WebSphere® Application Server for IBM i, Version 7.0

Migrating, coexisting, and interoperating

Note

Before using this information, be sure to read the general information under “Notices” on page 63.

Compilation date: September 9, 2008

Contents

How to send your comments

Your feedback is important in helping to provide the most accurate and highest quality information.

- To send comments on articles in the WebSphere Application Server Information Center
  1. Display the article in your Web browser and scroll to the end of the article.
  2. Click on the **Feedback** link at the bottom of the article, and a separate window containing an e-mail form appears.
  3. Fill out the e-mail form as instructed, and click on **Submit feedback** .
- To send comments on PDF books, you can e-mail your comments to: **wasdoc@us.ibm.com** or fax them to 919-254-5250.

  Be sure to include the document name and number, the WebSphere Application Server version you are using, and, if applicable, the specific page, table, or figure number on which you are commenting.

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Changes to serve you more quickly

Print sections directly from the information center navigation

PDF books are provided as a convenience format for easy printing, reading, and offline use. The information center is the official delivery format for IBM WebSphere Application Server documentation. If you use the PDF books primarily for convenient printing, it is now easier to print various parts of the information center as needed, quickly and directly from the information center navigation tree.

To print a section of the information center navigation:

1. Hover your cursor over an entry in the information center navigation until the **Open Quick Menu** icon is displayed beside the entry.
2. Right-click the icon to display a menu for printing or searching your selected section of the navigation tree.
3. If you select **Print this topic and subtopics** from the menu, the selected section is launched in a separate browser window as one HTML file. The HTML file includes each of the topics in the section, with a table of contents at the top.
4. Print the HTML file.

For performance reasons, the number of topics you can print at one time is limited. You are notified if your selection contains too many topics. If the current limit is too restrictive, use the feedback link to suggest a preferable limit. The feedback link is available at the end of most information center pages.

Under construction!

The Information Development Team for IBM WebSphere Application Server is changing its PDF book delivery strategy to respond better to user needs. The intention is to deliver the content to you in PDF format more frequently. During a temporary transition phase, you might experience broken links. During the transition phase, expect the following link behavior:

- Links to Web addresses beginning with http:// work
- Links that refer to specific page numbers within the same PDF book work
- The remaining links will *not* work. You receive an error message when you click them

Thanks for your patience, in the short term, to facilitate the transition to more frequent PDF book updates.

Chapter 1. Migration, coexistence, and interoperability

The goal of migration is to reconstruct your earlier version of WebSphere® Application Server in a Version 7.0 environment. Coexistence allows you to create a mixed-version environment that is not in conflict and allows the nodes of all versions to start and run at the same time. Coexistence also facilitates rollback and allows one or the other version to run at one time. Interoperating is exchanging data between two coexisting product installations or between products on different systems.

Overview of migration, coexistence, and interoperability

The goal of migration is to reconstruct your earlier version of WebSphere Application Server in a Version 7.0 environment. Coexistence allows you to create a mixed-version environment that is not in conflict and allows the nodes of all versions to start and run at the same time. Coexistence also facilitates rollback and allows one or the other version to run at one time. Interoperating is exchanging data between two coexisting product installations or between products on different systems.

Introduction

WebSphere Application Server Version 7.0 can coexist with Version 5.1.x and Version 6.x. Depending on the previous version of WebSphere Application Server, port conflicts might exist that must be resolved. Read “Coexistence support” on page 47 and Chapter 8, “Configuring ports,” on page 51 for more information.

WebSphere Application Server Version 7.0 migration leverages the existing configuration and applications and changes them to be compatible with the WebSphere Application Server Version 7.0 environment. Existing application components and configuration settings are applied to the Version 7.0 environment during the migration process.

If you use an earlier version of WebSphere Application Server, the system administrator might have fine-tuned various application and server settings for your environment. It is important to have a strategy for migrating these settings with maximum efficiency.

You can perform incremental migration of your WebSphere Application Server Version 5.1.x or Version 6.x configuration by running the migration tools multiple times, each time specifying a different set of instances or profiles.

Migration involves the following main steps:

1. Test your applications in a non-production WebSphere Application Server Version 7.0 environment, and make any changes to the applications that are necessary to ensure that they run in that environment.
2. Migrate those applications and your configuration to Version 7.0.

   You can complete this task by using the migration tools that are included with the product.

Use the migration tools to migrate applications and configuration information to the new version as described in Chapter 3, “Migrating product configurations,” on page 11. Read “Using the migration tools to migrate product configurations” on page 16 for more information.

Important reference articles for this migration include the following articles:

- manageprofiles command.
- Migrating from Version 5.1 embedded messaging
- Managing WebSphere Application Server Version 5 JMS use of WebSphere Application Server Version 7.0 messaging resources

If you neither migrate nor coexist with an earlier version of WebSphere Application Server, you are choosing to ignore the previous installation and you can run only one version at a time because of conflicting default port assignments. It is possible for both versions to run at the same time without conflict if you use non-default ports in one version. To set up coexistence with WebSphere Application Server Version 5.1.x or Version 6.x, ensure that unique ports are selected during profile creation for the Version 7.0 installation. To set up coexistence with an existing installation of Version 7.0, select the **Install a new copy of the V70 Application Server product** radio button during installation.

You can resolve conflicting port assignments by specifying port assignments for coexistence during profile creation, by wsadmin scripting, or by using the **Servers > Application Servers > server1 > Ports** administrative console page to ensure that WebSphere Application Server Version 7.0 can run with an earlier version. Read the ″Wsadmin tool″ article in the information center for more information on wsadmin scripting.

Coexistence processing changes the following configuration files:

- virtualhosts.xml
- serverindex.xml

Read “Port number settings in WebSphere Application Server versions” on page 51 for more information.

Consider the following issues in a migration or coexistence scenario:

- Conflicting context roots when attempting to share the same Web server.

  Follow the procedure in Chapter 4, “Migrating Web server configurations,” on page 37 to learn how to configure a Web server for sharing between WebSphere Application Server versions.

Migrating and coexisting

Migrating involves collecting the configuration information from a previous release of a WebSphere Application Server product and merging it into a configuration for a new release. Coexisting involves running a new release of a WebSphere Application Server product on the same machine at the same time as you run an earlier release.

Before you begin

Read “Overview of migration, coexistence, and interoperability” on page 1 and “Premigration considerations” on page 4. For resources to help you plan and perform your migration, visit Knowledge Collection: Migration planning for WebSphere Application Server.

The migration tools basically save the existing WebSphere configurations and user applications in a backup directory and then process the contents of this backup directory to migrate the configurations and your applications from previous WebSphere Application Server releases to the latest release.

If you have a previous version of WebSphere Application Server, you must decide whether to migrate the configuration and applications of the previous version to the new version.

Migration does not uninstall the previous version.

If you run an earlier release at the same time as the WebSphere Application Server Version 7.0 installation, the two versions are *coexisting*.

To support coexistence, you must either use the -portBlock and -replacePorts options when you migrate a profile or you must resolve port conflicts manually so that the two releases do not attempt to use the same ports. Any ports bound when the first profile starts will prevent the second profile from starting because the port is in use. No port changes are required if only one release of the profile is active at any given time.

For help in troubleshooting problems when migrating, read Chapter 9, “Troubleshooting migration,” on page 57.

About this task

For information on migrating to Version 7.0, read Chapter 3, “Migrating product configurations,” on page 11. For more information on coexistence among releases, read “Coexistence support” on page 47.

1. Update product prerequisites and corequisites to supported versions.

   Refer to the IBM® WebSphere Application Server supported hardware, software, and APIs site for current requirements.
2. Install the WebSphere Application Server Version 7.0 product.

   Read the ″Task overview: installing″ article in the information center for more information.
3. Migrate your WebSphere Application Server Version 5.1.x or Version 6.x product configuration to Version 7.0.

   You have the choice between migrating your configuration automatically using the migration tools or manually.
   - Use the migration tools to automatically migrate your configuration.

     Read “Using the migration tools to migrate product configurations” on page 16 for more information.

     Consider the following points related to using the migration tools to automatically migrate your configuration:
     - Advantages
       - You copy the old configuration automatically.

         This includes all resource definitions, virtual host definitions, security settings, and so forth.
       - You recreate the same exact Version 5.1.x or Version 6.x configuration in Version 7.0, including the server definitions and deployed applications by default.
       - You can enable support for script compatibility.

         Read “WASPostUpgrade command” on page 23 for more information.
     - Consideration

       This approach will copy the configuration exactly. A different approach, such as the manual approach, is required if your existing configuration is different from that which you want for the new configuration.
   - Manually migrate your configuration.

     Migrating your configuration manually involves the following activities:
     - You start with a clean slate and build up a new environment for Version 7.
     - Ideally, you would use an existing set of administration scripts to set up the complete Version 7.0 environment.
     - You move your applications to Version 7.0 as they are tested on Version 7.

     Consider the following points related to manually migrating your configuration:
     - Advantages
       - You can reuse the scripts for maintenance, replication, and disaster recovery.
       - You can easily refactor the topology if you desire.
     - Considerations
       - A complete set of administration scripts is a significant investment.
       - You must address script incompatibilities and changes before you migrate.
4. Migrate Web server plug-ins as described in Chapter 4, “Migrating Web server configurations,” on page 37.
5. Optional: Set up multiple versions of WebSphere Application Server to coexist.

No runtime conflicts can exist for multiple instances and versions of WebSphere Application Server if they are going to run at the same time on the same machine. Potential conflicts can occur with your port assignments. Read “Port number settings in WebSphere Application Server versions” on page 51 for more information.

Premigration considerations

Before you begin the process of migrating to WebSphere Application Server Version 7.0, there are some considerations of which you need to be aware.

- Before you perform the migration, evaluate the items deprecated in WebSphere Application Server Version 7.0.

  For more information, read the ″Deprecated and removed features″ article in the information center.
- Before you migrate to Java™ Standard Edition (SE) Development Kit (JDK) 6 from JDK 5 or JDK 1.4 , review your applications for necessary changes based on the Sun Microsystems Java specification.

  Read “API and specification migration” on page 6 for more information.
- Websphere Application Server profiles from previous releases that are configured to use the i5/OS® Java Developer Kit Java Virtual Machine (JVM), also known as the ″classic″ JVM, generally exhibit better application performance when they are migrated to a Websphere Application Server Version 7.0 profile that is configured to use the IBM Java SE 6 32 bit JVM (which is the default for Websphere Application Server Version 7.0). This is due to the reduced size of Java reference objects (4 byte versus 8 byte) and the smaller Java heap storage requirements.

  However, large applications might require a larger Java heap than the maximum heap size allowed by the IBM Java SE 6 32 bit JVM and might not run efficiently or at all. In addition, if the source profile has explicit settings for the initial Java heap size or Java maximum heap size in the application server’s genericJvmArguments that are larger then the allowed maximum Java heap size of the IBM Java SE 6 32 bit JVM (2.5 G), the target profile’s application server will not start. You can resolve these types of Java heap related problems using one of the following two methods:
  – Use the administrative console to alter or remove the explicit Java heap settings (-Xms or -Xmx java arguments) from the genericJvmArguments of the source application server before migration or of the target application server after migration, and continue to use the IBM Java SE 6 32 bit JVM.
  – Use the enableJvm script to change the JVM configuration of the target profile to use either the classic i5/OS Java Developer Kit 6.0 JVM or the IBM Java SE 6 64 bit JVM. Neither of these 64-bit JVMs limit the maximum heap size.
- The migration articles in this information center assume that WebSphere Application Server Version 7.0 is being installed in an environment where it must coexist with prior levels of WebSphere Application Server.

  Consider the following items when planning to enable coexistence:
  – Update prerequisites to the levels required by WebSphere Application Server Version 7.0.

    Prior levels of WebSphere Application Server continue to run at the higher prerequisite levels.
  – Review the ports that have been defined to ensure that the WebSphere Application Server Version 7.0 installation does not conflict.

    Read “Port number settings in WebSphere Application Server versions” on page 51 for default port information.

  Read Chapter 6, “Coexisting,” on page 47 for more information.
- WebSphere Application Server Version 7.0 migration converts HTTP transports to channel-framework Web container transport chains.

  For more information on WebSphere Application Server Version 7.0 transport support, read the following articles in the information center:
  – Configuring transport chains
  – HTTP transport channel settings
  – Transport chains

- If you create a profile that does not meet the migration requirements such as naming requirements, you can remove the previous profile and create a new one rather than uninstalling and reinstalling the WebSphere Application Server Version 7.0 product.
- The migration tools create a migration backup directory containing a backup copy of the configuration from the previous version. The following guidelines might help you to determine the amount of file-system space that this directory might require:
  - If you are migrating from Version 5.1.x, the space available for this directory must be at least the size of the configuration directory and applications from the previous version.
  - If you are migrating from Version 6.x, the space available for this directory must be at least the size of the configuration directory and applications from the previous profile.
- If you use the migration tools to create more than one Version 7.0 target profile on the same host or installation instance and you use the default port settings, there is a chance that the target profiles will share the same ports for some of the new Version 7.0 port definitions. This will cause startup problems if both of the migrated profiles are used.

  If you are migrating two or more profiles that reside on the same host or installation instance, perform the following actions for each additional target profile:

  1. Before using the migration tools, use the Profile Management tool or manageprofiles command to create the target profile and make sure that you select unique ports rather than using the default ports.
  2. When you use the migration tools, select the target profile rather than letting the tools create it.
- The amount of storage that your system requires during migration to Version 7.0 depends on your environment as well as on the migration tool that you are using.
  - **WASPreUpgrade storage requirements**
    - **Location:** Backup directory specified as a parameter of the WASPreUpgrade command
    - **Amount:** For an estimate of your storage requirements when using this command, add the following amounts.
      - Size of the following items for the previous profile or instance that you are migrating:
        - *profile_root*/installableApps directory
        - *profile_root*/installedApps directory
        - *profile_root*/config directory
        - *profile_root*/properties directory
        - Shared libraries referenced in the libraries.xml configuration files
        - Resource adapter archive (RAR) files referenced in the resources.xml configuration files
      - If trace is enabled, which is the default, up to 200 MB (depending on the size and complexity of your configuration)

    For more information about this command, read “WASPreUpgrade command” on page 20.
  - **WASPostUpgrade storage requirements**
    - For an estimate of your storage requirements when using this command, add the following amounts.
      - **Location:** New configuration relative to the new *profile_root* directory

        Size of the following items for the previous profile or instance that you are migrating:
        - *profile_root*/installableApps directory
        - *profile_root*/installedApps directory
        - *profile_root*/config directory
        - *profile_root*/properties directory
        - Shared libraries referenced in the libraries.xml configuration files
        - RAR files referenced in the resources.xml configuration files

  - **Location:** Backup directory specified as a parameter of the WASPreUpgrade and WASPostUpgrade commands

    If trace is enabled, which is the default, up to 1 GB (depending on the size and complexity of your configuration)

  For more information about this command, read “WASPostUpgrade command” on page 23.

- If you use isolated data repositories—specifically, nonshared data repositories such as transaction logs for SIB and IBM Cloudscape or Apache Derby databases—and you migrate from a previous release, your existing databases and transaction logs are saved when the WASPreUpgrade tool is run. Any database changes that you make after the WASPreUpgrade tool is run will not be reflected in the migrated environment.
  - If you have mission-critical information that is stored in these local data repositories, you should safely shut down all servers that interact with those repositories before attempting migration. Those servers should remain offline until the migration has been successfully completed or rolled back.
  - If you make multiple attempts at migration, either because of unexpected rollback or to apply fixes, rerun the WASPreUpgrade tool so that any changes to your isolated data repositories are reflected in the migrated environment.

  After the migration is complete or you have rolled back to the previous version, you can restart the servers that interact with these isolated data repositories.

- Before you migrate an Apache Derby database, ensure that any application servers hosting applications that are using the Apache Derby database are closed. Otherwise, the Apache Derby migration fails.
- After you use the migration tools to migrate to WebSphere Application Server Version 7.0, you might need to perform some actions that are not done automatically by the migration tools.
  - Examine any Lightweight Third-Party Authentication (LTPA) security settings that you might have used in WebSphere Application Server Version 5.1.x or Version 6.x, and verify that Version 7.0 security is set appropriately.

    Read the ″Lightweight Third Party Authentication″ article in the information center for more information.
  - Check the WASPostUpgrade.log file in the logs directory for details about any JavaServer Pages (JSP) objects that the migration tools did not migrate.

    If Version 7.0 does not support a level for which JSP objects are configured, the migration tools recognize the objects in the output and log them.
  - Verify the results of the automatic Apache Derby database migration, and manually migrate any Apache Derby databases that are not automatically migrated by the tools.

