toolkit
- Drag and drop support
- Animactions and effects

You can get more information about Dojo at the following Web site:

`http://dojotoolkit.org`

11.2.1 Using Dojo in Community Edition

A common practice for using Dojo is to incorporate all of the library files in the Web application and to reference them from both static and dynamic content, such as servlets, JSPs, and static HTML pages. In this scenario, each Web application has its own private copy of the library, which leads to three major drawbacks:

- High memory consumption
- Browsers cannot use a single copy from cache
- Managing multiple upgrades of the library

To overcome these problems, the Dojo plug-in acts as a single point-of-access that provides a shared copy of the library files in the context /dojo of the Community Edition server.

You can check the availability of the plug-in in the Community Edition server with the following URL:

`http://localhost:8080/dojo/dojo.js`

The browser displays the page shown in Figure 11-13.

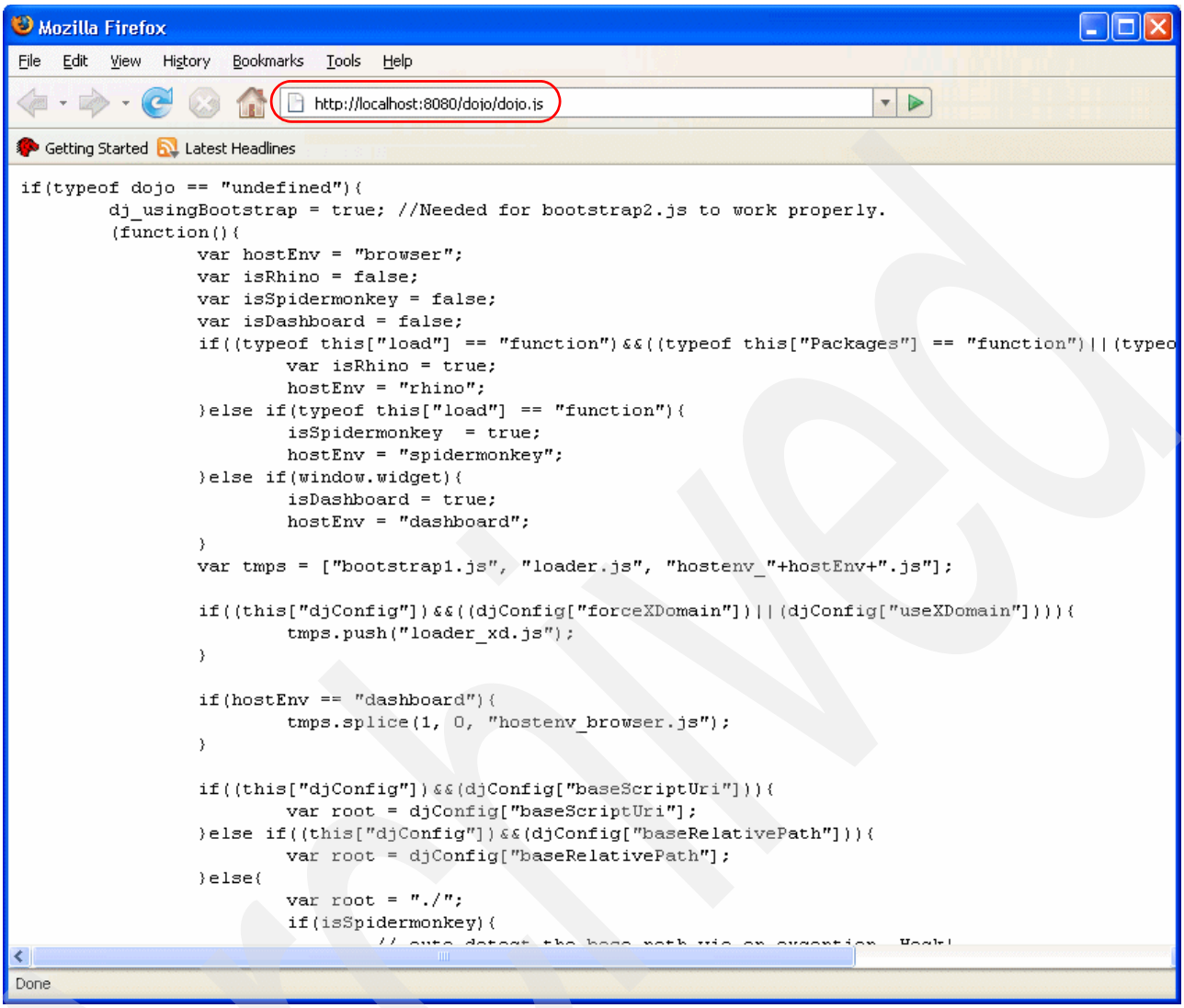


Figure 11-13 The Dojo plug-in is working

Using Dojo

To use Dojo from a Web application, simply put the following line on top of your html pages:

```
<script type="text/javascript" src="/dojo/dojo.js"></script>
```

11.3 Summary of references

- Apache Geronimo Plugin Central
  http://geronimoplugincentral.org
- Geronimo Plugins
  http://geronimoplugins.com
- WebSphere Application Server Community Edition V2.0 Server Plug-ins
  http://download.boulder.ibm.com/ibmdl/pub/software/websphere/wasce/plugins-2.0/
- Dojo Javascript toolkit
  http://dojotoolkit.org

12

Troubleshooting techniques

In this chapter, we describe common problems that can occur during the installation or use of WebSphere Application Server Community Edition. We provide you with debugging tools and information about how to address problems that might occur.

This chapter includes the following topics:

12.1 Diagnostic tools

To diagnose a problem, you need to collect information about the circumstances that the problem occurs in, for example:

- What was the application doing before the failure?
- What did the server do before it crashed?
- Is the memory exhausted and how is it allocated?

Various tools are available to help you create a picture of what was happening when the error occurred. These tools include logs that record actions taken by the application server and debug viewers that are available from the administrative console.

Operating system indicators can be valuable tools to diagnose memory and CPU usage problems. In a Windows machine, you can use the task manager to see the overall CPU and memory usage. In UNIX® machines, you can use commands, such as vmstat, top, and so on to check the CPU and memory usage.

The Community Edition administrative console has a Memory monitor that gives a graph of current memory usage. See 6.3.1, “Monitoring the memory usage” on page 66 for more information about the memory monitor.

12.1.1 Server logs

The following log files are useful in troubleshooting situations.

Note: Log files can become large. To isolate a log error, it is a good idea to shut down the server and rename the desired log file (for example, server.log.old). At server startup, a new log file is created.

