



Cúram 8.2.2

Development Environment Installation Guide

Note

Before using this information and the product it supports, read the information in [Notices on page 53](#)

Edition

This edition applies to Cúram 8.2.2.

© Merative US L.P. 2012, 2026

Merative and the Merative Logo are trademarks of Merative US L.P. in the United States and other countries.

Contents

Note.....	iii
Edition.....	v
1 Installing a development environment.....	9
1.1 Installing prerequisites.....	11
8.1.1 prerequisites.....	11
8.1 prerequisites.....	11
Installing and configuring Apache Ant.....	11
Installing and configuring a Java™ runtime.....	12
1.2 Installing IBM® Installation Manager.....	12
1.3 Installing an application server.....	13
Installing and configuring IBM® WebSphere® Application Server.....	14
Installing and configuring Oracle WebLogic Server.....	14
Installing and configuring IBM WebSphere® Application Server Liberty.....	15
1.4 Installing a database.....	16
H2 database.....	16
IBM® Db2® database.....	17
Oracle database.....	20
1.5 Installing the Cúram software.....	23
Installing fixes and maintainance updates.....	27
Installing the Editor Applications asset to remove browser Flash dependencies.....	28
Uninstalling Cúram.....	29
1.6 Cúram postinstallation configuration.....	29
Setting the Cúram environment variables.....	29
Setting the Cúram credentials.....	30
Configuring the H2 database.....	33
Configuring the IBM® Db2® database.....	36
Configuring the Oracle Database.....	38
Testing the configuration.....	38
Running build commands for the server and client applications.....	39
Starting the XML server.....	39
Customizing cryptography for production environment protection.....	40
Configuring <code>curam.referer.domains</code> for Cross-Site Request Forgery (CSRF) protection.....	40
1.7 Installing and configuring Eclipse and Apache Tomcat.....	41
Configuring Tomcat.....	41
Configuring the Java™ runtime for Eclipse.....	43
Configuring the Eclipse class path variables and the Eclipse Tomcat Plugin.....	44

Importing and configuring the Cúram projects in Eclipse.....	45
Starting the Tomcat server and the Cúram servers.....	47
Using Eclipse to validate the tabbed configuration artifacts.....	47
Supported Eclipse text file encoding.....	48
1.8 Installing and configuring Rational® Software Architect Designer.....	48
Installing the Cúram plug-ins.....	49
Setting up your Rational® Software Architect Designer workspace.....	49
1.9 Getting started with the development environment.....	50
Starting the server, Tomcat, and the Login Client.....	50
Logging on to Cúram.....	51
Deploying Cúram.....	52
1.10 Upgrading Cúram.....	52
Notices.....	53
Privacy policy.....	54
Trademarks.....	54

1 Installing a development environment

You can install a light or a full Cúram development environment to develop Cúram applications. Development is supported only on Microsoft™ Windows™, and you can build on both Microsoft™ Windows™ and Linux®.

Development environments rely on a Java runtime for execution. OpenJDK is the recommended Java runtime for development environments.

- **Light Java™ development environment**

A light development environment consists of Cúram, an Eclipse and Tomcat IDE, the lightweight H2 database, and the IBM® Rational® Software Architect Designer UML modeling tool.

- **Full Java™ development environment**

A full development environment replaces H2 with a supported enterprise database and adds a supported enterprise application server for full development and testing.

- **Java™ build environment**

A build environment consists of Cúram, and a supported database and application server.

- **Supplemental front-end development**

- If you want to extend UIM with JavaScript™ and the IBM® Carbon Design System, install the open source Cúram UI Addon Development Environment.

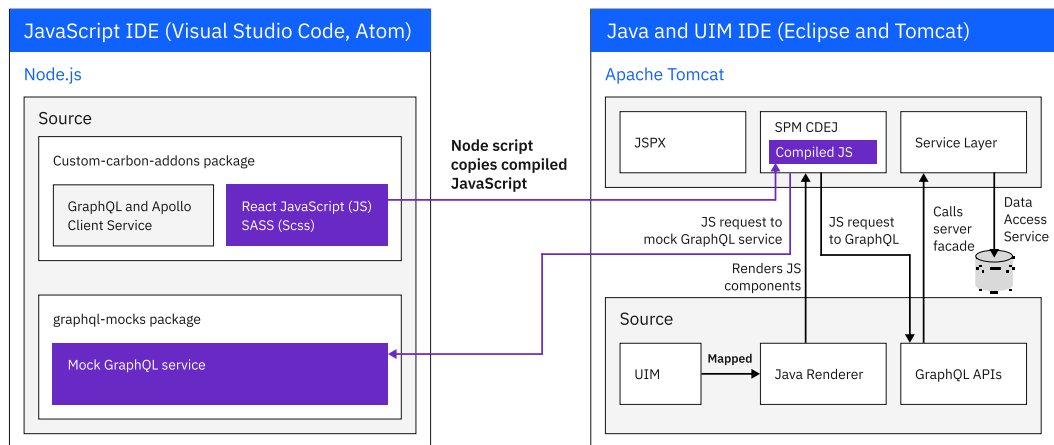


Figure 1: React and Java™ development environments for UI components

For more information, see <https://merative.github.io/spm-ui-addon-devenv>.

- If you want to work with the , install the asset.

Figure 2: React and Java™ lightweight application development environments for Universal Access

For more information about installing a React JavaScript™ development environment for Universal Access development, see the *Universal Access Responsive Web Application Guide*.

1.1 Installing prerequisites

You must install prerequisites before you install the Cúram software.

Related information

[Cúram prerequisites and supported software](#)

8.1.1 prerequisites

Supported operating systems, prerequisites, and supported software for Cúram 8.1.1.

8.1 prerequisites

Supported operating systems, prerequisites, and supported software for Cúram 8.1.0.

Installing and configuring Apache Ant

Apache Ant is a Java™ library and command-line tool for building Java™ applications.

Procedure

1. Download the Ant compressed file from the Apache website. For more information, see [Installing Apache Ant](#).
2. Extract the file to a directory of your choice on your computer. By default the file extracts to the `apache-ant-version` directory. The installation is now complete.
3. Create an ANT_HOME system environment variable with the value set to the Apache Ant installation directory.
 - a) Windows
Add `%ANT_HOME%\bin` to the PATH environment variable.
 - b) Linux
Add `$ANT_HOME/bin` to the PATH environment variable.
4. Create an ANT_OPTS system environment variable with the value:

```
ANT_OPTS=-Xmx1400m -Dcmp.maxmemory=1400m
```

Installing and configuring a Java™ runtime

You can install a Java™ runtime as a standalone installation, or use the runtime that is provided with a supported application server.

About this task

The application server becomes a prerequisite if you use the Java runtime that it provides.



OpenJDK-based distributions are supported and are the default choice unless the selected application server requires a specific Java runtime.

OpenJDK is the recommended Java runtime for development environments.

Follow the documentation for your chosen Java™ distribution to install and configure the Java™ runtime.

Regardless of which Java™ runtime you use, you must complete the following configuration steps. You might need multiple versions of a Java™ runtime installed on a single computer. If so, you can choose the scope for these Microsoft™ Windows™ environment variables. For example, system wide, or through a script file or symbolic links.

Procedure

1. Create a `JAVA_HOME` environment variable that points to the installed Java™ runtime.
 -  Place `%JAVA_HOME%\bin` at the beginning of the `PATH` environment variable.
 -  Place `$JAVA_HOME/bin` at the beginning of the `PATH` environment variable.
2. Create a `J2EE_JAR` environment variable that points to the required Java EE API JAR file: Refer to the application server documentation to find details of the Java EE API JAR file that is provided with that application server type and version. Ensure the appropriate J2EE reference is set up in your environment. This varies by application server as follows-
 - Set the `J2EE_JAR` environment variable to the installed Java™ EE JAR as specified in the WebSphere documentation.
 - Set the `J2EE_JAR` environment variable to the installed Java™ EE JAR as specified in the WebLogic documentation.
 - Set the `WLP_HOME` environment variable to your root wlp directory and create an `AppServer.properties` file (see *CuramServerDeveloperGuide*) with `as.vendor` set to “wlp”.

1.2 Installing IBM® Installation Manager

IBM® Installation Manager is used to install many IBM products. Install IBM® Installation Manager if you want to install IBM® WebSphere® Application Server or IBM® Db2®.

Procedure

1. Download the IBM® Installation Manager from www-01.ibm.com/support/docview.wss?uid=swg27025142. The different versions of IBM® Installation Manager are available if you scroll down the page.

2. Select the **Download document** hyperlink for the version of IBM® Installation Manager that you want to install.
3. On the Installation Manager (version) page, scroll down to the 'Change History' section and select the appropriate Fix Central (FC) link for Microsoft™ Windows™.
4. On the 'Select Fixes' page, select the version of IBM® Installation Manager appropriate for the version of Microsoft™ Windows™.
5. Click **Continue** to navigate to the download page.
6. Download the IBM® Installation Manager compressed file for your version of Microsoft™ Windows™.
7. Extract the downloaded file to a temporary location.
For example, *C : \Temp*
8. After the IBM® Installation Manager software is downloaded and extracted, navigate to the *Install.exe* file.
For example, *C : \Temp\IM\install.exe*.
9. Right-click on *Install.exe* and select the **Run as administrator** option.
10. On the package selection, select the checkbox for the most recent version of the IBM® Installation Manager.
11. Click **Next**.
12. Review the International Program License Agreement and select the radio button to accept the terms and continue with the installation.
13. Click **Next**.
14. Select a location for the IBM Installation Manager.
For example, *C : \IBM\Installation Manager\eclipse*.
Typically, IBM® Installation Manager, IBM® WebSphere® Application Server, and IBM® Db2® are installed in the same folder, *C : \IBM*. Do not install these programs into *C : \Program Files*, or any other location with a space in the name as it can cause problems later.
15. Click **Install**.
16. Click **Finish**.

1.3 Installing an application server

You can install an application server to give you a complete set of development tools or for a build environment. The application server is a prerequisite if you want to use the Java™ runtime that is bundled with IBM® WebSphere® Application Server or Oracle WebLogic Server.

Note: You must not install an application server on an environment where the hostname contains an underscore.

The Java runtime may be provided or constrained by the selected application server. If not using an application server provided runtime, you may install and configure a compatible Java runtime independently.

