Common Event Infrastructure Developer's Guide

Version 6.0.0.3

Contents

Chapter 1. Introduction to the Common Event Infrastructure

The Common Event Infrastructure is an embeddable technology intended to provide basic event management services to applications that require those services. This event infrastructure serves as an integration point for consolidation and persistence of raw events from multiple, heterogeneous sources, and distribution of those events to event consumers. Events are represented using the Common Base Event model, a standard, XML-based format defining the structure of an event. For more information, see "The Common Base Event model" on page 2.

By using this common infrastructure, diverse products that are not tightly coupled with one another can integrate their management of events, providing an end-to-end view of enterprise resources and correlating events across domain boundaries. For example, events generated by a network monitoring application can be correlated with events generated by a security application. Such correlation can be difficult to achieve when each product uses its own approach to event management.

The Common Event Infrastructure provides facilities for generation, propagation, persistence, and consumption of events, but it does not define the events themselves. Instead, application developers and administrators define event types, event groups, filtering, and correlation.

Common Event Infrastructure components

The Common Event Infrastructure consists of the following major components:

Common Base Event
The Common Base Event component supports the creation of events and access to their property data. Event sources use the Common Base Event APIs to create new events conforming to the Common Base Event model (see "The Common Base Event model" on page 2); event consumers use the APIs to read property data from received events. In addition, applications can convert events to and from XML text format, supporting interchange with other tools. The Common Base Event component is part of the Eclipse Test and Performance Tools (TPTP) platform. For more information about the Common Base Event component, see Chapter 5, "Working with events," on page 47.

Emitter
The emitter component supports the sending of events. After an event source creates an event and populates it with data, the event source submits the event to an emitter. The emitter optionally performs automatic content completion and then validates the event to ensure that it conforms to the Common Base Event specification. It also compares the event to configurable filter criteria. If the event is valid and passes the filter criteria, the emitter sends the event to the event server. An emitter can send events to the event server either synchronously (using Enterprise JavaBeans™ calls) or asynchronously (using a Java™ Messaging Service queue). For more information about emitters, see Chapter 6, "Developing an event source," on page 55.

Event server
The event server is the conduit between event sources and event consumers. The event server receives events submitted to emitters by event sources. It stores events in a persistent data store, and then distributes them asynchronously to subscribed event consumers. In addition, the event server supports synchronous queries of historical events from the persistent store. For more information about receiving events from the event server, see Chapter 7, "Developing an event consumer," on page 63.

Event catalog
The event catalog is a repository of event metadata. Applications use the event catalog to retrieve information about classes of events and their permitted content. For more information about the event catalog, see Chapter 8, "Developing an event catalog application," on page 83.

In addition, an application or solution using the Common Event Infrastructure might also include the following components (which are not part of the infrastructure itself):

Event source
An event source is any application that uses an emitter to send events to the event server.

Event consumer
An event consumer is any application that receives events from the event server.

Event catalog application
An event catalog application is any application that stores or retrieves event metadata in the event catalog. This might be a management or development tool; it might also be an event source or event consumer.

The Common Base Event model

The Common Base Event model is a standard defining a common representation of events that is intended for use by enterprise management and business applications. This standard, developed by the IBM® Autonomic Computing Architecture Board, supports encoding of logging, tracing, management, and business events using a common XML-based format, making it possible to correlate different types of events that originate from different applications. The Common Base Event model is part of the IBM Autonomic Computing Toolkit; for more information, see http://www.ibm.com/autonomic.

The Common Event Infrastructure currently supports version 1.0.1 of the specification.

The basic concept behind the Common Base Event model is the *situation*. A situation can be anything that happens anywhere in the computing infrastructure, such as a server shutdown, a disk-drive failure, or a failed user login. The Common Base Event model defines a set of standard situation types that accommodate most of the situations that might arise (for example, StartSituation and CreateSituation).

An *event* is a structured notification that reports information related to a situation. An event reports three kinds of information:

- The situation itself (what has happened)

- The identity of the affected component (for example, the server that has shut down)
- The identity of the component that is reporting the situation (which might be the same as the affected component)

The Common Base Event specification defines an event as an XML element containing properties that provide all three kinds of information. These properties are encoded as attributes and subelements of the root element, CommonBaseEvent.

The Common Base Event format is extensible. In addition to the standard event properties, an event can also contain extended data elements, which are application-specific elements that can contain any kind of information relevant to the situation. The *extensionName* attribute labels an event with an optional classification name (an event class), which indicates to applications what sort of extended data elements to expect. The event catalog stores event definitions that describe these event classes and their allowed content.

For complete details on the Common Base Event format, see the specification document and XSD schema included in the IBM Autonomic Computing Toolkit.

Chapter 2. Installation, migration, and uninstallation

The Common Event Infrastructure is designed to be embedded as part of an application. This chapter explains how to include the required Common Event Infrastructure components in your application's installation, migration, or uninstallation process.

Installing Common Event Infrastructure components

To include Common Event Infrastructure runtime components in your application installation package, you must configure your application installation process to use the silent installer.

Note: The information in this topic is intended for internal developers building applications that use the Common Event Infrastructure component. This section should not be exposed to external customers.

Because the Common Event Infrastructure is intended to be embedded by other products, it does not have its own installation interface. Instead, it includes a silent installer and a set of configuration scripts, which an embedding product can use to install and configure the Common Event Infrastructure components without exposing it to users.

The silent installer uses response files to specify installation parameters. These response files are typically built by the embedding product installation process, using input provided by the user as well as hard-coded values for the embedding product.

After installation, the user must complete several post-installation configuration tasks before using the Common Event Infrastructure. (The silent installer can optionally perform some of these configuration steps automatically, based on parameters specified in the response files.) For more information about post-installation configuration, see "Configuring the Common Event Infrastructure" on page 19.

Designing an installation process

You can use the Common Event Infrastructure installation APIs to design an installation process that detects previously installed components and takes the appropriate steps.

The com.ibm.events.install Java package includes a set of installation APIs your product installer can use to detect previously installed Common Event Infrastructure components. By using these APIs, you can design a product installation process that avoids installation errors. To use these utilities, use the com.ibm.events.install.InstallUtilFactory class to create an instance of the installation utilities class. (For more information, refer to the Javadoc API documentation for the com.ibm.events.install package.)

Your overall installation process should follow these steps:

1. Make sure the appropriate prerequisite software is installed.

2. Use the isCeiProductInstalled() method to determine whether a version of the Common Event Infrastructure component is already installed.
3. If no version is already installed, proceed with a new installation.
4. If the Common Event Infrastructure component is already installed, use the checkInstalledCeiVersion() method to compare the installed version to the version now being installed. Depending on the situation, you might also need to prompt the user to determine how to proceed. Keep in mind that all installed products use the same version of the Common Event Infrastructure, so upgrading a previous installation might have compatibility implications for other products.
   - If the versions are identical, you can register the installing product as a user of the installed version by using the registerCeiUser() method. (If you proceed with installation in this situation, the installer will register the product without reinstalling the component.)
   - If the installed version is older, use the checkCompatibilityForInstallation() method to determine whether the installed version can be upgraded. If so, prompt the user for confirmation and then proceed with an upgrade installation. To upgrade, use the getCeiHome() method to determine the installed location of the Common Event Infrastructure, and specify that location for the CEI_HOME parameter in the installation response file. If the installed version cannot be upgraded, proceed with migration.
   - If the installed version is older and the user does not want to upgrade, determine whether the installing product is compatible with the older version of the Common Event Infrastructure. If it is, you can use the previously installed version of the Common Event Infrastructure installer (located in $CEI_HOME/installer/cei_installer.jar) to register the installing product as a user of the existing component. To determine the location of the existing $CEI_HOME directory, use the getCeiHome() method.
   - If the installed version is newer, use the previously installed version of the Common Event Infrastructure installer (located in $CEI_HOME/installer/cei_installer.jar) to register the installing product as a user of the existing component. To determine the location of the existing $CEI_HOME directory, use the getCeiHome() method.

Common Event Infrastructure prerequisites

The target system must satisfy the following software prerequisites before your product installation can install the Common Event Infrastructure:

- WebSphere® Application Server Version 6.0 must be installed. (For z/OS systems, WebSphere Application Server Version 6.0.2 or later is required.) The server must be stopped during installation.

  Note: The Common Event Infrastructure is automatically installed as a part of WebSphere Process Server Version 6.0.
- On Windows® systems, long directory paths can cause installation to fail. To avoid this problem, minimize the depth of the directory tree containing the WebSphere profile, as well as the lengths of the directory, profile, and cell names. The combined length of the WebSphere profile directory (including trailing backslash), profile name, and cell name must not exceed 64 characters. For example, the following values are valid:

Name	Value	Length
Profile path	`c:\Program Files\IBM\WebSphere\Profiles\`	40
Profile name	`CEIPROFILE`	10

Name	Value	Length
Cell name	MyMachineCELL	13
Total		63

- If you want to use a separate Java Messaging Service (JMS) provider for asynchronous event transmission or subscription, the JMS provider must be installed. A separate JMS provider is required only if you do not want to use the WebSphere Application Server embedded messaging feature.
- If you want to use external database software instead of the Cloudscape™ database, one of the following databases must be installed, with the latest fix packs applied:
  - IBM DB2 Universal Database™ for Linux®, UNIX®, and Windows, version 8.1 or 8.2
  - IBM DB2 Universal Database for z/OS® version 7.1 or 8.1
  - Oracle9i Database Release 2 with SQL*Plus
  - Oracle Database 10g with SQL*Plus
  - IBM Informix® Dynamic Server Version 9.4, 9.5, or 10.0
  - Microsoft® SQL Server 2000 Enterprise Edition with Service Pack 3
  - Sybase Adaptive Server Enterprise 12.5 or Adaptive Server Enterprise 15 with the Distributed Transaction Management feature enabled
- An XA-compliant Java Database Connectivity (JDBC) driver supported by WebSphere Application Server must be installed.

DB2® and Oracle database servers can be installed either on the same host as the Common Event Infrastructure server or on a remote host. If the database is installed on a remote host, make sure the necessary components are also installed on the local host:

- For a Type 2 JDBC provider, the database client must be installed and configured on the local host.
- For a Type 4 JDBC provider, the JAR files for the JDBC driver must be installed on the local host.

If you are using a DB2 database on a remote host, the node to be used by the DB2 client must be catalogued and ready for use.

Network Deployment considerations

If you are installing the Common Event Infrastructure components on a federated node or a Deployment Manager node in a Network Deployment cell, there are several additional considerations to keep in mind when planning your installation process.

- The Deployment Manager must be running when you augment the profiles of any federated nodes. However, be sure to stop the Deployment Manager before you augment the profile of the Deployment Manager node itself.
- The profile of the Deployment Manager node must be augmented with the Common Event Infrastructure before the profiles of the federated nodes in the cell are augmented.
- If you want to federate another node with the Common Event Infrastructure after augmenting the Deployment Manager profile, first augment the profile of the new node (as a stand-alone node) and then add it to the cell using the **addNode** command. Adding a node without the Common Event Infrastructure and then augmenting it is not supported.

- After using the **addNode** command to add a server augmented with the Common Event Infrastructure, you must manually run the **event-variables.jacl** script to manually complete the configuration:
  1. Go to the *profile_path*/event/bin directory (*profile_path* is the path to the profile for the server you are adding to the cluster).
  2. Run the **event-variables.jacl** script:

     ```
     wsadmin -f event-variables.jacl -action enableServer
     ```
- Migration is not supported from a Version 5 cluster to a Version 6 cluster. If you have the Common Event Infrastructure installed in a Version 5 cluster and need to upgrade to Version 6, you must create a Version 6 cluster and then perform a new installation of the Common Event Infrastructure in the new cluster. You can then upgrade the existing event database (preserving existing event data) and use the upgraded database with the new installation.
- If you need to remove a federated node that is augmented with the Common Event Infrastructure, unaugment the node before removing it from the cell.
- If you need to remove the Common Event Infrastructure components from a cluster, some manual cleanup is required on the Deployment Manager node. To remove the Common Event Infrastructure from a cluster, follow these steps:
  1. Unaugment CEI from the profiles of all servers that are members of the cluster.
  2. On the Deployment Manager node, go to the $CEI_HOME/scripts directory and run the following command to remove the Common Event Infrastructure enterprise applications:

     ```
     wsadmin -profile ./event-profile.jacl -f event-application.jacl
       -action uninstall -cluster cluster
     ```

     Replace *cluster* with the name of the cluster where the enterprise application is installed.
  3. If you installed the default messaging configuration, run the following command to remove the messaging enterprise application:

     ```
     wsadmin -profile ./event-profile.jacl -f default-event-message.jacl
       -action uninstall -cluster cluster
     ```

     Replace *cluster* with the name of the cluster where the messaging enterprise application is installed.
  4. If necessary, use the WebSphere Administrative Console to remove the JDBC data source from the scope at which it is currently configured.
  5. Unaugment CEI from the profile of the Deployment Manager node.

Installing the core files

To install the Common Event Infrastructure core files, your application installation process must build an installation response file and then run the silent installer.

The core files must be installed on each WebSphere Application Server system where the Common Event Infrastructure will be used. Only one copy of the core files needs to be installed for each WebSphere Application Server installation; multiple runtime environments (defined by WebSphere profiles) share a single copy of the core files.

The silent installer installs the core files based on parameters specified in an installation response file. To install the Common Event Infrastructure components, your application installation process must build the response file and start the silent installer:

1. Gather the required information and build the response file.

You can use hard-coded values or information gathered during the installation process. For information about the response file parameters and syntax, refer to the cei_install.rsp sample response file, which is included in the ceiinstallsample.zip installation package file.

An installation response file must specify `ACTION=INSTALL`.

2. Stop the application server.
3. Navigate to the `installer` directory within the Common Event Infrastructure installation image.
4. Run the silent installer. The silent installer is implemented as the Java class com.ibm.events.install.impl.CeiInstaller, packaged in cei_installer.jar. To start the installer, use the **java** command:

```
java -Djava.util.logging.config.file=logging.properties
  -cp cei_installer.jar com.ibm.events.install.impl.CeiInstaller -r resp_file
```

The parameter *resp_file* is the name of your installation response file. The following example starts the installer with a response file called install.rsp:

```
java -Djava.util.logging.config.file=logging.properties
  -cp cei_installer.jar com.ibm.events.install.impl.CeiInstaller
  -r install.rsp
```

If the installer detects that the same version of the core files is already installed, it registers the newly installed application as a user of the existing files.

The installer creates two files containing messages resulting from the installation:

- events_install_msg.log (log messages)
- events_install_trc.log (detailed trace messages)

These files are written to the location specified by the logging.properties file (the default is the temp directory defined for the system). You can change this location by modifying the logging.properties file. If the `COPY_LOGFILES` parameter in the installation response file specifies `true` (the default value), the log files are also copied from the temp directory to the $WAS_HOME/logs directory. (If a log file already exists, the new messages are appended to the existing file.) However, if you change the location specified in the logging.properties file, you must also set `COPY_LOGFILES` to `false`.

The installer exits with a return code indicating the result of the operation:

Return code	Explanation
0	Installation completed successfully.
1	The installation failed because of errors. Check the log and trace files for messages that might indicate the cause of the failure.
5	Install has failed because the core files are already installed in a location different from the one specified in the response file. Only one copy of the core files can be installed for each installation of WebSphere Application Server.
6	Installation of the core files succeeded, but an error occurred during registration of the installing product as a user of the files. Check the log and trace files to determine the cause of the failure. After correcting the problem, use the InstallUtil.registerCeiUser() method to attempt registration again. For more information, refer to the Javadoc documentation for the com.ibm.events.install.InstallUtil interface.

Return code	Explanation
7	Installation failed because a more recent version of the Common Event Infrastructure is already installed. Determine the location of the existing installation and use the version of the installer in the $CEI_HOME directory to install the new application.

Augmenting WebSphere Application Server profiles

After installing the core files, your application installation process must augment the appropriate WebSphere Application Server profiles.

WebSphere Application Server Version 6 uses profiles to define the server runtime environments that use the core files for an application. The Common Event Infrastructure silent installer augments the WebSphere profiles based on parameters specified in a profile augmentation response file. You can augment a deployment manager profile or an application server profile to use the Common Event Infrastructure; custom profiles are not supported.

Note: The profile augmentation process used by the Common Event Infrastructure is different from the profile augmentation process used by WebSphere Application Server. A profile augmented using the Common Event Infrastructure augmentation utility cannot be unaugmented using the WebSphere Application Server process.

In a Network Deployment environment, the Deployment Manager profile must be augmented before the profiles of the federated nodes. If the Deployment Manager node is already augmented and the installation needs to add an additional node with the Common Event Infrastructure, the new node should be augmented first (as a stand-alone node) and then federated.

To augment a profile, the application installation must follow these steps:

1. Gather the required information and build the profile augmentation response file.

   You can use hard-coded values or information gathered during the installation process. For information about the profile augmentation response file parameters and syntax, refer to the cei_augment.rsp sample response file, which is included in the ceiinstallsample.zip installation package file.

   A profile augmentation response file must specify `ACTION=AUGMENT`.
2. If your profile augmentation response file specifies `DEPLOY_EJB=true`, `CONFIGURE_DB=true`, or `CONFIGURE_DEFAULT_EVENT_MESSAGING=true`, your product installation program must also gather any necessary WebSphere Application Server and database user ID and password information.

   Your product installation program must then build a string containing the necessary parameters as `KEY=VALUE` pairs separated by colons; this string can then be supplied to the silent installer through standard input. You can specify any of the following parameters in this input string:

   WAS_ADMIN
   WAS_PASSWORD
   The WebSphere Application Server user ID and password to use for deploying the enterprise application or configuring the event database. These values are required if WebSphere security is enabled and your installation response file specifies `DEPLOY_EJB=true` or `CONFIGURE_DB=true`.

   DB_USER

DB_PASSWORD
The database user ID and password to use for configuring the event database. These values are required if your installation response file specifies `CONFIGURE_DB=true` and you are using database software other than Cloudscape.

For a DB2 database, the user ID is typically the db2instance user (such as db2inst1) or the schema user that has authority to access the existing database and tables.

For other database types, the specified user ID and password are created during database configuration.

The user ID used during database configuration is also used at run time for authentication and access to the event database.

SYS_USER
SYS_PASSWORD
ORACLE_ROLE
The Oracle SYSTEM username, password, and role to use for configuring the event database. These values are required if your installation response file specifies `CONFIGURE_DB=true` and you are configuring an Oracle event database.

Note: For the ORACLE_ROLE parameter, specify `sysdba`.

SA_USER
SA_PASSWORD
The system administration (sa) user ID and password to use for creating databases and users for a Sybase or SQL Server event database. (Do not specify the password if no sa password is set.)

JMS_USER
JMS_PASSWORD
The user name and password to use for configuring the JMS queue and connection factory with the WebSphere embedded messaging feature. These values are required if your installation response file specifies `CONFIGURE_DEFAULT_EVENT_MESSAGING=true`.

3. Stop the application server. If the product installation is augmenting a Deployment Manager profile, stop the Deployment Manager. (If you are augmenting the profile for a federated node, the Deployment Manager must be running.)
4. Run the silent installer to augment the profile. The silent installer is implemented as the Java class com.ibm.events.install.impl.CeiInstaller, packaged in cei_installer.jar. To make sure you are using the correct version of the installer, use the copy of cei_installer.jar installed in the $CEI_HOME directory (%CEI_HOME% on Windows systems). To start the installer, use the **java** command:

```
java -Djava.util.logging.config.file=$CEI_HOME/installer/logging.properties
  -cp $CEI_HOME/installer/cei_installer.jar com.ibm.events.install.impl.CeiInstaller
  -r resp_file
```

The parameter *resp_file* is the name of your profile augmentation response file. If you need to supply WebSphere or database user IDs and passwords, pipe the formatted string to the installer.

If the installer detects that the profile is already augmented, it registers the newly installed application as a user of the existing event server.

The installer writes informational and error messages to two files:

- events_install_msg.log (log messages)
- events_install_trc.log (detailed trace messages)

These files are written to the location specified by the logging.properties file (the default is the temp directory defined for the system). You can change this location by modifying the logging.properties file. If the `COPY_LOGFILES` parameter in the profile augmentation response file specifies `true` (the default value), the log files are also copied from the temp directory to the logs subdirectory of the profile directory. (If a log file already exists, the new messages are appended to the existing file.) However, if you change the location specified in the logging.properties file, you must also set `COPY_LOGFILES` to `false`.

The installer exits with a return code indicating the result of the operation:

Return code	Explanation
0	Profile augmentation completed successfully.
10	Profile augmentation succeeded, but errors occurred during post-installation database configuration, enterprise application deployment, or messaging configuration. This can happen if the profile augmentation response file specifies `CONFIGURE_DB=true`, `DEPLOY_EJB=true`, or `CONFIGURE_DEFAULT_EVENT_MESSAGING=true`, but an error prevents successful completion of the configuration script. Check the log and trace files for messages indicating the cause of the problem. After correcting the problem, use the appropriate script to configure the event database, deploy the enterprise application, or configure messaging.
11	Profile augmentation succeeded, but an error occurred during registration of the installing product as a user of the augmented profile. Check the log and trace files to determine the cause of the failure. After correcting the problem, use the InstallUtil.registerCeiUserForProfile() method to attempt registration again. For more information, refer to the Javadoc documentation for the com.ibm.events.install.InstallUtil interface, or repeat the profile augmentation step.
12	Profile augmentation failed because of errors. Check the log and trace files to determine the cause of the failure.
13	Profile augmentation failed because the version of the installed core files does not match the version of the installer. To make sure the versions match, use the copy of cei_installer.jar installed in the $CEI_HOME directory.

Migrating Common Event Infrastructure components from a previous version

If your application supports migration from a previous version that used the Common Event Infrastructure Version 5.1, you can migrate the previous configuration to a new Version 6 installation.

Note: The information in this section is intended for internal developers building applications that use the Common Event Infrastructure component. This section should not be exposed to external customers.

Migration includes setting up the target Version 6 environment, migrating the Version 5 configuration to the new environment, and upgrading the migrated components to the new version.

To migrate the Common Event Infrastructure components from a previous version, your application migration process must perform the following steps:

1. If necessary, install any required software prerequisites on the target system. The prerequisites include WebSphere Application Server Version 6.0.
2. Follow the WebSphere Application Server process for backing up your Version 5.x environment to a backup directory (for example, WAS_BACKUP_DIR). For more information, refer to the WebSphere Application Server documentation.
3. Copy the $WAS_5X_HOME/events directory to the WebSphere Application Server backup directory tree. The events directory is not included in the WebSphere Application Server backup, so you must copy this directory manually in order to make it available for migration. To do this, copy the events directory from the WebSphere Application Server Version 5 home directory to the websphere_backup subdirectory of the location you specified for the backup. For example, if your backup directory is called WAS_BACKUP_DIR, you would copy the events directory to WAS_BACKUP_DIR/websphere_backup/events.
4. Follow the WebSphere Application Server process for creating a profile on the target system for migration of your Version 5 environment. For more information, refer to the WebSphere Application Server migration documentation.

   Note: On Windows systems, long directory paths can cause installation to fail. To avoid this problem, minimize the depth of the directory tree containing the WebSphere profile, as well as the lengths of the directory, profile, and cell names. The combined length of the WebSphere profile directory (including trailing backslash), profile name, and cell name must not exceed 64 characters. For example, the following values are valid:

Name	Value	Length
Profile path	`c:\Program Files\IBM\WebSphere\Profiles\`	40
Profile name	`CEIPROFILE`	10
Cell name	`MyMachineCELL`	13
Total		63

5. Install the Common Event Infrastructure components in the target environment using the standard installation process. For more information, see “Installing the core files” on page 8.
6. Augment the target WebSphere profile using the Common Event Infrastructure profile augmentation process.

   Note: Do not deploy the Common Event Infrastructure enterprise applications as part of the profile augmentation step.
7. Follow the WebSphere Application Server process for migrating the Version 5 configuration saved in the backup directory to the new environment. For more information, refer to the WebSphere Application Server migration documentation.
8. Gather the required information and build a migration response file.

   You can use hard-coded values or information gathered during the installation process. For information about the migration response file parameters and syntax, refer to the cei_migration.rsp sample response file, which is included in the ceiinstallsample.zip installation package file.

9. Navigate to the `installer` subdirectory within the Common Event Infrastructure installation image.
10. Run the Common Event Infrastructure migration utility. The migration utility is implemented as the Java class com.ibm.events.install.impl.CeiMigration,packaged in cei_installer.jar. To start the migration utility, use the **java** command:

    ```
    java -Djava.util.logging.config.file=logging.properties
      -cp cei_install.jar com.ibm.events.install.impl.CeiMigration -r resp_file
    ```

    The parameter *resp_file* is the name of your migration response file.

The migration utility writes informational and error messages to two files:

- events_install_msg.log (log messages)
- events_install_trc.log (detailed trace messages)

These files are written to the location specified by the logging.properties file (the default is the temp directory defined for the system). You can change this location by modifying the logging.properties file.

The migration utility exits with a return code indicating the result of the operation:

Return code	Explanation
0	Migration completed successfully.
1	Migration failed because the target profile is not augmented to use the Common Event Infrastructure core files.
2	Migration failed because of errors.

Uninstalling the Common Event Infrastructure components

To uninstall the Common Event Infrastructure components, your application uninstallation utility must unaugment the WebSphere profile and uninstall the core files.

Note: The information in this section is intended for internal developers building applications that use the Common Event Infrastructure component. This section should not be exposed to external customers.

Unaugmenting WebSphere Application Server profiles

Before uninstalling the core files, your application uninstallation process must unaugment any WebSphere Application Server profiles that have previously been augmented with the Common Event Infrastructure.

You must unaugment the profile for each runtime environment (stand-alone node, federated node, or Deployment Manager node) from which you want to uninstall the Common Event Infrastructure.

The profile unaugmentation utility can optionally perform additional configuration steps to automatically remove the enterprise applications and event database, based on parameters specified in the response file. If these resources are not removed during unaugmentation, they must be removed separately. Before unaugmentation, you can use the provided scripts to remove these resources; after unaugmentation, the scripts are not available, so you must use the WebSphere Application Server administrative interfaces to remove the resources.

Note: In a Network Deployment environment, the profiles of managed nodes must be unaugmented before the Deployment Manager profile.

Repeat these steps for each profile you want to unaugment:

1. Gather the required information and build the profile unaugmentation response file.

   For information about the profile unaugmentation response file parameters and syntax, refer to the cei_unaugment.rsp sample response file, which is included in the ceiinstallsample.zip installation package file.

   A profile unaugmentation response file must specify `ACTION=UNAUGMENT`.
2. Stop the application server. For a Deployment Manager profile, stop the Deployment Manager.

   Note: If you are unaugmenting the profile for a federated node, the Deployment Manager must be running.
3. Run the silent uninstaller to unaugment the profile. The uninstaller is implemented as the Java class com.ibm.events.install.impl.CeiUninstaller, packaged in cei_installer.jar. To make sure you are using the correct version of the uninstaller, use the copy of cei_installer.jar installed in the $CEI_HOME directory (%CEI_HOME% on Windows systems). To start the uninstaller, use the **java** command:

   ```
   java -Djava.util.logging.config.file=$CEI_HOME/installer/logging.properties
     -cp $CEI_HOME/installer/cei_installer.jar com.ibm.events.install.impl.CeiUninstaller
     -r resp_file
   ```

   The parameter *resp_file* is the name of your profile unaugmentation response file.

The uninstaller writes informational and error messages to two files:

- events_install_msg.log (log messages)
- events_install_trc.log (detailed trace messages)

These files are written to the location specified by the logging.properties file (the default is the temp directory defined for the system). You can change this location by modifying the logging.properties file. If the `COPY_LOGFILES` parameter in the uninstallation response file specifies `true` (the default value), the log files are also copied from the temp directory to the logs subdirectory of the profile directory. (If a log file already exists, the new messages are appended to the existing file.) However, if you change the location specified in the logging.properties file, you must also set `COPY_LOGFILES` to `false`.

The uninstaller exits with a return code indicating the result of the operation:

Return code	Explanation
0	Profile unaugmentation completed successfully.
2	Unaugmentation failed because of errors removing the event database or event application. Check the log and trace files for messages indicating the cause of the problem.
6	Profile unaugmentation failed because the installed version of the Common Event Infrastructure does not match the version of the uninstaller. To make sure the versions match, use the copy of cei_installer.jar installed in the $CEI_HOME directory.
7	Profile unaugmentation failed because of errors. Check the log and trace files to determine the cause of the failure.

Return code	Explanation
8	Profile unaugmentation succeeded, but an error occurred during removal of WebSphere environment variables or custom properties. These must be removed manually using the WebSphere Administrative Console: 1. Navigate to **Environment > WebSphere Variables > Node scope** and delete the CEI_HOME variable. 2. Navigate to the **Servers > Application Servers** page and select the application server where the Common Event Infrastructure was installed. 3. Navigate to **Java and Process Management > Process Definition > Java Virtual Machine > Custom Properties**. Select the ws.ext.dirs property and remove all references to CEI_HOME/lib and CEI_HOME/client.
9	Profile unaugmentation failed because the Common Event Infrastructure product file could not be updated to unregister the uninstalling product.
10	The profile was not unaugmented because another product is registered as a user of the augmented profile. The uninstalling product has been successfully unregistered. If the unaugmentation response file specified `UNINSTALL_DB=true`, the database was not removed because it is used by another application.

Uninstalling the core files

To remove the core files, your application uninstallation process must build an uninstallation response file and then run the silent uninstaller.

Before the core files are uninstalled, any WebSphere profiles that are using the files must first be unaugmented. If you uninstall the core files without first unaugmenting the profiles, the uninstaller attempts to unaugment the affected profiles; however, this is not possible under some circumstances and might fail.

The silent uninstaller removes the Common Event Infrastructure core files on your system, based on parameters specified in the uninstallation response file. To remove the core files, build the response file and start the silent uninstaller:

1. Gather the required information and build the uninstallation response file.

   For information about the uninstallation response file parameters and syntax, refer to the cei_uninstall.rsp sample response file, which is included in the ceiinstallsample.zip installation package file.

   An uninstallation response file must specify `ACTION=UNINSTALL`.
2. Run the silent uninstaller. The silent uninstaller is implemented as the Java class com.ibm.events.install.impl.CeiUninstaller, packaged in cei_installer.jar. To make sure you are using the correct version of the uninstaller, use the copy of cei_installer.jar installed in the $CEI_HOME directory (%CEI_HOME% on Windows systems). To start the uninstaller, use the **java** command:

   ```
   java -Djava.util.logging.config.file=$CEI_HOME/installer/logging.properties
     -cp $CEI_HOME/installer/cei_installer.jar com.ibm.events.install.impl.CeiUninstaller
     -r resp_file
   ```

   The parameter *resp_file* is the name of your uninstallation response file.

The uninstaller writes informational and error messages to two files:

- events_install_msg.log (log messages)
- events_install_trc.log (detailed trace messages)

These files are written to the location specified by the logging.properties file (the default is the temp directory defined for the system). You can change this location by modifying the logging.properties file. If the `COPY_LOGFILES` parameter in the

uninstallation response file specifies `true` (the default value), the log files are also copied from the temp directory to the $WAS_HOME/logs directory. (If a log file already exists, the new messages are appended to the existing file.) However, if you change the location specified in the logging.properties file, you must also set `COPY_LOGFILES` to `false`.

The uninstaller exits with a return code indicating the result of the operation:

Return code	Explanation
0	Uninstallation completed successfully.
1	Uninstallation failed because of errors. Check the log and trace files to determine the cause of the failure.
3	The installation process completed successfully, but the core files have not been removed because another product is registered as a user of the files. The uninstalling product has been unregistered.
4	The core files were removed, but an error occurred during removal of WebSphere environment variables or custom properties. These must be removed manually using the WebSphere Administrative Console: 1. Navigate to **Environment > WebSphere Variables > Node scope** and delete the CEI_HOME variable. 2. Navigate to the **Servers > Application Servers** page and select the application server where the Common Event Infrastructure was installed. 3. Navigate to **Java and Process Management > Process Definition > Java Virtual Machine > Custom Properties**. Select the ws.ext.dirs property and remove all references to CEI_HOME/lib and CEI_HOME/client.
5	Uninstallation failed because the Common Event Infrastructure product file could not be updated to unregister the uninstalling product.
6	Uninstallation failed because the installed version of the Common Event Infrastructure does not match the version of the uninstaller. To make sure the versions match, use the copy of cei_installer.jar installed in the $CEI_HOME directory.

Note: On Windows systems, the cei_installer.jar file is not removed by the uninstallation process. You must delete this file after the uninstallation process completes.

Chapter 3. Configuration

Configuring the Common Event Infrastructure

Before you can begin using the Common Event Infrastructure, you must complete several post-installation configuration tasks.

Configuring the event database

Database configuration includes creating the required tables and configuring JDBC data sources.

The event database is required to support persistence of events. You must configure a new event database under any of the following circumstances:

- You are setting up a new installation of the Common Event Infrastructure.
- You have migrated from a previous version with a Cloudscape event database. Migration is not supported for a Cloudscape event database.

Configuring the event database (Linux, UNIX, and Windows systems)

You can configure an event database on a Linux, UNIX, or Windows system using Cloudscape, DB2, or Oracle database software.

To configure the event database:

1. Go to the *profile_path*/event/dbconfig directory for the profile defining the WebSphere Application Server runtime environment where you want to configure the database. (Replace *profile_path* with the path to the directory containing the WebSphere Application Server profile.)
2. Using an ASCII text editor, open one of the provided sample database response files. Select the sample response file for the database software you are using:

Database	Sample response file
Cloudscape	CloudscapeResponseFile.txt
DB2 Universal Database for Linux, UNIX, and Windows	DB2ResponseFile.txt
Oracle Database	OracleResponseFile.txt
Informix Dynamic Server	InformixResponseFile.txt
SQL Server Enterprise	SQLServerResponseFile.txt
Sybase Adaptive Server Enterprise	SybaseResponseFile.txt

3. Modify the database response file with the correct information for your environment. Refer to the comments in the sample response file for more information about the parameters, including complete syntax information. Typically, you must modify the parameters described in the following table:

Table 1. Minimal parameters to modify in database response file

Database type	Parameters
IBM DB2 Universal Database (Linux, UNIX, or Windows systems)	**JDBC_CLASSPATH** The path to the DB2 JDBC driver. **UNIVERSAL_JDBC_CLASSPATH** The path to the Universal JDBC driver.
Oracle Database	**ORACLE_HOME** The path to the Oracle home directory. **JDBC_CLASSPATH** The JDBC driver class path.
IBM Informix Dynamic Server	**JDBC_CLASSPATH** The JDBC driver class path. **DB_HOST_NAME** The TCP/IP host name of the Informix database server machine. (This must be a valid host name, not `localhost`. **DB_SERVER_NAME** The Informix database server name. **DB_SYSTEM_DIR** The path to the directory where the event database data files are stored. **DB_INFORMIX_DIR** The path to the directory where the Informix software is installed.
Sybase Adaptive Server	**JDBC_CLASSPATH** The JDBC driver class path. **DB_SERVER_NAME** The database server name. **DB_SYSTEM_DIR** The path to the directory where the event database data files are stored.
Microsoft SQL Server	**JDBC_CLASSPATH** The fully qualified JDBC driver class path. **DB_SERVER_NAME** The TCP/IP host name of the database server. **DB_SYSTEM_DIR** The fully qualified path to the directory where the event database data files are stored. This directory must be on the same machine where the SQL Server database server is installed.

4. Run the database configuration script for your operating system, specifying the name of the database response file as a parameter. Use one of the following commands:

   Windows systems

   ```
   config_event_database.bat response_file
   ```

   Linux and UNIX systems

   ```
   config_event_database.sh response_file
   ```

For example, the following command configures a DB2 event database on a Windows system: `/config_event_database.bat DB2ResponseFile.txt`

The script configures the event database and creates two JDBC data sources: one for the event database and one for the event catalog. A message is displayed when database configuration is complete.

Note: If your database response file specifies `EXECUTE_SCRIPTS=false`, you must complete the database configuration by manually running the generated scripts. The default value in the sample database response file is `EXECUTE_SCRIPTS=true`.

After you configure the event database, you must restart the application server.

Manually running database configuration scripts:

If your database response file specifies `EXECUTE_SCRIPTS=false`, you must complete the database configuration process by manually running the generated scripts.

Database configuration is a two-step process. The **config_event_database** script first generates database-specific scripts for your environment; these generated scripts then create the database and data sources. If your database response file specifies `EXECUTE_SCRIPTS=true`, the **config_event_database** performs both steps automatically. However, if your database response file specifies `EXECUTE_SCRIPTS=false`, you must complete the database configuration by manually running the generated scripts.

The default value in the sample database response file specifies `EXECUTE_SCRIPTS=true`. Unless you changed this value in your customized response file, you do not need to run the generated scripts manually.

To manually run the generated scripts:

1. Go to the database-specific subdirectory containing the generated scripts for creating the event database.

   The scripts are placed in one of the following directories, depending upon the database software you are using:

   - *profile_path*/event/dbscripts/cloudscape
   - *profile_path*/event/dbscripts/db2
   - *profile_path*/event/dbscripts/oracle
   - *profile_path*/event/dbscripts/informix
   - *profile_path*/event/dbscripts/sybase
   - *profile_path*/event/dbscripts/sqlserver

   Replace *profile_path* with the path to the directory containing the profile for the WebSphere Application Server runtime environment in which you are configuring the event database.
2. Run the generated database creation script for your operating system and database software:

Type	Operating system	Database script
Cloudscape	Windows	cr_event_cloudscape.bat
Cloudscape	Linux/UNIX	cr_event_cloudscape.sh
DB2	Windows	cr_event_db2.bat

Type	Operating system	Database script
DB2	Linux/UNIX	cr_event_db2.sh
Oracle	Windows	cr_event_oracle.bat
Oracle	Linux/UNIX	cr_event_oracle.sh
Informix	Windows	cr_event_informix.bat
Informix	Linux/UNIX	cr_event_informix.sh
Sybase	Windows	cr_event_sybase.bat
Sybase	Linux/UNIX	cr_event_sybase.sh
SQL Server	Windows	cr_event_mssql.bat
SQL Server	Linux/UNIX	cr_event_mssql.sh

3. Go to the database-specific subdirectory containing the generated scripts for creating the JDBC data sources.

   The scripts are placed in one of the following directories, depending upon the database software you are using:

   - *profile_path*/event/dsscripts/db2
   - *profile_path*/event/dsscripts/db2zos
   - *profile_path*/event/dsscripts/cloudscape
   - *profile_path*/event/dsscripts/oracle
   - *profile_path*/event/dsscripts/informix
   - *profile_path*/event/dsscripts/sybase
   - *profile_path*/event/dsscripts/sqlserver

   Replace *profile_path* with the path to the directory containing the profile for the WebSphere Application Server runtime environment in which you are configuring the event database.
4. Run the generated JDBC configuration script for your operating system and database software:

Type	Operating system	JDBC configuration script
Cloudscape	Windows	cr_cloudscape_jdbc_provider.bat
Cloudscape	Linux/UNIX	cr_cloudscape_jdbc_provider.sh
DB2	Windows	cr_db2_jdbc_provider.sh
DB2	Linux/UNIX	cr_db2_jdbc_provider.sh
Oracle	Windows	cr_oracle_jdbc_provider.bat
Oracle	Linux/UNIX	cr_oracle_jdbc_provider.sh
Informix	Windows	cr_informix_jdbc_provider.bat
Informix	Linux/UNIX	cr_informix_jdbc_provider.sh
Sybase	Windows	cr_sybase_jdbc_provider.bat
Sybase	Linux/UNIX	cr_sybase_jdbc_provider.sh
SQL Server	Windows	cr_mssql_jdbc_provider.bat
SQL Server	Linux/UNIX	cr_mssql_jdbc_provider.sh

   Use the following command to run the JDBC configuration script:

   Windows systems

   ```
   cr_db_jdbc_provider scope [server_name]
   ```

Linux and UNIX systems

```
cr_db_jdbc_provider.sh scope [server_name]
```

The parameters are as follows:

scope The scope where you want to configure the JDBC provider. The valid values are `cell`, `node`, `server`, or `cluster`.

server_name
The name of the WebSphere server or cluster where you want to configure the JDBC provider, if **scope** is `server` or `cluster`. (If **scope** is `cell` or `node`, this parameter is ignored.

5. Restart the application server. For a federated node, you must also stop and restart the node agent using the **stopNode** and **startNode** commands.

After you finish configuring the database, you can use the WebSphere Administrative Console to test the database configuration. To do this, navigate to the appropriate JDBC data source and select the **Test Connection** option.

Configuring a DB2 database on a z/OS system

You can configure an event database on a z/OS system using DB2 database software.

To configure the DB2 database from a remote client, you must have the DB2 Connect product installed with the latest fix packs.

Note: Follow this procedure only if you are configuring a DB2 event database on a z/OS system.

To configure the event database:

1. On the z/OS system, use the DB2 administration menu to create a new subsystem.
2. Create a storage group. You will also need to specify the storage group name in the database response file; the default value is `sysdeflt`.
3. Grant the necessary permissions to the user ID you want the WebSphere Application Server data source to use. This user ID must have rights to access the database and storage group you created; it must also have permission to create new tables, table spaces, and indexes for the database.
4. Catalog the remote database. Run the following commands, either in a script or in a DB2 command-line window:

   ```
   catalog tcpip node zosnode remote hostname server IP_port system db_subsystem
   catalog database db_name as db_name at node zosnode authentication DCS
   ```

   For more information about how to catalog a nodes and databases, refer to the DB2 Connect™ documentation.
5. Verify that you can establish a connection to the remote subsystem. You can check this by running the following command:

   ```
   db2 connect to subsystem user userid using password
   ```

6. Bind to the host database. Run the following commands:

   ```
   db2 connect to db_name user userid using password
   db2 bind path/bnd/@ddcsmvs.lst blocking all sqlerror continue message
        mvs.msg grant public
   db2 connect reset
   ```

   For more information about binding a client to a host database, refer to the DB2 Connect documentation.

7. Go to the *profile_path*/event/dbconfig directory for the profile defining the WebSphere Application Server runtime environment where you want to configure the database. (Replace *profile_path* with the path to the directory containing the WebSphere Application Server profile.)
8. Using an ASCII text editor, open the DB2ZOSResponseFile.txt sample database response file.
9. Modify the database response file with the correct information for your environment. (See the comments in the sample response file for more information about the parameters, including complete syntax information.) Specify the following parameter values:

   DB_NAME
   The name of the z/OS database you created for the event database.

