Installing and configuring the IBM MobileFirst Foundation Application Center

Overview

You install the Application Center as part of the MobileFirst Server installation. You can install it with one of the following methods:

- Installation with IBM Installation Manager
- Installation with Ant tasks
- Manual installation

Optionally, you can create the database of your choice before you install MobileFirst Server with the Application Center.
After you installed the Application Center in the web application server of your choice, you have additional configuration to do. For more information, see Configuring Application Center after installation below. If you chose a manual setup in the installer, see the documentation of the server of your choice.

> **Note:** If you intend to install applications on iOS devices through the Application Center, you must first configure the Application Center server with SSL.

For a list of installed files and tools, see Distribution structure of MobileFirst Server (../installation-manager/#distribution-structure-of-mobilefirst-server).

Jump to

Installing Application Center with IBM Installation Manager

With IBM Installation Manager, you can install Application Center, create its database, and deploy it on an Application Server.
Before you begin, verify that the user who runs IBM Installation Manager has the privileges that are described in File system prerequisites (../appserver/#file-system-prerequisites).

To install IBM Application Center with IBM Installation Manager, complete the followings steps.

1. Optional: You can manually create databases for Application Center, as described in Optional creation of databases below. IBM Installation Manager can create the Application Center databases for you with default settings.
2. Run IBM Installation Manager, as described in Running IBM Installation Manager (../installation-manager).
3. Select **Yes** to the question **Install IBM Application Center**.

Jump to

Optional creation of databases

If you want to activate the option to install the Application Center when you run the MobileFirst Server installer, you need to have certain database access rights that entitle you to create the tables that are required by the Application Center.

If you have sufficient database administration credentials, and if you enter the administrator user name and password in the installer when prompted, the installer can create the databases for you. Otherwise, you need to ask your database administrator to create the required database for you. The database needs to be created before you start the MobileFirst Server installer.

The following topics describe the procedure for the supported database management systems.

Jump to

Creating the DB2 database for Application Center

During IBM MobileFirst Foundation installation, the installer can create the Application Center database for you.

The installer can create the Application Center database for you if you enter the name and password of a user account on the database server that has the DB2 SYSADM or SYSCTRL privilege, and the account can be accessed through SSH. Otherwise, the database administrator can create the Application Center database for you. For more information, see the DB2 Solution (http://ibm.biz/knowctr#SSEPGG_9.7.0/com.ibm.db2.luw.admin.sec.doc/doc/c0055206.html) user documentation.

When you manually create the database, you can replace the database name (here APPCNTR) and the password with a database name and password of your choosing.

> **Important:** You can name your database and user differently, or set a different password, but ensure that you enter the appropriate database name, user name, and password correctly across the DB2 database setup. DB2 has a database name limit of 8 characters on all platforms, and has a user name and password length limit of 8 characters for UNIX and Linux systems, and 30 characters for Windows.

1. Create a system user, for example, named **wluser** in a DB2 admin group such as **DB2USERS**, using the appropriate commands for your operating system. Give it a password, for example, **wluser**. If you want multiple instances of IBM MobileFirst Server to connect to the same database, use a different user name for each connection. Each database user has a separate default schema. For more information about database users, see the DB2 documentation and the documentation for your operating system.
2. Open a DB2 command line processor, with a user that has **SYSADM** or **SYSCTRL** permissions:

- On Windows systems, click **Start → IBM DB2 → Command Line Processor**
- On Linux or UNIX systems, navigate to ~/**sqllib**/**bin** and enter `./db2`.
- Enter database manager and SQL statements similar to the following example to create the Application Center database, replacing the user name **wluser** with your chosen user names:

```
CREATE DATABASE APPCNTR COLLATE USING SYSTEM PAGESIZE 32768
CONNECT TO APPCNTR
GRANT CONNECT ON DATABASE TO USER wluser
DISCONNECT APPCNTR
QUIT
```

- The installer can create the database tables and objects for Application Center in a specific schema. This allows you to use the same database for Application Center and for a MobileFirst project. If the IMPLICIT_SCHEMA authority is granted to the user created in step 1 (the default in the database creation script in step 2), no further action is required. If the user does not have the IMPLICIT_SCHEMA authority, you need to create a SCHEMA for the Application Center database tables and objects.

Creating the MySQL database for Application Center

During the MobileFirst installation, the installer can create the Application Center database for you.

The installer can create the database for you if you enter the name and password of the superuser account. For more information, see Securing the Initial MySQL Accounts (http://dev.mysql.com/doc/refman/5.1/en/default-privileges.html) on your MySQL database server. Your database administrator can also create the databases for you. When you manually create the database, you can replace the database name (here APPCNTR) and password with a database name and password of your choosing. Note that MySQL database names are case-sensitive on UNIX.

1. Start the MySQL command-line tool.
2. Enter the following commands:

```
CREATE DATABASE APPCNTR CHARACTER SET utf8 COLLATE utf8_general_ci;
GRANT ALL PRIVILEGES ON APPCNTR.* TO 'worklight'@'Worklight-host' IDENTIFIED BY 'password';
GRANT ALL PRIVILEGES ON APPCNTR.* TO 'worklight'@'localhost' IDENTIFIED BY 'password';
FLUSH PRIVILEGES;
```

   Here, you need to replace **Worklight-host** with the name of the host on which IBM MobileFirst Foundation runs.

Creating the Oracle database for Application Center

During the installation, the installer can create the Application Center database, except for the Oracle 12c database type, or the user and schema inside an existing database for you.

The installer can create the database, except for the Oracle 12c database type, or the user and schema inside an existing database if you enter the name and password of the Oracle administrator on the database server, and the account can be accessed through SSH. Otherwise, the database administrator can create the database or user and schema for you. When you manually create the database or user, you can use database names, user names, and a password of your choosing. Note that lowercase characters in Oracle user names can lead to trouble.

1. If you do not already have a database named **ORCL**, use the Oracle Database Configuration Assistant (DBCA) and follow the steps in the wizard to create a new general-purpose database named **ORCL**:

- Use global database name **ORCL_your_domain**, and system identifier (SID) **ORCL**.
- On the **Custom Scripts** tab of the step **Database Content**, do not run the SQL scripts, because you must first create a user account.
- On the **Character Sets** tab of the step **Initialization Parameters**, select **Use Unicode (AL32UTF8) character set and UTF8 - Unicode 3.0 UTF-8 national character set**.
- Complete the procedure, accepting the default values.

2. Create a database user either by using **Oracle Database Control**, or by using the **Oracle SQLPlu**s command-line interpreter.
   - Using **Oracle Database Control**:
     - Connect as **SYSDBA**.
     - Go to the **Users** page: click **Server**, then **Users** in the **Security** section.
     - Create a user, for example, named **APPCENTER**. If you want multiple instances of IBM MobileFirst Server to connect to the same general-purpose database you created in step 1, use a different user name for each connection. Each database user has a separate default schema.
     - Assign the following attributes:
       - Profile: **DEFAULT**
       - Authentication: **password**
       - Default tablespace: **USERS**
       - Temporary tablespace: **TEMP**
       - Status: **Unlocked**
       - Add system privilege: **CREATE SESSION**
       - Add system privilege: **CREATE SEQUENCE**
       - Add system privilege: **CREATE TABLE**
       - Add quota: **Unlimited for tablespace USERS**
   - Using the **Oracle SQLPlus** command-line interpreter:
     The commands in the following example create a user named APPCENTER for the database:

```
CONNECT SYSTEM/<SYSTEM_password>@ORCL
CREATE USER APPCENTER IDENTIFIED BY password DEFAULT TABLESPACE USERS QUOTA UNLIMITED ON USERS;
GRANT CREATE SESSION, CREATE SEQUENCE, CREATE TABLE TO APPCENTER;
DISCONNECT;
```

Installing Application Center in WebSphere Application Server Network Deployment

To install Application Center in a set of WebSphere Application Server Network Deployment servers, run IBM Installation Manager on the machine where the deployment manager is running.

1. When IBM Installation Manager prompts you to specify the database type, select any option other than **Apache Derby**. IBM MobileFirst Foundation supports Apache Derby only in embedded mode, and this choice is incompatible with deployment through WebSphere Application Server Network Deployment.
2. In the installer panel in which you specify the WebSphere Application Server installation directory, select the deployment manager profile.

   > **Attention:** Do not select an application server profile and then a single managed server: doing so causes the deployment manager to overwrite the configuration of the server regardless of whether you install on the machine on which the deployment manager is running or on a different machine.

3. Select the required scope depending on where you want Application Center to be installed. The following table lists the available scopes:

Scope	Explanation
Cell	Installs Application Center in all application servers of the cell.
Cluster	Installs Application Center in all application servers of the specified cluster.
Node	(excluding clusters) Installs Application Center in all application servers of the specified node that are not in a cluster.
Server	Installs Application Center in the specified server, which is not in a cluster.

4. Restart the target servers by following the procedure in Completing the installation below.

The installation has no effect outside the set of servers in the specified scope. The JDBC providers and JDBC data sources are defined with the specified scope. The entities that have a cell-wide scope (the applications and, for DB2 , the authentication alias) have a suffix in their name that makes them unique. So, you can install Application Center in different configurations or even different versions of Application Center, in different clusters of the same cell.

> **Note:** Because the JDBC driver is installed only in the specified set of application servers, the Test connection button for the JDBC data sources in the WebSphere Application Server administration console of the deployment manager might not work.

If you use a front-end HTTP server, you need to configure the public URL as well.

Completing the installation

When installation is complete, you must restart the web application server in certain cases.
You must restart the web application server in the following circumstances:

- When you are using WebSphere Application Server with DB2 as database type.
- When you are using WebSphere Application Server and have opened it without the application security enabled before you installed IBM MobileFirst Application Center or MobileFirst Server.

The MobileFirst installer must activate the application security of WebSphere Application Server (if not active yet) to install Application Center. Then, for this activation to take place, restart the application server after the installation of MobileFirst Server completed.