- server.log

  Messages from the server, including messages from applications and the messages during deployment to an active server, are logged to the *wasceHome*/var/log/server.log file. You can change the file location, name, and the log levels by editing the *wasceHome*/var/log/server-log4j.properties file, for example, if you want to change the file log level to DEBUG, change the root log level and log4j.appender.FILE.Threshold property in server-log4j.properties to DEBUG, as shown in Figure 12-1 on page 189.

```
log4j.rootLogger=DEBUG, CONSOLE, FILE

log4j.appender.CONSOLE=org.apache.log4j.ConsoleAppender
log4j.appender.CONSOLE.Threshold=${org.apache.geronimo.log.ConsoleLogLevel}
log4j.appender.CONSOLE.Target=System.out
log4j.appender.CONSOLE.layout=org.apache.log4j.PatternLayout
log4j.appender.CONSOLE.layout.ConversionPattern=%d{ABSOLUTE} %-5p [%c{1}]
%m%n

log4j.appender.FILE=org.apache.log4j.RollingFileAppender
log4j.appender.FILE.Threshold=DEBUG
log4j.appender.FILE.layout=org.apache.log4j.PatternLayout
log4j.appender.FILE.layout.ConversionPattern=%d{ABSOLUTE} %-5p [%c{1}] %m%n
```

Figure 12-1 Property file for server.log: server-log4j.properties

You can view the server log file and edit its properties by selecting the **Server Logs** portlet in the administrative console. Using the console to view the logs allows you to use a filter to narrow down the entries that are shown. In Linux, AIX, or Solaris you can use the **`tail -f server.log`** command to display current entries to the server.log file while the server is running.

- client.log

  Messages from J2EE client applications are written to the *wasceHome*/var/log/client.log file. You can change the file location, name, and the log levels by editing the *wasceHome*/var/log/client-log4j.properties.

- deployer.log

  Messages that are generated when an application is deployed to an inactive server are logged to the *wasceHome*/var/log/deployer.log file. You can change the file location, name, and the log levels in *wasceHome*/var/log/deployer-log4j.properties.

- System.out and System.err

  System.out and System.err messages are printed to the Community Edition command-line console. If you are running the Community Edition as a background process in Linux, AIX, or Solaris, you must redirect the output messages to a file. You can do it by using the following command:

  ```
  [prompt]$./startup.sh > ../system.log 2>&1
  ```

  This command redirects both standard output and error messages to system.log file.

- Web server access.log

  Web server logs contain information about requests to the Web container. This information includes the type of request (GET or POST), the time of access, and other related data. You can see the Web access log files for the Web container in the *wasceHome*/var/catalina/logs folder.

  You can view the Web server log file and edit its properties by selecting the **Server Logs** portlet and scrolling down to **Web Access Log Viewer** portlet in the administrative console. Using the console to view the logs allows you to use a filter to narrow down the entries shown.

- The Derby database log

  Derby server logs information about the embedded database manager, including database start and database shutdown information. You can view the Derby server log file in *wasceHome*/var/derby/derby.log.

  You can view the Derby database log file by selecting the **Server Logs** portlet in the administrative console. Using the console to view the logs allows you to use a filter to narrow down the entries that are shown.

Visit the following Web site to get information about how to configure the log files and how to use the administrative console to view log files.

http://cwiki.apache.org/GMOxDOC20/configure-log-level.html

12.1.2 Debug viewers

Community Edition V2.0 includes five debug views in the administrative console, which we show in Figure 12-2.

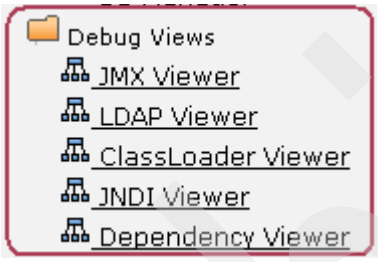


Figure 12-2

The Debug Views can help you troubleshoot problems because they provide information about system and user defined modules. Using most of the views you can search for your application name within the tree-like structure. Under the Debug Views section, the troubleshooter can find:

- The *JMX Viewer* displays the available Geronimo MBeans in a tree-like structure and provides attribute and statistical information. You can also execute any of the managed bean's supported operations.
- One way to store user information across an enterprise is to use an LDAP server. The *LDAP Viewer* helps you to connect to an LDAP server and browse its contents.
- The *ClassLoader Viewer* displays the different classloaders, their relationship, and the classes loaded by them. This view is helpful in debugging classloading related issues.
- The *JNDI Viewer* displays the JNDI context of various modules.
- Using the *Dependency Viewer* you can view the dependencies of all modules.

For more information about these viewers, see the *Enhanced Web console* section of the following developerWorks article:

http://www.ibm.com/developerworks/websphere/library/techarticles/0709_jain/0709_jain.html?S_TACT=105AGX10&S_CMP=WASCE#sec1

12.1.3 MustGather documents

MustGather documents aid you in gathering information for problem determination of problems. They are useful both as an initial approach to determining what information is

useful in a troubleshooting situation but also as a guide in collecting documentation for reporting problems to IBM technical support.

There are many MustGather documents that are available and tailored to the product type, version, and even for specific components or symptoms. The following Web site is a good place to start if you are not sure where to start, or if you have a problem that is not addressed by a specific MustGather document:

MustGather: Read first for all WebSphere Application Server Community Edition components:

http://www-1.ibm.com/support/docview.wss?rs=2359&context=SS6JMN&dc=DB520&uid=swg21231419&loc=en_US&cs=UTF-8&lang=en

12.1.4 Online resources

A lot of information related to diagnosing problems in Community Edition is available on the Community Edition support site. You can find links to FAQs, common problems and their solutions, documentation, and other information at this Web site:

http://www-306.ibm.com/software/webservers/appserv/community/support/

The Community Edition discussion forum is another resource you can use. If you cannot diagnose a problem, you can post a question on the forum to request feedback from others. The forum is at:

http://www.ibm.com/developerworks/forums/forum.jspa?forumID=541

If you have IBM support for Community Edition, you can contact the IBM support team for your questions or defects. More information about the support that is available with Community Edition is in 1.6, “Support and training” on page 6.

12.1.5 JConsole

JConsole is a graphical tool that allows you to monitor the behavior of both the JVM and the Java applications. It provides useful information about performance and resources utilization of the JVM. See “JConsole” on page 67 for information about using JConsole.

12.2 Collecting information for IBM support

In this section, we provide you with items that you need to gather for IBM support.

System and JVM dumps

Advanced problem determination might require you to generate and collect dumps that show the state of the system at the time of the error. For more information about how to generate various dump files, visit the following Web site:

http://www-1.ibm.com/support/docview.wss?rs=2359&context=SS6JMN&dc=DB520&dc=DB560&uid=swg21259615&loc=en_US&cs=utf-8&lang=en

Using the Collector tool

The Collector tool is a Community Edition configuration dump tool that you can use to take a snapshot of a running server configuration.The configuration dump might help the Community Edition support team for easier problem determination. You can run the command using the

wasceHome\bin\collector.bat in Windows systems or *wasceHome*/bin/collector.sh in Linux, AIX, or Solaris systems.

For a more information about using the Collector tool, visit:

http://publib.boulder.ibm.com/wasce/V2.0.0/en/collector.html

12.3 Server startup problems

In this section, we discuss diagnostic techniques and possible causes of an application server start failure.