   JDBC_CLASSPATH
   The path to the DB2 JDBC driver.

   UNIVERSAL_JDBC_DRIVER_NATIVEPATH
   The path to the Universal JDBC native library path.

   UNIVERSAL_JDBC_CLASSPATH
   The path to the Universal JDBC driver.
10. Run the database configuration script for your client operating system, specifying the name of the database response file as a parameter. Run one of the following commands:

    Windows systems
    `config_event_database.bat` *response_file*

    Linux and UNIX systems
    `config_event_database.sh` *response_file*
11. Restart the application server. For a federated node, you must also stop and restart the node agent using the **stopNode** and **startNode** commands.

The script configures the event database and creates two JDBC data sources: one for the event database and one for the event catalog. A message is displayed when database configuration is complete.

Note: If your database response file specifies `EXECUTE_SCRIPTS=false`, you must complete the database configuration by manually running the generated scripts. The default value in the sample database response file is `EXECUTE_SCRIPTS=true`.

After you have finished configuring the database, you can use the WebSphere Administrative Console to test the database configuration. To do this, navigate to the appropriate JDBC data source and select the **Test Connection** option.

Manually running z/OS database configuration scripts:

If your database response file specifies `EXECUTE_SCRIPTS=false`, you must complete the database configuration process by manually running the generated scripts.

To run the scripts manually:

1. Go to the *profile_path*/event/dbscripts/db2zos directory.

   Replace profile_path with the path to the directory containing the profile for the WebSphere Application Server runtime environment in which you are configuring the event database.

2. Use the SQL Processor Using File Input (SPUFI) to load and run the generated DDL scripts. Run the scripts in the following order:
   - $WAS_HOME/event/dbscripts/db2zos/ddl/cr_db.db2
   - $WAS_HOME/event/dbscripts/db2zos/ddl/cr_db_catalog.db2
   - $WAS_HOME/event/dbscripts/db2zos/ddl/cr_tbl.db2
   - $WAS_HOME/event/dbscripts/db2zos/ddl/cr_tbl_catalog.db2
   - $WAS_HOME/event/dbscripts/db2zos/ddl/ins_metadata.db2
   - $WAS_HOME/event/dbscripts/db2zos/ddl/catalogSeed.db2
3. Go to the *profile_path*/event/dsscripts/db2zos directory.

   Replace profile_path with the path to the directory containing the profile for the WebSphere Application Server runtime environment in which you are configuring the event database.
4. Run the **$WAS_HOME/event/dsscripts/cr_db2zos_jdbc_provider** script to create the event data source. Specify the scope where the JDBC provider is to be configured:

   ```
   cr_db2zos_jdbc_provider scope [server_name]
   ```

After the event database is configured, you must restart the application server.

Oracle database limitations

Some limitations apply to configurations using Oracle database software. If you configured an Oracle event database, keep the following considerations in mind:

- The Oracle 10g JDBC thin driver imposes some size restrictions for string values if you are using a Unicode character set. This can result in an Oracle ORA-01461 error when events containing large values (such as a long message attribute) are stored in the event database. For more information about this restriction, refer to the Oracle 10g documentation.

  To avoid this problem, use the Oracle 10g OCI driver or the Oracle 9i thin driver.
- Oracle database software treats a blank string as a NULL value. If you specify a blank string as an event attribute value, that string is converted to a NULL when it is stored in an Oracle event database.

Deploying the Common Event Infrastructure application

The event server enterprise application must be deployed in each WebSphere runtime environment where the profile has been augmented to use the Common Event Infrastructure.

The event server enterprise application is packaged in the event-application.ear EAR file. The **event-application.jacl** script installs this application in the WebSphere application server.

1. Go to the *profile_path*/event/application directory for the profile defining the WebSphere Application Server runtime environment where you want to deploy the application. (Replace *profile_path* with the path to the directory containing the WebSphere Application Server profile.)
2. Run the **event-application.jacl** script using the **wsadmin** command:

   Windows systems

   ```
   profile_path\bin\wsadmin [-conntype none] -profile event-profile.jacl
     -f event-application.jacl  -action action -earfile event-application.ear
     -backendid backend_id -node node_name -server server_name
     [-cluster cluster_name] [-appname app_name] [-trace]
   ```

Linux and UNIX systems

```
profile_path/bin/wsadmin.sh [-conntype none] -profile event-profile.jacl
  -f event-application.jacl  -action action -earfile event-application.ear
  -backendid backend_id -node node_name -server server_name
  [-cluster cluster_name] [-appname app_name] [-trace]
```

The parameters are as follows:

action
The action to perform. To install the enterprise application, specify `install`. To update an existing event server application that is already installed, specify `update`. During an update, the script makes a backup copy of the existing application EAR file in the current directory; if necessary, you can later use this backup copy to restore the previous version of the application.

backend_id
The type of database back end to be used by the enterprise application. This must be one of the following values:

- CLOUDSCAPE_V51_1
- DB2UDBNT_V82_1
- DB2UDBNT_V8_1
- ORACLE_V10_1
- ORACLE_V9_1
- DB2UDBOS390_V8_1
- DB2UDBOS390_V7_1
- INFORMIX_V94_1
- SYBASE_V125_1
- MSSQLSERVER_V7_1

node_name
The WebSphere Application Server node in which the event server is to be deployed. To find out the node name, follow these steps:

a. Run the ***profile_path*/bin/setupCmdLine** script.
b. Run the command `echo $WAS_NODE` (Linux/UNIX systems) or `echo %WAS_NODE%` (Windows systems).

This value is case-sensitive. If you are deploying the enterprise application in a cluster, omit this parameter.

server_name
The WebSphere server into which the event server enterprise application is to be deployed. This value is case-sensitive. If you are deploying the enterprise application in a cluster, omit this parameter.

cluster_name
The WebSphere cluster into which the event server enterprise application is to be deployed. Specify this parameter only if you are deploying the application in a cluster and you are not specifying a node and server.

app_name
The name to use for the Common Event Infrastructure enterprise application. This parameter is optional; the default value is `EventServer`.

The optional **-trace** parameter causes additional debugging information to be displayed on the standard output.

Note:

- If you specify a fully qualified path for the location of the event-application.ear file, make sure you use forward slashes (/) in the path, even on Windows systems.
- If you are deploying the application on a stand-alone node, specify the optional `-conntype none` parameter to run **wsadmin** in local mode.

For more information about the **wsadmin** utility, refer to the WebSphere Application Server documentation.

After the **event-application.jacl** script completes, the Common Event Infrastructure enterprise application is deployed in the specified server or cluster. In a WebSphere

Application Server Network Deployment environment, if the application is already installed, the script only adds the deployment information for the specified node and server.

Configuring event messaging

If you want to use Java Messaging Service (JMS) queues for asynchronous message transmission to the event server, you must configure event messaging.

Configuring default event messaging

The default messaging configuration for asynchronous event transport uses the WebSphere Application Server embedded messaging feature as the Java Messaging Service (JMS) provider.

The **default-event-message.jacl** script provides a way to quickly set up a default messaging configuration, using the WebSphere embedded messaging feature as the JMS provider. This script sets up all of the configuration objects required for asynchronous event transmission:

- It creates a JMS queue and a queue connection factory using the embedded messaging feature.
- It creates a service integration bus and adds members to the bus, associating the bus with queues, topics, and connection factories.
- It creates a JMS transmission profile using the created queue and connection factory.
- It configures the default emitter factory profile to use the created JMS transmission profile for asynchronous event transmission.
- It deploys the message-driven bean used by the Common Event Infrastructure to receive events sent asynchronously to the event server.

To configure default messaging:

1. Go to the *profile_path*/event/application directory for the profile defining the WebSphere Application Server runtime environment where you want to configure default messaging. (Replace *profile_path* with the path to the directory containing the WebSphere Application Server profile.)
2. Run the **default-event-message.jacl** script using the **wsadmin** command:

   Windows systems

   ```
   profile_path\bin\wsadmin [conntype -none] -profile event-profile.jacl
     -f default-event-message.jacl -action action -earfile event-message.ear
     -node node_name -server server_name
     [-cluster cluster_name] [-appname app_name] [-trace]
   ```

   Linux and UNIX systems

   ```
   profile_path/bin/wsadmin.sh [conntype -none] -profile event-profile.jacl
     -f default-event-message.jacl -action install -earfile event-message.ear
     -node node_name -server server_name
     [-cluster cluster_name] [-appname app_name] [-trace]
   ```

The parameters are as follows:

node_name
The WebSphere Application Server node in which the messaging application is to be deployed. To find out the node name:

a. Run the ***profile_path*/bin/setupCmdLine** script.
b. Run the command `echo $WAS_NODE` (Linux/UNIX systems) or `echo %WAS_NODE%` (Windows systems).

This value is case-sensitive. If you are deploying the application in a cluster, omit this parameter.

server_name
The WebSphere server into which the messaging application is to be deployed. This value is case-sensitive. If you are deploying the application in a cluster, omit this parameter.

cluster_name
The WebSphere cluster into which you want to deploy the messaging application. Specify this parameter only if you are configuring messaging in a cluster and you are not specifying a node and server.

Note: The default messaging configuration uses a Cloudscape event database. The Cloudscape database does not support clustering in a production environment. You can use the WebSphere Administrative Console to specify a different messaging engine, specifying a data source that connects to an event database that supports multiple connections.

app_name
The name to use for the messaging enterprise application. This parameter is optional; the default value is `EventServerMdb`.

The optional **-trace** parameter causes additional debugging information to be displayed on the standard output.

Note:

- If you specify a fully qualified path for the location of the event-message.ear file, make sure you use forward slashes (/) in the path, even on Windows systems.
- If you are deploying the messaging application on a stand-alone node, specify the optional `-conntype none` parameter to run **wsadmin** in local mode

After you start the script, you are prompted for your JMS user ID and password.

Configuring event messaging using another JMS provider

If you do not want to use the WebSphere Application Server embedded messaging feature for event transmission, you can configure asynchronous message transport to use a different Java Messaging Service (JMS) provider.

Before you can configure event messaging using an external JMS provider, you must first create a JMS queue and connection factory using the appropriate interfaces for your JMS provider.

The **event-message.jacl** script sets up the configuration objects required for asynchronous event transmission using an external JMS provider such as WebSphere MQ:

- It creates a JMS transmission profile using the JMS queue and connection factory you specify.
- It creates an emitter factory profile using the created JMS transmission profile for asynchronous event transmission.
- It deploys the message-driven bean used by the Common Event Infrastructure to receive events sent asynchronously to the event server, using either a listener port or a JMS activation specification.

If you want to set up more than one JMS queue to the event server, you can run this script multiple times, specifying different enterprise application names and JMS queues. Each time you run the script, it deploys an additional message-driven bean and configures new resources to use the specified JMS queue.

To configure event messaging to use an external JMS provider:

1. Go to the $WAS_HOME/event/application directory for the profile defining the WebSphere Application Server runtime environment where you want to configure default messaging. (Replace *profile_path* with the path to the directory containing the WebSphere Application Server profile.)
2. Run the **event-message.jacl** script using the **wsadmin** command:

   Windows systems

   ```
   %WAS_HOME%\bin\wsadmin -profile event-profile.jacl -f event-message.jacl
     -action install -earfile event-message.ear -node node_name
     [-server server_name] [cluster cluster_name]
     -appname app_name -qjndi queue -qcfjndi connection_factory
     [-listenerport listener_port] [-activationspecjndi spec_name]
     [-eventprofilescope scope] [-trace]
   ```

   Linux and UNIX systems

   ```
   $WAS_HOME/bin/wsadmin -profile event-profile.jacl -f event-message.jacl
     -action install -earfile event-message.ear -node node_name
     [-server server_name] [cluster cluster_name]
     -appname app_name -qjndi queue -qcfjndi connection_factory
     [-listenerport listener_port] [-activationspecjndi spec_name]
     [-eventprofilescope scope] [-trace]
   ```

   The parameters of the **event-message.jacl** script are as follows:

   node_name
   The WebSphere Application Server node in which the messaging application is to be deployed. To find out the node name:

   a. Run the ***profile_path*/bin/setupCmdLine** script.
   b. Run the command `echo $WAS_NODE` (Linux/UNIX systems) or `echo %WAS_NODE%` (Windows systems).

   This value is case-sensitive. If you are deploying the application in a cluster, omit this parameter.

   server_name
   The WebSphere server into which the messaging application is to be deployed. This value is case-sensitive. If you are deploying the application in a cluster, omit this parameter.

   cluster_name
   The WebSphere cluster into which you want to deploy the messaging application. Specify this parameter only if you are configuring messaging in a cluster and you are not specifying a node and server.

Note: If you are using a Cloudscape event database, clustering is supported only for testing purposes (not in a production environment).

app_name
The name to use for the messaging enterprise application. This parameter is required.

queue
The JNDI name of the JMS queue to be used by the messaging enterprise application. This queue is used for asynchronous message transport to the event server. If you deploy the messaging application multiple times, you must specify a different JMS queue each time.

connection_factory
The JNDI name of the JMS connection factory to be used by the messaging enterprise application.

listener_port
The name of the listener port used by the messaging enterprise application. The listener port is specified in the deployment descriptor. Specify this parameter only if your messaging application uses a listener port.

spec_name
The JNDI name of the activation spec used by the messaging enterprise application. The activation spec is specified in the deployment descriptor. Specify this parameter only if your messaging application uses an application spec.

scope
The scope of the configuration profile objects to be created for event messaging. This parameter is optional; if you specify a scope, a JMS transmission profile and emitter factory profile are created at the specified scope. The valid values are `cell`, `node`, and `server`.

The optional **-trace** parameter causes additional debugging information to be displayed on the standard output.

Note:

- If you specify a fully qualified path for the location of the event-message.ear file, make sure you use forward slashes (/) in the path, even on Windows systems.
- If you are deploying the messaging application on a stand-alone node, specify the optional `-conntype none` parameter to run **wsadmin** in local mode

After you start the script, you are prompted for your JMS user ID and password.

Upgrading the event database

If you have migrated from a previous version of the Common Event Infrastructure and you are using event persistence, you must upgrade the event database after migration.

The database upgrade script upgrades the schema and metadata of the existing event database to the current version.

Note: Upgrading the event database is not supported for Cloudscape databases. If you have migrated from a previous version that used a Cloudscape event

database, you must configure a new event database. You can then manually export the event catalog data from the previous version and import it into the new database. Event data cannot be imported from a previous version.

To upgrade the event database:

1. **Optional:** If your event database uses a version of database software that is no longer supported by Common Event Infrastructure 6.0, you must first migrate the database to a supported version using the appropriate procedure for the database software.
2. Locate the database response file used to configure the existing Version 5 database. This response file is typically located in the $WAS_HOME/event/dbconfig directory of the WebSphere Application Server Version 5 installation.
3. Copy the Version 5 database response file to the *profile_path*/event/dbconfig directory. (Replace *profile_path* with the directory for the profile defining the target WebSphere Application Server Version 6 profile.)
4. Go to the *profile_path*/event/dbconfig directory and run the database upgrade script for your operating system, specifying the name of the database response file (optionally including the path) that was used to configure the previous (Version 5.x) event database . Use one of the following commands:

   Windows systems

   ```
   upgrade_event_database.bat v5_response_file
   ```

   Linux and UNIX systems

   ```
   upgrade_event_database.sh v5_response_file
   ```

5. **Optional:** If you are upgrading from a database software version that is no longer supported, you must also update the environment variables and configuration to use the new database back end.
   a. In the WebSphere Administrative Console, navigate to **Environment > WebSphere Variables**.
   b. Select node scope.
   c. Modify all outdated database path information to the new values.

      If you are upgrading from a DB2 Version 7 database, modify the following environment variables:

      - DB2UNIVERSAL_JDBC_DRIVER_PATH
      - UNIVERSAL_JDBC_DRIVER_PATH

      If you are upgrading from an Oracle8i database, modify the ORACLE_JDBC_DRIVER_PATH environment variable.

      Save your changes and close the console.

   d. Use the wsadmin utility to update the database back end used by the Common Event Infrastructure enterprise application (replace *profile_path* with the path to the directory containing the profile for the WebSphere runtime environment):

      ```
      profile_path/bin/wsadmin -profile "profile_path/event/application/event-profile.jacl"
        -f "profile_path/event/application/update-event-application.jacl"
        -earpath "$ProfileHome/event/application" -node node_name
        -backendid backend_id
      ```

      The parameters are as follows:

      node_name
      The name of the WebSphere Application Server node where the Common Event Infrastructure application is deployed.

backend_id
The new database back end to be used by the enterprise application. This must be one of the following values:

- CLOUDSCAPE_V51_1
- DB2UDBNT_V82_1
- DB2UDBNT_V8_1
- ORACLE_V10_1
- ORACLE_V9_1
- DB2UDBOS390_V8_1
- DB2UDBOS390_V7_1
- INFORMIX_V94_1
- SYBASE_V125_1
- MSSQLSERVER_V7_1

If the database response file specifies `EXECUTE_SCRIPTS=NO`, you must manually complete the database migration process by running the generated scripts in the *profile_path*/event/dbupgrade/*dbtype* directory. If the database response file specifies `EXECUTE_SCRIPTS=YES`, this was done automatically. The default value in the sample database response file is `EXECUTE_SCRIPTS=YES`.

Manually running the generated database upgrade script

If your database response file specifies EXECUTE_SCRIPTS=NO, you must manually complete the database upgrade process by running the generated database upgrade script.

If the database response file specifies EXECUTE_SCRIPTS=NO, the **upgrade_event_database** script generates a customized script for upgrading the event database but does not run that script. To complete the database upgrade process, you must manually run the generated script.

This step is not necessary if the database response file specifies EXECUTE_SCRIPTS=YES. The default value in the sample database response file is EXECUTE_SCRIPTS=YES.

1. Go to the *profile_path*/event/dbupgrade/*dbtype* directory:
   - *profile_path* is the name of the directory containing the target WebSphere Application Server Version 6 profile.
   - *dbtype* is the type of database you are upgrading (`db2`, `db2zos`, or `oracle`).
2. Run the generated script to complete the database upgrade. Use one of the following commands:

   Windows systems
   `upgrade_db.bat`

   Linux and UNIX systems
   upgrade_db.sh

Removing the event database

If you need to remove the event database manually, you can use the provided scripts.

When the database is configured, the configuration script also creates scripts for removing the database and the JDBC provider. The scripts for removing the event database are placed in database-specific subdirectories of $WAS_HOME/event/

dbscripts. The scripts for removing the JDBC provider are placed in database-specific subdirectories of $WAS_HOME/event/dsscripts.

Note: The event database can be shared among multiple event servers using the same JDBC provider configuration. Therefore, the JDBC provider configuration should be removed only if the associated event database has been uninstalled.

To remove the event database and JDBC provider, run the appropriate script listed in following table.

Type	Operating system	Database script	JDBC configuration script
Cloudscape	Windows	rm_event_cloudscape.bat	rm_cloudscape_jdbc_provider.bat
Cloudscape	Linux/UNIX	rm_event_cloudscape.sh	rm_cloudscape_jdbc_provider.sh
DB2	Windows	rm_event_db2.bat	rm_db2_jdbc_provider.bat
DB2	Linux/UNIX	rm_event_db2.sh	rm_db2_jdbc_provider.sh
DB2	z/OS (Windows script)	rm_event_db2zos.bat	rm_db2zos_jdbc_provider.bat
DB2	z/OS (Linux/UNIX script)	rm_event_db2zos.sh	rm_db2zos_jdbc_provider.sh
Oracle	Windows	rm_event_oracle.bat	rm_oracle_jdbc_provider.bat
Oracle	Linux/UNIX	rm_event_oracle.sh	rm_oracle_jdbc_provider.sh
Informix	Windows	rm_event_informix.bat	
Informix	Linux/UNIX	rm_event_informix.sh	
Sybase	Windows	rm_event_sybase.bat	
Sybase	Linux/UNIX	rm_event_sybase.sh	
SQL Server	Windows	rm_event_mssql.bat	
SQL Server	Linux/UNIX	rm_event_mssql.sh	

You can remove the event database or JDBC provider at any time by running the appropriate script. To remove the JDBC provider, use the appropriate script and specify the scope from which you want to remove the JDBC provider:

`rm_db_jdbc_provider scope [server_name]`

The generated scripts use these parameters:

scope The scope from which you want to remove the JDBC provider. The valid values are `cell`, `node`, `server`, and `cluster`.

server_name
The name of the WebSphere server or cluster from which you want to remove the JDBC provider, if **scope** is `server` or `cluster`. (If **scope** is `cell` or `node`, this parameter is ignored.

Removing the Common Event Infrastructure application

If you need to manually remove the event server enterprise application and resources from WebSphere Application Server, you can use the **event-application.jacl** script.

The Common Event Infrastructure enterprise application can be removed automatically when you unaugment a WebSphere profile. It is only necessary to remove the enterprise application manually if the profile unaugmentation response file specifies `UNINSTALL_EJB=false`. If you are removing the enterprise application manually, you must do so before unaugmenting the profile, because profile unaugmentation removes the enterprise application configuration script.

Note: If you prefer, you can remove the event server enterprise applications manually using the WebSphere administrative console rather than using the **event-application.jacl** script. If you do this, you must also manually remove all other Common Event Infrastructure resources.

1. To remove the event server enterprise application, use the wsadmin tool to run the **event-application.jacl** script.

   To run the script on a Windows system, go to the *profile_path*\event\application directory and run the following command (all on one line):

   ```
   wsadmin [-conntype none] -profile event-profile.jacl
     -f event-application.jacl -action uninstall
     -node node_name [-server server_name]
     [-cluster cluster_name] [-trace]
   ```

   To run the script on a Linux or UNIX system, go to the *profile_path*/event/application directory and run the following command (all on one line):

   ```
   wsadmin.sh [-conntype none] -profile event-profile.jacl
     -f event-application.jacl -action uninstall
     -node node_name -server server_name
     -[cluster cluster_name] [-trace]
   ```

   The **event-application.jacl** script uses these parameters:

   node_name
   The WebSphere Application Server node from which you want to remove the event server enterprise application.

   server_name
   The WebSphere server from which you want to remove the event server enterprise application. If you are removing the application from a cluster, omit this parameter. If you do not specify a server or cluster, the enterprise application is removed from all servers in the node.

   cluster_name
   The WebSphere cluster from which you want to remove the event server enterprise application. Specify this parameter only if you are removing the application from a cluster.

   The optional **-trace** parameter causes additional debugging information to be displayed on the standard output.

2. **Optional:** If you are completely removing the Common Event Infrastructure from a cluster, the last step is to remove the enterprise application from the Deployment Manager node. Go to the *profile_path*/bin directory and run the following command on the Deployment Manager node to remove the Common Event Infrastructure enterprise application:

   ```
   wsadmin -profile ./event-profile.jacl -f event-application.jacl
     -action uninstall -cluster cluster
   ```

   Replace *cluster* with the name of the cluster where the enterprise application is installed.

If you are completely removing the Common Event Infrastructure from a cluster, you must remove the enterprise application from the Deployment Manager node after removing it from the other nodes in the cluster, but before completely

uninstalling the Common Event Infrastructure files. To remove the enterprise application from the Deployment Manager node, use the copy of the **event-application.jacl** script installed in the *profile_path*/bin directory.

Removing the event messaging enterprise application

Before uninstalling the Common Event Infrastructure, you must remove the event messaging enterprise application.

To remove the event messaging enterprise application, use the wsadmin tool to run the **event-message.jacl** script.

To run the script on a Windows system, go to the *profile_path*\event\application directory and run the following command (all on one line):

```
wsadmin [-conntype none] -profile event-profile.jacl
  -f event-message.jacl -action uninstall
  -node node_name [-server server_name]
  [-cluster cluster_name] -appname app_name
  [-trace]
```

To run the script on a Linux or UNIX system, go to the *profile_path*/event/application directory and run the following command (all on one line):

```
wsadmin.sh [-conntype none] -profile event-profile.jacl
  -f event-message.jacl -action uninstall
  -node node_name [-server server_name]
  [-cluster cluster_name] -appname app_name
  [-trace]
```

The parameters of the **event-message.jacl** script are as follows:

node_name
: The WebSphere Application Server node from which you want to remove the event messaging enterprise application. To find out the node name, follow these steps:

  1. Run the **$WAS_HOME/bin/setupCmdLine** script.
  2. Run the command `echo $WAS_NODE`.

server_name
: The WebSphere server from which you want to remove the event messaging enterprise application. If you are removing the application from a cluster, omit this parameter. If you do not specify a server or cluster, the application is removed from all servers in the specified node.

cluster_name
: The WebSphere cluster from which you want to remove the event messaging enterprise application. Specify this parameter only if you are removing the application from a cluster.

app_name
: The name of the deployed messaging enterprise application you want to remove. This parameter is required.

The optional **-trace** parameter causes additional debugging information to be displayed on the standard output.

If you are completely removing the Common Event Infrastructure from a cluster, you must remove the messaging enterprise application from the Deployment Manager node after removing it from the other nodes in the cluster, but before completely uninstalling the Common Event Infrastructure files. To remove the

messaging enterprise application from the Deployment Manager node, use the copy of the **event-message.jacl** script installed in the *profile_path*/bin directory.

Runtime configuration

You can configure the Common Event Infrastructure resources using the WebSphere Application Server administrative console, or from the command line with the wsadmin tool.

To see the Common Event Infrastructure resources in the administrative console, expand the **Resources** list and click **Common Event Infrastructure Provider**. The types of resources are displayed in the **Additional Properties** list. From here you can navigate to the individual resources of each type (for example, you can view a list of all event group profiles or emitter factory profiles). To change the configuration of a resource, click the resource name in the list and then edit the properties you want to change.

Refer to the WebSphere Application Server documentation for more information about the administrative console and the wsadmin tool.

In most circumstances, only certain properties need to be configured. For complete information about these resources and their properties, refer to the online help for the Common Event Infrastructure resources in the WebSphere administrative console.

Note: After changing the Common Event Infrastructure configuration, you must restart the WebSphere server.

Default configuration

The Common Event Infrastructure components are installed as a set of WebSphere Application Server applications, services, and default resources.

You can customize the Common Event Infrastructure by configuring the provided resources or creating additional resources; for more information, see "Runtime configuration."

The default configuration consists of the following objects:

Common Event Infrastructure service
A service installed into the WebSphere server. This service enables WebSphere applications and clients to use the Common Event Infrastructure.

Common Event Infrastructure enterprise application
The enterprise application for the event server. The deployment descriptor of the enterprise application associates the event server with the Common Event Infrastructure resources it uses.

Common Event Infrastructure messaging application
The enterprise application for the message-driven bean that supports asynchronous event transmission to the event server. This application is available only if you have configured event messaging; for more information, see "Configuring event messaging" on page 27.

Common Event Infrastructure Provider
A collection object containing the resources used by Common Event Infrastructure components, event sources, and event consumers.

Data store profile

A data store profile defines properties used by the default data store plug-in, which is used to persistently store events received by the event server. A default data store profile is provided; usually, no configuration is necessary for this resource, but in some circumstances you might want to adjust some properties for your environment. You might also need to create additional data store profiles if you want to set up multiple event servers in the same cell.

Event bus transmission profile

An event bus transmission profile defines properties used by emitters to access the event server synchronously using EJB calls; these profiles are used by emitter factory profiles. A default transmission profile is provided; usually, no configuration is necessary for this resource.

Event group profile list

An event group profile list is a collection containing the event group profiles used by the event server. The event group profile list used by an event server is specified in the deployment descriptor of the event server enterprise application. Usually, no configuration is necessary for this resource, but you might need to create additional event group profile lists if you want to set up multiple event servers in the same cell.

Event group profile

An event group profile defines an event group (a logical collection of events). Event groups are used to categorize events according to their content; when querying events from the event server or subscribing to event distribution, an event consumer can specify an event group to retrieve only the events in that group.

A default event group profile is provided; this profile defines an event group containing all events, and is associated with the JMS topic jms/cei/notification/AllEventsTopic. You can create additional event group profiles specifying whatever event criteria are appropriate for your application.

Emitter factory profile

An emitter factory profile defines properties used by emitters. The properties in an emitter factory profile affect the behavior of any emitter that is created using the associated emitter factory. The default emitter factory profile specifies synchronous transmission, no filtering, and sending each event as part of the current transaction. You might want to create an additional emitter factory profile to specify a different transaction mode or transmission profile.

Event server profile

A profile defining properties used by the event server. The default event server profile enables event distribution and persistence, and it is configured to use the default data store plug-in. Usually, no configuration is necessary for this resource, but you might need to create additional event server profiles if you want to set up multiple event servers in the same cell.

JMS Transmission Profile

A JMS transmission profile defines properties used by emitters to access the event server asynchronously using a JMS queue; it is referenced by

emitter factory profiles. This profile is available only if you have configured event messaging; for more information, see "Configuring event messaging" on page 27.

Creating an emitter factory profile

An emitter factory profile defines properties used for an emitter factory, which event sources use to create emitters. The properties in an emitter factory profile affect the behavior of any emitter that is created using the associated emitter factory. You can use the default emitter factory profile or create additional profiles for your event sources to use. You might want to create an additional emitter factory profile to specify a different transaction mode or synchronous transmission profile. For more information about how these options affect the behavior of the emitter, see Chapter 6, "Developing an event source," on page 55.

To create an emitter factory profile, follow these steps:

1. In the WebSphere administrative console, click **Common Event Infrastructure Provider > Emitter Factory Profile > New**.
2. Specify the properties of the new profile. Refer to the online help for the Emitter Factory Profile Settings page for detailed information about these properties.
3. Click **OK** to save your changes and create the emitter factory profile.

Event sources can now use the configured emitter factory to obtain emitters.

Creating an event group

An event group defines a logical collection of events based on the content of their property data. An event group can be used when querying events from the event server, and it can optionally be associated with a JMS destination for asynchronous event distribution.

To create an event group, follow these steps:

1. **Optional:** Set up one or more JMS destinations for the event group. An event group can be associated with one JMS topic, and one or more JMS queues. Refer to the documentation for your JMS provider for information on how to create JMS destinations and connection factories and bind them into a JNDI namespace.

   Note: If WebSphere security is enabled, the configuration for the JMS destination must specify an authentication alias.
2. Create a new event group profile. In the WebSphere administrative console, click **Common Event Infrastructure Provider > Event Group Profile List > Event groups list > Event Group Profile > New**.
3. Specify the properties of the new event group, including the event selector and optional JMS destinations. For more information about event selectors, see "Writing event selectors" on page 78.
4. Click **OK** to save your changes and create the event group profile.

Event consumers can now specify the event group when querying events. If event distribution is enabled in the event server profile, events belonging to the event group are also published to any JMS destinations specified in the event group profile. Event consumers can then receive events asynchronously by subscribing to the appropriate destinations.

Chapter 4. Administration

You can perform several administrative tasks to control the operation of the Common Event Infrastructure components at run time.

Enabling and disabling the event server

You can enable and disable the event server using the WebSphere administrative console.

To enable or disable the event server, modify the properties of the Common Event Infrastructure service:

1. In the WebSphere administrative console, click **Servers > Application Servers**.
2. Click the name of the WebSphere server where the Common Event Infrastructure is installed.
3. Click **Container Services**.
4. Click **Common Event Infrastructure Service**.
5. Select or deselect the **Startup** property. If the check box is selected, the Common Event Infrastructure service starts when the WebSphere server starts.
6. Click **OK** to save your changes.
7. Restart the server.

Note: If you disable the Common Event Infrastructure service from the administrative console, the enterprise application is also disabled automatically. However, if you use the wsadmin tool to disable the service from a script, you must also disable the enterprise application separately.

Logging and tracing

You can enable logging and tracing in order to debug problems with an application using the Common Event Infrastructure.

The Common Event Infrastructure components use the JSR47 Java logging framework, which is available in WebSphere Application Server server and client environments. For more information about using the logging framework, refer to the WebSphere Application Server troubleshooting documentation.

The following table indicates the logger names used by the Common Event Infrastructure components.

Table 2. Logger names

Component	Logger name
Root logger name	com.ibm.events
Event catalog	com.ibm.events.catalog
Event server subcomponents	com.ibm.events.access com.ibm.events.bus com.ibm.events.distribution com.ibm.events.server

Table 2. Logger names (continued)

Component	Logger name
Default data store plug-in	com.ibm.events.datastore
Event emitter	com.ibm.events.emitter
Notification helper	com.ibm.events.notification
Configuration	com.ibm.events.configuration
Installation	com.ibm.events.install
Miscellaneous utilities	com.ibm.events.util

Database maintenance

If you are using a DB2 event database on a Linux, UNIX, or Windows system, you should periodically perform database maintenance by running the provided scripts.

Updating database statistics

To enable the DB2 database to optimize queries and find free space, update the database statistics using the **runstats** script.

Updating the database statistics is recommended on a regular basis, and especially under any of these circumstances:

- Events have been deleted from the database, using either the event deletion interfaces of the event server or the rapid event purge utility of the default data store plug-in
- A large number of events have been inserted into the database
- Tables have been reorganized using the **reorg** script
- Indexes have been added or removed from a table

The **runstats** script is located in the $WAS_HOME/event/dbscripts/db2 directory.

To update the database statistics, run one of the following commands:

- On Windows systems:
  `runstats.bat db_user [password=db_password]`
- On Linux and UNIX systems:
  `runstats.sh db_user [password=db_password]`

The parameters are as follows:

db_user
The database user ID to use. This parameter is required.

db_password
The database password. This parameter is optional; if you do not specify the password on the command line, the DB2 database prompts you.

For example, the following command updates the DB2

database statistics on a Windows

system, using the database user ID `dbadmin` and the password `mypassword`:

```
runstats.bat dbadmin mypassword
```

Reorganizing database tables

After events are purged or deleted from a DB2 event database, reorganize the database tables using the **reorg** script.

The **reorg** script is located in the $WAS_HOME/event/dbscripts/db2 directory.

To reorganize the event database tables, run one of the following commands:

- On Windows systems:

  `reorg.bat db_alias db_user [password=db_password]`
- On Linux and UNIX systems:

  `reorg.sh db_alias db_user [password=db_password]`

The parameters are as follows:

db_alias
: The database alias. The event database must be catalogued on the DB2 client; if you are running the script on the DB2 server, the database is already catalogued.

db_user
: The database user ID to use. This parameter is required.

db_password
: The database password. This parameter is optional; if you do not specify the password on the command line, the DB2 database prompts you.

For example, the following command reorganizes the event database tables on a Windows

system, using the database alias `eventdb`, user ID `dbadmin` , and the password `mypassword`:

```
reorg.bat eventdb dbadmin mypassword
```

After running the **reorg** script, you should update the database statistics using the **runstats** script. For more information, see "Updating database statistics" on page 40.

Purging events from the event database

You can use the provided scripts to rapidly purge large numbers of events from the event database.

The default data store plug-in provides a set of utilities you can use to periodically perform a rapid purge of large numbers of old events from the event database. These utilities are different from the **eventpurge.jacl** event server command, which deletes events matching specified criteria.

The rapid purge capability uses the concept of *buckets*. A bucket is a set of tables used to store events in the event database. The default data store plug-in uses two buckets:

- The active bucket is the bucket containing the most recent events; new events are stored in the active bucket. The active bucket cannot be purged using the rapid purge utility.

- The inactive bucket contains older events. Events stored in the inactive bucket can be queried, deleted, or modified, but typically no new events are stored in the inactive bucket. The inactive bucket can also be purged by the rapid purge utility.

Each event is stored in only one bucket. From the perspective of an event consumer, there is no distinction between the active and inactive buckets; a consumer can query, modify, or delete a specific event without knowing which bucket the event is stored in. The advantage of this approach is that the inactive bucket can be rapidly purged using database-specific interfaces without affecting the active bucket; usual event traffic can continue even while the purge operation is taking place.

After the inactive bucket is purged, you can then swap the buckets so that the active bucket becomes inactive and the inactive bucket becomes active. You can swap the buckets only when the inactive bucket is empty.

Note: Although new events are generally stored only in the active bucket, under some circumstances events might be stored in the inactive bucket immediately after the buckets are swapped. The data store plug-in checks periodically to determine which bucket is currently marked as active, but until the next check takes place, some events might continue to be stored in the inactive bucket. Similarly, events sent as part of a batch are all stored in the same bucket, even if that bucket becomes inactive while the batch is still being processed.

If you want to use the fast purge capability, it is your responsibility to determine how frequently to swap buckets or purge the inactive bucket, depending upon event traffic, storage space, archival requirements, or other considerations.

Viewing or changing the active bucket status

The active bucket status indicates which bucket is currently active and which is currently inactive.

To view or change the active bucket status, use the **eventbucket.jacl** script:

```
wsadmin -f eventbucket.jacl [-status] [-change] [-serverName server_name]
```

This command has the following options:

-status
Use this option to see information about the current bucket configuration, including the active bucket setting and the bucket check interval (the frequency with which the data store plug-in checks to determine which bucket is active).

-change
Use this option to swap the active and inactive buckets. The inactive bucket must be empty before you can use this option.

-serverName *server_name*
Use this option to specify the name of the application server where the event server is deployed. Specify this parameter if there are multiple application servers running on the WebSphere node.

Purging the inactive bucket

The method used to purge the inactive bucket varies depending on the database software.

Note: The rapid purge utility is not supported for a Cloudscape event database.

Purging the inactive bucket for a DB2 database (Linux, UNIX, or Windows systems):

On Linux, UNIX, and Windows systems, the rapid purge utility for a DB2 database is implemented as a shell script or batch file.

To purge the inactive bucket, use one of the following commands:

- To purge the inactive bucket on a Windows system, run the following command:

  `fastpurge dbalias dbuser [password=dbpassword] [copydir=copydir]`
- To purge the inactive bucket on a Linux or UNIX system, run the following command:

  `fastpurge.sh dbalias dbuser [password=dbpassword] [copydir=copydir]`

The parameters of this command are as follows:

dbalias
: The database alias. The event database must be catalogued on the DB2 client; if you are running the script on the DB2 server, the database is already catalogued.

dbuser
: The database user ID to use when connecting to the event database.

dbpassword
: The password to use for the specified user ID. This parameter is optional; if you do not specify the password, the DB2 database prompts for it.

copydir
: The path to a directory to be used for the files generated by the load utility. This parameter is required only if you have enabled forward recovery for the event database (with the LOGRETAIN or USEREXIT database configuration settings enabled). By default, the event database does not use forward recovery.

Purging the inactive bucket for a DB2 database (z/OS systems):

On z/OS systems, the rapid purge utility for a DB2 event database is implemented using the DB2 load utility.

To purge the inactive bucket:

1. Use the **eventbucket.jacl** command to identify the inactive bucket (bucket 0 or bucket 1).
2. Upload the appropriate utility control file. These files are generated during database configuration and are located in the *profile_path*/event/dbscripts/db2zos directory (*profile_path* is the path to the directory containing the profile for the WebSphere Application Server runtime environment. Upload one of the following files:
   - If bucket 0 is inactive, upload fastpurge00.ctl.
   - If bucket 1 is inactive, upload fastpurge01.ctl.

   Note: The control file must be uploaded with a fixed record format and a logical record length of 80.
3. On the z/OS host, go to the ISPF DB2I Primary Option Menu and select the **Utilities** option.

4. Specify the following information:

Field	Value
Function	EDITJCL
Utility	LOAD
Statement Data Set	The name of the data set containing the uploaded control file
LISTDEF	NO
Template	NO

5. Press the Enter key to continue to the next panel.
6. In the recdsn entry field, specify the name of the data set containing the uploaded control file.
7. Press the Enter key. The JCL script to purge the inactive bucket is generated.
8. Press the Enter key to clear the output messages.
9. Edit the generated JCL script as needed.
10. Submit the JCL script.

Purging the inactive bucket for an Oracle database:

The rapid purge utility for an Oracle event database is implemented as a stored procedure.

To use the utility for an Oracle database, you must have SQL*Plus installed on the Oracle client, and the client must be configured to communicate with the Oracle database (the tnsnames.ora file must be configured properly). Refer to the Oracle documentation for more information.

To purge the inactive bucket, run the stored procedure using SQL*Plus:

```
sqlplus connect_string @fastpurge.sql
```

The *connect_string* parameter is the Oracle connection string. Use the same database user ID used to create the event database.

Purging the inactive bucket for an Informix database:

The rapid purge utility for an Informix event database is implemented as a stored procedure.

To use the utility for an Informix database, you must run the Informix dbaccess command in an environment that has been properly sourced for the Informix environment variables. Refer to the Informix documentation for more information.

To purge the inactive bucket, run the stored procedure using dbaccess:

```
dbaccess - fastpurge.sql
```

Purging the inactive bucket for a Sybase database:

The rapid purge utility for a Sybase event database is implemented as a stored procedure.

To use the utility for a Sybase database, you must have the isql utility installed on the Sybase client. Refer to the Sybase documentation for more information.

To purge the inactive bucket, run the stored procedure using isql:

```
isql -Sserver_name -Udbuser -Pdbpassword -Deventdb -i fastpurge.sql
```

The parameters of this command are as follows:

server_name
The database server name.

dbuser
The database user ID to use when connecting to the event database.

dbpassword
The password to use for the specified user ID. This parameter is optional; if you do not specify the password, the isql utility prompts for it.

eventdb
The name of the event database (typically `event`).

Purging the inactive bucket for a SQL Server database:

The rapid purge utility for a SQL Server event database is implemented as a stored procedure.

To use the utility for a SQL Server database, you must have the osql utility installed on the SQL Server client. Refer to the SQL Server documentation for more information.

Note: Do not use the isql utility for running the rapid purge stored procedure.

To purge the inactive bucket, run the stored procedure using osql:

```
osql -Sserver_name -Udbuser -Pdbpassword -deventdb -Q"fast_purge"
```

The parameters of this command are as follows:

server_name
The database server name.

dbuser
The database user ID to use when connecting to the event database.

dbpassword
The password to use for the specified user ID. This parameter is optional; if you do not specify the password, the osql utility prompts for it.

eventdb
The name of the event database (typically `event`).

Changing the bucket check interval

The bucket check interval specifies how frequently the data store plug-in checks to determine which bucket is active. This value is specified as a custom property of the data store profile.

The default bucket check interval is 5 minutes (300 seconds). A shorter interval reduces the likelihood that events will be stored in the inactive bucket after swapping, but might decrease performance.

To change the bucket check interval:

1. In the WebSphere Administrative Console, navigate to **Common Event Infrastructure Provider > Data Store Profile > Default Common Event Infrastructure data store > Custom properties**.