Installing and configuring IBM® WebSphere® Application Server

Install WebSphere® Application Server or IBM® WebSphere® Application Server Network Deployment from the installation media and set the required Microsoft Windows environment variable.

Before you begin

IBM® Installation Manager is used to install IBM® WebSphere® Application Server, see [1.2 Installing IBM® Installation Manager on page 12](#).

If you want to install WebSphere® Application Server as a service, you must first create a user account with administrator privileges to use as the credentials for the service.

Important:

- **Windows** Do not install WebSphere® Application Server in a directory that contains spaces in the name, such as the default *Program Files* directory, as it can cause problems later.
- Do not install the WebSphere® Application Server sample applications. The sample application Apache Derby data source results in a class path conflict with the application web client use of Derby.

Setting the WebSphere® Application Server environment variable

Set the *WAS_HOME* environment variable to the server directory of the WebSphere® Application Server installation.

Windows For example, *C:\IBM\WebSphere\AppServer*

Linux For example, */opt/IBM/WebSphere/AppServer*

Installing and configuring Oracle WebLogic Server

Complete the following steps to install and configure Oracle WebLogic Server.

Procedure

1. Run the Oracle installer. When prompted in the installation wizard, choose the following options:
 - For the installation type, choose **WebLogic Server**.
 - Deselect the **Automatically Launch the Configuration Wizard** checkbox.
2. Set the *WLS_HOME* environment variable to the *server* directory of the WebLogic Server installation.

Windows For example, *server_directory\wlserver_version\server*

Linux For example, */opt/wlserver/wlserver/server*.

3. Configure the WebLogic Server application transaction timeout settings, see [Configuring the Oracle WebLogic Server application transaction timeout settings on page 15](#).

Configuring the Oracle WebLogic Server application transaction timeout settings

Set the minimum application transaction timeout settings for Oracle WebLogic Server to 120 seconds.

About this task

Warning: These settings are for testing or development and are not advised for production systems where they represent minimum values. Tuning must be done to find the correct settings for your environment. The minimum value is recommended for the following components:

- Cúram Child Welfare
- Cúram Income Support
- Cúram Income Support for Medical Assistance
- Cúram Workers Compensation
- Cúram Youth Services

Procedure

1. Log on to the localhost console from `https://localhost:7002/console`.
2. Select the domain name under the **Domain Structure** section.
3. Select the **Configuration** tab.
4. Select the **JTA** tab.
5. Set **Timeout Seconds** to be 120 seconds.

What to do next

To test these components on Oracle WebLogic Server during development cycles, you can use the same steps to increase the JTA timeout settings.

Installing and configuring IBM WebSphere® Application Server Liberty

Install WebSphere® Application Server or IBM® WebSphere® Application Server Liberty from the installation media and set the required Microsoft Windows environment variable.

Before you begin

IBM® WebSphere® Application Server Liberty can be installed either using IBM® Installation Manager, see [1.2 Installing IBM® Installation Manager on page 12](#) or via archive extraction.

Important:

- **Windows** Do not install WebSphere® Application Server in a directory that contains spaces in the name, such as the default *Program Files* directory, as it can cause problems later.
- You need the **core** and **base** application packages only.
- Do not install the ND version of Liberty, as Cúram only supports creation of containers from the non-ND version.

Setting the WebSphere® Application Server Liberty environment variable

Set the *WLP_HOME* environment variable to the server directory of the WebSphere® Application Server installation.

Windows For example, *C:\IBM\WLP*

Linux For example, */opt/ibm/wlp/*

1.4 Installing a database

The H2 database is supported as a development database. IBM® Db2® and Oracle Database are supported as database servers.

Note: No particular character set is required for the installation and setup of the DBMS. Configure a character set that is appropriate for the character range that is needed in the application.

H2 database

H2 is an SQL database engine written in Java™ that implements the JDBC API. A browser-based console application is included. The H2 database is preinstalled with the Cúram software.

After you install the Cúram platform software, the self-contained database is in the *%CURAMSDEJ%\drivers\h2<version_number>.jar* file (where *<version_number>* is the version of the H2 driver) file.

If you plan to use the H2 database, select the IBM® Db2® option during installation. Enter values for Db2® as you proceed through the wizard. After you complete the installation, you must edit the database properties in the *%CURAM_DIR%\EJBServer\project\properties\Bootstrap.properties* file for the H2 database instead. *%CURAM_DIR%* is the Cúram installation directory, by default *C:\Merative\Curam\Development*.

These limitations apply to the support of the H2 database:

- For development use only.
- Not supported at run time.
- EAR files cannot be built for the H2 database.

- You cannot run the **configure** target while the H2 database is in use. This target automatically configures the application server.

Related information

[H2 Database Engine](#)

IBM® Db2® database

IBM® Db2® is supported as a database server. IBM® Db2® Community Edition may also be used in Cúram development and testing. The following installation and configuration steps work for both editions of Db2.

Note: It is possible to use Cúram against a remote database with the Db2® Universal Type 4 Driver. The driver is supplied with the Server Development Environment for Java™ (SDEJ).

Db2® database encoding options

If you plan to install IBM® Db2®, read this important background information about issues with Db2® database encoding and related sizing information. During a Db2® installation, you must identify your requirement for SBCS or MBCS data. Depending on your choice, you might have to complete some extra post-configuration steps before you build the Cúram database.

What is the issue?

For a multi-byte character set (MBCS) or encoding, Db2® processes columns by their byte size, not their character length. Therefore, for multi-byte characters, a CHAR, VARCHAR, or CLOB column might store fewer characters than the column length specification indicates, depending on the actual character length.

Consider the following example:

- A CHAR or VARCHAR column that is modeled with a length of 16.
- The 16-character string, "Marge says hello" that does not have accented character, requires 16 bytes for storage in a single-byte character set (SBCS).
- A similar 16-character string, but with accented characters, "Márge says hélló", requires 18 bytes for storage in UTF-8, a multi-byte character set (MBCS).

For the single-byte data, the string fits and processing is successful. For the multi-byte data, the string does not fit, resulting in overflow errors at run time. The Cúram web client usually captures and reports field size errors in a user-friendly manner. In this case, the user receives an "unhandled server exception" error, which is an underlying SQL Code -302 error. This is because the client does not capture this size mismatch as it checks the number of characters, and not the byte length.

How Cúram addresses the issue

Cúram provides modeling and build-time capabilities to resize its database columns to address this issue. These capabilities are described further in the *Modeling Reference Guide* and *Server Developer's Guide*.

As Cúram provides support for multiple languages, support for MBCS data is enabled by default with the maximum expansion set. These expansion settings are appropriate to ensure that new users, testing environments, and so on, do not encounter any errors because of their language, encoding, and database sizing. Also, users can find they require MBCS data when they import or paste data from other applications into their Cúram system. However, these defaults might not be appropriate for all environments. The following section describes some considerations for changing these expansion settings.

What you must consider

It is important to carefully consider your data encoding requirements regarding Db2® and Cúram to avoid unexpected behavior with how the database stores characters.

The preceding example represents a boundary case in that the data length matches the maximum column width. In many cases it is unlikely that, even with MBCS characters, an overflow situation will occur. Most data does not reach the maximum defined size. However, you must be prepared for the possibility of these error situations.

Use the database character set encoding appropriate to your application and environment. If possible, consider using an SBCS and encoding that supports your requirements. For example, CP1252 supports most Western European characters. However, CP1252 (or other SBCS encodings) might not support characters from different or "broader" character sets or encodings (for example, UTF-8) that users might copy and paste into their browser.

When installing your Db2® database, you must only identify your requirement for SBCS or MBCS data and be prepared to take appropriate action before you build your Cúram database:

- If you require characters that use multiple bytes, then you must consider whether the default Cúram settings are appropriate. The necessary database space is dependent on various factors such as the following factors.
 - The specific character sizes. In Db2® and Db2® for z/OS, MBCS data can range from 1 to 4 bytes.
 - The frequency of MBCS characters, which can depend on the application, language, locale, column usage within the application, and so on.
 - The information density of the language and locale. For example, while some languages can require more bytes per character, each character can represent more information than, for instance, an alphabetic character and might fit into a field without any size adjustment.

If an SBCS is adequate, plan to disable database expansion. For more information about MBCS data sizing considerations, see the *Server Developer's Guide*.

Installing IBM® Db2®

Ensure that your account has administrative privileges and then follow the Db2® installer instructions to complete a default installation. You do not need to manually create a Db2® database. The platform software provides Ant scripts that you can run as a postinstallation step to create a basic test database.

Note the following options that are presented during a default installation:

- The *name* and *password* of the administrator account. Use an account and password that meets the standards and requirements of your site and Db2®.

Windows If it is an existing user, that user must be a member of the Administrator group.

Linux The specified user must be a Linux® user on your system.

- Certain editions of the Db2® installer support federated databases. If the installer presents an option that is defaulted to **This machine will be the instance-owning database partition server**, then change this option to **This machine will be a single-partition database server**.
- You must choose MBCS or SBCS, depending on your requirements. If you are unsure of what database encoding option to select, see [Db2® database encoding options on page 17](#).

For more information, see [Installing Db2®](#).

Configuring for circular transaction logging

When you use a database with circular transaction logging enabled, certain transactions can exceed the available log file space and fail. To avoid this issue, either use archive logging or set the available log size and quantity appropriately until it meets the needs of the transaction.

About this task

A common point for this failure is when the *prepare.application.data* Ant target is running, as this target publishes all the CER rule sets on the system. This Ant target is typically run after a clean database build. If the log is too small, it can result in an SQLCODE -964 error.

You can use the following example to help you to increase the Db2® log file size and quantity. The exact amount of log file storage that is required varies from system to system. For more information about increasing the number and size of the log files available, see the specific documentation for your database.

Procedure

1. Open a command prompt and enter `db2cmd`.
2. Enter the following command:

```
db2 connect to db_name user db_user_name using db_password
```

Where *db_name*, *db_user_name*, *db_password* are the credentials of the database.

3. Enter the following commands:

```
db2 update db cfg for db_name using logfilesiz log_file_size
```

```
db2 update db cfg for db_name using logprimary primary_log_files
```

```
db2 update db cfg for db_name using logsecond secondary_log_files
```

Where the temporary values are as follows:

- The log file size. Set *log_file_size* to 1024.
- The number of primary log files. Set *primary_log_files* to 50.
- The number of secondary log files. Set *secondary_log_files* to 100.