- When you are using WebSphere Application Server Liberty or Apache Tomcat.
- After you upgraded from a previous version of MobileFirst Server.

If you are using WebSphere Application Server Network Deployment and chose an installation through the deployment manager:

- You must restart the servers that were running during the installation and on which the MobileFirst Server web applications are installed.

To restart these servers with the deployment manager console, select **Applications → Application Types → WebSphere enterprise applications → IBM_Application_Center_Services → Target specific application status**.

- You do not have to restart the deployment manager or the node agents.

> **Note:** Only the Application Center is installed in the application server. A MobileFirst Operations Console is not installed by default. To install a MobileFirst Operations Console.

Default logins and passwords created by IBM Installation Manager for the Application Center

IBM Installation Manager creates the logins by default for the Application Center, according to your application server. You can use these logins to test the Application Center.

WebSphere Application Server full profile

The login **appcenteradmin** is created with a password that is generated and displayed during the installation.

All users authenticated in the application realm are also authorized to access the **appcenteradmin** role. This is not meant for a production environment, especially if WebSphere Application Server is configured with a single security domain.

For more information about how to modify these logins, see Configuring the Java EE security roles on WebSphere Application Server full profile.

WebSphere Application Server Liberty profile

- The login demo is created in the basicRegistry with the password demo.
- The login appcenteradmin is created in the basicRegistry with the password admin.

For more information about how to modify these logins, see Configuring the Java EE security roles on WebSphere Application Server Liberty profile.

Apache Tomcat

- The login demo is created with the password demo.
- The login guest is created with the password guest.
- The login appcenteradmin is created with the password admin.

For more information about how to modify these logins, see Configuring the Java EE security roles on Apache Tomcat.

Installing the Application Center with Ant tasks

Learn about the Ant tasks that you can use to install Application Center.

Jump to

- Creating and configuring the database for Application Center with Ant tasks

- Deploying the Application Center Console and Services with Ant tasks

Creating and configuring the database for Application Center with Ant tasks

If you did not manually create the database, you can use Ant tasks to create and configure your database for Application Center. If your database already exists, you can perform only the configuration steps with Ant tasks.

Before you begin, make sure that a database management system (DBMS) is installed and running on a database server, which can be on the same computer, or a different one.

The Ant tasks for Application Center are in the **ApplicationCenter/configuration-samples** directory of the MobileFirst Server distribution.

If you want to start the Ant task from a computer where MobileFirst Server is not installed, you must copy the following files to that computer:

- The library **mf_server_install_dir/MobileFirstServer/mfp-ant-deployer.jar**
- The directory that contains binary files of the aapt program, from the Android SDK platform-tools package: **mf_server_install_dir/ApplicationCenter/tools/android-sdk**
- The Ant sample files that are in **mf_server_install_dir/ApplicationCenter/configuration-samples**

> **Note:** The **mf_server_install_dir** placeholder represents the directory where you installed MobileFirst Server.

If you did not create your database manually, as described in Optional creation of databases, follow steps 1 to 3 below. If your database already exists, you must create only the database tables. Follow steps 4 to 7 below.

1. Copy the sample Ant file that corresponds to your DBMS. The files for creating a database are named after the following pattern:

   ```
   create-appcenter-database-<dbms>.xml
   ```

2. Edit the Ant file, and replace the placeholder values with the properties at the beginning of the file.
3. Run the following commands to create the Application Center database:

   ```
   ant -f create-appcenter-database-<dbms>.xml databases
   ```

   You can find the Ant command in **mf_server_install_dir/shortcuts**.

   If the database already exists, then you must create only the database tables by completing the following steps:
4. Copy the sample Ant file that corresponds to both your application server, and your DBMS. The files for configuring an existing database are named after this pattern:

   ```
   configure-appcenter-<appServer>-<dbms>.xml
   ```

5. Edit the Ant file, and replace the placeholder values with the properties at the beginning of the file.
6. Run the following commands to configure the database:

   ```
   ant -f configure-appcenter-<appServer>-<dbms>.xml databases
   ```

   You can find the Ant command in **mf_server_install_dir/shortcuts**.
7. Save the Ant file. You might need it later to apply a fix pack, or perform an upgrade.

If you do not want to save the passwords, you can replace them by "**********" (12 stars) for interactive prompting.

Deploying the Application Center Console and Services with Ant tasks

Use Ant tasks to deploy the Application Center Console and Services to an application server, and configure data sources, properties, and database drivers that are used by Application Center.

Before you begin,

- Complete the procedure at Creating and configuring the database for Application Center with Ant tasks.
- You must run the Ant task on the computer where the application server is installed, or the Network Deployment Manager for WebSphere Application Server Network Deployment. If you want to start the Ant task from a computer where MobileFirst Server is not installed, you must copy the following files and directories to that computer:
  - The library **mf_server_install_dir/MobileFirstServer/mfp-ant-deployer.jar**
  - The web applications (WAR and EAR files) in **mf_server_install_dir/ApplicationCenter/console**
  - The directory that contains the binary files of the aapt program, from the Android SDK platform-tools package: **mf_server_install_dir/ApplicationCenter/tools/android-sdk**
  - The Ant sample files that are in **mf_server_install_dir/ApplicationCenter/configuration-samples**

Note: The mf*server*install_dir placeholder represents the directory where you installed MobileFirst Server.

1. Copy the Ant file that corresponds both to your application server, and your DBMS. The files for configuring Application Center are named after the following pattern:

   ```
   configure-appcenter-<appserver>-<dbms>.xml
   ```

2. Edit the Ant file, and replace the placeholder values with the properties at the beginning of the file.
3. Run the following command to deploy the Application Center Console and Services to an application server:

   ```
   ant -f configure-appcenter-<appserver>-<dbms>.xml install
   ```

   You can find the Ant command in **mf_server_install_dir/shortcuts**.

   Note: With these Ant files, you can also do the following actions:
   - Uninstall Application Center, with the target **uninstall**.
   - Update Application Center with the target **minimal-update**, to apply a fix pack.

4. Save the Ant file. You might need it later to apply a fix pack or perform an upgrade. If you do not want to save the passwords, you can replace them by "**********" (12 stars) for interactive prompting.
5. If you installed on WebSphere Application Server Liberty profile, or Apache Tomcat, check that the aapt program is executable for all users. If needed, you must set the proper user rights. For example, on UNIX / Linux systems:

   ```
   chmod a+x mf_server_install_dir/ApplicationCenter/tools/android-sdk/*/aapt*
   ```

Manually installing Application Center

A reconfiguration is necessary for the MobileFirst Server to use a database or schema that is different from the one that was specified during its installation. This reconfiguration depends on the type of database and on the kind of application server.

On application servers other than Apache Tomcat, you can deploy Application Center from two WAR files or one EAR file.

Restriction: Whether you install Application Center with IBM Installation Manager as part of the MobileFirst Server installation or manually, remember that "rolling updates" of Application Center are not supported. That is, you cannot install two versions of Application Center (for example, V5.0.6 and V6.0.0) that operate on the same database.

Jump to

Configuring the DB2 database manually for Application Center

You configure the DB2 database manually by creating the database, creating the database tables, and then configuring the relevant application server to use this database setup.

1. Create the database. This step is described in Creating the DB2 database for Application Center.
2. Create the tables in the database. This step is described in Setting up your DB2 database manually for Application Center.
3. Perform the application server-specific setup as the following list shows.

Jump to

Setting up your DB2 database manually for Application Center

Set up your DB2 database for Application Center by creating the database schema.

1. Create a system user, **worklight**, in a DB2 admin group such as **DB2USERS**, by using the appropriate commands for your

operating system. Give it the password **worklight**. For more information, see the DB2 documentation and the documentation for your operating system.

> **Important:** You can name your user differently, or set a different password, but ensure that you enter the appropriate user name and password correctly across the DB2 database setup. DB2 has a user name and password length limit of 8 characters for UNIX and Linux systems, and 30 characters for Windows.

1. Open a DB2 command line processor, with a user that has **SYSADM** or **SYSCTRL** permissions:
   - On Windows systems, click **Start → IBM DB2 → Command Line Processor**.
   - On Linux or UNIX systems, go to ~/**sqllib**/**bin** and enter `./db2`.
2. Enter the following database manager and SQL statements to create a database that is called **APPCNTR**:

   ```
   CREATE DATABASE APPCNTR COLLATE USING SYSTEM PAGESIZE 32768
   CONNECT TO APPCNTR
   GRANT CONNECT ON DATABASE TO USER worklight
   QUIT
   ```

3. Run DB2 with the following commands to create the **APPCNTR** tables, in a schema named **APPSCHM** (the name of the schema can be changed). This command can be run on an existing database that has a page size compatible with the one defined in step 3.

   ```
   db2 CONNECT TO APPCNTR
   db2 SET CURRENT SCHEMA = 'APPSCHM'
   db2 -vf product_install_dir/ApplicationCenter/databases/create-appcenter-db2.sql -t
   ```

Configuring Liberty profile for DB2 manually for Application Center

You can set up and configure your DB2 database manually for Application Center with WebSphere Application Server Liberty profile. Complete the DB2 Database Setup procedure before continuing.

1. Add the DB2 JDBC driver JAR file to **$LIBERTY_HOME/wlp/usr/shared/resources/db2**.

   If that directory does not exist, create it. You can retrieve the file in one of two ways:
   - Download it from DB2 JDBC Driver Versions (http://www.ibm.com/support/docview.wss?uid=swg21363866).
   - Fetch it from the **db2_install_dir/java** on the DB2 server directory.
2. Configure the data source in the **$LIBERTY_HOME/wlp/usr/servers/worklightServer/server.xml** file as follows:

   In this path, you can replace **worklightServer** by the name of your server.