2. Modify the value of the BucketCheckInterval property to specify the bucket check interval in seconds.

Chapter 5. Working with events

The Common Event Infrastructure represents events as Java objects. Specifically, each event is an instance of a class implementing the org.eclipse.hyades.logging.events.cbe.CommonBaseEvent interface, which is a Java representation of the Common Base Event specification. For more information about this specification, see "The Common Base Event model" on page 2. The org.eclipse.hyades.logging.events.cbe package is part of the Eclipse Test and Performance Tools (TPTP) platform, which is a set of standards and open-source tools for testing, tracing, and monitoring. For more information, see http://www.eclipse.org/tptp/.

The typical life cycle of an event is as follows:

1. To send an event, an event source creates a new instance of CommonBaseEvent, populates it with property data, and then submits it to an emitter.
2. The emitter optionally uses the content completion mechanism (if implemented) to populate the event with required property data. The emitter then validates the event and checks it against the currently configured filter criteria. If the event is valid and passes the filter criteria, the emitter sends the event to the event server. For more information about event processing by the emitter, see "Sending events" on page 56.
3. If persistence is enabled, the event server stores the event in a persistent data store.
4. If publishing is enabled, the event server publishes the event to one or more Java Messaging Service (JMS) destinations. Event consumers subscribing to these destinations then receive notifications of the new event. The event consumers then use the Notification Helper to convert the received JMS messages back into a CommonBaseEvent instance.

   An event consumer might also submit a query to retrieve the event from the data store. Typically, a consumer uses the query interface to retrieve historical events, especially during startup processing.

   After receiving the event, an event consumer reads the event property data and processes the event.

   For more information about event consumers, see Chapter 7, "Developing an event consumer," on page 63.
5. When it is no longer needed, the event can be purged from the data store.

The Common Base Event specification, which is based on the XML Schema definition language, defines two kinds of event property data:

- Properties represented by simple data types, encoded in XML as attributes of the CommonBaseEvent element. These include properties such as *globalInstanceId*, *severity*, and *msg*. The CommonBaseEvent Java class represents these values as strings or integers, as appropriate.
- Properties represented by complex data types and encoded in XML as subelements of the CommonBaseEvent element. These include properties such as *situation*, *sourceComponentId*, and *extendedDataElements*, each of which has nested properties of its own. These complex types are represented by specialized Java classes defined in the org.eclipse.hyades.logging.events.cbe package. For example, the *sourceComponentId* property is represented by an instance of ComponentIdentifier.

The CommonBaseEvent interface defines getter and setter methods for each property, as well as helper methods to simplify creation of complex properties. An event source uses the setter methods (or the helper methods) to populate an event with property data before submitting it to an emitter; an event consumer uses the getter methods to retrieve the property data from a received event.

For more information about the XML Schema specification, see http://www.w3.org/XML/Schema.

Creating an event object

To create new events in your event source, you use an *event factory*, which is an object that returns new instances of CommonBaseEvent or of the specialized classes representing complex property data types.

There are two ways you can access an event factory:

- You can create a new event factory using the *event factory factory*. Use this approach if no appropriate event factory is already available. When you create a new event factory, you can optionally specify a content handler to provide automatic content completion. For more information about content completion, see "Completing event content automatically" on page 50.
- You can use an existing event factory that has been bound into a Java Naming and Directory Interface (JNDI) namespace. Use this approach if an administrator has provided an event factory for you to use; this ensures that any events you create conform to the appropriate business rules, because the event factory might be configured with a content handler.

Creating a new event factory

To create a new event factory, use the event factory factory, implemented as the class EventFactoryFactory. This class has no instances; instead, it provides two static methods used to create event factories. The choice of which method to use depends upon whether you want to use a content handler to implement automatic content completion. For more information, see "Completing event content automatically" on page 50.

To create a generic event factory with no content handler, use the createEventFactory() static method of EventFactoryFactory:

```
EventFactory eventFactory =
     (EventFactory) EventFactoryFactory.createEventFactory();
```

To create an event factory with a content handler, use the createEventFactory(ContentHandler) method, specifying the content handler you want to use:

```
EventFactory eventFactory =
     (EventFactory) EventFactoryFactory.createEventFactory(contentHandler);
```

In either case, the returned object is an event factory you can use to create new events.

Getting an event factory by JNDI lookup

If an administrator has bound an existing event factory into JNDI for event sources to use, perform a standard JNDI lookup to retrieve the event factory:

```
import javax.naming.*
import org.eclipse.hyades.logging.events.cbe.*

Context context = new InitialContext();
EventFactory eventFactory =
     (EventFactory) context.lookup("com/ibm/events/EventFactory");
```

The returned object is the provided event factory; if the event factory is configured with a content handler, an instance of the content handler is also created locally. For more information about content handlers and JNDI, see "Completing event content automatically" on page 50.

Creating and populating an event

After obtaining an event factory, you can create event objects and populate them with property data. Most event properties are defined as optional by the Common Base Event specification, but the following properties are required:

- *version* (a string attribute)
- *creationTime* (an XML Schema dateTime attribute; see http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/#dateTime)
- *sourceComponentId* (a complex ComponentIdentification element)
- *situation* (a complex Situation element)

Note: The *version* attribute is defined as optional by the Common Base Event specification, but if it is not specified, the default value `1.0` is assumed. Because the Common Event Infrastructure supports only version 1.0.1 of the specification, this value must be specified.

If you try to send an event that is missing any of these properties, the emitter rejects the event and throws an EventsException exception.

The following code fragment creates an event and populates it with the minimal required property data:

```
CommonBaseEvent event = eventFactory.createCommonBaseEvent();

event.setVersion("1.0.1");                    // set version

long currentTime = System.currentTimeMillis(); // get current time
event.setCreationTimeAsLong(currentTime);      // and set creationTime

// set sourceComponentId (a complex type)
event.setSourceComponentId("Windows",          // application
                           "svchost.exe",      // component
                           "tlntsvr.exe",      // subcomponent
                           "http://www.ibm.com/namespaces/autonomic/Windows",
                                               // componentType
                           "win386_svc",       // componentIdType
                           "9.45.72.138",      // location
                           "IPV4"              // locationType
                           );

// create situation object
Situation situation = eventFactory.createSituation();

// set situationType to AvailableSituation (a complex type)
situation.setAvailableSituation("EXTERNAL",          // reasoningScope
                                "NOT AVAILABLE",     // availabilityDisposition
                                "STARTABLE",         // operationDisposition
```

```
                                        "FUNCTION_PROCESS"); // processingDisposition

// set situation
event.setSituation(situation);
```

This example first uses an event factory to create a new event instance, *event*. First it sets the *version* property; then it retrieves the current system time and uses the setCreationTimeAsLong(long) method to set the value of the *creationTime* property. An alternative is to use the setCreationTime(String) method, which sets the creation time using the XML dateTime format (for example, `"2004-07-29T13:12:00-05:00"`).

The next required property, *sourceComponentId*, is a complex property represented by an instance of ComponentIdentification, which has properties of its own. However, it is not necessary to directly instantiate or interact with this object (although it is possible to do so). Instead, the next statement in the example uses a helper method, setSourceComponentId(), to specify the nested properties; the helper method uses these values to create an instance of ComponentIdentification, which it then uses to set the value of the *sourceComponentId* property of the event.

Similar helper methods exist for setting other complex properties (for example, setMsgDataElement(), addAssociatedEvent, and addExtendedDataElement()). Many of these methods exist in multiple versions with different signatures, making it possible to specify property values in different ways. Refer to the Javadoc API documentation for complete information on these methods.

The last required property in the example, *Situation*, is another complex property. In this case the situation object must be instantiated directly using an event factory; the example then uses a helper method to set the *situationType* property, which is itself a complex subelement.

In an actual application, a useful event needs to include more information than is shown here, but this is the minimum required by the Common Base Event specification and the Common Event Infrastructure. The event is now valid and can be submitted to an emitter.

Completing event content automatically

In some situations, you might want some event property data to be automatically set for every event you create. This is a way to fill in certain standard values that do not change (such as the application name), or to set some properties based on information available from the runtime environment (such as creation time or thread information). You can also set policies that govern event content according to business rules; for example, you might require that any event with a particular extension name have its severity set to a certain value.

You can do this by creating a *content handler*. A content handler is an object that automatically sets the property values of each event based on any arbitrary policies you want to use. The Common Event Infrastructure does not restrict how a content handler can modify event data, so long as the event still conforms to the Common Base Event specification.

To ensure that all event sources comply with the same policies, you can create an event factory associated with a content handler (using EventFactoryFactory) and then bind the created event factory into a JNDI namespace. Instead of creating their own event factories, event sources can then perform JNDI lookups to access

the event factory that already exists, without any knowledge of the content handler. If your business rules later change, you can modify the content handler in one place.

An event source does not need to do anything to enable content completion. If an event factory is associated with a content handler, each event it creates carries with it a reference to that content handler. When the event is submitted to an emitter, the event calls the completeEvent() method of the content handler, passing a reference to itself. This ensures that the correct policies are applied to the event after the event source is finished setting event-specific properties, but before the event is validated and processed by the emitter.

Note: When an event is transmitted from one process to another, the reference to the content handler is not transmitted with it. This is because content completion relies upon the environment where the event originates, and the necessary information might not be available elsewhere. This restriction does not affect calls between applications that are local to one another (for example, a call to an enterprise bean using its local interface).

To create a content handler, follow these steps:

1. Create a new Java class implementing the org.eclipse.hyades.logging.events.cbe.ContentHandler interface. This interface defines a single method called completeEvent(CommonBaseEvent); the parameter is the event whose content is to be completed. In your implementation of this method, you can use the getter and setter methods of CommonBaseEvent to process the event property data in accordance with any policies that apply.

   Note: When an event source uses JNDI to retrieve an event factory, the content handler is returned along with the event factory. For this reason, the content handler must be serializable.

   The following example is a simple content handler that automatically sets the extension name of each event:

   ```
   import java.io.Serializable;
   import org.eclipse.hyades.logging.events.cbe.*;

   public class BusinessContentHandler
        implements ContentHandler, Serializable {

          public void completeEvent(CommonBaseEvent event)
             throws CompletionException {
                 event.setExtensionName("business");
        }
   }
   ```

2. Associate the content handler with an event factory. To do this, specify the content handler when creating the event factory:

   ```
   EventFactory eventFactory =
        (EventFactory) EventFactoryFactory.createEventFactory(contentHandler);
   ```

   The returned event factory is permanently associated with the specified content handler.

Retrieving data from a received event

When an event source receives an event, it can then use the getter methods of CommonBaseEvent to retrieve the event property data. For example, the following code fragment retrieves a single event and then reads the content of the *msg* property.

```
CommonBaseEvent event = eventAccess.queryEventByGlobalInstanceId(eventId);
String eventMessage = event.getMsg();
```

If the property you want to retrieve is a complex property (a subelement of CommonBaseEvent in the Common Base Event specification), the returned value is an instance of the specialized class representing the complex data type. You can then use the getter methods of the returned object to retrieve the property data from that object. For example, the following code fragment retrieves the value of *componentId*, which is a complex property; it then retrieves the content of the nested *component* property, which is a string, to read the name of the source component.

```
CommonBaseEvent event = eventAccess.queryEventByGlobalInstanceId(eventId);
ComponentIdentification componentId = event.getSourceComponentId();
String componentName = componentId.getComponent();
```

Converting XML events

In addition to creating new events, an event source might convert events received from other applications in XML format. Similarly, an event consumer might convert events to XML format for forwarding to another application. The org.eclipse.hyades.logging.events.cbe.EventFormatter class provides methods you can use to convert between CommonBaseEvent instances and XML.

Using EventFormatter, you can convert an instance of CommonBaseEvent into a string containing either an XML document or an XML fragment. Similarly, you can convert from an XML document or fragment into an instance of CommonBaseEvent.

For more information about EventFormatter, refer to the Javadoc documentation in the org.eclipse.hyades.logging.events.cbe package.

Accessing event instance metadata

The org.eclipse.hyades.logging.events.cbe package, which provides the classes and interfaces required for working with event objects, is based on the Eclipse Modeling Framework (EMF). EMF is a Java framework used to generate application code based on a structured data model; it also provides interfaces in the generated code that can be used to access metadata describing the data model. (Refer to the Eclipse Modeling Framework documentation at http://www.eclipse.org/emf for more information about EMF.)

By using these interfaces, EMF-compatible tools can interact with CommonBaseEvent event data without any prior knowledge of the data model or access to the implementation. This makes it possible for development tools to generate code that transfers data from other data models into the CommonBaseEvent model. Application developers can then focus on writing code that uses the data rather than code that builds the data.

For example, consider an event source that monitors network events and describes its own data model in terms of EMF. With access to both data models, a development tool could display the fields of the event source data model alongside the fields of the CommonBaseEvent data model. A developer could then use a graphical interface to indicate how the fields in the event source model are mapped to fields in the CommonBaseEvent model; for example, a field called Workstation.name in the event source data model might correspond to the CommonBaseEvent.sourceComponentId.location field in the CommonBaseEvent data model. Because both data models are described using standard EMF interfaces, the tool could generate code that handles the transfer of data between the two models.

The following code fragment is a simplified example of how a development tool might use EMF interfaces to query information about the CommonBaseEvent data model and then use that information to interact with an event instance. This example could be part of a simple event consumer; it iterates through all of the fields of an event instance and, for each one, prints the name and value of the field.

```
// event is a valid CommonBaseEvent instance

// Get list of event instance structural features (fields)
List features = event.eClass().getEAllStructuralFeatures();

// iterate through list; print names and values
for (int i = 0 ; i < features.size() ; i++)
  {
    EStructuralFeature feature = (EStructuralFeature)features.get(i);
    Object value = eObj.eGet(feature);
    System.out.println(feature.getName() + ":" + value);
  }
```

The CommonBaseEvent data model is described in the EMF files cbe.ecore and cbe.genmodel. These files are included with the Common Event Infrastructure SDK; you can import them into an Eclipse-based development environment and then use EMF to generate code that interacts with CommonBaseEvent objects.

Chapter 6. Developing an event source

An *event source* is any application that uses an emitter to send events to the event server. The following applications are examples of event sources:

- An adapter or monitor that generates events related to monitored resources
- An application that generates notification events
- An application that forwards events from other sources

An event source is implemented in the Java programming language, using either the Java 2 Platform, Standard Edition (J2SE) or the Java 2 Platform, Enterprise Edition (J2EE). An event source must submit valid events conforming to the Common Base Event model (for more information, see "The Common Base Event model" on page 2). Each event is represented as a Java object.

Emitters and emitter factories

An event source does not interact directly with the event server; instead, it interacts with an object called an emitter (an implementation of the com.ibm.events.emitter.Emitter interface). An emitter is a local object that provides methods for sending events.

In general, the emitter handles the details of event transmission; the developer of an event source does not need to be concerned about the event server location, the filter settings, or the underlying transmission mechanism. Details such as these are governed by the *emitter factory*, an object configured by an administrator and bound in a Java Naming and Directory Interface (JNDI) namespace. An emitter factory is an an instance of com.ibm.events.emitter.EmitterFactory and is used to create emitter objects. It also defines the behavior of the emitters it creates; it includes settings for the following:

- The preferred *transaction mode*. This setting specifies whether the emitter attempts to send each event in a new transaction or within the current transaction. An event source can change this setting for a particular emitter or event submission, but the profile specifies the default value. (This setting is valid only in a J2EE container; the J2SE platform does not provide transaction controls.)
- The preferred *synchronization mode*. This setting specifies whether events are sent using synchronous or asynchronous transmission. *Synchronous transmission* means that the sendEvent() method does not return control to the caller until the event has been processed; *asynchronous transmission* means that the method returns immediately after the event is submitted, and the caller has no further information about event processing. An event source can change this setting for an emitter or for an event submission, but the default value is specified by the profile.
- The *transmission profiles* to use. A transmission profile is a configuration object that defines a specific transmission mechanism for sending events to the event server. An emitter factory profile can specify two transmission profiles, one for synchronous transmission and one for asynchronous transmission. An event source cannot change the transmission profiles used by an emitter.
- The filter configuration to use for the emitter. The filter configuration defines what filtering plug-in is used to filter events submitted to the emitter. The Common Event Infrastructure includes a default filter plug-in, but you can also

implement your own filter plug-in if you want to use a different filtering engine. For more information, see "Filtering events" on page 61.

An administrator can create multiple emitter factory profiles, each one defining a different emitter configuration. An event source obtains an emitter using the emitter factory associated with an existing emitter factory profile; therefore, all emitters created by a particular emitter factory will have the same default behavior. For more information, see "Obtaining an emitter."

Note: If your event source is running with Java 2 security enabled, and you want to generate your own globally unique identifiers (GUIDs), you must modify your policy file to enable correct processing. Add the following entries:

```
permission java.io.FilePermission "${java.io.tmpdir}${/}guid.lock",
  "read, write, delete";
permission java.net.SocketPermission "*", "resolve";
```

Obtaining an emitter

Before you can obtain an emitter, there must be at least one emitter factory profile configured. For each emitter factory profile, an emitter factory is automatically created and is accessible using the JNDI name of the emitter factory profile. For more information, see Chapter 3, "Configuration," on page 19.

To obtain an emitter, follow these steps:

1. Perform a JNDI lookup specifying the name of the emitter factory you want to use for your emitter. This is the JNDI name specified by an administrator when defining an emitter factory profile.
2. Call the *getEmitter()* method of the emitter factory. The returned object is an emitter configured according to the options defined in the emitter factory profile you specified. If the emitter factory is unable to obtain an emitter, it throws an EmitterException exception.

   Note: If your event source is a J2EE client application running in a secure environment, and the emitter profile you are using specifies asynchronous transmission profiles, you must specify a JMS user name and password in order to get an emitter. To do this, use the *getEmitter(String, String)* method, passing the JMS user name and password you want to use. For more information, refer to the Javadoc documentation for the com.ibm.events.emitter class.

The following code fragment obtains an emitter configured with the profile *Default*:

```
import javax.naming.*
import com.ibm.events.emitter.*

Context context = new InitialContext();
EmitterFactory emitterFactory =
  (EmitterFactory) context.lookup("com/ibm/events/configuration/emitter/Default");
Emitter emitter = emitterFactory.getEmitter();
```

Sending events

An event source sends events in the form of Java objects. Specifically, each event is an instance of a class implementing the org.eclipse.hyades.logging.events.cbe.CommonBaseEvent interface, which is a Java representation of the Common Base Event specification. For more information, see "The Common Base Event model" on page 2.

To send an event, use the sendEvent() methods of the Emitter interface. When you submit an event to an emitter, the following occurs:

1. The emitter calls the complete() method of the event, triggering optional content completion. See "Completing event content automatically" on page 50 for more information.
2. The emitter assigns a sequence number and global instance identifier to any event that does not already have them.
3. The emitter validates the event to ensure that it conforms to the Common Base Event specification.

   Note: The current Common Base Event specification allows only one extended data element with a given name at each level of the event containment hierarchy, but this restriction will not be included in future versions of the specification and is not enforced by the Common Event Infrastructure.
4. If filtering is active, the emitter checks the event against the current filter criteria to determine whether the event should be sent or discarded.
5. Finally, if the event is valid and passes the filter criteria, the emitter sends the event to the event server for persistence and distribution to event consumers.

If the event is not valid, or if the emitter encounters a problem when trying to send the event to the event server, an exception is thrown.

Note: If WebSphere security is enabled, the application user ID must be mapped to the eventCreator or eventAdministrator role to send events using synchronous EJB transmission.

Sending an event with the current emitter settings

If you do not need to specify a particular transmission mode or transaction mode, you can send an event using the current emitter settings. These settings are initially defined by an administrator in the emitter factory profile, but they can later be changed by event consumers.

To send an event using the current emitter settings, use the `sendEvent(CommonBaseEvent)` method.

```
String eventId = emitter.sendEvent(event);
```

In this example, `emitter` is an Emitter instance, and `event` is a CommonBaseEvent instance.

The returned value, `eventId`, is the globally unique identifier of the event (the value of the *globalInstanceId* field of CommonBaseEvent). If the event does not have a global instance identifier when you submit it, the emitter assigns one automatically.

Note: Submitting an event to an emitter does not guarantee that the event is sent to the event server, because the filter settings might cause the event to be discarded. A successful call to sendEvent() means only that the event was successfully processed by the emitter.

Overriding the current emitter settings

When sending an event, you can specify options that override the current transaction mode, synchronization mode, or both, currently configured for the emitter. These settings are initially defined by an administrator in the emitter factory profile, but they can later be changed by event consumers.

An emitter might not support all synchronization and transaction modes. The available modes are subject to the following limitations:

- The synchronization modes supported by an emitter are defined by the emitter factory profile. You can find out which modes are supported by a particular emitter by calling the isSynchronizationModeSupported() method; see the Javadoc API documentation for com.ibm.events.emitter.Emitter for more information.
- Transactions are supported only in a J2EE container.

If you attempt to use a mode that is not supported, the emitter throws a TransactionModeNotSupportedException or SynchronizationModeNotSupportedException exception.

To override the emitter settings, use the sendEvent(CommonBaseEvent, int, int) method.

```
String eventId = emitter.sendEvent(event,
                                   synchronizationMode,
                                   transactionMode);
```

The parameters are as follows:

event
The event object (an instance of CommonBaseEvent) you want to send.

synchronizationMode
An integer constant defined by the interface SynchronizationMode. This should be one of the following constants:

- SynchronizationMode.ASYNCHRONOUS (send the event asynchronously)
- SynchronizationMode.SYNCHRONOUS (send the event synchronously)
- SynchronizationMode.DEFAULT (use the current emitter setting)

transactionMode
An integer constant defined by the interface TransactionMode:

- TransactionMode.NEW (send the event in a new transaction)
- TransactionMode.SAME (send the event in the current transaction)
- TransactionMode.DEFAULT (use the current emitter setting)

The event is sent with the options you specify. These options apply only to the single event being sent; no changes are made to the emitter settings, and subsequent event submissions are not affected.

The returned value, `eventId`, is the globally unique identifier of the event (the value of the `globalInstanceId` field of CommonBaseEvent). If the event does not have a globalInstanceId when you submit it, the emitter assigns one automatically.

Note: Submitting an event to an emitter does not guarantee that the event is sent to the event server, because the filter settings might cause the event to be discarded. A successful call to sendEvent() means only that the event was successfully processed by the emitter.

The following example overrides the emitter setting to send an event in a new transaction, but does not override the synchronization mode:

```
String eventId = sendEvent(event,
                           SynchronizationMode.DEFAULT,
                           TransactionMode.NEW);
```

Sending multiple events

If your event source needs to send multiple events in a batch, you can improve performance by sending them with a single call to the sendEvents() method.

Batching events in this way can also be useful for logical groups of events that should be sent only if an underlying transaction has completed successfully. All of the submitted events are sent as part of a single transaction.

- To send multiple events with the current emitter settings, use the `sendEvents(CommonBaseEvent[])` method:

  ```
  String[] eventIds = emitter.sendEvents(events);
  ```

  In this example, `emitter` is an Emitter instance, and `events` is an array of CommonBaseEvent instances.
- To send multiple events and override the current emitter settings, use the sendEvents(CommonBaseEvent, int, int) method, specifying the synchronization mode and transaction mode you want to use:

  ```
  String eventId = emitter.sendEvent(event,
                                     synchronizationMode,
                                     transactionMode);
  ```

The returned value, `eventIds`, is an array containing the globally unique identifiers of the sent events.

Each event is validated and checked against the current filter criteria. All of the valid events that pass the filter criteria are then sent using the appropriate mechanism:

- If you are using synchronous event transport, the events are sent using a single EJB call. If an error occurs during the EJB call, an exception is thrown and none of the events are sent.
- If you are using asynchronous event transport, all of the events are sent using a single JMS message. If an error occurs during JMS processing, an exception is thrown and none of the events are sent.

Changing the emitter settings

An event source can make changes to the transaction mode and synchronization mode configured for the emitter. These settings are initially defined by the emitter factory profile. In addition, an event source can query the current transaction mode to determine what setting is currently in effect for the emitter.

Changing the synchronization mode

An event source can change the synchronization mode being used by an emitter. This change remains in effect for subsequent event submissions, but it does not change the preferred synchronization mode defined in the emitter factory profile.

The synchronization modes supported by an emitter are defined by the emitter factory profile. You can find out which modes are supported by a particular emitter by calling the isSynchronizationModeSupported() method; see the Javadoc

API documentation for com.ibm.events.emitter.Emitter for more information. If you attempt to use a mode that is not supported, the emitter throws a SynchronizationModeNotSupportedException exception.

To change the synchronization mode, use the setSynchronizationMode(int) method.

```
emitter.setSynchronizationMode(synchronizationMode);
```

The *synchronizationMode* is an integer constant defined by the interface SynchronizationMode:

- SynchronizationMode.ASYNCHRONOUS (send the event asynchronously)
- SynchronizationMode.SYNCHRONOUS (send the event synchronously)
- SynchronizationMode.DEFAULT (send the event using the current emitter settings)

Changing the transaction mode

An event source can change the transaction mode being used by an emitter. This change remains in effect for subsequent event submissions, but it does not change the transaction mode defined in the emitter factory profile.

Note: Transactions are supported only in a J2EE container.

To change the transaction mode, use the setTransactionMode(int) method.

```
emitter.setTransactionMode(transactionMode);
```

The *transactionMode* is an integer constant defined by the interface TransactionMode:

- TransactionMode.NEW (send the event in a new transaction)
- TransactionMode.SAME (send the event in the current transaction)
- TransactionMode.DEFAULT (send the event using the current emitter settings)

Querying the transaction mode

An event source can query the transaction mode currently being used by an emitter.

Note: Transactions are supported only in a J2EE container.

To query the current transaction mode, use the getTransactionMode() method.

```
int transactionMode = emitter.getTransactionMode();
```

The returned value is an integer corresponding to one of the transaction mode constants:

- TransactionMode.NEW
- TransactionMode.SAME

Freeing emitter resources

If your event source has finished sending events with a particular emitter, you should free the resources the emitter is using.

To free the emitter resources, use the close() method:

```
emitter.close();
```

This method releases all resources being used by the emitter.

Filtering events

An emitter can optionally be configured to filter events at the source.

Event filtering provides a mechanism for reducing event traffic by screening out events that are not important. Each time an event source submits an event to an emitter, the emitter checks the event against the current filter criteria. If the event passes the filter criteria, the emitter sends the event to the event server; otherwise, the emitter discards the event. In any case, an event source cannot change the filter settings, which are configured by an administrator.

The emitter filter is implemented as a separate component called a *filter plug-in*. The Common Event Infrastructure includes a default filter plug-in, which provides filtering of submitted events based on XPath event selectors. If you want to use a different filter mechanism, you can implement your own filter plug-in.

In the Common Event Infrastructure configuration, each emitter factory is associated with a *filter factory*. A filter factory is an object used to create instances of a filter plug-in. When you create an emitter using an emitter factory, the emitter is automatically associated with an instance of the specified filter plug-in, which provides filtering of events submitted to that emitter.

Filtering events with the default filter plug-in

The Common Event Infrastructure includes a default emitter filter plug-in, which can be configured with an XPath event selector defining which events are sent to the event server and which are discarded. For example, the filter settings might specify that only events with severity greater than 20 (harmless) should be sent.

To filter events using the default filter plug-in, follow these steps:

1. In the WebSphere administrative console, navigate to the **Common Event Infrastructure Provider > Filter Factory Profile** page.
2. Create a new filter factory profile. For more information, see the online help for the administrative console.
3. In the **Filter Configuration String** field, type an XPath event selector describing events you want to use for filtering events. Events matching this event selector are sent to the event server; events that do not match are discarded by the emitter. For information about writing event selectors, see "Writing event selectors" on page 78.
4. Navigate to the **Common Event Infrastructure Provider > Emitter Factory Profile** page.
5. Create a new emitter factory profile, or go to an existing emitter factory profile. For more information, see the online help for the administrative console.
6. In the **Filter Factory JNDI Name** field, type the JNDI name of the new filter factory profile you created.

Event sources can now use the new emitter factory to create instances of an emitter using the new filter configuration. If you later want to adjust the filter settings for event sources using that emitter factory, you can modify the event selector specified in the filter factory.

You can use tracing to find out what events are being discarded by the default filter plug-in; for more information, see "Logging and tracing" on page 39.

Note: The default filter plug-in uses the Apache JXPath component to process XPath event selectors. If Java 2 security is enabled, you must modify your policy file to include entries allowing the required access:

```
permission java.util.PropertyPermission "*", "read";
permission java.io.FilePermission
  "${was.install.root}${/}java${/}jre${/}lib${/}jxpath.properties",
  "read";
```

Implementing a filter plug-in

If you want to use your own filtering engine as an emitter filter, you can implement a custom filter plug-in by following these steps:

1. Develop your filter plug-in as a Java class implementing the interface com.ibm.events.filter.Filter. This interface defines the following methods:

   isEventEnabled(CommonBaseEvent)
   Returns a boolean value indicating whether the specified event passes the filter criteria. Each time an event is submitted to an emitter, the emitter calls this method, passing the submitted event. If the return value is `true`, the emitter sends the event to the event server for persistence and distribution. If the return value is `false`, the emitter discards the event.

   getMetaData()
   Returns information about the filter plug-in, such as the provider name and version number.

   close() Frees all resources used by the filter plug-in. This method is called when the close() method of an emitter is called.
2. Develop a filter factory class that implements the interface com.ibm.events.filter.FilterFactory. This interface defines a single method, getFilter(), which returns an instance of your filter class (an implementation of the Filter interface).
3. Bind an instance of your filter factory into a JNDI namespace. During initialization, an emitter performs a JNDI lookup to access the filter factory.
4. In the WebSphere Application Server administrative console, modify your emitter factory profile or create a new profile. In the **Filter Factory JNDI Name** field, specify the JNDI name of your FilterFactory implementation. For more information about emitter factory profiles, see the online help for the administrative console.

When you create an emitter using the emitter factory profile that specifies your filter factory, the new emitter uses an instance of your filter implementation. You can now send events using the standard emitter interfaces, and your filter plug-in is used.

Chapter 7. Developing an event consumer

An *event consumer* is any application that receives events from the event server. This might be an application that receives asynchronous event notifications, or it might be an application that queries and processes historical event data from the persistent data store. The event consumer receives events in the form of Java objects; it can then use the CommonBaseEvent interface to retrieve event property data, or convert the event to another supported format (such as XML) for forwarding to another application.

An event consumer can receive events in either of two ways:

- It can use the Java Messaging Service (JMS) interface to subscribe to a queue or topic, receiving event notifications asynchronously as JMS messages. This is the most efficient approach for an event consumer that needs to process new and changed events as they arrive at the event server.
- It can use the Event Access interface to query historical events from the persistent data store, retrieving the requested events synchronously. This is useful for startup processing; by querying the data store for historical events, an event consumer can determine current state information before beginning to receive new events through JMS.

In addition to receiving events, an event consumer can also modify events, delete events, and purge old events from the data store.

Using the Java Messaging Service interface

Using the Java Messaging Service (JMS) interface, you can develop event consumers that receive event notifications asynchronously from JMS queues or topics. An event consumer can be implemented as a standard Java class or as a Message-Driven Bean (MDB).

By using the JMS interface, you can implement your event consumer using standard Java tools and programming models, and you can avoid the performance disadvantages of directly querying the event data store. Instead of interacting with the Common Event Infrastructure directly, your event consumer subscribes to JMS destinations (queues and topics) and receives event notifications in the form of JMS messages.

The Common Event Infrastructure organizes events in event groups, which are logical collections of events defined in the Common Event Infrastructure configuration. A particular event consumer typically needs to receive only events from specific event groups.

The configuration profile for each event group associates that event group with one or more JMS destinations through which notifications related to that event group are distributed. The relationships between event groups and JMS destinations are as follows:

- An event group can be associated with multiple queues.
- An event group can be associated with only one topic. (Multiple event consumers can subscribe to the same topic, so publishing the same event group to more than one topic is redundant.)

- A JMS destination (queue or topic) should typically be associated with only one event group.

To receive messages from an event group, a JMS consumer subscribes to the appropriate destination. Each time an event matching the associated event group is created, modified, or deleted, a notification is delivered in the form of a JMS message containing an event notification. The content of the notification depends upon its type:

- For a new or modified event, the notification includes the complete event data, which can be converted into a CommonBaseEvent instance.
- For a deleted event, the notification includes the global instance identifier of the event that has been deleted.

In addition to the standard JMS interfaces, a JMS event consumer interacts with a facility called the Notification Helper. The Notification Helper translates between Common Event Infrastructure entities (events and event groups) and equivalent JMS entities (messages and destinations). The Notification Helper provides the following functions:

- It identifies the JMS topic or queues associated with a specified event group. Your event consumer can then use the appropriate destination to create subscriptions.
- It converts event notifications for new and changed events into instances of CommonBaseEvent.
- It can provide filtering of events at the consumer. Each Notification Helper can be associated with an event selector specifying which events should be returned to consumers. When a consumer uses the Notification Helper to convert an event notification into an event instance, the event instance is returned only if it matches the specified event selector.

  Note: The notification helper uses the Apache JXPath component to process XPath event selectors. If Java 2 security is enabled, you must modify your policy file to include the following entries:

```
permission java.util.PropertyPermission "*", "read";
permission java.io.FilePermission
  "${was.install.root}${/}java${/}jre${/}lib${/}jxpath.properties",
  "read";
```

Developing an event consumer as a message-driven bean (MDB)

A J2EE event consumer is implemented as a message-driven bean, which is associated with a JMS destination and connection factory at deployment time. To receive events, follow these steps:

1. Obtain a notification helper. A JMS event consumer uses a notification helper to identify JMS destinations associated with an event group, to convert received JMS messages into event notifications, and to perform filtering of received events. To obtain a notification helper, use a notification helper factory, which is an instance of NotificationHelperFactory that has been bound into a JNDI namespace. The following code fragment uses a notification helper factory to obtain a notification helper.

```
// Get notification helper factory from JNDI
InitialContext context = new InitialContext();
Object notificationHelperFactoryObject =
  context.lookup("com/ibm/events/NotificationHelperFactory");
NotificationHelperFactory nhFactory = (NotificationHelperFactory)
  PortableRemoteObject.narrow(notificationHelperFactoryObject,
```

```
                                NotificationHelperFactory.class);

// Create notification helper
NotificationHelper notificationHelper =
  nhFactory.getNotificationHelper();
```

2. **Optional:** Specify the event selector. If you want to filter received events, you can use the setEventSelector() method to set an event selector on the notification helper. Your event consumer can then use the notification helper to check received events against the event selector. The following code fragment sets an event selector specifying events with severity greater than 30 (warning).

   ```
   notificationHelper.setEventSelector("CommonBaseEvent[@severity > 30]");
   ```

3. Convert received messages into event notifications.

   In the onMessage() method of your listener, use the notification helper to convert each received JMS message into an array containing an event notification. (If the event does not match the event selector specified on the Notification Helper, the array is empty.) An event notification is an instance of a class implementing the EventNotification interface.

   ```
   public void onMessage(Message msg) {
     EventNotification[] notifications =
                         notificationHelper.getEventNotifications(msg);
     // ...
   ```

4. Check the notification type and retrieve the event data as appropriate. Each event notification has a field representing the notification type (an integer whose value is one of the notification type constants defined by the NotificationHelper interface). Three notification types are currently supported:

Notification type	Description
CREATE_EVENT_NOTIFICATION_TYPE	A new event has been created in the event group associated with the destination. This means either that a new event has been sent, or that an existing event has changed so that it now matches the event group definition. The notification also contains the complete event data.
REMOVE_EVENT_NOTIFICATION_TYPE	An event stored in the event database has been removed from the event group associated with the destination. This means either that an event has been deleted from the event database, or that an existing event has changed so that it no longer matches the event group definition. The notification also contains the global instance identifier of the deleted event.
UPDATE_EVENT_NOTIFICATION_TYPE	An event stored in the event database has been updated in a way that does not change its membership in the event group associated with the destination. The notification also contains the complete event data.

   Use the getNotificationType() method of EventNotification to check the notification type of each received notification. Based on the notification type, you can determine whether your event consumer should process the notification further, and what kind of event data the notification contains:

   - If the notification type is CREATE_EVENT_NOTIFICATION_TYPE or UPDATE_EVENT_NOTIFICATION_TYPE, your consumer can use

EventNotification.getEvent() to attempt to retrieve the new or updated event. This method is valid only for notifications of new or updated events.

- If the notification is REMOVE_EVENT_NOTIFICATION_TYPE, your consumer can use EventNotification.getGlobalInstanceId() to retrieve the global instance identifier of the deleted event. This method is valid only for notifications of deleted events.

```
for (int i = 0; i < notifications.length; i++)
{
  int notifType = notifications[i].getNotificationType();

  if(notifType == NotificationHelper.CREATE_EVENT_NOTIFICATION_TYPE)
  {
    CommonBaseEvent event = notifications[i].getEvent();
    if (event != null) {
      // process the new event
      // ...
    }
  }

  else if(notifType == NotificationHelper.UPDATE_EVENT_NOTIFICATION_TYPE)
  {
    CommonBaseEvent event = notifications[i].getEvent();
    if (event != null) {
      // process the updated event
      // ...
    }
  }

  else if(notifType == NotificationHelper.REMOVE_EVENT_NOTIFICATION_TYPE)
  {
    String eventId = notifications.[i].getGlobalInstanceId();
    // process the event deletion
    // ...
  }
}
```

In its deployment descriptor, a message-driven bean must be associated with a listener port, which specifies a JMS destination and connection factory. You must create a listener port for your event consumer before deploying the MDB, specifying the destination and connection factory associated with the event group from which you want to receive events (these are defined in the event group profile).

Note: Do not use the CommonEventInfrastructure_ListenerPort listener port when deploying your MDB. This listener port is used by the event server and is not intended for use by event consumers.

Developing a non-MDB event consumer

To write an event consumer that is not a message-driven bean, follow these steps:

1. Obtain a notification helper. A JMS event consumer uses a notification helper to identify JMS destinations associated with an event group, to convert received JMS messages into event notifications, and to perform filtering of received events. To obtain a notification helper, use a notification helper factory, which is an instance of NotificationHelperFactory that has been bound into a JNDI namespace. The following code fragment uses a notification helper factory to obtain a notification helper.

```
// Get notification helper factory from JNDI
InitialContext context = new InitialContext();
Object notificationHelperFactoryObject =
  context.lookup("com/ibm/events/NotificationHelperFactory");
```

```
NotificationHelperFactory nhFactory = (NotificationHelperFactory)
  PortableRemoteObject.narrow(notificationHelperFactoryObject,
                              NotificationHelperFactory.class);

// Create notification helper
NotificationHelper notificationHelper =
  nhFactory.getNotificationHelper();
```

2. **Optional:** Specify the event selector. If you want to filter received events, you can use the setEventSelector() method to set an event selector on the notification helper. Your event consumer can then use the notification helper to check received events against the event selector. The following code fragment sets an event selector specifying events with severity greater than 30 (warning).

   ```
   notificationHelper.setEventSelector("CommonBaseEvent[@severity > 30]");
   ```

3. Use the notification helper to find the JMS destination to subscribe to.

   Each event group can be associated with a single JMS topic and any number of JMS queues. You can query the notification helper to find out what destinations are associated with a particular event group. To find the topic associated with an event group, use the getJmsTopic(String) method of NotificationHelper, specifying the name of the event group:

   ```
   MessagePort msgPort = notificationHelper.getJmsTopic("critical_events");
   ```

   To find the queues associated with an event group, use the getJmsQueues(String) method:

   ```
   MessagePort[] msgPorts = notificationHelper.getJmsQueues("critical_events");
   ```

   The returned object is either a single MessagePort object representing a JMS topic or an array of MessagePort objects representing JMS queues. A MessagePort instance is a wrapper object containing the JNDI names of the destination and its connection factory.

4. Connect to the destination. Use the getter methods of MessagePort to retrieve the JNDI names of the destination and connection factory. You can then use the standard JMS interfaces to connect to the destination. The following code fragment subscribes to a JMS topic:

   ```
   String connectionFactoryName = msgPort.getConnectionFactoryJndiName();
   String destinationName = msgPort.getDestinationJndiName();

   // create connection and session
   ConnectionFactory connectionFactory =
        (ConnectionFactory) context.lookup(connectionFactoryName);
   Connection connection = connectionFactory.createConnection();
   Session session = connection.createSesion(false,
                                             Session.CLIENT_ACKNOWLEDGE);

   // Create consumer and register listener
   Topic topic = (Topic) context.lookup(destinationName);
   MessageConsumer consumer = session.createConsumer(topic);
   consumer.setMessageListener(this);
   connection.start();
   ```

5. Convert received messages into event notifications.

   In the onMessage() method of your listener, use the notification helper to convert each received JMS message into an array containing an event notification. (If the event does not match the event selector specified on the Notification Helper, the array is empty.) An event notification is an instance of a class implementing the EventNotification interface.

   ```
   public void onMessage(Message msg) {
     EventNotification[] notifications =
                         notificationHelper.getEventNotifications(msg);
     // ...
   ```

6. Check the notification type and retrieve the event data as appropriate. Each event notification has a field representing the notification type (an integer whose value is one of the notification type constants defined by the NotificationHelper interface). Three notification types are currently supported:

Notification type	Description
CREATE_EVENT_NOTIFICATION_TYPE	A new event has been created in the event group associated with the destination. This means either that a new event has been sent, or that an existing event has changed so that it now matches the event group definition. The notification also contains the complete event data.
REMOVE_EVENT_NOTIFICATION_TYPE	An event stored in the event database has been removed from the event group associated with the destination. This means either that an event has been deleted from the event database, or that an existing event has changed so that it no longer matches the event group definition. The notification also contains the global instance identifier of the deleted event.
UPDATE_EVENT_NOTIFICATION_TYPE	An event stored in the event database has been updated in a way that does not change its membership in the event group associated with the destination. The notification also contains the complete event data.