4. Restart the database by entering the following commands:

```
db2stop  
db2start
```

Oracle database

Oracle database is supported as a database server.

Note: You can use a remote database by using the Oracle Type 4 Driver that is supplied with the SDEJ.

Oracle database encoding options

If you plan to install Oracle, it is important to consider the character set for the data that you plan to store in your database when you configure the database for use with Cúram.

For Oracle, there are two parameters to consider: NLS_CHARACTERSET and NLS_LENGTH_SEMANTICS.

- The NLS_CHARACTERSET parameter details the allowable character set of any data that is loaded to the database, generally AL32UTF8 is recommended by Oracle.
- The NLS_LENGTH_SEMANTICS determines how Oracle interprets length specifiers on CHAR and VARCHAR columns. To handle supplementary characters, for example, ß in German, where the storage of the character would be 2 bytes and might overrun the length of a defined column, set the NLS_LENGTH_SEMANTICS parameter to CHAR. This setting directs the database to size columns with a character length rather than byte length.

Oracle extended data types

Starting with Oracle Database 12c, the maximum value of the VARCHAR2 data type can be extended from 4000 to 32767 bytes by changing the MAX_STRING_SIZE initialization parameter from its default value of STANDARD to EXTENDED. However, take care when enabling Oracle extended data types as there are some important caveats that apply. For more information about setting and using extended data types, see the Oracle documentation.

Important: The process of switching to use Oracle extended data types is a one-way operation. After you switch to extended data types, you can't switch back without some form of database recovery. You must ensure that extensive regression testing is performed before switching in production environments.

Postinstallation configuration for the Oracle database

Complete the following postinstallation tasks on the Oracle Database.

Depending on the chosen Oracle architecture

1. When using a Pluggable Database (PDB), ensure that the database service name is used when configuring the connection.
2. When using the CDB root, ensure that the configured database username matches the created user, including any required prefix (for example, C##CURAM).

The database connection properties must align with how the database user and roles are created

Creating an Oracle role for application servers

The application needs certain privileges to use the Oracle XA interface. Later, when you configure the application, the username under which the server connects to Oracle is specified. The appropriate privileges must be assigned to this username for the server to work successfully.

About this task

Note: Important.

Oracle 21c introduces enforcement of Multitenant Architecture (CDB/PDB), which may impact how database users and roles are created during Cúram installation.

Depending on how your Oracle database is configured, you may need to adjust either your SQL scripts or your connection context:

1. If executing scripts within a Pluggable Database (PDB), existing SQL statements can be used without modification.
2. If executing scripts in the Container Database root (CDB\$ROOT), database users and roles must be created as common objects using a C## prefix.

Ensure you understand your Oracle deployment model before running the configuration steps below.

To bundle together the various privileges that are needed, you can create an Oracle role. Privileges can be granted to this role. Later this role can be granted to your users, granting all the privileges that are associated with that role.

The following commands create a role that is called CURAM_SERVER and give it the necessary privileges. A CURAM_USER user is then assigned that role and given the password PASSWORD. You run these commands inside an Oracle SQL*Plus window.

Oracle Multitenant context

The following SQL statements must be executed in the correct Oracle container.

Recommended approach:

```
ALTER SESSION SET CONTAINER = <PDB_NAME>;
```

When connected to a Pluggable Database (PDB), local users and roles can be created without modification.

If scripts are executed in the CDB root (CDB\$ROOT):

1. Users and roles must be created as common objects.
2. Names must be prefixed with C## (for example, CREATE ROLE "C##CURAM_SERVER").

Common users and roles are shared across all PDBs and should only be used where required by site policy.

Procedure

1. To run the commands from SQL*Plus, type the following command at a command line:

```
sqlplus ?/? as SYSDBA
```

2. Run the following SQL commands:

If executing within a Pluggable Database (PDB):

```
CREATE ROLE "CURAM_SERVER";
GRANT RESOURCE TO "CURAM_SERVER";
```

Windows @%ORACLE_HOME%\RDBMS\ADMIN\xaview.sql

Linux
@\${ORACLE_HOME}/rdbms/admin/xaview.sql

```
GRANT SELECT ON V$XATRANS$ TO PUBLIC;
GRANT SELECT ON PENDING_TRANS$ TO PUBLIC;
GRANT SELECT ON DBA_2PC_PENDING TO PUBLIC;
GRANT SELECT ON DBA_PENDING_TRANSACTIONS TO PUBLIC;
GRANT EXECUTE ON DBMS_SYSTEM TO CURAM_SERVER;
CREATE USER CURAM_USER IDENTIFIED BY
PASSWORD DEFAULT TABLESPACE "USERS" TEMPORARY TABLESPACE "TEMP";
GRANT "CONNECT", "CURAM_SERVER", UNLIMITED TABLESPACE TO <CURAM_USER>;
```

Where CURAM_USER and PASSWORD are the database user credentials.

If executing in the CDB root (CDB\$ROOT):

```
CREATE ROLE "C##CURAM_SERVER";
GRANT RESOURCE TO "C##CURAM_SERVER";
```

Windows @%ORACLE_HOME%\RDBMS\ADMIN\xaview.sql

Linux
@\${ORACLE_HOME}/rdbms/admin/xaview.sql

```
GRANT SELECT ON V$XATRANS$ TO PUBLIC;
GRANT SELECT ON PENDING_TRANS$ TO PUBLIC;
GRANT SELECT ON DBA_2PC_PENDING TO PUBLIC;
GRANT SELECT ON DBA_PENDING_TRANSACTIONS TO PUBLIC;
GRANT EXECUTE ON DBMS_SYSTEM TO C##CURAM_SERVER;
CREATE USER CURAM_USER IDENTIFIED BY
PASSWORD DEFAULT TABLESPACE "USERS" TEMPORARY TABLESPACE "TEMP";
GRANT "CONNECT", "C##CURAM_SERVER", UNLIMITED TABLESPACE TO <CURAM_USER>;
```

Where CURAM_USER and PASSWORD are the database user credentials.

Note:

If creating users in the CDB root, the CURAM_USER may also need to follow the common user naming convention (for example, C##CURAM), depending on site policy.

Configuring for circular transaction logging

When you use a database with circular transaction logging enabled, certain transactions can exceed the available log file space and fail. To avoid this issue, either use archive logging or set the available log size and quantity appropriately until it meets the needs of the transaction.

A common point for this failure is when the *prepare.application.data* Ant target is running, as this target publishes all the CER rule sets on the system. This Ant target is typically run after a clean database build.

For information about increasing the number and size of the log files available, see the specific documentation for your database. The exact amount of log file storage that is required varies from system to system.

Configuring redo log space

Certain Cúram transactions that have significant insert activity are affected by the available redo log space. To avoid this issue, allocate the appropriate redo log space for your system.

A common point for this failure is when the *prepare.application.data* Ant target is running, as this target publishes all the CER rule sets on the system. This Ant target is typically run after a clean database build.

For information about allocating the appropriate size for the redo logs, see the Oracle documentation. The exact amount of activity and the required redo log space varies from system to system.

1.5 Installing the Cúram software

For Long Term Support (LTS) releases, or Continuous Delivery (CD) releases with full development installers, run the Cúram platform installer to install the base platform on which all the other modules are installed. Then, install each of the Cúram application modules that you are licensed for. For Continuous Delivery (CD) releases with delta installers, run the installer on the appropriate full version.

Before you begin

Cúram software and maintenance releases are available to download from [Cúram Support](#).

LTS releases contain the Cúram Platform and all of the Cúram application modules. LTS releases are delivered with full development installers.

CD releases are delivered either with full development installers, denoted by the second digit being incremented such as 8.1.0, or delta installers, such as 8.0.3, which must be installed on the appropriate full release.

Ensure that you have the required installers and all of the required information before you start the installation:

- Review the Cúram release notes.
- Review the Cúram Security Bulletins. Security Bulletins are now available from Cúram Support. You must be a technical contact to access Security Bulletins, open a support case if you need access. Select **SPM Software Download** and enter your technical contact credentials to access Security Bulletins under Knowledge articles.

Note: The Structured Decision Making (SDM) Add-on module is not translated and must not be installed with Cúram Child Welfare on a non-English language installation.

Cúram Income Support has a dependency on Cúram Outcome Management. If you plan to install Cúram Income Support, you must install Cúram Outcome Management.

Cúram Income Support for Medical Assistance does not have a dependency on Cúram Outcome Management.

Note: If you are installing a modification release, install it in a new installation location and not in a previous installation location.

About this task

Cúram software and maintenance fix releases are available from the [Merative Software Downloads](#) site. You must request access to download software.

For a full installation, you must download the Cúram platform installer, plus individual installers for each component you plan to install. Typically, you install application modules, and optional associated add-ons.

Before you start, you must have the following information:

- A list of the components of the application for which you are licensed.
- The organization name and address.
- The database server name, port number, database name, database username, and database password.

If your project is stored in a source-controlled environment, you might take the following approach to the installation, depending on your requirements:

- Install Cúram and any optional components.
- Place the installed code base under source control.
- To support future installations, the files in the */Installer* folder must also be maintained under source control.
- Use your source-control procedures to distribute the environment to other developers.

During the installation, all installation process and the installation history are saved to the following log files:

- */Installer/CuramInstaller.log*
- */Installer/Installhistory.txt*

Procedure

1. Go to [Cúram Support](#), locate the appropriate downloads for your version, and download the software.
2. Extract the files to a temporary directory and change to that directory.
3. To run the installer wizard, enter the command `java -jar <installername.jar>`. If the installer name includes spaces, put the name in quotation marks.

For example, `java -jar "Merative Curam SPM Installer Development.jar"`.

By default, the platform installer also installs the following platform application modules:

- Cúram Verification Engine
- Cúram Evidence Broker
- Cúram Life Event Management

You must have a valid license to use the platform application modules.

4. On the welcome page, click **Next**.
5. Do not alter the default installation path. Accept the default installation path by clicking **Next > OK** to create the directory.
If the target installation directory exists, you are prompted to overwrite the existing files.
6. Select a language. If left blank, the language defaults to `English - US`.
7. If you are licensed for Universal Access, select the checkbox for any extra languages needed.
8. Click **Next** to accept the license type.
9. The components that you are licensed for are displayed. Confirm the components that you require are selected and click **Next**.
10. Enter the **Organization Name** and **Organization Address**, and click **Next**.
11. Select the **Cúram Database Platform** that you plan to use with the application.