   ```
   <library id="DB2Lib">
      <fileset dir="${shared.resource.dir}/db2" includes="*.jar"/>
   </library>

   <!-- Declare the IBM Application Center database. -->
   <dataSource jndiName="jdbc/AppCenterDS" transactional="false">
     <jdbcDriver libraryRef="DB2Lib"/>
     <properties.db2.jcc databaseName="APPCNTR"  currentSchema="APPSCHM"
          serverName="db2server" portNumber="50000"
          user="worklight" password="worklight"/>
   </dataSource>
   ```

   The **worklight** placeholder after **user=** is the name of the system user with **CONNECT** access to the **APPCNTR** database that you have previously created.

   The **worklight** placeholder after **password=** is this user's password. If you have defined either a different user name, or a different password, or both, replace **worklight** accordingly. Also, replace **db2server** with the host name of your DB2 server (for example, **localhost**, if it is on the same computer).

   DB2 has a user name and password length limit of 8 characters for UNIX and Linux systems, and 30 characters for Windows.
3. You can encrypt the database password with the securityUtility program in **liberty_install_dir/bin**.

Configuring WebSphere Application Server for DB2 manually for Application Center

You can set up and configure your DB2 database manually for Application Center with WebSphere Application Server.

1. Determine a suitable directory for the JDBC driver JAR file in the WebSphere Application Server installation directory.
   - For a stand-alone server, you can use a directory such as **was_install_dir/optionalLibraries/IBM/Worklight/db2**.
   - For deployment to a WebSphere Application Server ND cell, use **was_install_dir/profiles/profile-name/config/cells/cell-name/Worklight/db2**.
   - For deployment to a WebSphere Application Server ND cluster, use **was_install_dir/profiles/profile-name/config/cells/cell-name/clusters/cluster-name/Worklight/db2**.

- For deployment to a WebSphere Application Server ND node, use **was_install_dir/profiles/profile-name/config/cells/cell-name/nodes/node-name/Worklight/db2**.
- For deployment to a WebSphere Application Server ND server, use **was_install_dir/profiles/profile-name/config/cells/cell-name/nodes/node-name/servers/server-name/Worklight/db2**.

If this directory does not exist, create it.

2. Add the DB2 JDBC driver JAR file and its associated license files, if any, to the directory that you determined in step 1.
   You can retrieve the driver file in one of two ways:
   - Download it from DB2 JDBC Driver Versions (http://www.ibm.com/support/docview.wss?uid=swg21363866).
   - Fetch it from the **db2_install_dir/java** directory on the DB2 server.
3. In the WebSphere Application Server console, click **Resources → JDBC → JDBC Providers**.
   - Select the appropriate scope from the **Scope** combination box.
   - Click **New**.
   - Set **Database type** to **DB2**.
   - Set **Provider type** to **DB2 Using IBM JCC Driver**.
   - Set **Implementation Type** to **Connection pool data source**.
   - Set **Name** to **DB2 Using IBM JCC Driver**.
   - Click **Next**.
   - Set the class path to the set of JAR files in the directory that you determined in step 1, replacing **was_install_dir/profiles/profile-name** with the WebSphere Application Server variable reference `${USER_INSTALL_ROOT}`.
   - Do not set **Native library path**.
   - Click **Next**.
   - Click **Finish**.
   - The JDBC provider is created.
   - Click **Save**.
4. Create a data source for the Application Center database:
   - Click **Resources → JDBC → Data sources**.
   - Select the appropriate scope from the **Scope** combination box.
   - Click **New** to create a data source.
   - Set the **Data source name** to **Application Center Database**.
   - Set **JNDI Name** to **jdbc/AppCenterDS**.
   - Click **Next**.
   - Enter properties for the data source, for example:
     - **Driver type**: 4
     - **Database Name**: APPCNTR
     - **Server name**: localhost
     - **Port number**: 50000 (default)
   - Click **Next**.
   - Create JAAS-J2C authentication data, specifying the DB2 user name and password as its properties. If necessary, go back to the data source creation wizard, by repeating steps 4.a to 4.h.
   - Select the authentication alias that you created in the **Component-managed authentication alias** combination box (not in the **Container-managed authentication alias** combination box).
   - Click **Next** and **Finish**.
   - Click **Save**.
   - In **Resources → JDBC → Data sources**, select the new data source.
   - Click **WebSphere Application Server data source properties**.
   - Select the **Non-transactional data source** check box.
   - Click **OK**.
   - Click **Save**.
   - Click **Custom properties for the data source**, select the property **currentSchema**, and set the value to the schema used to create the Application Center tables (APPSCHM in this example).
5. Test the data source connection by selecting **Data Source** and clicking **Test Connection**.

Leave **Use this data source in (CMP)** selected.

Configuring Apache Tomcat for DB2 manually for Application Center

If you want to manually set up and configure your DB2 database for Application Center with Apache Tomcat server, use the following procedure.
Before you continue, complete the DB2 database setup procedure.

1. Add the DB2 JDBC driver JAR file.

   You can retrieve this JAR file in one of the following ways:

   - Download it from DB2 JDBC Driver Versions (http://www.ibm.com/support/docview.wss?uid=swg21363866).
   - Or fetch it from the directory **db2_install_dir/java** on the DB2 server) to **$TOMCAT_HOME/lib**.

2. Prepare an XML statement that defines the data source, as shown in the following code example.

   ```
   <Resource auth="Container"
       driverClassName="com.ibm.db2.jcc.DB2Driver"
       name="jdbc/AppCenterDS"
       username="worklight"
       password="password"
       type="javax.sql.DataSource"
       url="jdbc:db2://server:50000/APPCNTR:currentSchema=APPSCHM;"/>
   ```

   The **worklight** parameter after **username=** is the name of the system user with "CONNECT" access to the **APPCNTR** database that you have previously created. The **password** parameter after **password=** is this user's password. If you have defined either a different user name, or a different password, or both, replace these entries accordingly.

   DB2 enforces limits on the length of user names and passwords.

   - For UNIX and Linux systems: 8 characters
   - For Windows: 30 characters

3. Insert this statement in the server.xml file, as indicated in Configuring Apache Tomcat for Application Center manually.

Configuring the Apache Derby database manually for Application Center

You configure the Apache Derby database manually by creating the database and database tables, and then configuring the relevant application server to use this database setup.

1. Create the database and the tables within them. This step is described in Setting up your Apache Derby database manually for Application Center.
2. Configure the application server to use this database setup. Go to one of the following topics.

Jump to

- Setting up your Apache Derby database manually for Application Center
- Configuring Liberty profile for Derby manually for Application Center
- Configuring WebSphere Application Server for Derby manually for Application Center
- Configuring Apache Tomcat for Derby manually for Application Center

Setting up your Apache Derby database manually for Application Center

Set up your Apache Derby database for Application Center by creating the database schema.

1. In the location where you want the database to be created, run **ij.bat** on Windows systems or **ij.sh** on UNIX and Linux systems.

   > **Note:** The ij program is part of Apache Derby. If you do not already have it installed, you can download it fromApache Derby: Downloads (http://db.apache.org/derby/derby_downloads).

   For supported versions of Apache Derby, see System requirements (../../../product-overview/requirements).

   The script displays ij version number.

2. At the command prompt, enter the following commands:

   ```
   connect 'jdbc:derby:APPCNTR;user=APPCENTER;create=true';
   run '<product_install_dir>/ApplicationCenter/databases/create-appcenter-derby.sql';
   quit;
   ```

Configuring Liberty profile for Derby manually for Application Center

If you want to manually set up and configure your Apache Derby database for Application Center with WebSphere Application Server Liberty profile, use the following procedure. Complete the Apache Derby database setup procedure before continuing.

Configure the data source in the $LIBERTY_HOME/usr/servers/worklightServer/server.xml file (worklightServer may be replaced in this path by the name of your server) as follows:

```
<!-- Declare the jar files for Derby access through JDBC. -->
<library id="derbyLib">
  <fileset dir="C:/Drivers/derby" includes="derby.jar" />
</library>

<!-- Declare the IBM Application Center database. -->
<dataSource jndiName="jdbc/AppCenterDS" transactional="false" statementCacheSize="10">
  <jdbcDriver libraryRef="derbyLib"
          javax.sql.ConnectionPoolDataSource="org.apache.derby.jdbc.EmbeddedConnectionPoolDataSource40"/>
  <properties.derby.embedded databaseName="DERBY_DATABASES_DIR/APPCNTR" user="APPCENTER"
                    shutdownDatabase="false" connectionAttributes="upgrade=true"/>
  <connectionManager connectionTimeout="180"
                maxPoolSize="10" minPoolSize="1"
                reapTime="180" maxIdleTime="1800"
                agedTimeout="7200" purgePolicy="EntirePool"/>
</dataSource>
```

Configuring WebSphere Application Server for Derby manually for Application Center

You can set up and configure your Apache Derby database manually for Application Center with WebSphere Application Server. Complete the Apache Derby database setup procedure before continuing.