Use the getNotificationType() method of EventNotification to check the notification type of each received notification. Based on the notification type, you can determine whether your event consumer should process the notification further, and what kind of event data the notification contains:

- If the notification type is CREATE_EVENT_NOTIFICATION_TYPE or UPDATE_EVENT_NOTIFICATION_TYPE, your consumer can use EventNotification.getEvent() to attempt to retrieve the new or updated event. This method is valid only for notifications of new or updated events.
- If the notification is REMOVE_EVENT_NOTIFICATION_TYPE, your consumer can use EventNotification.getGlobalInstanceId() to retrieve the global instance identifier of the deleted event. This method is valid only for notifications of deleted events.

```
for (int i = 0; i < notifications.length; i++)
{
  int notifType = notifications[i].getNotificationType();

  if(notifType == NotificationHelper.CREATE_EVENT_NOTIFICATION_TYPE)
  {
    CommonBaseEvent event = notifications[i].getEvent();
    if (event != null) {
      // process the new event
      // ...
    }
  }

  else if(notifType == NotificationHelper.UPDATE_EVENT_NOTIFICATION_TYPE)
  {
    CommonBaseEvent event = notifications[i].getEvent();
    if (event != null) {
      // process the updated event
```

```
        // ...
      }
    }

    else if(notifType == NotificationHelper.REMOVE_EVENT_NOTIFICATION_TYPE)
    {
      String eventId = notifications.[i].getGlobalInstanceId();
      // process the event deletion
      // ...
    }
  }
```

Querying events from the event server

An event consumer can synchronously retrieve historical events from the persistent data store by querying the event server. You can query events in four ways:

- You can specify a global instance identifier to retrieve a specific single event.
- You can specify an event group and retrieve events associated with that event group. You can optionally refine the query further by specifying an additional event selector, retrieving only events that match both the event group and event selector. For more information about event groups, see Chapter 3, "Configuration," on page 19.
- You can query the existence of events associated with a specified event group without retrieving the events.
- You can specify a known event and an association type, retrieving events that are associated with the known event.

To query the event server, you use the event access interface. This is a J2EE stateless session bean whose interface provides methods for querying the event server. An event consumer uses an instance of the event access bean for all synchronous event queries.

The persistent data store is implemented as a separate component called a *data store plug-in*. The Common Event Infrastructure includes a default data store plug-in, which supports event queries based on a subset of XPath syntax. If you want to use a different data store, you can implement your own data store plug-in.

Note: If WebSphere security is enabled, the application user ID must be mapped to the eventConsumer or eventAdministrator role to query events.

Creating an event access bean

The event access interface is implemented as a stateless session bean using the Enterprise JavaBeans architecture. To query the event server using the event access interface, an event source must first create an instance of the event access session bean. The event access bean can be either local or remote.

To create an instance of the event access session bean, use the appropriate home interface (either EventAccessHome or EventAccessLocalHome).

```
// use home interface to create remote event access bean
InitialContext context = new InitialContext();
Object eventAccessHomeObj = context.lookup("ejb/com/ibm/events/access/EventAccess");
EventAccessHome eventAccessHome = (EventAccessHome)
  PortableRemoteObject.narrow(eventAccessHomeObj,
                              EventAccessHome.class);
eventAccess = (EventAccess) eventAccessHome.create();
```

Querying events by global instance identifier

The Common Base Event specification defines an event property called globalInstanceId which can be used as a primary key for event identification. The content of this property is a globally unique identifier, generated either by the application or by the emitter. Although the Common Base Event specification defines the globalInstanceId property as optional, the event emitter automatically assigns an identifier to any event that does not already have one. For more information, see "Sending events" on page 56.

You can retrieve a specific single event from the event server by querying with the globalInstanceId property of the event you want to retrieve. This can be useful for testing purposes (to confirm that events are being stored in the event database), or to retrieve an event associated with one that was received previously.

To query an event by global instance identifier, use the queryEventByGlobalInstanceId() method of the event access bean.

1. If necessary, create an event access bean.
2. Call the queryEventByGlobalInstanceId(String) method of the EventAccess bean, specifying the global instance identifier of the event you want to retrieve.

   ```
   CommonBaseEvent event = eventAccess.queryEventByGlobalInstanceId(eventId);
   ```

   The returned object is the event with the specified global instance identifier. If there is no matching event in the persistent data store, the returned object is null.

Querying events by event group

An event can be associated with one or more *event groups*. An event group is a logical grouping of events that match a particular event selector; event groups are defined in the event infrastructure configuration. For more information about event groups, see Chapter 3, "Configuration," on page 19.

You can use the event access interface to retrieve events that belong to a specified event group. You can further restrict the query results by specifying an additional event selector.

The event access interface provides two methods for querying by event group; the first method returns a limited number of events, while the other returns all events belonging to the specified event group (which might be a large number of events).

Querying a limited number of events from an event group

To query a limited number of events from an event group, use the queryEventsByEventGroup(String, String, boolean, int) method of the EventAccess bean.

1. If necessary, create an event access bean.
2. Call the EventAccess.queryEventsByEventGroup(String, String, boolean, int) method.

   ```
   CommonBaseEvent[] events = eventAccess.queryEventsByEventGroup(eventGroup,
                                                                  eventSelector,
                                                                  ascendingOrder,
                                                                  maxEvents);
   ```

   The parameters of this method are as follows:

eventGroup
A string containing the name of the event group you want to query events from. This must be the name of an existing event group defined in the event infrastructure configuration.

eventSelector
A string containing an optional event selector that further refines the query. The query returns only events that match both the specified event group and the additional event selector. An event selector is specified in the form of an XPath expression (for more information, see "Writing event selectors" on page 78). If you do not want to specify an additional event selector, this parameter can be null.

ascendingOrder
A boolean value specifying whether the returned events are sorted in ascending or descending order according to the value of the creationTime property. If this parameter is *true*, the events are sorted in ascending (chronological) order; if it is *false*, they are sorted in descending (reverse chronological) order.

maxEvents
An integer specifying the maximum number of events you want returned.

The returned object is an array containing the events from the specified event group.

Note: If the number of matching events exceeds the query threshold defined in the data store profile, a QueryThresholdExceededException exception is thrown. The default query threshold is 100 000.

The following code fragment returns all events that belong to an event group called *critical_hosts* and whose severity is greater than 30 (warning), but specifies that no more than 5000 matching events should be returned:

```
CommonBaseEvent[] events =
        eventAccess.queryByEventGroup("critical_hosts",
                                      "CommonBaseEvent[@severity > 30]",
                                      true,
                                      5000);
```

Querying all events from an event group

To query all events from an event group, use the queryEventsByEventGroup(String, String, boolean) method of the EventAccess bean.

1. If necessary, create an event access bean.
2. Call the EventAccess.queryEventsByEventGroup(String, String, boolean) method.

   ```
   CommonBaseEvent[] events = eventAccess.queryEventsByEventGroup(eventGroup,
                                                                  eventSelector,
                                                                  ascendingOrder);
   ```

   The parameters of this method are as follows:

   eventGroup
   A string containing the name of the event group you want to query events from. This must be the name of an existing event group defined in the event infrastructure configuration.

   eventSelector
   A string containing an optional event selector that further refines the query.

The query returns only events that match both the specified event group and the additional event selector. An event selector is specified in the form of an XPath expression (for more information, see "Writing event selectors" on page 78). If you do not want to specify an additional event selector, this parameter can be null.

ascendingOrder
A boolean value specifying whether the returned events are sorted in ascending or descending order according to the value of the creationTime property. If this parameter is *true*, the events are sorted in ascending (chronological) order; if it is *false*, they are sorted in descending (reverse chronological) order.

The returned object is an array containing the events from the specified event group.

Note: If the number of matching events exceeds the query threshold defined in the data store profile, a QueryThresholdExceededException exception is thrown. The default query threshold is 100 000.

The following code fragment returns all events that belong to an event group called *critical_hosts* and whose severity is greater than 30 (warning):

```
CommonBaseEvent[] events =
        eventAccess.queryByEventGroup("critical_hosts",
                                      "CommonBaseEvent[@severity > 30]",
                                      true);
```

Querying the existence of events in an event group

In some situations, you might want to find out whether any events exist in a particular event group without actually retrieving the events. To do this, use the eventExists() method of the event access bean.

1. If necessary, create an event access bean.
2. Call the eventExists(String, String) method of the EventAccess bean.

   ```
   boolean hasEvents = eventAccess.eventExists(eventGroup,
                                               eventSelector);
   ```

   The parameters of this method are as follows:

   eventGroup
   A string containing the name of the event group you want to check for events. This must be the name of an existing event group defined in the event infrastructure configuration.

   eventSelector
   A string containing an optional event selector that further refines the query. The query only checks for events that match both the specified event group and the additional event selector. An event selector is specified in the form of an XPath expression (for more information, see "Writing event selectors" on page 78). If you do not want to specify an additional event selector, this parameter can be null.

The returned boolean object equals `true` if any events exist that match the specified event group and event selector, `false` if none exist.

The following code fragment checks for the existence of any events in an event group called *critical_hosts* and retrieves any that exist.

```
if (eventAccess.eventExists("critical_hosts",null)) {
   CommonBaseEvent[] events =
     eventAccess.queryByEventGroup("critical_hosts",
                                   null,
                                   true);
   }
```

Querying events by association type

The Common Base Event specification defines properties that establish relationships between events. The associatedEvents property is a complex element containing one or more subelements of the AssociatedEvent type, each representing an associated event. Each AssociatedEvent element, in turn, contains subelements identifying the type of association and the application that established the association. Examples of association types might include `CausedBy` or `Correlated`.

By specifying the global instance identifier of a known event and a type of association, you can retrieve events that satisfy the specified association. To query events by association type, use the EventAccess.queryEventsByAssocation(String, String) method.

1. If necessary, create an event access bean.
2. Call the EventAccess.queryEventsByAssociation(String, String) method.

   ```
   CommonBaseEvent[] events = eventAccess.queryEventsByAssocation(associationType,
                                                                  eventId);
   ```

   The parameters of this method are as follows:

   associationType
   : The type of association. This should be the name of an association type specified by the associationEngineInfo property.

   eventId
   : The global instance identifier of a known event.

   The returned object is an array containing the events that satisfy the specified type of association with the known event. Only events that are still in the event database at the time of the query are returned (an associated event might be purged from the database).

The following code fragment returns all events from the event database that have a `CausedBy` association with a known event:

```
String eventId = causeEvent.getGlobalInstanceId();
CommonBaseEvent[] resultEvents = eventAccess.queryEventsByAssociation("CausedBy",
                                                                      eventId);
```

Deleting events from the data store

An event consumer or administrative tool can delete events from the data store using the event access interface. You can delete all events from the data store, or you can limit the deleted events by specifying event groups, event selectors, or both.

Note: If WebSphere security is enabled, the application user ID must be mapped to the eventAdministrator role to delete events.

To delete events from the data store, use the purgeEvents() method of the event access bean.

```
int purged = eventAccess.purgeEvents(eventGroup,
                                     eventSelector,
                                     transactionSize);
```

The parameters are as follows:

eventGroup
A string containing the name of the event group that includes the events you want to delete. This must be the name of an existing event group defined in the event infrastructure configuration. If you do not want to specify an event group, this parameter can be null.

eventSelector
A string containing an optional event selector that identifies the events to delete. An event selector is specified in the form of an XPath expression (for more information, see "Writing event selectors" on page 78). If you do not want to specify an event selector, this parameter can be null.

transactionSize
A nonzero integer specifying the number of events you want deleted in a single database transaction. In most cases, you can use the constant DEFAULT_PURGE_TRANSACTION_SIZE, which is defined by the EventAccess interface.

The purgeEvents() method deletes all of the events that match all of the criteria you specify. If the *eventGroup* and *eventSelector* parameters are both null, all events in the data store are deleted. Events that arrive after the delete operation starts are not purged. The returned value is an integer specifying how many events were deleted.

Note: If the value of the *transactionSize* parameter exceeds the maximum purge transaction size defined in the data store profile, a PurgeThresholdExceededException exception is thrown and no events are deleted. The default maximum purge transaction size is 100 000.

Updating events in the event database

An event consumer can update the contents of events that are stored in the event database. To update an event, you must first create an event update request object and then submit the request to the Event Access bean.

Updating an event causes an event update notification to be sent to event consumers subscribing to the affected event groups. The update notification indicates that an existing event has been changed and includes the complete updated event data. Depending on the event selectors specified in the event group definitions, a change to an event might cause it to change event groups.

Note: If WebSphere security is enabled, the application user ID must be mapped to the eventUpdater or eventAdministrator role to update events.

Creating an event change request

To update an existing event, you must first create an event change request.

An event change request is a Java object specifying the update you want to request. Five types of event update requests are supported, each represented by a class implementing the EventChangeRequest interface. These classes provide methods you can use to specify what event data you want to change.

You can submit multiple change requests affecting multiple events at the same time.

Creating an UpdateEventAttribute request

Use the UpdateEventAttribute class to update an attribute value of an existing event.

To update an event attribute:

1. Create an instance of UpdateEventAttribute:

   ```
   UpdateEventAttribute request = new UpdateEventAttribute(eventId);
   ```

   The parameter *eventId* is a string specifying the global instance identifier of the event you want to update. The specified event must exist in the event database.
2. Use the appropriate setter method to set the new value of the attribute you want to change. The UpdateEventAttribute class provides setter methods for all of the attributes that can be updated. For example, the following statement specifies that the priority of the event should be changed to 70:

   ```
   request.setPriority(70);
   ```

   Refer to the Javadoc documentation for the UpdateEventAttribute class for complete information on the available setter methods.

   Note: As defined by the Common Base Event specification, the severity attribute is optional but immutable once set. Therefore, you can use an UpdateEventAttribute request to set the severity of an event only if the severity attribute is currently unset. If you attempt to change the severity of an event that already has a severity attribute value, SeverityAlreadySetException is thrown.

Creating an UpdateEventAssociation request

Use the UpdateEventAssociation class to add associated events to an existing event.

To add an association, create an instance of UpdateEventAssociation, specifying the association engine and an array of associated events:

```
UpdateEventAssociation request = new UpdateEventAssociation(eventId,
                                                            associationEngine,
                                                            resolvedEvents);
```

The parameters of this constructor are as follows:

eventId
: A string specifying the global instance identifier of the event to which you want to add associations. The specified event must exist in the event database.

associationEngine
: An instance of a class implementing the org.eclipse.hyades.logging.events.cbe.AssociationEngine interface. This class represents the AssociationEngine property of the event; it specifies the application creating the association and the type of association. For more information, refer to the Javadoc documentation for the AssociationEngine interface.

resolvedEvents
: An array of events (instances of org.eclipse.hyades.logging.events.CommonBaseEvent). This array specifies the

events that are to be associated with the existing event as specified by the AssociationEngine. These events are added to any already associated with the updated event.

Creating an UpdateExtendedDataElementValues request

Use the UpdateExtendedDataElementValues class to update an extended data element of an event.

You can either modify an existing extended data element or add a new extended data element.

To update an extended data element, create an instance of UpdateExtendedDataElementValues, specifying the updated extended data element.

```
UpdateExtendedDataElementValues request =
     new UpdateExtendedDataElementValues(eventId,
                                         parentLocationPath,
                                         extendedDataElement);
```

The parameters of this constructor are as follows:

eventId
A string specifying the global instance identifier of the event you want to update. The specified event must exist in the event database.

parentLocationPath
An XPath expression specifying the location of the parent element of the extended data element in the XML containment hierarchy.

extendedDataElement
An instance of org.eclipse.hyades.logging.events.cbe.ExtendedDataElement representing the new extended data element. If an extended data element of the same name already exists at the specified location in the containment hierarchy, it is replaced with the new element. If no extended data element currently exists with the specified name, the new element is added.

Note: The new extended data element cannot contain children.

Creating an AddExtendedDataElement request

Use the AddExtendedDataElement class to add a new extended data element to an event.

Note: You cannot add an extended data element that contains children.

To add an extended data element, create an instance of AddExtendedDataElement, specifying the new extended data element.

```
AddExtendedDataElement request =
     new AddExtendedDataElement(eventId,
                                parentLocationPath,
                                extendedDataElement);
```

The parameters of this constructor are as follows:

eventId
A string specifying the global instance identifier of the event to which you want to add the extended data element. The specified event must exist in the event database.

parentLocationPath
An XPath expression specifying the location of the parent element of the new extended data element in the XML containment hierarchy.

extendedDataElement
An instance of org.eclipse.hyades.logging.events.cbe.ExtendedDataElement representing the new extended data element. The new element is added even if another extended data element of the same name already exists at the specified location in the containment hierarchy.

Creating an AddContextDataElement request

Use the AddContextDataElement class to add a new context data element to an event.

To add a context data element, create an instance of AddContextDataElement, specifying the new context data element.

```
AddContextDataElement request =
     new AddContextDataElement(eventId,
                               contextDataElement);
```

The parameters of this constructor are as follows:

eventId
A string specifying the global instance identifier of the event to which you want to add the context data element. The specified event must exist in the event database.

extendedDataElement
An instance of org.eclipse.hyades.logging.events.cbe.ContextDataElement representing the new extended data element.

Submitting an event change request

After you create one or more event change requests, use the updateEvents() method to submit the requests to the event server.

You can submit multiple requests at the same time; these requests can be of different types and can affect different events. To submit event change requests to the event server:

1. If necessary, create an event access bean.
2. Add the new event change request objects to an array of type EventChangeRequest[].
3. Call the updateEvents(EventChangeRequest[]) method of the Event Access bean, specifying the array containing the change requests you want to submit:

   ```
   eventAccess.updateEvents(changeRequests);
   ```

All of the requests contained in the specified array are processed by the event server. If any of the requests fail, then the current transaction is rolled back and none of the updates are applied.

After each update request is processed by the event server, an event notification of type UPDATE_EVENT_NOTIFICATION_TYPE is sent to consumers subscribing to event notifications for any affected event groups.

Writing event selectors

An event selector is a regular expression that defines a set of events based on their property data (attributes or subelements). For example, an event selector might specify all events from a particular host whose severity is greater than 30 (warning). Event selectors are used to define event groups, specify filter criteria, and query the event server.

Because the Common Base Event specification is based on XML, event selectors are written using a subset of XPath syntax. The specific syntax you can use for an event selector depends on how the event selector is to be used, as summarized by the following table.

Event selector purpose	Syntax
Event group definition	Limited to XPath subset supported by default data store plug-in
Event query and purge through event access interface	Limited to XPath subset supported by default data store plug-in
Emitter filter configuration	Any valid XPath
Subscription through Notification Helper interface	Any valid XPath

Note: The default data store plug-in uses a subset of XPath syntax. However, if you are using a different data store plug-in, it might support a different subset of XPath. The event selectors you write for event group definitions and for the event access interface must use the syntax that is supported by your data store plug-in.

Writing XPath event selectors

XPath is a standard language used to identify parts of an XML document; for more information, see the XPath specification at http://www.w3.org/TR/xpath.

A simple XPath event selector that specifies an attribute value takes the following form:

```
CommonBaseEvent[@attribute = value]
```

The *value* can be either a numeric value or a string enclosed in single or double quotation marks.

You can also specify an attribute of a subelement:

```
CommonBaseEvent[/subelement/@attribute = value]
```

When using XPath operators, keep the following general rules in mind:

- When used to compare XML dateTime values, the comparison operators perform logical comparisons that recognize time zone differences.
- Arithmetic comparison operators should not be used with attributes that might be unset, because an unset attribute value is evaluated as zero. For example, the comparison `@severity < 30` would evaluate as true if the severity attribute is unset. (The creationTime attribute is required and is therefore always set.)
- Logical operators and function names must be specified using all lowercase letters (for example, `and` rather than `AND`).

- Operators must be separated with white space from the surrounding attribute names and values (`@severity = 30` rather than `@severity=30`).
- Parentheses can be used to change operator precedence.

The following examples are valid XPath event selectors.

Event selector	Description
`CommonBaseEvent[@extensionName = 'ApplicationStarted']`	All events with the *extensionName* attribute `ApplicationStarted`
`CommonBaseEvent[sourceComponentId/ @location = "server1"]`	All events containing a sourceComponentId element with the *location* attribute `server1`
`CommonBaseEvent[@severity]`	All events with a *severity* attribute, regardless of its value
`CommonBaseEvent[@creationTime < '2003-12-10T12:00:00-05:00' and @severity = 30]`	All events created before noon EST on 10 December 2003 and with severity of 30 (warning)
`CommonBaseEvent[contains(@msg, 'disk full')]`	All events with the phrase `disk full` occurring within the msg attribute
`CommonBaseEvent[(@severity = 30 or @severity = 50) and @priority = 100]`	All events whose *severity* atrribute is equal to 30 or 50, and whose *priority* is equal to 100

Writing event selectors for the default data store plug-in

If your event selector might be used to define an event group or to query the persistent data store, it is subject to the restrictions of the default data store plug-in. These restrictions are as follows:

- An event property can be specified only on the left side of an operator or XPath function. The value on the right side of an operator must be a literal value. The following example is not a valid event selector:

  ```
  CommonBaseEvent[30 = @priority and
                  contains('this message', @msg)]
  ```

  This could be rewritten as the following valid event selector:

  ```
  CommonBaseEvent[@priority = 30 and
                  contains(@msg, 'this message')]
  ```

- Only the following XPath functions are supported:
  - contains
  - starts-with
  - false
  - true
  - not
- The union operator (`|`) is not supported.
- An event selector must take the following form:

  `CommonBaseEvent[`*`predicate_expression`*`]`

  Only a single predicate expression can be associated with the CommonBaseEvent element. Stacked predicates are not supported (for example, `CommonBaseEvent[@extensionName = "server_down"][@severity = 10]`).
- A predicate can be only be associated with the last step of a location path. The following example is not a valid event selector:

```
CommonBaseEvent[contextDataElement[@contextValue = "myContextValue"]
                /@contextId = "myContextId"]
```

Instead, this could be rewritten as follows:

```
CommonBaseEvent[contextDataElement[@contextValue = "myContextValue"
                and @contextId = "myContextId"]]
```

- If an event selector refers to properties of extended data elements that are at different levels of the XML containment hierarchy, these elements must be grouped together by level. The following example is not a valid event selector, because the references to the *type* and *value* attributes (both top-level) of *extendedDataElements* are separated:

  ```
  CommonBaseEvent[extendedDataElements[@type = 'int' and
                                       children/@type = 'intArray' and
                                       children/@name = 'myName' and
                                       @value = 10]]
  ```

  Instead, this could be rewritten as follows, grouping the top-level and second-level attributes together:

  ```
  CommonBaseEvent[extendedDataElements[@type = 'int' and
                                       @value = 10 and
                                       children/@type = 'intArray' and
                                       children/@type = 'myName']]
  ```

- Node indexes are not supported (for example, `CommonBaseEvent[extendedDataElements[1]]`).
- Wildcard characters are not supported (for example, `CommonBaseEvent[extendedDataElements/*/children/values = "text"]`).
- When referring to the *values* property of an extended data element, you must specify not only the value but also the type of the property:

  ```
  CommonBaseEvent[extendedDataElements[values = "myVal"
    and @type = "string"]]
  ```

  You can specify the type for multiple comparisons within a compound expression by grouping them with parentheses:

  ```
  CommonBaseEvent[extendedDataElements[(values = "myVal" or
                                        values = "yourVal") and
                                        @type = "string"]]
  ```

  In this example, the *type* expression applies to both parts of the compound expression in parentheses. You cannot override this by specifying a different *type* expression inside the parentheses.

  You can also group together multiple related types by using the `starts-with` or `contains` functions. For example, the following expression would match a property with either the string or stringArray type:

  ```
  CommonBaseEvent[extendedDataElements[values = "myVal" and
                                       starts-with(@type, 'string')]]
  ```

Implementing a data store plug-in

To use your own data store for persistent storage of events, you can implement a custom data store plug-in by following these steps:

1. Develop your data store plug-in as an enterprise bean with the provided local interface. Your data store plug-in must implement the interface com.ibm.events.datastore.DataStoreLocal. The DataStoreLocal interface defines the following methods (refer to the Javadoc API documentation for complete information):

createEvent(CommonBaseEvent)
Stores a new event in the data store.

createEvents(EventCreationRequest[])
Stores multiple new events in the data store.

eventExists(String)
Returns a boolean indicating whether any events currently in the data store match the specified event selector.

purgeEvents(String[])
Deletes events matching the specified global instance identifiers.

purgeEvents(String,int)
Deletes events matching the specified event selector, with the specified maximum transaction size.

queryEventByGlobalInstanceId(String)
Returns the event whose global instance identifier matches the specified value (or null if no matching event is found).

queryEvents(String, boolean)
Returns an array of events that match the specified event selector.

queryEvents(String, boolean, int)
Returns an array of events that match the specified event selector, limiting the array to the specified size.

queryEventsByAssociation(String, String)
Returns an array of events that satisfy the specified relationship to a known event.

queryGlobalInstanceIds(String, int)
Returns an array of global instance identifiers for events that match a specified event selector, limiting the array to the specified size.

updateEvents(EventUpdateRequest[])
Updates events stored in the event database.

getMetaData()
Returns metadata describing the data store plug-in, including the version of the Common Base Event specification it supports. The metadata must be represented as an instance of DataStoreMetaData.

A data store plug-in must also satisfy the following requirements:

- It must use XPath syntax, or a subset of XPath syntax, for specifying event selectors.
- It must store all of the data associated with each received event.
- Its query methods must return event objects that are identical to those originally stored.

2. Deploy your data store plug-in in WebSphere Application Server. See the WebSphere Application Server documentation for more information about how to deploy an application.
3. In the WebSphere Application Server administrative console, modify the default event server profile. In the **Data Store JNDI Name** field, specify the JNDI name of your data store plug-in. For more information about the event server profile, see the online help for the administrative console.

When you start the Common Event Infrastructure server, the event server uses the specified JNDI name to access the local home interface of the data store enterprise bean. It then uses the local home interface to create an instance of the data store plug-in bean.

Chapter 8. Developing an event catalog application

The *event catalog* is a repository of event metadata. This metadata consists of event definitions, which describe classes of events and their allowed content. (This is distinct from the event instance metadata you can access using the Eclipse Modeling Framework interfaces described in "Accessing event instance metadata" on page 52.) Applications can use the event catalog to manage their enterprise-specific event definitions, but must implement their own validation logic to ensure that events conform to these definitions.

Events defined according to the Common Base Event specification can be categorized into event classes based upon extension name (the value of the *extensionName* attribute). Using the event catalog, you can define the permitted content of a particular class of event by specifying what extended data elements events of that class can contain, as well as the permitted values for other Common Base Event properties. An event definition defines constraints on event content above and beyond those of the Common Base Event specification.

Event definitions are defined hierarchically and inherit the definitions of their parents. A single root event definition, `event`, defines the basic requirements of any event that conforms to the Common Base Event specification. All other event definitions inherit from this root definition. By default, this root event definition is automatically installed in the Event Catalog, along with event definitions for Event Catalog notification events (for more information, see "Change notification" on page 88).

Note: Currently, event definitions do not support all of the forms of constraints required to fully describe the Common Base Event specification (for example, the requirement that the *globalInstanceId* property must begin with an alphabetic character). Therefore, it is possible that an event might conform to the `event` event definition and still not pass validation by the event emitter.

By using the Event Catalog interfaces, you can create, delete, and query event definitions. (Once created, an event definition cannot be modified.) You can also list existing event definitions in a readable format, as well as importing and exporting event definitions in XML format.

Event definitions

An *event definition* contains several kinds of information:

Name The name of the event definition, which is the same as the extension name of the events described by the definition. All events with a particular extension name share the same event definition.

Parent The name of the parent event definition. Any event definition (with the exception of the root definition `event`) has a parent event definition from which it inherits property descriptions and extended data element descriptions (although some aspects of the inherited data can be overriden). The parent can be any valid event definition that exists in the event catalog.

Property descriptions
Descriptions of the permitted Common Base Event properties for the event definition. A property description can describe any property defined in the Common Base Event specification as a simple type, including properties of complex subelements.

Extended data element descriptions
Descriptions of the permitted extended data elements for the event definition. An extended data element description defines the name and type of the extended data element; it can also define default values, how many of the extended data element are allowed, and descriptions of child extended data elements.

Represented as an XML document, an event definition takes the following general form:

```
<eventDefinition name="eventDefinitionName"
                 parent="parentEventDefinitionName">
     <property name="propertyName" ... />
     <extendedDataElement name="extendedDataElementName"
                          type="type" ... />
</eventDefinition>
```

Property descriptions

A property description describes a property that an event can contain. This can be any property defined by the Common Base Event specification as a simple type. A property description cannot describe a complex property such as *msgDataElement*, but it can describe a simple property that is a child of a complex property. An event definition can contain any number of property descriptions (including none).

A property description includes the following fields:

name The name of the property. This must be the name of an attribute of the CommonBaseEvent element, or an attribute of a complex subelement of CommonBaseEvent. Some examples are `severity`, `priority`, and `globalInstanceId`.

path An XPath location path specifying the path to the property, if the property is not an attribute of CommonBaseEvent. The path identifies the parent property of the property being described. These are examples:

- To describe a property of CommonBaseEvent such as *severity*, do not specify a path. A null path specifies a top-level property.
- To describe a property of *msgDataElement*, which is a complex property of CommonBaseEvent, you specify the path `msgDataElement`.
- To describe a property of *msgHelp*, which is itself a complex property of *msgDataElement*, specify the path `msgDataElement/msgHelp`.

The path can also describe a specific instance of a repeated property. For example, if an event definition describes several *contextDataElements* properties, you might specify one called *businessContext*, you would use the path `contextDataElements[@name='businessContext']`.

defaultValue
The default value of the property. The default value represents the value that should be used during content completion for an event that is missing a required property. (Therefore, it is meaningful for a property description to be required and to define a default value.) This field is optional.

required
A boolean value specifying whether the property is required or optional. If this field is equal to `true`, the property is required. This field is optional; if it is not specified, the property is assumed to be optional.

permittedValue
A permitted value for the property. If an event definition allows only certain values for a property, each one is represented by a *permittedValue* field in the property description. A property description can include any number of permitted values. This field is optional and must not be specified if the *minValue* or *maxValue* fields are specified.

minValue
maxValue
The minimum and maximum permitted values for the property. If an event definition allows a range of values for a property, these fields defines the lower and upper bounds of that range. If you specify only *minValue*, the permitted range has no upper bound; similarly, if you specify only *maxValue*, the permitted range has no lower bound. These fields are optional and must not be specified if *permittedValue* fields are specified.

Extended data element descriptions

An extended data element description describes an extended data element that an event of a particular event class can contain. An event definition can contain any number of extended data element descriptions (including none).

An extended data element description includes the following fields:

name The name of the extended data element. This defines the value of the *name* attribute of the element.

type The data type of the extended data element. This defines the value of the *type* attribute of the element. This must be one of the following supported data types:

- noValue
- byte
- short
- int
- long
- float
- double
- string
- dateTime
- boolean
- byteArray
- shortArray
- intArray
- longArray
- floatArray
- doubleArray
- stringArray
- dateTimeArray
- booleanArray

- hexBinary

defaultValue
The default value of the extended data element, or multiple default values if the type is an array. The default value represents the value that should be used during content completion for an event that is missing a required extended data element. This field is optional.

minOccurs
The minimum number of instances of the extended data element that must appear. This field is optional; the default value is 1.

maxOccurs
The maximum number of instances of the extended data element that can appear. This field is optional; the default value is 1.

Note: The current Common Base Event specification allows only one extended data element with a given name at each level of the event containment hierarchy, but this restriction will not be included in future versions of the specification and is not enforced by the Common Event Infrastructure.

Inheritance

By default, an event definition inherits the property descriptions and extended data element descriptions of its parent. However, a child event definition can override these inherited descriptions, subject to certain restrictions. When you add an event definition to the event catalog, the catalog verifies that the new event definition does not violate the rules governing inheritance; if it does, an InheritanceNotValidException exception is thrown. Similarly, if you replace an existing event definition that has descendants, the event catalog verifies the validity of the existing inheritance relationships and throws an InheritanceNotValidException exception if any of them are no longer valid. In either case, the new event definition is not added to the catalog unless all inheritance relationships are valid.

An event definition can exist in either of two forms, *unresolved* and *resolved*:

- An unresolved event definition includes only those property definitions and extended data element descriptions that are defined within the event definition itself.
- A resolved event definition includes the data in the unresolved event definition in addition to the property definitions and extended data element descriptions it inherits.

Overriding inherited property descriptions

A child event definition inherits each property description from its parent without change unless it already has a locally defined property description of the same name and path (note that case is significant). If the child has a property description of the same name and path, the fields of the child description can override the fields of the parent description as follows:

Default value
The child can override the default value specified by the parent property description. If the child does not specify a default value, it inherits the value from the parent.

Required or optional
The child always overrides the parent. However, if the parent defines a

property as required, the child must also specify that it is required. An inherited required property cannot be redefined as optional.

Permitted values or minimum and maximum values
If the parent defines permitted values or minimum and maximum values, the child can override these by specifying either permitted values or minimum and maximum values. Note that an event definition can contain only permitted values or minimum and maximum values, not both:

- If the parent defines minimum and maximum values, but the child defines permitted values, the minimum and maximum values defined by the parent are ignored.
- If the parent defines permitted values, but the child defines minimum and maximum values, the permitted values defined by the parent are ignored.
- If the parent defines only a maximum value, but the child defines only a minimum value, the child inherits the maximum value defined by the parent.
- If the child does not specify permitted values or minimum and maximum values, the values specified by the parent are inherited.

Overriding inherited extended data element descriptions

A child event definition inherits each extended data element description from its parent without change unless it already has a locally defined extended data element description of the same name. If the child does have an extended data element description of the same name, the fields of the child description can override the fields of the parent description as follows:

Type The child must specify the same type as the parent.

Minimum occurrence
The child always overrides the parent.

Maximum occurrence
The child always overrides the parent.

Default values
The child can override the default values specified by the parent extended data element description. If the child does not specify default values, it inherits the values from the parent.

Default hexadecimal value
The child can override the default hexadecimal value specified by the parent extended data element description. If the child does not specify a default hexadecimal value, it inherits the value from the parent.

Nested extended data element description
The child can override a nested extended data element description by defining a nested description of the same name. If the child overrides an inherited nested description, the same rules apply to overriding the individual fields. If the child does not specify a nested extended data element description of the same name, it inherits the nested description from the parent.

Change notification

Each time an event definition is added, removed, or replaced, the event catalog sends an event to the event server indicating that this has happened. An event consumer can subscribe to these events to receive notification of changes in the event catalog. By default, the event catalog uses the default emitter factory to obtain an emitter for sending these events; however, this can be changed in the Event Catalog configuration.

The event catalog can send three classes of notification events, using the following extension names:

- cei_event_definition_added
- cei_event_definition_replaced
- cei_event_definition_removed

These three event classes inherit property descriptions from a common parent class, cei_event_definition. Event definitions for all four event classes are automatically loaded into the event catalog during installation, along with the default root event definition.

Note: When an event definition is removed from the event catalog, any children or other descendants of that event definition are also removed. The event catalog sends a separate change notification event for each event definition that is removed.

Each change notification event contains the following properties:

Property	Value
version	`1.0.1`
globalInstanceId	A globally unique identifier for the event
creationTime	Current date and time when the event is generated
severity	10 (information)
priority	10 (low)
sourceComponentId	Identification of the Event Catalog component and event server host machine
situation	Situation data, including one of the following values for situation category: • CreateSituation (event definition added) • ConfigureSituation (event definition replaced) • DestroySituation (event definition removed)
extensionName	One of the following values: • cei_event_definition_added • cei_event_definition_replaced • cei_event_definition_removed
extendedDataElements	A single extended data element with one attribute, *eventDefinitionName*. This attribute is a string specifying the name of the event definition that has been added, replaced, or removed.

Creating an event definition

An event definition is an instance of the class EventDefinition. To create an event definition, first create a new instance of this class and then populate it with property descriptions and extended data element descriptions. After you have created an event definition, you can add it to the event catalog; for more information, see "Adding an event definition to the event catalog" on page 92.

To create a new, empty event definition, create an instance of EventDefinition:

```
EventDefinition definition = new EventDefinition(name, parent);
```

The parameters of this constructor are as follows:

name
The name of the event definition. This is the value of the *extensionName* attribute for the events you are describing.

parent
The name of the parent event definition. If you do not want your event definition to inherit any property descriptions or extended data element descriptions other than those required by the Common Base Event specification, this parameter should be `event`. If this parameter is null, the new event definition is defined as a root event definition; a root event definition can only be added to the catalog if it is empty, or if you intend to replace the current root event definition.

The returned object is a new unresolved event definition containing no property descriptions or extended data element descriptions.

The following code fragment creates a new event definition called `insurance_claim_start_auto`, which is a child of the event definition `insurance_claim_start`:

```
EventDefinition definition = new EventDefinition("insurance_claim_start_auto",
                                                 "insurance_claim_start");
```

You can now populate the event definition with property descriptions and extended data element descriptions.

Adding property descriptions to an event definition

A property description is an instance of the class PropertyDescription. To add a property description to an event definition, you must first create a new property description and then set the values of its fields. You can then add the property description to the event definition.

1. To create a new property description, create an instance of PropertyDescription, specifying the name and path of the property.

   ```
   PropertyDescription propDesc = new PropertyDescription(name, path);
   ```

   The parameters of this constructor are as follows:

   name
   The name of the property. This must be the name of a simple property either of the CommonBaseEvent element or one of its children.

path
An XPath location path specifying the path to the property. For a top-level property of CommonBaseEvent (such as *severity* or *priority*), *path* should be null.

The returned object is a new PropertyDescription object.

2. Populate the fields of the property description. The PropertyDescription class provides a setter method for each of the fields in a property description. Refer to the Javadoc API documentation for complete information about these methods. For example, to specify that a property is required, you would set the *required* property to `true` using the setRequired(boolean) method:

   ```
   propDesc.setRequired(true);
   ```

3. Add the property description to the event definition using the EventDefinition.addPropertyDescription() method.

   ```
   definition.addPropertyDescription(propDesc);
   ```

   If the event definition already includes another property description with the same name and path, a DescriptionExistsException exception is thrown.

The following code fragment creates a new property description, populates it with data, and adds it to an event definition.

```
PropertyDescription propDesc = new PropertyDescription("severity",null);
propDesc.setRequired(true);
propDesc.setMinValue('30');

// definition is a valid event definition
definition.addPropertyDescription(propDesc);
```

Adding extended data element descriptions to an event definition

An extended data element description is an instance of the ExtendedDataElementDescription class. To add an extended data element description to an event definition, you must first create a new extended data element description and then set the values of its fields. You can also add nested (child) extended data element descriptions, which describe nested extended data elements. You can then add the extended data element description to the event definition.

1. To create a new extended data element description, create an instance of ExtendedDataElementDescription, specifying the name and type of the extended data element.

   ```
   ExtendedDataElementDescription edeDesc =
        new ExtendedDataElementDescription(name, type);
   ```

   The parameters of this constructor are as follows:

   name
   The name of the extended data element. This must be the value of the *name* property of the extended data element you want to describe.

   type
   The data type of the extended data element. This must be one of the following integer constants defined by the org.eclipse.hyades.logging.events.cbe.ExtendedDataElement class:
   - TYPE_BOOLEAN_ARRAY_VALUE
   - TYPE_BOOLEAN_VALUE

- TYPE_BYTE_ARRAY_VALUE
- TYPE_BYTE_ARRAY
- TYPE_DATE_TIME_ARRAY_VALUE
- TYPE_DATE_TIME_VALUE
- TYPE_DOUBLE_ARRAY_VALUE
- TYPE_DOUBLE_VALUE
- TYPE_FLOAT_ARRAY_VALUE
- TYPE_FLOAT_VALUE
- TYPE_HEX_BINARY_VALUE
- TYPE_INT_ARRAY_VALUE
- TYPE_INT_VALUE
- TYPE_LONG_ARRAY_VALUE
- TYPE_LONG_VALUE
- TYPE_NO_VALUE_VALUE
- TYPE_SHORT_ARRAY_VALUE
- TYPE_SHORT_VALUE
- TYPE_STRING_ARRAY_VALUE
- TYPE_STRING_VALUE

The returned object is a new ExtendedDataElementDescription object.

2. Populate the fields of the extended data element description. The ExtendedDataElementDescription class provides a setter method for each of the fields in an extended data element description. Refer to the Javadoc API documentation for complete information about these methods. For example, to specify that an extended data element must occur at least once, you would set the *maxOccurs* property to 4 using the setMaxOccurs(int) method:

   ```
   edeDesc.setMaxOccurs(4);
   ```

3. **Optional:** To add a child extended data element description, use the ExtendedDataElementDescription.addChild() method.

   ```
   edeDesc.addChild(childEdeDesc);
   ```

   The *childEdeDesc* parameter must be a valid extended data element description.

4. Add the extended data element description to the event definition using the EventDefinition.addExtendedDataElementDescription() method.

   ```
   definition.addExtendedDataElementDescription(edeDesc);
   ```

   If the event definition already includes another extended data element description with the same name and path, a DescriptionExistsException exception is thrown.

The following code fragment creates a new extended data element description, populates it with data, and adds it to an event definition.

```
ExtendedDataElementDescription edeDesc =
     new ExtendedDataElementDescription("age", TYPE_SHORT_VALUE);
edeDesc.setMinOccurs(1);
edeDesc.setMaxOccurs(1);

// definition is a valid event definition
definition.addExtendedDataElementDescription(edeDesc);
```

Creating an event catalog bean

The event catalog is implemented as a stateless session bean using the Enterprise JavaBeans

architecture. To access the event catalog, an event catalog application must first create an instance of the event catalog session bean.

Use the home interface to create an instance of the event catalog session bean.

```
//use home interface to create event catalog bean
InitialContext context = new InitialContext();
Object eventCatalogHomeObj =
     context.lookup("ejb/com/ibm/events/catalog/EventCatalog");
EventCatalogHome eventCatalogHome = (EventCatalogHome)
  PortableRemoteObject.narrow(eventCatalogHomeObj,
                              EventCatalogHome.class);
eventCatalog = (EventCatalog) eventCatalogHome.create();
```

Adding an event definition to the event catalog

After you have created a new event definition and populated it with property descriptions and extended data element descriptions, you can add it to the event catalog. Once added to the event catalog, an event definition cannot be modified, but it can be replaced.

Note: If WebSphere security is enabled, the application user ID must be mapped to the catalogAdministrator role to add event definitions to the event catalog.

To add an event definition to the event catalog, use the addEventDefinition method.

```
boolean result = eventCatalog.addEventDefinition(definition, replace)
```

The parameters of this method are as follows:

definition
The event definition you want to add. This must be a valid instance of EventDefinition.

replace
A boolean value indicating whether the specified event definition replaces an existing definition that has the same name.

If the *replace* parameter is `false`, the name of the specified event definition must not match that of any existing event definition in the catalog. If it does, an EventDefinitionExistsException exception is thrown.

If the *replace* parameter is `true`, the new event definition replaces any existing event definition with the same name that is already in the catalog. However, to preserve the inheritance hierarchy, the new event definition must name the same parent as the old event definition; otherwise, a ParentNotValidException exception is thrown.

The returned boolean indicates whether an existing event definition was replaced. This is equal to true only if *replace* is equal to `true` and an event definition with the same name was replaced by the new definition.