For example, **DB2/UDB**.

If you intend to use the H2 database, select **DB2/UDB** during the installation. Enter values for Db2® so you can proceed through the wizard. After you complete the installation, you must edit the database properties in the `%CURAM_DIR%\EJBServer\project\properties\Bootstrap.properties` file for the H2 database instead. `%CURAM_DIR%` is the Cúram installation directory, which by default is `C:\Merative\Curam\Development`.

12. In the **Database Account Logon** field, enter the value as defined during the database installation and click **Next**.

Note: This is the database user name. The database password will be configured in a later step.

13. In the **Database Server Name** field, enter the fully qualified hostname of the computer on which you installed the database. Enter a value for the **Database Server Port** field, such as 50000 for Db2®. Enter the database name in the **Curam Database Name** and click **Next**.
The installation files are extracted. This step can take several minutes.
14. When the extraction completes, click **Next**.
The installation files are configured based on the inputs that are provided in the previous steps.
15. Click **Next > Done** to complete the platform installation. Review the release notes and complete any relevant postinstallation steps.
16. Run the installers for each of the application modules for which you have a valid license:
 - Merative™ Cúram Universal Access

- Cúram Outcome Management
- Cúram Provider Management
- Cúram Social Enterprise Collaboration
- Cúram Business Intelligence and Analytics
- Cúram Appeals
- Cúram Workers Compensation - Do not install any further modules with this module.
- Either Cúram Income Support or Cúram Income Support for Medical Assistance (but not both). You can then install these modules in this sequence:

1. Cúram Income Support Screening
 2. Cúram Business Intelligence and Analytics Reports for Income Support
- Cúram Child Welfare. You can then install:
 1. Cúram Business Intelligence and Analytics Reports for Child Welfare
 2. Cúram Child Welfare Structured Decision Making (SDM) Add-on
 3. Cúram Outcome Management Structured Decision Making (SDM) Add-on

Note: Cúram Outcome Management Structured Decision Making Add-on is installed on Cúram Child Welfare only for a modular Child Welfare licensing strategy and where only the Cúram Child Welfare Case Management module of Child Welfare is installed. Structured Decision Making Add-on includes only the SDM assessments that relate to an Outcome Management plan within an ongoing Child Welfare case.

- Cúram Youth Services
 - Any additional assets such as demonstrations
17. Verify your installation by checking the installation history to determine what was installed. A text file for each installer and the *InstallHistory.txt* file that lists all of the installers that ran are found in the installation folder.
For example, *C:\Merative\Curam\Development\Installer*.
 18. Review the release notes for each of the application modules and complete any postinstallation steps that are relevant to your configuration.
 19. If you plan to develop reports for Cúram Business Intelligence and Analytics, complete any additional installation steps to install your reporting development environment, see the *Business Intelligence and Analytics Guide*.
 20. If you plan to use Citizen Engagement, you must also install the development environment. For more information, see the *Universal Access Responsive Web Application Guide*.

Related information

[Cúram prerequisites and supported software](#)

Installing fixes and maintenance updates

After you install a Cúram Long Time Support (LTS) or Continuous Delivery (CD) release, install the most recent maintenance updates.

About this task

Cúram software and maintenance releases are available from [Cúram Support](#). You must request access to download software.

- Maintenance on an LTS release is delivered in Fix packs that include critical and significant defects and security updates, and increment the fourth digit of the release number. For example, 8.0.1.1. They do not contain new features or enhancements. Updates to an LTS release do not force an upgrade impact unless it is unavoidable for reasons of security or stability. Fix Packs are delivered in a delta installer.
- Maintenance on a CD release is delivered in the next CD release. Critical defects and security patches are also delivered separately on the two most recent CD releases.

CD releases deliver maintenance updates, plus new optional features and functions. They are cumulative and contain all the fixes and functions in the previous CD releases. CD releases are delivered either with full development installers, denoted by the second digit being incremented such as 8.1.0, or delta installers, such as 8.0.3, which must be installed on the appropriate full release. New features and functions in CD releases either have no impact, or are disabled by default.

- You can choose to benefit only from the maintenance updates, without incurring any impact from the new features and functions.
- You can choose to enable new features and functions on a case-by-case basis, and handle the associated impacts, if any.
- Occasionally, obsolete JAR files can remain in your development environment after you install a maintenance update. For example, if you upgrade from a Fix Pack to a later Fix Pack by using the delta installers, JAR files that were removed in the later Fix Pack might remain in your environment. To avoid future issues with these files, we recommend that you manually remove them.

For more information about updated or removed JAR files, read the release notes for your version and see the *Product Overview Guide*.

Procedure

1. Read the Cúram release notes for the update. Take note of any preinstallation steps, requirements, restrictions, installation steps, and postinstallation steps that might apply.
2. Go to [Cúram Support](#), locate the appropriate download for your version, and download the software.
3. Extract the contents of the .zip file. The installer is in the *INSTALLER* directory.
4. Ensure that files in your Cúram installation are writable.
5. Run the installer.
6. When prompted to move obsolete files, select **Yes**.
Moving the files can take some time, during which no progress indicator is displayed.

7. Click **Finish** to complete the installation.
8. Complete any postinstallation steps.
9. After installing the release, run the appropriate build targets for your installation.
10. Manually remove obsolete JAR files, if needed.

Related information

[Cúram prerequisites and supported software](#)

Installing the Editor Applications asset to remove browser Flash dependencies

As browsers no longer support Adobe™ Flash, you must replace the editors that depend on Adobe™ Flash in the browser with stand-alone versions that run on Adobe™ Air. To replace the editors, you must download and run the Editor Applications asset installer.

About this task

The Editor Applications asset enables stand-alone versions of editors that previously required Adobe™ Flash support in the browser for Cúram. A client installation registers the editors so you can start them by using a URL. In addition, the Cúram development environment is updated to automatically start the new stand-alone editors by default. The following editors are updated:

- Cúram Express Rules (CER) Editor
- Data Mapping Editor
- Data Store Editor
- Intelligent Evidence Gathering (IEG) Editor
- Decision Matrix Editor
- Dynamic Evidence Editor

Procedure

1. The Editor Applications asset is available to download from [Cúram Support](#). Download the asset and extract the contents, which are described in this table.

Table 1: The Editor Applications asset contents

File	Description
<i>EditorApplicationsClient.msi</i>	The installer for the editors.
<i>Readme.txt</i>	A brief description of the asset and how to install it.

2. You must run a separate client installation, the *.msi* file, on any client on which the editors are to run. The installation is not available from a running server so the installation must be distributed. To install the editors, complete the following steps on each client computer:
 - a) Copy *EditorApplicationsClient.msi* to the computer.
 - b) Double-click *EditorApplicationsClient.msi*.
 - c) Select the default location for the installation.
3. Validate that the editors are installed correctly.

- a) Access the newly deployed development server.
- b) Open any editor, for example the Data Mapping Editor. The editor's start tab displays as blank or with a spinner. A dialog box is displayed to request that you want to start the editor.
- c) Indicate that you want to start the editor. The editor starts as a separate, full-screen application. In your browser, the editor's start tab shows that the editor started correctly.

Results

You can now use the new editors during development. For more information about accessibility for the editors, see the *Product Overview Guide*.

Some known limitations apply to the standalone Cúram Express Rules Editor. For more information, see the *Troubleshooting Guide*.

Uninstalling Cúram

An uninstallation file is created in the `<install>\Uninstaller\uninstaller.jar` directory during a Cúram installation. You can use this file to uninstall the application.

About this task

JAR files might be recognized as executable by being associated with a suitable launcher, such as `javaw`. If this is the case for your operating system, start the Cúram Uninstaller with the standard method that is supported by your operating system. For example, double-clicking the Cúram Uninstaller file.

Note: The uninstaller does not reset any system variables that are set by a previous installation.

Procedure

1. Navigate to the `<install>\Uninstaller\` directory.
2. Double-click the `uninstaller.jar` file to uninstall Cúram.

1.6 Cúram postinstallation configuration

Complete any required postinstallation configuration tasks after you install the Cúram software.

Setting the Cúram environment variables

Before you proceed, you must run a script to set some required Cúram environment variables.

Procedure

1. Change to the `%CURAM_DIR%` directory.

`%CURAM_DIR%` is the Cúram installation directory, which by default is `C:\Merative\Curam\Development` or `opt/merative/Curam/Development`.

2. Run the following command:

Windows

```
SetEnvironment.bat
```

Linux

```
SetEnvironment.sh
```

Setting the Cúram credentials

Before you begin

Before you proceed, you must run scripts to define a number of required Cúram credentials.

Some passwords, for example, `curam.db.password` in `Bootstrap.properties` and `security.password` in `AppServer.properties`, are stored in their encrypted form. Encrypting them requires a secret key contained in a keystore. Decrypting them at runtime will use the same keystore. The keystore is not delivered by the installer, the steps to create it are described below.

Other passwords such as passwords for Cúram application users are digested before being stored on the database so it will be necessary to regenerate these digests.

About this task

Procedure

1. Change to the `%SERVER_DIR%` directory. `%SERVER_DIR%` is the server directory of the Cúram installation directory, which by default is `C:\Merative\Curam\Development\EJBServer` or `opt/merative/Curam/Development/EJBServer`.
2. If you have customized any settings in `EJBServer/project/properties/CryptoConfig.properties` you should now re-apply them to the OOTB version which was delivered by the installer wizard.

Note:

Property `'curam.security.crypto.cipher.keystore.storepass'` is mandatory and must be added to `CryptoConfig.properties` if not already present.

3. Run the following command to create a new keystore:.

Note:

The value `'<keystore-password>'` for `'sys.keystore.password'` and the value of property `'curam.security.crypto.cipher.keystore.storepass'` in `CryptoConfig.properties` must match.

Windows

```
build.bat createkeystore installcryptojar -Doverwrite.keystore=true -
Dsys.keystore.password=<keystore-password>
```

Linux

```
build.sh createkeystore installcryptojar -Doverwrite.keystore=true -
Dsys.keystore.password=<keystore-password>
```

This produces files CuramSample.keystore and CryptoConfig.jar .

These files contain the keystore and CryptoConfig.properties which are necessary to encrypt/decrypt passwords and login to the application. These must be distributed as part of deploying the application.