1. Determine a suitable directory for the JDBC driver JAR file in the WebSphere Application Server installation directory. If this directory does not exist, create it.
    - For a standalone server, you can use a directory such as **was_install_dir/optionalLibraries/IBM/Worklight/derby**.
    - For deployment to a WebSphere Application Server ND cell, use **was_install_dir/profiles/profile-name/config/cells/cell-name/Worklight/derby**.
    - For deployment to a WebSphere Application Server ND cluster, use **was_install_dir/profiles/profile-name/config/cells/cell-name/clusters/cluster-name/Worklight/derby**.
    - For deployment to a WebSphere Application Server ND node, use **was_install_dir/profiles/profile-name/config/cells/cell-name/nodes/node-name/Worklight/derby**.
    - For deployment to a WebSphere Application Server ND server, use **was_install_dir/profiles/profile-name/config/cells/cell-name/nodes/node-name/servers/server-name/Worklight/derby**.
2. Add the **Derby** JAR file from **product_install_dir/ApplicationCenter/tools/lib/derby.jar** to the directory determined in step 1.
3. Set up the JDBC provider.
    - In the WebSphere Application Server console, click **Resources → JDBC → JDBC Providers**.
    - Select the appropriate scope from the **Scope** combination box.
    - Click **New**.
    - Set **Database Type** to **User-defined**.
    - Set **class Implementation name** to **org.apache.derby.jdbc.EmbeddedConnectionPoolDataSource40**.
    - Set **Name** to **Worklight - Derby JDBC Provider**.
    - Set **Description** to **Derby JDBC provider for Worklight**.
    - Click **Next**.
    - Set the **Class path** to the JAR file in the directory determined in step 1, replacing **was_install_dir/profiles/profile-name** with the WebSphere Application Server variable reference **${USER_INSTALL_ROOT}**.
    - Click **Finish**.
4. Create the data source for the **Worklight** database.
    - In the WebSphere Application Server console, click **Resources → JDBC → Data sources**.
    - Select the appropriate scope from the **Scope** combination box.
    - Click **New**.
    - Set **Data source Name** to **Application Center Database**.
    - Set **JNDI** name to **jdbc/AppCenterDS**.
    - Click **Next**.
    - Select the existing JDBC Provider that is named **Worklight - Derby JDBC Provider**.
    - Click **Next**.
    - Click **Next**.
    - Click **Finish**.
    - Click **Save**.
    - In the table, click the **Application Center Database** data source that you created.
    - Under **Additional Properties**, click **Custom properties**.
    - Click **databaseName**.

- Set **Value** to the path to the **APPCNTR** database that is created in Setting up your Apache Derby database manually for Application Center.
- Click **OK**.
- Click **Save**.
- At the top of the page, click **Application Center Database**.
- Under **Additional Properties**, click **WebSphere Application Server data source properties**.
- Select **Non-transactional datasource**.
- Click **OK**.
- Click **Save**.
- In the table, select the **Application Center Database** data source that you created.
- Optional: Only if you are not on the console of a WebSphere Application Server Deployment Manager, click **test connection**.

Configuring Apache Tomcat for Derby manually for Application Center

You can set up and configure your Apache Derby database manually for Application Center with the Apache Tomcat application server. Complete the Apache Derby database setup procedure before continuing.

1. Add the **Derby** JAR file from **product_install_dir/ApplicationCenter/tools/lib/derby.jar** to the directory **$TOMCAT_HOME/lib**.
2. Prepare an XML statement that defines the data source, as shown in the following code example.

```
<Resource auth="Container"
    driverClassName="org.apache.derby.jdbc.EmbeddedDriver"
    name="jdbc/AppCenterDS"
    username="APPCENTER"
    password=""
    type="javax.sql.DataSource"
    url="jdbc:derby:DERBY_DATABASES_DIR/APPCNTR"/>
```

3. Insert this statement in the **server.xml** file, as indicated in Configuring Apache Tomcat for Application Center manually.

Configuring the MySQL database manually for Application Center

You configure the MySQL database manually by creating the database, creating the database tables, and then configuring the relevant application server to use this database setup.

1. Create the database. This step is described in Creating the MySQL database for Application Center.
2. Create the tables in the database. This step is described in Setting up your MySQL database manually for Application Center.
3. Perform the application server-specific setup as the following list shows.

Jump to

Setting up your MySQL database manually for Application Center

Complete the following procedure to set up your MySQL database.

1. Create the database schema.
   - Run a MySQL command line client with the option `-u root`.
   - Enter the following commands:

```
CREATE DATABASE APPCNTR CHARACTER SET utf8 COLLATE utf8_general_ci;
GRANT ALL PRIVILEGES ON APPCNTR.* TO 'worklight'@'Worklight-host'IDENTIFIED BY 'worklight';
GRANT ALL PRIVILEGES ON APPCNTR.* TO 'worklight'@'localhost' IDENTIFIED BY 'worklight';
FLUSH PRIVILEGES;

USE APPCNTR;
SOURCE product_install_dir/ApplicationCenter/databases/create-appcenter-mysql.sql;
```

   Where **worklight** before the "at" sign (@) is the user name, **worklight** after `IDENTIFIED BY` is its password, and **Worklight-host** is the name of the host on which IBM MobileFirst Foundation runs.
2. Add the following property to your MySQL option file: max*allowed*packet=256M.
   For more information about option files, see the MySQL documentation at MySQL.
3. Add the following property to your MySQL option file: innodb*log*file*size = 250M*
   *For more information about the innodb*log*file*size property, see the MySQL documentation, section innodb*log*file_size.

Configuring Liberty profile for MySQL manually for Application Center

If you want to manually set up and configure your MySQL database for Application Center with WebSphere Application Server Liberty profile, use the following procedure. Complete the MySQL database setup procedure before continuing.

> **Note:** MySQL in combination with WebSphere Application Server Liberty profile or WebSphere Application Server full profile is not classified as a supported configuration. For more information, see WebSphere Application Server Support Statement (http://www.ibm.com/support/docview.wss?uid=swg27004311). You can use IBM DB2 or another database supported by WebSphere Application Server to benefit from a configuration that is fully supported by IBM Support.

1. Add the MySQL JDBC driver JAR file to **$LIBERTY_HOME/wlp/usr/shared/resources/mysql**. If that directory does not exist, create it.
2. Configure the data source in the **$LIBERTY_HOME/usr/servers/worklightServer/server.xml** file (**worklightServer** may be replaced in this path by the name of your server) as follows:

```
<!-- Declare the jar files for MySQL access through JDBC. -->
<library id="MySQLLib">
  <fileset dir="${shared.resource.dir}/mysql" includes="*.jar"/>
</library>


<!-- Declare the IBM Application Center database. -->
<dataSource jndiName="jdbc/AppCenterDS" transactional="false">
  <jdbcDriver libraryRef="MySQLLib"/>
  <properties databaseName="APPCNTR"
            serverName="mysqlserver" portNumber="3306"
            user="worklight" password="worklight"/>
</dataSource>
```

where **worklight** after **user=** is the user name, **worklight** after **password=** is this user's password, and **mysqlserver** is the host name of your MySQL server (for example, localhost, if it is on the same machine).

1. You can encrypt the database password with the securityUtility program in `<liberty_install_dir>/bin`.

Configuring WebSphere Application Server for MySQL manually for Application Center

If you want to manually set up and configure your MySQL database for Application Center with WebSphere Application Server, use the following procedure. Complete the MySQL database setup procedure before continuing.

> **Note:** MySQL in combination with WebSphere Application Server Liberty profile or WebSphere Application Server full profile is not classified as a supported configuration. For more information, see WebSphere Application Server Support Statement (http://www.ibm.com/support/docview.wss?uid=swg27004311). We suggest that you use IBM DB2 or another database supported by WebSphere Application Server to benefit from a configuration that is fully supported by IBM Support.

1. Determine a suitable directory for the JDBC driver JAR file in the WebSphere Application Server installation directory.
   - For a standalone server, you can use a directory such as **WAS_INSTALL_DIR/optionalLibraries/IBM/Worklight/mysql**.
   - For deployment to a WebSphere Application Server ND cell, use **WAS_INSTALL_DIR/profiles/profile-name/config/cells/cell-name/Worklight/mysql**.
   - For deployment to a WebSphere Application Serverr ND cluster, use **WAS_INSTALL_DIR/profiles/profile-name/config/cells/cell-name/clusters/cluster-name/Worklight/mysql**.
   - For deployment to a WebSphere Application Server ND node, use **WAS_INSTALL_DIR/profiles/profile-name/config/cells/cell-name/nodes/node-name/Worklight/mysql**.
   - For deployment to a WebSphere Application Server ND server, use **WAS_INSTALL_DIR/profiles/profile-name/config/cells/cell-name/nodes/node-name/servers/server-name/Worklight/mysql**.

   If this directory does not exist, create it.
2. Add the MySQL JDBC driver JAR file downloaded from Download Connector/J (http://dev.mysql.com/downloads/connector/j/) to the directory determined in step 1.
3. Set up the JDBC provider:
   - In the WebSphere Application Server console, click **Resources → JDBC → JDBC Providers**.
   - Select the appropriate scope from the **Scope** combination box.
   - Click **New**.
   - Create a **JDBC provider** named **MySQL**.
   - Set **Database type** to **User defined**.
   - Set **Scope** to **Cell**.
   - Set **Implementation class** to **com.mysql.jdbc.jdbc2.optional.MysqlConnectionPoolDataSource**.

- Set **Database classpath** to the **JAR file** in the directory determined in step 1, replacing **WAS_INSTALL_DIR**/**profiles**/**profile-name** with the WebSphere Application Server variable reference **${USER*INSTALL*ROOT}**.
- Save your changes.

4. Create a data source for the IBM Application Center database:
   - Click **Resources → JDBC → Data sources**.
   - Select the appropriate scope from the **Scope** combination box.
   - Click **New** to create a data source.
   - Type any name (for example, Application Center Database).
   - Set **JNDI Name** to **jdbc**/**AppCenterDS**.
   - Use the existing JDBC Provider MySQL, defined in the previous step.
   - Set **Scope** to **New**.
   - On the **Configuration** tab, select **Non-transactional data source**.
   - Click **Next** a number of times, leaving all other settings as defaults.
   - Save your changes.
5. Set the custom properties of the new data source.
   - Select the new data source.
   - Click **Custom properties**. Set the following properties:

```
portNumber = 3306
relaxAutoCommit=true
databaseName = APPCNTR
serverName = the host name of the MySQL server
user = the user name of the MySQL server
password = the password associated with the user name
```

6. Set the WebSphere Application Server custom properties of the new data source.
   - In **Resources → JDBC → Data sources**, select the **new data source**.
   - Click **WebSphere Application Server data source properties**.
   - Select **Non-transactional data source**.
   - Click **OK**.
   - Click **Save**.