When an event definition is added to the event catalog, the event catalog sends an event to the event server notifying event consumers that this change occurred. See "Change notification" on page 88.

Note: If you attempt to add an event definition that violates inheritance rules, an InheritanceNotValidException exception is thrown and the event definition is not added to the catalog. This can happen if a new event definition overrides inherited property or extended data element descriptions in ways that are not valid, or if replacing an existing event definition would cause descendants to override inherited descriptions in ways that are not valid. For more information, see "Inheritance" on page 86.

Removing an event definition from the catalog

If an event definition is no longer needed, you can remove it from the event catalog.

Note: If WebSphere security is enabled, the application user ID must be mapped to the catalogAdministrator role to remove event definitions from the event catalog.

To remove an event definition from the event catalog, use the removeEventDefinition method.

```
eventCatalog.removeEventDefinition(name)
```

The *name* parameter is the name of the event definition you want to remove from the event catalog. If no matching event definition exists in the event catalog, an EventDefinitionNotFoundException exception is thrown.

When an event definition is removed from the event catalog, its children and all other descendants are also removed. For each event definition that is removed, the event catalog sends an event to the event server notifying event consumers that this change has taken place. See "Change notification" on page 88 for more information.

Note: When an event definition is removed, the event catalog does not check the event server to determine whether any existing events in the event data store are described by that event definition. Therefore, you should make certain that an event definition is no longer needed before removing it from the event catalog.

Querying event definitions

You can use the methods of the event catalog bean in order to query existing event definitions. Queries exist for retrieving event definitions by name and for retrieving event definitions that satisfy specific inheritance relationships.

Note: If WebSphere security is enabled, the application user ID must be mapped to the catalogReader or catalogAdministrator role to query events definitions.

To query event definitions, use the appropriate method of the EventCatalog class. Methods exist for querying specific event definition or querying multiple event definitions based on name or inheritance. You can also query the root event definition.

Querying an event definition by name

To query a specific event definition by name, use the getEventDefinition() method:

```
EventDefinition definition =
  eventCatalog.getEventDefinition(name, resolve);
```

The parameters of this method are as follows:

name
A string specifying the name of the event definition you want to query.

resolve
A boolean value indicating whether you want the returned event definition to be resolved (`true`) or unresolved (`false`). See "Inheritance" on page 86 for more information.

The returned object is the event definition matching the specified name. If no matching event definition exists in the catalog, the returned object is null.

Querying event definitions by pattern

To query all event definitions whose names match a specified pattern, use the getEventDefinitions() method:

```
EventDefinition[] definitions =
  eventCatalog.getEventDefinitions(pattern, resolve);
```

The parameters of this method are as follows:

pattern

A string specifying the pattern to be compared to the names of event definitions. In this string, a percent character (`%`) matches any sequence of zero or more characters, and an underscore (_) matches any single character. All other characters are treated literally. For example, the pattern `insurance%` would match all event definitions whose names begin with the word `insurance`.

You can use a backslash (`\`) escape character to specify a literal percent or underscore character. For example, the pattern `insurance\_` would match all event definitions whose names begin with the string `insurance_`. (To specify a backslash as part of the pattern, type two backslashes.)

resolve
A boolean value indicating whether you want the returned event definitions to be resolved (`true`) or unresolved (`false`). See "Inheritance" on page 86 for more information.

The returned object is an array containing all event definitions that match the specified pattern. If no matching event definitions exist in the event catalog, the returned array is empty.

Querying the parent of an event definition

To query the immediate parent of a specified event definition, use the getParent() method:

```
EventDefinition definition =
  eventCatalog.getParent(name, resolve);
```

The parameters of this method are as follows:

name
A string specifying the name of the event definition whose parent you want to query.

resolve
A boolean value indicating whether you want the returned event definition to be resolved (`true`) or unresolved (`false`). See "Inheritance" on page 86 for more information.

The returned object is the immediate parent event definition of the specified event definition. If the specified event definition has no parent (which is true only of the root definition), this method returns null. If no event definition in the catalog matches the specified name, an EventDefinitionNotFoundException exception is thrown.

Querying the ancestors of an event definition

To query the ancestors of a specified event definition (all of the event definitions from which it inherits, either directly or indirectly), use the getAncestors() method:

```
EventDefinition[] definitions =
  eventCatalog.getAncestors(name, resolve);
```

The parameters of this method are as follows:

name
A string specifying the name of the event definition whose ancestors you want to query.

resolve
A boolean value indicating whether you want the returned event definitions to be resolved (`true`) or unresolved (`false`). See "Inheritance" on page 86 for more information.

The returned object is an array containing all of the ancestors of the specified event definition. If the specified event definition has no ancestors (which is true only of the root definition), this method returns an empty array. If no event definition in the catalog matches the specified name, an EventDefinitionNotFoundException exception is thrown.

Querying the children of an event definition

To query the immediate children of a specified event definition, use the getChildren() method:

```
EventDefinition[] definitions =
  eventCatalog.getChildren(name, resolve);
```

The parameters of this method are as follows:

name
A string specifying the name of the event definition whose children you want to query.

resolve
A boolean value indicating whether you want the returned event definitions to be resolved (`true`) or unresolved (`false`). See "Inheritance" on page 86 for more information.

The returned object is an array containing all of the immediate children of the specified event definition. If the specified event definition has no children, the

returned array is empty. If no event definition in the catalog matches the specified name, an EventDefinitionNotFoundException exception is thrown.

Querying the descendants of an event definition

To query the descendants of a specified event definition (all of the event definitions that inherit from it, either directly or indirectly), use the getDescendants() method:

```
EventDefinition[] definitions =
  eventCatalog.getDescendants(name, resolve);
```

The parameters of this method are as follows:

name
A string specifying the name of the event definition whose descendants you want to query.

resolve
A boolean value indicating whether you want the returned event definitions to be resolved (`true`) or unresolved (`false`). See "Inheritance" on page 86 for more information.

The returned object is an array containing all of the descendants of the specified event definition. If the specified event definition has no descendants, this method returns an empty array. If no event definition in the catalog matches the specified name, an EventDefinitionNotFoundException exception is thrown.

Querying the root event definition

To query the root event definition, use the getRoot() method:

```
EventDefinition definition = eventCatalog.getRoot();
```

The returned object is the root event definition, which by default is `event`. If the event catalog is empty, this method returns null.

Event classes and source categories

The event catalog supports binding event definitions to source categories. This binding associates an event class (identified by extension name) to the name of an arbitrarily defined source category. Applications can use these categories to manage event classes in logical groups. (Note that these categories are entirely distinct from event groups, which are used to categorize event instances according to their content.)

The event catalog does not parse or interpret the source category; the mapping between event classes and categories can be anything that is meaningful to applications that use the event catalog. An event class can belong to multiple source categories.

An event catalog application can create new bindings between event classes and source categories, remove an existing binding, and perform various queries based on source categories.

Note: If WebSphere security is enabled, the application user ID must be mapped to the catalogReader or catalogAdministrator role to query source category bindings. To create or delete source category bindings, the application user ID must be mapped to the catalogAdministrator role.

Creating a source category binding

To bind an event class to a source category, use the EventCatalog.bindEventExtensionToSourceCategory method:

```
eventCatalog.bindEventExtensionToSourceCategory(extensionName, sourceCategory);
```

The parameters of this method are as follows:

extensionName
A string specifying the extension name of an event class (the value of the *extensionName* attribute). This must be a legal extension name as defined by the CommonBaseEvent specification.

sourceCategory
A string specifying a source category. This can be any string, provided it is no longer than 64 characters.

Removing a source category binding

To remove a binding between an event class and a source category, use the EventCatalog.unbindEventExtensionFromSourceCategory method:

```
eventCatalog.unbindEventExtensionFromSourceCategory(extensionName, sourceCategory);
```

The parameters of this method are as follows:

extensionName
A string specifying the extension name of an event class (the value of the *extensionName* attribute). This must be a legal extension name as defined by the CommonBaseEvent specification.

sourceCategory
A string specifying a source category. This can be any string, provided it is no longer than 64 characters.

Querying source category bindings

You can use the methods of the event catalog bean in query source category bindings. Queries exist for retrieving the event classes that belong to a source category, the source categories associated with an event class, or a set of source category bindings that match a specified pattern.

To query source category bindings, use the appropriate method of the EventCatalog class.

getEventExtensionNamesForSourceCategory()
Queries the event classes that belong to a specified source category:

```
String[] evClasses =
  eventCatalog.getEventExtensionNamesForSourceCategory(name);
```

The parameters of this method are as follows:

name
A string specifying the name of a source category. This value cannot be longer than 64 characters.

The returned object is an array of strings, each one specifying an event class (the value of the *extensionName* attribute of events belonging to the class). If no event classes are bound to the specified source category, the returned array is empty.

getSourceCategoriesForEventExtension()

Queries the source categories associated with a specified event class::

```
String[] categories =
  eventCatalog.getSourceCategoriesForEventExtension(name);
```

The parameters of this method are as follows:

name
: A string specifying the name of an event class. This is the value of the *extensionName* attribute of events that belong to the class.

The returned object is an array of strings, each one specifying the name of a source category to which the specified event class belongs. If the specified event class is a member of any source categories, the returned array is empty.

getEventExtensionToSourceCategoryBindings()

Queries source category bindings for those that match a specified pattern:

```
java.util.Collection bindings =
  eventCatalog.getEventExtensionToSourceCategoryBindings(eventClassPattern,
                                                         categoryPattern);
```

The parameters of this method are as follows:

eventClassPattern
: A string specifying the pattern you want to compare to the event class names. Only bindings whose event class name match the specified pattern are returned. In this string, a percent character (%) matches any sequence of zero or more characters, and an underscore (_) matches any single character. All other characters are treated literally.

categoryPattern
: A string specifying the pattern you want to compare to the source category names. Only bindings whose source category name match the specified pattern are returned. In this string, a percent character (%) matches any sequence of zero or more characters, and an underscore (_) matches any single character. All other characters are treated literally.

The returned object is a collection of two-element arrays, each representing a source category binding. In each array, the first element is a string specifying the name of an event class; the second element is a string specifying the name of a source category. These arrays are sorted in ascending order, first by source category name and then by event class name. If no bindings exist, or no existing bindings match the specified patterns, the collection is empty.

For example, the following code fragment queries all bindings of event classes whose names begin with `insurance_claim`:

```
Collection bindings =
  catalog.getEventExtensionToSourceCategoryBindings("insurance_claim%",
                                                    "%");
```

Note: Remember that source category bindings simply associate event class names with source category names. The existence of a source category binding for a particular class name does not guarantee that an event definition exists for that event class. To retrieve the event definition associated with an event

class, you must use the EventCatalog methods for querying event definitions. See “Querying event definitions” on page 93 for more information.

Chapter 9. Security

You can use WebSphere method-level declarative security to restrict access to Common Event Infrastructure functions.

The Common Event Infrastructure defines seven security roles, each one associated with a related group of functions. These security roles control access to both programming interfaces and commands.

The following table describes the security roles and the types of users associated with each role.

Table 3. Security roles and user types

Security role	User types
eventCreator	Event sources that need to submit events to an emitter using synchronous EJB calls. This role provides access to the following interfaces: • Emitter.sendEvent() • Emitter.sendEvents() • emitevent.jacl **Note:** The eventCreator role restricts access to event submission only if the emitter is configured to use synchronous EJB calls for event transmission. If the emitter uses asynchronous JMS messaging for event transmission, you must use JMS security to restrict access to the destination used to submit events.
eventUpdater	Event consumers that need to update events stored in the event database. This role provides access to the following interfaces: • EventAccess.updateEvents() • EventAccess.eventExists() • EventAccess.queryEventByGlobalInstanceId() • EventAccess.queryEventsByAssociation() • EventAccess.queryEventsByEventGroup() • eventquery.jacl
eventConsumer	Event consumers that need to query events stored in the event database. This role provides access to the following interfaces: • EventAccess.eventExists() • EventAccess.queryEventByGlobalInstanceId() • EventAccess.queryEventsByAssociation() • EventAccess.queryEventsByEventGroup() • eventquery.jacl

Table 3. Security roles and user types (continued)

Security role	User types
eventAdministrator	Event consumers that need to query, update, and delete events stored in the event database. This role provides access to the following interfaces: • EventAccess.purgeEvents() • EventAccess.eventExists() • EventAccess.queryEventByGlobalInstanceId() • EventAccess.queryEventsByAssociation() • EventAccess.queryEventsByEventGroup() • EventAccess.updateEvents() • Emitter.sendEvent() • Emitter.sendEvents() • eventquery.jacl • eventpurge.jacl • emitevent.jacl • eventbucket.jacl
catalogReader	Event catalog applications that need to retrieve event definitions from the event catalog. This role provides access to the following interfaces: • EventCatalog.getAncestors() • EventCatalog.getChildren() • EventCatalog.getDescendants() • EventCatalog.getEventDefinition() • EventCatalog.getEventDefinitions() • EventCatalog.getEventExtensionNamesForSourceCategory() • EventCatalog.getEventExtensionToSourceCategoryBindings() • EventCatalog.getParent() • EventCatalog.getRoot() • EventCatalog.getSourceCategoriesForEventExtension() • eventcatalog.jacl (-listdefinitions option) • eventcatalog.jacl (-listcategories option) • eventcatalog.jacl (-exportdefinitions option)
catalogAdministrator	Event catalog applications that need to create, update, delete, or retrieve event definitions in the event catalog. This role provides access to all methods of the EventCatalog interface and all functions of the **eventcatalog.jacl** script. Because changes to the event catalog can result in generation of events, this role also provides access to event submission interfaces.

The event server message-driven bean runs using the WebSphere Application Server user identity. If you are using asynchronous JMS transmission to submit events to the event server, and you have enabled method-based security, you must map this user identity to the eventCreator role.

Note:

If Java 2 security is enabled, you must modify your policy file to enable access to certain functions:

- If you are running an event source application and you want to generate your own globally unique identifiers (GUIDs), add the following entries:

```
permission java.io.FilePermission "${java.io.tmpdir}${/}guid.lock",
  "read, write, delete";
permission java.net.SocketPermission "*", "resolve";
```

- If you are using the default filter plug-in or the notification helper to filter events using XPath event selectors, add the following entries:

```
permission java.util.PropertyPermission "*", "read";
permission java.io.FilePermission
  "${was.install.root}${/}java${/}jre${/}lib${/}jxpath.properties",
  "read";
```

Chapter 10. Command reference

Command-line scripts are available to access some functions of the Common Event Infrastructure. These scripts are implemented as Jacl scripts, which must be run using the WebSphere wsadmin tool (located in the $WAS_HOME/bin directory). For more information about the wsadmin tool, see the WebSphere Application Server documentation.

These scripts must be run using the following syntax:

```
wsadmin -f scriptname.jacl
```

Parameter names can be shortened, as long as you provide enough of the name to distinguish it from other parameters. For example, you can type `-ex` instead of `-extensionname` when using the **eventquery.jacl** script. (However, `-e` would not be valid, because it could represent either `-extensionname` or `-end`.)

To get help with syntax and usage for a command, type the command followed by the word `help`:

```
wsadmin -f scriptname.jacl help
```

Note: If you are using the wsadmin tool with the SOAP protocol, a command might time out before the operation can complete. For example, this might happen if you query or purge a large number of events from the event server. If this happens, the wsadmin tool displays an error message indicating a failed SOAP RPC call:

```
Failed to make the SOAP RPC call: invoke
```

If this happens, run the command again, specifying RMI as the connection type and 2809 as the destination port. For example, the following command purges events from the event server using an RMI connection:

```
wsadmin -conntype rmi -port 2809 eventpurge.jacl -seconds 0
```

For more information about the `-conntype` parameter of the wsadmin tool, refer to the WebSphere Application Server documentation.

emitevent.jacl

Purpose

Sends an event to the event server.

wsadmin -f emitevent.jacl [**-xml** *url*] [**-msg** *message*] [**-severity** *severity*] [**-extensionname** *extension_name*] [**-emitter** *profile_name*] [**-synchronous** | **-asynchronous**] [**-serverName** *server_name*]

Description

The **emitevent.jacl** script provides a command-line interface for submitting events to the event server. You can provide the event content by providing a source XML file or by specifying certain property values on the command line.

Events generated by this script have the following default content:

```
<CommonBaseEvent creationTime=current_system_time version="1.0.1">
 <sourceComponentId component="emitevent.jacl" componentIdType="Application"
  location=local_hostname locationType="Hostname"
  subComponent="com.ibm.events.cli.util.EmitEventCliHelper"
  componentType="http://www.ibm.com/namespaces/autonomic/Tivoli/EventInfrastructure"/>
 <situation categoryName="ReportSituation">
  <situationType xsi:type="ReportSituation" reasoningScope="EXTERNAL"
   reportCategory="CLI"/>
 </situation>
</CommonBaseEvent>
```

The *current_system_time* parameter is the system time when the event is generated, specified as an XML dateTime string.

You can override any of these default values, or provide additional property values, by specifying a source XML file or by specifying property values on the command line.

Parameters

-xml *url*

A uniform resource locator (URL) specifying the location of an XML document containing the event to be submitted. This XML document must conform to the Common Base Event version 1.0.1 XSD schema. If no URL scheme (such as `http://`) is specified, a local file is assumed. This parameter is optional.

Two sample XML files, eventsample1.xml and eventsample2.xml, are availabe in $WAS_HOME/events/samples.

-msg *message*

The value to use for the *message* property of the event. If the message contains spaces, enclose this value in quotation marks. This parameter is optional. If you specify this parameter in addition to an XML file, the value of the **-msg** parameter overrides any value specified in the XML file for the *msg* property.

-severity *severity*

The value to use for the *severity* property of the event. This parameter is optional. If you specify this parameter in addition to an XML file, the value of the **-severity** parameter overrides any value specified in the XML file for the *severity* property.

-extensionname *extension_name*

The value to use for the *extensionName* property of the event. If the extension name contains spaces, enclose this value in quotation marks. This parameter is optional. If you specify this parameter in addition to an XML file, the value of the **-extensionname** parameter overrides any value specified in the XML file for the *extensionName* property.

-emitter *profile_name*

The JNDI name of the emitter factory profile to use when obtaining an emitter. This parameter is optional; if it is not specified, the default emitter factory profile (`/com/ibm/events/configuration/emitter/Default`) is used.

-synchronous | -asynchronous

The synchronization mode to use for event transmission. This parameter is optional; if it is not specified, the preferred synchronization mode configured for the emitter is used.

-serverName *server_name*

The name of the application server where the event server is deployed. You must specify this parameter if there are multiple application servers running on the WebSphere node.

Examples

The following example sends an event to the event server, specifying a severity of 30 and the extension name `test_event` (all other properties have the default values):

```
wsadmin -f emitevent.jacl -severity 30 -extensionname test_event
```

The following example sends an event using the properties specified in eventsample1.xml:

```
wsadmin -f emitevent.jacl -xml ../samples/eventsample1.xml
```

eventbucket.jacl

Purpose

Displays or changes the event database bucket configuration.

wsadmin -f eventbucket.jacl [**-status**] [**-change**] [**-serverName** *server_name*]

Description

The **eventbucket.jacl** script displays or changes the event database bucket configuration. Buckets are used by the rapid purge utility to purge old event data from the event database. By running this command, you can determine the current bucket configuration, or you can swap the active and inactive buckets.

Note: If WebSphere security is enabled, your user ID must be mapped to the eventAdministrator role to view or change the event database bucket configuration.

Parameters

-status
Displays information about the current bucket configuration, including the active bucket setting and the bucket check interval (the frequency with which the data store plug-in checks to determine which bucket is active).

-change
Swaps the buckets so the active bucket becomes inactive and the inactive bucket becomes active. The inactive bucket must be empty before you can use this option.

-serverName *server_name*
The name of the application server where the event server is deployed. You must specify this parameter if there are multiple application servers running on the WebSphere node.

Examples

This example displays the current bucket configuration:

```
wsadmin -f eventbucket.jacl -status
```

This example swaps the active and inactive buckets:

```
wsadmin -f eventbucket.jacl -change
```

eventcatalog.jacl

Purpose

Lists event definitions or source categories in the Event Catalog and imports and exports event definitions.

wsadmin -f eventcatalog.jacl [**-listdefinitions** | **-listcategories** | **-exportdefinitions** | **-importdefinitions**] [**-file** *filename*] [**-name** *event_def_name*] [**-pattern**] [**-resolve**] [**-replace**] [**-serverName** *server_name*]

Description

The **eventcatalog.jacl** script provides command-line access to the contents of the Event Catalog. It also provides support for importing and exporting event definitions.

Note: If WebSphere security is enabled, your user ID must be mapped to the catalogReader or catalogAdministrator role to list or export event definitions or to list source categories. To import event definitions, your user ID must be mapped to the catalogAdministrator role.

Parameters

-listdefinitions
Lists specified event definitions in a readable format, sorted by name in ascending order. The listing is written to the file specified by the **-file** parameter; if this parameter is not specified, the listing is written to the standard output.

-listcategories
Lists all defined event source categories and the event classes they contain, sorted by source category in ascending order. The listing is written to the file specified by the **-file** parameter; if this parameter is not specified, the listing is written to the standard output.

-exportdefinitions
Lists specified event definitions in a format suitable for importing. The listing is written as an XML document conforming to the eventdefinition5_0_1.xsd XSD schema, which is packaged in the events-client.jar JAR file. The listing is written to the file specified by the **-file** parameter; if this parameter is not specified, the listing is written to the standard output.

-importdefinitions
Reads a listing of event definitions from a file and adds the event definitions to the Event Catalog. The listing of event definitions to import must be written as an XML document conforming to the eventdefinition.xsd XSD schema.

-file *filename*
For a list or export operation, the name of the file to which the output is written. For an import operation, the file containing the event definitions to be imported. This parameter is required for import operations and optional for list and export operations. If this parameter is not specified for a list or export operation, the output is written to the standard output.

-name *event_def_name*
A name identifying the event definitions to be listed or exported. If the **-pattern** parameter is not specified, the **-name** parameter identifies a single specific event definition. If **-pattern** is specified, **-name** specifies a pattern

against which event definition names are compared. In this pattern, a percent character (%) matches any sequence of zero or more characters, and an underscore (_) matches any single character. All other characters are treated literally.

This parameter is valid only with the **-listdefinitions** and **-exportdefinitions** options. It is not valid with the **-listcategories** or **-importdefinitions** options.

-pattern
Specifies that the value specified with the **-name** parameter is to be treated as a pattern. This parameter is valid only with the **-listdefinitions** and **-exportdefinitions** options. It is not valid with the **-listcategories** or **-importdefinitions** options.

-resolve
Specifies that the event definitions to be listed or exported should be resolved rather than raw. A resolved event definition includes the property and extended data element descriptions inherited from its ancestors in the inheritance hierarchy. If this parameter is not specified, the event definition listing contains only the raw event definitions.

This parameter is valid only with the **-listdefinitions** and **-exportdefinitions** options. It is not valid with the **-listcategories** or **-importdefinitions** options.

-replace
Specifies that the event definitions being imported should replace any existing event definitions with the same names. If this parameter is not specified, a name collision between an existing event definition and an imported event definition results in an error, and no event definitions are imported.

This parameter is valid only with the **-importdefinitions** option. It is not valid with the **-listdefinitions**, **-listcategories**, or **-exportdefinitions** options.

-serverName *server_name*
The name of the application server where the event server is deployed. You must specify this parameter if there are multiple application servers running on the WebSphere node.

Examples

This example displays the contents of a single, resolved event definition named `insurance_claim_start`, writing the result to standard output:

```
wsadmin -f eventcatalog.jacl -listdefinitions -name insurance_claim_start -resolve
```

This example exports a set of event definitions whose names begin with the string `insurance_claim_start`, writing the result to an XML file:

```
wsadmin -f eventcatalog.jacl -exportdefinitions -file d:\myexport.xml
  -name insurance_claim_start% -pattern
```

This example imports a set of event definitions from the file myimport.xml, replacing any existing definitions with the same names:

```
wsadmin -f eventcatalog.jacl -importdefinitions -file d:\myimport.xml -replace
```

This example displays a listing of all defined event source categories and the events they contain, writing the result to standard output:

```
wsadmin -f eventcatalog.jacl -listcategories
```

eventpurge.jacl

Purpose

Purges events from the event database.

wsadmin -f eventpurge.jacl [**-seconds** *seconds* | **-end** *end_time*] [**-group** *event_group*] [**-severity** *severity*] [**-extensionname** *extension_name*] [**-start** *start_time*] [**-size** *size*] [**-serverName** *server_name*]

Description

The **eventpurge.jacl** script purges events from the event database. You can purge all events from the event database, or you can limit the purge to events meeting certain criteria.

Note: If WebSphere security is enabled, your user ID must be mapped to the eventAdministrator role to delete events.

Parameters

-seconds *seconds*
The minimum age of events you want purged. The *seconds* value must be an integer. Only events older than the specified number of seconds are purged. This parameter is required if you do not specify the **-end** parameter.

-end *end_time*

The end time of the group of events you want to delete. Only events generated before the specified time are deleted. The *end_time* value must be specified in the XML dateTime format (`CCYY-MM-DDThh:mm:ss`). For example, noon on 1 January 2006 in Eastern Standard Time would be `2006-01-01T12:00:00-05:00`. For more information about the dateTime data type, refer to the XML schema at www.w3.org.

This parameter is required if you do not specify the **-seconds** parameter.

-group *eventGroup*
The event group from which to purge events. The *event_group* value must be the name of an event group defined in the Common Event Infrastructure configuration. This parameter is optional.

-severity *severity*
The severity of events you want purged. The *severity* value must be an integer; only events whose severity is equal to the value you specify are purged. This parameter is optional.

-extensionname *extension_name*
The extension name of events you want included in the purge. Use this parameter to restrict the purge to events of a specific type. Only events whose extensionName property is equal to *extensionName* are purged. This parameter is optional.

-start *start_time*
The beginning time of the group of events you want to delete. Only events generated after the specified time are deleted. The *start_time* value must be specified in the XML dateTime format (`CCYY-MM-DDThh:mm:ss`). This parameter is optional.

-size *size*
The number of events to purge in a single transaction. The *size* value must be an integer. After this number of events have been purged, the command commits the transaction before continuing in a new transaction. This parameter is optional.

-serverName *server_name*
The name of the application server where the event server is deployed. You must specify this parameter if there are multiple application servers running on the WebSphere node.

Example

The following example purges all events from the database whose severity is 20 (harmless) and were generated earlier than 10 minutes ago.

```
eventpurge.jacl -group "All events" -severity 20 -seconds 600
```

eventquery.jacl

Purpose

Generates a report listing events in the event database.

wsadmin -f eventquery.jacl [**-globalinstanceid** *global_instance_id* | **-group** *event_group*] [**-severity** *severity*] [**-extensionname** *extension_name*] [**-start** *start_time*] [**-end** *end_time*] [**-number** *number*] [**-ascending**|**-descending**] [**-serverName** *server_name*]

Description

The **eventquery.jacl** script queries the event database and generates a report listing the result. You can query events based event group, severity, or extension name; you can also query events that were created during a specified period of time.

Note: If WebSphere security is enabled, your user ID must be mapped to the eventConsumer or eventAdministrator role to query events.

Parameters

-globalinstanceid *global_instance_id*
The global instance identifier of the event to query. Either this parameter or **-group** (but not both) is required.

-group *event_group*
The event group from which to query events. The *event_group* value must be the name of an event group defined in the Common Event Infrastructure configuration. Either this parameter or **-globalinstanceid** (but not both) is required.

-severity *severity*
The severity of events you want included in the report. The *severity* value must be an integer; only events whose severity is equal to the value you specify are included in the report. This parameter is optional.

-extensionname *extension_name*
The extension name of events you want included in the report. Use this

parameter to restrict the query to events of a specific type. Only events whose extensionName property is equal to *extensionName* are included in the report. This parameter is optional.

-start *start_time*
The earliest time of events you want included in the report. Use this parameter to restrict the query to events generated after a specified date and time. This parameter must be a date and time specified according to the XML dateTime data type. The basic format is `CCYY-MM-DDThh:mm:ss`, optionally followed by a time zone indicator. For example, noon on 1 January 2006 in Eastern Standard Time would be `2006-01-01T12:00:00-05:00`. For more information about the dateTime data type, refer to the XML schema at www.w3.org.

-end *end_time*
The latest time of events you want included in the report. Use this parameter to restrict the query to events generated before a specified date and time. This parameter must be a date and time specified according to the XML dateTime data type (for more information, see the description of the **-start** parameter.

-number *number*
The maximum number of events you want included in the report. This parameter must be an integer. If the number of matching events in the database exceeds the specified value, the report is truncated. If the report is sorted in ascending order, this means that the most recent matching events are omitted; if the report is sorted in descending order, the oldest matching events are omitted.

-ascending|-descending
The chronological order in which the events in the report are sorted. This must be one of the following values:

ascending
Ascending (chronological) order, with the oldest events first. This is the default value.

descending
Descending (reverse chronological) order, with the most recent events first.

-serverName *server_name*
The name of the application server where the event server is deployed. You must specify this parameter if there are multiple application servers running on the WebSphere node.

Example

The following example lists all events from the database that belong to the **Critical events** event group and were generated on 17 February 2004, sorting the report in reverse chronological order:

```
eventquery.jacl -group "All events" -start "2004-02-17T00:00:00-05:00"
  -end "2004-02-17T23:59:59-05:00" -order DESC
```

Message reference

Message reference

CEICA0001E The value of the parameter *parameter_name* **cannot be null for the object** *object_name* **.**

Explanation: See message.

Administrator Response: Repeat the operation and provide a value for the specified parameter.

CEICA0002E The value of the parameter *parameter_name* **is not valid because the value is set to an empty string for the object** *object_name* **.**

Explanation: See message.

Administrator Response: Repeat the operation and provide a value for the specified parameter.

CEICA0003E The value of an element of the string array specified is not valid because it is null.
Object type: *object_type*
Object name: *object_name*
Parameter name: *parameter_name*
Index: *index*

Explanation: See message.

Administrator Response: Repeat the operation and provide a value for the element.

CEICA0004E The value of a parameter is not valid because it exceeds the maximum number of characters.
Property name: *property_name*
Parameter: *parameter_name*
Maximum number of characters: *maximum_length*
Value: *value*

Explanation: The number of characters that can be used for the value of the specified parameter is restricted because either it corresponds to a field in the Common Base Event specification that has a maximum length or the database field in which the value is stored is of a limited size.

Administrator Response: Repeat the operation and provide a value for the specified parameter that does not exceed the maximum number of characters.

CEICA0005E The element of the string array is not valid because it exceeds the maximum number of characters.
Property name: *property_name*
Parameter name: *parameter_name*
Index: *index*
Maximum number of characters: *maximum_length*
Value: *value*

Explanation: The number of characters that can be used for the value of the element is restricted because either it corresponds to a field in the Common Base Event specification that has a maximum length or because the database field in which the value is stored is of a limited size.

Administrator Response: Repeat the operation and provide a value for the element that does not exceed the maximum number of characters.

CEICA0006E The event definition *event_definition_name* cannot be its own parent.

Explanation: Event definitions are arranged in a hierarchy in which an event definition can be the parent of any number of other event definitions. The name of an event definition cannot be used as its own parent name.

Administrator Response: Repeat the operation and provide a valid value for the parentName parameter.

CEICA0007E The value *value* for the parameter *parameter_name* is not valid because it is a negative number.

Explanation: See message.

Administrator Response: Repeat the operation and provide a value for the specified parameter that is greater than or equal to zero.

CEICA0008E The event definition already describes the extended data element.
Event definition name: *event_definition_name*
Extended data element name: *extended_data_element_name*

Explanation: An event definition can describe a number of extended data elements. Each of these extended data elements must have a unique name.

Administrator Response: Do one of the following:

- Recreate the extended data element description using a different name. Then add it to the event definition.
- Recreate the event definition without the original extended data element description. Then add the new extended data element description to the event definition.

CEICA0009E The extended data element description already describes the named child extended data element.
Extended data element name: *extended_data_element_name*
Child extended data element name: *child_extended_data_element_name*

Explanation: An extended data element description can contain a number of child extended data element descriptions. Each of the immediate child extended data element descriptions must have a unique name.

Administrator Response: Do one of the following:

- Recreate the child extended data element using a different name. Then add it to the extended data element description.
- Recreate the event definition without the original child extended data element description. Then add the new child extended data element description to the extended data element description.

CEICA0010E The event definition already describes the named property.
Event definition name: *event_definition_name*
Property name: *property_name*

Explanation: An event definition can describe a number of properties. Each of these properties must have a unique name.

Administrator Response: Do one of the following:

- Recreate the property description using a different name. Then add it to the event definition.
- Recreate the event definition without the original property description. Then add the new property description to the event definition.

CEICA0011E The value specified for the type parameter is not valid.
Extended data element name: *extended_data_element_name*
Type: *extended_data_element_type_integer*

Explanation: See message.

Administrator Response: Repeat the operation and provide a valid value for the type parameter. Valid values for the type of an extended data element are listed in the documentation for the class org.eclipse.hyades.logging.events.cbe.ExtendedDataElement.

CEICA0013E Default string values cannot be set for an extended data element of type hexBinary. If required, use the setDefaultHexValue method to set a default hexadecimal value.
Extended data element name: *extended_data_element_name*
Type: *extended_data_element_type_integer*

Explanation: Most types of extended data element can have a number of default string values. The hexBinary type, however, can have only a single default hexadecimal value.

Administrator Response: See message.

CEICA0014E A default hexadecimal value cannot be set for an extended data element whose type is not hexBinary. If required, use the setDefaultValues method to set default string values.
Extended data element name: *extended_data_element_name*
Type: *extended_data_element_type_integer*

Explanation: The only type of extended data element that can have a default hexadecimal value is hexBinary. Most other types can have some number of default string values.

Administrator Response: See message.

CEICA0015E Default values cannot be set for an extended data element of type noValue.
Extended data element name: *extended_data_element_name*
Type: *extended_data_element_type_integer*

Explanation: See message.

Administrator Response: Do not use the setDefaultValues method for an extended data element of type noValue.

CEICA0016E Only one default value can be set for the extended data element because it is a single-valued type, such as string, float, or byte.
Extended data element name: *extended_data_element_name*
Type: *extended_data_element_type_integer*

Explanation: See message.

Administrator Response: Use a single value in the call to the setDefaultValues method.

CEICA0017E The default hexadecimal value specified in the parameter exceeds the maximum number of bytes.
Extended data element name: *extended_data_element_name*
Number of bytes: *value_length*
Maximum number of bytes: *maximum_length*

Explanation: The value of the default hexadecimal value is limited due to a restriction of the database field in which the value is stored.

Administrator Response: Repeat the operation and provide a value that does not exceed the maximum number of bytes.

CEICA0018E The value specified for the minOccurs parameter is greater than the current value of the maxOccurs parameter.
Extended data element name: *extended_data_element_name*
minOccurs parameter: *integer*
maxOccurs parameter: *integer*

Explanation: The minimum number of times an extended data element can occur must be less than or equal to the maximum number of times.

Administrator Response: Set the value of the maxOccurs parameter first, or specify a smaller value for the minOccurs parameter.

CEICA0019E The value specified for the maxOccurs parameter is less than the current value of the minOccurs parameter.
Extended data element name: *extended_data_element_name*
maxOccurs parameter: *integer*
minOccurs parameter: *integer*

Explanation: The minimum number of times an extended data element can occur must be less than or equal to the maximum number of times.

Administrator Response: Set the minOccurs parameter first, or specify a larger value for the maxOccurs parameter.

CEICA0020E Permitted values cannot be set for a property when a minimum or a maximum value is already set.
Property name: *property_name*
Minimum value: *minimum_value*
Maximum value: *maximum_value*

Explanation: A property description cannot have permitted values if it has a minimum or a maximum value. Similarly, a property description cannot have a minimum or a maximum value if it has permitted values.

Administrator Response: Repeat the operation and do not use the setPermittedValues method.

CEICA0021E A minimum or a maximum value cannot be set for a property when permitted values are already set.
Property name: *property_name*
Parameter: *parameter_name*
Permitted values: *permitted_values*

Explanation: A property description cannot have a minimum a maximum value if it has permitted values. Similarly, a property description cannot have permitted values if it has a minimum or a maximum value.

Administrator Response: Repeat the operation and do not use the setMinimumValue method or the setMaximumValue method.

CEICA0022E The event definition *event_definition_name* **cannot be found.**

Explanation: The event catalog does not contain an event definition with the specified name.

Administrator Response: Verify that the name you are using is the correct name.

CEICA0023E The event definition *event_definition_name* **was not added because an event definition with the same name already exists.**

Explanation: Each event definition in the event catalog must have a unique name. When adding an event definition, you can replace an existing event definition.

Administrator Response: See message.

CEICA0024E The event definition was not added because it has a different parent than the event definition that it is replacing.
Event definition name: *event_definition_name*
Parent name: *parent_event_definition_name*
Parent name of existing event definition: *existing_parent_event_definition_name*

Explanation: When replacing an event definition in the event catalog, the new event definition and the existing definition must have the same parent. In other words, the event definitions must occupy the same position in the hierarchy of event definitions.

Administrator Response: Recreate the event definition using the parent name of the existing event definition. Then add the event definition to the event catalog.

CEICA0025E The event definition *event_definition_name* **was not added to the event catalog because it is a root definition and a root definition already exists.**

Explanation: The event catalog can contain only one root definition. When adding an event definition, you can replace an existing event definition.

Administrator Response: See message.

CEICA0026E The event definition *event_definition_name* **was not added to the event catalog because its parent event definition** *parent_event_definition_name* **does not exist.**

Explanation: The event catalog does not contain an event definition for the specified parent name.

Administrator Response: Verify that the name of the parent definition that you are using is the correct name.

CEICA0027E The event definition was not added to the event catalog because it contains an extended data element with a type that is different than the type of its ancestor event definition.
Event definition name: *event_definition_name*
Extended data element name: *extended_data_element_name*
Type: *extended_data_element_type*
Ancestor event definition name: *event_definition_name*
Ancestor event definition type: *extended_data_element_type*

Explanation: Inheritance by an event definition of the extended data element descriptions of an ancestor is governed by event definition inheritance rules. These rules are described in the documentation for class com.ibm.events.catalog.EventDefinition in the event catalog API. An event definition cannot change the type of an extended data element description that it inherits.

Administrator Response: Recreate the event definition either omitting the extended data element description or making its type the same as the type of the ancestor event definition. Then add the event definition to the event catalog.

CEICA0028E The event definition was not added because the ancestor event definition requires the property.
Event definition name: *event_definition_name*
Ancestor event definition name: *event_definition_name*
Property name: *property_name*

Explanation: Inheritance by an event definition of the property descriptions of an ancestor is governed by event definition inheritance rules. These rules are described in the documentation for class com.ibm.events.catalog.EventDefinition in the event catalog API. An event definition cannot change the `required` setting of a property description that it inherits from true to false.

Administrator Response: Recreate the event definition omitting the property description or making its `required` setting true. Then add the event definition to the event catalog.

CEICA0029E The event definition was not replaced because it contains an extended data element with a different type than a descendant event definition.
Event definition name: *event_definition_name*
Extended data element name: *extended_data_element_name*
Type: *extended_data_element_type*
Descendent event definition name: *event_definition_name*
Descendent event definition type name: *extended_data_element_type* **.**

Explanation: Inheritance by an event definition of the extended data element descriptions of an ancestor is governed by event definition inheritance rules. These rules are described in the documentation for class com.ibm.events.catalog.EventDefinition in the Event Catalog API. An event definition cannot change the type of an extended data element description that it inherits. Replacing the event definition would cause a descendant event definition to violate this rule.

Administrator Response: Recreate the event definition either omitting the extended data element description or making its type the same as the type of the descendant event definition, or delete the descendant event definition from the event catalog. Then add the event definition to the event catalog.

CEICA0030E The event definition was not replaced because the required setting of a property description is not the same as the required setting for a descendant event definition.
Event definition name: *event_definition_name*
Property name: *property_name*
Required setting: *required*
Descendant event definition name: *event_definition_name*
Descendant required setting: *descendant_required*

Explanation: Inheritance by an event definition of the property descriptions of an ancestor is governed by event definition inheritance rules. These rules are described in the documentation for class com.ibm.events.catalog.EventDefinition in the event catalog API. An event definition cannot change the `required` setting of a property description that it inherits. Replacing the event definition would cause a descendant event definition to violate this rule.

Administrator Response: Recreate the event definition either omitting the property description or making its `required` setting false, or delete the descendant event definition from the event catalog. Then add the event definition to the event catalog.

CEICA0031E The file *file_name* was not found in the class path.

Explanation: The event catalog expected to find a file in one of the directories listed in the class path, but the file does not exist. The file is a Common Event Infrastructure internal file. This could indicate a problem with the file system or an error in the installation of the Common Event Infrastructure application.

Administrator Response: Verify that the Common Event Infrastructure application is installed correctly.

CEICA0034E The XML document is not valid because it does not conform to the XML schema eventdefinition.xsd.
Parsing messages: *parse_error_messages*

Explanation: See message.

Administrator Response: Correct the XML document, then repeat the operation.

CEICA0035E An error occurred when a Document Object Model document was serialized to an XML document.

Explanation: This could indicate an error in the installation of the Common Event Infrastructure application.

Administrator Response: For more information about the cause of the error, see the exception message. If necessary, verify that the Common Event Infrastructure application is installed correctly.

CEICA0036E An error occurred when the XSLT style sheet *stylesheet_filename* was applied to a Document Object Model document.

Explanation: This problem could indicate an error in the installation of the Common Event Infrastructure application.

Administrator Response: For more information about the cause of the problem, see the exception message. If necessary, verify that the Common Event Infrastructure application is installed correctly.

CEICA0037E The instance of the enterprise bean *bean_name* could not be created.

Explanation: An error occurred when the create method was called on the local or remote home of the specified enterprise bean.

Administrator Response: For more information about the cause of the problem, see the exception message. If necessary, verify that the Common Event Infrastructure application is installed correctly and that the Common Event Infrastructure server is running correctly.

CEICA0038E The instance of the enterprise bean *bean_name* could not be removed.

Explanation: An error occurred when the remove method was called on the EJB local home of the specified EJB.

Administrator Response: For more information about the cause of the problem, see the exception message. If necessary, verify that the Common Event Infrastructure application is installed correctly and that the Common Event Infrastructure server is running correctly.

CEICA0039E A call using a finder method to find an instance of the enterprise bean failed.
Finder method name: *method_name*
Enterprise bean name: *bean_name*

Explanation: An error occurred when the finder method was called on the EJB local home of the specified enterprise bean.