    Read “Migrating IBM Cloudscape or Apache Derby databases” on page 30 for more information.
  - WebSphere Application Server Version 7.0 does not include the WebSphere Connect JDBC driver for SQL Server. The WebSphereConnectJDBCDriverConversion tool is provided to convert data sources from the WebSphere Connect JDBC driver to the DataDirect Connect JDBC driver or the Microsoft® SQL Server 2005 JDBC driver.

    Read “Migrating from the WebSphere Connect JDBC driver” on page 32 for more information.

API and specification migration

Migrating application programming interfaces (APIs) and specifications involves moving to the current Java component level as well as to other technologies that WebSphere Application Server Version 7.0 supports. If your existing applications currently support different specification levels than are supported by this version of the product, it is likely that you must update at least some aspects of the applications to comply with the new specifications.

In many cases, IBM provides additional features and customization options that extend the specification level even further. If your existing applications use IBM extensions from earlier product versions, it might be necessary for you to perform mandatory or optional migration to use the same kinds of extensions in Version 7.0.

Note:

- WebSphere Application Server began supporting Java SE Development Kit (JDK) 6 in Version 7.0.

  Read JSR 270: Java SE 6 Release Contents and Java SE 6 for more information about JDK 6.
- In general, existing Version 5.1.x and 6.x application binaries that were developed using JDK 1.4 and 5 are highly compatible and typically do not require modifications to run. However, recompilation of the JDK 1.4 or 5 applications at the JDK 6 level might necessitate modifications of the source code to conform to incompatible changes that are present in JDK 6. As part of your migration planning, you should review the JDK compatibility restrictions that are documented by Sun Microsystems at Java SE 6 Release Notes®: Compatibility.
- A mixed cell containing Version 5.1.x or Version 6.x and Version 7.0 nodes requires that all application binaries deployed on Version 5.1.x or Version 6.x remain at the lowest JDK level associated with the Version 5.1.x or Version 6.x nodes. Although you can successfully migrate Version 5.1.x or Version 6.x applications to Version 7.0, this is only meant to be a temporary state as you transition to Version 7.0. After you begin migration to Version 7.0, plan to complete the migration of the entire cell, update your tooling to Version 7.0, and update your applications to conform to JDK 6 requirements. Complete this action before any further application changes. After you have completely migrated your cell to Version 7.0, upgrade your application binaries to the JDK 6 level the next time that you make application modifications that require recompiling. This action might require source code changes to your application to conform to the JDK 6 API changes as documented by Sun Microsystems.
- The Java Virtual Machine Debug Interface (JVMDI) and the Java Virtual Machine Profiler Interface (JVMPI) were deprecated in JDK 5 and removed in JDK 6.

  Read Java SE 6 Release Deprecated API for more information.

Read the ″Specifications and API documentation″ article in the information center for a summary of the specifications and API documentation supported in current and prior product releases.

For more information on the items deprecated in WebSphere Application Server Version 7.0, read the ″Deprecated and removed features″ article in the information center.

Programming model extension migration

This product edition contains several programming model extensions (PMEs) that previously were available only by obtaining a different product edition.

This article describes the movement of a subset of PMEs to WebSphere Application Server Version 7.0 from the following products:

- WebSphere Business Integration Server Foundation Version 5.1.x
- WebSphere Process Server for Multiplatforms Version 6

Overview of PME migration

The migration of PME services to WebSphere Application Server Version 7.0 is handled on an individual basis. For PME services that are not supported in WebSphere Application Server Version 7.0, all configuration information is removed. For PME services that are supported in WebSphere Application Server Version 7.0, the configuration from the previous release environment overwrites the values in the new release.

Validating PMEs

As part of application installation, both during migration and outside of migration, applications are validated to ensure that they only use resources for services that are supported by WebSphere Application Server Version 7.0. Any application that uses a resource for a service that is not supported by WebSphere Application Server Version 7.0 will not work correctly, and an error will be issued indicating that the application cannot be installed.

Chapter 2. How do I migrate, coexist, and interoperate?

Use the documentation provided to answer your questions about migration, coexistence, and interoperability.

Follow these shortcuts to get started quickly with popular tasks. When you visit a task, look for the **IBM Suggests** feature at the bottom of the page. Use it to find available tutorials, demonstrations, presentations, articles, IBM Redbooks, support documents, and more.

“Overview of migration, coexistence, and interoperability” on page 1

“Configuration mapping during product-configuration migration” on page 11

“Using the migration tools to migrate product configurations” on page 16

Chapter 3, “Migrating product configurations,” on page 11

Chapter 4, “Migrating Web server configurations,” on page 37

Review the software and hardware prerequisites

Chapter 6, “Coexisting,” on page 47

Chapter 8, “Configuring ports,” on page 51

Chapter 7, “Interoperating,” on page 49

Chapter 9, “Troubleshooting migration,” on page 57

Chapter 3. Migrating product configurations

Use the WebSphere Application Server Version 7.0 migration tools to migrate your product configurations. These migration tools support migration from Version 5.1.x and Version 6.x.

Before you begin

Read “Overview of migration, coexistence, and interoperability” on page 1 and “Premigration considerations” on page 4. For resources to help you plan and perform your migration, visit Knowledge Collection: Migration planning for WebSphere Application Server.

The following configuration upgrades of WebSphere Application Server versions and offerings are directly supported.

Table 1. Directly Supported Configuration Upgrades

Migration Source	WebSphere Application Server Version 7.0 Target
	Base Standalone Profile
WebSphere Application Server Version 5.1.x or Version 6.x base standalone application server	Supported
WebSphere Application Server Version 5.1.x or Version 6.x Network Deployment standalone application server	Supported
WebSphere Application Server Version 5.1.x or Version 6.x Network Deployment federated application server	Supported
WebSphere Application Server Version 5.1.x or Version 6.x Network Deployment deployment manager	Not supported
WebSphere Application Server Version 5.1.x or Version 6.x Express standalone application server	Supported

You can migrate your product configurations using the WebSphere Application Server Version 7.0 command-line migration tools.

Before using the migration tools, consult the IBM WebSphere Application Server supported hardware, software, and APIs Web site to understand what fixes you must apply to earlier versions. Applying fixes to an earlier version might also apply fixes to files that have a role in the migration. Apply any fixes to ensure the most effective migration of configurations and applications.

Configuration mapping during product-configuration migration

Various configurations are mapped during product-configuration migration.

Migration always involves migrating a single profile to another single profile on the same machine or a separate machine. Go to the information center for the WebSphere Application Server Network Deployment product to learn how the migration tools map models, clones, server groups, clusters, and Lightweight Third Party Authentication (LTPA) security settings.

Many migration scenarios are possible. The migration tools map objects and attributes existing in the version from which you are migrating to the corresponding objects and attributes in the Version 7.0 environment.

Bootstrap port

The migration tools carry the old release value into the Version 7.0 environment.

If a value for the -portBlock parameter is specified during the call to WASPostUpgrade, however, a new port value is given to each application server that is migrated to Version 7.0.

Command-line parameters

The migration tools convert appropriate command-line parameters to Java Virtual Machine (JVM) settings in the server process definition. Most settings are mapped directly. Some settings are not migrated because their roles in the WebSphere Application Server Version 7.0 configuration do not exist, have different meanings, or have different scopes.

For information on how to change the process-definition settings, read the ″Process definition settings″ article in the information center. For information on how to change the JVM settings, read the ″Java virtual machine settings″ article in the information center.

Generic server

In WebSphere Application Server Version 5.1.x, a generic server was an APPLICATION_SERVER fitted to manage external resources. In Version 6.x and later, it has its own type called GENERIC_SERVER. Migration will perform this conversion, but migration cannot accurately migrate the external resources that the generic server references. After migration has completed migrating the generic server settings, you might need to perform additional tasks. If the old resource that the generic server was managing is located under the old WebSphere Application Server installation, perform the following tasks:

1. Copy any related files to the new installation.
2. Run any setup required to put the external application back into a valid and working state.

   It is best that you reinstall the resource into the new WebSphere Application Server directory. Whatever you choose to do, the final step is to reset the reference to the new location of the application.

If the old resource that the generic server was managing is not installed under the old WebSphere Application Server installation, nothing further is required.

Policy files

WebSphere Application Server Version 7.0 migrates all the policy files that are installed with Version 5.1.x or Version 6.x by merging settings into the Version 7.0 policy files with the following characteristics:

- Any comments located in the Version 7.0 policy files will be preserved. Any comments contained in the Version 5.1.x or Version 6.x policy files will not be included in the Version 7.0 file.
- Migration will not attempt to merge permissions or grants; it is strictly an add-type migration. If the permission or grant is not located in the Version 7.0 file, the migration will bring it over.
- Security is a critical component; thus, the migration makes any additions at the end of the original .policy files right after the comment `MIGR0372I: Migrated grant permissions follow`. This is done to help administrators verify any policy-file changes that the migration has made.

Properties and classes directories

Migration copies files from prior version directories into the WebSphere Application Server Version 7.0 configuration.

Property files

WebSphere Application Server Version 7.0 migrates all the property files that are installed with Version 5.1.x or Version 6.x by merging settings into the Version 7.0 property files with these exceptions for Version 5.1.x files:

- j2c.properties (migrated into resources.xml files)
- samples.properties

Resource adapter archives (RARs) referenced by J2C resources

RARs that are referenced by J2C resources are migrated if those RARs are in the old WebSphere Application Server installation. In this case, the RARs are copied over to the corresponding location in the new WebSphere Application Server installation. Relational Resource Adapter RARs will not be migrated.

Samples

No migration of samples from previous versions is available. There are equivalent WebSphere Application Server Version 7.0 samples that you can install.

Security

Java 2 security is enabled by default when you enable security in WebSphere Application Server Version 7.0. Java 2 security requires you to grant security permissions explicitly.

There are several techniques that you can use to define different levels of Java 2 security in Version 7.0. One is to create a was.policy file as part of the application to enable all security permissions. The migration tools call the wsadmin command to add an existing was.policy file in the Version 7.0 properties directory to enterprise applications as they are being migrated.

When migrating to WebSphere Application Server Version 7.0, your choice of whether or not to migrate to support script compatibility results in one of two different outcomes.

- If you choose to migrate to support script compatibility, your security configuration is brought over to Version 7.0 without any changes.

  This is the default.
- If you choose not to migrate to support script compatibility, the security configuration is converted to the default configuration for WebSphere Application Server Version 7.0. The default security configuration for Version 6.1 and later acts almost the same as in the previous versions, but there are some changes.

  For example, existing keyfiles and trustfiles are moved out of the SSLConfig repertoire and new keystore and truststore objects are created.

For more information on migrating your security configurations to Version 7.0, read the ″Migrating, coexisting, and interoperating – Security considerations″ article in the information center.

Stdin, stdout, stderr, passivation, and working directories

The location for these directories is typically the logs directory under the WebSphere Application Server profile directory. For WebSphere Application Server Version 7.0, the default location for the stdin, stdout, and stderr files is the logs directory located under the WebSphere Application Server profile directory—for example, the logs directory for the default profile is /QIBM/UserData/WebSphere/AppServer/V70/Base/profiles/default/logs.

The migration tools attempt to migrate existing passivation and working directories. Otherwise, appropriate Version 7.0 defaults are used.

In a coexistence scenario, using common directories between versions can create problems.

Transport ports

The migration tools migrate all ports. You must resolve any port conflicts before you can run servers at the same time.

Note: If ports are already defined in a configuration being migrated, the migration tools fix the port conflicts in the Version 7.0 configuration and log the changes for your verification.

If you specify the -portBlock parameter in the WASPostUpgrade command, a new value is assigned to each transport that is migrated.

If you specify true for the -replacePorts parameter in the WASPostUpgrade command, all port values from the old configuration are used in the new configuration. If you specify false for the

-replacePorts parameter, the default port definitions in the new profile are not replaced with the values from the old configuration during migration. .

Choosing -scriptCompatibility=″true″ or -scriptCompatibility=″false″ results in two different outcomes for transport ports if you are migrating from WebSphere Application Server Version 5.1.x:

- -scriptCompatibility=″true″

  This results in your transport ports being brought over as they are. This is the default.
- -scriptCompatibility=″false″

  This results in the transport ports being converted to the implementation of channels and chains. From an external application usage standpoint, they will still act the same; but they have been moved to the TransportChannelService.

For more information on the WASPostUpgrade command, read "WASPostUpgrade command" on page 23.

For further information on transport chains and channels, read the 'Transport chains' article in the information center.

You must manually add virtual host alias entries for each port. For more information, read the ″Configuring virtual hosts″ article in the information center.

Web modules

The specification level of the Java Platform, Enterprise Edition (Java EE) implemented in WebSphere Application Server Version 6.0.x required behavior changes in the Web container for setting the content type. If a default servlet writer does not set the content type, not only does the Web container no longer default to it but the Web container returns the call as ″null.″ This situation might cause some browsers to display resulting Web container tags incorrectly. To prevent this problem from occurring, migration sets the autoResponseEncoding IBM extension to ″true″ for Web modules as it migrates enterprise applications.

JVM system properties

If you migrate a Version 6.1 configuration that has feature packs installed, the migration tools might add one or two JVM system properties for each Java server in your configuration, including your administrative servers. Web servers are not affected. The properties are set to indicate to the JVM that the configuration should use a Java annotation scan policy other than the Version 7.0 default scan policy.

- If you migrate a Version 6.1 profile that has the Feature Pack for EJB 3.0 installed, the migration tools add the following system property to the JVM definitions for all Java servers defined on that node:

  ```
  com.ibm.websphere.ejb.UseEJB61FEPScanPolicy = true
  ```
- If you migrate a Version 6.1 profile that has the Feature Pack for Web Services installed, the migration tools add the following system property to the JVM definitions for all Java servers defined on that node:

  ```
  com.ibm.websphere.webservices.UseWSFEP61ScanPolicy = true
  ```
- If you migrate a Version 6.1 profile that has both the Feature Pack for EJB 3.0 and the Feature Pack for Web Services installed, the migration tools add both of the system properties to the JVM definitions for all Java servers defined on that node:

  ```
  com.ibm.websphere.ejb.UseEJB61FEPScanPolicy = true
  com.ibm.websphere.webservices.UseWSFEP61ScanPolicy = true
  ```

If these properties are set, the following two changes take place in the default Version 7.0 behavior:

- Application installation generates classes based on the annotation scan policy associated with the settings for those two properties.

  This means that you can potentially use the following four annotation scan policies:

  - Version 7 default behavior
  - Feature Pack for EJB 3.0 behavior
  - Feature Pack for Web Services behavior
  - Net behavior from having both the Feature Pack for EJB 3.0 and the Feature Pack for Web Services installed
- The servers use the generated annotation classes based on the properties set, resulting in four potential behaviors.

You can change the scan policy behavior by adding or removing the custom JVM system properties from your server.xml files. After changing the properties, you must reinstall or update your applications and then resynchronize the cell to implement the change.

Preparing for product-configuration migration

You must prepare your system for migrating to WebSphere Application Server Version 7.0.

Before you begin

Read "Overview of migration, coexistence, and interoperability" on page 1 and "Premigration considerations" on page 4.

For help, read Chapter 9, "Troubleshooting migration," on page 57.

1. Install WebSphere Application Server Version 7.0.
2. Familiarize yourself with the tools and features of WebSphere Application Server Version 7.0.
3. Verify that you have the minimum prerequisites required for migration.

   For more information, read "Checking for the product-configuration migration prerequisites."
4. Evaluate the items deprecated in WebSphere Application Server Version 7.

   For more information, read the ″Deprecated and removed features″ article in the information center.
5. Evaluate the changes to API specification levels to determine which applications you need to migrate.

   To plan your application migration requirements, use "API and specification migration" on page 6.
6. Evaluate the changes to configuration settings.

   For more information, read "Configuration mapping during product-configuration migration" on page 11.

Checking for the product-configuration migration prerequisites

Before you migrate your old version of WebSphere Application Server to Version 7.0, you must make sure that you meet certain requirements.

- Make sure that you have the minimum version of source WebSphere Application Server that is required.
  - If you are migrating from WebSphere Application Server Version 5.1.x, you must be at Version 5.1 or higher.
  - If you are migrating from WebSphere Application Server 6.x, you must be at Version 6.0 or higher.

  To determine the current level of WebSphere Application Server installed on your system, perform these steps:

  Version 5.1.x:

  1. Enter the following command on the command line:

     ```
     WRKLNK '/QIBM/ProdData/product_dir/properties/version/product_file
     ```

     where *product_dir* is:
     - `WebASE51/ASE` for Express Version 5.1.x
     - `WebAS51/Base` for base Version 5.1.x
     - `WebAS51/ND` for Network Deployment Version 5.1.x

and *product_file* is:

- `Express.product` for Express Version 5.1.x
- `BASE.product` for base Version 5.1.x
- `ND.product` for Network Deployment Version 5.1.x

2. Specify option 5 (Display) next to the product file to view the contents. The number within the <version> tags shows the current version that you have installed.