Note: The `createkeystore` target must use the same Java runtime implementation and version as the deployed application. This is to ensure that it creates a JCEKS keystore which is compatible with the deployed application

If your installation utilizes more than one Java runtime implementation then you must copy the newly created key from its keystore into the keystore of the other JDK(s).

This process is described in detail in the *Cúram Security Handbook* .

4. Run the following command to define and update the credentials required for the Cúram software:.

Windows

```
build.bat init_passwords
```

Linux

```
build.sh init_passwords
```

This interactive script will prompt you to supply a number of usernames and/or passwords. The passwords will be encrypted for you and written to the relevant source files. The script will update the following credentials:

Credential Name	Description	Files Updated
curam.db	The database credentials	<SERVER_DIR>/ project/properties/ Bootstrap.properties
security	The application server credentials	<SERVER_DIR>/ project/properties/ AppServer.properties
keystore	The XML Server keystore password	<CURAMSDEJ>/xmlserver/ TLSKeystore.properties

curam.security.credentials.dbtojms	The DBTOJMS password	<SERVER_DIR>/components/ core/data/initial/USERS.dmx <SERVER_DIR>/components/ core/properties/ Application.prx
curam.security.credentials.ws	The web services (WEBSVCS) password	<SERVER_DIR>/components/ core/data/initial/USERS.dmx <SERVER_DIR>/components/ core/properties/ Application.prx
curam.security.credentials.async	The SYSTEM password	<SERVER_DIR>/components/ core/data/initial/USERS.dmx <SERVER_DIR>/ project/properties/ AppServer.properties <SERVER_DIR>/components/ core/properties/ Application.prx <CURAMSDEJ>/ear/templates/ ibm-application-bnd.xml BIApp/CuramBIRTViewer/ear/ */META-INF/ibm-application- bnd.xml BIApp/CuramBIRTViewer/ear/ */META-INF/ibm-application- bnd.xmi
curam.ldap	The LDAP server credentials	<SERVER_DIR>/components/ core/properties/ Application.prx
curam.meeting.request.reply	The Curam meeting requests credentials	<SERVER_DIR>/components/ core/properties/ Application.prx
internal.user	The password for all internal Curam users	<SERVER_DIR>/components/*/ data/*/USERS.dmx
external.user	The password for all external Curam users	<SERVER_DIR>/components/*/ data/*/EXTERNALUSER.dmx

Credentials can also be set in a non-interactive way by providing them in a comma-separated list in the format credential:username:password. See example:

Windows

```
build.bat init_passwords -Dcredentials=
"curam.db:db2admin:db2admin, curam.ldap:ldapuser:ldappassword,
internal.user:userpassword"
```

Linux

```
build.sh init_passwords \
-Dcredentials=curam.db:db2admin:db2admin,internal.user::userpassword
```

Windows

Configuring the H2 database

Complete the following postinstallation steps for the H2 database. To use the H2 database, you must update the *Bootstrap.properties* file with the correct credentials to connect to the H2 database. Ensure that you encrypt the password.

About this task

For example, here is typical H2 database content from a *Bootstrap.properties* file.

```
curam.db.type=h2
curam.db.name=curamdb
curam.db.username=curam
curam.db.password=qqnscP4c4+s=
# H2 directory.
# Default is home directory
# (i.e. C:/Documents and Settings/<username>). (Optional)
curam.db.h2.directory=C:/H2
# Mode remote|embedded
curam.db.h2.mode=embedded
# For remote mode also specify:
curam.db.serverport=9092
curam.db.servername=localhost
# Lock Time Out in ms. Default is 1000, i.e. 1 second. (Optional)
curam.db.h2.locktimeout=20000
# Property to specify NON_KEYWORDS, comma delimited list of words which H2 will ignore
# as key words during a database build. The default value for this property is
# CURRENT_DATE,CURRENT_TIMESTAMP,KEY,MONTH,OFFSET,VALUE,YEAR
curam.db.h2.non.keywords=<place keywords here>
```

After you update the *Bootstrap.properties* file and rebuild the server and database, you can develop in the same way as you would with Oracle or Db2®.

For more information about the *Bootstrap.properties* file, see the *Server Developer's Guide*.

Procedure

1. Edit the *Bootstrap.properties* file.
2. Ensure that each of the database properties has the correct values for the H2 database.

Encrypting passwords

You must encrypt passwords before you put them in the *Bootstrap.properties* file.

Procedure

1. Open a command prompt and change to the `%CURAM_DIR%\EJBServer` directory.
`%CURAM_DIR%` is the Cúram installation directory, which by default is `C:\Merative\Curam\Development`.

2. Issue the following command:

```
build encrypt -Dpassword=password
```

where *password* is the password you want to encrypt.

3. Copy the encrypted string in the output to the correct location in the *Bootstrap.properties* file.
For example, the *curam.db.password* parameter.

Setting the H2 mode

Set your preferred mode for developing applications.

About this task

The following H2 modes are supported for application development:

- **Embedded mode**
In embedded mode, an application opens the database from within the same JVM by using JDBC. This mode is the fastest and easiest connection mode. The disadvantage is that a database can be open in only one virtual machine (and class loader) at any time.
- **Remote mode**
In remote mode, sometimes called client/server mode, an application opens the database remotely by using the JDBC or the ODBC API. Many applications can connect to the same database at the same time. The remote mode is slower than the embedded mode because all data is transferred over TCP/IP.

Procedure

1. Edit the *%CURAM_DIR%\EJBServer\project\properties\Bootstrap.properties* file.
%CURAM_DIR% is the Cúram installation directory, which by default is *C:\Merative\Curam\Development*.
2. Specify the mode in the *curam.db.h2.mode* property.
For example:

```
# Mode remote|embedded  
curam.db.h2.mode=embedded
```

Bypassing non keywords during a build

H2 reserved words which have been implemented as a database table or column name can be specified in a property which will ensure that the H2 database build will ignore them.

About this task

New versions of the H2 driver disallow H2 reserved words from being used as database table names, column names and in handcrafted SQL syntax. The driver now more strictly adheres to the SQL standards in this regard. The list of H2 reserved words can be seen here: www.h2database.com/html/advanced.html#keywords

H2 have provided a compatibility setting called “SET NON_KEYWORDS” that can be used as a workaround for applications that use keywords as unquoted identifiers. This can be set in the connection string for the database.

The OOTB default setting for this property is
 “CURRENT_DATE,CURRENT_TIMESTAMP,KEY,MONTH,OFFSET,VALUE,YEAR”.

SQL keywords present in custom database table and column names can be specified to be ignored in the H2 database build by using the `curam.db.h2.non.keywords` property in the `Bootstrap.properties` file.

Procedure

1. Edit the `%CURAM_DIR%\EJBServer\project\properties\Bootstrap.properties` file. `%CURAM_DIR%` is the Cúram installation directory, which by default is `C:\Merative\Curam\Development`.
2. Specify the SQL keywords to be ignored in the `curam.db.h2.non.keywords` property. For example:

```
# NON_KEYWORDS, comma delimited list of words which H2 will ignore
# as key words during a database build.
curam.db.h2.non.keywords=<H2_RESERVED_KEYWORD1>,<H2_RESERVED_KEYWORD2>
```

Starting the H2 Web Console

Start the H2 Web Console by running the `org.h2.tools.Server` class in `h2.jar` as follows:

```
java -cp %CuramSDEJ%\drivers\h2-2.2.224.jar org.h2.tools.Server -tcp -web -IfNotExists
```

You can access the H2 Web Console at the following URL:

```
http://localhost:8082/
```

The JDBC connection URL that you specify in the login screen is based on the `curam.db.name`, `curam.db.username`, and `curam.db.h2.directory` values in `Bootstrap.properties`. These values define the database name, SCHEMA name, and the database location in the file system. The default keywords for the NON_KEYWORDS property should also be specified in the connection URL. So, if your database name is `curamdb`, your user name is `curam` and `curam.db.h2.directory` defaults to your home directory, then your JDBC string would look like this example:

```
jdbc:h2:tcp://localhost/~/  
curamdb;schema=curam;FILE_LOCK=SOCKET;AUTO_SERVER=TRUE;NON_KEYWORDS=  
CURRENT_DATE,CURRENT_TIMESTAMP,KEY,MONTH,OFFSET,VALUE,YEAR;
```

For example, if the `curam.db.h2.directory` is `C:/H2`, then your JDBC string would look like this example:

```
jdbc:h2:tcp://localhost/file:C:/H2/  
curamdb;schema=curam;FILE_LOCK=SOCKET;AUTO_SERVER=TRUE;NON_KEYWORDS=  
CURRENT_DATE,CURRENT_TIMESTAMP,KEY,MONTH,OFFSET,VALUE,YEAR;
```

Specify the values for **User Name** and **Password** as in your *Bootstrap.properties* file and then click the **Connect button** (or **Test Connect button**). When connected, a SQL text control is available.

Configuring the IBM® Db2® database

After you install Cúram, complete the following postinstallation configuration steps on IBM® Db2®

Providing a Db2® License File

An empty *db2jcc_license_cu.jar* file exists to allow for Eclipse class path dependencies in the *CuramSDEJ* project. The file is in the Windows `%CURAMSDEJ%\drivers` or Linux `%CURAMSDEJ%/drivers` directory, where `%CURAMSDEJ%` is the root *CuramSDEJ* location directory.

You must overwrite this empty JAR file with a real license to access IBM® Db2®.

You can find the IBM® Db2® license file in the following directory:

- Windows `<Db2_directory>\java\db2jcc_license_cu.jar`
- Linux `<Db2_directory>/java/db2jcc_license_cu.jar`

where *Db2_directory* is the Db2® installation path, for example:

- Windows `C:\IBM\SQLLIB.`
- Linux `/opt/ibm/db2`

Creating and configuring a Db2® database with scripts on Microsoft™ Windows™

Windows

Use the provided Ant scripts to create and configure a basic test database. The Ant scripts use the database properties from your *Bootstrap.properties* file.