12.3.1 Port conflict

One common cause for a startup failure is a port conflict, which occurs most often because another application server is running on the same system. If a port conflict occurs, you will see an exception trace that resembles Figure 12-3 in server.log.

```
15:49:10,703 INFO  [Log4jService] ------------------------------
15:49:12,015 WARN  [RMIRegistryService] RMI Registry failed
15:49:12,015 ERROR [GBeanInstanceState] Error while starting: GBean
java.rmi.server.ExportException: Port already in use: 1099; nested e
        at sun.rmi.transport.tcp.TCPTransport.listen(TCPTransport.ja
        at sun.rmi.transport.tcp.TCPTransport.exportObject(TCPTransp
        at sun.rmi.transport.tcp.TCPEndpoint.exportObject(TCPEndpoin
        at sun.rmi.transport.LiveRef.exportObject(LiveRef.java:131)
        at sun.rmi.server.UnicastServerRef.exportObject(UnicastServe
        at sun.rmi.registry.RegistryImpl.setup(RegistryImpl.java:107
        at sun.rmi.registry.RegistryImpl.<init>(RegistryImpl.java:93
        at java.rmi.registry.LocateRegistry.createRegistry(LocateReg
        at org.apache.geronimo.kernel.rmi.RMIRegistryService.doStart
        at org.apache.geronimo.gbean.runtime.GBeanInstance.createIns
        at org.apache.geronimo.gbean.runtime.GBeanInstanceState.atte
        at org.apache.geronimo.gbean.runtime.GBeanInstanceState.star
        at org.apache.geronimo.gbean.runtime.GBeanInstanceState.star
        at org.apache.geronimo.gbean.runtime.GBeanInstance.startRecu
        at org.apache.geronimo.kernel.basic.BasicKernel.startRecursi
        at org.apache.geronimo.kernel.config.ConfigurationUtil.start
        at org.apache.geronimo.kernel.config.KernelConfigurationMana
        at org.apache.geronimo.kernel.config.SimpleConfigurationMana
        at org.apache.geronimo.kernel.config.SimpleConfigurationMana
        at net.sf.cglib.reflect.FastMethod.invoke(FastMethod.java:53
        at org.apache.geronimo.gbean.runtime.FastMethodInvoker.invok
        at org.apache.geronimo.gbean.runtime.GBeanOperation.invoke(G
        at org.apache.geronimo.gbean.runtime.GBeanInstance.invoke(GB
        at org.apache.geronimo.gbean.runtime.RawInvoker.invoke(RawIn
        at org.apache.geronimo.kernel.basic.RawOperationInvoker.invo
        at org.apache.geronimo.kernel.basic.ProxyMethodInterceptor.i
        at org.apache.geronimo.kernel.config.EditableConfigurationMa
        at org.apache.geronimo.system.main.EmbeddedDaemon.doStartup(
        at org.apache.geronimo.system.main.EmbeddedDaemon.execute(Em
        at org.apache.geronimo.kernel.util.MainConfigurationBootstra
        at org.apache.geronimo.cli.AbstractCLI.executeMain(AbstractC
        at org.apache.geronimo.cli.daemon.DaemonCLI.main(DaemonCLI.j
Caused by:
```

Figure 12-3 Port error messages

Check the *wasceHome*/var/config/config_substitutions.properties file for the list of ports that Community Edition uses. If any of these ports are already in use, the server fails to start. If port conflict occurs, you can do one of the following:

- Identify and stop the service that is using the conflicted port

  If possible, you can stop the service using the conflicting port, and start Community Edition again, which should resolve the problem.

- Assign a different port for the conflicting port in Community Edition

  You can change the ports that Community Edition uses by editing the var\config\config.xml or by changing the ports in config-substitutions.properties.

  The *portOffset* property in the config-substitutions.properties file adds an offset to all of the port properties that are specified in the property file, for example, by default, the RMI Registry in Community Edition uses port 1099. If you specify portOffset as 100, the RMI Registry port is 1199. Specify a different portOffset value for each instance of Community Edition running in the same machine.

12.3.2 Unable to find the Java home location

During installation, Community Edition identifies and stores the Java runtime location in *wasceHome*/bin/setenv.bat. If you uninstall the Java runtime and install it in a new location, the application server cannot find the new location and does not start. Edit setenv.bat, and point WASCE_JAVA_HOME to the new Java home directory, as Figure 12-4 illustrates.

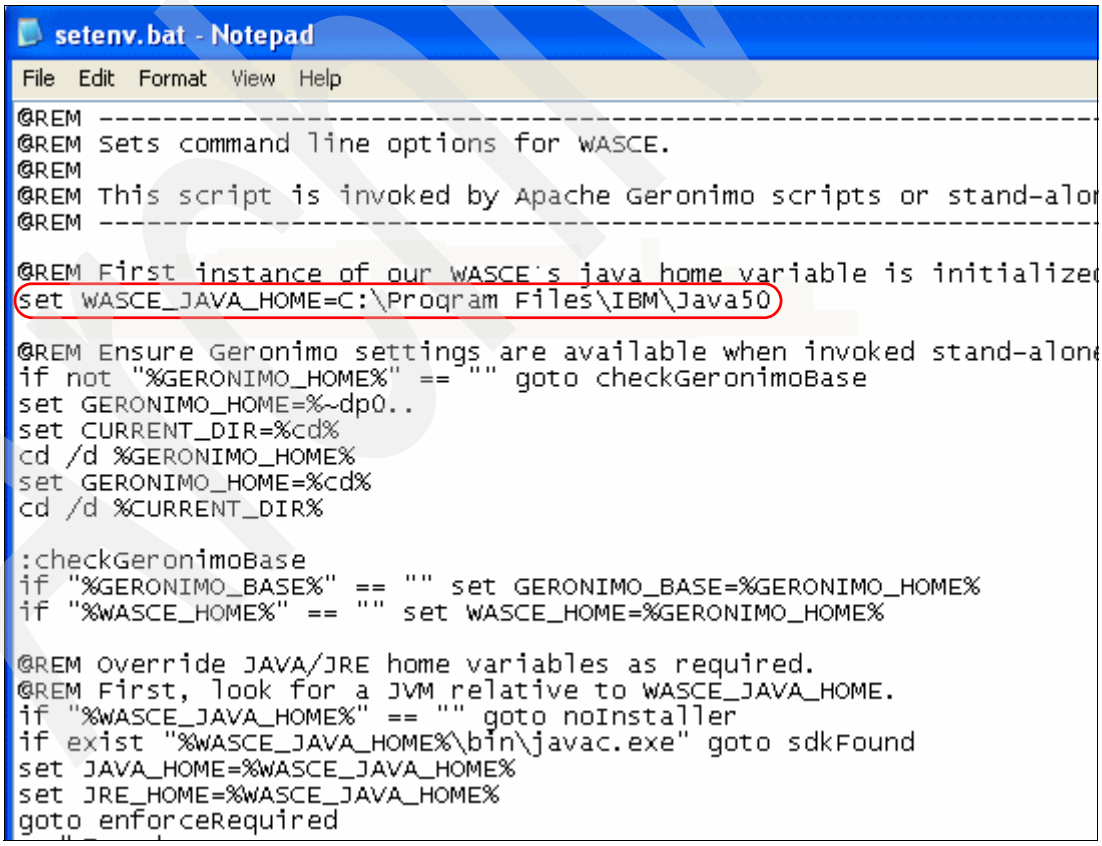


Figure 12-4 WASCE_JAVA_HOME in setenv.bat

12.4 100% CPU usage

Analyzing a 100% CPU usage situation can require you to gather a number of diagnostic indicators for IBM support to analyze, which can include generating a coredump, analyzing the network activities, analyzing verbose GC output, CPU profiling, and running various diagnostic tools. Use the following resources:

- The following Web site provides MustGather information that you will need to gather and how to gather it on a Linux platform.

  http://www-1.ibm.com/support/docview.wss?rs=2359&context=SS6JMN&dc=DB520&dc=DB560&uid=swg21259526&loc=en_US&cs=UTF-8&lang=en&rss=ct2359websphere

- The following Web site provides MustGather information that you will need to gather and how to gather it on a Windows platform.

  http://www-1.ibm.com/support/docview.wss?rs=2359&context=SS6JMN&dc=DB520&dc=DB560&uid=swg21264875&loc=en_US&cs=utf-8&lang=en

After you have this documentation, contact IBM support.

12.5 100% memory usage

Analyzing a 100% memory usage situation can require you to gather a number of diagnostic indicators for IBM support to analyze, which can include generating a coredump, analyzing the network activities, analyzing verbose GC output, analyzing the memory settings, and running various diagnostic tools. Use the following resources:

- The following Web site provides MustGather information that you will need to gather and how to gather it on a Linux platform.

  http://www-1.ibm.com/support/docview.wss?rs=2359&context=SS6JMN&dc=DB520&dc=DB560&uid=swg21264702&loc=en_US&cs=utf-8&lang=en

- The following Web site MustGather information that you will need to gather and how to gather it on a Windows platform.

  http://www-1.ibm.com/support/docview.wss?rs=2359&context=SS6JMN&dc=DB520&dc=DB560&uid=swg21264612&loc=en_US&cs=utf-8&lang=en

12.6 Server crash

Analyzing a server crash can require you to gather a number of diagnostic indicators for IBM support to analyze, which can include generating a crash dump, coredump, and running various tools. Use the following resources:

- The following Web site provides MustGather information and how to gather it on a Linux platform.

  http://www-1.ibm.com/support/docview.wss?rs=2359&context=SS6JMN&dc=DB520&dc=DB560&uid=swg21246239&loc=en_US&cs=utf-8&lang=en

- Refer the following Web site for a more detailed information about the must gathers and how to gather it on Windows platform.

  http://www-1.ibm.com/support/docview.wss?rs=2359&context=SS6JMN&dc=DB520&dc=DB560&uid=swg21246238&loc=en_US&cs=utf-8&lang=en

12.7 Summary of references

- WebSphere Application Server Community Edition V2.0 documentation

  http://publib.boulder.ibm.com/wasce/V2.0.0/en/index.html

  - collector command

    http://publib.boulder.ibm.com/wasce/V2.0.0/en/collector.html

- Apache Geronimo v2.0 documentation

  http://cwiki.apache.org/GMOxDOC20/documentation.html

  - Configure log level

    http://cwiki.apache.org/GMOxDOC20/configure-log-level.html

- What is new in WebSphere Application Server Community Edition V2.0: Enhanced Web console

  http://www.ibm.com/developerworks/websphere/library/techarticles/0709_jain/0709_jain.html?S_TACT=105AGX10&S_CMP=WASCE#sec1

- MustGather: Read first for all WebSphere Application Server Community Edition components

  http://www-1.ibm.com/support/docview.wss?rs=2359&context=SS6JMN&dc=DB520&uid=swg21231419&loc=en_US&cs=UTF-8&lang=en

- WebSphere Application Server Community Edition Product support

  http://www-306.ibm.com/software/webservers/appserv/community/support/

- IBM WebSphere Application Server Community Edition and Apache Geronimo forum

  http://www.ibm.com/developerworks/forums/forum.jspa?forumID=541

- How to generate system core dump, Java core dump, heap dump, and a snap dump when WebSphere Application Server Community Edition is running as an operating system service

  http://www-1.ibm.com/support/docview.wss?rs=2359&context=SS6JMN&dc=DB520&dc=DB560&uid=swg21259615&loc=en_US&cs=utf-8&lang=en

- MustGather: WebSphere Application Server Community Edition 100% CPU on Linux

  http://www-1.ibm.com/support/docview.wss?rs=2359&context=SS6JMN&dc=DB520&dc=DB560&uid=swg21259526&loc=en_US&cs=UTF-8&lang=en&rss=ct2359websphere

- MustGather: information that you must gather and how to gather it on a Windows platform

  http://www-1.ibm.com/support/docview.wss?rs=2359&context=SS6JMN&dc=DB520&dc=DB560&uid=swg21264875&loc=en_US&cs=utf-8&lang=en

- MustGather: WebSphere Application Server Community Edition OutOfMemory errors on Linux

  http://www-1.ibm.com/support/docview.wss?rs=2359&context=SS6JMN&dc=DB520&dc=DB560&uid=swg21264702&loc=en_US&cs=utf-8&lang=en

- MustGather: WebSphere Application Server Community Edition OutOfMemory errors on Windows

  http://www-1.ibm.com/support/docview.wss?rs=2359&context=SS6JMN&dc=DB520&dc=DB560&uid=swg21264612&loc=en_US&cs=utf-8&lang=en

- MustGather: WebSphere Application Server Community Edition JVM crash on Linux

  http://www-1.ibm.com/support/docview.wss?rs=2359&context=SS6JMN&dc=DB520&dc=DB560&uid=swg21246239&loc=en_US&cs=utf-8&lang=en

- MustGather: WebSphere Application Server Community Edition JVM Crash on Windows

  http://www-1.ibm.com/support/docview.wss?rs=2359&context=SS6JMN&dc=DB520&dc=DB560&uid=swg21246238&loc=en_US&cs=utf-8&lang=en

- Using jconsole

  http://java.sun.com/j2se/1.5.0/docs/guide/management/jconsole.html

13

Migration to Community Edition

In this chapter, we discuss the various migration paths to Community Edition, which includes the following topics:

13.1 Migration from Community Edition v1.1 to Community Edition 2.0

In this section, we discuss the migration possibilities from Community Edition V1.1 to Community Edition V2.0.

13.1.1 Differences between V1.1 and V2.0

Table 13-1 shows a summary of the major differences between Community Edition 1.1 and 2.0.

Table 13-1 Differences from V1.1 and V2.0

Feature	V1.1	V2.0
Java EE	1.4	5
Servlet	2.4	2.5
JSP	2.0	2.1
JSTL		1.2
EJB	2.1	3.0
JSF		1.2
Java Persistence API		1.0
JMS	1.1	1.1
JavaMail	1.3	1.4
WSEE	1.1	1.2
WS Metadata 2.0		2.0
JAX-WS		2.0

More details about new features in Community Edition V2.0 are available at:

http://www.ibm.com/developerworks/websphere/library/techarticles/0709_jain/0709_jain.html

13.1.2 Migration considerations

Certainly, the most important change with Community Edition V2.0 is the adoption of new Java EE 5 specifications (see 1.5.1, “Support for Java Enterprise Edition V5” on page 4). Therefore, if you plan to migrate your applications from Community Edition V1.1, you might also consider updating them to the new standard so that they benefit from new functions and technologies.