Version 6.x:

1. Enter the following command on the command line:

```
WRKLNK '/QIBM/ProdData/WebSphere/AppServer/V6/product_dir/properties/version/WAS.product
```

   where *product_dir* is:

   - `Base` for base and Express Version 6.x
   - `ND` for Network Deployment Version 6.x

2. Specify option 5 (Display) next to the product file to view the contents. The number within the <version> tags shows the current version that you have installed.

If you do not meet the minimum version, obtain the latest group PTF. Read WebSphere Application Server PTFs for iSeries® for information on the correct group PTF for your operating-system release level and WebSphere Application Server Version 5.1.x or Version 6.x product.

- Make sure that WebSphere Application Server Version 7.0 is installed.

  WebSphere Application Server Version 7.0 Network Deployment needs to be installed if you are migrating to Network Deployment.

- Make sure that your user profile has *ALLOBJ authority.

  When you call the WASPreUpgrade and WASPostUpgrade migration tools, your user profile must have *ALLOBJ authority.

Using the migration tools to migrate product configurations

Migration support consists of tools that included with WebSphere Application Server Version 7.0. These tools primarily provide support for saving the configuration and applications from a previous version of the product into a migration-specific backup directory and then importing that configuration into Version 7.0.

Before you begin

Read “Overview of migration, coexistence, and interoperability” on page 1 and “Premigration considerations” on page 4. For resources to help you plan and perform your migration, visit Knowledge Collection: Migration planning for WebSphere Application Server.

Note: Use the migration tools for the version of WebSphere Application Server that you are installing. The tools change over time. If you use migration tools from an earlier release of WebSphere Application Server, you are likely to encounter a problem with the migration.

The migration scripts are located in the *app_server_root*/bin directory after installation.

Select the appropriate migration tools to migrate your product configurations.

WASPreUpgrade tool

You use the WASPostUpgrade tool to save the applications and configuration data from a previous installation of WebSphere Application Server to a backup directory.

The WASPostUpgrade tool restores the configuration data from the directory to the new installation.

Read “WASPreUpgrade command” on page 20 for more information.

WASPostUpgrade tool

You use the WASPostUpgrade tool to restore the configuration data from a previous release.

The WASPostUpgrade tool reads the data from the backup directory where the WASPreUpgrade tool stored the data.

Read “WASPostUpgrade command” on page 23 for more information.

clientUpgrade tool

You can use the clientUpgrade tool to upgrade the client application to a new release level.

Read “clientUpgrade script” for more information.

convertScriptCompatibility tool

Administrators use the convertScriptCompatibility tool to convert their configuration from a mode that supports backward compatibility of Version 5.1.x or Version 6.x administration scripts to a mode that is fully Version 7.0.

Read “convertScriptCompatibility command” on page 18 for more information.

convertSelfSignedCertificatesToChained task

Chained certificates are the default certificate type in Websphere Application Server Version 7.0. Administrators can use the convertSelfSignedCertificatesToChained task with the wsadmin tool to convert self-signed certificates to chained certificates.

Read the ″SSLMigrationCommands command group for the AdminTask object″ article for more information.

Note: For help, read Chapter 9, “Troubleshooting migration,” on page 57.

What to do next

Use the selected tools to migrate your product configuration.

clientUpgrade script

The clientUpgrade script migrates application client modules and their resources in an enterprise archive (EAR) file so that these application clients can run in WebSphere Application Server Version 7. The script converts an EAR file that you want to migrate and then overwrites the original EAR file with the converted EAR file.

The following title provides information about the clientUpgrade script.

Note: This command was deprecated in Version 6.1.

Type	Description
Product	The clientUpgrade script is available in the WebSphere Application Server (Express and Base) product only.
Authority	To run this script, your user profile must have *ALLOBJ authority.

Syntax	The syntax of the clientUpgrade script is:
	`clientUpgrade` *`EAR_file`* `[ -clientJAR` *`client_JAR_file`* `]`
	`[ -logFileLocation` *`logFileLocation`* `]`
	`[ -traceString` *`trace_spec`* `[ -traceFile` *`file_name`* `] ]`

Type	Description
Parameters	The parameters of the clientUpgrade script are: • *EAR_file* -- This is a required parameter. The value EAR_file specifies the fully-qualified path of the EAR file that contains the application client modules that you want to migrate. • -clientJAR -- This is an optional parameter. The value client_JAR_file specifies a JAR file that you want to migrate. The script overwrites the original EAR file with a new EAR file that contains only the specified JAR files. If you do not specify this parameter, the clientUpgrade script migrates all client JAR files in the EAR file. • -logFileLocation -- Use this optional parameter to specify an alternate location to store the log output. • -traceString -- This is an optional parameter. The value trace_spec specifies the trace information that you want to collect. To gather all trace information, specify ″*=all=enabled″ (including the double quotation marks (″)). By default, the script does not gather trace information. If you specify this parameter, you must also specify the -traceFile parameter. • -traceFile -- This is an optional parameter. The value file_name The value file_name specifies the name of the output file for trace information. If you specify the -traceString parameter but do not specify the -traceFile parameter, the script does not generate a trace file.
Logging	The clientUpgrade script displays status while it runs. It also saves more extensive logging information to the clientupgrade.log file. This file is located in the /QIBM/UserData/WebSphere/AppServer/V7/edition/profiles/default/logs directory (for a default installation using the default profile) or in the location specified by the -logFileLocation parameter.

These examples demonstrate correct syntax. In this example, the My51Application.ear file is migrated from WebSphere Application Server Version 5.1, The script overwrites the original EAR file with a new file that you can deploy in your WebSphere Application Server Version 7 profile.

```
clientUpgrade /My51Application/My51Application.ear
```

In this example, only the `myJarFile.jar` client JAR file is migrated. The script overwrites `My51Application.ear` with an EAR file that contains `myJarFile.jar`. You can deploy the new EAR file in your WebSphere Application Server profile.

```
clientUpgrade /My51Application/My51Application.ear -clientJAR myJarFile.jar
```

convertScriptCompatibility command

The convertScriptCompatibility command is used by administrators to convert their configurations from a mode that supports backward compatibility of WebSphere Application Server Version 5.1.x or Version 6.0.x administration scripts to a mode that is fully in the Version 7.0 configuration model.

The scope of the configuration changes depend on the type of profile that is being processed.

- For standalone configurations, the default is to convert all servers owned by the node in that configuration.

  Use the -serverName parameter for more granular control.

Nodes are checked to verify that they are at a WebSphere Application Server Version 7.0 level before they are processed in order to support mixed-node configurations. Client environments are not processed.

The following conversions take place with this tool:

- `processDef` to `processDefs`

  WCCM objects of type `processDef` from WebSphere Application Server Version 5.1.x are converted to use `processDefs` as defined in the Version 7.0 server.xml model. The existing `processDef` object remains in the configuration and is ignored by the runtime.
- transports to channels

Existing transport entries in the configuration from WebSphere Application Server Version 5.1.x are mapped to channel support. This affects server.xml and serverindex.xml files. The values of the transport settings are used to create new channel entries.

- SSL configuration

  WebSphere Application Server Version 7.0 contains enhancements to SSL configuration that result in refactoring the existing SSL configuration model. Both the old and the new model are supported. The default is to map to the WebSphere Application Server Version 5.1.x or Version 6.x SSL configuration model.

- `bootstrapAddress` to `bootstrapAddresses`

  Each single bootstrap address configuration is converted to a new bootstrap address list configuration containing that single bootstrap address.

- `ObjectRequestBroker` from not using the server thread pool to using it

  For example, `<ObjectRequestBroker useServerThreadPool="false"...>` is changed to `<ObjectRequestBroker useServerThreadPool="true">`.

Location

The convertScriptCompatibility command is located in the following directory.

- *app_server_root*/bin

Syntax

The syntax is as follows:

```
convertScriptCompatibility [-help]

convertScriptCompatibility [-profileName profile_name]
                    [-backupConfig true | false]
                    [-nodeName node_name  [-serverName server_name]]
                    [-traceString trace_spec  [-traceFile file_name]]
```

Parameters

Supported arguments include the following parameters:

-help
This displays help for this command

-backupConfig
This is an optional parameter that is used to back up the existing configuration of the current profile. The default is true—that is, to use the backupConfig command to save a copy of the current configuration into the *profile_name*/temp directory.

Use the restoreConfig command to restore that configuration as required.

Read the ″restoreConfig command″ article in the information center for more information.

-profileName
This is an optional parameter that is used to specify the profile configuration in the Version 7.0 environment. If this is not specified, the default profile is used. If the default profile has not been set or cannot be found, the system returns an error.

-nodeName
This is an optional parameter that is used to specify a particular node name be processed rather than every node in the configuration. If this is not specified, all nodes in the configuration are converted.

-serverName
This is an optional parameter that is used to specify a particular server name to be processed rather than every server in the configuration. It can be used on all profile types and can be used in

conjunction with the -nodeName parameter when processing Network Deployment configurations. If this parameter is not specified, all servers in the configuration are converted. If it is used in conjunction with the -nodeName parameter, all processing is limited to the specified node name.

-traceString
This is an optional parameter. The value *trace_spec* specifies the trace information that you want to collect. To gather all trace information, specify `"*=all=enabled"` (with quotation marks). The default is to not gather trace information. If you specify this parameter, you must also specify the -traceFile parameter.

-traceFile
This is an optional parameter. The value *file_name* specifies the name of the output file for trace information. If you specify the -traceString parameter but do not specify the -traceFile parameter, the command does not generate a trace file.

Usage

Standalone application server profile

Example scenario

1. Run the WASPostUpgrade command and specify `-scriptCompatibility=true` or do not specify a value for the -scriptCompatibility parameter (which has a default value of true).
2. Follow these steps to convert all servers under this standalone profile:
   a. Start a Qshell session.
   ```
   STRQSH
   ```
   b. Change to the Version 7.0 *app_server_root*/bin directory.
   c. Run the following command:
   ```
   convertScriptCompatibility -profileName profile_name
   ```

For more information about where to run this command, read the "Using command line tools" article in the information center.

WASPreUpgrade command

The WASPreUpgrade command for WebSphere Application Server Version 7.0 saves the configuration of a previously installed version of WebSphere Application Server into a migration-specific backup directory.

Location

The command file is located in and must be run from the Version 7.0 *app_server_root*/bin directory.

Authority

To run this command script, your user profile must have *ALLOBJ authority.

Syntax

```
WASPreUpgrade backupDirectory
              currentWebSphereDirectory
              [-traceString trace_spec [-traceFile file_name ]]
              [-machineChange true | false]
              [-workspaceRoot profile1=user_workspace_folder_name_1;profile2=user_workspace_folder_name_2]
```

Parameters

The command has the following parameters:

backupDirectory

This is a required parameter and must be the first parameter that you specify. The value *backupDirectory* specifies the name of the directory where the command script stores the saved configuration.

This is also the directory from which the WASPostUpgrade command reads the configuration.

If the directory does not exist, the WASPreUpgrade command script creates it.

currentWebSphereDirectory

This is a required parameter and must be the second parameter that you specify. This can be any edition of WebSphere Application Server Version 5.1.x or Version 6.x for which migration is supported.

The value *currentWebSphereDirectory* specifies the name of the instance or profile root directory for the current WebSphere Application Server Version 5.1.x instance or Version 6.x profile that you want to migrate.

- **For Version 5.1.x Express:** /QIBM/UserData/WebASE51/ASE/*instance*
- **For Version 5.1.x base:** /QIBM/UserData/WebAS51/Base/*instance*
- **For Version 5.1.x Network Deployment:** /QIBM/UserData/WebAS51/ND/*instance*
- In Version 6.x, the profile root may be a unique value chosen during profile creation but the following directories are the defaults:
  - **For Version 6.0 Express or base:** /QIBM/UserData/WebSphere/AppServer/V6/Base/profiles/*profile*
  - **For Version 6.0 Network Deployment:** /QIBM/UserData/WebSphere/AppServer/V6/ND/profiles/*profile*
  - **For Version 6.1 Express or base:** /QIBM/UserData/WebSphere/AppServer/V61/Base/profiles/*profile*
  - **For Version 6.1 Network Deployment:** /QIBM/UserData/WebSphere/AppServer/V61/ND/profiles/*profile*

-traceString

This is an optional parameter. The value *trace_spec* specifies the trace information that you want to collect.

To gather all trace information, specify `"*=all=enabled"` (with quotation marks).

If you do not specify the -traceString or -traceFile parameter, the command creates a trace file by default and places it in the *backupDirectory*/logs directory.

If you specify this parameter, you must also specify the -traceFile parameter.

-traceFile

This is an optional parameter. The value *file_name* specifies the name of the output file for trace information.

If you do not specify the -traceString or -traceFile parameter, the command creates a trace file by default and places it in the *backupDirectory*/logs directory.

If you specify the -traceString parameter but do not specify the -traceFile parameter, the script does not generate a trace file.

-machineChange

This is an optional parameter used for a migration involving cross operating-system and machine boundaries. If specified as true, this parameter provides support for changing physical hardware when migrating by backing up items that are stored outside the WebSphere Application Server installation or profile folder hierarchy. If specified as false, only files stored under the WebSphere Application Server installation folder or profile folders are copied to the backup directory during migration.

The default is false.

When this value is false, migration assumes that the new and old WebSphere Application Server installations are on the same physical machine with shared access to the file system. Therefore, any files located outside the WebSphere directories are communal and can be shared. Migration does not copy files outside the WebSphere Application Server tree into the backup directory when -machineChange is false. If you select -machineChange=false, you must run the WASPostUpgrade command on the same physical hardware.

If you intend to run the WASPostUpgrade command on a different machine or file system, you should run the WASPreUpgrade command with -machineChange=true. If you select -machineChange=true, migration creates an additional subdirectory (/migrated/) in the migration backup directory that contains any files referenced by the WebSphere Application Server configuration that reside outside the product or profile directories. When you run the WASPostUpgrade command, these files are returned to their original paths on the new machine.

Note:

If you migrate with Service Integration Bus (SIB) busses configured with file-system file-store repositories, you might require additional space in your migration heap and migration backup directory. Each bus has three file-store values—a log, a tempspace, and a repository. These three files vary in size, but they can be as much as 100-500 MB each. When migration is running, it backs up any file stores that are in the WebSphere Application Server tree during the pre-upgrade process. There needs to be sufficient space on the file system to permit this. If file stores exist at the destination location already during the post-upgrade process, migration backs up the file stores in memory to support rollback.

If you run the WASPreUpgrade command with -machineChange=true, resulting in a backup directory that contains shared file-store objects, you might find that the post-upgrade process suffers from out-of-memory exceptions because the default maximum heap is too small to contain the file-store backups in support of rollback. To resolve this issue, perform one of the following three tasks:

- If the file stores at the system location are valid, delete the copies from the backup directory before running the WASPostUpgrade command.

  By deleting the entire /migrated/ subdirectory from the migration backup directory before running the WASPostUpgrade command, you essentially convert your pre-upgrade backup from -machineChange=true to -machineChange=false.
- If the copies of the file stores in the backup directory are valid, delete the versions at the destination location.

  This changes the rollback support so that the destination files do not exist and will not occupy space in memory during the migration.
- If you require rollback support and you need both the files in the backup directory as well as the files on the file system, increase your maximum heap size for the post-upgrade process to some value great enough to support all of the SIB files that conflict.

-workspaceRoot
This is an optional parameter. The value *user_workspace_folder_name_x* specifies the location of the administrative console customized ″My tasks″ settings for one or more profiles.

Logging

The WASPreUpgrade tool displays status to the screen while it runs. The tool also saves a more extensive set of logging information in the WASPreUpgrade.*time_stamp*.log file written to the *backupDirectory* directory, where *backupDirectory* is the value specified for the backupDirectory parameter. You can view the WASPreUpgrade.*time_stamp*.log file with a text editor.

Migrated resources

WASPreUpgrade saves all of your resources, but it does not migrate entities in your classes directory.

Migration saves the following files in the *backupDirectory* directory.

- config
- properties

WASPostUpgrade command

The WASPostUpgrade command for WebSphere Application Server Version 7.0 retrieves the saved configuration that was created by the WASPreUpgrade command from the *backupDirectory* that you specified. The WASPostUpgrade script for WebSphere Application Server Version 7.0 reads the configuration from this directory to migrate to WebSphere Application Server Version 7.0 and adds all migrated applications into the *app_server_root*/installedApps directory for the Version 7.0 installation.

Location

The command file is located in and must be run from the Version 7.0 *app_server_root*/bin directory.

Authority

To run this command script, your user profile must have *ALLOBJ authority.