To create a database, issue the following commands:

```
ant -f %CURAMSDEJ%\util\db2_createdb.xml
ant -f %CURAMSDEJ%\util\db2_postconfig.xml -Ddb2.dir=db2_directory
ant -f %CURAMSDEJ%\util\db2_createdb.xml restart.db2
```

Note: The *db2_createdb.xml restart.db2* script restarts your Db2® system.

```
ant -f %CURAMSDEJ%\util\db2_optimizedbrecreation.xml
```

where *db2_directory* is the Db2® installation path. By default, `c:\IBM\SQLLIB.`

If you have any problems with creating the database, you can run the following script to drop the database and try again:

```
ant -f %CURAMSDEJ%\util\db2_createdb.xml dropdb
```

Linux®

Linux

To create a database, issue the following commands:

```
ant -f $CURAMSDEJ/util/db2_createdb.xml
ant -f $CURAMSDEJ/util/db2_postconfig.xml -Ddb2.dir=db2_directory
ant -f $CURAMSDEJ/util/db2_createdb.xml restart.db2
```

Note: The `db2_createdb.xml restart.db2` script restarts your Db2® system.

```
ant -f $CURAMSDEJ/util/db2_optimizedbrecreation.xml
```

where `db2_directory` is the Db2® installation path. By default, `/opt/ibm/db2`.

If you have any problems with creating the database, you can run the following script to drop the database and try again:

```
ant -f $CURAMSDEJ/util/db2_createdb.xml dropdb
```

Replacing the packaged Db2® drivers

Usually the most recent JDBC drivers available at the time of release are packaged with Cúram. However, if you want to replace the drivers that are included in Windows `%CURAMSDEJ%\drivers` or Linux `$CURAMSDEJ/drivers`, copy the following files from Windows `<Db2_directory>\java` or Linux `$CURAMSDEJ/drivers` and replace the existing files.

Where `<Db2_directory>` is the Db2® installation path. For example, Windows `C:\IBM\SQLLIB` or Linux `/IBM/SQLLIB`.

- `db2jcc.jar`
- `db2jcc_license_cu.jar`
- `sqlj.zip`

Using Db2® pureScale®

To use Db2® pureScale® with Cúram, you must complete the following steps to set the necessary data source property or properties for using Db2® from the command line. For example, with batch processing. See the IBM® Db2® and IBM® WebSphere® Application Server documentation for their specific pureScale settings.

You must generate a `.bindings` file based on your `Bootstrap.properties` file database settings, which specify the Db2® pureScale connect member. To do this:

1. In your `Bootstrap.properties` file set property `curam.db.enable.bindings.generation=true` and specify a valid location value for property `curam.environment.bindings.location`. For example, Windows `curam.environment.bindings.location=C:/Temp` or Windows `curam.environment.bindings.location=/Curam`.
2. Run the Ant **configtest** target, to generate the `.bindings` file in the specified location.

3. In your *Bootstrap.properties* file, remove `curam.db.enable.bindings.generation=true` or set it to `false` and set `curam.db.disable.bindings.generation=true`.
4. Set the Content value for the relevant pureScale® data source properties in the *.bindings* file. This is easier if you sort it first. For example, set `enableSysplexWLB` to 'true'. Save the changes.

From this point onwards, the Cúram Db2® data source uses these properties when used from the command line. Changes to the database properties in *Bootstrap.properties* must be reflected in *.bindings* or by rerunning the procedure above. However, Ant scripts using the `<sql>` task do not use these pureScale® settings. These Ant scripts, for example the **database** target, are typically not run frequently and don't have a processing profile that requires pureScale settings. However, you can modify scripts as needed to specify these properties by using the Ant `<connectionProperty>` nested element.

Configuring the Oracle Database

After you install Cúram, complete the following postinstallation configuration steps on Oracle Database

Replacing the packaged Oracle JDBC drivers

Typically, the most recent JDBC drivers that are available at the time of release are packaged with Cúram. However, you can replace the included JDBC drivers if needed.

Complete the following steps to replace the included JDBC drivers in the `<CURAMSDEJ>/drivers` directory.

1. Copy and rename the `ojdbc<version>.jar` driver from an installation of your Oracle Database version to `ojdbc<version>.jar` in the `<CURAMSDEJ>/drivers` directory.
2. Replace the `runtime<version>.jar` and `translator.jar` drivers in the *drivers* directory with the drivers from your Oracle Database client installation.

Testing the configuration

Cúram includes a configuration test tool, which helps to confirm that the installation and third-party tools are set up correctly. You can run this tool to detect problems with your installation.

Before you begin

If you are using the H2 database, ensure that you complete these steps before you start this task.

- Build the server and the database.
- If you are using H2 in remote mode, ensure that the H2 Web Console is started.

Procedure

1. Open a command prompt.

2. Go to the `%Curam%/EJBServer` directory and issue the following command to check the application server configuration and the database connectivity:

```
build configtest
```

3. Ensure that the build is successful before proceeding.

Running build commands for the server and client applications

Before you can log on to an Cúram application or to the Universal Access home page, you must run a number of build commands.

About this task

Important:

Ensure that you are in the correct directory before you run each of these commands.

Procedure

1. Open a command line.
2. Go to the `%Curam%/EJBServer` directory and issue the following commands:

`%CURAM_DIR%` is the Cúram installation directory, which by default is Windows `C:\Merative\Curam\Development` or Linux `/opt/merative/curam/development`.

```
build clean server
build database
build prepare.application.data
build createClasspaths
```

3. Go to the `%Curam%/webclient` directory and issue the following command:

```
build clean client
build external-client -Dapp=CitizenPortal
```

4. Ensure that the builds are successful before proceeding.

Starting the XML server

Before you start the Cúram application, you must start the XML server in your ADE.

Procedure

1. Change to the `%CURAM_DIR%\CuramSDEJ\xmlserver` directory.

`%CURAM_DIR%` is the Cúram installation directory, which by default is Windows `C:\Merative\Curam\Development` or Linux `opt/merative/curam/development`.

2. Run the following build target to start the XML server:

```
ant -f xmlserver.xml
```

Customizing cryptography for production environment protection

Cúram provides a default cryptography configuration for development and testing environments only. To strengthen your production environment security, we strongly recommend that you customize the cryptography configuration by creating a new keystore and secret key, and regularly rotating the secret key.

In a production environment, relying on the default cryptography configuration can expose your system to significant risks and vulnerabilities. A system administrator must customize these settings to safeguard against potential data breaches, unauthorized access, and other security threats. For more information about how to customize the cryptography configuration, see the *Security Guide*.

Configuring `curam.referer.domains` for Cross-Site Request Forgery (CSRF) protection

The Cúram user interface (UI) infrastructure uses the HTTP referrer header check to further protect Cúram against Cross-Site Request Forgery (CSRF).

Only requests from trusted domains are permitted. You configure trusted domains on the server as a comma-separated list by using the dynamic system property `curam.referer.domains`. As the default property is set to `localhost`, only an administrator who is logged on to the server host computer can access the application. All other users are blocked.

Configuring `curam.referer.domains` after installation

The following list outlines the two ways that you can configure `curam.referer.domains` after installation:

1. An administrator on the server host computer can access the Cúram system administration application. An administrator can set the dynamic system property `curam.referer.domains` directly to the required host domains to which the administrator wants to grant access.

For more information about Cross-Site Request Forgery (CSRF) protection for Cúram web pages, see the *Security Guide*.

2. Developers and administrators can provide a custom component that overrides the default *Application.prx* and that sets the property `curam.referer.domains` to the required domains.

For more information about how to merge an *application.prx* file, see the *Server Developer's Guide*.

1.7 Installing and configuring Eclipse and Apache Tomcat

You need the following software for an Eclipse and Apache Tomcat IDE. You can download the software from the web, as described in the *Cúram Supported Prerequisites* technote, and follow the product installation instructions and these post-installation steps.

- **Eclipse IDE**
An IDE that you can use to develop an application. If you are unsure about which Eclipse package to download, you can download and install the Eclipse IDE for Java Developers or an equivalent Java development distribution.
- **Apache Tomcat**
A servlet container that you can use to run the client web application.
- **Eclipse Tomcat Plugin**
An open source Eclipse plug-in that integrates with a Tomcat installation to start Tomcat from within Eclipse.

Install the plug-in by extracting the plug-in archive file to the Eclipse *dropins* folder. The default *eclipse/dropins* folder is assumed.

- **Java runtime and Java EE APIs**
You can use the Java runtime that was installed as a prerequisite for the Cúram software for the Eclipse IDE.

Related information

[Cúram prerequisites and supported software](#)

Configuring Tomcat

After installation, you must update the default Tomcat configuration files with the appropriate settings.

About this task

- **UTF-8**
By default, Tomcat assumes that requests are encoded with ISO-8859-1 instead of UTF-8. This default setting can break string handling if request parameters contained UTF-8 extended characters. For correct string handling, you must add the `useBodyEncodingForURI="true"` parameter to the `<Connector>` element of the `server.xml` configuration file.
- **POST Data limit**
By default Tomcat limits POST data to 2 MB. This limit can cause an issue when you use rule sets, which can post data greater than this limit. To disable the POST limit in Tomcat, you can add the `maxPostSize="-1"` attribute to the `<Connector>` element of the `server.xml` configuration file.
- **Non-ASCII characters in Java source files**
Tomcat converts JSPs into servlets that are contained in UTF-8 encoded Java sources files by default (for multi-byte character support). These files are generated into the `work` folder

of the project. The Sysdeo plug-in marks the *work* folder as an Eclipse source folder. If you use the Eclipse build command, the Java compiler expects system encoding sources files by default. If any source file in the *work* folder contains non-ASCII characters, such as *ú*, an `Invalid Character` compiler error is generated and you cannot access the page in a web browser.

The `keepgenerated` attribute prevents Tomcat from saving the source files in the *work* folder and avoids this problem. You can prevent this occurring by updating the Tomcat *web.xml* configuration file with a new `init-param` element.

The Eclipse compiler cannot be changed to compile UTF-8 source files because of a second source folder that is called *JavaSource* that contains files that are not in UTF-8 encoding. Changing this setting does not affect the use of the application in any way. The `keepgenerated` parameter can be set to true if you want to view and debug through source files that are generated by Tomcat, but the error and browser access problem then occurs.

- **Relaxed Query Characters**

In Apache Tomcat, **relaxedQueryChars** is a connector attribute (set in the *server.xml* file) that allows you to specify additional characters that are permitted in the **query string** of HTTP requests. If an issue is encountered with a Cúram query string which contains characters that are not allowed by Tomcat's default configuration, then this configuration parameter can be set to allow such characters (e.g. **relaxedQueryChars**="[,]"). Be cautious when relaxing allowed characters using this parameter, as it can introduce security risks or unexpected behaviour if user input is not properly validated.