To accomplish the migration from J2EE 1.4 to Java EE 5:

1. Refactor old application code to reflect the adoption of new standard.
2. Modify the deployment plans using new schema files, according to Table 13-2 on page 199.

Table 13-2 Deployment plan migration from V1.1 to V2.0

Module type	Deployment plan	V1.1 schema	V2.0 schema
Web Application (WAR)	WEB-INF/geronimo-web.xml	geronimo-web-1.1.xsd	geronimo-web-2.0.xsd
EJB Application (JAR)	META-INF/openejb-jar.xml	openejb-jar-2.1.xsd	geronimo-openejb-2.1.xsd
Enterprise Application (EAR)	META-INF/geronimo-application.xml	geronimo-application-1.1.xsd	geronimo-application-2.0.xsd
J2EE Connectors (RAR)	META-INF/geronimo-ra.xml	geronimo-connector-1.1.xsd	geronimo-connector-1.2.xsd
Client Application	META-INF/geronimo-application-client.xml	geronimo-application-client-1.1.xsd	geronimo-application-client-2.0.xsd

For more information about deployment plans, see Chapter 10, “Packaging, deploying, and managing applications” on page 133.

For a complete list of new features in Java EE 5, refer to:

http://jcp.org/en/jsr/detail?id=244

Migrating J2EE1.4 applications

If you do not plan to upgrade your J2EE1.4 applications to Java EE 5, you can continue to deploy them *as-is* because the old schemas are still valid and accepted. However, before you migrate a J2EE1.4 application, we highly recommend that you update at least the deployment plans to reflect the changes in the namespace references.

The following sections describe the most important differences between the deployment plans for the following JEE assets:

- WEB application
- EJB application
- EAR application

For information about the schema files, see:

- For Community Edition V1.1:
  - http://publib.boulder.ibm.com/wasce/V1.1.0/en/Reference/Plans/index.html
  - http://geronimo.apache.org/apache-geronimo-v11-xml-schemas.html
- For Community Edition V2.0:
  - http://publib.boulder.ibm.com/wasce/V2.0.0/en/developing-deployment-plans.html
  - http://geronimo.apache.org/apache-geronimo-v20-xml-schemas.html

13.1.3 Changes in security configuration

When you migrate from V1.1 to V2.0, you must update deployment plans with security elements to change the way the `<default-subject>` and `<run-as-subject>` elements are specified. Example 13-1 shows the new elements (changes in bold).

Example 13-1 Changes in security for V2.0

```
<security:security
  doas-current-caller="choice1"
  use-context-handler="choice2"
  default-role="role">
...
   <default-subject>
```

```
    <realm>realm</realm>
    <id>name</id>
   </default-subject>
...
   <role-mappings>
      <role role-name="role">
...
         <run-as-subject>
            <realm>realm</realm>
            <id>name</id>
         </run-as-subject>
...
      </role>
  </role-mappings>
...
</security>
```

We discuss security in deployment plans in 10.6.3, “Security configuration” on page 151”.

13.1.4 Migrating a J2EE 1.4 Web application

A J2EE 1.4 Web application that runs on Community Edition V1.1 should be deployable *as-is* into the V2.0 without any problems, except for the `<default-subject>` and `<run-as-subject>` security configurations, as we discussed in 13.1.3, “Changes in security configuration” on page 199. However, we recommend that you update META-INF/geronimo-web.xml, which is embedded in the WAR, to reflect the new namespaces, as shown in Example 13-2 (changes in bold):

Example 13-2 geronimo-web.xml namespace changes for V2.0

```
<?xml version="1.0" encoding="UTF-8"?>

<web-app xmlns="http://geronimo.apache.org/xml/ns/j2ee/web-2.0"
  xmlns:dep="http://geronimo.apache.org/xml/ns/deployment-1.2"
  xmlns:naming="http://geronimo.apache.org/xml/ns/naming-1.2"
  xmlns:security="http://geronimo.apache.org/xml/ns/security-2.0">
...
</web-app>
```

We discuss, in detail, Web module deployment plans for Community Edition V2.0 in 10.6.5, “Web module deployment plan” on page 157.

For information about the geronimo-web.xml deployment plan, see:

- For Community Edition V1.1:
  - http://publib.boulder.ibm.com/wasce/V1.1.0/en/Reference/Plans/geronimo-web-1.1.xsd.html
  - http://cwiki.apache.org/GMOxDOC11/geronimo-webxml.html
- For Community Edition V2.0:

  http://publib.boulder.ibm.com/wasce/V2.0.0/en/war.html

13.1.5 Migrating a J2EE 1.4 EJB application

It is possible to deploy an EJB artifact that was previously deployed into Community Edition V1.1 into V2.0 without any problems, except for the `<default-subject>` and `<run-as-subject>` security configurations that we discussed in 13.1.3, “Changes in security configuration” on page 199. Update the **META-INF/openejb-jar.xml** that is embedded in the EJB jar to reflect the new namespaces, as shown in Example 13-3 (changes in bold):

Example 13-3 openejb-xml.xml namespace changes for V2.0

```
<?xml version="1.0" encoding="UTF-8"?>

<openejb-jar xmlns=xmlns="http://www.openejb.org/xml/ns/openejb-jar-2.1"
  xmlns:dep="http://geronimo.apache.org/xml/ns/deployment-1.2"
  xmlns:naming="http://geronimo.apache.org/xml/ns/naming-1.2"
  xmlns:security="http://geronimo.apache.org/xml/ns/security-2.0"
  xmlns:pkgen="http://www.openejb.org/xml/ns/pkgen-2.0">
...
</openejb-jar>
```

We discuss EJB module deployment plans in 10.6.7, “EJB module deployment plan” on page 162.

For a complete reference about the openejb-jar.xml deployment plan, see:

- For Community Edition1.1:
  - http://publib.boulder.ibm.com/wasce/V1.1.0/en/Reference/Plans/openejb-jar-2.1.xsd.html
  - http://cwiki.apache.org/GMOxDOC11/openejb-jarxml.html
- For Community Edition V2.0:
  - http://publib.boulder.ibm.com/wasce/V2.0.0/en/ejb.html

13.1.6 Migrating a J2EE 1.4 EAR application

For an enterprise application, we recommend that you update the **META-INF/geronimo-application.xml** embedded in the EAR archive. As illustrated in Example 13-4, changes are required to the `<default-subject>` and `<run-as-subject>` security configurations, which we discussed in 13.1.3, “Changes in security configuration” on page 199.

Example 13-4 geronimo-application.xml namespace changes for V2.0

```
<?xml version="1.0" encoding="UTF-8"?>

<web-app xmlns="http://geronimo.apache.org/xml/ns/j2ee/application-2.0"
  xmlns:dep="http://geronimo.apache.org/xml/ns/deployment-1.2"
  xmlns:security="http://geronimo.apache.org/xml/ns/security-2.0">
```

We discuss enterprise module deployment plans in 10.6.6, “Enterprise module deployment plan” on page 159.