Administrator Response: For more information about the cause of the problem, see the exception message. If necessary, verify that the Common Event Infrastructure application is installed correctly and that the Common Event Infrastructure server is running correctly.

CEICA0040E The lookup of the JNDI name failed.
JNDI name: *jndi_name*
Class name: *class_name*

Explanation: The JNDI service might not be running, or it cannot be reached.

Administrator Response: For more information about the cause of the problem, see the exception message. If necessary, verify that the Common Event Infrastructure application is installed correctly and that the Common Event Infrastructure server is running correctly.

CEICA0041E The parsing of the XML file *file_name* failed because an IO error occurred when processing the file.

Explanation: See message.

Administrator Response: Verify that the specified file exists and that is accessible by the Common Event Infrastructure server.

CEICA0042E The parsing of the XML file *file* failed because of a parsing exception.

Explanation: See message.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEICA0043E The XML could not be processed because an instance of the javax.xml.parsers.DocumentBuilder could not be created.

Explanation: See message.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEICL0001E The event access MBean failed to look up and create an instance of the event access enterprise bean.
JNDI name: *jndi_name*

Explanation: See message.

Programmer response: See the exception message for additional information about the cause of the problem.

CEICL0003E The MBean *mbean_name* cannot be found.

Explanation: The command requested the specified MBean, but that MBean is not registered with the MBean server.

Administrator Response: Verify that the Common Event Infrastructure service started correctly by looking in the application server log.

CEICL0004E A valid operation to perform was not provided.

Explanation: See message.

Administrator Response: See message.

CEICL0005E The JNDI lookup of the emitter factory failed.
Emitter factory JNDI name: *jndi_name*

Explanation: The specified emitter factory JNDI name is not correct, a configuration error has occurred, or a JNDI error has occurred.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEICL0006E The XML file could not be parsed.
XML file: *file_name*
Exception: *exception_class*
Exception message: *exception_message*

Explanation: The specified XML file contains an error and, therefore, it could not be parsed.

Programmer response: Correct the XML file and run the command again.

CEICL0007E The option *option_name* **is not valid, or the option was specified without a value.**

Explanation: See message.

Programmer response: Run the command again with the correct syntax.

CEICL0008E The value for specified option is not valid because either it is not a number, or the number is outside the valid range.
Option: *option_name*
Value: *value*
Valid range: *valid_range*

Explanation:

Programmer response: Run the command again with the correct syntax.

CEICL0009E The specified XML file could not be found.
File: *file_name*

Explanation: The specified file is not a valid URL or an XML file.

Programmer response: Run the command again, specifying a valid XML file name.

CEICL0010E The event catalog MBean failed to look up and create an instance of the event catalog enterprise bean.
JNDI name: *jndi_name*

Explanation: See message.

Programmer response: See the exception message for additional information about the cause of the problem.

CEICL0011E The required option *option_name* **was not specified.**

Explanation: See message.

Programmer response: Run the command again with the correct syntax.

CEICL0012E Either the option *first_option* **or the option** *second_option* **must be specified.**

Explanation: See message.

Programmer response: Run the command again with the correct syntax.

CEICL0013E The start_time *start_time* **must be an earlier time point than the end_time** *end_time***.**

Explanation: When specifying both start_time and end_time, the start_time must represent a time point before the time point represented by end_time. Because the parameters were not valid the command did not run.

Programmer response: Run the command again with the correct syntax.

CEICM0001E The event selector parser failed to parse the event selector because it did not contain a valid XPath expression.
Event selector: *event_selector*

Explanation: See message.

Administrator Response: Ensure that the event selector contains a valid XPath expression.

Event selectors are specifed by an XPath expression which must be of the form CommonBaseEvent[...]. If the event selector does not start with CommonBaseEvent or if it contains values that are not valid, the event selector is rejected.

CEICM0002E The event selector does not represent a valid event selector for filtering events because it did not contain a valid XPath expression.
Event selector: *event_selector*

Explanation: The event selector used by either event distribution or the notification helper must contain a valid XPath expression, it must select on CommonBaseEvent, and it must contain at least one predicate.

Administrator Response: Ensure that the event selector contains a valid XPath expression.

Event selectors are specifed by an XPath expression which must be of the form CommonBaseEvent[...]. If the event selector does not start with CommonBaseEvent or if it contains values that are not valid, the event selector is rejected.

CEICM0003E The event selector subexpression is not valid because the creationTime property must be compared to a valid xsd:datetime value.
Event selector: *event_selector*

Explanation: See message.

Administrator Response: Ensure that the event selector contains a valid xsd:datetime value in the format: CCYY-MM-DDThh:mm:ss.

CEICM0004E The event selector subexpression is not valid because it contains an event selector expression that is not supported.
Event selector: *event_selector*

Explanation: See message.

Administrator Response: Ensure that the event selector contains a valid XPath expression.

Event selectors are specifed by an XPath expression which must be of the form CommonBaseEvent[...]. If the event selector does not start with CommonBaseEvent or if it contains values that are not valid, the event selector is rejected.

CEICM0005E The extended data element parent location path could not be parsed because it does not contain a valid XPath expression.
Extended data element parent location path: *parent_location_path*

Explanation: See message.

Administrator Response: Ensure that the extended data element parent location path contains a valid XPath expression. The XPath expression must refer to one or more extended data element instances in the event or to the event itself.

Extended data element parent location paths are specified by an XPath expression that which must be of the form:

- Any valid Xpath expression beginning with CommonBaseEvent that refers to one or more extended data element instances in the event. For example: CommonBaseEvent/extendedDataElement[@name='myElement']
- If referring to the event itself, simply CommonBaseEvent

CEICM0006E The extended data element parent location path for the event with ID *global_id* **does not refer to one or more extended data element instances or to the event itself.**
Extended data element parent location path: *parent_location_path*

Explanation: See message.

Administrator Response: Ensure that the extended data element parent location path contains a valid XPath expression. The XPath expression must refer to one or more extended data element instances in the event or to the event itself.

Extended data element parent location paths are specified by an XPath expression that which must be of the form:

- Any valid Xpath expression beginning with CommonBaseEvent that refers to one or more extended data element instances in the event. For example: CommonBaseEvent/extendedDataElement[@name='myElement']
- If referring to the event itself, simply CommonBaseEvent

CEIC0001W The event group profile list contains two event group profiles configured with the same event group name.
Event group names: *event_group_names*

Explanation: Event group names must be unique. The first event group profile with the duplicated event group name will be overridden.

Administrator Response: Modify the event group profile list configuration so that each event group profile contains an event group name that is unique.

CEIC0002E The Common Event Infrastructure service failed to start because the Common Event Infrastructure resources failed to bind into JNDI.
Exception message: *binding_error*

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to ensure that the Common Event Infrastructure service is installed successfully. Also, see the exception message for additional information about the cause of the problem.

CEIC0003E The Common Event Infrastructure service failed to start because a Common Event Infrastructure object failed to bind into JNDI.
Object type: *binding_object*
Exception message: *binding_object_error*

Explanation: See message.

Programmer response: Consult the Common Event Infrastructure log files to ensure that installation of the Common Event Infrastructure service completed successfully. Also, see the exception message for additional information about the cause of the problem.

CEIC0004E The binding in JNDI could not be created for the object type: *binding_object*
Exception message: *reference_exception*

Explanation: See message.

Programmer response: Consult the Common Event Infrastructure log files to ensure that the installation of the Common Event Infrastructure application completed successfully. Also, ensure that the configuration settings are correct.

CEIC0005E The object to bind into JNDI of type *binding_object* **could not be created. Exception message:** *reference_exception*

Explanation: See message.

Programmer response: Consult the Common Event Infrastructure log files to ensure that the installation of the Common Event Infrastructure service completed successfully. Also, ensure that the configuration settings are correct.

CEIC0006I The Common Event Infrastructure service is started.

Explanation: No additional information is available for this message.

CEIC0007E An exception occurred when attempting to load the configuration information from the reference address. Exception message: *reference_exception*

Explanation: See message.

Programmer response: Ensure that the configuration information for the Common Event Infrastructure was bound correctly into the JNDI.

CEIC0008E The object found in JNDI was null.

Explanation: See message.

Programmer response: Ensure that the configuration information for the Common Event Infrastructure was bound correctly into the JNDI.

CEIC0009W The server-cei.xml file could not be located in the configuration. Exception message: *configuration_exception*

Explanation: See message.

Programmer response: Consult the Common Event Infrastructure log files to ensure that the installation of the Common Event Infrastructure service completed successfully.

CEIC0010W The configuration object was not the correct type. Expected EventInfrastructureService but found *configuration_type*

Explanation: Two services with similar names may have been installed.

Programmer response: Consult the Common Event Infrastructure log files to ensure that the installation of the Common Event Infrastructure service completed successfully.

CEIC0011E An error occurred trying to register the MBean *mbean_type***. Exception message:** *admin_exception*

Explanation: See message.

Programmer response: Consult the Common Event Infrastructure log files to ensure that the installation of the Common Event Infrastructure service completed successfully.

CEIC0012W The Common Event Infrastructure application with name *application_name* **was not present on the server.**

Explanation: When the Common Event Infrastructure service was enabled or disabled, an attempt was made to enable or disable the named enterprise application, but it could not be found. The names of the known Common Event Infrastructure applications are stored as custom properties of the Common Event Infrastructure service.

Programmer response: Adjust the name of the Common Event Infrastructure application or update the name listed in the Common Event Infrastructure service custom properties.

CEIC0013E An error ocurred when trying to update the Common Event Infrastructure application. Exception message: *configuration_exception*.

Explanation: When the Common Event Infrastructure service was enabled or disabled, an error occurred when an attempt was made to enable or disable the enterprise application. See message for details.

Programmer response: Consult the Common Event Infrastructure log files to ensure that the installation of the Common Event Infrastructure service and application completed successfully.

CEIDS0001E The specified relational database management system is not supported.
Database: *rdbms_name*
Version: *rdbms_version*

Explanation: See message.

Administrator Response: Reconfigure the common event infrastructure data source to use a supported relational database management system.

CEIDS0002E The relational database management system reported the following error.
Data source resource reference: *data_source*
SQL state: *sql_state*
Vendor code: *vendor_code*
Message: *message*

Explanation: The SQL state is the standard JDBC error code for the reported problem. The vendor code is the database vendor specific error code. The message parameter is the localized error message that is returned by the relational database management system.

Administrator Response: Refer to the appropriate relational database documentation for information about the SQL state, the vendor code, and the error message.

CEIDS0003E The mapping of a Common Base Event element or attribute to database tables and columns cannot be found.
Attribute: *attribute_name*

Explanation: The data store stores metadata information in its database. This metadata information specifies how Common Base Event elements and attributes map to tables and columns in the database. Either an error occurred during the installation of the event database or the metadata table information is corrupted.

Administrator Response: Install the event database again.

CEIDS0004I The data store is connecting to the specified relational database management system.
Database: *rdbms_name*
Version: *rdbms_version*

Explanation: No additional information is available for this message.

CEIDS0005E The maximum purge transaction size *purge_threshold* **has been exceeded. The purge operation has been stopped, the transaction has been rolled back, and no events have been purged.**

Explanation: An event purge operation attempted to delete a large number of events. Because this action can adversely affect system performance and event throughput, the purge operation was stopped.

Programmer response: To reduce the number of events to be purged in a single transaction, do one or all of the following actions:

- Add additional filters to the purge event selector expression.
- Set the maximum purge transaction size to a lower value.

CEIDS0006E The query threshold *query_threshold* **has been exceeded. The query operation has been stopped and no events have been returned.**

Explanation: An event query attempted to return a very large number of events. Because this action can adversely affect system performance and event throughput, the query operation was stopped.

Programmer response: To reduce the number of events to be returned, do one or more of the following actions:

- Add additional filters to the event selector expression for the query.
- Specify the maximum number of events to be returned.
- If you have sufficient system resources, increase the query threshold value.

CEIDS0007E An event with the global instance identifier, *global_id* **, already exists in the database. The event was not stored in the database.**
Existing event: *old_event*
New event: *new_event*

Explanation: Each event instance must have a unique global instance identifier.

Programmer response: Create the event with a unique global instance identifier.

CEIDS0008E The WebSphere Application Server connection pooling system returned a database connection that is no longer valid.
Data source resource reference: *data_source*

Explanation: See message.

Programmer response: Try the operation again.

CEIDS0009E A database connection could not be obtained from the connection pool after *maximum_retries* **attempts to obtain a connection.**
Data source resource reference: *data_source*

Explanation: The WebSphere Application Server connection pooling system was not able to return a database connection using the current configuration of the connection pool and the configured number of retries.

Administrator Response: Do one or more of the following actions:

- Reconfigure the data source, increasing the maximum number of connections. This option improves performance and event throughput.
- Reconfigure the data source, increasing the connection time out value.
- In the data store profile, increase the number of retries.

CEIDS0010I The maxCacheEntries configuration parameter must be greater than or equal to *minimum_cache_entries***. However,** *maximum_cache_entries* **was specified. The maxCacheEntries value has been set to** *new_cache_entries* **.**

Explanation: The maxCacheEntries configuration parameter specifies how many translated event selector expressions can be cached by the data store. Caching translated event selector expressions improves the performance of the event query and the event purge because event selector expressions in the cache do not need to be parsed and translated to SQL.

When the maxCacheEntries parameter is set to zero (0), no translated event selector expressions are cached. Therefore, each event selector expression specified in a call to the event query and the event purge methods must be parsed and translated to a valid SQL expression.

Administrator Response: In the data store profile, set the maxCacheEntries value to zero (0) or greater.

CEIDS0011E The data store cannot find the resource reference in JNDI.
Resource reference: *resource_reference_name*

Explanation: The data store uses resource references to obtain the data source that is used to obtain connections to the database and for the object that contains the configuration for the data store. The resource reference information

specified during installation is not valid. For example, a JNDI name that is not valid was specified for the resource reference.

If the specified resource reference name is java:comp/env/DefaultDataStoreProfileReference, the data store cannot access its configuration data. If the specified resource reference name is java:comp/env/EventDataSourceReference, the data store cannot access its data source. If the specified resource reference name is java:comp/env/ejb/DataStoreHelperEJB, the data store cannot access the data store helper enterprise java bean.

Administrator Response: In the WebSphere Administrative Console, reconfigure the specified resource reference to a valid JNDI resource.

CEIDS0013E The attribute or element *name* specified in the event selector expression *expression* is not supported.

Explanation: See message.

Programmer response: Specify an event selector expression that contains only attributes and elements that are supported.

CEIDS0014E The predicate expression specified in the event selector expression, *expression*, is not valid, and it could not be translated to a valid SQL expression.

Explanation: The most likely cause of this problem is specifying only a constant value as the predicate in an expression.

Programmer response: Specify an event selector expression with a valid predicate expression.

CEIDS0015E The *function* function used in the event selector expression, *expression*, has an argument that is not valid. The first argument of the function must either be a string constant or a location path that can be translated to a column in the event data store. The second argument of the function must be a string constant.
Argument: *argument*

Explanation: The first argument of the function must either be a string constant or a location path that can be translated to a column in the event data store.

The second argument of the function must be a string constant. The SQL equivalent to this function, the LIKE function, requires that the first argument be either a table column or a string constant and the second argument be a string constant.

Programmer response: Specify a string constant or a location path that maps to a table column in the event database for the first argument of the function used in the event selector expression. Specify a string constant for the second argument of the function that is used in the event selector expression.

This example shows a valid event selector expression using the starts-with function.

CommonBaseEvent[starts-with(@msg,'My message')]

CEIDS0019E No arguments were specified for a function used in the event selector expression.
Event selector expression: *expression*
Function: *function*

Explanation: You must specify one argument for this function.

Programmer response: Specify one argument for the function.

CEIDS0020E The function used in the event selector expression is not supported.
Event selector expression: *expression*
Function: *function*

Explanation: See message.

Administrator Response: Specify an event selector expression that uses a supported function.

CEIDS0021E The XPath expression type specified in the event selector expression is not supported.
Expression type: *expression_type*
Event selector expression: *expression*

Explanation: These are the expression types that are supported for this release:

- Constant

```
1,2,'thisString'
```

- Location Path

```
CommonBaseEvent[@severity=10]
```

- Operation

```
@severity+10=20
```

- Function

```
contains(@msg)
```

Programmer response: Specify an XPath expression that uses a supported expression type.

CEIDS0022E The event selector expression *expression* **is not valid. The event selector parsing tool returned the following message:** *message*

Explanation: These are the are possible reasons for this exception:

- A predicate bracket is not closed. For example, CommonBaseEvent[@severity='10 is missing a closing bracket, and it should be specified as CommonBaseEvent[@severity='10'].
- A function has the wrong number of arguments. For example, the starts-with and the contains function require two arguments.
- A quote on a string constant is not closed. For example, CommonBaseEvent[@severity='10] is missing a closing quote, and it should be specified as CommonBaseEvent[@severity='10'].
- An open parenthesis is not closed. For example, CommonBaseEvent[(@severity+10=20] is missing a closing parentheses, and it should be specified as CommonBaseEvent[(@severity+10)=20].
- A string constant that is not contained by quotes. For example, CommonBaseEvent[@creationTime=2003-12-31T12:00:00Z] is missing quotes around the date time sting, and it should be specified as CommonBaseEvent[@creationTime='2003-12-31T12:00:00Z'].

Programmer response: Specify a valid event selector expression.

CEIDS0023E The event selector expression *expression* **is not valid. The CommonBaseEvent element does not have an associated predicate.**

Explanation: See message.

Programmer response: Specify an event selector expression with one predicate associated with the CommonBaseEvent element.

For example, the following event selector expression returns a SQL expression that selects all events from the event database. For example, CommonBaseEvent[@globalInstanceId]

CEIDS0024E The *function* **function used in the event selector expression,** *expression***, is specified with an incorrect number of arguments. The function should be specified with** *number_of_arguments* **arguments.**

Explanation: See message.

Programmer response: Specify the function with a valid number of arguments.

CEIDS0025E The arithmetic operation contains an operand that is not a numeric value.
Operation: *operation*

Explanation: Only numeric operands are supported for the following arithmetic operators: +, -, *, div, mod, <, >, <=, >=

Programmer response: Specify two numeric values for this operation.

The following example shows a valid event selector expression using the addition operation.

CommonBaseEvent[@severity = 10 + 1]

CEIDS0026E The operation contains operands that are not compatible.
Operation: *operation*

Explanation: The most likely cause of this problem is using a numeric value for one operand and a string value that cannot be converted to a numeric value for the other operand.

Programmer response: Specify operands with compatible types.

The following example shows a valid event selector expression using the equality operation.

CommonBaseEvent[@severity = '10']

Because the string value 10 can be converted to the numeric value 10, the operation is valid and the types are compatible.

CEIDS0027E An error occurred when writing the CommonBaseEvent any element to the database. The operation was stopped.
Event: *event*

Explanation: See message.

Administrator Response: See message.

CEIDS0028E An error occurred when reading the CommonBaseEvent any element from the database. The operation was stopped.
Global instance identifier: *global_id*

Explanation: See message.

Administrator Response: See message.

CEIDS0029E An error occurred when writing a CommonBaseEvent extended data element that has a hexBinary data type to the database. The operation was stopped.
Event: *event*

Explanation: See message.

Administrator Response: See message.

CEIDS0030E An error occurred when reading a CommonBaseEvent extended data element that has a hexBinary data type from the database. The operation was stopped.
Global instance identifier: *global_id*

Explanation: See message.

Administrator Response: See message.

CEIDS0031W The configuration value for connection retries, *connection_retries_old_value***, is not valid. The value has been changed to** *connection_retries_new_value***.**

Explanation: The connection retries configuration setting specifies the number of times the data store attempts to obtain a database connection from the connection pool. If a connection cannot be obtained in the specified number of retries, the data store throws an exception. A value of 0 specifies that the data store does not retry the operation.

Administrator Response: In the data store profile, set the connection retries value to 0 or greater.

CEIDS0032W The configuration value for query threshold, *query_threshold_old_value*, is not valid. The value has been changed to *query_threshold_new_value*.

Explanation: The query threshold configuration setting specifies the maximum number of events that can be returned by the queryEvents methods. This value must be 1 or greater.

Administrator Response: In the data store profile, set the query threshold value to 1 or greater.

CEIDS0033W The maximum purge transaction size, *purge_threshold_old_value*, is not valid. The value has been changed to *purge_threshold_new_value*.

Explanation: The maximum purge transaction size configuration setting specifies the maximum number of events that can be deleted in a single purgeEvents call. This value must be 1 or greater.

Administrator Response: In the data store profile, set the maximum purge transaction size to 1 or greater.

CEIDS0034E The operator used in the the event selector expression operation *operation* is not supported.

Explanation: These are the supported event selector expression operators:

- +,-,*, div, mod
- =, !=, <, >, <=, >=
- and, or

Programmer response: Specify an event selector expression operation that uses a supported operator.

CEIDS0035E The implementation class that supports the configured relational database system cannot be loaded.
Implementation class name: *class_name*
Relational database name: *rdbms_name*
Database version: *rdbms_version*

Explanation: See message.

Administrator Response: Ensure that the specified implementation class is in the CLASSPATH.

CEIDS0036E The relational database management system reported the following error.
Data source resource reference: *data_source*
Database product: *rdbms_name*
Database version: *rdbms_version*
SQL state: *sql_state*
Vendor code: *vendor_code*
Message: *message*

Explanation: The SQL state is the standard JDBC error code for the reported problem. The vendor code is the database vendor specific error code. The message parameter is the localized error message that is returned by the relational database management system.

Administrator Response: Refer to the appropriate relational database documentation for information about the SQL state, the vendor code, and the error message.

CEIDS0037E The values element is referenced in the specified event selector expression, *expression*, without specifying a corresponding extended data element type.

Explanation: When the values element is specified in an event selector expression, a corresponding event selector type must also be specified.

Programmer response: Identify the event selector type of the values element specified in the event selector expression using the and operator.

A valid event selector expression referencing the values element looks similar to the following example.

CommonBaseEvent[extendedDataElements[values=1 and @type='intArray']]

The and operation on the values element and the type attribute expressions must be located within a predicate expression of an extendedDataElements element.

CEIDS0038E The XPath pattern-matching character *character* that is used in the specified event selector expression, *expression*, is not supported.

Explanation: The XPath pattern-matching characters are not supported.

Programmer response: Specify the XPath expression without using a XPath pattern-matching character.

CEIDS0039E The type attribute used in the event selector expression, *expression*, conflicts with another type attribute that has already been specified.

Explanation: Extended data element type attributes cannot be specified in conjunction with one another using the and operator.

Programmer response: Specify the event selector expression without using two extended data element type attributes that are conflicting.

For example, the following event selector expression is not valid because the type attributes are conflicting.

CommonBaseEvent[extendedDataElements[values=1 and @type='intArray' and @type='float']]

This expression is correctly rewritten as follows:

CommonBaseEvent[extendedDataElements[values=1 and @type='intArray'] and extendedDataElements[@type='float']]

CEIDS0040E The event selector expression, *expression*, was specified with stacked predicates. Stacked predicates are not supported.

Explanation: See message.

Programmer response: Specify the event selector expression without using stacked predicates.

For example, the following event selector expression is not valid because there are multiple predicates on the CommonBaseEvent element.

CommonBaseEvent[@severity=10][@priority=10]

This expression is correctly rewritten as follows:

CommonBaseEvent[@severity=10 and @priority=10]

CEIDS0041E The parameter value *function_parameter* specified for the function *xpath_function* in the event selector expression *expression* does not match any known extended data element types.

Explanation: See message.

Programmer response: Specify a value for the function that matches at least one extended data element type.

For example, the following event selector expression is valid because the starts-with function matches the int and intArray extended data element types.

CommonBaseEvent[extendedDataElements[starts-with(@type,'int')]]

CEIDS0042I The current bucket is being changed from bucket *old_bucket* to bucket *new_bucket*.

Explanation: No additional information is available for this message.

CEIDS0043E The value *property_value* for the property *property_name* cannot be converted to an integer. The default value of *default_value* is used.

Explanation: The specified property value is being read from the cei_t_properties table. The value stored in the property_value column, which is a string, cannot be converted to an integer value. The contents of the cei_t_properties table can be reset to the default values by running the ins_metadata script.

CEIDS0044E The value *property_value* for the property *property_name* is greater than the maximum allowed value of *maximum_value*. The *default_value* default value is used.

Explanation: The specified property value is being read from the cei_t_properties table. The value stored in the property_value column, which is a string, can be converted to an integer value, but the value is larger than the maximum allowed value. The contents of the cei_t_properties table can be reset to the default values by running the ins_metadata script.

CEIDS0045E The value *property_value* for the property *property_name* is less than the minimum allowed value of *minimum_value*. The default value *default_value* is used.

Explanation: The specified property value is being read from the cei_t_properties table. The value stored in the property_value column, which is a string, is converted to an integer value, but the value is smaller than the minimum allowed value. The contents of the cei_t_properties table are reset to the default values by running the ins_metadata script.

CEIDS0046I The property *property_name* cannot be found. The default value *default_value* is used.

Explanation: No additional information is available for this message.

CEIDS0047I A fast purge is in progress. Only the active bucket is accessed. The active bucket is *active_bucket*.

Explanation: No additional information is available for this message.

CEIDS0048I Querying the database for the event with the *global_id* ID in the *search_bucket* bucket. The bucket is bypassed because the fast purge is in progress.

Explanation: No additional information is available for this message.

CEIDS0049E The severity attribute for the event with the global instance ID of *global_id* cannot be set to *new_severity_value* because it is already set to *old_severity_value*.

Explanation: The CommonBaseEvent schema does not allow the severity attribute to be changed once it is set. An event update request was submitted that attempted to set the severity attribute on an event that already had a severity value. The calling application should be changed so that it does not attempt to set the severity attribute on the specified event.

CEIDS0050I Events cannot be inserted into the *bucket_number* bucket because it is not active. The events are inserted into the active bucket. The active bucket is *active_bucket*.

Explanation: No additional information is available for this message.

CEIDS0051E The event with the global instance ID of *global_id* cannot be updated because it does not exist in the database.

Explanation: An event update request specified an event that no longer exists in the database. This might occur if an event purge was performed.

CEIDS0052E The current active bucket cannot be changed from bucket *active_bucket* to bucket *new_bucket* because bucket *new_bucket* still contains event data.

Explanation: All of the events in the inactive bucket must be purged before the active bucket can be switched. Use the fast purge to purge all the events in the inactive bucket.

CEIDS0053E The event database is configured to only use one bucket. The current active bucket cannot be changed.

Explanation: The event database only contains one set of tables, a bucket, to store event data. Therefore, there is only an active bucket. The eventbucket.jacl script cannot be used to change the active bucket.

CEIDS0054I Number of buckets: *number_buckets*
Current bucket: *active_bucket*
Bucket check interval (seconds): *bucket_check_interval*

Explanation: No additional information is available for this message.

CEIDS0055E The database schema is not compatible with the default data store. The database schema version is *actual_schema_version*. **the required version is** *required_schema_version*.

Explanation: See message.

CEIDS0056I The database schema version is *actual_schema_version*.

Explanation: No additional information is available for this message.

CEIDS0057E The parent location path *parent_location_xpath* **in the event update request for event with global instance ID** *global_id* **does not refer to any extended data element instances in the event. The event update request cannot be performed.**

Explanation: The Xpath expression for the parent extended data element is a valid Xpath expression and it does not refer to any existing extended data element instances. The calling application needs to be changed so that the Xpath expression refers to at least one extended data element in the event.

CEIEM0003E The filter factory failed to create a filter instance for the emitter.
Filter factory: *filter_factory*
Exception: *exception_class*
Exception message: *exception_message*

Explanation: Emitter does not get initialized if the filter failed to be created.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEIEM0006E The event filter returned an exception when filtering an event.
Event: *event*
Filter: *filter*
FilterFactory: *filter_factory*
Exception: *exception_class*
Exception message: *exception_message*

Explanation: No events are sent when this exception is thrown.

Programmer response: See the exception message for additional information about the cause of the problem.

CEIEM0007E The emitter failed to obtain the filter metadata from the filter.
Filter: *filter*
Exception: *exception_class*
Exception message: *exception_message*

Explanation: An exception occurred while retrieving the metadata information from this filter.

Programmer response: See the exception message for additional information about the cause of the problem.

CEIEM0008E The filter failed to close. Resources held by the filter instance might not have been released.
Filter: *filter*
Exception: *exception_class*
Exception message: *exception_message*

Explanation: An exception occurred while closing the filter.

Programmer response: See the exception message for additional information about the cause of the problem.

CEIEM0015E The emitter does not support the specified synchronization mode: *mode*.
Expected values: *filter*

Explanation: The synchronization mode selected for an emitter operation is valid, but it is not supported by this emitter. No events are sent when this exception is thrown.

Administrator Response: Specify a supported synchronization mode in the configuration.

CEIEM0016E The emitter does not support the specified transaction mode: *mode*.
Expected values: *filter*

Explanation: The transaction mode selected for the emitter operation is not valid or it is not supported by this emitter. No events are sent when this exception is thrown.

Administrator Response: Specify a supported transaction mode in your configuration.

CEIEM0020E The emitter failed to initialize because the JNDI lookup on the event bus home name failed.
JNDI name: *JNDI_name*
Context: *nameInNameSpaceOfContext*
Exception (if any): *exception_class*

Explanation: See message.

Administrator Response: Ensure that the JNDI name of the event bus specified in the synchronous transmission profile is correct.

CEIEM0023E The emitter failed to initialize because of a failure in the synchronous transmission sender. An exception occurred during initialization of the event bus sender.
Exception: *exception_class*

Explanation: See message.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEIEM0024E An error occurred when closing the emitter. The synchronous transmission profile could not release resources. EventBus Interface: *localOrRemote*
Exception: *exception_class*

Explanation: A call to the EventBus.ejbRemove() method or the EventBusLocal.ejbRemove() method threw an exception.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEIEM0025E The emitter failed to send the events to the event server. The *localOrRemote* **event bus enterprise bean on the event server failed during event processing.**
Transaction mode: *transaction_mode*
Exception: *exception_class*
Events: *event_list*

Explanation: The server is not available, is not configured correctly, or an unexpected runtime error has occurred. No events are sent when this exception is thrown.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEIEM0026E The emitter did not send the event to the event server because the content handler threw an exception.
Event: *event*
Exception: *exception_class*

Explanation: See message.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEIEM0027E The emitter did not send the event to the event server because the common base event is not valid.
Event: *event*
Exception: *exception_class*

Explanation: The validate() method on the common base event threw an exception indicating that the event is not valid.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEIEM0031E The JNDI lookup of a JMS queue connection factory failed because the JNDI name defined in the emitter profile is not bound.
Context: *Context_name*
JNDI name: *JNDI_name*

Explanation: See message.

Administrator Response: Ensure that the configuration of the JMS queue connection factory asynchronous transmission profile for this event emitter is correct.

CEIEM0032E The JNDI lookup of a JMS queue connection factory failed because the JNDI name does not resolve to an instance of javax.jms.QueueConnectionFactory.
Context: *Context_name*
JNDI name: *JNDI_name*

Explanation: See message.

Administrator Response: Ensure that the configuration of the JMS queue connection factory asynchronous transmission profile for this event emitter is correct.

CEIEM0034E The JNDI lookup of a JMS queue failed because the JNDI name defined in the emitter profile is not bound in the JNDI.
Context: *Context_name*
JNDI name: *JNDI_name*

Explanation: See message.

Administrator Response: Ensure that the configuration of the JMS queue and asynchronous transmission profile for this event emitter is correct.

CEIEM0035E The JNDI lookup of a JMS queue failed because the JNDI name does not resolve to an instance of javax.jms.Queue.
Context: *Context_name*
JNDI name: *JNDI_name*

Explanation: See message.

Administrator Response: Ensure that the configuration of the JMS queue and asynchronous transmission profile for this event emitter is correct.

CEIEM0037E A failure occurred when attempting to suspend or resume the current unit of work.
Exception: *exception_class*

Explanation: See message.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEIEM0038E An error occurred when attempting to initialize a JMS session.
JMS Connection: *connection*
Session type: *session_type*
Acknowledge type: *session_type*
Exception: *exception_class*

Explanation: See message.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEIEM0039E An error occurred when attempting to initialize a JMS queue sender.
JMS session: *session*
JMS queue: *queue*
Exception: *exception_class*

Explanation: See message.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEIEM0040E An error occurred when attempting to close a JMS connection.
JMS connection: *connection*
Exception: *exception_class*

Explanation: See message.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEIEM0041E The emitter failed to send the events to the event server because of a JMS failure.
Transaction mode: *transaction_mode*
Exception: *exception_class*
Events: *event_list*

Explanation: Either the server is not available, is not configured correctly, or an unexpected runtime error has occurred. No events are sent when this exception is thrown

Administrator Response: See the exception message for additional information about the cause of the problem.

CEIEM0042E An error occurred when attempting to open a JMS connection.
JMS connection factory: *connection_factory*
Exception: *exception_class*

Explanation: See message.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEIES0003E The event server failed to initialize an event group because the specified JMS destination could not be found in JNDI.
Event group name: *event_group_name*
JMS connection factory JNDI name: *destination_factory_jndi_name*
JMS destination JNDI name: *destination_jndi_name*
Context: *context_name*

Explanation: The event server failed to initialize an event group for distribution of events, and the event server does not use the event group when distributing event notifications. This failure stops the event server from processing events. The event server uses all valid event groups when distributing event notifications.

Administrator Response: Ensure that the JNDI names are correct for the following resources as appropriate:

- Topic connection factory
- Topic
- Queue connection factory
- Queue

CEIES0004E No event notifications were sent because the event server could not connect to the JMS destination.
Event group name: *event_group_name*
JMS connection factory JNDI name: *destination_factory_jndi_name*
JMS destination JNDI name: *destination_jndi_name*

Explanation: Before sending event notifications, the event server tries to connect to all JMS destinations for any event groups that an event matches. If during this processing one of the connection attempts fail, the event notification is not sent to any of the matching event groups and the transaction is marked for rollback.

Administrator Response: Ensure that the event group definition contains a valid JMS destination. Also, ensure that the JMS provider is working properly.

CEIES0005E The event server failed to create a JMS message for the event notification being sent to the JMS destination.
Event group name: *event_group_name*
Events: *event_instances*
JMS connection factory JNDI name: *destination_factory_jndi_name*
JMS destination JNDI name: *destination_jndi_name*

Explanation: The events are not processed, and the current transaction is marked for rollback.

Administrator Response: Ensure that the event group definition contains a valid JMS destination.

CEIES0006E The event server failed to publish the event notification to the JMS destination.
Event group name: *event_group_name*
Events: *event_instances*
JMS Message: *event_instance*
JMS connection factory JNDI name: *destination_factory_jndi_name*
JMS destination JNDI name: *destination_jndi_name*

Explanation: The events are not processed, and the current transaction is marked for rollback.

Administrator Response: Ensure that the event group definition contains a valid JMS destination.

CEIES0007W The event server failed to disconnect from the JMS destination.
Event group name: *event_group_name*
JMS connection factory JNDI name: *destination_factory_jndi_name*
JMS destination JNDI name: *destination_jndi_name*
Exception message: *exception_message*

Explanation: No recovery is done because the JMS destination is closed.

Administrator Response: If this is a recurring problem, ensure that the JMS provider is working properly.

CEIES0008E The event server failed to store any events in the data store because the global instance ID in an event already exists in the data store.
Exception message: *exception_message*
Events: *event_list*

Explanation: The global instance ID uniquely identifies the event in the data store. It is assumed that one of the passed events is a duplicate of an event that already exists in the data store. Therefore, the events are not processed, and the current transaction is marked for rollback.

Administrator Response: If this is a recurring problem, inspect the source of the event to determine if the source is creating duplicate global instance IDs.

CEIES0009E The event server failed to store any events in the data store because the data store could not be found using the specified JNDI name.
JNDI name: *data_store_jndi_name*
Exception message: *exception_message*
Events: *event_list*

Explanation: The events are not processed, and the current transaction is marked for rollback.

Administrator Response: Ensure that the data store exists at the specified JNDI name. Also, see the exception message for additional information about the cause of the problem.

CEIES0010E The event server failed to store any events in the data store referenced by the JNDI name because the configured data store threw an exception.
JNDI name: *data_store_jndi_name*
Exception message: *exception_message*
Events: *event_list*

Explanation: The events are not processed, and the current transaction is marked for rollback.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEIES0011E The event server failed to distribute an event notification.
Exception message: *exception_message*
Events: *event_list*

Explanation: The events are not processed, and the current transaction is marked for rollback.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEIES0012E The JMS message did not contain a value in the JMS message type property.
JMS message: *message*

Explanation: See message.

Administrator Response: If this is a recurring problem, inspect the source of the event to determine if the source is creating JMS messages that are not valid.

CEIES0013E The JMS message did not contain an event.
JMS message: *message*

Explanation: See message.

Administrator Response: If this is a recurring problem, inspect the source of the event to determine if the source is creating JMS messages that are not valid.

CEIES0014W The JMS message was not processed because the value in the JMS message type property is not valid.
JMS message: *message*

Explanation: See message.

Administrator Response: If this is a recurring problem, inspect the source of the event to determine if the source is creating JMS messages that are not valid.

CEIES0015E The creation of a JMS message for an event batch failed.
Events: *event_list*

Explanation: The JMS provider threw an exception when the event server tried to create a JMS message to contain an event notification. Because the event server was unable to send the event notification for a single event all notifications will be stopped. The global transaction is marked for roll back, so if persistence is enabled the events are not stored in the persistent store.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEIES0016E The notification helper failed to set the event selector because the event selector is not in the correct format.
Event selector: *event_selector*
Exception message: *exception_message*

Explanation: See message.

Administrator Response: Ensure that the calling application is passing a valid event selector.

Event selectors are specifed by an XPath expression which must be of the form CommonBaseEvent[...]. If the event selector does not start with CommonBaseEvent or if it contains values that are not valid, the event selector is rejected.

CEIES0018E The event server failed to process an event batch because the event server does not support the event version specified in the event.
Global Instance Id: *global_instance_ID*
Event version: *event_version*
Event server version: *event_server_version*
Events: *event_list*

Explanation: The events are not processed, and the current transaction is marked for rollback.

Administrator Response: If this is a recurring problem, inspect the source of the event to determine the cause of the problem.

CEIES0019E The *method* **event access method failed because the data store component is not enabled.**

Explanation: See message.

Administrator Response: If the data store should be enabled, modify the event server profile by selecting **Enable Data Store** .

CEIES0021E The configured data store could not be found using the specified JNDI name.
JNDI name: *jndi_name*
Context: *context*

Explanation: See message.

Administrator Response: Ensure that the correct JNDI name is configured for the data store. Also, ensure that the data store is deployed to the specified JNDI name.

CEIES0023E The event server failed to create an instance of the configured data store because the data store event version and the event server event version are not compatible.
Data store version: *datastore_cbe_version*
Event server version: *eventserver_cbe_version*

Explanation: See message.

Administrator Response: The event server must use a data store that supports the same event version.

CEIES0024E The event server failed to create an instance of the configured data store because the component metadata could not be retrieved from the data store.

Explanation: See message.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEIES0025E The event server failed to create an instance of the configured data store.

Explanation: See message.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEIES0026E The configuration of the event server is not valid because the data store is enabled but the data store JNDI name is null. It is not valid to enable the data store and to have a null data store JNDI name.
Event server JNDI name: *jndi_name*
Event server configuration object: *configuration_object*

Explanation: See message.

Administrator Response: Update the configuration of the event server profile to either disable the data store or to provide a valid data store JNDI name.

CEIES0027E The configuration object for the event server at the specified JNDI name could not be found.
JNDI name: *jndi_name*

Explanation: See message.

Administrator Response: Verify that the Common Event Infrastructure service started. Also, ensure that the failing enterprise bean has a correct EJB reference in the deployment descriptor to the event server configuration object.

CEIES0028E The configuration object for the event server could not be found when accessing the configuration of event group list at the specified JNDI name.
JNDI name: *jndi_name*

Explanation: See message.

Administrator Response: Verify that the Common Event Infrastructure service started. Also, ensure that the failing enterprise bean has a correct EJB reference in the deployment descriptor to the event server configuration object.

CEIES0029E The reference obtained from the specified JNDI name is not a reference to a notification helper factory.
JNDI name: *jndi_name*

Explanation: An application attempted to look up a notification helper factory from a JNDI namespace, but the object found in the specified location was not a reference to a notification helper factory.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEIES0030W The event server failed to decode the JMS message because the message does not contain a valid message type.
JMS message: *message*
Message type: *message_type*

Explanation: The message is not processed, and the current transaction is marked for rollback causing the message to be returned to the JMS provider for handling.

Administrator Response: If this is a recurring problem, check the source of the message to determine the cause of the problem.

CEIES0031E The event server failed to retrieve an event from the JMS message.
JMS message: *message*
Exception message: *exception_message*

Explanation: The event is not processed, and the current transaction is marked for rollback.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEIES0032E The event server failed to process a JMS message.
Exception message: *exception_message*
JMS Message: *message*

Explanation: The JMS message is not processed, and the current transaction is marked for rollback.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEIES0033E The event server failed to process the JMS message.
JMS message: *message*
Exception message: *exception_message*

Explanation: The message is not processed, and the current transaction is marked for rollback.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEIES0034E The event server failed to create an instance of the event bus because it could not be found using the specified JNDI name.
JNDI name: *jndi_name*
Context: *context*

Explanation: See message.

Administrator Response: Ensure that the event bus message driven bean has a correct EJB reference in the deployment descriptor to the event bus.

CEIES0035E The event server failed to create an instance of the event bus.

Explanation: See message.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEIES0036E The event server failed to create an instance of the event bus. The event was not persisted, and it was not published to any message consumers.
Event: *event_instance*
Exception message: *exception_message*

Explanation: The event is not processed, and the current transaction is marked for rollback.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEIES0042E The notification helper failed to retrieve the JMS destinations for the specified event group because the event group configuration could not be found in JNDI.
Event group: *event_group*
JNDI name: *jndi_name*

Explanation: See message.

Administrator Response: Ensure that the event group is configured correctly.

CEIES0044E The event server failed to create an instance of event distribution because it could not be found using the specified JNDI name.
JNDI name: *jndi_name*
Context: *context*

Explanation: See message.

Administrator Response: Ensure that the event bus has a correct EJB reference in the deployment descriptor to the event distribution.

CEIES0045E The event server failed to create an instance of event distribution.

Explanation: See message.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEIES0048E The event group is not defined in the event group list that the event server instance is using.
Event group: *event_group_name*

Explanation: See message.