- **Relaxed Path Characters**

In Apache Tomcat, **relaxedPathChars** is a connector attribute (set in the *server.xml* file) that allows you to specify additional characters that are permitted in the **path** portion of an HTTP request URL. If an issue is encountered with the **path** portion of a Cúram HTTP request URL which contains characters that are not allowed by Tomcat's default configuration, then this configuration parameter can be set to allow such characters (e.g. **relaxedPathChars**="[,]"). Be cautious when relaxing allowed characters using this parameter as it can introduce security risks or unexpected behaviour if user input is not properly validated.

Procedure

1. Edit the *tomcat_install_dir\conf\server.xml* configuration file and update the `<Connector>` element as follows.

Where *tomcat_install_dir\conf\server.xml* is the directory where you installed Tomcat.

- a) Change the default port number to `port="9080"`.
- b) Add the `useBodyEncodingForURI="true"` attribute.
- c) If you intend to use rule sets, add the `maxPostSize="-1"` attribute.
- d) If you encounter issues with either the query string or the path portion of Cúram HTTP request URLs, the **relaxedQueryChars** and **relaxedPathChars** attributes may be used to specify additional allowed characters.

For example:

```
<Connector port="9080" maxThreads="150" minSpareThreads="25"
maxSpareThreads="75" enableLookups="false" redirectPort="8443"
acceptCount="100" connectionTimeout="20000" disableUploadTimeout="true"
useBodyEncodingForURI="true" maxPostSize="-1"
relaxedQueryChars="[,]" relaxedPathChars="[,]" />
```

2. Edit the `tomcat_install_dir\conf\context.xml` configuration file. Update the `<Context>` element to include a `reloadable="true"` attribute.

For example:

```
<Context reloadable="true">
```

3. Edit the `tomcat_install_dir\conf\web.xml` configuration file. Update the `org.apache.jasper.servlet.JspServlet` servlet with a new `init-param` element with the value `false`.

For example:

```
<init-param>
<param-name>keepgenerated</param-name>
<param-value>>false</param-value>
</init-param>
```

Configuring the Java™ runtime for Eclipse

You must ensure that Eclipse always starts with the correct Java™ runtime and set the default VM arguments. Multiple Java™ runtime installations can be present on your computer from other products that are based on Java™.

Before you begin

Ensure that Eclipse starts with the correct Java™ runtime by using one of the following methods:

- Put the correct Java™ runtime first on the Microsoft™ Windows™ system path.
- Use the `-vm` command-line argument with the `eclipse.exe` command. For more information about Eclipse commands, see the Eclipse documentation.

Procedure

1. Start Eclipse by double-clicking the `eclipse.exe` executable file.
2. After you start Eclipse, select **Window > Preferences > Java > Installed JREs**. On the **Installed JREs** page, ensure that the checkbox for the correct Java™ runtime is selected as the default.
3. Add default VM arguments by selecting the Java™ runtime and clicking **Edit**.
4. In the **Default VM Arguments** field, enter `-Xmx4096M -Xms512m`.
5. Click **OK**.

Configuring the Eclipse class path variables and the Eclipse Tomcat Plugin

In Eclipse, you must set the class path variables and configure the Eclipse Tomcat Plugin.

Procedure

Set the class path variables:

1. In Eclipse, go to **Window > Preferences > Java > Build Path > class path Variables**.
2. Click **New**, enter the following information, and click **OK**.
 - **Name** J2EE_JAR
 - **Path**
 - The path to the JAR file of your Java™ EE API implementation. The Java™ EE API JAR is independent of the Java runtime and can be used with any supported Java™ runtime, including OpenJDK. For example, for WebSphere® Application Server, enter: `C:\IBM\WebSphere\AppServer\lib\j2ee.jar`
 - For example, for Oracle WebLogic Server, enter: `C:\Oracle\Weblogic\wlserver\server\lib\weblogic.jar`
 - If neither WebSphere® Application Server or Oracle WebLogic Server is installed. Download Jakarta `javaee-api-8.0.1.jar` from your trusted artifact repository or Maven Central to a location in your development environment and enter this fully qualified file path for J2EE_JAR setting
3. Click **New**, enter the following information, and click **OK**.
 - **Name** JAVAMAIL_HOME
 - **Path** The folder that contains `mail.jar` and `activation.jar` files for your Java™ EE API implementation.

Note:

If your version of Java™ EE API does not contain these files, you can download JavaMail API and Activation Framework (JAF) from the Oracle website and copy the files to any folder, for example `C:\Tools\JAVA_MAIL`. Then, configure JAVAMAIL_HOME to point to that folder.

4. Click **OK** to save the preferences.

Configure the Eclipse Tomcat Plugin:

The plugin adds a toolbar and more menu options to Eclipse to configure and use Tomcat.

5. In Eclipse, go to **Window > Preferences > Tomcat**.

If you don't see Tomcat in the high-level preferences tree, you might have the wrong Tomcat plugin, it might not be installed properly, or you might need a clean Eclipse start.

6. Select the appropriate Tomcat version, such as version 7.x.
7. Set Tomcat home to where you extracted the downloaded archive.
For example, `C:\Tools\Tomcat\apache-tomcat-<version>`.

8. Select JVM Settings and in the **Append to JVM Parameters** field, enter `-Xmx1024m`.

Importing and configuring the Cúram projects in Eclipse

Import the Cúram projects into Eclipse. The projects are automatically built when you import them. After the projects are imported, you can configure them for use.

Procedure

Import the projects:

1. In Eclipse, click **File > Import**.
2. Expand **General**, select **Existing Projects into Workspace**, and click **Next**.
3. Set the **Select root directory field** field to, for example `C:\Merative\Curam\Development`.

Eclipse automatically searches for and identifies projects.

4. Ensure that **Copy projects into workspace** is NOT selected.
5. Click **Finish**.

Eclipse immediately starts building the workspace. The following error is displayed:

```
Project 'CuramBITransforms' is missing required Java project:
'CuramBITools' CuramBITransforms Build path Build Path Problem
```

6. Manually import the CuramBITransforms project. In Eclipse, click **File > Import**.
7. Expand **General**, select **Existing Projects into Workspace**, and click **Next**.
8. Set the **Select root directory field** field to `C:\Merative\Curam\Development\Reporting\components\BIBuildTools`.

Eclipse automatically searches for and identifies projects.

9. Ensure that **Copy projects into workspace** is NOT selected.
10. Click **Finish**.

Configure the Curam project:

11. In Eclipse Package Explorer, right-click the **Curam** project, select **Properties > Tomcat** and confirm the following settings.

- **Is a Tomcat Project:** Selected
- **Context Name:** `/Curam`
- **Can update server.xml file:** Selected
- **Mark this context as reloadable:** Selected
- **Redirect context logger to Eclipse console:** Selected
- **Subdirectory to set as application root:** `/WebContent`
- **Subdirectory to set as application work:** `/work`

12. Click **OK**.
13. Update the Tomcat server.xml file with an entry for the Curam application by right-clicking the **CitizenPortal** project and selecting **Properties > Tomcat Project > Update context definition**.

Configure the CitizenPortal project:

14. In Eclipse Package Explorer, right-click the **CitizenPortal** project and select **Properties** > **Tomcat** and confirm the following settings.
 - **Is a Tomcat Project:** Selected
 - **Context Name:** /CitizenPortal
 - **Can update server.xml file:** Selected
 - **Mark this context as reloadable:** Selected
 - **Redirect context logger to Eclipse console:** Selected
 - **Subdirectory to set as application root:** /WebContent
 - **Subdirectory to set as application work:** /work
15. Click **OK**.
16. Update the Tomcat server.xml file with an entry for the CitizenPortal application by right-clicking the **CitizenPortal** project and selecting **Properties** > **Tomcat Project** > **Update context definition**.

Configure the CuramBIRTViewer Project:

17. In Eclipse Package Explorer, right-click the **CuramBIRTViewer** project and select **Properties** > **Tomcat** and confirm the following settings.
 - **Is a Tomcat Project:** Selected
 - **Context Name:** /CuramBIRTViewer
 - **Can update server.xml file:** Selected
 - **Mark this context as reloadable:** Selected
 - **Redirect context logger to Eclipse console:** Selected
 - **Subdirectory to set as application root:** /WebContent
 - **Subdirectory to set as application work:** /work
18. Update the Tomcat server.xml file with an entry for the CuramBIRTViewer application by right-clicking the **CuramBIRTViewer** project and selecting **Properties** > **Tomcat Project** > **Update context definition**.
19. Click **OK**.
20. In your Curam Development command window, enter:

```
cd %CURAM_DIR%\BICContent
```

Where %CURAM_DIR% is the Cúram installation directory, which by default is *C:\Merative\Curam\Development*.

21. Enter:

```
build client.birt
```

If you are developing new Business Intelligence and Analytics content, see the BIRT developer's information for more details on how to set up a development environment.

Starting the Tomcat server and the Cúram servers

Start the Tomcat and Cúram servers by following the instructions in the related links.

About this task

After the servers are started, applications are available at these URL:

- The Cúram client application is available at the following URL: `http://localhost:8080/Curam/AppController.do`
- The CitizenPortal application is available at the following URL: `http://localhost:8080/CitizenPortal/application.do`
- The Cúram Business Intelligence and Analytics Viewer application is available at the following URL: `http://localhost:8080/CuramBIRTViewer`

Related tasks

[Starting the server, Tomcat, and the Login Client on page 50](#)

Start the server, Tomcat, and the Login Client so that you can log in and test the installation.

Using Eclipse to validate the tabbed configuration artifacts

You can set up Eclipse to validate the tabbed configuration files with the correct schema.

Open the Eclipse **Preferences** dialog by selecting **Window > Preferences** and complete the following steps:

- Select **XML > XML Catalog**.
 - Click **Add** to add an entry.
 - For the **Location**, point at the schema file (for example, `tab.xsd`) in the `%CURAMSDEJ%\lib` directory.
 - Leave the rest as defaults and click **OK**.
 - Repeat for each of the schema files for the tabbed configuration artifacts. For the full list of schema files associated with configuration files, see the *Web Client Reference Manual*.
 - Click **OK** to exit the **XML Catalog** window.
- Select **General > Editors > File Associations**.
 - Click **Add...** to add an entry: `*.tab`.
 - Select the new `*.tab` entry and click **Add** to add the XML Editor as the **Associated Editor**.
 - Repeat for all the tabbed configuration artifact file extensions. For the full list of extensions for configuration files, see the *Web Client Reference Manual*.
- Select **General > Content Types**.
 - Expand **Text** and select **XML**.
 - Click **Add** to enter a file association for XML content and click **OK**. Do this step for each of the file extensions.
- Click **OK** to save the preference changes.