Configuring Apache Tomcat for MySQL manually for Application Center

If you want to manually set up and configure your MySQL database for Application Center with the Apache Tomcat server, use the following procedure. Complete the MySQL database setup procedure before continuing.

1. Add the MySQL Connector/J JAR file to the **$TOMCAT_HOME/lib** directory.
2. Prepare an XML statement that defines the data source, as shown in the following code example. Insert this statement in the server.xml file, as indicated in Configuring Apache Tomcat for Application Center manually.

```
<Resource name="jdbc/AppCenterDS"
        auth="Container"
        type="javax.sql.DataSource"
        maxActive="100"
        maxIdle="30"
        maxWait="10000"
        username="worklight"
        password="worklight"
        driverClassName="com.mysql.jdbc.Driver"
        url="jdbc:mysql://server:3306/APPCNTR"/>
```

Configuring the Oracle database manually for Application Center

You configure the Oracle database manually by creating the database, creating the database tables, and then configuring the relevant application server to use this database setup.

1. Create the database. This step is described in Creating the Oracle database for Application Center.
2. Create the tables in the database. This step is described in Setting up your Oracle database manually for Application Center.
3. Perform the application server-specific setup as the following list shows.

Jump to

- Setting up your Oracle database manually for Application Center

- Configuring Liberty profile for Oracle manually for Application Center
- Configuring WebSphere Application Server for Oracle manually for Application Center
- Configuring Apache Tomcat for Oracle manually for Application Center

Setting up your Oracle database manually for Application Center

Complete the following procedure to set up your Oracle database.

1. Ensure that you have at least one Oracle database.

   In many Oracle installations, the default database has the SID (name) ORCL. For best results, specify **Unicode (AL32UTF8)** as the character set of the database.

   If the Oracle installation is on a UNIX or Linux computer, make sure that the database is started next time the Oracle installation is restarted. To this effect, make sure that the line in /etc/oratab that corresponds to the database ends with a Y, not with an N.
2. Create the user APPCENTER, either by using Oracle Database Control, or by using the Oracle SQLPlus command-line interpreter.
   - To create the user for the Application Center database/schema, by using Oracle Database Control, proceed as follows:
     - Connect as **SYSDBA**.
     - Go to the Users page.
     - Click **Server**, then **Users** in the Security section.
     - Create a user, named **APPCENTER** with the following attributes:

     ```
     Profile: DEFAULT
     Authentication: password
     Default tablespace: USERS
     Temporary tablespace: TEMP
     Status: Unlocked
     Add system privilege: CREATE SESSION
     Add system privilege: CREATE SEQUENCE
     Add system privilege: CREATE TABLE
     Add quota: Unlimited for tablespace USERS
     ```

   - To create the user by using Oracle SQLPlus, enter the following commands:

     ```
     CONNECT SYSTEM/<SYSTEM_password>@ORCL
     CREATE USER APPCENTER IDENTIFIED BY password DEFAULT TABLESPACE USERS QUOTA UNLIMITED ON USERS;
     GRANT CREATE SESSION, CREATE SEQUENCE, CREATE TABLE TO APPCENTER;
     DISCONNECT;
     ```

3. Create the tables for the Application Center database:
   - Using the Oracle SQLPlus command-line interpreter, create the tables for the Application Center database by running the **create-appcenter-oracle.sql** file:

   ```
   CONNECT APPCENTER/APPCENTER_password@ORCL
   @product_install_dir/ApplicationCenter/databases/create-appcenter-oracle.sql
   DISCONNECT;
   ```

4. Download and configure the Oracle JDBC driver:
   - Download the JDBC driver from the Oracle website at Oracle: JDBC, SQLJ, Oracle JPublisher and Universal Connection Pool (UCP) (http://www.oracle.com/technetwork/database/features/jdbc/index-091264.html).
   - Ensure that the Oracle JDBC driver is in the system path. The driver file is **ojdbc6.jar**.

Configuring Liberty profile for Oracle manually for Application Center

You can set up and configure your Oracle database manually for Application Center with WebSphere Application Server Liberty profile by adding the JAR file of the Oracle JDBC driver. Before continuing, set up the Oracle database.

1. Add the JAR file of the Oracle JDBC driver to **$LIBERTY_HOME/wlp/usr/shared/resources/oracle**. If that directory does not exist, create it.
2. If you are using JNDI, configure the data sources in the **$LIBERTY_HOME/wlp/usr/servers/mobileFirstServer/server.xml** file as shown in the following JNDI code example:

   Note: In this path, you can replace mobileFirstServer with the name of your server.

```
<!-- Declare the jar files for Oracle access through JDBC. -->
<library id="OracleLib">
  <fileset dir="${shared.resource.dir}/oracle" includes="*.jar"/>
</library>

<!-- Declare the IBM Application Center database. -->
<dataSource jndiName="jdbc/AppCenterDS" transactional="false">
  <jdbcDriver libraryRef="OracleLib"/>
  <properties.oracle driverType="thin"
                     serverName="oserver" portNumber="1521"
                     databaseName="ORCL"
                     user="APPCENTER" password="APPCENTER_password"/>
</dataSource>
```

where:

- **APPCENTER** after **user=** is the user name,
- **APPCENTER_password** after **password=** is this user's password, and
- **oserver** is the host name of your Oracle server (for example, localhost if it is on the same machine).

> **Note:** For more information on how to connect the Liberty server to the Oracle database with a service name, or with a URL, see the WebSphere Application Server Liberty Core 8.5.5 documentation (http://www-01.ibm.com/support/knowledgecenter/SSD28V_8.5.5/com.ibm.websphere.wlp.core.doc/autodita/rwlp_metatype_core.html?cp=SSD28V_8.5.5%2F1-5-0), section **properties.oracle**.

3. You can encrypt the database password with the securityUtility program in **liberty_install_dir/bin**.

Configuring WebSphere Application Server for Oracle manually for Application Center

If you want to manually set up and configure your Oracle database for Application Center with WebSphere Application Server, use the following procedure. Complete the Oracle database setup procedure before continuing.

1. Determine a suitable directory for the JDBC driver JAR file in the WebSphere Application Server installation directory.
   - For a standalone server, you can use a directory such as WAS *INSTALL*DIR/optionalLibraries/IBM/Worklight/oracle.
   - For deployment to a WebSphere Application Server ND cell, use **WAS_INSTALL_DIR/profiles/profile-name/config/cells/cell-name/Worklight/oracle**.
   - For deployment to a WebSphere Application Server ND cluster, use **WAS_INSTALL_DIR/profiles/profile-name/config/cells/cell-name/clusters/cluster-name/Worklight/oracle**.
   - For deployment to a WebSphere Application Server ND node, use **WAS_INSTALL_DIR/profiles/profile-name/config/cells/cell-name/nodes/node-name/Worklight/oracle**.
   - For deployment to a WebSphere Application Server ND server, use **WAS_INSTALL_DIR/profiles/profile-name/config/cells/cell-name/nodes/node-name/servers/server-name/Worklight/oracle**.

   If this is directory does not exist, create it.
2. Add the Oracle **ojdbc6.jar** file downloaded from JDBC and Universal Connection Pool (UCP) (http://www.oracle.com/technetwork/database/features/jdbc/index-091264.html) to the directory determined in step 1.
3. Set up the JDBC provider:
   - In the WebSphere Application Server console, click **Resources → JDBC → JDBC Providers**.
   - Select the appropriate scope from the **Scope** combination box.
   - Click **New**.
   - Complete the **JDBC Provider** fields as indicated in the following table:

Field	Value
Databasetype	Oracle
Provider type	Oracle JDBC Driver
Implementation type	Connection pool data source
Name	Oracle JDBC Driver

   - Click **Next**.
   - Set the **class path** to the JAR file in the directory determined in step 1, replacing **WAS_INSTALL_DIR/profiles/profile-name** with the WebSphere Application Server variable reference **${USER*INSTALL*ROOT}**
   - Click **Next**.

The JDBC provider is created.

4. Create a data source for the Worklight database:
    - Click **Resources → JDBC → Data sources**.
    - Select the appropriate scope from the **Scope** combination box.
    - Click **New**.
    - Set **Data source name** to **Oracle JDBC Driver DataSource**.
    - Set **JNDI name** to **jdbc/AppCenterDS**.
    - Click **Next**.
    - Click **Select an existing JDBC provider** and select **Oracle JDBC driver** from the list.
    - Click **Next**.
    - Set the **URL** value to **jdbc:oracle:thin:@oserver:1521:ORCL**, where **oserver** is the host name of your Oracle server (for example, **localhost**, if it is on the same machine).
    - Click **Next** twice.
    - Click **Resources → JDBC → Data sources → Oracle JDBC Driver DataSource → Custom properties**.
    - Set **oracleLogPackageName** to **oracle.jdbc.driver**.
    - Set **user = APPCENTER**.
    - Set **password = APPCENTER_password**.
    - Click **OK** and save the changes.
    - In **Resources → JDBC → Data sources**, select the new data source.
    - Click **WebSphere Application Server data source properties**.
    - Select the **Non-transactional data source** check box.
    - Click **OK**.
    - Click **Save**.

Configuring Apache Tomcat for Oracle manually for Application Center

If you want to manually set up and configure your Oracle database for Application Center with the Apache Tomcat server, use the following procedure. Complete the Oracle database setup procedure before continuing.

1. Add the Oracle JDBC driver JAR file to the directory **$TOMCAT_HOME/lib**.
2. Prepare an XML statement that defines the data source, as shown in the following code example. Insert this statement in the server.xml file, as indicated in Configuring Apache Tomcat for Application Center manually

```
<Resource name="jdbc/AppCenterDS"
    auth="Container"
    type="javax.sql.DataSource"
    driverClassName="oracle.jdbc.driver.OracleDriver"
    url="jdbc:oracle:thin:@oserver:1521:ORCL"
    username="APPCENTER"
    password="APPCENTER_password"/>
```

Where **APPCENTER** after **username=** is the name of the system user with "CONNECT" access to the **APPCNTR** database that you have previously created, and **APPCENTER_password** after password= is this user's password. If you have defined either a different user name, or a different password, or both, replace these values accordingly.