Syntax

```
WASPostUpgrade backupDirectory
                    [-username userID]
                    [-password password]
                    [-profileName profile_name]
                    [-scriptCompatibility true | false]
                    [-portBlock port_starting_number]
                    [-backupConfig true | false]
                    [-replacePorts true | false]
                    [-includeApps true | false | script]
                    [[-appInstallDirectory user_specified_directory] | [-keepAppDirectory true | false]]
                    [-traceString trace_spec [-traceFile file_name]]
```

Parameters

The command has the following parameters:

backupDirectory
This is a required parameter. The value *backupDirectory* specifies the name of the directory in which the WASPreUpgrade tool stores the saved configuration and files and from which the WASPostUpgrade tool reads the configuration and files.

-username
This is an optional parameter. The value *userID* specifies the administrative user name of the current WebSphere Application Server Version 5.1.x or Version 6.x installation.

-password
This is an optional parameter. The value *password* specifies the password for the administrative user name of the current WebSphere Application Server Version 5.1.x or Version 6.x installation.

-profileName
This is an optional parameter for migrating to specific profiles in WebSphere Application Server Version 7.0. The value *profile_name* specifies the name of the Version 7.0 profile to which the script migrates your configuration. You must have already created this profile before calling the WASPostUpgrade command.

If the -profileName parameter is not specified, the default profile is used. If no default profile is found, the system reports an error.

Note: When migrating a standalone application server from Version 6.x to Version 7.0, you can choose a standalone application server node that has already been registered with an administrative agent as the target of the migration.

-scriptCompatibility

This is an optional parameter used to specify whether or not migration should create the following Version 5.1.x or Version 6.x configuration definitions:

- Transport
- ProcessDef
- Version 5.1.x or Version 6.x SSL

instead of the following Version 7.0 configuration definitions:

- Channels
- ProcessDefs
- Version 7.0 SSL

The default value is true.

Specify true for this parameter in order to minimize impacts to existing administration scripts. If you have existing wsadmin scripts or programs that use third-party configuration APIs to create or modify the Version 5.1.x or Version 6.x configuration definitions, for example, you might want to specify true for this option during migration.

Note: This is meant to provide a temporary transition until all of the nodes in the environment are at the Version 7.0 level. When they are all at the Version 7.0 level, you should perform the following actions:

1. Modify your administration scripts to use all of the Version 7.0 settings.
2. Use the convertScriptCompatability command to convert your configurations to match all of the Version 7.0 settings.

   Read “convertScriptCompatibility command” on page 18 for more information.

-backupConfig

This is an optional parameter used to specify whether the existing WebSphere Application Server Version 7.0 configuration is saved before any changes are made by the WASPostUpgrade tool. The default is true—that is, to use the backupConfig command to save a copy of the current configuration into the *profile_name*/temp directory.

Use the restoreConfig command to restore that configuration as required. Read the ″restoreConfig command″ article in the information center for more information.

-portBlock

This is an optional parameter. The *port_starting_number* value specifies the starting value of a block of consecutive port numbers to assign when creating new ports.

By default, this parameter is not set.

If a value is specified for this parameter, any new ports that are assigned are set based on this value. Every time a new port value is required, the port is created based on this value and the seed value is incremented for the next usage. No duplicate ports are assigned.

-replacePorts

This optional parameter is used to specify how to map port values.

- True

  Use all port values from the old configuration in the new configuration. All ports from the old configuration supersede the settings for the same ports in the new configuration.

  - If the -portBlock parameter is not set, a conflicting port is renumbered incrementally from its original value until a nonconflicting port is identified.
  - If the -portBlock parameter is set, a conflicting port is renumbered incrementally from the port block number until a nonconflicting port is identified.

  This value is the default.
- False

  Do not replace the default port definitions in the new profile with the values from the old configuration during migration. All ports values from the new configuration supersede the settings for the same ports in the old configuration

  - If the -portBlock parameter is not set, a conflicting port is renumbered incrementally from its original value until a nonconflicting port is identified.
  - If the -portBlock parameter is set, a conflicting port is renumbered incrementally from the port block number until a nonconflicting port is identified.

-includeApps

This is an optional parameter that can be specified in the following ways:

- True

  Include user enterprise applications as part of the migration.

  This value is the default.
- False

  Do nothing with user enterprise applications during WASPostUpgrade processing.
- Script

  Prepare user enterprise applications for installation in the WebSphere Application Server Version 7.0 installableApps directory without actually installing them during WASPostUpgrade processing.

  Scripts that can be used to install these applications are generated and saved in the *backupDirectory* directory. You can then run these files at any point and in any combination after the WASPostUpgrade command has completed. You can also reorganize and combine these files for better applications installation efficiency if you want.

WebSphere Application Server system applications migrate regardless of the value set by this parameter.

-keepAppDirectory

This is an optional parameter used to specify whether to install all applications to the same directories in which they are currently located. The default is false.

If this parameter is specified as true, each individual application retains its location.

If you specify this parameter, you cannot specify the -appInstallDirectory parameter.

Note: If this parameter is specified as true, the location is shared by the existing WebSphere Application Server Version 5.1.x or Version 6.x installation and the Version 7.0 installation. If you keep the migrated applications in the same locations as those of the previous version, the following restrictions apply:

- The WebSphere Application Server Version 7.0 mixed-node support limitations must be followed. This means that the following support cannot be used when evoking the wsadmin command:
  - Precompile JSP
  - Use Binary Configuration
  - Deploy EJB
- You risk losing the migrated applications unintentionally if you later delete applications from these locations when administering (uninstalling for example) your Version 5.1.x or Version 6.x installation.

-appInstallDirectory

This is an optional parameter that is used to pass the directory name to use when installing all applications during migration. The default of *profile_name*\installedApps is used if this parameter is not specified.

If you specify this parameter, you cannot specify the -keepAppDirectory parameter.

Quotes must be used around the directory name if one or more spaces are in the name.

If you use this parameter, the migration tools investigate the node-level variables for the node being migrated both in the backup directory (variables for the old release) and in the destination profile (variables from the new release). If the path is part of any of the following variables in either of these releases, the tools contract the path information to use the related variable:

- APP_INSTALL_ROOT
- USER_INSTALL_ROOT
- WAS_INSTALL_ROOT

When the contraction takes place, you receive the following warning message that tells you that the tools changed your specified value and what that contracted value is:

```
MIGR0341W: Application install directory has been updated to {0}.
```

For example:

```
MIGR0341W: Application install directory has been updated to ${USER_INSTALL_ROOT}\customAppDirectory.
```

or

```
MIGR0341W: Application install directory has been updated to ${APP_INSTALL_ROOT}\
cellName\customAppDirectory\.
```

-traceString

This is an optional parameter. The value *trace_spec* specifies the trace information that you want to collect.

To gather all trace information, specify `"*=all=enabled"` (with quotation marks).

If you do not specify the -traceString or -traceFile parameter, the command creates a trace file by default and places it in the *backupDirectory*/logs directory.

If you specify this parameter, you must also specify the -traceFile parameter.

-traceFile

This is an optional parameter. The value *file_name* specifies the name of the output file for trace information.

If you do not specify the -traceString or -traceFile parameter, the command creates a trace file by default and places it in the *backupDirectory*/logs directory.

If you specify the -traceString parameter but do not specify the -traceFile parameter, the script does not generate a trace file.

Security considerations

The target system must have security disabled before migration. If you migrate from a source configuration that has security enabled, the WASPostUpgrade command automatically enables security for the Version 7.0 target configuration during the migration.

Migrating profiles

After you have migrated your applications, you need to migrate your instance configurations to profiles.

Before you begin

Read “Overview of migration, coexistence, and interoperability” on page 1 and “Premigration considerations” on page 4.

For resources to help you plan and perform your migration, visit Knowledge Collection: Migration planning for WebSphere Application Server.

Read “Checking for the product-configuration migration prerequisites” on page 15 to see how to determine the currently installed product level of WebSphere Application Server.

Select the appropriate option to obtain instructions on how to migrate from your old version of WebSphere Application Server to a new or default Version 7.0 profile.

- “Migrating to a Version 7.0 standalone application server profile”

  This article contains instructions for migrating a WebSphere Application Server Version 5.1.x or Version 6.x profile to a Version 7.0 standalone application server profile.

Note: For help, read Chapter 9, “Troubleshooting migration,” on page 57.

Migrating to a Version 7.0 standalone application server profile

Use the migration tools to migrate from WebSphere Application Server Version 5.1.x or Version 6.x to a new Version 7.0 standalone application server profile.

Before you begin

Read “Overview of migration, coexistence, and interoperability” on page 1 and “Premigration considerations” on page 4.

For resources to help you plan and perform your migration, visit Knowledge Collection: Migration planning for WebSphere Application Server.

For help, read Chapter 9, “Troubleshooting migration,” on page 57.

Before following these instructions, perform the actions in “Preparing for product-configuration migration” on page 15.

Note: Before migrating a WebSphere Application Server Version 5.1.x or Version 6.x standalone application server profile, use the backupConfig command or your own preferred backup utility to back up your existing configuration if you want to be able to restore it to its previous state after migration. Read the ″backupConfig command″ article in the information center for more information. Make sure that you note the exact name and location of this backed-up configuration.

1. Create a WebSphere Application Server Version 7.0 profile to receive the Version 5.1.x or Version 6.x configuration.
   a. Start the Qshell environment so that you can run WebSphere Application Server scripts.

      Enter the following command from a command line:

      ```
      STRQSH
      ```

   b. Run the dspwasinst script to obtain the node name and server name for the Version 5.1.x or Version 6.x instance or profile that is to be migrated.

      Use the following parameters:

      ```
      app_server_root/bin/dspwasinst
       -instance 51x_or_6x_profile_name
      ```

      where

- *app_server_root* is the location of the Version 5.1.x or Version 6.x installation that contains the instance or profile to be migrated
- *51x_or_6x_profile_name* is the name of the Version 5.1.x or Version 6.x instance or profile that is to be migrated

The name of the Version 5.1.x or Version 6.x node is listed in the **Node** section, and the name of the server is listed in the **Information for server** section.

c. Run the manageprofiles script.

Use the following parameters:

```
app_server_root/bin/manageprofiles
 -create
 -profileName 70_profile_name
 -startingPort starting_port_number
 -templatePath app_server_root/profileTemplates/default
 -serverName 51x_or_6x_application_server_name
 -nodeName 51x_or_6x_node_name
```

where

- *app_server_root* is the location where Version 7.0 is installed
- *70_profile_name* is the name of your Version 7.0 profile

  This parameter must be identical to the Version 5.1.x or Version 6.x instance or profile that is to be migrated.
- *starting_port_number* is the first of a block of 13 consecutive ports
- *51x_or_6x_application_server_name* is the name of the Version 5.1.x or Version 6.x application server obtained in the previous step
- *51x_or_6x_node_name* is the Version 5.1.x or Version 6.x node name obtained in the previous step

  The source and target node names must be identical when migrating to Version 7.0.

For details on the syntax and parameters of the manageprofiles command, read the ″manageprofiles command″ article in the information center.

2. Save the WebSphere Application Server Version 5.1.x or Version 6.x configuration.

a. Start the Qshell environment so that you can run WebSphere Application Server scripts.

Enter the following command from a command line:

```
STRQSH
```

b. Run the WASPreUpgrade script.

Use the following parameters:

```
app_server_root/bin/WASPreUpgrade
 backup_directory_name
 profile_root
```

where

- *app_server_root* is the location where Version 7.0 is installed
- *backup_directory_name* (required parameter) is the fully qualified path to the integrated file system directory where the WASPreUpgrade migration tool stores the saved configuration and files

  The directory is created if it does not already exist. It is also the directory where the WASPreUpgrade migration tool writes a log file called WASPreUpgrade.log that chronicles the steps taken by the WASPreUpgrade command.
- *profile_root* (required parameter) is the path to the Version 5.1.x or Version 6.x instance or profile that is to be migrated

For a full explanation of the WASPreUpgrade command and its parameters, read “WASPreUpgrade command” on page 20.

3. Restore the WebSphere Application Server Version 5.1.x or Version 6.x configuration into a Version 7.0 profile.
   a. Start the Qshell environment so that you can run WebSphere Application Server scripts.

      Enter the following command from a command line:

      ```
      STRQSH
      ```
   b. Run the WASPostUpgrade script.

      Use the following parameters:

      ```
      app_server_root/bin/WASPostUpgrade
       backup_directory_name
       -profileName 70_profile_name
       [-portBlock port_starting_number]
      ```

      where
      - *app_server_root* is the location where Version 7.0 is installed
      - *backup_directory_name* is the required name of the directory in which the WASPreUpgrade tool stored the saved configuration and files and from which the WASPostUpgrade tool reads the configuration and files
      - *70_profile_name* is the name of the Version 7.0 profile to which the script migrates your configuration
      - *port_starting_number* specifies the first of a block of 10 to 15 consecutive port numbers that are not in use on the iSeries server where the migration is being performed

        It is recommended that you always specify the -portBlock parameter if you do not want your profile's ports to conflict with the default profile's ports.

        Note: When migrating a standalone application server from Version 6.x to Version 7.0, you can choose a standalone application server node that has already been registered with an administrative agent as the target of the migration.

      For a full explanation of the WASPostUpgrade command and its parameters, read "WASPostUpgrade command" on page 23.
4. Start the WebSphere Application Server Version 7.0 profile that receives the Version 5.1.x or Version 6.x configuration.
   a. Start the QWAS7 subsystem if it is not already started.

      Enter the following command from a command line:

      ```
      STRSBS QWAS7/QWAS7
      ```
   b. Start the Qshell environment so that you can run WebSphere Application Server scripts.

      Enter the following command from a command line:

      ```
      STRQSH
      ```
   c. Run the startServer script.

      Use the following parameters:

      ```
      app_server_root/bin/startServer
       -profileName 70_profile_name
       51x_or_6x_server_name
      ```

      where
      - *app_server_root* is the location where Version 7.0 is installed
      - *70_profile_name* is the name of the Version 7.0 profile created in an earlier step
      - *51x_or_6x_server_name* is the name of the Version 5.1.x or Version 6.x application server that was migrated

Related concepts

"Overview of migration, coexistence, and interoperability" on page 1
The goal of migration is to reconstruct your earlier version of WebSphere Application Server in a Version 7.0 environment. Coexistence allows you to create a mixed-version environment that is not in conflict and allows the nodes of all versions to start and run at the same time. Coexistence also facilitates rollback and allows one or the other version to run at one time. Interoperating is exchanging data between two coexisting product installations or between products on different systems.

"Premigration considerations" on page 4
Before you begin the process of migrating to WebSphere Application Server Version 7.0, there are some considerations of which you need to be aware.

Related tasks

Chapter 9, "Troubleshooting migration," on page 57
You might encounter problems while migrating from an older version of WebSphere Application Server.

"Preparing for product-configuration migration" on page 15
You must prepare your system for migrating to WebSphere Application Server Version 7.0.

Related reference

"WASPreUpgrade command" on page 20
The WASPreUpgrade command for WebSphere Application Server Version 7.0 saves the configuration of a previously installed version of WebSphere Application Server into a migration-specific backup directory.

"WASPostUpgrade command" on page 23
The WASPostUpgrade command for WebSphere Application Server Version 7.0 retrieves the saved configuration that was created by the WASPreUpgrade command from the *backupDirectory* that you specified. The WASPostUpgrade script for WebSphere Application Server Version 7.0 reads the configuration from this directory to migrate to WebSphere Application Server Version 7.0 and adds all migrated applications into the *app_server_root*/installedApps directory for the Version 7.0 installation.

Migrating IBM Cloudscape or Apache Derby databases

The migration tools migrate any IBM Cloudscape® database instances to Apache Derby instances in the new configuration, and they copy any Apache Derby instances that are stored in the previous release's WebSphere Application Server configuration tree to the new release's configuration tree. After you use the migration tools, you should verify the results of the database migration and manually migrate any Cloudscape database instances or copy any Derby database instances that are not automatically migrated or copied by the tools.

Before you begin

Read "Overview of migration, coexistence, and interoperability" on page 1 and "Premigration considerations" on page 4. For resources to help you plan and perform your migration, visit Knowledge Collection: Migration planning for WebSphere Application Server.

Note:

- Before you run the migration tools, ensure that any application servers hosting applications that are using a Cloudscape or a Derby database are closed.

  Otherwise, the database migration will fail.
- Before you run the migration tools, ensure that the *debug migration trace* is active.

  By default, this trace function is enabled. To reactivate the debug migration trace if it is disabled, set one of the following trace options:
  - `all traces*=all`
  - `com.ibm.ws.migration.WASUpgrade=all`

About this task

WebSphere Application Server Version 7.0 requires Apache Derby Version 10.3 or later. Apache Derby Version 10.3 is a pure Java database server that combines the Derby runtime with the opportunity to use the full services of IBM Software Support. For comprehensive information about Apache Derby Version 10.3, read the Apache Derby Web site.