Administrator Response: Ensure that the event group is defined for this instance of the event server. Also, restart the application server that is running the instance of the event server to incorporate the latest configuration.

CEIES0050E The event server failed to initialize an event group because the event selector for the event group is not in the correct format.
Event group name: *event_group_name*
Event selector: *event_selector*
Exception message: *exception_message*

Explanation: The event server continues to operate normally, but the event server does not send event notifications to the specified event group.

Administrator Response: Event selectors are specifed by an XPath expression which must be of the form CommonBaseEvent[...]. If the event selector does not start with CommonBaseEvent or if it contains values that are not valid, the event selector is rejected.

Correct the event selector for the specified event group to match the CommonBaseEvent[...] format. Also, restart the application server that is running the instance of the event server to incorporate the latest configuration.

CEIES0051E The event server received an exception when initializing an event group.
Event group name: *event_group_name*
Exception message: *exception_message*

Explanation: The event server continues to operate normally, but the event server does not send event notifications to the specified event group.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEIES0052E The event server failed to publish the event notification to a JMS destination in the event group.
Event group name: *event_group_name*
Event: *event_instance*
Exception message: *exception_message*

Explanation: The event server attempts to publish the event notification to other JMS destinations.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEIES0053E The event server failed to retrieve a needed configuration object.
Configuration object class: *configuration_classname*

Explanation: See message.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEIES0054E The notification helper received an exception while looking up the event server in JNDI.

Explanation: See message.

Administrator Response: Verify that the Common Event Infrastructure service started. Also, see the exception message for additional information about the cause of the problem.

CEIES0056E The notification helper failed to locate the event bus in JNDI.

Explanation: See message.

Administrator Response: Verify that the Common Event Infrastructure service started.

CEIES0057E The data store returned more than the requested number of events for an event query.
Requested number of events: *maximum_number*
Returned number of events: *number_returned*
JNDI name: *data_store_jndi_name*

Explanation: See message.

Administrator Response: Contact the provider of the data store to correct the error.

CEIES0058E The eventExists event access method failed.
Event group: *event_group*
Event selector: *event_selector*
Exception message: *exception_message*

Explanation: See message.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEIES0059E The purgeEvents event access method failed.
Event group: *event_group*
Event selector: *event_selector*
Transaction size: *transaction_size*
Number events purged: *number_events_purged*

Explanation: See message.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEIES0060E The queryEventByGlobalInstanceId event access method failed.
Global instance ID: *global_instance_id*
Exception message: *exception_message*

Explanation: See message.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEIES0061E The queryEventsByAssociation event access method failed.
Association Type: *association_type*
Global instance ID: *global_instance_id*
Exception message: *exception_message*

Explanation: See message.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEIES0062E The queryEventsByEventGroup event access method failed.
Event group: *event_group*
Event selector: *event_selector*
Ascending Order: *ascending_order*
Exception message: *exception_message*

Explanation: See message.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEIES0063E The event access method queryEventsByEventGroup failed.
Event group: *event_group*
Event selector: *event_selector*
Ascending Order: *ascending_order*
Maximum Number: *maximum_number*
Exception message: *exception_message*

Explanation: See message.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEIES0064E The notification helper failed to read the event group configuration from the event server.
Event group: *event_group*

Explanation: See message.

Administrator Response: Verify that the Common Event Infrastructure service started. See the exception message for additional information about the cause of the problem.

CEIES0065E The notification helper failed to read an event from the JMS message because the JMS message type is not known.
JMS message: *jms_message*

Explanation: See message.

Administrator Response: The message is not processed by the notification helper because the notification helper cannot determine if the message contains an event.

CEIES0066E The event server failed to initialize an event group. The specified JNDI name does not point to an instance of the correct JMS class.
Event group name: *event_group_name*
JNDI name: *jndi_name*
JMS class: *jms_class*

Explanation: The event server failed to initialize an event group for distribution of events, and the event server does not use the event group when distributing event notifications. This failure does not stop the event server from processing events. The event server uses all valid event groups when distributing event notifications.

Administrator Response: Ensure that the JNDI names are correct for the following resources:

- Topic connection factory
- Topic
- Queue connection factory
- Queue

CEIES0067E The event update failed because the configured data store does not support event updates.
Data Store JNDI name: *jndi_name*

Explanation: See message.

Administrator Response: Upgrade the data store to support event updates.

CEIES0068E The updateEvents event access method failed.
Event Change Requests: *event_change_requests*
Exception message: *exception_message*

Explanation: See message.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEIES0070W A profile property was specified but did not have the expected type. The property was ignored.
Profile property name: *property_name*
Specified type: *specified_type*
Expected type: *expected_type*

Explanation: See message.

Administrator Response: Specify the property with the correct type.

CEIEF0001E The default filter factory cannot create any filter instances.

Explanation: This error indicates an internal error.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEIEF0002E The event cannot be processed by the filter .
Event: *event*
Filter: *filter*

Explanation: See message.

Administrator Response: Ensure that the filter syntax is valid and that all the specified fields are defined. See the exception message for additional information.

CEIEF0003E A filter rule could not be removed from the default filter plug-in because the filter rule *rule_ID* **was not found.**

Explanation: This error indicates a problem with the emitter.

Administrator Response: See the exception message for additional information about the cause of the problem.

CEIEF0004E A filter rule *rule_ID* could not be installed into the default filter plug-in.

Explanation: See message.

Administrator Response: Ensure that the filter specification for the default filter is valid.

CEIIN0001E The directory *directory_name* failed to be created.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0006E The file *file_name* was not found.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0007E The Common Event Infrastructure database and data source scripts for *database_type* failed to be created.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0008E Creation of the event database on *database_type* for the Common Event Infrastructure application failed.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0009E Creation of the data source *data_source* for the Common Event Infrastructure failed.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0016E The response file *file_name* is not valid because it does not contain the required parameters, or it contains values that are not valid.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0017E A required parameter is missing from the response file *response_file*.
Required parameter: *parameter_name*

Explanation: See message.

Programmer response: Ensure that the response file contains the specified parameter.

CEIIN0018I The Common Event Infrastructure configuration files were uninstalled successfully.

Explanation: No additional information is available for this message.

CEIIN0019E The directory *directory_name* was not found.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0020E The DB_TYPE parameter is not specified in the response file *file_name***. Valid values are:** *valid_values*

Explanation: See message.

Administrator Response: Ensure that the DB_TYPE parameter is specified in the response file.

CEIIN0021E The DB_TYPE parameter in the response file *file_name***, contains the value** *value***, which is not valid. Valid values are:** *valid_values*

Explanation: See message.

Administrator Response: Ensure that the DB_TYPE parameter that is specified in the response file contains a valid value.

CEIIN0022E The *page_cache_size* **parameter in the response file** *file_name* **contains the value** *page_cache_size_value***, which is not valid.**

Explanation: The PAGE_CACHE_SIZE parameter specifies the number of memory pages to use to cache data.

Administrator Response: Ensure that the PAGE_CACHE_SIZE parameter contains a numeric value. For information about setting the PAGE_CACHE_SIZE parameter, refer to the information about the db2j.storage.pageCacheSize parameter in the Tuning Cloudscape manual.

CEIIN0023E The installation of the Common Event Infrastructure application completed, but an error occurred registering the product.

Explanation: The request completed, but the installer was not able to update the Common Event Infrastructure product version file.

Administrator Response: Determine why the product version file could not be updated, correct the problem, then use the Common Event Infrastructure APIs to register the product.

CEIIN0024E The parameter *parameter_name* **in the response file** *file_name* **contains the value** *value***, which is not valid. Valid values are:** *valid_values***.**

Explanation: See message.

Administrator Response: Change the value of the specified parameter to a valid value.

CEIIN0025E The Cloudscape database already exists in the specified directory path.
Database name: *rdbms_name*
Directory path: *path*

Explanation: See message.

Administrator Response: Remove the database before attempting to create a database with the same name.

CEIIN0026I The enterprise application must be deployed to use the Common Event Infrastructure server.
Application: *application_name*

Explanation: See message.

Administrator Response: Deploy the specified application manually using the wsadmin tool.

CEIIN0027E Creation of the Common Event Infrastructure database failed.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0028E When running the Common Event Infrastructure installer from the command line, the correct command syntax was not provided.
Valid arguments: *valid_arguments*
Specified arguments: *specified_arguments*

Explanation: See message.

Programmer response: Provide the correct command syntax.

CEIIN0029E The response file *file_name* **does not exist.**

Explanation: See message.

Administrator Response: Ensure that you specify the correct response file name and location.

CEIIN0030I The Common Event Infrastructure server is not installed.

Explanation: No additional information is available for this message.

CEIIN0031E The directory *directory_name* **is not a valid WAS_HOME directory.**

Explanation: A WebSphere Application Server installation is not found in the specified directory.

Administrator Response: Ensure that you have specified the correct location for the WebSphere Application Server installation.

CEIIN0033E The WebSphere Application Server product version file *file_name1* **was not found.**

Explanation: See message.

Administrator Response: Verify that the installation of the WebSphere Application Server product completed successfully.

CEIIN0034I The Common Event Infrastructure server is already installed.

Explanation: No additional information is available for this message.

CEIIN0035I The enterprise application has already been deployed to the specified node.
Application: *application_name*
Node: *node_name*
Server: *server_name*

Explanation: See message.

Administrator Response: No action is required.

CEIIN0036I The enterprise application has not been deployed to the specified node.
Application: *application_name*
Node: *node_name*

Explanation: No additional information is available for this message.

CEIIN0037E The file *file_name* **could not be created.**
Exception message: *exception_message*

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0038E The file *source_file_name* **could not be copied to file** *destination_file_name***.**
Exception message: *exception_message*

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0039E The file *file_name* **could not be copied into the directory** *directory_name***.**
Exception message: *exception_message*

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0040E The WebSphere Application Server configuration directory *directory_name* **was not found.**

Explanation: See message.

Administrator Response: Verify that the installation of the WebSphere Application Server completed successfully.

CEIIN0041E The WebSphere Application Server configuration directory *directory_name* **does not contain any files.**

Explanation: See message.

Administrator Response: Verify that the installation of the WebSphere Application Server completed successfully.

CEIIN0042I The installation of the Common Event Infrastructure application completed successfully.

Explanation: No additional information is available for this message.

CEIIN0043E The installation of the Common Event Infrastructure application failed.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0044E The enterprise application failed to uninstall because the uninstallation script was not found.
Application: *application_name*
Script name: *script_name*

Explanation: You might have previously uninstalled the enterprise application.

Administrator Response: Verify that the installation of the Common Event Infrastructure application completed successfully.

CEIIN0045I The installation of the Common Event Infrastructure configuration files completed successfully.

Explanation: No additional information is available for this message.

CEIIN0046E Installation of the Common Event Infrastructure configuration files failed.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0047E Uninstallation of the Common Event Infrastructure configuration files failed.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0048E The WebSphere Application Server *server_name* is not running. Start the server to configure the Common Event Infrastructure JDBC provider and data source.

Explanation: See message.

Administrator Response: See message.

CEIIN0049I Version *version_number* of the Common Event Infrastructure application is installed.

Explanation: No additional information is available for this message.

CEIIN0050W Version *version_number* of the Common Event Infrastructure application is installed, but this version is not valid.

Explanation: Installation or uninstallation of the Common Event Infrastructure is attempted.

Administrator Response: Verify that the installation or uninstallation of the Common Event Infrastructure application completed successfully.

CEIIN0051I Installing version *version_number* of the Common Event Infrastructure application.

Explanation: No additional information is available for this message.

CEIIN0052I Uninstalling version *version_number* of the Common Event Infrastructure application.

Explanation: No additional information is available for this message.

CEIIN0053I The installation of the Common Event Infrastructure library files completed successfully.

Explanation: No additional information is available for this message.

CEIIN0054E The installation of the Common Event Infrastructure library files failed.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0055I Uninstallation of the Common Event Infrastructure library files completed successfully.

Explanation: No additional information is available for this message.

CEIIN0056E Uninstallation of the Common Event Infrastructure library files failed.

Explanation: See message.

Administrator Response: Consult the event infrastructure log files to determine the cause of the problem.

CEIIN0057I Installation of the Common Event Infrastructure tool files completed successfully.

Explanation: No additional information is available for this message.

CEIIN0058E Installation of the Common Event Infrastructure tool files failed.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0059I Uninstallation of the Common Event Infrastructure tool files completed successfully.

Explanation: No additional information is available for this message.

CEIIN0060E Uninstallation of the Common Event Infrastructure tool files failed.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0061W Setting execute permissions on some files in the directory *directory_name* **failed.**

Explanation: See message.

Administrator Response: Manually set the execute permissions for the shell scripts in the WAS_HOME/event directory using the **chmod +x** command.

CEIIN0062I Uninstallation of the Common Event Infrastructure application completed successfully.

Explanation: No additional information is available for this message.

CEIIN0063E Uninstallation of the Common Event Infrastructure application failed.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0064W The Common Event Infrastructure version file *file_name* **could not be deleted. Manually delete the specified file before reinstalling the Common Event Infrastructure application.**

Explanation: See message.

Administrator Response: See message.

CEIIN0065I Uninstallation of the common event infrastructure database completed successfully.

Explanation: No additional information is available for this message.

CEIIN0066E Uninstallation of the Common Event Infrastructure database failed.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0067E The WebSphere Deployment Manager is not running. The WebSphere Deployment Manager must be running before configuring or uninstalling the Common Event Infrastructure application.

Explanation: See message.

Administrator Response: Start the WebSphere Deployment Manager.

CEIIN0068E An error occurred when installing the Common Event Infrastructure application. Exception message: *exception_message*

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0069E An error occurred when uninstalling the Common Event Infrastructure application. Exception message: *exception_message*

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0070I Consult the log file for more information.

Explanation: No additional information is available for this message.

CEIIN0071I The installation of the Common Event Infrastructure administrative panel completed successfully.

Explanation: No additional information is available for this message.

CEIIN0072E The installation of the Common Event Infrastructure administrative panel failed.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0073I The uninstallation of the Common Event Infrastructure administrative panel completed successfully.

Explanation: No additional information is available for this message.

CEIIN0074E The uninstallation of the Common Event Infrastructure administrative panel failed.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0075I The Common Event Infrastructure administrative panel is not installed.

Explanation: No additional information is available for this message.

CEIIN0076E The file *file_name* **could not be deleted.**

Explanation: See message.

Administrator Response: Manually delete the specified file.

CEIIN0077W The directory *directory* **could not be deleted.**

Explanation: See message.

Administrator Response: Manually delete the specified directory.

CEIIN0078E Writing a file in EBCDIC format failed. Exception message: *exception_message*

Explanation: See message.

Administrator Response: See the exception message for additional information.

CEIIN0079E Reading a file in EBCDIC format failed. Exception message: *exception_message*

Explanation: See message.

Administrator Response: See the exception information for more information.

CEIIN0083E The system could not determine if WebSphere Application Server is operating in a stand-alone or a distributed environment.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0084E The enterprise application, *application_name*, could not be uninstalled because WebSphere Application Server security is enabled.

Explanation: See message.

Administrator Response: Manually uninstall the specified application.

CEIIN0085E The Common Event Infrastructure enterprise application, *application_name*, was not uninstalled.

Explanation: See message.

Administrator Response: Consult the event infrastructure log files to determine the cause of the problem. Correct the cause of the error and unaugment the profile again.

CEIIN0086E The Common Event Infrastructure data source could not be uninstalled because WebSphere Application Server security is enabled.

Explanation: See message.

Administrator Response: Manually uninstall the Common Event Infrastructure data source.

CEIIN0087E The Common Event Infrastructure database or data source could not be uninstalled. Manually uninstall the Common Event Infrastructure database and data source, then delete the directory *directory_name*.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem. Manually uninstall the Common Event Infrastructure database and data source.

CEIIN0088E The LOG_FILE_SIZE parameter in the response file *respfile* contains the value *value*, which is not valid.

Explanation: The LOG_FILE_SIZE parameter defines the size of each primary and secondary log file in 4K pages. The log file size limits the number of log records that can be written to the log file before it becomes full and a new log file is required.

Administrator Response: Enter a valid value in the response file for the LOG_FILE_SIZE parameter. Refer to the appropriate DB2 documentation for additional information.

CEIIN0089E The NUM_PRIMARY_LOG parameter in the response file *respfile* contains the value *value*, which is not valid.

Explanation: The primary log files establish a fixed amount of storage allocated to the recovery log files. Use the NUM_PRIMARY_LOG parameter to specify the number of primary log files to be allocated.

Administrator Response: Enter a valid value in the response file for the NUM_PRIMARY_LOG parameter. Refer to the appropriate DB2 documentation for additional information.

CEIIN0090E The NUM_SECONDARY_LOG parameter in the response file *respfile* contains the value *value*, which is not valid.

Explanation: The NUM_SECONDARY_LOG parameter specifies the number of secondary log files that are created and used for recovery log files, if needed.

Administrator Response: Enter a valid value in the response file for the NUM_SECONDARY_LOG parameter in the response file. Refer to the appropriate DB2 documentation for more information.

CEIIN0091E An error occurred when running the PluginProcessor command.

Explanation: See message.

Administrator Response: Perform the following steps:

1. Manually delete the Common Event Infrastructure files that were not deleted. The previous message specifies the files that need to be deleted.
2. Run the following command: **$WAS_HOME/bin/PluginProcessor -restore**

CEIIN0092I You must manually restore the administrative console enterprise application.

Explanation: See message.

Administrator Response: Perform the following steps:

1. Delete the Common Event Infrastructure files that were not deleted. The previous message specifies the files that need to be deleted.
2. Run the following command: **$WAS_HOME/bin/PluginProcessor -restore**

CEIIN0093I To restore the system, copy the file *source_file_name* **manually to the file** *destination_file_name***.**

Explanation: No additional information is available for this message.

CEIIN0094I To restore the system, manually collapse the enterprise application directory *directory_name* **to the temporary file** *source_file_name* **using the $WAS_HOME/bin/EARExpander command. Replace the file** *destination_file_name* **with the temporary file.**

Explanation: No additional information is available for this message.

CEIIN0095E The backup of the file *source_file_name* **to the file** *destination_file_name* **failed.**

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0096E The file *file_name* **cannot be restored from the backup copy of the file** *backup_file_name***.**

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem. Manually restore the file from the backup copy of the file.

CEIIN0097E Creation of a collapsed format of the administrative console enterprise application in the directory *directory_name* **to the file** *file_name* **failed.**

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0098E The database configuration script file, *file_name***, was not found.**

Explanation: See message.

Administrator Response: Verify that the preceding steps in the installation of the Common Event Infrastructure application completed successfully.

CEIIN0099I The installation of the Common Event Infrastructure database completed successfully.

Explanation: No additional information is available for this message.

CEIIN0100E The installation of the Common Event Infrastructure database failed.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem. Manually create the Common Event Infrastructure database.

CEIIN0102I You must manually create the Common Event Infrastructure database using the database configuration scripts.

Explanation: See message.

Administrator Response: For information about how to manually create the database, refer to the Common Event Infrastructure documentation.

CEIIN0103E The system could not determine if the WebSphere Application Server product installation is an application server or a Network Deployment installation.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0104E The configuration of the default Common Event Infrastructure service for profile *profile_path* **failed.**

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem. Manually configure the Common Event Infrastructure service.

CEIIN0105E The version *version_number* **of the Common Event Infrastructure application is installed, but this is an earlier version.**

Explanation: A request was made to augment a profile or configure a database and the jar file used to handle the request is a later version than the currently installed version.

Administrator Response: Use the installed copy of the Common Event Infrastructure installer for the request.

CEIIN0106I The enterprise application must be updated to use the Common Event Infrastructure server. Application: *application_name*

Explanation: See message.

Administrator Response: Update the specified application manually using the wsadmin tool.

CEIIN0107I Restoring Common Event Infrastructure application from version *from_version_number* **to version** *to_version_number***.**

Explanation: No additional information is available for this message.

CEIIN0108I The Common Event Infrastructure application has been restored successfully.

Explanation: No additional information is available for this message.

CEIIN0109E The Common Event Infrastructure application could not be restored.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0110E The Common Event Infrastructure application cannot be restored.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0111I The installation of the Common Event Infrastructure localization files completed successfully.

Explanation: No additional information is available for this message.

CEIIN0112E The installation of the Common Event Infrastructure localization files failed.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0113I The uninstallation of the Common Event Infrastructure localization files completed successfully.

Explanation: No additional information is available for this message.

CEIIN0114E The uninstallation of the Common Event Infrastructure localization files failed.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0115E An error occurred during the creation of the JAR file *file_name***.**
Exception message: *exception_message*

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0116I The enterprise application must be restored to the previous version to use the Common Event Infrastructure server.
Application: *application_name*

Explanation: See message.

Administrator Response: Update the specified application manually using the wsadmin tool.

CEIIN0123E The value of the parameter *parameter_name* **is** *parameter_value* **, which is not valid because it must be a positive number.**

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0124I The upgrade of the Common Event Infrastructure application completed successfully.

Explanation: No additional information is available for this message.

CEIIN0125E The upgrade of the Common Event Infrastructure application failed.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0126E The upgrade of the Common Event Infrastructure application completed, but errors occurred.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0127I Upgrading version *previous_version_number* **of the Common Event Infrastructure application to version** *current_version_number***.**

Explanation: No additional information is available for this message.

CEIIN0128I The Common Event Infrastructure localization files have been restored successfully.

Explanation: No additional information is available for this message.

CEIIN0129E The Common Event Infrastructure localization files could not be restored.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0130I A backup of the Common Event Infrastructure tool files completed successfully.

Explanation: No additional information is available for this message.

CEIIN0131E A backup of the Common Event Infrastructure tool files failed.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0132I The Common Event Infrastructure tool files have been restored successfully.

Explanation: No additional information is available for this message.

CEIIN0133E The Common Event Infrastructure tool files could not be restored.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0134I The Common Event Infrastructure tool files have been upgraded successfully.

Explanation: No additional information is available for this message.

CEIIN0135E The Common Event Infrastructure tool files could not be upgraded.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0136I The Common Event Infrastructure library files have been backed up successfully.

Explanation: No additional information is available for this message.

CEIIN0137E The Common Event Infrastructure library files could not be backed up.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0138I The Common Event Infrastructure library files have been restored successfully.

Explanation: No additional information is available for this message.

CEIIN0139E The Common Event Infrastructure library files could not be restored.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0140I The Common Event Infrastructure library files have been upgraded successfully.

Explanation: No additional information is available for this message.

CEIIN0141E The Common Event Infrastructure library files could not be upgraded.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0142I The Common Event Infrastructure administrative panel has been backed up successfully.

Explanation: No additional information is available for this message.

CEIIN0143E The Common Event Infrastructure administrative panel could not be backed up.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0144I The Common Event Infrastructure administrative panel has been restored successfully.

Explanation: No additional information is available for this message.

CEIIN0145E The Common Event Infrastructure administrative panel could not be restored.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0146I The Common Event Infrastructure administrative panel has been upgraded successfully.

Explanation: No additional information is available for this message.

CEIIN0147E The Common Event Infrastructure administrative panel could not be upgraded.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0148I The Common Event Infrastructure configuration files have been backed up successfully.

Explanation: No additional information is available for this message.

CEIIN0149E The Common Event Infrastructure configuration files could not be backed up.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0150I The Common Event Infrastructure configuration files have been restored successfully.

Explanation: No additional information is available for this message.

CEIIN0151E The Common Event Infrastructure configuration files could not be restored.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0152I The Common Event Infrastructure configuration files have been upgraded successfully.

Explanation: No additional information is available for this message.

CEIIN0153E The Common Event Infrastructure configuration files could not be upgraded.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0154E The enterprise application, *application_name*, could not be installed because WebSphere Application Server security is enabled and WebSphere user credentials were not provided.

Explanation: See message.

Administrator Response: Manually install the specified application.

CEIIN0155E The enterprise application, *application_name*, could not be uninstalled because WebSphere Application Server security is enabled and WebSphere user credentials were not provided.

Explanation: See message.

Administrator Response: Manually uninstall the specified application.

CEIIN0156I This is a new installation of the Common Event Infrastructure application.

Explanation: No additional information is available for this message.

CEIIN0157I This is an upgrade installation of the Common Event Infrastructure application.

Explanation: No additional information is available for this message.

CEIIN0158E The passwords do not match. Enter the passwords again.

Explanation: The password and the retype password must be the same.

Administrator Response: See message.

CEIIN0159W The event database could not be uninstalled because the uninstallation script was not found.
Database type: *database_type*
Script name: *script_name*

Explanation: You might have previously uninstalled the enterprise application.

Administrator Response: Verify that the installation of the Common Event Infrastructure application completed successfully.

CEIIN0160E An error occurred while upgrading configuration files.
Exception message: *exception_message*

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0161I The database must be updated to use the Common Event Infrastructure server.

Explanation: See message.

Administrator Response: Update the database manually using the database update script.

CEIIN0162I The database must be restored to the previous version to use the Common Event Infrastructure server.

Explanation: See message.

Administrator Response: Restore the database manually.

CEIIN0163I The data source *data_source* **is found.**

Explanation: No additional information is available for this message.

CEIIN0164I The data source *data_source* **is not found.**

Explanation: No additional information is available for this message.

CEIIN0165E The backup of the directory *source_directory_name* **to the directory** *destination_directory_name* **failed.**

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0166E The directory *directory_name* **cannot be restored from the backup copy of the directory** *backup_directory_name***.**

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem. Manually restore the directory from the backup copy of the directory.

CEIIN0167E The backup of the directories *database_directory_name* **and** *data_source_directory_name* **failed.**

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem. Manually back up the specified directories.

CEIIN0168E The directories *database_directory_name* **and** *data_source_directory_name* **could not be restored.**

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem. Manually restore the specified directories.

CEIIN0169I The backup of the directories *database_directory_name* **and** *data_source_directory_name* **completed successfully.**

Explanation: No additional information is available for this message.

CEIIN0170I The directories *database_directory_name* **and** *data_source_directory_name* **have been restored successfully.**

Explanation: No additional information is available for this message.

CEIIN0171I The Common Event Infrastructure application was not uninstalled because it is being used by other products.
Product names: *product_names*

Explanation: No additional information is available for this message.

CEIIN0172E The Cloudscape database *database_name* already exists in *path*. You cannot create another database with the same name.

Explanation: See message.

Administrator Response: Either remove the existing Cloudscape database, or create the database with a different name.

CEIIN0173I The Cloudscape database *database* has been removed successfully

Explanation: No additional information is available for this message.

CEIIN0174E The removal of the Cloudscape database *database* failed because it does not exist.

Explanation: See Message

Administrator Response: Verify that the database exists and manually remove it.

CEIIN0176I Creating the database and data source scripts.

Explanation: No additional information is available for this message.

CEIIN0177I The output directory for the database scripts is *directory_name*.

Explanation: No additional information is available for this message.

CEIIN0178I The output directory for the data source scripts is *directory_name*.

Explanation: No additional information is available for this message.

CEIIN0179I Running the database and data source creation scripts.

Explanation: No additional information is available for this message.

CEIIN0180I The Cloudscape system directory is *directory_name*.

Explanation: No additional information is available for this message.

CEIIN0181I Using the response file *response_file_name*.

Explanation: No additional information is available for this message.

CEIIN0182I Configuring the Common Event Infrastructure database and data sources.

Explanation: No additional information is available for this message.

CEIIN0183E The command *command* failed.
Exception message: *exception_message*

Explanation:

Administrator Response: See the exception message and consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0184I The creation of the event database on *database_type* for the Common Event Infrastructure application completed successfully.

Explanation: No additional information is available for this message.

CEIIN0185I The creation of the data source *data_source* for the Common Event Infrastructure completed successfully.

Explanation: No additional information is available for this message.

CEIIN0186I The Common Event Infrastructure database and data source scripts for *database_type* created successfully.

Explanation: No additional information is available for this message.

CEIIN0187I The Common Event Infrastructure installer cannot create the DB2 database on a z/OS system. You must manually create the database using the database scripts in the *directory_name* directory.

Explanation: The DB2 database on a z/OS system needs to be created manually using DB2 tools. For more information, refer to the Common Event Infrastructure documentation.

CEIIN0188I The Common Event Infrastructure installer cannot remove the DB2 database from a z/OS system. You must manually remove the database using the database scripts in the *directory_name* directory.

Explanation: The DB2 database on a z/OS system needs to be removed manually using DB2 tools. For more information, refer to the Common Event Infrastructure documentation.

CEIIN0189E The value of the parameter must be a positive integer.
Parameter: *parameter_name*
Value: *parameter_value*

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0190E The following error occurred: *exception_message*

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0191E An error occurred when creating or updating the Common Event Infrastructure product information file.
Product information file: *file_name*
Exception message: *exception_message*

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0192E The type of DB2 database installation (server or client) could not be determined.

Explanation: The type of DB2 database installation (server or client) was not registered during product installation.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files.

CEIIN0193E Modification of the WebSphere Application Server connection property file failed.
Property File: *property_file*

Explanation: A failure occurred when the WebSphere Application Server connection property file was being modified with encrypted user ID and password information.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0194E The WebSphere Application Server connection property file could not be restored from the backup file.
Property file: *property_file*
Backup file: *backup_file*

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem, and manually restore the property file from the backup file.

CEIIN0195I Using WebSphere Application Server directory *was_directory***.**

Explanation: No additional information is available for this message.

CEIIN0196I Creating the file *file_name***.**

Explanation: No additional information is available for this message.

CEIIN0197I The configuration of the Common Event Infrastructure database completed successfully.

Explanation: No additional information is available for this message.

CEIIN0198E The location of the Common Event Infrastructure installation images could not be determined from the classpath settings.
Classpath: *classpath*

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0199I The Common Event Infrastructure application is partially installed. The configuration and the administrative panel files of the Common Event Infrastructure application, version *version_number* **are being installed.**

Explanation: The Common Event Infrastructure application is partially installed because the initial installation of the Common Event Infrastructure application was performed on a federated node that is now an unfederated node. For more information, refer to the Common Event Infrastructure documentation.

CEIIN0200E The Common Event Infrastructure application could not be set up to use the messaging services because the user ID or password for embedded messaging authentication were not provided. The enterprise application *application_name* **could not be installed**

Explanation: See message.

Administrator Response: Manually install the specified application.

CEIIN0201E A later version of the Common Event Infrastructure server *ver* **is already installed.**

Explanation: An attempt was made to install the Common Event Infrastructure server when a later version was already installed. The installation failed.

Administrator Response: Use the version of the cei-installer.jar file in the Common Event Infrastructure installation directory to start the installer.

CEIIN0202I A backup of the Common Event Infrastructure localization files completed successfully.

Explanation: No additional information is available for this message.

CEIIN0203E A backup of the Common Event Infrastructure localization files failed.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0204I The Common Event Infrastructure localization files have been upgraded successfully.

Explanation: No additional information is available for this message.

CEIIN0205E The Common Event Infrastructure localization files could not be upgraded.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0206E The version *version_number* **of WebSphere Application Server is installed, but this version is not compatible for the installation of the Common Event Infrastructure.**

Explanation: The installation or uninstallation of the Common Event Infrastructure could not be completed. An incompatible version of WebSphere Application Server was detected on your system.

Administrator Response: Verify that you have a compatible version of WebSphere Application Server installed on your system.

CEIIN0207I The Common Event Infrastructure enterprise applications must be updated to use the Common Event Infrastructure server.

Explanation: See message.

Administrator Response: Update the Common Event Infrastructure enterprise applications manually using the wsadmin tool.

CEIIN0208I Upgrading the Common Event Infrastructure database.

Explanation: No additional information is available for this message.

CEIIN0209I The output directory for the database upgrade scripts is *directory_name***.**

Explanation: No additional information is available for this message.

CEIIN0210I The Common Event Infrastructure database upgrade completed successfully.

Explanation: No additional information is available for this message.

CEIIN0211I The Common Event Infrastructure databases are current.

Explanation: No additional information is available for this message.

CEIIN0212E An upgrade of the Common Event Infrastructure database failed.

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0213I The Common Event Infrastructure database upgrade on the native z/OS system must be done manually. Convert the upgrade DDL script *file_path* **to a dataset then use SPUFI to load and run the dataset. Consult the Common Event Infrastructure documentation on upgrading a database.**

Explanation: No additional information is available for this message.

CEIIN0214W The Common Event Infrastructure database upgrade scripts were generated but not run because EXECUTE_SCRIPT=NO is set in the response file. To complete the database upgrade, run *file_path***.**

Explanation: No additional information is available for this message.

CEIIN0215E The directory *directory_name* **is not a valid CEI_HOME directory.**

Explanation: The Common Event Infrastructure installation directory could not be found in the specified directory.

Administrator Response: Ensure that you have specified the correct location for the Common Event Infrastructure installation.

CEIIN0216E The directory *path_name* **specified for the key PROFILE_PATH in the response file does not exist.**

Explanation: The WebSphere Application Server profile path could not be found in the specified directory.

Administrator Response: Ensure that you have specified the correct location for the WebSphere Application Server profile.

CEIIN0217I Using WebSphere Application Server profile directory *profile_directory***.**

Explanation: No additional information is available for this message.

CEIIN0218E The directory *directory_name* **is not a valid PROFILE_PATH directory.**

Explanation: The specified path is not a WebSphere Application Server profile path.

Administrator Response: Ensure that you have specified the correct location for the WebSphere Application Server profile.

CEIIN0219E A required property is missing from the feature file *feature_file_name***. Required property:** *property_name***.**

Explanation: An error occurred while parsing the feature file. The specified property was missing.

Administrator Response: Ensure that the specified file has not been manually modified.

CEIIN0220E Common Event Infrastructure is already installed in the location *cei_install_path* **which differs from the path specified in the response file:** *response_file_path***.**

Explanation: The CEI_HOME value specified in the installation response file is different from the location where it is already installed. The installation failed.

Administrator Response: Specify the location where Common Event Infrastructure is already installed for the CEI_HOME key in the response file.

CEIIN0221E The Common Event Infrastructure application is not installed.

Explanation: The Common Event Infrastructure application must be installed before using this function.

Administrator Response: Install the Common Event Infrastructure before calling the function.

CEIIN0222E The version *cei_version* **of the Common Event Infrastructure is installed. It is a more recent version than the one being used to process the request.**

Explanation: The version of the Common Event Infrastructure that is running is at a lower level than the one that is already installed.

Administrator Response: Set the CLASSPATH to point to the copy of the Common Event Infrastructure that is already installed.

CEIIN0223E Either SERVER_NAME or CLUSTER_NAME can be specified but the response file contained values for both.

Explanation: Specify only one of these values in the response file.

Administrator Response: Update the response file parameters and issue the request again.

CEIIN0224I The augmentation of the Common Event Infrastructure configuration files to profile *profile_path* **completed successfully.**

Explanation: No additional information is available for this message.

CEIIN0225I The augmentation of the Common Event Infrastructure enterprise application to profile *profile_path* **completed successfully.**

Explanation: No additional information is available for this message.

CEIIN0226I The creation of the Common Event Infrastructure database for profile *profile_path* **completed successfully.**

Explanation: No additional information is available for this message.

CEIIN0227I The augmentation of the Common Event Infrastructure administration panels for profile *profile_path* **completed successfully.**

Explanation: No additional information is available for this message.

CEIIN0228I The augmentation of the Common Event Infrastructure library files for profile *profile_path* **completed successfully.**

Explanation: No additional information is available for this message.

CEIIN0229I The augmentation of the Common Event Infrastructure tools for profile *profile_path* **completed successfully.**

Explanation: No additional information is available for this message.

CEIIN0230I The Common Event Infrastructure configuration files were successfully uninstalled from profile *profile_path***.**

Explanation: No additional information is available for this message.

CEIIN0231I The Common Event Infrastructure enterprise application was successfully uninstalled from profile *profile_path***.**

Explanation: No additional information is available for this message.

CEIIN0232I The Common Event Infrastructure database was successfully uninstalled from profile *profile_path***.**

Explanation: No additional information is available for this message.

CEIIN0233I The Common Event Infrastructure administration panels were successfully uninstalled from profile *profile_path***.**

Explanation: No additional information is available for this message.

CEIIN0234I The Common Event Infrastructure library files were successfully uninstalled from profile *profile_path***.**

Explanation: No additional information is available for this message.

CEIIN0235I The Common Event Infrastructure tools were successfully uninstalled from profile *profile_path***.**

Explanation: No additional information is available for this message.

CEIIN0236E The augmentation of the Common Event Infrastructure configuration files to profile *profile_path* **failed.**

Explanation: No additional information is available for this message.

CEIIN0237E The augmentation of the Common Event Infrastructure enterprise application to profile *profile_path* **failed.**

Explanation: No additional information is available for this message.

CEIIN0238E The creation of the Common Event Infrastructure database for profile *profile_path* **failed.**

Explanation: No additional information is available for this message.

CEIIN0239E The augmentation of the Common Event Infrastructure administration panels for profile *profile_path* **failed.**

Explanation: No additional information is available for this message.

CEIIN0240E The augmentation of the Common Event Infrastructure library files for profile *profile_path* **failed.**

Explanation: No additional information is available for this message.

CEIIN0241E The augmentation of the Common Event Infrastructure tools for profile *profile_path* **failed.**

Explanation: No additional information is available for this message.

CEIIN0242E The removal of the Common Event Infrastructure configuration files from profile *profile_path* **failed.**

Explanation: No additional information is available for this message.

CEIIN0243E The removal of the Common Event Infrastructure enterprise application from profile *profile_path* **failed.**

Explanation: No additional information is available for this message.

CEIIN0244E The removal of the Common Event Infrastructure database for profile *profile_path* **failed.**

Explanation: No additional information is available for this message.

CEIIN0245E The removal of the Common Event Infrastructure administration panels from profile *profile_path* **failed.**

Explanation: No additional information is available for this message.

CEIIN0246E The removal of the Common Event Infrastructure library files from profile *profile_path* **failed.**

Explanation: No additional information is available for this message.

CEIIN0247E The removal of the Common Event Infrastructure administration tools from profile *profile_path* **failed.**

Explanation: No additional information is available for this message.

CEIIN0248I The Common Event Infrastructure was previously augmented to profile *profile_path* **as a client. This profile will be augmented as a server for product** *product_id***.**

Explanation: A request was made to augment the specified profile as a server, but it is currently augmented with the Common Event Infrastructure as a client. The profile will be augmented as a server and the product is registered as a user of this profile.

CEIIN0249I The Common Event Infrastructure was previously augmented to profile *profile_path* **as a server. Product** *product_id* **is registered as a user of this server profile.**

Explanation: A request was made to augment the specified profile as a server, but it was already augmented with the Common Event Infrastructure as a server. The product is registered as a user of this profile.

CEIIN0250I The Common Event Infrastructure was not removed from profile *profile_path* **because it is being used by products other than** *product_id***.**

Explanation: A request was made to remove the Common Event Infrastructure application from the specified profile, but no files were removed because other products are registered as Common Event Infrastructure users for that profile. This product is removed from the list of registered users for this profile.

CEIIN0251I The product *product_id* **is not a registered user of profile** *profile_path***.**

Explanation: A request was made to remove the Common Event Infrastructure application or database from the specified profile, but the requesting product ID is not registered as a user of this profile. If there are no other registered users of this profile, an attempt is made to handle the request.

Administrator Response: Check the Common Event Infrastructure log files for more details.

CEIIN0252E The unaugmentation of at least one profile failed during the uninstallation of the Common Event Infrastructure for product *product_id***. The product was not uninstalled.**

Explanation: A request was made to uninstall the Common Event Infrastructure, but the uninstaller was not able to unaugment one or more profiles registered by the specified product. Other profiles might have been unaugmented, but the product was not uninstalled.

Administrator Response: Consult the log for messages that indicate which profile or profiles could not be unaugmented and why. Correct the cause of the problem and unaugment each failed profile separately, then issue the uninstall request again.

CEIIN0253E An error occurred when updating the Common Event Infrastructure product information file for profile *profile_path***.**
Product information file: *file_name*
Exception message: *exception_message*

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0254I The Common Event Infrastructure was previously augmented to profile *profile_path* **as a federated node. It will be augmented as a standalone server.**

Explanation: A request was made to augment the specified profile as a server. This profile was originally augmented as a server when it was a federated node. The Common Event Infrastructure components required to make this a standalone server will be installed and the product will be registered.

CEIIN0255E An error occurred when augmenting the Common Event Infrastructure to a profile. Exception message: *exc_message*.

Explanation: A request was made to augment a profile and an exception occurred during processing.

Administrator Response: Consult the exception message and correct the cause of the error, then try the request again.

CEIIN0256E An error occurred when unaugmenting the Common Event Infrastructure from a profile or removing a database. Exception message: *exc_message*.

Explanation: A request was made to unaugment a profile or remove a database, and an exception occurred during processing.

Administrator Response: Consult the exception message and correct the cause of the error, then try the request again.

CEIIN0257E A database cannot be configured on profile *profile_path* **because it is augmented as a Common Event Infrastructure client.**

Explanation: A request was made to configure a database for the specified profile, but either the request specified that the profile be augmented as a client, or the profile is currently augmented as a client and the request did not specify that the profile should be augmented as a server.

Administrator Response: Verify that the profile you specfied is the profile that you want to augment. If it is, then it cannot be augmented as a client. Specify CLIENT_ONLY=false in the response file.

CEIIN0258E A database cannot be configured for profile *profile_path* **because is not augmented with the Common Event Infrastructure.**

Explanation: A request was made to configure the Common Event Infrastructure database for the specified profile, but the profile is not currently augmented with the Common Event Infrastructure application.

Administrator Response: Verify that the profile you specfied is the profile that you want to augment. If it is, then augment it before or at the same time that you configure the database.

CEIIN0259I The augmentation of the Common Event Infrastructure completed successfully for profile *profile_path*.

Explanation: No additional information is available for this message.

CEIIN0260E The augmentation of the Common Event Infrastructure failed for profile *profile_path*.

Explanation: A request was made to augment the specified profile, but the request failed.

Administrator Response: Refer to additional messages in the log file for the reason for the failure.

CEIIN0261E The augmentation of the Common Event Infrastructure to profile *profile_path* **completed, but an error occurred during the registration of the product.**

Explanation: A request was made to augment the specified profile, and the augmentation was successful, but the user was not registered as a user. The Common Event Infrastructure does function in that profile, but could be uninstalled by another product.

Administrator Response: Determine why the Common Event Infrastructure product version file could not be updated, correct the problem, then use the Common Event Infrastructure install Java APIs to register the product as a user of this profile.

CEIIN0262I The profile *profile_path* was already augmented with the Common Event Infrastructure application. The product ID has been registered as a user.