For a complete reference about the geronimo-application.xml deployment plan, see:

- For Community Edition V1.1:
  - http://publib.boulder.ibm.com/wasce/V1.1.0/en/Reference/Plans/geronimo-application-1.1.xsd.html
  - http://cwiki.apache.org/GMOxDOC11/geronimo-applicationxml.html
- For Community Edition V2.0:
  - http://publib.boulder.ibm.com/wasce/V2.0.0/en/ear.html

13.2 JBoss to Community Edition migration

If you wrote your applications as per Java EE 5 or J2EE 1.4 standard, migrating your application to Community Edition is straightforward. All you need to do is create Community Edition specific deployment plans that correspond to the JBoss specific deployment plans. If you use any JBoss specific features in your application, you must find similar features in Community Edition that satisfy the requirement. In this section, we give you some guidance on the considerations and options that are available for migrating applications from JBoss to Community Edition.

Migrating from JBoss to Community Edition involves migrating the following artifacts:

- Servlets and JSPs
- JDBC
- JCA
- EJBs
- Web services
- Security
- JBoss specific classes

Table 13-3 gives you a comparison of Community Edition 2.0 and JBoss 4.2 feature sets.

Table 13-3 Comparison chart

Feature	Community Edition 2.0	JBoss 4.2
Java EE certification	5	1.4
Servlet	2.5	2.5
JSP	2.1	2.1
JSF	1.2	1.2
EJB	3.0	3.0 (not compliant)
Java Persistence API	1.0 (OpenJPA)	1.0 (w/Hibernate)
JMS	1.1	1.1
Web Service	2.0	1.x

13.2.1 Manual migration

If you have a small number of artifacts to migrate, you might choose to do a manual migration, which involves creating Community Edition-specific deployment plans that

correspond to JBoss-specific deployment plans and finding similar features in Community Edition for JBoss specific features used.

The following Web site has more details about how to migrate J2EE artifacts from JBoss to Community Edition.

http://cwiki.apache.org/GMOxDOC20/migrating-to-apache-geronimo.html

13.2.2 J2G tool for migration

J2G is a freely downloadable tool from Apache Software Foundation that can help you migrate an enterprise application to Community Edition. The tool has three components:

- Source Identification Tool
- Descriptor Conversion Tool
- Resource Conversion Tool

Run one after another, these tools migrate your JBoss application to Community Edition. You can download a J2G nightly build from:

http://people.apache.org/dist/geronimo/j2g/nightly/

See the following article for information about how to use the J2G tool:

http://cwiki.apache.org/GMOxDOC11/using-j2g.html#UsingJ2G-Components

13.3 Tomcat to Community Edition migration

Community Edition has Tomcat as its Web-tier container, which makes a transition from Tomcat to Community Edition fairly smooth. Table 13-4 provides a comparison between WebSphere Community Edition 2.0 and Apache Tomcat 6.0

Table 13-4 Comparison chart

Feature	WebSphere Community Edition 2.0	Apache Tomcat 6.0
Java EE certification	5	
Servlet	2.5	2.5
JSP	2.1	2.1
JSF	1.2	1.2
EJB	3.0	
Java Persistence API	1.0 (OpenJPA)	
JMS	1.1	
Web Service	2.0	

The Tomcat servlet, within Community Edition, does not include the server.xml or context.xml files; instead, these parameters are included in the GBean definition for Tomcat in the config.xml file and the application geronimo-web.xml configuration file.

In this section, we describe the points to consider while migrating applications from Tomcat to Community Edition:

- Servlets, JSPs and JSF pages

  Because Community Edition has Tomcat as its Web container, it is not necessary for you to migrate any of these artifacts. You get the same behavior in Community Edition.

- WEB-INF/lib JAR files

  For Tomcat applications that require features, such as JMS, Web services, logging, and so on, the developer includes (and has to integrate and support) JAR files for these classpaths in the WEB-INF/lib folder. Compare JAR files in this folder (for example, axis2, common, jax, log4j) with JAR files within the *wasceHome*/repository (versions might be different) and remove duplicate JAR files. If the duplicate JAR file is required within the WEB-INF/lib due to version or custom coding, use the <hidden-classes> parameter within the configuration file to configure the classloader to use the application JAR file.

- Tomcat connectors

  Connectors enable Tomcat to communicate with different protocols. The connectors that are supported in Tomcat are available in Community Edition also. You can configure connectors using the administrative console or by editing config.xml.

  See 5.4.2, “Configuring a connector” on page 54 for more information about how to configure Tomcat connectors in Community Edition.

- Security realms

  Security realms are collections of users and groups that are used for authenticating requests to the server. Security realms are configured through the server.xml file in Tomcat. See 5.5, “Configuring application security” on page 58 for information about the supported security realms and how to configure them to use in Community Edition.

  After you configure the security realm, you should specify that realm in the Web deployment plan (geronimo-web.xml) of your Web application. See 10.6.3, “Security configuration” on page 151 for more information about how to configure security in Community Edition.

- JNDI

  In Tomcat, you can configure JNDI contexts in server.xml or by using application-specific META-INF/context.xml.

  Applications can access the GBeans or resources that Community Edition manages. The resources include JMS resources and data sources. If these resources were made available to applications through JNDI in Tomcat, you can do the same in Community Edition.

  See 7.5, “How to use JMS resources in an application” on page 92 for information about how to access JMS resources from an application.

  See Chapter 8, “Using databases” on page 97 for information about how to access databases from an application.

  If you were exposing resources other than JMS or database resources through JNDI, you can do the same by writing GBeans to act as a wrapper around your resources. First, check for any plug-ins that are already available for exposing such resources as part of JNDI in Community Edition.

  For information about GBeans, see:

  http://cwiki.apache.org/GMOxDEV/gbeansarticle1.html#GBeansArticle1-SampleCode

- Virtual hosts

  Using virtual hosting you can deploy applications in more than one domain in a single server. Community Edition supports virtual hosting.

  See the 5.4.3, “Configuring virtual hosts” on page 55 for information about how to configure virtual hosts in Community Edition.

  See 10.6.5, “Web module deployment plan” on page 157 for information about how to specify virtual hosts in your Web application.

- **Valves**

  Valves are components that you can insert into the request processing chain of Tomcat. Valves are supported in Community Edition.

  See 5.4.4, “Valves management” on page 56 for information about how to configure valves in Community Edition.

13.4 Migrating from other application servers to Community Edition

Applications that are written according to the Java EE 5 or J2EE 1.4 standard should be easy to migrate to Community Edition. Migration will consist primarily of creating deployment plans for Community Edition. If features specific to the current application server are being used, you must find similar features in Community Edition.