For help, read Chapter 9, “Troubleshooting migration,” on page 57.

Note: Derby-to-Derby migration performs a file-system copy of the data at a given point in time. This snapshot will not remain in sync with the database in the previous installation. If you roll back to the previous release, any updates to the database that you made after migration will not be reflected in the previous installation.

1. Migrate the configuration to Version 7.0.
2. Verify the automatic migration of Cloudscape database instances or copying of Derby database instances.

   When you migrate from WebSphere Application Server Version 5.1.x or Version 6.x to Version 7.0, the migration tools automatically upgrade Cloudscape or Derby database instances that are accessed through the embedded framework by some internal components such as the UDDI registry. The tools also attempt to upgrade Cloudscape or Derby instances that your applications access through the embedded framework. You must verify these migration results after running the migration tools.

   - To distinguish between a partially and a completely successful Cloudscape-to-Derby migration, verify the automatic-migration results by performing the following tasks:
     a. Check the general migration post-upgrade log for database error messages.

        These exceptions indicate database migration failures. The migration tool references all database exceptions with the prefix `DSRA`.

     b. Check the individual database migration logs.

        These logs have the same timestamp as that of the general migration post-upgrade log. The individual logs display more detail about errors that are listed in the general post-upgrade log as well as expose errors that are not documented by the general log.

        The path name of each database log is *app_server_root*/profiles/*profileName*/logs/*myFulldbPathName*_migrationLog*timestamp*.log.

     c. Look at the debug log that corresponds with the database migration log.

        The WebSphere Application Server migration utility triggers a *debug migration trace* by default; this trace function generates the database debug logs.

        The full path name of each debug log is *app_server_root*/profiles/*profileName*/logs/*myFulldbPathName*_migrationDebug*timestamp*.log.

     Performing these tasks gives you vital diagnostic data to troubleshoot the partially migrated databases as well as those that fail automatic migration completely. Ultimately, you must migrate databases that were not completely migrated automatically through a manual process. The log messages contain the exact old and new database path names that you must use to run the manual migration. Note these new path names precisely.

     Read the ″Verifying the Cloudscape automatic migration″ article in the information center for more information.

   - Verify that any Derby database instances that are stored in the previous release’s WebSphere Application Server configuration tree were copied to the new release’s configuration tree

     Check the general migration post-upgrade log for database error messages. These exceptions indicate database migration failures. The migration tool references all database exceptions with the prefix `DSRA`..

3. Manually migrate Cloudscape database instances or copy Derby database instances where necessary.

- The Version 7.0 migration tools do not attempt to migrate database instances that transact with applications through the Cloudscape Network Server or the Apache Derby Network Server framework. This exclusion eliminates the risk of corrupting third-party applications that access the same database instances as those accessed by WebSphere Application Server.

  To minimize the risk of migration errors for databases that were only partially upgraded during the automatic migration process, delete the new database. Troubleshoot the original database according to the log diagnostic data, then perform manual migration of the original database.

  Read the ″Upgrading Cloudscape manually″ article in the information center for more information.
- The Version 7.0 migration tools do not copy any Derby database instances outside the WebSphere Application Server configuration tree.

  If migration does not copy a Derby database instance automatically, copy the database instance manually.

4. Manually migrate your UDDI registry if it uses a database on the Cloudscape Network Server or the Apache Derby Network Server framework.

   Read the ″Migrating the UDDI registry″ article in the information center for more information.

What to do next

Service integration bus-enabled Web services use a Service Data Objects (SDO) repository for storing and serving WSDL definitions. If you migrate a configuration that uses a Cloudscape database as the SDO repository, the SDO application will still be configured to use Cloudscape in the new configuration. Migration converts the Cloudscape database to Derby, but you must still update any SDO application's backend ID to use the new database. After you migrate all of the nodes on a server with an SDO repository application that uses Cloudscape, perform the following actions to reset the database type used by the SDO application on the new configuration to Derby:

1. Read about the basic usage for the installSdoRepository.jacl script inside the script file.
2. Run the installSdoRepository.jacl script by changing to the *app_server_root*/bin/ directory and running the following command:

   ```
   wsadmin.extension -f app_server_root/bin/installSdoRepository.jacl -editBackendId DERBY_V10
   ```

Read the ″Installing and configuring the SDO repository″ article in the information center for more information on upgrading the SDO repository application to Version 7.0 .

Migrating from the WebSphere Connect JDBC driver

WebSphere Application Server Version 7.0 does not include the WebSphere Connect JDBC driver for SQL Server. Use the WebSphereConnectJDBCDriverConversion command to convert data sources from the WebSphere Connect JDBC driver to the DataDirect Connect JDBC driver or the Microsoft SQL Server 2005 JDBC driver. The WebSphereConnectJDBCDriverConversion command processes resources.xml files, and there are many options that can be specified to indicate which resources.xml files to process.

Before you begin

Read "Overview of migration, coexistence, and interoperability" on page 1 and "Premigration considerations" on page 4.

About this task

Syntax

```
WebSphereConnectJDBCDriverConversion
   [-profileName profile_name]
   [-driverType MS | DD]
   [-classPath class_path]
   [-nativePath native_path]
```

```
[-pathSeparator separator]
[[-cellName ALL | cell_name [-clusterName ALL | cluster_name] |
  [-applicationName ALL | application_name] |
  [-nodeName ALL | node_name] [-serverName ALL | server_name]]]
[-backupConfig true | false]
[-username userID]
[-password password]
[[-traceString trace_spec [-traceFile file_name]]
```

Parameters

-profileName *profile_name*

This optional parameter is used to specify a specific profile configuration in the Version 7.0 environment.

If this parameter is not specified, the default profile is used. If the default is not set or cannot be found, an error is returned.

If the command is run from within the *profile_name*/bin directory, it is implicit that the *profile_name* profile should be migrated and this parameter is not needed.

-driverType MS | DD

This required parameter is used to indicate the type of conversion that you want to perform.

Specifying MS converts to the Microsoft SQL Server 2005 JDBC Driver, and specifying DD converts to the DataDirect Connect JDBC driver.

-classPath *class_path*

This required parameter is used to specify the class path for the new JDBC driver.

-nativePath *native_path*

This optional parameter is used to specify the native path for the new JDBC driver.

-pathSeparator *separator*

This optional parameter is used to specify a path separator other than the default.

The default is operating-system dependant and is listed in the command-line help.

-cellName ALL | *cell_name*

This optional parameter is used to specify the specific name of the cell to process or to specify all cells.

The default is ALL

-clusterName ALL | *cluster_name*

This optional parameter is used to specify the specific name of the cluster to process or to specify all clusters.

The default is ALL

-application nameName ALL | *application_name*

This optional parameter is used to specify applications to process. (Some resource.xml files might exist in applications if enhanced EAR file support is used.)

The default is ALL

-nodeName ALL | *node_name*

This optional parameter is used to specify the specific name of the node to process or to specify all node names in the configuration.

The default is ALL.

-serverName ALL | *server_name*

This optional parameter is used to specify the specific name of a server or to specify all server names in a node.

The default is ALL.

-backupConfig true | false

This parameter is used to specify whether (true) or not (false) to back up the existing configuration before the command makes changes to the configuration.

The default is true.

-username *userID*

This optional parameter is used to specify the user name to be used by this command.

-password *password*

This optional parameter is used to specify the password to be used by this command.

-traceString *trace_spec*

This optional parameter is used with -traceFile to gather trace information for use by IBM service personnel.

The value of traceString is ″*=all=enabled″ and must be specified with quotation marks to be processed correctly.

-traceFile *file_name*

This optional parameter is used with -traceString to gather trace information for use by IBM service personnel.

Rolling back your environment

After migrating to a WebSphere Application Server Version 7.0 environment, you can roll back to a Version 5.1.x or Version 6.x environment. This returns the configuration to the state that it was in before migration. After rolling back the environment, you can restart the migration process.

About this task

Generally, migration does not modify anything in the configuration of the prior release; however, there are cases where minimal changes are made that are reversible.

To roll back a Version 7.0 standalone application server to Version 5.1.x or Version 6.x, follow the instructions in “Rolling back a standalone application server.”

Results

The configuration should now be returned to the state that it was in before migration.

What to do next

You can now restart the migration process if you want to do so.

Rolling back a standalone application server

You can use the restoreConfig and wsadmin commands to roll back a migrated WebSphere Application Server Version 7.0 standalone application server to the state that it was in before migration.

Before you begin

Note: When migrating a Version 5.1.x or Version 6.x standalone application server, the best practice is to perform the following actions if you want to be able to roll it back to its previous state after migration:

1. Run the backupConfig command or your own preferred utility to back up the Version 5.1.x or Version 6.x standalone application server configuration.

   Note: Make sure that you note the exact name and location of this backed-up configuration.

   Read the ″backupConfig command″ article in the information center for more information.
   2. Migrate the standalone application server.
   3. If necessary, you can now roll back the standalone application server that you just migrated.

1. Stop all of the servers that are currently running in the Version 7.0 environment.
2. Perform one of the following actions to restore the Version 5.1.x or Version 6.x configuration for the standalone application server.
   - Run the restoreConfig command or your own preferred utility to restore the Version 5.1.x or Version 6.x configuration.

     Note: Make sure that you restore the same backed-up configuration that you created just before you migrated this standalone application server.

     Read the ″restoreConfig command″ article in the information center for more information.
   - Use the wsadmin command to run the migrationDisablementReversal.jacl script from the Version 5.1.x or Version 6.x *profile_root*/bin directory of the standalone application server.

     Use the following parameters:

     ```
     app_server_root/bin/wsadmin -instance instance -conntype NONE
     -f profile_root/bin/migrationDisablementReversal.jacl
     ```
3. If you chose to keep the installed applications in the same location as the prior release during migration to Version 7.0 and any of the Version 7.0 applications are not compatible with the prior release, install applications that are compatible.
4. Delete the Version 7.0 profile for the standalone application server.

   Read the ″Deleting a profile″ article in the information center for more information.
5. Start the rolled-back standalone application server in the Version 7.0 environment.

Results

The configuration should now be returned to the state that it was in before migration.

What to do next

You can now restart the migration process if you want to do so.

Chapter 4. Migrating Web server configurations

You can migrate a Web server from supporting an earlier version of WebSphere Application Server to support the current version.

1. Configure an HTTP server instance.

   Read the ″Configuring an HTTP server instance″ article in the information center for more information.

   There are two options from which to choose:

   - Create a new HTTP server instance to be used by the WebSphere Application Server Version 7.0 profile.

     This method allows WebSphere Application Server Version 5.1.x or Version 6.x and Version 7.0 profiles to continue operating correctly.
   - Update the HTTP server instance configuration for the WebSphere Application Server Version 5.1.x or Version 6.x profile that is being migrated.

     This method changes the HTTP instance configuration to work with the WebSphere Application Server Version 7.0 profile and makes the WebSphere Application Server Version 5.1.x or Version 6.x profile no longer usable.
2. Configure the virtual host for the WebSphere Application Server Version 7.0 profile.

   Read the ″Configuring virtual hosts″ article in the information center for more information.

   This step ensures that both the host and HTTP transport port number exist in the virtual host list.

   If you created a new HTTP server in the previous step or if you used the -portBlock parameter when performing the migration, the virtual host will not contain the correct port for communication with your HTTP server. You need to add a host alias for the port used by your HTTP server.
3. Configure communication with Web servers.

   Read the ″Communicating with Web servers″ article in the information center for more information.

   This step regenerates the plug-in configuration file, plugin-cfg.xml. It needs to be done after any configuration changes have been made.

   Additional configuration is required if Secure Sockets Layer (SSL) is enabled on a plug-in transport. In addition to copying the .kdb file to the Version 7.0 profile, you must edit the plug-in to specify the .kdb file required for the plug-in to use the transport.

   For more information on copying the .kbd files to the Version 7.0 profile, read the section on J2EE security in “Configuration mapping during product-configuration migration” on page 11.

What to do next

Plug-in considerations when you are migrating from WebSphere Application Server Version 5.1.x to Version 7.0 In WebSphere Application Server Version 7.0, the plug-in configuration file has a one-to-one relationship with a Web server.

- The plug-in configuration file (plugin-cfg.xml) generated after successful migration from Version 5.1.x to Version 7.0 is topology centric—that is, it includes all the applications within a cell. You can manage this cell-wide plug-in configuration file from the Version 7.0 administrative console, by using the GenPluginCfg command, or by using the Plug-in Config Generator MBean.

  Be aware that regenerating the plug-in configuration can overwrite manual configuration changes that you might want to preserve.
- The application-centric generation of the plugin-cfg.xml file is supported using the Version 7.0 administrative console. Being application centric means that the plugin-cfg.xml file generated in the administrative console has a granularity that allows each application to be mapped to its specific Web or application server.

- To set up the administrative console so that you can use it to manage the Web server plug-in configuration, you must first create a default Web server configuration and then use the administrative console to add the plug-in properties from your migrated plugin-cfg.xml file to this Web server configuration.
  - To create a default Web server configuration and then add the plug-in properties from your migrated plugin-cfg.xml file in a standalone application server configuration, perform the following tasks:
    1. Create a default Web server configuration.

       Read the ″Selecting a Web server topology diagram and roadmap″ article or the ″Setting up a remote Web server″ article in the information center for more information.
    2. Use the Version 7.0 administrative console to edit the configuration and define the plug-in properties.

       Read the ″Communicating with Web servers″ and ″Web server plug-in configuration properties″ articles in the information center for more information.

Migrating from WebSphere Application Server Version 6.x: This information is only applicable if you migrated from WebSphere Application Server Version 6.x; it is not applicable if you migrated from Version 5.1.x.

- Only Web servers defined on managed nodes are migrated to WebSphere Application Server Version 7.0.
- If you are migrating a Web server and plug-ins from WebSphere Application Server Version 6.x to Version 7.0 and the Web server is defined on an unmanaged node, the Web server creation and application mapping must be done manually.

  To create the Web server definition manually, perform one of the following actions:
  - Use the administration console wizard.

    To generate mapping to all applications that are installed at Web server creation, use the mapping `ALL` option in the wizard.
  - Use the wsadmin command.

    ```
    $AdminTask createWebServer -interactive
    ```

    and reply `ALL` to the mapping applications prompt.
  - Use the configureWebserverDefintion.jacl script.

    This script maps all installed applications to the Web server. The script updates all of the information related to the Web server plug-in such as the locations of the plug-in installation root, log file, configuration file, and key stores on the Web server system. However, the script does not update other properties related to the Web server if the Web server definition already exists.

Chapter 5. Migrating administrative scripts

You can migrate administrative scripts using scripting and the wsadmin tool.

About this task

WebSphere Application Server Version 7.0 supports migrating administrative scripts from Version 5.1.x and Version 6.x.

- If you are migrating administrative scripts from Version 5.1.x, see:

  "Migrating administrative scripts from Version 5.1.x"
- If you are migrating administrative scripts from Version 6.x, see:

  "Migrating administrative scripts from Version 6.x to Version 7.0" on page 43

Migrating administrative scripts from Version 5.1.x

There are some changes you should be aware of when migrating from WebSphere Application Server Version 5.1.x.

About this task

There are a few changes to be aware of that are required for your existing scripts when moving to WebSphere Application Server Version 7.0. In general, the administration model has changed very little. However, there are some changes required when moving from Version 5.1.x to Version 7.0.

- Be aware of the implications of migrating JMS applications from the embedded messaging in WebSphere Application Server Version 5.1.x to the default messaging provider in WebSphere Application Server Version 7.0.
- A new version of Jacl (1.3.2) was shipped beginning with WebSphere Application Server Version 6.x. With this Jacl version, regexp command supports only Tcl 8.0 regexp command syntax. If your existing Version 5.1.x Jacl script uses the regexp command syntax that is supported in Jacl 1.2.6 but not anymore in Jacl 1.3.2, you might not get a match anymore or you might get a compile error for your regexp command similar to the following:

  ```
  com.ibm.bsf.BSFException: error while eval'ing Jacl expression:
  couldn't compile regular expression pattern: ?+* follows nothing
       while executing
  "regexp {(?x)
       ..."
       ("if" test expression)
       invoked from within
  "if {[regexp {(?x)
       ..."
       (file testregexp.jacl line 2)
       (file line 2)
       invoked from within
  "source testregexp.jacl"
  ```

  There is no workaround for this regression problem. Jacl has indicated that this is by design and there is no simple patch to fix this design decision.
- For WSADMIN $AdminConfig: The PME CacheInstanceService type is no longer used. If your scripts contain code to set the **enable** attribute on the CacheInstanceService type, remove the code. It is not needed in Version 7.0.
- There are a few changes to be aware of that are required for your existing Version 5.1.x scripts when moving to WebSphere Application Server Version 7.0. These types of changes can be evolved directly without the assistance of script compatibility support. The data can be accessed from multiple locations, including the old and new locations. As long as the new location is not updated, the data is accessed

from the old location. Once the new location is updated, it becomes the current data and is used for further accesses and updates. Warning messages are logged when the old location is still being used.