Explanation: A request was made to augment the specified profile when it was already augmented with the Common Event Infrastructure application. No additional augmentation was done to the profile, but the requesting product ID was added to the list of registered users.

CEIIN0263W The database configuration for profile *profile_path* might already have been performed by another product. Verify that the database has been configured correctly.

Explanation: A request was made to augment the specified profile as a server, but it was already augmented with the Common Event Infrastructure application by another product. The database might have been configured with different settings than the current request.

Administrator Response: Manually verify that the Common Event Infrastructure database is configured for this profile. If not, use the Common Event Infrastructure installation scripts to configure the database.

CEIIN0264E The version of the Common Event Infrastructure installer that is handling the request does not match the version of the Common Event Infrastructure application that is currently installed.

Explanation: A request was made to augment or unaugment a profile, or to uninstall the Common Event Infrastructure application. The version of the Common Event Infrastructure that is being used to handle the request is not at the same level as the version of the Common Event Infrastructure that is currently installed. The versions must be at the same level. The request failed.

Administrator Response: Use the version of the Common Event Infrastructure application that is currently installed to handle the request.

CEIIN0265E The Common Event Infrastructure database configuration for profile *profile_path* failed because the product ID *product_id* is not a registered user for that profile.

Explanation: See message.

Administrator Response: The requesting product must augment the profile or use the Common Event Infrastructure installation Java APIs to register itself as a user for this profile before configuring the database.

CEIIN0266I The Common Event Infrastructure application is being augmented to profile *profile_path*.

Explanation: No additional information is available for this message.

CEIIN0267I The CEI_HOME value specified in the response file is already associated with another instance of WebSphere Application Server.

Explanation: The Common Event Infrastructure is already installed in the CEI_HOME location specified in the installation response file, however it is not associated with the WAS_HOME keyword specified in the response file. This instance of Common Event Infrastructure application is in use by another instance of WebSphere Application Server. A single installation of the Common Event Infrastructure cannot be used by more than one instance of WebSphere Application Sever. The installation fails.

Administrator Response: Specify a different value for CEI_HOME in the response file and issue the request again.

CEIIN0268I The version *cei_version* of the Common Event Infrastructure is installed. A later version *cei_version2* is being used to process the request.

Explanation: The version of the Common Event Infrastructure that is running is at a higher level than the one that is installed.

Administrator Response: If it is a supported upgrade, the installed copy of the Common Event Infrastructure is upgraded to the higher version. If the upgrade is not supported, the request fails.

CEIIN0269I The Common Event Infrastructure database is not installed for profile *profile_path* **because it is a WebSphere Deployment Manager.**

Explanation: A request was made to add or remove the Common Event Infrastructure database from the specified profile, it is a WebSphere Deployment Manager, and the Common Event Infrastructure does not create a database on the WebSphere Deployment Manager.

CEIIN0270E The parameter *parameter_name* **in the response file** *response_file_name* **cannot be specified if the keyword ACTION is** *action***.**

Explanation: The specified parameter is not valid for the requested action.

Administrator Response: Update the response file parameter and issue the request again.

CEIIN0271I The product *product_id* **is not a registered user of the Common Event Infrastructure application.**

Explanation: A request was made to uninstall the Common Event Infrastructure application, or augment a profile, but the requesting product is not registered as a user of the Common Event Infrastructure application.

Administrator Response: If the request was to uninstall, the uninstallation is attempted if there are no other registered users. If the request was to augment a profile, the request fails.

CEIIN0272I The Common Event Infrastructure application is being unaugmented from profile *profile_path***.**

Explanation: No additional information is available for this message.

CEIIN0273I The installation of the Common Event Infrastructure image files completed successfully.

Explanation: No additional information is available for this message.

CEIIN0274E The installation of the Common Event Infrastructure image files failed.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0275I The uninstallation of the Common Event Infrastructure image files completed successfully.

Explanation: No additional information is available for this message.

CEIIN0276E The uninstallation of the Common Event Infrastructure image files failed.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0277I The unaugmentation of the Common Event Infrastructure or removal of the database completed successfully for profile *profile_path***.**

Explanation: No additional information is available for this message.

CEIIN0278E The unaugmentation of the Common Event Infrastructure application or removal of the database failed for profile *profile_path***.**

Explanation: See message.

Administrator Response: Refer to additional messages in the log file for the failure.

CEIIN0279I The Common Event Infrastructure database was not removed from profile *profile_path* because it is being used by products other than *product_id*.

Explanation: See message.

CEIIN0280E The unaugmentation of the Common Event Infrastructure application from profile *profile_path* failed because the product could not be removed from the product file.

Explanation: A request was made to unaugment the specified profile but the user could not be unregistered.

Administrator Response: Determine why the uninstaller was not able to update the Common Event Infrastructure product version file. Correct the problem and issue the request again.

CEIIN0281E The uninstallation of the Common Event Infrastructure application failed because the product could not be removed from the product version file.

Explanation: There are multiple products registered as users of the Common Event Infrastructure and this product could not be removed from the list of registered users.

Administrator Response: Determine why the Common Event Infrastructure product version file could not be updated, correct the problem and issue the request again.

CEIIN0282E The removal of the Common Event Infrastructure enterprise application or database failed for profile *profile_path* during unaugmentation.

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem. Correct the problem and unaugment the profile again.

CEIIN0283E A value for either SERVER_NAME or CLUSTER_NAME must be specified in the response file when it contains DEPLOY_EJB=true.

Explanation: When the response file contains DEPLOY_EJB=true, either SERVER_NAME or CLUSTER_NAME must be specified. The SERVER_NAME or the CLUSTER_NAME was specified in the response file.

Administrator Response: Administrator Response: Ensure that the response file contains a value for either SERVER_NAME or CLUSTER_NAME when DEPLOY_EJB=true, and issue the request again.

CEIIN0284E It is not valid to specify the keywords CONFIGURE_DB or DEPLOY_EJB when the profile is being augmented as a client.

Explanation: See message.

Programmer response: Correct the response file and issue the request again.

CEIIN0285I The Common Event Infrastructure application is processing your request. Request processing can take several minutes.

Explanation: See message.

CEIIN0291E The WebSphere Application Server environment variables could not be created because WebSphere Application Server security is enabled and WebSphere user credentials were not provided.

Explanation: See message.

Administrator Response: Create the WebSphere Application Server variables by running a configuration script provided by the Common Event Infrastructure application. See the Common Event Infrastructure Developer's Guide for further information.

CEIIN0292E The WebSphere Application Server environment variables could not be removed because WebSphere Application Server security is enabled and WebSphere user credentials were not provided.

Explanation: See message.

Administrator Response: Remove the WebSphere Application Server variables by running a configuration script provided by the Common Event Infrastructure application. See the Common Event Infrastructure Developer's Guide for further information.

CEIIN0293E The WebSphere Application Server environment variables or properties modified by the Common Event Infrastructure application could not be removed for profile *profile_name***. They can be removed manually.**

Explanation: During profile augmentation the Common Event Infrastructure application creates some WebSphere Application Server environment variables and modifies the custom ws.ext.dirs property. The variables or the property could not be removed. It is probably due to either authentication failure or because the WebSphere Deployment Manager was not running. Clean these up manually using the WebSphere Administrative Console.

Administrator Response: Manually modify or remove the WebSphere Application Server variables and custom properties.

CEIIN0294E The Common Event Infrastructure application could not be augmented to profile *profile_name* **because a previous unaugmentation of the Common Event Infrastructure application failed.**

Explanation: The specified profile was previously augmented and then unaugmented with the Common Event Infrastructure application. However, the unaugmentation failed, and the profile is in a partially augmented state. The augmentation request failed.

Administrator Response: Determine the cause of the previous unaugmentation failure, correct the problem, and unaugment the profile. After that is successful, augment the profile with the Common Event Infrastructure application.

CEIIN0450I The migration of the Common Event Infrastructure application completed successfully for profile *profile_path***.**

Explanation: No additional information is available for this message.

CEIIN0451E The Common Event Infrastructure application migration failed for profile *profile_path***.**

Explanation: A request was made to migrate a Common Event Infrastructure Version 5.1.x to the specified profile, but the request failed.

Administrator Response: Refer to additional messages in the log file for the failure.

CEIIN0452E The migration could not be completed because the destination profile *profile_path* **is not augmented with Common Event Infrastructure application.**

Explanation: See message.

Administrator Response: Invoke the Common Event Infrastructure application migration after you augment the profile with Common Event Infrastructure application.

CEIIN0453E The Common Event Infrastructure migration failed because WebSphere Application Server security is enabled and WebSphere user credentials were not provided.

Explanation: See message.

Administrator Response: Invoke the Common Event Infrastructure application migration and provide the appropriate credentials.

CEIIN0454E The directory *directory_name* **is not a valid WebSphere backup directory.**

Explanation: The value specified for WAS_BACKUP_DIR in the response file is not a WebSphere backup directory.

Administrator Response: Ensure that you have specified the correct location for the WebSphere backup directory.

CEIIN0455E Unable to copy the configuration files from the WebSphere backup directory *backup_directory_name* **to the destination profile directory** *profile_directory_name***.**

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem. Verify that the WebSphere Application Server migration completed successfully before attempting to run the Common Event Infrastructure migration.

CEIIN0456E The combination of *combination* **is not valid in the response file.**

Explanation: If SCOPE=server, then SERVER_NAME must be set to a valid WebSphere Application Server server name. If SCOPE=cluster, then CLUSTER_NAME must be set to a valid WebSphere Application Server cluster name.

CEIIN0500I The enterprise application *application_name* **is detected.**

Explanation: No additional information is available for this message.

CEIIN0501I The enterprise application *application_name* **is not detected.**

Explanation: No additional information is available for this message.

CEIIN0502I The enterprise application *application_name* **has been deployed to node** *node_name***.**

Explanation: No additional information is available for this message.

CEIIN0503I The enterprise application *application_name* **has not been deployed to any server at node** *node_name***.**

Explanation: No additional information is available for this message.

CEIIN0504I The enterprise application *application_name* **has been deployed to node** *node_name* **on server** *server_name***.**

Explanation: No additional information is available for this message.

CEIIN0505I The enterprise application *application_name* **has not been deployed to node** *node_name* **on server** *server_name***.**

Explanation: No additional information is available for this message.

CEIIN0506E Deployment information for the enterprise application *application_name* **cannot be found.**

Explanation: Although the enterprise application has been deployed, the deployment information could not be found.

Administrator Response: Verify that the WebSphere Application Server configuration is valid.

CEIIN0507I The enterprise application *application_name* **at node** *node_name* **on server** *server_name* **is disabled.**

Explanation: No additional information is available for this message.

CEIIN0510I The enterprise application *application_name* **has been installed successfully on node** *node_name* **on server** *server_name***.**

Explanation: No additional information is available for this message.

CEIIN0511E The enterprise application *application_name* **failed to install at node** *node_name* **on server** *server_name***.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0512W Updates to the enterprise application *application_name* **are done in local mode.**

Explanation: See message.

CEIIN0513I Updates to the enterprise application *application_name* **completed successfully.**

Explanation: No additional information is available for this message.

CEIIN0514E Updates to the enterprise application *application_name* **failed.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0515I The enterprise application *application_name* **has been uninstalled successfully from node** *node_name* **on server** *server_name***.**

Explanation: No additional information is available for this message.

CEIIN0516E The enterprise application *application_name* **could not be uninstalled from node** *node_name* **on server** *server_name***.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0517I The enterprise application *application_name* **has been uninstalled successfully.**

Explanation: No additional information is available for this message.

CEIIN0518E The enterprise application *application_name* **could not be uninstalled.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0519E The JMS provider *jms_provider_name* **at scope** *scope* **was not found.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output. Also, verify that the WebSphere Application Server configuration is valid.

CEIIN0520E A message listener service at scope *scope* was not found.

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output. Also, verify that the WebSphere Application Server configuration is valid.

CEIIN0521E An application server at scope *scope* was not found.

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output. Also, verify that the WebSphere Application Server configuration is valid.

CEIIN0522E The EJB container for the application server *application_server* was not found.

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output. Also, verify that the WebSphere Application Server configuration is valid.

CEIIN0523I The WebSphere Application Server queue *queue_name* at scope *scope* has been created successfully.

Explanation: No additional information is available for this message.

CEIIN0524E The WebSphere Application Server queue *queue_name* at scope *scope* could not be created.

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0525I The WebSphere Application Server queue *queue_name* at scope *scope* has been removed successfully.

Explanation: No additional information is available for this message.

CEIIN0526E The WebSphere Application Server queue *queue_name* at scope *scope* could not be deleted.

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0527I The WebSphere Application Server queue connection factory *queue_connection_factory_name* at scope *scope* has been created successfully.

Explanation: No additional information is available for this message.

CEIIN0528E The WebSphere Application Server queue connection factory *queue_connection_factory_name* at scope *scope* could not be created.

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0529I The WebSphere Application Server queue connection factory *queue_connection_factory_name* **at scope** *scope* **has been removed successfully.**

Explanation: No additional information is available for this message.

CEIIN0530E The WebSphere Application Server queue connection factory *queue_connection_factory_name* **at scope** *scope* **could not be removed.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0531I The listener port *queue_name* **at scope** *scope* **has been created successfully.**

Explanation: No additional information is available for this message.

CEIIN0532E The listener port *queue_name* **at scope** *scope* **could not be created.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0533I The listener port *queue_name* **at scope** *scope* **has been removed successfully.**

Explanation: No additional information is available for this message.

CEIIN0534E The listener port *queue_name* **at scope** *scope* **could not be removed.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0535W The JMS server at scope *scope* **could not be found.**

Explanation: See message.

CEIIN0536I The JMS server at scope *scope* **is configured to listen on queue** *queue_name***.**

Explanation: No additional information is available for this message.

CEIIN0537I The queue *queue_name* **has been added successfully to the configuration of the JMS server at scope** *scope***.**

Explanation: No additional information is available for this message.

CEIIN0538E The queue *queue_name* **could not be added to configuration of the JMS server at scope** *scope***.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0539I The queue *queue_name* **has been removed successfully from the configuration of the JMS server at scope** *scope***.**

Explanation: No additional information is available for this message.

CEIIN0540W The queue *queue_name* **could not be removed from the configuration of the JMS server at scope** *scope*

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0541I Installing messaging resources.

Explanation: No additional information is available for this message.

CEIIN0542I Uninstalling messaging resources.

Explanation: No additional information is available for this message.

CEIIN0543I Installing the enterprise application, *application_name* **on node** *node_name* **on server** *server_name***.**

Explanation: No additional information is available for this message.

CEIIN0544I Uninstalling the enterprise application.

Explanation: No additional information is available for this message.

CEIIN0545I Updating the enterprise application.

Explanation: No additional information is available for this message.

CEIIN0546I Configuration updates have been saved.

Explanation: No additional information is available for this message.

CEIIN0547I Configuration updates have not been saved.

Explanation: No additional information is available for this message.

CEIIN0548E Enterprise application updates could not been saved.

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0549E A general error occurred during processing.
Exception message: *exception_message*

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0550E The node *node_name* **could not be found.**

Explanation: See message.

Administrator Response: Verify that the specified node exists on WebSphere Application Server installation.

CEIIN0551E The server *server_name* **could not be found at node** *node_name***.**

Explanation: See message.

Administrator Response: Verify that the specified server exists on WebSphere Application Server node.

CEIIN0552E A general error occurred during the update of the enterprise application.
Application: *application_name*
Exception message: *exception_message*

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0553E The application management MBean could not be found.

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output. Verify that the WebSphere Application Server configuration is valid.

CEIIN0554I The WebSphere Application Server topic *topic_name* **at scope** *scope* **has been created successfully.**

Explanation: No additional information is available for this message.

CEIIN0555E The WebSphere Application Server topic *topic_name* **at scope** *scope* **could not be created.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0556I The WebSphere Application Server topic *topic_name* **at scope** *scope* **has been removed successfully.**

Explanation: No additional information is available for this message.

CEIIN0557E The WebSphere Application Server topic *topic_name* **at scope** *scope* **could not be removed.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0558I The WebSphere Application Server topic connection factory *topic_connection_factory_name* **at scope** *scope* **has been created successfully.**

Explanation: No additional information is available for this message.

CEIIN0559E The WebSphere Application Server topic connection factory *topic_connection_factory_name* **at scope** *scope* **could not be created.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0560I The WebSphere Application Server topic connection factory *topic_connection_factory_name* **at scope** *scope* **has been removed successfully.**

Explanation: No additional information is available for this message.

CEIIN0561E The WebSphere Application Server topic connection factory *topic_connection_factory_name* **at scope** *scope* **could not be removed.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files

or, if you ran the script manually, consult the standard output.

CEIIN0564I The WebSphere Application Server *server_name* **is in cell** *cell_name*

Explanation: No additional information is available for this message.

CEIIN0565I The WebSphere Application Server *server_name* **is a standalone server.**

Explanation: No additional information is available for this message.

CEIIN0566I The share database is specified to *scope_name* **scope** *scope_path***.**

Explanation: No additional information is available for this message.

CEIIN0567I Removing JDBC provider *provider_name* **and the data sources:** *data_source_names* **from** *scope_name* **scope** *scope_path***.**

Explanation: No additional information is available for this message.

CEIIN0568E The scope *scope* **is not valid. The scope must be set to either node or server on WebSphere Application Server in a stand-alone environment.**

Explanation: See message.

Administrator Response: Specify the scope as a server or a node.

CEIIN0569E The scope *scope* **is not valid. The scope must be set to cell, node, or server on WebSphere Application Server.**

Explanation: See message.

Administrator Response: Specify the scope as cell, node, or a server.

CEIIN0571I The JDBC provider *provider_name* **and the data sources** *data_source_names* **have been created successfully.**

Explanation: No additional information is available for this message.

CEIIN0572E The JDBC provider *provider_name* **and the data sources** *data_source_names* **could not be created.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0573I The Common Event Infrastructure JDBC provider *provider_name* **and the data sources** *data_sources* **have been removed successfully.**

Explanation: No additional information is available for this message.

CEIIN0574E Failed to remove the JDBC provider *provider_name* **and the data sources** *data_sources***.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0575I Saving the configuration.

Explanation: No additional information is available for this message.

CEIIN0576E The data source *data_source_name* **is already defined at scope** *scope_name***.**

Explanation: You cannot duplicate a data source.

Administrator Response: If you want to recreate the data source, remove the existing data source first.

CEIIN0577E The data source for the JDBC provider *provider_name* **was not found at scope** *scope_name***.**

Explanation: See message

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output. Also, verify that the WebSphere Application Server configuration is valid.

CEIIN0578I Creating the data source *data_source_name***.**

Explanation: No additional information is available for this message.

CEIIN0579I Creating the connection factory *connection_factory_name***.**

Explanation: No additional information is available for this message.

CEIIN0580I Removing the existing connection factory *connection_factory_name***.**

Explanation: No additional information is available for this message.

CEIIN0581E The data source *data_source_name* **could not be created.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0582E The authentication alias *authentication_alias* **for the Common Event Infrastructure application could not be created.**

Explanation: See message

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0583I Creating the JDBC provider *provider_name* **at scope** *scope_name***.**

Explanation: No additional information is available for this message.

CEIIN0584I The JDBC provider template *template_name* **has been found.**

Explanation: No additional information is available for this message.

CEIIN0587I Modifying the connection pool properties for the data source *data_source_name* **at scope** *scope_name***.**

Explanation: No additional information is available for this message.

CEIIN0588E The connection pool properties for the data source *data_source_name* **at scope** *scope_name* **could not be modified.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0589E The JDBC provider template *template_name* **was not found.**

Explanation: See message

Administrator Response: Check the JDBC_PROVIDER key in the response file to see if the JDBC driver name is supported by WebSphere Application Server. The JDBC provider name is case sensitive.

CEIIN0590I The data source *data_source_name* **was not found at scope** *scope_name***.**

Explanation: No additional information is available for this message.

CEIIN0591I Removing the data source *data_source_name* **at scope** *scope_name***.**

Explanation: No additional information is available for this message.

CEIIN0592I Removing the CMP connection factory *factory_name***.**

Explanation: No additional information is available for this message.

CEIIN0593I Removing the JDBC provider *provider_name* **at scope** *scope_name***.**

Explanation: No additional information is available for this message.

CEIIN0594W The JDBC provider *provider_name* **at scope** *scope_name* **cannot be removed because it contains other data sources.**

Explanation: The JDBC provider cannot be removed because it contains data sources that are not known to the Common Event Infrastructure uninstaller. For example, the user created a data source named `test` and a data source named `production` under the JDBC provider created by the Common Event Infrastructure installer.

Administrator Response: Manually remove the JDBC provider if it is not needed.

CEIIN0595I The JDBC provider *provider_name* **has been found at scope** *scope_name***.**

Explanation: No additional information is available for this message.

CEIIN0596I The JDBC provider *provider_name* **was not found at scope** *scope_name***.**

Explanation: No additional information is available for this message.

CEIIN0597I Removing the authentication alias *alias_name***.**

Explanation: No additional information is available for this message.

CEIIN0598E The Common Event Infrastructure Provider *cei_provider* **at scope** *scope* **was not found.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output. Also, verify that the installation of the Common Event Infrastructure application completed successfully and that the WebSphere Application Server configuration is valid.

CEIIN0599I The JMS transmission profile *jms_transmission_profile_name* **at scope** *scope* **has been created successfully.**

Explanation: No additional information is available for this message.

CEIIN0600E The JMS transmission profile *jms_transmission_profile_name* **at scope** *scope* **could not be created.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0601I The JMS transmission profile *jms_transmission_profile_name* **at scope** *scope* **has been removed successfully.**

Explanation: No additional information is available for this message.

CEIIN0602E The JMS transmission profile *jms_transmission_profile_name* **at scope** *scope* **could not be removed.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0606E The database *database_name* **does not exist. The database cannot be upgraded.**

Explanation: See message.

Administrator Response: If the previous version of the database exists, verify that the response file contains the correct information. If the previous version of the database does not exists, create the database using the appropriate database configuration command.

CEIIN0607E The data source *data_source_name* **does not exist. The data source cannot be upgraded.**

Explanation: See message.

Administrator Response: If the previous version of the database exists, verify that the response file contains the correct information. If the previous version of the database does not exist, create the database using the appropriate database configuration command.

CEIIN0608E The databases: *database_names* **and the data sources** *data_source_names* **cannot be upgraded.**

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files to determine the cause of the problem.

CEIIN0609I The databases: *database_names* **and the data sources** *data_source_names* **have been upgraded successfully.**

Explanation: No additional information is available for this message.

CEIIN0610I The database and data source upgrades are currently only available for the Cloudscape database.

Explanation: No additional information is available for this message.

CEIIN0611E The event catalog database *database_name* **is found. The database cannot be upgraded.**

Explanation: See message.

Administrator Response: The Common Event Infrastructure databases are up to date. Using the administrative console, verify that the event data source and the event catalog data source exist. Manually create these data sources if they do not exist. Consult the Common Event Infrastructure Developer's Guide about how to manually create the data source.

CEIIN0612E The event catalog data source *data_source_name* **is found. The data source cannot be upgraded.**

Explanation: See message.

Administrator Response: The Common Event Infrastructure databases are up to date. Verify that the database event and event_catalog databases exist in the WAS_HOME\event\CloudscapeEventDB directory. If the databases do not exist, manually create these databases. Consult the Common Event Infrastructure Developer's Guide on how to manually create the database.

CEIIN0613I The emitter factory profile *emitter_factory_profile_name* **at scope** *scope* **has been created successfully.**

Explanation: No additional information is available for this message.

CEIIN0614E The emitter factory profile *emitter_factory_profile_name* **at scope** *scope* **could not be created.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0615I The emitter factory profile *emitter_factory_profile_name* **at scope** *scope* **was removed successfully.**

Explanation: No additional information is available for this message.

CEIIN0616E The emitter factory profile *emitter_factory_profile_name* **at scope** *scope* **could not be removed.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0617E The WAS_ADMIN key value was not found in the standard input.

Explanation: The WAS_ADMIN key value is required, but it was not found in the standard input.

Administrator Response: Pass the WAS_ADMIN key value to the standard input. Consult the Common Event Infrastructure Developer's Guide for information about how to pass the WAS_ADMIN key value to standard input.

CEIIN0618E The WAS_PASSWORD key value was not found in the standard input.

Explanation: The WAS_PASSWORD key value is required, but it was not found in the standard input.

Administrator Response: Pass the WAS_PASSWORD key value to the standard input. Consult the Common Event Infrastructure Developer's Guide for information about how to pass the WAS_PASSWORD key value to the standard input.

CEIIN0619E The DB_USER key value was not found in the standard input.

Explanation: The DB_USER key value is required, but it was not found in the standard input.

Administrator Response: Pass the DB_USER key value to the standard input. Consult the Common Event Infrastructure Developer's Guide for information about how to pass the DB_USER key value to the standard input.

CEIIN0620E The DB_PASSWORD key value was not found in the standard input.

Explanation: The DB_PASSWORD key value is required, but it was not found in the standard input.

Administrator Response: Pass the DB_PASSWORD key value to the standard input. Consult the Common Event Infrastructure Developer's Guide for information about how to pass the DB_PASSWORD key value to the standard input.

CEIIN0621E The SYS_USER key value was not found in the standard input.

Explanation: The SYS_USER key value is required for the Oracle database, but it was not found in the standard input.

Administrator Response: Pass the SYS_USER key value to the standard input. Consult the Common Event Infrastructure Developer's Guide for information about how to pass the SYS_USER key value to standard input.

CEIIN0622W The SYS_PASSWORD key value was not found in the standard input.

Explanation: The Common Event Infrastructure installer and uninstaller assumes that the Oracle database was configured to log in without a valid password.

Administrator Response: If the Oracle database does not require a password, no action is necessary. If the Oracle database requires a password, pass the SYS_PASSWORD key value to the standard input. Consult the Common Event Infrastructure Developer's Guide for information about how to pass the SYS_USER key value to the standard input.

CEIIN0623E The ORACLE_ROLE key value was not found in the standard input.

Explanation: The ORACLE_ROLE key value is required for the Oracle database, but it was not found in the standard input.

Administrator Response: Pass the ORACLE_ROLE key value to the standard input. Consult the Common Event Infrastructure Developer's Guide for information about how to pass the ORACLE_ROLE key value to the standard input.

CEIIN0624E The option *option* **must be specified.**

Explanation: See message.

Administrator Response: Repeat the operation and provide the specified option.

CEIIN0625E The specified value for option *option* **is not valid.**
Specified value: *specified_value*
Valid values: *valid_value*

Explanation: See message.

Administrator Response: Specify a valid value for the option.

CEIIN0626I Usage: *usage*

Explanation: No additional information is available for this message.

CEIIN0627I A backup of the enterprise application *application_name* **has been saved to** *file_name***.**

Explanation: No additional information is available for this message.

CEIIN0628I The default resources are already defined.

Explanation: No additional information is available for this message.

CEIIN0629I The Common Event Infrastructure data sources have been removed successfully.

Explanation: No additional information is available for this message.

CEIIN0630I The JDBC provider *provider_name* **at scope** *scope_name* **contains the data source** *data_source_name***.**

Explanation: No additional information is available for this message.

CEIIN0631I The specified BackendId is not valid.
BackendId: *BackendId*
Valid BackendId List: *valid_BackendIds*

Explanation: See message.

Administrator Response: Specify a valid value for the BackendId.

CEIIN0632E The default resource for the Common Event Infrastructure application is not found.
Resource Type: *resource_type*
Scope: *scope*
Name: *name*

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output. Also, verify that the installation of the Common Event Infrastructure application completed successfully and that the WebSphere Application Server configuration is valid.

CEIIN0633E The default resource for the Common Event Infrastructure application has been modified after installation. The default resource cannot be modified again.
Resource Type: *resource_type*
Scope: *scope*
Name: *name*

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0634I The default resource for the Common Event Infrastructure application has been modified successfully.
Resource Type: *resource_type*
Scope: *scope*
Name: *name*

Explanation: No additional information is available for this message.

CEIIN0635E Modification of the default resource for the Common Event Infrastructure application failed.
Resource Type: *resource_type*
Scope: *scope*
Name: *name*

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0636E A Common Event Infrastructure service at scope *scope* **was not found.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output. Also, verify that the installation of the Common Event Infrastructure application completed successfully and that the WebSphere Application Server configuration is valid.

CEIIN0638I The registered enterprise application of the Common Event Infrastructure service at scope *scope* **is** *application_name***.**

Explanation: No additional information is available for this message.

CEIIN0639W The installer cannot modify the schema name in the default data store profile because the data store profile does not exist. You must manually create the default data store profile and set the schema name to *schema_name***.**

Explanation: See message.

Administrator Response: Check to see if the Common Event Infrastructure application is installed. Use the WebSphere Administrative Console to create the default data store profile and set the schema name.

CEIIN0640I Modifying the schema name in default data store profile from *old_schema_name* **to** *new_schema_name***.**

Explanation: No additional information is available for this message.

CEIIN0641I The authentication alias *authentication_alias* **for the Common Event Infrastructure application was created.**

Explanation: No additional information is available for this message.

CEIIN0642I There are no Common Event Infrastructure enterprise applications to update on node *node***.**

Explanation: No additional information is available for this message.

CEIIN0643E The required keyword *keyword* **was not found in response file** *file*

Explanation: No additional information is available for this message.

CEIIN0644I The Common Event Infrastructure path has been removed successfully from the WebSphere Application Server property *property_name*

Explanation: No additional information is available for this message.

CEIIN0645E The removal of the Common Event Infrastructure path from the WebSphere Application Server property *property_name* **failed.**

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files for the failure. Correct the error and issue the request again.

CEIIN0646I The Common Event Infrastructure path was not found in the WebSphere Application Server property *property_name***.**

Explanation: No additional information is available for this message.

CEIIN0647I The Common Event Infrastructure path has been set successfully in the WebSphere Application Server property *property_name***.**

Explanation: No additional information is available for this message.

CEIIN0648E An error occurred while setting the Common Event Infrastructure path in the WebSphere Application Server property *property_name***.**

Explanation: See message.

Administrator Response: Consult the Common Event Infrastructure log files for the failure. Correct the error and issue the request again.

CEIIN0649I The Common Event Infrastructure path already existed in the WebSphere Application Server property *property_name***.**

Explanation: No additional information is available for this message.

CEIIN0650I The WebSphere Application Server JMS bus *bus_name* **has been created successfully.**

Explanation: No additional information is available for this message.

CEIIN0651E The WebSphere Application Server JMS bus *bus_name* **could not be created.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0652I The WebSphere Application Server JMS bus member *member_name* **has been added to JMS bus** *bus_name* **successfully.**

Explanation: No additional information is available for this message.

CEIIN0653E The WebSphere Application Server JMS bus member *member_name* **could not be added to JMS bus** *bus_name***.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0654I The WebSphere Application Server JMS bus member *member_name* **has been removed from bus** *bus_name* **successfully.**

Explanation: No additional information is available for this message.

CEIIN0655E The WebSphere Application Server JMS bus member *member_name* **could not be removed from bus** *bus_name***.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0656I The JMS destination *destination_name* **has been created successfully.**

Explanation: No additional information is available for this message.

CEIIN0657E The JMS destination *destination_name* **could not be created.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0658I The JMS destination *destination_name* **has been removed successfully.**

Explanation: No additional information is available for this message.

CEIIN0659E The JMS destination *destination_name* **could not be removed.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0660I The WebSphere Application Server JMS Activation Specification *actspec* **has been created successfully.**

Explanation: No additional information is available for this message.

CEIIN0661E The WebSphere Application Server JMS Activation Specification *actspec* **could not be created.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0662I The WebSphere Application Server JMS Activation Specification *actspec* **has been removed successfully.**

Explanation: No additional information is available for this message.

CEIIN0663E The WebSphere Application Server JMS Activation Specification *actspec* **could not be removed.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0664I The WebSphere Application Server JMS bus *bus_name* **has been removed successfully.**

Explanation: No additional information is available for this message.

CEIIN0665E The WebSphere Application Server JMS bus *bus_name* **could not be removed.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0666I The enterprise application *application_name* **has been deployed to cluster** *cluster_name***.**

Explanation: No additional information is available for this message.

CEIIN0667I The enterprise application *application_name* **has not been deployed to cluster** *cluster_name***.**

Explanation: No additional information is available for this message.

CEIIN0668I The enterprise application *application_name* **has been installed successfully on cluster** *cluster_name***.**

Explanation: No additional information is available for this message.

CEIIN0669E The enterprise application *application_name* **failed to install on cluster** *cluster_name***.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0670I The enterprise application *application_name* **has been uninstalled successfully from cluster** *cluster_name***.**

Explanation: No additional information is available for this message.

CEIIN0671E The enterprise application *application_name* **could not be uninstalled from cluster** *cluster_name***.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0672E The cluster *cluster_name* **could not be found.**

Explanation: See message.

Administrator Response: Verify that the specified cluster exists on this WebSphere Application Server installation.

CEIIN0673E The WebSphere Application Server *server_name* **is a member of a cluster. The application** *appln_name* **cannot be installed to the specified server.**

Explanation: See message.

Administrator Response: The specified application cannot be installed to a server which is a cluster member. Either specify a server which is not a member of a cluster or specify a cluster for the installation of the application.

CEIIN0674E The WebSphere Application Server JMS Activation Specification JNDI name *actspec* **could not be found.**

Explanation: An Activation Specification with the specified JNDI name does not exist.

Administrator Response: Specify a JNDI name of an existing JMS Activation Specification.

CEIIN0675E The WebSphere Application Server Listener Port *listener_port* **could not be found.**

Explanation: See message.

Administrator Response: Specify an existing listener port.

CEIIN0676I The enterprise application *application_name* **is being installed on cluster** *cluster_name***.**

Explanation: No additional information is available for this message.

CEIIN0677E Both option *option1* **and** *option2* **cannot be specified.**

Explanation: See Message

Administrator Response: See the usage details for the script and specify the required options.

CEIIN0678E Either option *option1* **or** *option2* **must be specified.**

Explanation: See Message

Administrator Response: See the usage details for the script and specify the required options.

CEIIN0679W The Provider endpoints configuration on the Queue and Topic Connection Factories created for use by the Common Event Infrastructure are not setup correctly, because the default SIB_END_POINT_ADDRESS and SIB_END_POINT_SECURE_ADDRESS ports for the inbound messaging engine were not found on the WebSphere Application Server.

Explanation: See Message

Administrator Response: Manually configure the provider endpoints configuration on the Queue and Topic Connection Factories created for use by the Common Event Infrastructure.

CEIIN0680I The Common Event Infrastructure resource of type *res_type* **with the name** *res_name* **at scope** *scope* **has been created successfully.**

Explanation: No additional information is available for this message.

CEIIN0681E The Common Event Infrastructure resource of type *res_type* **with the name** *res_name* **at scope** *scope* **could not be created.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0682I The Common Event Infrastructure resource of type *res_type* **with the name** *res_name* **at scope** *scope* **has been removed successfully.**

Explanation: No additional information is available for this message.

CEIIN0683E The Common Event Infrastructure resource of type *res_type* **with the name** *res_name* **at scope** *scope* **could not be removed.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0684I The data store profile *data_store_profile_name* **is already defined at scope** *scope_name***.**

Explanation: No additional information is available for this message.

CEIIN0685I Creating the data store profile *data_store_profile_name***.**

Explanation: No additional information is available for this message.

CEIIN0686E The data store profile *data_store_profile_name* **could not be created.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0687I The creation of the data store profile *data_store_profile* **for the Common Event Infrastructure completed successfully.**

Explanation: No additional information is available for this message.

CEIIN0688I Updating the event server profile with the xml data store jndi name *data_store_jndi_name***.**

Explanation: No additional information is available for this message.

CEIIN0689E The event server profile *event_server_profile_name* **could not be updated with the xml data store JNDI name.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0690I The event server profile *event_server_profile* **was successfully updated with the xml data store JNDI name.**

Explanation: No additional information is available for this message.

CEIIN0691E The Event Server Profile *event_server_profile* **at scope** *scope* **was not found.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output. Also, verify that the installation of the Common Event Infrastructure application completed successfully and that the WebSphere Application Server configuration is valid.

CEIIN0692I The removal of the data store profile *data_store_profile* **for the Common Event Infrastructure completed successfully.**

Explanation: No additional information is available for this message.

CEIIN0693I Removing the data store profile *data_store_profile_name***.**

Explanation: No additional information is available for this message.

CEIIN0694E The data store profile *data_store_profile_name* **could not be removed.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0695I Removing the event server profile with the xml data store jndi name *data_store_jndi_name***.**

Explanation: No additional information is available for this message.

CEIIN0696E The profile property *event_server_profile_property* **could not be removed from the event server profile** *event_server_profile* **.**

Explanation: See message.

Administrator Response: To determine the cause of the problem, consult the Common Event Infrastructure log files or, if you ran the script manually, consult the standard output.

CEIIN0697I The profile property *event_server_profile_property* **was successfully removed from the event server profile** *event_server_profile***.**

Explanation: No additional information is available for this message.

CEIDS0027W The configuration value for the bucket check interval, in seconds, that checks the current bucket number *interval_old_value***, is not valid. The value has been changed to the default value** *interval_new_value***.**

Explanation: The XML store writes an event to the current active bucket. The bucketCheckInterval custom property determines the maximum time the XML store can go without validating if the active bucket has been changed. Because the property value was set to less than 0, the XML store uses the default value.

Administrator Response: In the administrative console, set the bucketCheckInterval custom property to an integer value that is greater than 0.

CEIXS0001E The specified relational database management system is not supported.
Database: *rdbms_name*
Version: *rdbms_version*

Explanation: See message.

Administrator Response: Configure the common event infrastructure data source to use a supported relational database management system.

CEIXS0002E The relational database management system reported the following error.
Data source resource reference: *data_source*
SQL state: *sql_state*
Vendor code: *vendor_code*
Message: *message*

Explanation: The SQL state is the standard JDBC error code for the reported problem. The vendor code is the database vendor specific error code. The message parameter is the localized error message that is returned by the relational database management system.

Administrator Response: Refer to the appropriate relational database documentation for information about the SQL state, the vendor code, and the error message.

CEIXS0004I The XML store is connecting to the specified relational database management system.
Database: *rdbms_name*
Version: *rdbms_version*

Explanation: No additional information is available for this message.

CEIXS0005I The database schema version is *actual_schema_version***.**

Explanation: No additional information is available for this message.

CEIXS0006E The database schema is not compatible with the XML store. The database schema version is *actual_schema_version***, but the required version is** *required_schema_version***.**

Explanation: See message.

CEIXS0008E The WebSphere Application Server connection pooling system returned a database connection that is not valid.
Data source resource reference: *data_source*

Explanation: See message.

Programmer response: Try the operation again.

CEIXS0009E A database connection could not be obtained from the connection pool after *maximum_retries* **attempts were made to obtain a connection.**
Data source resource reference: *data_source*

Explanation: The WebSphere Application Server connection pooling system was not able to return a database connection using the current configuration of the connection pool and the configured number of retries.

Administrator Response: Try one or more of the following actions:

- Configure the data source again and increase the maximum number of connections. This option improves performance and event throughput.
- Configure the data source again and increase the connection timeout value.
- Increase the number of retries in the XML store profile.

CEIXS0011E The XML store cannot find the resource reference in JNDI.
Resource reference: *resource_reference_name*

Explanation: The XML store uses resource references to obtain the data source that is used to obtain connections to the database and for the object that contains the configuration for the XML store. The resource reference information that is specified during the installation is not valid. For example, a JNDI name that is not valid was specified for the resource reference.

If the specified resource reference name is java:comp/env/XmlStoreProfileReference, the XML store cannot access the configuration data. If the specified resource reference name is java:comp/env/XmlDataSourceReference, the XML store cannot access the data source.

Administrator Response: In the administrative console, configure the specified resource reference to a valid JNDI resource.

CEIXS0012E The XML store is configured to use only one bucket. The current active bucket cannot be changed.

Explanation: See message.

CEIXS0014E The relational database management system reported the following error.
Data source resource reference: *data_source*
Database product: *rdbms_name*
Database version: *rdbms_version*
SQL state: *sql_state*
Vendor code: *vendor_code*
Message: *message*

Explanation: The SQL state is the standard JDBC error code for the reported problem. The vendor code is the database vendor specific error code. The message parameter is the localized error message that is returned by the relational database management system.

Administrator Response: Refer to the appropriate relational database documentation for information about the SQL state, the vendor code, and the error message.

CEIXS0015I The current bucket is being changed from bucket *old_bucket* **to bucket** *new_bucket***.**

Explanation: No additional information is available for this message.

CEIXS0016E The value *property_value* **for the property** *property_name* **cannot be converted to an integer. The default value of** *default_value* **is used.**

Explanation: See message.

CEIXS0017E The value *property_value* **for the property** *property_name* **is larger than the maximum allowed value of** *maximum_value***. The default value of** *default_value* **is used.**

Explanation: See message.

CEIXS0018E The value *property_value* **for the property** *property_name* **is less than the minimum allowed value of** *minimum_value***. The default value of** *default_value* **is used.**

Explanation: See message.

CEIXS0019I The property *property_name* **could not be located. The default value of** *default_value* **is used.**

Explanation: No additional information is available for this message.

CEIXS0020I A fast purge is in progress. Only the active bucket is accessed. The active bucket is currently bucket *active_bucket***.**

Explanation: No additional information is available for this message.

CEIXS0022E The current active bucket cannot be changed from bucket *active_bucket* **to bucket** *new_bucket* **because bucket** *new_bucket* **still contains event data.**

Explanation: See message.

CEIXS0023E The relational database management system reported the following error.
Data source resource reference: *data_source*
Database product: *rdbms_name*
Database version: *rdbms_version*
SQL state: *sql_state*
Vendor code: *vendor_code*
Message: *message*

Explanation: The SQL state is the standard JDBC error code for the reported problem. The vendor code is the database vendor specific error code. The message parameter is the localized error message that is returned by the relational database management system.

Administrator Response: Refer to the appropriate relational database documentation for information about the SQL state, the vendor code, and the error message.

CEIXS0024E The length of the XML column cannot be determined from the database metadata.

Explanation: The XML store uses the database metadata to determine length of the column. The XML store needs this information to determine whether to store the event in the main table or use both the main and the overflow tables.

CEIXS0025I The event cannot be compressed. It is stored in an uncompressed form in the database.

Explanation: See message.

Administrator Response: If the problem persists, in the administrative console, set the compress custom property to false.

CEIXS0026E The XML store cannot not convert an event from string format to binary format. The event cannot be stored.

Explanation: The XML store stores an event as binary data. It uses UTF-8 codeset to convert the string data to binary data. Because the string to binary format conversion failed, the event cannot be stored.