The following Web site gives you some general information about migrating your application to Community Edition:

http://publib.boulder.ibm.com/wasce/V2.0.0/en/migrating-applications.html

13.5 Summary of references

- WebSphere Application Server Community Edition V1 documenation

  http://publib.boulder.ibm.com/wasce/V1.1.0/en/index.html

  - Deployment plans

    http://publib.boulder.ibm.com/wasce/V1.1.0/en/Reference/Plans/index.html

  - Documentation for openejb-jar-2.1

    http://publib.boulder.ibm.com/wasce/V1.1.0/en/Reference/Plans/openejb-jar-2.1.xsd.html

  - Documentation for geronimo-web-1.1

    http://publib.boulder.ibm.com/wasce/V1.1.0/en/Reference/Plans/geronimo-web-1.1.xsd.html

  - Documentation for geronimo-application-1.1

    http://publib.boulder.ibm.com/wasce/V1.1.0/en/Reference/Plans/geronimo-application-1.1.xsd.html

- WebSphere Application Server Community Edition V2.0 documentation

  http://publib.boulder.ibm.com/wasce/V2.0.0/en/index.html

- Developing deployment plans

  http://publib.boulder.ibm.com/wasce/V2.0.0/en/developing-deployment-plans.html
- Developing a deployment plan for a Web ARchive (WAR)

  http://publib.boulder.ibm.com/wasce/V2.0.0/en/war.html
- Developing a deployment plan for an Enterprise Java Bean (EJB) archive

  http://publib.boulder.ibm.com/wasce/V2.0.0/en/ejb.html
- Developing a deployment plan for an Enterprise ARchive (EAR)

  http://publib.boulder.ibm.com/wasce/V2.0.0/en/ear.html
- Migrating from other application servers

  http://publib.boulder.ibm.com/wasce/V2.0.0/en/migrating-applications.html

- What is new in WebSphere Application Server Community Edition V2.0

  http://www.ibm.com/developerworks/websphere/library/techarticles/0709_jain/0709_jain.html
- JSR 244: Java Platform, Enterprise Edition 5 (Java EE 5) Specification

  http://jcp.org/en/jsr/detail?id=244
- Apache Geronimo v1.1 XML Schemas

  http://geronimo.apache.org/apache-geronimo-v11-xml-schemas.html
- Apache Geronimo v2.0 XML Schemas

  http://geronimo.apache.org/apache-geronimo-v20-xml-schemas.html
- Apache Geronimo V1.1 Documentation

  http://cwiki.apache.org/GMOxDOC11/documentation.html
  - geronimo-web.xml

    http://cwiki.apache.org/GMOxDOC11/geronimo-webxml.html
  - openejb-jar.xml

    http://cwiki.apache.org/GMOxDOC11/openejb-jarxml.html
  - geronimo-application.xml

    http://cwiki.apache.org/GMOxDOC11/geronimo-applicationxml.html
  - Using J2G: Components

    http://cwiki.apache.org/GMOxDOC11/using-j2g.html#UsingJ2G-Components
- Apache Geronimo V2.0 Documentation

  http://cwiki.apache.org/GMOxDOC20/documentation.html
  - Migrating to Apache Geronimo

    http://cwiki.apache.org/GMOxDOC20/migrating-to-apache-geronimo.html
- Index of /dist/geronimo/j2g/nightly builds

  http://people.apache.org/dist/geronimo/j2g/nightly/
- Geronimo GBean Architecture: Sample code

  http://cwiki.apache.org/GMOxDEV/gbeansarticle1.html#GBeansArticle1-SampleCode

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this book.

IBM Redbooks

For information about ordering these publications, see “How to get Redbooks” on page 211. Some of the documents referenced here might be available in softcopy only.

Migrating from WebSphere Application Server Community Edition to WebSphere Application Server, SG24-7433

Online resources

These Web sites are also relevant as further information sources:

- There are many developerWorks articles on Community Edition. Geronimo articles also apply. You can search for these articles at:

  http://www.ibm.com/developerworks
- IBM WebSphere Application Server Community Edition and Apache Geronimo forum

  http://www.ibm.com/developerworks/forums/forum.jspa?forumID=541
- WebSphere Application Server Community Edition zone: Technical page with links to developerWorks articles, Webcasts, and more

  http://www.ibm.com/developerworks/websphere/zones/was/wasce.html
- WebSphere Application Server Community Edition: Resource landing page with links to downloads, support, technical articles, documentation, education opportunities, migration resources, and more

  http://www.ibm.com/websphere/wasce
- WebSphere Application Server Community Edition V2.0 documentation

  http://publib.boulder.ibm.com/wasce/V2.0.0/en/index.html
- Chariot Solutions: Geronimo book by Aaron Mulder (not updated for 2.0)

  http://chariotsolutions.com/geronimo/
- Global WebSphere Community

  http://www.websphere.org/websphere/Site?page=home
- Sun Developer Network: Java EE Technologies at a Glance

  http://java.sun.com/javaee/technologies/
- Geronimo Web site

  http://geronimo.apache.org/
- Apache Geronimo v2.0 documentation

  http://cwiki.apache.org/GMOxDOC20/documentation.html

- Apache Geronim0 v1.0 documentation

  http://cwiki.apache.org/GMOxDOC10/documentation.html
- Apache Geronim0 v1.1 documentation

  http://cwiki.apache.org/GMOxDOC11/documentation.html
- Apache Geronim0 v1.2 documentation

  http://cwiki.apache.org/GMOxDOC12/documentation.html
- Apache Geronimo Architecture

  http://cwiki.apache.org/GMOxDEV/geronimo-architecture.html
- Apache Geronimo v1.1 XML Schemas

  http://geronimo.apache.org/apache-geronimo-v11-xml-schemas.html
- Apache Geronimo v2.0 XML Schemas

  http://geronimo.apache.org/apache-geronimo-v20-xml-schemas.html
- Apache Geronimo Plugin Central

  http://geronimoplugincentral.org
- Geronimo Plugins

  http://geronimoplugins.com
- Geronimo: Server Info portlet doesn't display the 'Server Memory Usage' live graph on Internet Explorer

  https://issues.apache.org/jira/browse/GERONIMO-1939
- Index of /dist/geronimo/j2g/nightly builds

  http://people.apache.org/dist/geronimo/j2g/nightly/
- Geronimo GBean Architecture: Sample code

  http://cwiki.apache.org/GMOxDEV/gbeansarticle1.html#GBeansArticle1-SampleCode
- Apache Maven Project

  http://maven.apache.org/
- Index of /xml/ns/j2ee

  http://geronimo.apache.org/xml/ns/j2ee/
- Apache Tomcat 6.0 documentation

  http://tomcat.apache.org/tomcat-6.0-doc/index.html
- Apache HTTP Server Project

  http://httpd.apache.org/
- The Apache Tomcat Connector

  http://tomcat.apache.org/connectors-doc/
- Apache ActiveMQ

  http://activemq.apache.org/
- Apache ActiveMQ Stomp support

  http://activemq.apache.org/stomp.html
- WebSphere MQ Information Center

  http://publib.boulder.ibm.com/infocenter/wmqv6/v6r0/index.jsp

- MQC6: WebSphere MQ V6.0 Clients

  http://www-1.ibm.com/support/docview.wss?rs=171&uid=swg24009961&loc=en_US&cs=utf-8&lang=en