Read “Example: Migrating - Changing transaction log directory using scripting” on page 41 for more information.

- There are a few changes to be aware of that are required for your existing scripts when moving from Version 5.1.x to WebSphere Application Server Version 7.0. These changes are assisted by the compatibility mode provided by “WASPostUpgrade command” on page 23. During migration, the default is to migrate using compatibility mode. If this option is taken, then the old object types are migrated into the new configuration; all existing scripts will run unchanged.

  See the “Example: Migrating - Changing process definitions using scripting” on page 42 article for more information.

- Be aware of removed features that might have an impact on administration scripts.

  These might include the following:

  - Support for the Secure Authentication Service (SAS) IIOP security protocol
  - Support for the Common Connector Framework (CCF)
  - Support for the IBM Cloudscape Version 5.1.x database
  - Support for the following Java Database Connectivity (JDBC) drivers:
    - WebSphere Connect JDBC driver
    - Microsoft SQL Server 2000 Driver for JDBC
    - WebSphere SequeLink JDBC driver for Microsoft SQL Server

    Read “Migrating from the WebSphere Connect JDBC driver” on page 32 for information on using the WebSphereConnectJDBCDriverConversion command to convert data sources from the WebSphere Connect JDBC driver to the DataDirect Connect JDBC driver or the Microsoft SQL Server 2005 JDBC driver.

  For more information, see the ″Deprecated and removed features″ article in the information center.

Example: Migrating - Allowing configuration overwrite when saving a configuration

These examples demonstrate how to enable configuration overwrite in network deployment for WebSphere Application Server Version 5.1.x and Version 6.x.

Use the following examples:

- wsadmin Version 5.1.x

  Using Jacl:

  ```
  $AdminConfig setSaveMode overwriteOnConflict
  ```

  Using Jython:

  ```
  AdminConfig.setSaveMode('overwriteOnConflict')
  ```

- wsadmin Version 6.x

  1. Enable configuration repository to allow configuration overwrite:

     Using Jacl:

     ```
     set s1AdminService [$AdminConfig getid /Server:dmgr/AdminService:/]

     set configRepository [$AdminConfig showAttribute $s1AdminService configRepository]
     set props [$AdminConfig showAttribute $configRepository properties]
     set foundAllowConfigOverwrites ""
     if {$props != "{}"} {
      foreach prop $props {
       if {[$AdminConfig showAttribute $prop name] == "allowConfigOverwrites"} {
        set foundAllowConfigOverwrites $prop
        break
       }
      }
     ```

```
}

if {$foundAllowConfigOverwrites == ""} {
 $AdminConfig create Property $configRepository {{name allowConfigOverwrites} {value true}}
} else {
 $AdminConfig modify $foundAllowConfigOverwrites {{value true}}
}

$AdminConfig save
```

Using Jython:

```
s1AdminService = AdminConfig.getid('/Server:dmgr/AdminService:/')
configRepository = AdminConfig.showAttribute(s1AdminService, 'configRepository')
props = AdminConfig.showAttribute(configRepository, 'properties')
foundAllowConfigOverwrites = ''
if props != '[]':
 properties = props[1:len(props)-1].split(' ')
 for prop in properties:
  name = AdminConfig.showAttribute(prop, 'name')
  if name == 'allowConfigOverwrites':
   foundAllowConfigOverwrites = prop
   break

if len(foundAllowConfigOverwrites) != 0:
 AdminConfig.modify(foundAllowConfigOverwrites, [['value', 'true']])
else:
 AdminConfig.create('Property', configRepository, [['name', 'allowConfigOverwrites'], ['value', 'true']])

AdminConfig.save()
```

2. Restart the deployment manager. From the `bin` directory of the deployment manager profile, run the following:
3. Allow configuration overwrite, for example:

   Using Jacl:

   ```
   $AdminConfig setSaveMode overwriteOnConflict
   ```

   Using Jython:

   ```
   AdminConfig.setSaveMode('overwriteOnConflict')
   ```

Example: Migrating - Changing transaction log directory using scripting

Prepare for evolutionary changes without script compatibility support.

The location of the `transaction logs directory` attribute has changed from the `ApplicationServer::TransactionService` to the `ServerEntry::recoveryLogs`. As long as the new location is not used, the value from the old location will continue to be used. Scripts that modify the old location can still be used; that value will take effect until a value in the new location is set. The change to scripts to use the new location is as follows:

Old location:

- Using Jacl:

  ```
  set transService [$AdminConfig list TransactionService $server1]
  $AdminConfig showAttribute $transService transactionLogDirectory
  ```

New Location:

- Using Jython:

  ```
  AdminConfig.list("ServerEntry")

  # Select one entry from the list, e.g the entry for server1:
  serverEntryId = AdminConfig.getid("/ServerEntry:server1")
  serverEntry = AdminConfig.list("ServerEntry", serverEntryId)
  ```

```
recoveryLog = AdminConfig.showAttribute(serverEntry, "recoveryLog")
AdminConfig.showAttribute(recoveryLog, "transactionLogDirectory")
```

- Using Jacl:

  ```
  $AdminConfig list ServerEntry $node
  set serverEntry <select one of the ServerEntry from output of above command>
  set recoveryLog [$AdminConfig showAttribute $serverEntry recoveryLog]
  $AdminConfig showAttribute $recoveryLog transactionLogDirectory
  ```

Example: Migrating - Changing process definitions using scripting

Prepare for evolutionary changes with script compatibility support.

The following changes can be made with **with** script compatibility support.

- **HTTP transports:** the architecture uses the new channel framework. HTTP definitions are mapped on top of this support. When compatibility mode is chosen, the old HTTPTransport objects are migrated and mapped onto the channel architecture. Existing scripts can modify these objects and will run unchanged.
- **Process definition:** The name of this object is changed from processDef to processDefs. You can mitigate this change by using the compatibility mode mapping provided by the migration tools. The change to scripts to use the new location is as follows:
  - Old example:
    - Using Jacl:

      ```
      set processDef [$AdminConfig list JavaProcessDef $server1]
      set processDef [$AdminConfig showAttribute $server1 processDefinition]
      ```

      Using Jython:

      ```
      processDef = AdminConfig.list('JavaProcessDef', server1)
      print processDef
      ```
  - New example. Identify the process definition belonging to this server and assign it to the processDefs variable:
    - Using Jacl:

      ```
      set processDefs [$AdminConfig list JavaProcessDef $server1]
      set processDefs [$AdminConfig showAttribute $server1 processDefinitions]
      ```

      Using Jython:

      ```
      processDefs = AdminConfig.list('JavaProcessDef', server1)
      print processDefs
      ```

Example: Migrating - Modifying Web container port numbers

These examples demonstrate how to modify Web container HTTP transport ports for WebSphere Application Server Version 5.1.x and Version 6.x.

Use the following examples:

- wsadmin Version 5.1.x

  Using Jacl:

  ```
  set httpPort  7575
  set server [$AdminConfig getid /Cell:myCell/Node:myNode/Server:server1/]
  set transports [$AdminConfig list HTTPTransport $server]
  set transport [lindex $transports 0]
  set endPoint [$AdminConfig showAttribute $transport address]
  $AdminConfig modify $endPoint [list [list port $httpPort]]
  $AdminConfig save
  ```

  Using Jython:

  ```
  httpPort = 7575
  server = AdminConfig.getid("/Cell:myCell/Node:myNode/Server:server1/")
  transports = AdminConfig.list("HTTPTransport", server).split(java.lang.System.getProperty("line.separator"))
  ```

```
transport = transports[0]
endPoint = AdminConfig.showAttribute(transport, "address")
AdminConfig.modify(endPoint, [["port", httpPort]])
AdminConfig.save()
```

- wsadmin Version 6.x

  Using Jacl:

```
set serverNm server1
set newPort  7575
set node [$AdminConfig getid /Cell:myCell/Node:myNode/]
set TCS [$AdminConfig getid /Cell:myCell/Node:myNode/Server:server1/TransportChannelService:/]
set chains [$AdminTask listChains $TCS {-acceptorFilter WebContainerInboundChannel}]

foreach chain $chains {
   set channels [lindex [$AdminConfig showAttribute $chain transportChannels] 0]
   foreach channel $channels {
      if {[catch {set channelEndPointName [$AdminConfig showAttribute $channel endPointName]} result]} {
         # ignore the error as not all channel has endPointName attribute
      } else {
         set serverEntries [$AdminConfig list ServerEntry $node]
         foreach serverEntry $serverEntries {
            set sName [$AdminConfig showAttribute $serverEntry serverName]
            if {$sName == $serverNm} {
               set specialEndPoints [lindex [$AdminConfig showAttribute $serverEntry specialEndpoints] 0]
               foreach specialEndPoint $specialEndPoints {
                  set endPointNm [$AdminConfig showAttribute $specialEndPoint endPointName]
                  if {$endPointNm == $channelEndPointName} {
                     set ePoint [$AdminConfig showAttribute $specialEndPoint endPoint]
                     $AdminConfig modify $ePoint [list [list port $newPort]]
                     break
                  }
               }
            }
         }
      }
   }
}

$AdminConfig save
```

  Using Jython:

```
serverNm = "server1"
newPort = "7575"
node = AdminConfig.getid("/Cell:myCell/Node:myNode/")
TCS = AdminConfig.getid("/Cell:myCell/Node:myNode/Server:server1/TransportChannelService:/")
chains = AdminTask.listChains(TCS, "[-acceptorFilter WebContainerInboundChannel]").split(java.lang.System.getProperty("line.separator"))

for chain in chains:
   channelString = AdminConfig.showAttribute(chain, "transportChannels")
   channelList = channelString[1:len(channelString)-1].split(" ")
   for channel in channelList:
      try:
         channelEndPointName = AdminConfig.showAttribute(channel, "endPointName")
         serverEntries = AdminConfig.list("ServerEntry", node).split(java.lang.System.getProperty("line.separator"))
         for serverEntry in serverEntries:
            sName = AdminConfig.showAttribute(serverEntry, "serverName")
            if sName == serverNm:
               sepString = AdminConfig.showAttribute(serverEntry, "specialEndpoints")
               sepList = sepString[1:len(sepString)-1].split(" ")
               for specialEndPoint in sepList:
                  endPointNm = AdminConfig.showAttribute(specialEndPoint, "endPointName")
                  if endPointNm == channelEndPointName:
                     ePoint = AdminConfig.showAttribute(specialEndPoint, "endPoint")
                     AdminConfig.modify(ePoint, [["port", newPort]])
                     break
      except:
         # ignore the error as not all channel has endPointName attribute
         pass

AdminConfig.save()
```

Migrating administrative scripts from Version 6.x to Version 7.0

There are some changes you should be aware of when migrating your administrative scripts from WebSphere Application Server Version 6.x to Version 7.0.

- Be aware of removed features that might have an impact on administration scripts.

  These might include the following:

  – Support for the Secure Authentication Service (SAS) IIOP security protocol

  - Support for the Common Connector Framework (CCF)
  - Support for the IBM Cloudscape Version 5.1.x database
  - Support for the following Java Database Connectivity (JDBC) drivers:
    - WebSphere Connect JDBC driver
    - Microsoft SQL Server 2000 Driver for JDBC
    - WebSphere SequeLink JDBC driver for Microsoft SQL Server

    Read "Migrating from the WebSphere Connect JDBC driver" on page 32 for information on using the WebSphereConnectJDBCDriverConversion command to convert data sources from the WebSphere Connect JDBC driver to the DataDirect Connect JDBC driver or the Microsoft SQL Server 2005 JDBC driver.

  For more information, see the ″Deprecated and removed features″ article in the information center.
- Be aware of the change required when creating Service Integration Bus (SIB) objects.

  For more information about the createSIBus command, see the ″createSIBus command″ article in the information center.

Updating SSL configurations to Version 7.0 configuration definitions after migration

When migrating to Version 7.0, you can update the format for SSL configuration or you can continue to use the format of the earlier version. If you encounter errors with your existing administration scripts for SSL configurations, use this task to manually convert your SSL configuration to the Version 7.0 format.

Before you begin

About this task

When migrating to Version 7.0, you can use the "WASPreUpgrade command" on page 20 to save the configuration of your previously installed version into a migration-specific backup directory. When migration is complete, you can use the "WASPostUpgrade command" on page 23 to retrieve the saved configuration and WASPostUpgrade script to migrate your previous configuration. The **-scriptCompatibility** parameter for the WASPostUpgrade command is used to specify whether to maintain the Version 5.1.x or 6.x configuration definitions or to upgrade the format to Version 7.0 configuration definitions. If you used the default value, or `-scriptCompatibility true` when migrating, you do not need to perform this task. If you set the scriptCompatibility parameter to `false` during migration, you may notice that your existing administration scripts for SSL configurations do not work correctly. If this occurs, use this task to convert your Version 5.1.x or 6.x SSL configuration definitions to Version 7.0 This process creates a new SSL configuration based on the existing configuration.

Follow the steps below to modify the existing SSL configuration:

```
<repertoire xmi:id="SSLConfig_1" alias="Node02/DefaultSSLSettings">
<setting xmi:id="SecureSocketLayer_1" keyFileName="$install_root/etc/MyServerKeyFile.jks"
keyFilePassword="password" keyFileFormat="JKS" trustFileName="$install_root/etc/MyServerTrustFile.jks"
trustFilePassword="password" trustFileFormat="JKS" clientAuthentication="false" securityLevel="HIGH"
enableCryptoHardwareSupport="false">
<cryptoHardware xmi:id="CryptoHardwareToken_1" tokenType="" libraryFile="" password="{custom}"/>
<properties xmi:id="Property_6" name="com.ibm.ssl.protocol" value="SSL"/>
<properties xmi:id="Property_7" name="com.ibm.ssl.contextProvider" value="IBMJSSE2"/>
</setting>
</repertoire>
```

1. Create a key store that references the key store attributes in the old configuration.
   a. In the existing configuration, find the **keyFileName**, **keyFilePassword**, and **keyFileFormat** attributes.

      ```
      keyFileName="${install_root}/etc/MyServerKeyFile.jks" keyFilePassword="password" keyFileFormat="JKS"
      ```

   b. Use the **keyFileName**, **keyFilePassword**, and **keyFileFormat** attributes to create a new KeyStore object. For this example, set the name as ″DefaultSSLSettings_KeyStore″.

      Using Jacl:

      ```
      $AdminTask createKeyStore {-keyStoreName DefaultSSLSettings_KeyStore -keyStoreLocation
      ${install_root}/etc/MyServerKeyFile.jks -keyStoreType JKS -keyStorePassword
      password -keyStorePasswordVerify password }
      ```

      The resulting configuration object in the security.xml file is:

      ```
      <keyStores xmi:id="KeyStore_1" name="DefaultSSLSettings_KeyStore" password="password"
      provider="IBMJCE" location="$install_root/etc/MyServerKeyFile.jks" type="JKS" fileBased="true"
      managementScope="ManagementScope_1"/>
      ```

      Note: If you specify the cryptoHardware values in your configuration, create the KeyStore object using these values instead. Associate the -keyStoreLocation parameter with the libraryFile attribute, the -keyStoreType parameter with the tokenType attribute, and the -keyStorePassword parameter with the password attribute.

      ```
      <cryptoHardware xmi:id="CryptoHardwareToken_1" tokenType="" libraryFile="" password=""/>
      ```

2. Create a trust store that references the trust store attributes from the existing configuration.

   a. Find the **trustFileName**, **trustFilePassword**, and **trustFileFormat** attributes in the existing configuration.

      ```
      trustFileName="$install_root/etc/MyServerTrustFile.jks" trustFilePassword="password"
      trustFileFormat="JKS"
      ```

   b. Use the **trustFileName**, **trustFilePassword**, and **trustFileFormat** attributes to create a new KeyStore object. For this example, set the name as ″DefaultSSLSettings_TrustStore″.

      Using Jacl:

      ```
      $AdminTask createKeyStore {-keyStoreName DefaultSSLSettings_TrustStore -keyStoreLocation
      $install_root/etc/MyServerTrustFile.jks -keyStoreType JKS -keyStorePassword password
      -keyStorePasswordVerify password }
      ```

      The resulting configuration object in the security.xml file is:

      ```
      <keyStores xmi:id="KeyStore_2" name="DefaultSSLSettings_TrustStore" password="password"
      provider="IBMJCE" location="$install_root/etc/MyServerTrustFile.jks" type="JKS" fileBased="true"
      managementScope="ManagementScope_1"/>
      ```

3. Create a new SSL configuration using the new key store and trust store. Include any other attributes from the existing configuration which are still valid.