Deploying the Application Center WAR files and configuring the application server manually

The procedure to manually deploy the Application Center WAR files manually to an application server depends on the type of application server being configured.
These manual instructions assume that you are familiar with your application server.

> **Note:** Using the MobileFirst Server installer to install Application Center is more reliable than installing manually, and should be used whenever possible.

If you prefer to use the manual process, follow these steps to configure your application server for Application Center. You must deploy the appcenterconsole.war and applicationcenter.war files to your Application Center. The files are located in **product_install_dir/ApplicationCenter/console**.

Jump to

- Configuring the Liberty profile for Application Center manually
- Configuring WebSphere Application Server for Application Center manually
- Configuring Apache Tomcat for Application Center manually

Configuring the Liberty profile for Application Center manually

To configure WebSphere Application Server Liberty profile manually for Application Center, you must modify the **server.xml** file.
In addition to modifications for the databases that are described in Manually installing Application Center, you must make the following modifications to the **server.xml** file.

1. Ensure that the `<featureManager>` element contains at least the following `<feature>` elements:

```
<feature>jdbc-4.0</feature>
<feature>appSecurity-2.0</feature>
<feature>servlet-3.0</feature>
<feature>usr:MFPDecoderFeature-1.0</feature>
```

2. Add the following declarations for Application Center:

```
<!-- The directory with binaries of the 'aapt' program, from the Android SDK's
    platform-tools package. -->
<jndiEntry jndiName="android.aapt.dir" value="product_install_dir/ApplicationCenter/tools/android-sdk"/>
<!-- Declare the Application Center Console application. -->
<application id="appcenterconsole"
             name="appcenterconsole"
             location="appcenterconsole.war"
             type="war">
  <application-bnd>
    <security-role name="appcenteradmin">
      <group name="appcentergroup"/>
    </security-role>
  </application-bnd>
  <classloader delegation="parentLast">
  </classloader>
</application>

<!-- Declare the IBM Application Center Services application. -->
<application id="applicationcenter"
             name="applicationcenter"
             location="applicationcenter.war"
             type="war">
  <application-bnd>
    <security-role name="appcenteradmin">
      <group name="appcentergroup"/>
    </security-role>
  </application-bnd>
  <classloader delegation="parentLast">
  </classloader>
</application>

<!-- Declare the user registry for the IBM Application Center. -->
<basicRegistry id="applicationcenter-registry"
               realm="ApplicationCenter">
  <!-- The users defined here are members of group "appcentergroup",
       thus have role "appcenteradmin", and can therefore perform
       administrative tasks through the Application Center Console. -->
  <user name="appcenteradmin" password="admin"/>
  <user name="demo" password="demo"/>
  <group name="appcentergroup">
    <member name="appcenteradmin"/>
    <member name="demo"/>
  </group>
</basicRegistry>
```

   The groups and users that are defined in the `basicRegistry` are example logins that you can use to test Application Center. Similarly, the groups that are defined in the `<security-role name="appcenteradmin">` for the Application Center console and the Application Center service are examples. For more information about how to modify these groups, see Configuring the Java EE security roles on WebSphere Application Server Liberty profile.

3. If the database is Oracle, add the **commonLibraryRef** attribute to the class loader of the Application Center service application.

```
...
<classloader delegation="parentLast"  commonLibraryRef="OracleLib">
...
```

   The name of the library reference (`OracleLib` in this example) must be the ID of the library that contains the JDBC JAR file. This ID is declared in the procedure that is documented in Configuring Liberty profile for Oracle manually for Application Center.

4. Copy the Application Center WAR files to your Liberty server.

- On UNIX and Linux systems:

```
mkdir -p LIBERTY_HOME/wlp/usr/servers/server_name/apps
cp product_install_dir/ApplicationCenter/console/*.war LIBERTY_HOME/wlp/usr/servers/server_name/apps/
```

- On Windows systems:

```
mmkdir LIBERTY_HOME\wlp\usr\servers\server_name\apps
copy /B product_install_dir\ApplicationCenter\console\appcenterconsole.war
LIBERTY_HOME\wlp\usr\servers\server_name\apps\appcenterconsole.war
copy /B product_install_dir\ApplicationCenter\console\applicationcenter.war
LIBERTY_HOME\wlp\usr\servers\server_name\apps\applicationcenter.war
```

5. Copy the password decoder user feature.
   - On UNIX and Linux systems:

```
mkdir -p LIBERTY_HOME/wlp/usr/extension/lib/features
cp product_install_dir/features/com.ibm.websphere.crypto_1.0.0.jar LIBERTY_HOME/wlp/usr/extension/lib/
cp product_install_dir/features/MFPDecoderFeature-1.0.mf LIBERTY_HOME/wlp/usr/extension/lib/features/
```

   - On Windows systems:

```
mkdir LIBERTY_HOME\wlp\usr\extension\lib
copy /B product_install_dir\features\com.ibm.websphere.crypto_1.0.0.jar
LIBERTY_HOME\wlp\usr\extension\lib\com.ibm.websphere.crypto_1.0.0.jar
mkdir LIBERTY_HOME\wlp\usr\extension\lib\features
copy /B product_install_dir\features\MFPDecoderFeature-1.0.mf
LIBERTY_HOME\wlp\usr\extension\lib\features\MFPDecoderFeature-1.0.mf
```

6. Start the Liberty server.

Configuring WebSphere Application Server for Application Center manually

To configure WebSphere Application Server for Application Center manually, you must configure variables, custom properties, and class loading policies. Make sure that a WebSphere Application Server profile exists.

1. Log on to the WebSphere Application Server administration console for your IBM MobileFirst Server.
2. Enable application security.
   - Click **Security → Global Security**.
   - Ensure that **Enable administrative security** is selected. Application security can be enabled only if administrative security is enabled.
   - Ensure that **Enable application security** is selected.
   - Click **OK**.
   - Save the changes.

   For more information, see Enabling security (http://ibm.biz/knowctr#SSEQTP_7.0.0/com.ibm.websphere.base.doc/info/aes/ae/tsec_csec2.html).
3. Create the Application Center JDBC data source and provider. See the appropriate section in Manually installing Application Center.
4. Install the Application Center console WAR file.
   - Depending on your version of WebSphere Application Server, click one of the following options:
     - **Applications → New → New Enterprise Application**
     - **Applications → New Application → New Enterprise Application**
   - Navigate to the MobileFirst Server installation directory **mfserver_install_dir**/**ApplicationCenter**/**console**.
   - Select **appcenterconsole.war** and click **Next**.
   - On the **How do you want to install the application?** page, click **Detailed**, and then click **Next**.
   - On the **Application Security Warnings** page, click **Continue**.
   - Click **Next** until you reach the "Map context roots for web modules" page.
   - In the **Context Root** field, type /**appcenterconsole**.
   - Click **Next** until you reach the "Map security roles to users or groups" page.
   - Select all roles, click **Map Special Subjects** and select **All Authenticated in Application's Realm**.
   - Click **Next** until you reach the Summary page.
   - Click **Finish** and save the configuration.
5. Configure the class loader policies and then start the application:
   - Click **Applications → Application types → WebSphere Enterprise Applications**.
   - From the list of applications, click **appcenterconsole_war**.

- In the **Detail Properties** section, click the **Class loading and update detection** link.
- In the **Class loader order** pane, click **Classes loaded with local class loader first (parent last)**.
- Click **OK**.
- In the **Modules section**, click **Manage Modules**.
- From the list of modules, click **ApplicationCenterConsole**.
- In the **Class loader order** pane, click **Classes loaded with local class loader first (parent last)**.
- Click **OK** twice.
- Click **Save**.
- Select **appcenterconsole_war** and click (Start).

6. Install the WAR file for Application Center services.
   - Depending on your version of WebSphere Application Server, click one of the following options:
     - **Applications → New → New Enterprise Application**
     - **Applications → New Application → New Enterprise Application**
   - Navigate to the MobileFirst Server installation directory **mfserver_install_dir/ApplicationCenter/console**.
   - Select **applicationcenter.war** and click **Next**.
   - On the **How do you want to install the application?** page, click **Detailed**, and then click **Next**.
   - On the **Application Security Warnings** page, click **Continue**.
   - Click **Next** until you reach the "Map resource references to resources" page.
   - Click **Browser** and select the data source with the **jdbc/AppCenterDS** JNDI name.
   - Click **Apply**.
   - In the **Context Root** field, type /**applicationcenter**.
   - Click **Next** until you reach the "Map security roles to users or groups" page.
   - Select **all roles**, click **Map Special Subjects**, and select **All Authenticated in Application's Realm**.
   - Click **Next** until you reach the **Summary** page.
   - Click **Finish** and save the configuration.
7. Repeat step 5.
   - Select **applicationcenter.war** from the list of applications in substeps b and k.
   - Select **ApplicationCenterServices** in substep g.
8. Review the server class loader policy: Depending on your version of WebSphere Application Server, click **Servers → Server Types → Application Servers or Servers → Server Types → WebSphere application servers** and then select the server.
   - If the class loader policy is set to **Multiple**, do nothing.
   - If the class loader policy is set to **Single** and **Class loading mode** is set to **Classes loaded with local class loader first (parent last)**, do nothing.
   - If **Classloader policy** is set to **Single** and **Class loading mode** is set to **Classes loaded with parent class loader first**, set **Classloader policy** to **Multiple** and set the **classloader policy** of all applications other than MobileFirst applications to **Classes loaded with parent class loader first**.
9. Save the configuration.
10. Configure a JNDI environment entry to indicate the directory with binary files of the aapt program, from the Android SDK platform-tools package.
    - Determine a suitable directory for the aapt binary files in the WebSphere Application Server installation directory.
      - For a stand-alone server, you can use a directory such as **WAS_INSTALL_DIR/optionalLibraries/IBM/mobilefirst/android-sdk**.
      - For deployment to a WebSphere Application Server Network Deployment cell, use **WAS_INSTALL_DIR/profiles/profile-name/config/cells/cell-name/mobilefirst/android-sdk**.
      - For deployment to a WebSphere Application Server Network Deployment cluster, use **WAS_INSTALL_DIR/profiles/profile-name/config/cells/cell-name/clusters/cluster-name/mobilefirst/android-sdk**.
      - For deployment to a WebSphere Application Server Network Deployment node, use **WAS_INSTALL_DIR/profiles/profile-name/config/cells/cell-name/nodes/node-name/mobilefirst/android-sdk**.
      - For deployment to a WebSphere Application Server Network Deployment server, use **WAS_INSTALL_DIR/profiles/profile-name/config/cells/cell-name/nodes/node-name/servers/server-name/mobilefirst/android-sdk**.
    - Copy the **product_install_dir/ApplicationCenter/tools/android-sdk** directory to the directory that you determined in Substep a.
    - For WebSphere Application Server Network Deployment, click **System administration → Nodes**, select the nodes, and click **Full Synchronize**.