- Download: WebSphere Application Server Community Edition 2.0

  http://www.ibm.com/developerworks/downloads/ws/wasce/

- Download: WebSphere Application Server Community Edition 2.0 free trial support program

  http://www.ibm.com/developerworks/downloads/ws/wasce/trialsupport.html

- Download: WebSphere Application Server Community Edition 2.0 self-help and training resources

  http://www.ibm.com/developerworks/downloads/ws/wasce/support.html

- WebSphere Application Server Community Edition detailed system requirements

  http://www-1.ibm.com/support/docview.wss?rs=2359&uid=swg27006834

- WebSphere Application Server Community Edition support offerings

  http://www-306.ibm.com/software/webservers/appserv/community/detail/table.html

- What's new in WebSphere Application Server Community Edition V2.0

  http://www.ibm.com/developerworks/websphere/library/techarticles/0709_jain/0709_jain.html

- Advanced administration in WebSphere Application Server Community Edition: Part 1: Working with database realms and security elements

  http://www.ibm.com/developerworks/websphere/library/techarticles/0702_krishnasamy/0702_krishnasamy.html

- Client authentication using digital certificates in WebSphere Application Server Community Edition

  http://www.ibm.com/developerworks/websphere/library/techarticles/0606_chillakuru/0606_chillakuru.html

- Set up a public key infrastructure with WebSphere Application Server Community Edition V2.0

  http://www.ibm.com/developerworks/websphere/library/techarticles/0710_vamsi/0710_vamsi.html?S_TACT=105AGX10&S_CMP=WASCE

- Configuring Web application security in WebSphere Application Server Community Edition V2.0

  http://www.ibm.com/developerworks/websphere/library/techarticles/0709_vamsi/0709_vamsi.html?S_TACT=105AGX10&S_CMP=WASCE

- KL-Patterns - Configuring WASCE 2.0 as Windows Service

  http://kl-patterns.com/articles/websphereas/configuring-wasce-2.0-as-windows-service.html

- WebSphere Application Server Community Edition V2.0 Server Plug-ins

  http://download.boulder.ibm.com/ibmdl/pub/software/websphere/wasce/plugins-2.0/

- Using jconsole

  http://java.sun.com/j2se/1.5.0/docs/guide/management/jconsole.html

- Java Diagnostics Guide 5.0

  http://publib.boulder.ibm.com/infocenter/javasdk/v5r0/index.jsp

- developerWorks Java technical library for performance related articles.

  http://www.ibm.com/developerworks/views/java/libraryview.jsp
- IBM developer kits

  http://www.ibm.com/developerworks/java/jdk
- Eclipse Update site for IBM WebSphere Application Server Community Edition

  http://download.boulder.ibm.com/ibmdl/pub/software/websphere/wasce/updates/
- Web Tools Platform downloads

  http://download.eclipse.org/webtools/downloads/
- Eclipse Classic

  http://www.eclipse.org/downloads/moreinfo/classic.php
- EMF download

  http://www.eclipse.org/modeling/emf/downloads
- Graphical Editing Framework (GEF) downloads

  http://download.eclipse.org/tools/gef/downloads/index.php
- Eclipse Data Tools Platform (DTP) Project Downloads

  http://www.eclipse.org/datatools/downloads.php
- Test and Performance Tools Platform (TPTP) Downloads

  http://www.eclipse.org/tptp/home/downloads
- Installing the WASCE WTP Server Adapter: prerequisite software

  http://publib.boulder.ibm.com/wasce/V2.0.0/en/eclipse.html#EclipsePrerequisiteSoftware
- Dojo Javascript toolkit

  http://dojotoolkit.org
- WebSphere Application Server Community Edition Product support

  http://www-306.ibm.com/software/webservers/appserv/community/support/
- MustGather: Read first for all WebSphere Application Server Community Edition components

  http://www-1.ibm.com/support/docview.wss?rs=2359&context=SS6JMN&dc=DB520&uid=swg21231419&loc=en_US&cs=UTF-8&lang=en
- How to generate system core dump, Java core dump, heap dump and a snap dump when WebSphere Application Server Community Edition is running as an operating system service

  http://www-1.ibm.com/support/docview.wss?rs=2359&context=SS6JMN&dc=DB520&dc=DB560&uid=swg21259615&loc=en_US&cs=utf-8&lang=en
- MustGather: WebSphere Application Server Community Edition 100% CPU on Linux

  http://www-1.ibm.com/support/docview.wss?rs=2359&context=SS6JMN&dc=DB520&dc=DB560&uid=swg21259526&loc=en_US&cs=UTF-8&lang=en&rss=ct2359websphere
- MustGather: WebSphere Application Server Community Edition hang or performance degradation problems on Windows

  http://www-1.ibm.com/support/docview.wss?rs=2359&context=SS6JMN&dc=DB520&dc=DB560&uid=swg21264875&loc=en_US&cs=utf-8&lang=en

- MustGather: WebSphere Application Server Community Edition OutOfMemory errors on Linux

  http://www-1.ibm.com/support/docview.wss?rs=2359&context=SS6JMN&dc=DB520&dc=DB560&uid=swg21264702&loc=en_US&cs=utf-8&lang=en

- MustGather: WebSphere Application Server Community Edition OutOfMemory errors on Windows

  http://www-1.ibm.com/support/docview.wss?rs=2359&context=SS6JMN&dc=DB520&dc=DB560&uid=swg21264612&loc=en_US&cs=utf-8&lang=en

- MustGather: WebSphere Application Server Community Edition JVM Crash on Linux

  http://www-1.ibm.com/support/docview.wss?rs=2359&context=SS6JMN&dc=DB520&dc=DB560&uid=swg21246239&loc=en_US&cs=utf-8&lang=en

- MustGather: WebSphere Application Server Community Edition JVM Crash on Windows

  http://www-1.ibm.com/support/docview.wss?rs=2359&context=SS6JMN&dc=DB520&dc=DB560&uid=swg21246238&loc=en_US&cs=utf-8&lang=en

- JSR 244: Java Platform, Enterprise Edition 5 (Java EE 5) Specification

  http://jcp.org/en/jsr/detail?id=244

How to get Redbooks

You can search for, view, or download Redbooks, Redpapers, Technotes, draft publications and Additional materials, as well as order hardcopy Redbooks, at this Web site:

ibm.com/redbooks

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Index

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

WebSphere Application Server Community Edition 2.0 User Guide

(0.2”spine)
0.17”<->0.473”
90<->249 pages

WebSphere Application Server Community Edition 2.0 User Guide

Installation and migration

Configuration and administration

Application packaging and deployment

This IBM® Redbooks® publication takes you through the basics of using WebSphere® Application Server Community Edition V2 to run applications. It includes information about planning and installing the Community Edition server, configuring the environment, preparing applications for deployment, and managing deployed applications. This book also contains pointers to information that can help you with more advanced topics and to find updated information on using the Community Edition.

The target audience is primarily administrators of Community Edition, but this book is also useful for developers who are packaging and deploying applications to Community Edition.

INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

For more information:
ibm.com/redbooks

SG24-7585-00 ISBN 0738485861