   Use a new alias for your updated SSL configuration. You can not create an SSL configuration with the same name as your existing configuration.

   Using Jacl:

   ```
   $AdminTask createSSLConfig {-alias DefaultSSLSettings -trustStoreName DefaultSSLSettings_TrustStore
    -keyStoreName DefaultSSLSettings_KeyStore -keyManagerName IbmX509 -trustManagerName IbmX509
   -clientAuthentication true -securityLevel HIGH -jsseProvider IBMJSSE2 -sslProtocol SSL  }
   ```

Results

The new SSL configuration is:

```
<repertoire xmi:id="SSLConfig_1" alias="DefaultSSLSettings" managementScope="ManagementScope_1">
<setting xmi:id="SecureSocketLayer_1" clientAuthentication="true" securityLevel="HIGH" enabledCiphers=""
jsseProvider="IBMJSSE2" sslProtocol="SSL" keyStore="KeyStore_1" trustStore="KeyStore_2"
trustManager="TrustManager_1" keyManager="KeyManager_1"/>
</repertoire>
```

Note: The default management scope is used if it is not specified.

Chapter 6. Coexisting

You can create an environment in which multiple versions of WebSphere Application Server can run independently on the same system at the same time. A major consideration in coexistence is the avoidance of port conflicts.

Before you begin

Coexisting, as it applies to WebSphere Application Server products, is running a new release of a WebSphere Application Server product on the same machine at the same time as you run an earlier release or running two installations of the same release of a WebSphere Application Server product on the same machine at the same time.

Read "Coexistence support."
This article discusses which coexistence scenarios are supported.

Coexistence support

Coexistence is a state in which multiple installations and multiple nodes from different versions of WebSphere Application Server run independently in the same environment at the same time.

As it applies to WebSphere Application Server products, coexistence primarily refers to the ability of multiple installations of WebSphere Application Server to run independently on the same machine at the same time. Multiple installations include multiple versions and multiple instances of one version. Coexistence also implies various combinations of Web server interaction.

WebSphere Application Server Version 7.0 products can coexist with the following supported versions:

- WebSphere Application Server Version 5.1.x
- WebSphere Application Server Network Deployment Version 5.1.x
- WebSphere Application Server Version 6.x
- WebSphere Application Server Network Deployment Version 6.x

All combinations of Version 5.1.x products, Version 6.x products, and Version 7.0 products can coexist so long as there are no port conflicts.

WebSphere Application Server Version 5.1.x and Version 6.x clients can coexist with Version 7.0 clients.

Table 2. WebSphere Application Server Version 5.1.x, Version 6.x, and Version 7.0 clients multiversion coexistence scenarios

Installed product	WebSphere Application Server Version 7.0 clients
WebSphere Application Server Version 5.1.x	Supported
WebSphere Application Server Network Deployment Version 5.1.x	Supported
WebSphere Application Server Version 6.x	Supported
WebSphere Application Server Network Deployment Version 6.x	Supported

WebSphere Application Server Version 5.1.x and Version 6.x products can coexist with Version 7.0 products.

Table 3. WebSphere Application Server Version 5.1.x, Version 6.x, and Version 7.0 multiversion coexistence support

Installed product	WebSphere Application Server Version 5.1.x			WebSphere Application Server Version 6.x			WebSphere Application Server Version 7.0		
	Application Server	**Network Deployment**	**Express**	**Application Server**	**Network Deployment**	**Express**	**Application Server**	**Network Deployment**	**Express**
WebSphere Application Server Version 5.1.x	Supported	Supported	Supported	Supported	Supported	Supported	Supported	Supported	Supported
WebSphere Application Server Network Deployment Version 5.1.x	Supported	Supported	Supported	Supported	Supported	Supported	Supported	Supported	Supported
WebSphere Application Server Integration Server Version 5.1.x	Supported	Supported	Supported	Supported	Supported	Supported	Supported	Supported	Supported
WebSphere Application Server clients Version 5.1.x	Supported	Supported	Supported	Supported	Supported	Supported	Supported	Supported	Supported
WebSphere Application Server Version 6.x	Supported	Supported	Supported	Supported	Supported	Supported	Supported	Supported	Supported
WebSphere Application Server Network Deployment Version 6.x	Supported	Supported	Supported	Supported	Supported	Supported	Supported	Supported	Supported
WebSphere Application Server clients Version 6.x	Supported	Supported	Supported	Supported	Supported	Supported	Supported	Supported	Supported
WebSphere Application Server Version 7	Supported	Supported	Supported	Supported	Supported	Supported	Supported	Supported	Supported
WebSphere Application Server Network Deployment Version 7.0	Supported	Supported	Supported	Supported	Supported	Supported	Supported	Supported	Supported
WebSphere Application Server clients Version 7.0	Supported	Supported	Supported	Supported	Supported	Supported	Supported	Supported	Supported

In addition to multiversion coexistence, WebSphere Application Server also lets you install multiple times on one machine (multiple installation instances) or install once and have multiple profiles. Multiple Version 7.0 installation instances on one machine include the following combinations:

- Multiple application server profiles from multiple installations of the WebSphere Application Server product
- Multiple application server profiles from a single installation of the WebSphere Application Server product

Chapter 7. Interoperating

WebSphere Application Server Version 7.0 is interoperable with other versions of the product under certain conditions.

Before you begin

Read “Overview of migration, coexistence, and interoperability” on page 1 and “Premigration considerations” on page 4. For resources to help you plan and perform your migration, visit Knowledge Collection: Migration planning for WebSphere Application Server.

WebSphere Application Server Version 7.0 is generally interoperable with Version 5.1.x and Version 6.x. However, there are specific requirements to address for each version. In general, you should be at the most recent group PTF level to support interoperability.

You can upgrade a portion of the nodes in a cell to WebSphere Application Server Version 7.0 while leaving others at the older release level. This means that, for a period of time, you might be administering servers that are at the current release and servers that are running the newer release in the same cell.

1. Follow the required guidelines for WebSphere Application Server Version 5.1.x.

   Guideline 1:

   Be aware of the level of WebSphere Application Server in which each function you use is supported. Applications that you intend to be interoperable must only use function that is supported by all levels of WebSphere Application Server in the cluster. For example, applications that use the `commonj.timer.TimerManager` resource, which was new in Version 6.0, should not be deployed to a cluster including both Version 5.1.x and Version 7.0 servers.

   Guideline 2:

   If you run related cross-domain interoperating applications (one server is in rtp.raleigh.ibm.com and the other is in cn.ibm.com for example), you need to use fully qualified host names (host9.rtp.raleigh.ibm.com instead of just host9 for example) when installing WebSphere Application Server Version 7.0.
2. Upgrade the Software Development Kit (SDK) used to one supported by Version 7.0.

   Read Recommended fixes for WebSphere Application Server for more information.

What to do next

This information is dynamic and might be augmented by information in technical articles that are available on the IBM DeveloperWorks WebSphere site. Check the site for the latest information.

Chapter 8. Configuring ports

When you configure WebSphere Application Server resources or assign port numbers to other applications, you must avoid conflicts with other assigned ports. In addition, you must explicitly enable access to particular port numbers when you configure a firewall.

Before you begin

For more information about port numbers that your iSeries system currently uses, enter the NETSTAT *CNN command on the command line. Press **F14** to view assigned port numbers.

You can also use the port validator tool to find port conflicts between different WebSphere Application Server profiles, products, and servers. Read the ″port validator tool″ article in the information center for more information.

1. Review the port number settings, especially when you are planning to coexist.

   You can use the dspwasinst command-line tool to display the port information for a profile. Read the ″dspwasinst command″ article in the information center for more information.

   You can use the dspwasinst command-line tool to display the port information for a profile. See the information center.
2. Optional: Change the port number settings.

   You can set port numbers when configuring the product after installation.

   - During profile creation using the manageprofiles command, you can accept the default port values or you can specify your port settings. If you want to specify ports, you can do so in any of the following ways:
     - Specify the use of a port file that contains the port values.
     - Specify the use of a starting port value.
     - Specify the use of the default port values.

     Read the ″manageprofiles command″ article in the information center for more information.
   - You can use the chgwassvr command to change the ports for an application server within a profile.

     Read the ″chgwassvr command″ article in the information center for more information.

Port number settings in WebSphere Application Server versions

You should be able to identify the default port numbers used in the various versions of WebSphere Application Server so that you can avoid port conflicts if you plan for an earlier version to coexist or interoperate with Version 7.0.

When you configure WebSphere Application Server resources or assign port numbers to other applications, you must avoid conflicts with other assigned ports. In addition, when you configure a firewall, you must explicitly enable access to particular port numbers.

If ports are already defined in a configuration being migrated, the migration tools fix the port conflicts in the Version 7.0 configuration and log the changes for your verification .

Note:

- When you install WebSphere Application Server, the default instance is created with the default port values. When you create a WebSphere Application Server profile with the manageprofiles script, you can specify different port values.
- For more information about port numbers that your iSeries system currently uses, enter the NETSTAT *CNN command on the CL command line. Press F14 to view assigned port numbers.

- You can also use the port validator tool to find port conflicts between different WebSphere Application Server profiles, products, and servers. Read the ″port validator tool″ article in the information center for more information.

Version 7.0 port numbers

Table 4. Port definitions for WebSphere Application Server Version 7.0

Port Name	Default Value					Files
	Application Server	Administrative Agent	Job Manager	Secure Proxy Server	Administrative Subsystem	
Administrative Console Port (WC_ adminhost)	9060	9060	9960	----	----	serverindex.xml and virtualhosts.xml
Administrative Console Secure Port (WC_ adminhost_ secure)	9043	9043	9943	----	----	
HTTP Transport Port (WC_ defaulthost)	9080	----	----	80	----	
HTTPS Transport Secure Port (WC_ defaulthost_ secure)	9443	----	----	443	----	
Bootstrap Port (BOOTSTRAP_ ADDRESS)	2809	9807	9808	----	----	serverindex.xml
Cell Discovery Address (CELL_ DISCOVERY_ ADDRESS)	----	----	----	----	----	
CSIV2 Client Authentication Listener Port (CSIV2_ SSL_ MUTUALAUTH_ LISTENER_ ADDRESS)	9402	9402	9402	----	----	
CSIV2 Server Authentication Listener Port (CSIV2_ SSL_ SERVERAUTH_ LISTENER_ ADDRESS)	9403	9403	9403	----	----	
High Availability Manager Communication Port (DCS_ UNICAST_ ADDRESS)	9353	----	----	----	----	
Internal JMS Server Port (JMSSERVER_ SECURITY_ PORT)	5557	----	----	----	----	
IPC Connector Port (IPC_ CONNECTOR_ ADDRESS)	9633	9630	9631	9633	9634	
MQ Transport Port (SIB_ MQ_ ENDPOINT_ ADDRESS)	5558	----	----	----	----	
MQ Transport Secure Port (SIB_ MQ_ ENDPOINT_ SECURE_ ADDRESS)	5578	----	----	----	----	

Table 4. Port definitions for WebSphere Application Server Version 7.0 (continued)

Port Name	Default Value					Files
	Application Server	Administrative Agent	Job Manager	Secure Proxy Server	Administrative Subsystem	
ORB Listener Port (ORB_ LISTENER_ ADDRESS)	9100	9098	9099	----	----	serverindex.xml
RMI Connector Port (RMI_ CONNECTOR_ ADDRESS)	----	----	----	----	9810	
JSR 160 RMI Connector Port (JSR160RMI_ CONNECTOR_ ADDRESS)	----	----	----	----	9811	
SAS_ SSL_ SERVERAUTH_ LISTENER_ ADDRESS	9401	9401	9401	----	----	
Service Integration Port (SIB_ ENDPOINT_ ADDRESS)	7276	----	----	----	----	
Service Integration Secure Port (SIB_ ENDPOINT_ SECURE_ ADDRESS)	7286	----	----	----	----	
SIP Container Port (SIP_ DEFAULTHOST)	5060	----	----	5060	----	
SIP Container Secure Port (SIP_ DEFAULTHOST_ SECURE)	5061	----	----	5061	----	
SOAP Connector Port (SOAP_ CONNECTOR_ ADDRESS)	8880	8877	8876	----	8881	
IBM HTTP Server Port	80	----	----	----	----	virtualhosts.xml, plugin-cfg.xml, and *web_ server_ root*/conf/ httpd.conf
IBM HTTPS Server Administration Port	8008	----	----	----	----	*web_ server_ root*/conf/ admin.conf

Version 6.1 port numbers

Table 5. Port definitions for WebSphere Application Server Version 6.1

Port Name	Default Value	Files
Administrative Console Port (WC_ adminhost)	9060	serverindex.xml and virtualhosts.xml
Administrative Console Secure Port (WC_ adminhost_ secure)	9043	
HTTP_ Transport Port (WC_ defaulthost)	9080	
HTTPS Transport Port (WC_ defaulthost_ secure)	9443	

Table 5. Port definitions for WebSphere Application Server Version 6.1 (continued)

Port Name	Default Value	Files
Bootstrap Port (BOOTSTRAP_ ADDRESS)	2809	serverindex.xml
CSIV2 Server Authentication Port (CSIV2_ SSL_ SERVERAUTH_ LISTENER_ ADDRESS)	9403	
CSIV2 Client Authentication Listener Port (CSIV2_ SSL_ MUTUALAUTH_ LISTENER_ ADDRESS)	9402	
DRS_ CLIENT_ ADDRESS **Note:** This port is deprecated and is no longer used in the current version of WebSphere Application Server.	7873	
High Availability Manager Communication Port (DCS_ UNICAST_ ADDRESS)	9353	
IBM HTTP Server Administration Port	2001	
Internal JMS Server (JMSSERVER_ SECURITY_ PORT)	5557	
MQ Transport (SIB_ MQ_ ENDPOINT_ ADDRESS)	5558	
MQ Transport secure (SIB_ MQ_ ENDPOINT_ SECURE_ ADDRESS)	5578	
ORB Listener Port (ORB_ LISTENER_ ADDRESS)	9100	
Proxy Server Port (PROXY_ HTTP_ ADDRESS)	80	
Proxy Server Secure Port (PROXY_ HTTPS_ ADDRESS)	443	
SAS_ SSL_ SERVERAUTH_ LISTENER_ ADDRESS	9401	
Service Integration Port (SIB_ ENDPOINT_ ADDRESS)	7276	
Service Integration Port Secure (SIB_ ENDPOINT_ SECURE_ ADDRESS)	7286	
SIP Container Port (SIP_ DEFAULTHOST)	5060	
SIP Container Secure Port (SIP_ DEFAULTHOST_ SECURE)	5061	
SOAP Connector Port (SOAP_ CONNECTOR_ ADDRESS)	8880	
IBM HTTP Server Port	80	virtualhosts.xml, plugin-cfg.xml, and *web_ server_ root*/conf/ httpd.conf

Version 6.0.x port numbers

Table 6. Port definitions for WebSphere Application Server Version 6.0.x

Port Name	Default Value	Files
Web Container Port (WC_ defaulthost)	9080	server.xml, plugin-cfg.xml, and virtualhosts.xml
Web Container Secure Port (WC_ defaulthost_ secure) **Note:** If you change this port number, remember the following information: • To use secure (SSL-enabled) ports you must have the i5/OS Digital Certificate Manager product (5722SS1 option 34) and a Cryptographic Access Provider product (such as 5722AC3) installed. • If you change this port, you must regenerate the Web server plug-in configuration for the application server.	9443	
Administrative Console Port (WC_ adminhost)	9060	server.xml and virtualhosts.xml
Administrative Console Secure Port (WC_ adminhost_ secure)	9043	

Table 6. Port definitions for WebSphere Application Server Version 6.0.x (continued)

Port Name	Default Value	Files
Name Service or RMI Connector Port (BOOTSTRAP_ ADDRESS)	2809	serverindex.xml
SOAP Port (SOAP_ CONNECTOR_ ADDRESS)	8880	
SAS_ SSL_ SERVERAUTH_ LISTENER_ ADDRESS	9501	
Common Secure Interoperability Version 2 (CSIV2) Server Transport Port (CSIV2_ SSL_ SERVERAUTH_ LISTENER_ ADDRESS)	9503	
CSIV2 Client Transport Port (CSIV2_ SSL_ MUTUALAUTH_ LISTENER_ ADDRESS)	9502	
Object Request Broker (ORB) Listener Port (ORB_ LISTENER_ ADDRESS)	----	
Java Message Service (JMS) Queued Port (JMSSERVER_ QUEUED_ ADDRESS)	5558	
JMS Direct Port (JMSSERVER_ DIRECT_ ADDRESS)	5559	
JMS Security Port (JMSSERVER_ SECURITY_ PORT)	5557	
Data Replication Service Client Port (DRS_ CLIENT_ ADDRESS) **Note:** This port is deprecated and is no longer used in the current version of WebSphere Application Server.	7873	
IBM HTTP Server Port	80	virtualhosts.xml, plugin-cfg.xml, and *web_ server_ root*/conf/ httpd.conf
IBM HTTP Server Administration Port	2001	serverindex.xml
NODE_ MULTICAST_ IPV6_ DISCOVERY_ ADDRESS	5001	

Version 5.1.x port numbers

Table 7. Port definitions for WebSphere Application Server Version 5.1.x

Port Name	Default Value	Files
HTTP_ TRANSPORT	9080	server.xml and virtualhosts.xml
HTTPS_ TRANSPORT	9443	
HTTP_ TRANSPORT_ ADMIN	9090	
HTTPS_ TRANSPORT_ ADMIN	9043	
JMSSERVER_ SECURITY_ PORT	5557	server.xml
JMSSERVER_ QUEUED_ ADDRESS	5558	serverindex.xml
JMSSERVER_ DIRECT_ ADDRESS	5559	
BOOTSTRAP_ ADDRESS	2809	
SOAP_ CONNECTOR_ ADDRESS	8880	
DRS_ CLIENT_ ADDRESS	7873	
SAS_ SSL_ SERVERAUTH_ LISTENER_ ADDRESS	0	
CSIV2_ SSL_ SERVERAUTH_ LISTENER_ ADDRESS	0	
CSIV2_ SSL_ MUTUALAUTH_ LISTENER_ ADDRESS	0	
ORB_ LISTENER_ ADDRESS	0	
IBM HTTP Server Port	80	virtualhosts.xml, plugin-cfg.xml, and *web_ server_ root*/conf/ httpd.conf
IBM HTTPS Server Administration Port	2001	----

Chapter 9. Troubleshooting migration

You might encounter problems while migrating from an older version of WebSphere Application Server.