- Configure the environment entry (JNDI property) android.aapt.dir, and set as its value the directory that you determined in Substep a. The **WAS_INSTALL_DIR/profiles/profile-name** profile is replaced with the WebSphere Application Server variable reference **${USER_INSTALL_ROOT}**.

You can now access the Application Center at `http://<server>:<port>/appcenterconsole`, where server is the host name of your server and port is the port number (by default 9080).

Configuring Apache Tomcat for Application Center manually

To configure Apache Tomcat for Application Center manually, you must copy JAR and WAR files to Tomcat, add database drivers, edit the **server.xml** file, and then start Tomcat.

1. Add the database drivers to the Tomcat lib directory. See the instructions for the appropriate DBMS in Manually installing Application Center.
2. Edit **tomcat_install_dir/conf/server.xml**.
   - Uncommon the following element, which is initially commented out: `<Valve className="org.apache.catalina.authenticator.SingleSignOn" />`.
   - Declare the Application Center console and services applications and a user registry:

```
<!-- Declare the IBM Application Center Console application. -->
<Context path="/appcenterconsole" docBase="appcenterconsole">

 <!-- Define the AppCenter services endpoint in order for the AppCenter
     console to be able to invoke the REST service.
     You need to enable this property if the server is behind a reverse
     proxy or if the context root of the Application Center Services
     application is different from '/applicationcenter'. -->
 <!-- <Environment name="ibm.appcenter.services.endpoint"
              value="http://proxy-host:proxy-port/applicationcenter"
              type="java.lang.String" override="false"/>
 -->

</Context>

<!-- Declare the IBM Application Center Services application. -->
<Context path="/applicationcenter" docBase="applicationcenter">
 <!-- The directory with binaries of the 'aapt' program, from
     the Android SDK's platform-tools package. -->
 <Environment name="android.aapt.dir"
          value="product_install_dir/ApplicationCenter/tools/android-sdk"
          type="java.lang.String" override="false"/>
 <!-- The protocol of the application resources URI.
     This property is optional. It is only needed if the protocol
     of the external and internal URI are different. -->
 <!-- <Environment name="ibm.appcenter.proxy.protocol"
              value="http" type="java.lang.String" override="false"/>
 -->

 <!-- The host name of the application resources URI. -->
 <!-- <Environment name="ibm.appcenter.proxy.host"
              value="proxy-host"
              type="java.lang.String" override="false"/>
 -->

 <!-- The port of the application resources URI.
     This property is optional. -->
 <!-- <Environment name="ibm.appcenter.proxy.port"
              value="proxy-port"
              type="java.lang.Integer" override="false"/> -->

 <!-- Declare the IBM Application Center Services database. -->
 <!-- <Resource name="jdbc/AppCenterDS" type="javax.sql.DataSource" ... -->

</Context>

<!-- Declare the user registry for the IBM Application Center.
    The MemoryRealm recognizes the users defined in conf/tomcat-users.xml.
    For other choices, see Apache Tomcat's "Realm Configuration HOW-TO"
    http://tomcat.apache.org/tomcat-7.0-doc/realm-howto.html . -->
<Realm className="org.apache.catalina.realm.MemoryRealm"/>
```

   where you fill in the `<Resource>` element as described in one of the sections:

- Configuring Apache Tomcat for DB2 manually for Application Center
- Configuring Apache Tomcat for Derby manually for Application Center
- Configuring Apache Tomcat for MySQL manually for Application Center
- Configuring Apache Tomcat for Oracle manually for Application Center

3. Copy the Application Center WAR files to Tomcat.
   - On UNIX and Linux systems:

     ```
     cp product_install_dir/ApplicationCenter/console/*.war TOMCAT_HOME/webapps/
     ```

   - On Windows systems:

     ```
     copy /B product_install_dir\ApplicationCenter\console\appcenterconsole.war tomcat_install_dir\webapps\appcenterconsole.war
     copy /B product_install_dir\ApplicationCenter\console\applicationcenter.war tomcat_install_dir\webapps\applicationcenter.war
     ```

4. Start Tomcat.

Deploying the Application Center EAR file and configuring the application server manually

As an alternative to the MobileFirst Server installer procedure, you can use a manual procedure to deploy the Application Center EAR file and configure your WebSphere application server manually. These manual instructions assume that you are familiar with your application server.

The procedure to deploy the Application Center EAR file manually to an application server depends on the type of application server. Manual deployment is supported only for WebSphere Application Server Liberty profile and WebSphere Application Server.

> **Tip:** It is more reliable to install Application Center through the MobileFirst Server installer than manually. Therefore, whenever possible, use the MobileFirst Server installer. If, however, you prefer the manual procedure, deploy the **appcentercenter.ear** file, which you can find in the **product_install_dir**/**ApplicationCenter**/**console** directory.

Configuring the Liberty profile for Application Center manually

After you deploy the Application Center EAR file, to configure WebSphere Application Server Liberty profile manually for Application Center, you must modify the server.xml file.

In addition to modifications for the databases that are described in Manually installing Application Center, you must make the following modifications to the **server.xml** file.

1. Ensure that the `<featureManager>` element contains at least the following `<feature>` elements:

   ```
   <feature>jdbc-4.0</feature>
   <feature>appSecurity-2.0</feature>
   <feature>servlet-3.0</feature>
   <feature>usr:MFPDecoderFeature-1.0</feature>
   ```

2. Add the following declarations for Application Center:

```
<!-- The directory with binaries of the 'aapt' program, from the Android SDK's platform-tools package. -->
<jndiEntry jndiName="android.aapt.dir" value="product_install_dir/ApplicationCenter/tools/android-sdk"/>

<!-- Declare the IBM Application Center application. -->
<application id="applicationcenter"
             name="applicationcenter"
             location="applicationcenter.ear"
             type="ear">
  <application-bnd>
    <security-role name="appcenteradmin">
      <group name="appcentergroup"/>
    </security-role>
  </application-bnd>
  <classloader delegation="parentLast">
  </classloader>
</application>

<!-- Declare the user registry for the IBM Application Center. -->
<basicRegistry id="applicationcenter-registry"
               realm="ApplicationCenter">
  <!-- The users defined here are members of group "appcentergroup",
       thus have role "appcenteradmin", and can therefore perform
       administrative tasks through the Application Center Console. -->
  <user name="appcenteradmin" password="admin"/>
  <user name="demo" password="demo"/>
  <group name="appcentergroup">
    <member name="appcenteradmin"/>
    <member name="demo"/>
  </group>
</basicRegistry>
```

The groups and users that are defined in the **basicRegistry** element are example logins, which you can use to test Application Center. Similarly, the groups that are defined in the `<security-role name="appcenteradmin">` element are examples. For more information about how to modify these groups, see Configuring the Java EE security roles on WebSphere Application Server Liberty profile.

3. If the database is Oracle, add the **commonLibraryRef** attribute to the class loader of the Application Center application.

```
...
<classloader delegation="parentLast"  commonLibraryRef="OracleLib">
...
```

The name of the library reference (**OracleLib** in this example) must be the ID of the library that contains the JDBC JAR file. This ID is declared in the procedure that is documented in Configuring Liberty profile for Oracle manually for Application Center.

4. Copy the Application Center EAR files to your Liberty server.
   - On UNIX and Linux systems:

     ```
     mkdir -p LIBERTY_HOME/wlp/usr/servers/server_name/apps
     cp product_install_dir/ApplicationCenter/console/*.ear LIBERTY_HOME/wlp/usr/servers/server_name/apps/
     ```

   - On Windows systems:

     ```
     mkdir LIBERTY_HOME\wlp\usr\servers\server_name\apps
     copy /B product_install_dir\ApplicationCenter\console\applicationcenter.ear
     LIBERTY_HOME\wlp\usr\servers\server_name\apps\applicationcenter.ear
     ```

5. Copy the password decoder user feature.
   - On UNIX and Linux systems:

     ```
     mkdir -p LIBERTY_HOME/wlp/usr/extension/lib/features
     cp product_install_dir/features/com.ibm.websphere.crypto_1.0.0.jar LIBERTY_HOME/wlp/usr/extension/lib/
     cp product_install_dir/features/MFPDecoderFeature-1.0.mf LIBERTY_HOME/wlp/usr/extension/lib/features/
     ```

   - On Windows systems:

```
mkdir LIBERTY_HOME\wlp\usr\extension\lib
copy /B product_install_dir\features\com.ibm.websphere.crypto_1.0.0.jar
LIBERTY_HOME\wlp\usr\extension\lib\com.ibm.websphere.crypto_1.0.0.jar
mkdir LIBERTY_HOME\wlp\usr\extension\lib\features
copy /B product_install_dir\features\MFPDecoderFeature-1.0.mf
LIBERTY_HOME\wlp\usr\extension\lib\features\MFPDecoderFeature-1.0.mf
```

6. Start the Liberty server.

Configuring WebSphere Application Server for Application Center manually

After you deploy the Application Center EAR file, to configure WebSphere Application Server profile manually for Application Center, you must configure variables, custom properties, and class loader policies. Make sure that a WebSphere Application Server profile exists.