Supported Eclipse text file encoding

In Eclipse, you can set the default text file encoding at the project level. Changing the text file encoding from the default is unsupported for Cúram projects within Eclipse.

This restriction does not affect your ability to save files in various encodings on a file-by-file basis.

1.8 Installing and configuring Rational® Software Architect Designer

IBM® Rational® Software Architect Designer is an Eclipse-based UML modeling tool that is required to do server development. The exact installation steps for installing IBM® Rational® Software Architect Designer can vary depending on the edition and version of your software.

Before you begin

IBM® Installation Manager is used to install Rational® Software Architect Designer. For more information about installing IBM® Installation Manager, see [1.2 Installing IBM® Installation Manager on page 12](#).

Procedure

1. Start the IBM® Installation Manager by clicking **Start > All Programs > IBM® Installation Manager > IBM® Installation Manager**.
2. From the **File** menu, select **Preferences**.
3. From the **Repositories** window, select **Add Repository**.
4. From the **Add Repository** window, select **Browse** to add an entry that points to your Rational® Software Architect Designer installation file and click **OK**.
5. From the **Repositories** window, ensure that this entry is the only selected repository and click **OK**.
6. From the **IBM Installation Manager** window, select **Install**.
7. From the **Install Packages** window, select appropriate version check box and click **Next**.
8. From the **Prerequisite** window, read the information that is displayed and click **Next**.
9. From the **Licenses** window, read the information, select **I accept the terms in the license agreements**, and click **Next**.
10. From the **Location** window, select **Browse** to select the installation directory or use the default value. For **Architecture Selection**, update the default values if required. Click **Next**.
11. The **Languages To Install** screen window is displayed. This can be left at the default ("English") if no further languages are required Click **Next**.
12. The select **Features to install** window is displayed, The Architecture selection is important as some of the required plug-ins do not function correctly with the 64-bit version. Make the following selections.
 - Installation Profiles – Select **Architect- Standard**.
 - Click the "**Rational Rose model import**" option.

- Leave the rest of the defaults.
 - Click **Next**.
13. The Review Summary **Summary** Information window is displayed. Review the information and when satisfied, click **Install**.
 14. The installation will progress. When completed, the Installation Summary window is displayed. Click **Finish** to complete and exit the installation process.

Installing the Cúram plug-ins

You must install the Cúram plug-ins to enable modeling support.

Procedure

1. From the Rational® Software Architect Designer installation directory, for example, the `C:\Program Files\IBM\SDP` directory, create a *dropins* directory.
 2. From the *dropins* directory, create a new file called *rsa_plugin.link* that contains the path to the plug-ins for Rational® Software Architect Designer. Ensure that you use forward slashes.
- For example,

```
path=C:/Curam/CuramSDEJ/rsa
```

3. Edit the Microsoft Windows shortcut that starts Rational® Software Architect Designer to pass the `-clean` option so that the plug-ins are picked up.
- For example:

```
...\eclipse.exe -clean -product com.ibm.rational....
```

4. Start or restart Rational® Software Architect Designer.

Setting up your Rational® Software Architect Designer workspace

Complete the following steps to set up your Eclipse workspace in Rational® Software Architect Designer.

Procedure

1. Start Rational® Software Architect Designer by clicking **Start > All Programs > IBM Software Delivery Platform > IBM® Rational® Software Architect Designer version > IBM® Rational® Software Architect Designer**.
2. From the **Workspace Launcher**, select **Browse** to navigate to the location where you want your Rational® Software Architect Designer workspace to be stored and click **OK**.
3. From the **Overview** window, hover over the top right icon to display Workbench. Select the **Workbench** icon.
4. Select **File > Import**.
5. From the **Import** window, expand the **General** folder and select **Existing Project into Workspace** and click **Next**.

6. From the **Import Projects** window, select **Browse** and select `%CURAM_DIR%\EJBServer` for the **Select root directory** and click **Finish**.
`%CURAM_DIR%` is the Cúram installation directory, which by default is `C:\Merative\Curam\Development`.
7. Repeat steps 4-6 to import the `%CURAMSDEJ%` root directory. You are now ready to start using Rational® Software Architect Designer.

1.9 Getting started with the development environment

The installation is now complete. Start the server, Tomcat, and the Login Client client so you can log in to the application.

Starting the server, Tomcat, and the Login Client

Start the server, Tomcat, and the Login Client so that you can log in and test the installation.

About this task

The server is started as a Java runtime process that starts the following threads:

In Java 8 Development Environment:

These environments rely on a configured Java runtime appropriate to the selected development setup.

- **tnameserv**
The Java Transient Name Server to facilitate a JNDI lookup service for finding resources such as Java classes. The Java tnameserv.exe is not stopped when you exit Eclipse. You can check for tnameserv.exe in Windows Task Manager.
- **RMI Server**
The RMILoginClient server application process, which provides login functionality.
- **JMSLite**
JMS Message Engine, which provides JMS-like functionality.

For more information about JMSLite, see the *Workflow Reference Guide*.

In Modern Java Development Environment:

- **Rest Server**
The RestLoginClient server application process, which provides login functionality.
- **JMSLite**
JMS Message Engine, which provides JMS-like functionality.

For more information about JMSLite, see the *Workflow Reference Guide*.

Procedure

1. Check that the database is running.
2. In Package Explorer, expand **EJBServer > components > core > lib**, right-click on **core.jar**, and select **Run As > Java Application**.

3. In the **Select Java Application** window, select **StartServer** and click **OK**.

When a console window is displayed in Eclipse, check for messages like these:

```
tnameserv.exe started:
    Initial Naming Context:

IOR:00bdbdbd0000002b49444c3a6f6d672e6f72672f436f734e616d696e672f4e616d696e67436f6e
746578744578743a312e3000bd000000010000000000000072000102bd0000000d392e3136312e3939
2e31333600bd04c5000000164c4d424900000015b3c2f20c00100000000400000000bdbd0000000300
0000010000001800bdbdbd000100010000000100010020000101000000000049424d0a0000000800bd
0011180000000000000260000000020002
    TransientNameServer: setting port for initial object references to: 1221
    Ready.

JMSLite started:
### ...
### Custom RIDPSecondaryRequestMockService loaded ###
```

4. To start Tomcat, on the Eclipse menu, click the Start Tomcat button on the Eclipse Tomcat Plugin toolbar.
Confirm a clean start in the Tomcat console window in Eclipse by waiting for a few moments until you see a message like this:

```
INFO: Server startup in 61316 ms
```

5. In Package Explorer, expand the **EJBServer > components > core > lib**, right-click on **core.jar**, depending on Development Environment Java Version select either:
 - a) **RMILoginClient** for Java 8 and select **Run as application** or
 - b) **RestLoginClient** for Modern Java and select **Run as application**
6. In the **Select Java Application** window, select **Login Client** and click **OK**.
7. In the **Login Client** window, enter your credentials.

Logging on to Cúram

You can access the Cúram application or the Citizen Portal application from any supported browser. From here, you can access features that are based on your role. For example, you can log on to administer the system.

Procedure

1. Enter one of the following URLs:

```
https://server_name:port/Curam/AppController.do
```

```
https://server_name:port/CitizenPortal/application.do
```

where:

- *server_name* is the name of the server where you installed the application.
- *port* is the port for the application. Default port numbers are as follows:
 - Apache Tomcat: 9080
 - IBM® WebSphere® Application Server: 9044

- Oracle WebLogic Server: 7002

2. Log on with the appropriate role.

Option	Description
sysadmin	The System Administrator user has access to technical administration features.
admin	The Administrator user has access to administration features.
caseworker	The caseworker role has access to caseworker features.
supervisor	The supervisor role has access to supervisor features.

Deploying Cúram

To test your applications with an enterprise application server, you can deploy the application and web services application to one of the supported application server and database combinations.

- IBM® WebSphere® Application Server and IBM® Db2®.
- IBM® WebSphere® Application Server and Oracle Database.
- Oracle WebLogic Server and Oracle Database.

For more information about deployment, see:

-
-
-

1.10 Upgrading Cúram

Use the Cúram Upgrade Helper Pack and the UI Upgrade Helper tool to help you to upgrade your custom Cúram application.

- **Cúram Upgrade Helper Pack**

The Cúram Upgrade Helper Pack contains tools and an upgrade guide, which describes the recommended process for upgrading Cúram applications and details any significant migrations that might require upgrade effort. The tools are available to assist with certain process steps. The Cúram Upgrade Helper Pack is available from the [Merative Software Downloads](#) site. You must be a technical contact to access this site, see [Cúram support](#) for details. Download the appropriate version for your upgrade.

- **UI Upgrade Helper**

You can use the Cúram UI Upgrade Helper tool to help you to upgrade your Cúram UI from version 7 to version 8. For more information about the UI Upgrade Helper, see <https://merative.github.io/spm-ui-upgrade-helper>.

Notices

Permissions for the use of these publications are granted subject to the following terms and conditions.

Applicability

These terms and conditions are in addition to any terms of use for the Merative website.

Personal use

You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of Merative

Commercial use

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of Merative.

Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

Merative reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by Merative, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

MERATIVE MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

Merative or its licensors may have patents or pending patent applications covering subject matter described in this document. The furnishing of this documentation does not grant you any license to these patents.

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

Any references in this information to non-Merative websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those

websites are not part of the materials for this Merative product and use of those websites is at your own risk.

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

The licensed program described in this document and all licensed material available for it are provided by Merative under terms of the Merative Client Agreement.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to Merative, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. Merative, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. Merative shall not be liable for any damages arising out of your use of the sample programs.

Privacy policy

The Merative privacy policy is available at <https://www.merative.com/privacy>.

Trademarks

Merative™ and the Merative™ logo are trademarks of Merative US L.P. in the United States and other countries.

IBM®, the IBM® logo, and ibm.com® are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide.

Adobe™, the Adobe™ logo, PostScript™, and the PostScript™ logo are either registered trademarks or trademarks of Adobe™ Systems Incorporated in the United States, and/or other countries.

Oracle and Java are registered trademarks of Oracle and/or its affiliates.

The registered trademark Linux® is used pursuant to a sublicense from the Linux Foundation, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis.

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

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

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