- If you encounter a problem when you are migrating from a previous version of WebSphere Application Server to Version 7.0, check your log files and other available information.
  1. Look for the log files, and browse them for clues.
     – *migration_backup_dir*/logs/WASPreUpgrade.*time_stamp*.log
     – *migration_backup_dir*/logs/WASPostUpgrade.*time_stamp*.log
     – *app_server_root*/logs/clientupgrade.*time_stamp*.log
  2. Look for `MIGR0259I: The migration has successfully completed` or `MIGR0271W: The migration completed with warnings` in one of the log files.
     – *migration_backup_dir*/logs/WASPreUpgrade.*time_stamp*.log
     – *migration_backup_dir*/logs/WASPostUpgrade.*time_stamp*.log
     – *app_server_root*/logs/clientupgrade.*time_stamp*.log

     If `MIGR0286E: The migration failed to complete` is displayed, attempt to correct any problems based on the error messages that appear in the log file. After correcting any errors, rerun the command from the bin directory of the product installation root.
  3. Open the service log of the server that is hosting the resource that you are trying to access, and browse error and warning messages.
  4. With WebSphere Application Server running, run the dumpNameSpace command and pipe, redirect, or ″more″ the output so that it can be easily viewed.

     This command results in a display of all objects in WebSphere Application Server's namespace, including the directory path and object name.
  5. If the object that a client needs to access does not appear, use the administrative console to verify the following conditions.
     – The server hosting the target resource is started.
     – The Web module or enterprise Java bean container hosting the target resource is running.
     – The JNDI name of the target resource is properly specified.

  For current information available from IBM Support on known problems and their resolution, read the IBM Support page. IBM Support also has documents that can save you time gathering information needed to resolve this problem. Before opening a PMR, read the IBM Support page.
- During the migration process, problems might occur while you are using the WASPreUpgrade tool or the WASPostUpgrade tool.
  – Problems can occur when you are using the WASPreUpgrade tool.
    - A ″Not found″ or ″No such file or directory″ message is returned.

      This problem can occur if you are trying to run the WASPreUpgrade tool from a directory other than the WebSphere Application Server Version 7.0 *app_server_root*\bin. Verify that the WASPreUpgrade script resides in the Version 7.0 *app_server_root*\bin directory, and launch the file from that location.
    - The DB2® JDBC driver and DB2 JDBC driver (XA) cannot be found in the drop-down list of supported JDBC providers in the administrative console.

      The administrative console no longer displays deprecated JDBC provider names. The new JDBC provider names used in the administrative console are more descriptive and less confusing. The new providers will differ only by name from the deprecated ones.

      The deprecated names will continue to exist in the jdbc-resource-provider-templates.xml file for migration reasons (for existing JACL scripts for example); however, you are encouraged to use the new JDBC provider names in your JACL scripts.
    - You receive the following message:

```
MIGR0108E: The specified WebSphere directory does not contain a WebSphere version that can be upgraded.
```

The following possible reasons for this error exist:

- If WebSphere Application Server Version 5.1.x or Version 6.x is installed, you might not have run the WASPreUpgrade tool from the bin directory of the Version 7.0 installation root.
  1. Look for something like the following message to display when the WASPreUpgrade tool runs: `IBM WebSphere Application Server, Release 5.1.`

     This message indicates that you are running the WebSphere Application Server Version 5.1 migration utility, not the Version 7.0 migration utility.
  2. Alter your environment path or change the current directory so that you can launch the WebSphere Application Server Version 7.0 WASPreUpgrade tool.
- An invalid directory might have been specified when launching the WASPreUpgrade tool.

Read “WASPreUpgrade command” on page 20 for more information.

– Problems can occur when you are using the WASPostUpgrade tool.

- You might receive the following error messages:

  ```
  MIGR0484E: No profiles or instances found with name -profileName wasio2651.
  MIGR0272E: The migration function cannot complete the command.
  ```

  The old and new profile names must match. Rerun the WASPostUpgrade command with a Version 7.0 profile that matches the name given as the value following -profileName in the MIGR0484E message.
- A ″Not found″ or ″No such file or directory″ message is returned.

  This problem can occur if you are trying to run the WASPostUpgrade tool from a directory other than the WebSphere Application Server Version 7.0 *app_server_root*\bin. Verify that the WASPostUpgrade script resides in the Version 7.0 *app_server_root*\bin directory, and launch the file from that location.
- You receive the following message:

  ```
  MIGR0102E: Invalid Command Line. MIGR0105E: You must specify the primary node name.
  ```

  The most likely cause of this error is that WebSphere Application Server Version 5.1.x or Version 6.x is installed and the WASPostUpgrade tool was not run from the bin directory of the Version 7.0 installation root.

  To correct this problem, run the WASPostUpgrade command from the bin directory of the WebSphere Application Server Version 7.0 installation root.
- You receive the ″Unable to copy document to temp file″ error message. Here is an example:

  ```
  MIGR0304I: The previous WebSphere environment is being restored.
  com.ibm.websphere.management.exception.DocumentIOException: Unable to copy document to temp file:
    cells/sunblade1Network/applications/LARGEApp.ear/LARGEApp.ear
  ```

  Your file system might be full. If your file system is full, clear some space and rerun the WASPostUpgrade command.
- You receive the following message:

  ```
  MIGR0108E: The specified WebSphere directory does not contain WebSphere version that can be upgraded.
  ```

  The following possible reasons for this error exist:

  - If WebSphere Application Server Version 5.1.x or Version 6.x is installed, you might not have run the WASPostUpgrade tool from the bin directory of the Version 7.0 installation root.
    1. Look for something like the following message to display when the WASPostUpgrade tool runs: `IBM WebSphere Application Server, Release 5.1.`

       This message indicates that you are running the WebSphere Application Server Release 5.1 migration utility, not the Version 7.0 migration utility.
    2. Alter your environment path or change the current directory so that you can launch the WebSphere Application Server Version 7.0 WASPostUpgrade tool.
  - An invalid directory might have been specified when launching the WASPreUpgrade tool or the WASPostUpgrade.

        - The WASPreUpgrade tool was not run.
    - You receive the following error message:

      ```
      MIGR0253E: The backup directory migration_backup_directory does not exist.
      ```

      The following possible reasons for this error exist:

      - The WASPreUpgrade tool was not run before the WASPostUpgrade tool.
        1. Check to see if the backup directory specified in the error message exists.
        2. If not, run the WASPreUpgrade tool.

           Read "WASPreUpgrade command" on page 20 for more information.
        3. Retry the WASPostUpgrade tool.
      - An invalid backup directory might be specified.

        For example, the directory might have been a subdirectory of the Version 5.1.x or Version 6.x tree that was deleted after the WASPreUpgrade tool was run and the older version of the product was uninstalled but before the WASPostUpgrade tool was run.
        1. Determine whether or not the full directory structure specified in the error message exists.
        2. If possible, rerun the WASPreUpgrade tool, specifying the correct full migration backup directory.
        3. If the backup directory does not exist and the older version it came from is gone, rebuild the older version from a backup repository or XML configuration file.
        4. Rerun the WASPreUpgrade tool.
    - Version 7.0 applications fail to install.

      Manually install the applications using the wsadmin command after WASPostUpgrade has completed.

      To manually install an application that failed to install during migration, use the wsadmin command to run the install_*application_name*.jacl script that the migration tools created in the backup directory.

      In a Linux® environment, for example, use the following parameters:

      ```
      ./wsadmin.sh -f migration_backup_directory/install_application_name.jacl -conntype NONE
      ```

      Use the following parameters:

      ```
      app_server_root/bin/wsadmin -f migration_backup_directory/install_application_name.jacl -conntype NONE
      ```

    Read "WASPostUpgrade command" on page 23 for more information.

- If you select the option for the migration process to install the enterprise applications that exist in the Version 5.1.x or Version 6.x configuration into the new Version 7.0 configuration, you might encounter some error messages during the application-installation phase of migration.

  The applications that exist in the Version 5.1.x or Version 6.x configuration might have incorrect deployment information—typically, invalid XML documents that were not validated sufficiently in previous WebSphere Application Server runtimes. The runtime now has an improved application-installation validation process and will fail to install these malformed EAR files. This results in a failure during the application-installation phase of WASPostUpgrade and produces an ″E:″ error message. This is considered a ″fatal″ migration error.

  If migration fails in this way during application installation, you can do one of the following:

  – Fix the problems in the Version 5.1.x or Version 6.x applications, and then remigrate.

  – Proceed with the migration and ignore these errors.

    In this case, the migration process does not install the failing applications but does complete all of the other migration steps.

    Later, you can fix the problems in the applications and then manually install them in the new Version 7.0 configuration using the administrative console or an install script.

- Because of the inclusion of the javax.ejb.Remote annotation in the EJB 3.0 specification, certain EJB 2.1 beans might fail to compile if Enterprise Java Beans are written to import the entire javax.ejb and java.rmi packages. Compilation errors similar to those in the following example might occur:

```
ejbModule/com/ibm/websphere/samples/trade/MarketSummaryDataBean.java(17):
  The serializable class MarketSummaryDataBean does not declare a static final serialVersionUID
  field of type long

ejbModule/com/ibm/websphere/samples/trade/HoldingDataBean.java(17):
  The serializable class HoldingDataBean does not declare a static final serialVersionUID
  field of type long

ejbModule/com/ibm/websphere/samples/trade/AccountProfileDataBean.java(14):
  The serializable class AccountProfileDataBean does not declare a static final serialVersionUID
  field of type long

ejbModule/com/ibm/websphere/samples/trade/AccountDataBean.java(17):
  The serializable class AccountDataBean does not declare a static final serialVersionUID
  field of type long
```

What to do next

If you did not find your problem listed, contact IBM support.

Appendix. Directory conventions

References in product information to *app_server_root*, *profile_root*, and other directories infer specific default directory locations. This topic describes the conventions in use for WebSphere Application Server.

Default product locations (i5/OS)

These file paths are default locations. You can install the product and other components in any directory where you have write access. You can create profiles in any valid directory where you have write access. Multiple installations of WebSphere Application Server products or components require multiple locations.

app_client_root
: The default installation root directory for the Java EE WebSphere Application Client is the /QIBM/ProdData/WebSphere/AppClient/V7/client directory.

app_client_user_data_root
: The default Java EE WebSphere Application Client user data root is the /QIBM/UserData/WebSphere/AppClient/V7/client directory.

app_client_profile_root
: The default Java EE WebSphere Application Client profile root is the /QIBM/UserData/WebSphere/AppClient/V7/client/profiles/*profile_name* directory.

app_server_root
: The default installation root directory for WebSphere Application Server is the /QIBM/ProdData/WebSphere/AppServer/V7/Base directory.

cip_app_server_root
: The default installation root directory is the /QIBM/ProdData/WebSphere/AppServer/V7/Base/cip/*cip_uid* directory for a customized installation package (CIP) produced by the Installation Factory.

A CIP is a WebSphere Application Server product bundled with optional maintenance packages, an optional configuration archive, one or more optional enterprise archive files, and other optional files and scripts.

cip_profile_root
: The default profile root directory is the /QIBM/UserData/WebSphere/AppServer/V7/Base/cip/*cip_uid*/profiles/*profile_name* directory for a customized installation package (CIP) produced by the Installation Factory.

cip_user_data_root
: The default user data root directory is the /QIBM/UserData/WebSphere/AppServer/V7/Base/cip/*cip_uid* directory for a customized installation package (CIP) produced by the Installation Factory.

if_root
: This directory represents the root directory of the IBM WebSphere Installation Factory. Because you can download and unpack the Installation Factory to any directory on the file system to which you have write access, this directory's location varies by user. The Installation Factory is an Eclipse-based tool which creates installation packages for installing WebSphere Application Server in a reliable and repeatable way, tailored to your specific needs.

iip_root
: This directory represents the root directory of an *integrated installation package* (IIP) produced by the IBM WebSphere Installation Factory. Because you can create and save an IIP to any directory on the file system to which you have write access, this directory's location varies by user. An IIP is an aggregated installation package created with the Installation Factory that can include one or more generally available installation packages, one or more customized installation packages (CIPs), and other user-specified files and directories.

java_home
: The following directories are the root directories for all supported Java Virtual Machines (JVMs).

JVM	Directory
Classic JVM	/QIBM/ProdData/Java400/jdk6
32–bit IBM Technology for Java	/QOpenSys/QIBM/ProdData/JavaVM/jdk60/32bit
64–bit IBM Technology for Java	/QOpenSys/QIBM/ProdData/JavaVM/jdk60/64bit

plugins_profile_root
: The default Web server plug-ins profile root is the /QIBM/UserData/WebSphere/Plugins/V7/webserver/profiles/*profile_name* directory.

plugins_root
: The default installation root directory for Web server plug-ins is the /QIBM/ProdData/WebSphere/Plugins/V7/webserver directory.

plugins_user_data_root
: The default Web server plug-ins user data root is the /QIBM/UserData/WebSphere/Plugins/V7/webserver directory.

product_library
product_lib
: This is the product library for the installed product. The product library for each Version 7.0 installation on the system contains the program and service program objects (similar to .exe, .dll, .so objects) for the installed product. The product library name is QWAS7*x* (where *x* is A, B, C, and so on). The product library for the first WebSphere Application Server Version 7.0 product installed on the system is `QWAS7A`. The *app_server_root*/properties/product.properties file contains the value for the product library of the installation, `was.install.library`, and is located under the *app_server_root* directory.

profile_root
: The default directory for a profile named *profile_name* for WebSphere Application Server is the /QIBM/UserData/WebSphere/AppServer/V7/Base/profiles/*profile_name* directory.

shared_product_library
: The shared product library, which contains all of the objects shared by all installations on the system, is `QWAS7`. This library contains objects such as the product definition, the subsystem description, the job description, and the job queue.

updi_root
: The default installation root directory for the Update Installer for WebSphere Software is the /QIBM/ProdData/WebSphere/UpdateInstaller/V7/updi directory.

user_data_root
: The default user data directory for WebSphere Application Server is the /QIBM/UserData/WebSphere/AppServer/V7/Base directory.

The profiles and profileRegistry subdirectories are created under this directory when you install the product.

web_server_root
: The default web server path is /www/*web_server_name*.

Notices

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM's product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program, or service. Evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, is the user's responsibility.

IBM may have patents or pending patent applications covering subject matter 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 Intellectual Property & Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
USA

Trademarks and service marks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol ($^{®}$ or $^{™}$), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. For a current list of IBM trademarks, visit the IBM Copyright and trademark information Web site (www.ibm.com/legal/copytrade.shtml).

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

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

Java is a trademark of Sun Microsystems, Inc. in the United States, other countries, or both.

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