1. Log on to the WebSphere Application Server administration console for your IBM MobileFirst Server.
2. Enable application security.
   - Click **Security → Global Security**.
   - Ensure that **Enable administrative security** is selected. Application security can be enabled only if administrative security is enabled.
   - Ensure that **Enable application security** is selected.
   - Click **OK**.
   - Save the changes.

   For more information, see Enabling security (http://ibm.biz/knowctr#SSEQTP_7.0.0/com.ibm.websphere.base.doc/info/aes/ae/tsec_csec2.html).
3. Create the Application Center JDBC data source and provider. See the appropriate section in Manually installing Application Center.
4. Install the Application Center console WAR file.
   - Depending on your version of WebSphere Application Server, click one of the following options:
     - **Applications → New → New Enterprise Application**
     - **Applications → New Application → New Enterprise Application**
   - Navigate to the MobileFirst Server installation directory **mfserver_install_dir**/**ApplicationCenter**/**console**.
   - Select **appcenterconsole.war** and click **Next**.
   - On the **How do you want to install the application?** page, click **Detailed**, and then click **Next**.
   - On the **Application Security Warnings** page, click **Continue**.
   - Click **Next** until you reach the "Map context roots for web modules" page.
   - In the **Context Root** field, type /**appcenterconsole**.
   - Click **Next** until you reach the "Map security roles to users or groups" page.
   - Select all roles, click **Map Special Subjects** and select **All Authenticated in Application's Realm**.
   - Click **Next** until you reach the Summary page.
   - Click **Finish** and save the configuration.
5. Configure the class loader policies and then start the application:
   - Click **Applications → Application types → WebSphere Enterprise Applications**.
   - From the list of applications, click **AppCenterEAR**.
   - In the **Detail Properties** section, click the **Class loading and update detection** link.
   - In the **Class loader order** pane, click **Classes loaded with local class loader first (parent last)**.
   - Click **OK**.
   - In the **Modules section**, click **Manage Modules**.
   - From the list of modules, click **ApplicationCenterConsole**.
   - In the **Class loader order** pane, click **Classes loaded with local class loader first (parent last)**.
   - Click **OK**.
   - From the list of modules, click **ApplicationCenterServices**.
   - In the **Class loader order** pane, click **Classes loaded with local class loader first (parent last)**.
   - Click **OK** twice.
   - Click **Save**.
   - Select **appcenterconsoleEAR** and click **Start**.
6. Review the server class loader policy:

   Depending on your version of WebSphere Application Server, click **Servers → Server Types → Application Servers or Servers → Server Types → WebSphere application servers** and then select the server. * If the class loader policy is set to **Multiple**, do nothing. * If the class loader policy is set to **Single** and **Class loading mode** is set to **Classes loaded with local class loader first**

(parent last), do nothing. * If **Classloader policy** is set to **Single** and **Class loading mode** is set to **Classes loaded with parent class loader first**, set **Classloader policy** to **Multiple** and set the **classloader policy** of all applications other than MobileFirst applications to **Classes loaded with parent class loader first**.

7. Save the configuration.
8. Configure a JNDI environment entry to indicate the directory with binary files of the **aapt** program, from the Android SDK **platform-tools** package.
    - Determine a suitable directory for the aapt binary files in the WebSphere Application Server installation directory.
        - For a stand-alone server, you can use a directory such as **WAS_INSTALL_DIR/optionalLibraries/IBM/mobilefirst/android-sdk**.
        - For deployment to a WebSphere Application Server Network Deployment cell, use **WAS_INSTALL_DIR/profiles/profile-name/config/cells/cell-name/mobilefirst/android-sdk**.
        - For deployment to a WebSphere Application Server Network Deployment cluster, use **WAS_INSTALL_DIR/profiles/profile-name/config/cells/cell-name/clusters/cluster-name/mobilefirst/android-sdk**.
        - For deployment to a WebSphere Application Server Network Deployment node, use **WAS_INSTALL_DIR/profiles/profile-name/config/cells/cell-name/nodes/node-name/mobilefirst/android-sdk**.
        - For deployment to a WebSphere Application Server Network Deployment server, use **WAS_INSTALL_DIR/profiles/profile-name/config/cells/cell-name/nodes/node-name/servers/server-name/mobilefirst/android-sdk**.
    - Copy the **product_install_dir/ApplicationCenter/tools/android-sdk** directory to the directory that you determined in Substep a.
    - For WebSphere Application Server Network Deployment, click **System administration → Nodes**, select the nodes, and click **Full Synchronize**.
    - Configure the environment entry (JNDI property) **android.aapt.dir** and set as its value the directory that you determined in Substep a. The **WAS_INSTALL_DIR/profiles/profile-name** profile is replaced with the WebSphere Application Server variable reference **${USER_INSTALL_ROOT}**.

    You can now access the Application Center at `http://<server>:<port>/appcenterconsole`, where server is the host name of your server and port is the port number (by default 9080).

Configuring Application Center after installation

After you install Application Center in the web application server that you designated, you have additional configuration to do.

Jump to

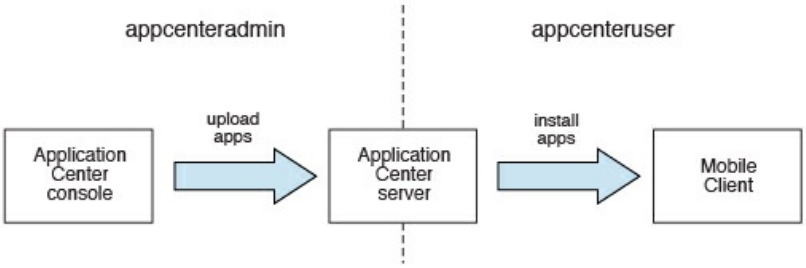
Configuring user authentication for Application Center

You configure user authentication and choose an authentication method. The configuration procedure depends on the web application server that you use. **Application Center requires user authentication**.

You must perform some configuration after the installer deploys Application Center web applications in the web application server. Application Center has two Java™ Platform, Enterprise Edition (Java EE) security roles defined:

- The **appcenteruser** role that represents an ordinary user of Application Center who can install mobile applications from the catalog to a mobile device belonging to that user.
- The **appcenteradmin** role that represents a user who can perform administrative tasks through the Application Center console.

You must map the roles to the corresponding sets of users.



If you choose to use an authentication method through a user repository such as LDAP, you can configure Application Center so that you can use users and groups with the user repository to define the Access Control List (ACL) of Application Center. This procedure is conditioned by the type and version of the web application server that you use. See Managing users with LDAP for information about LDAP used with Application Center.

After you configure authentication of the users of Application Center, which includes configuring LDAP if you plan to use it, you can, if necessary, define the endpoint of the application resources. You must then build the Application Center mobile client. The mobile client is used to install applications on mobile devices. See Preparations for using the mobile client (../../../appcenter/preparations/) for how to build the Application Center mobile client.

Jump to

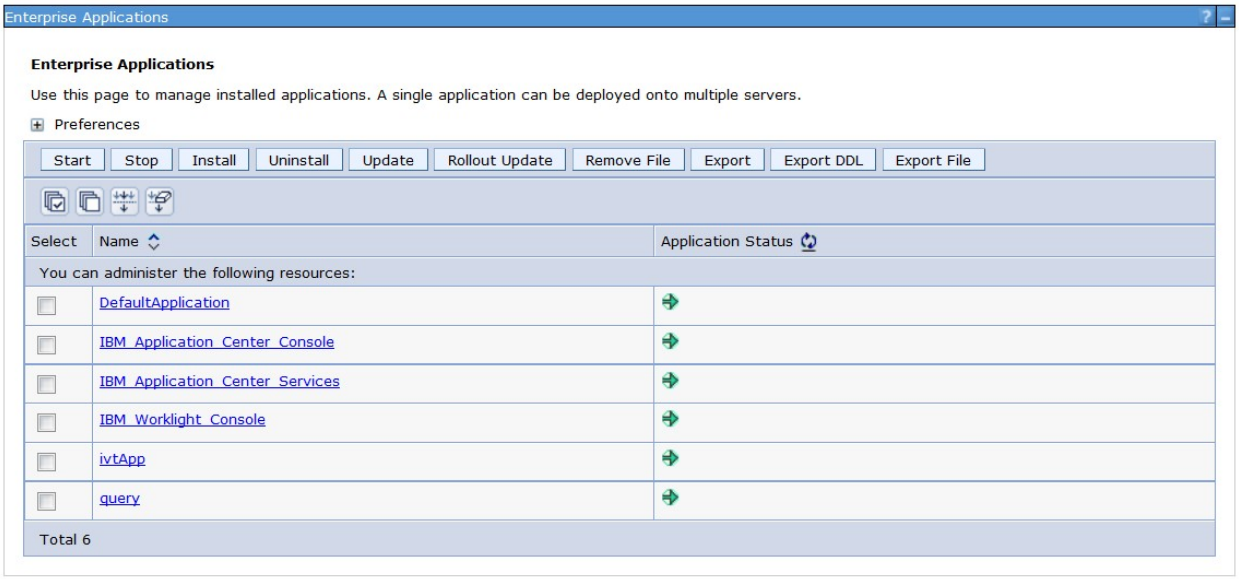
- Configuring the Java EE security roles on WebSphere Application Server full profile
- Configuring the Java EE security roles on WebSphere Application Server Liberty profile
- Configuring the Java EE security roles on Apache Tomcat

Configuring the Java EE security roles on WebSphere Application Server full profile

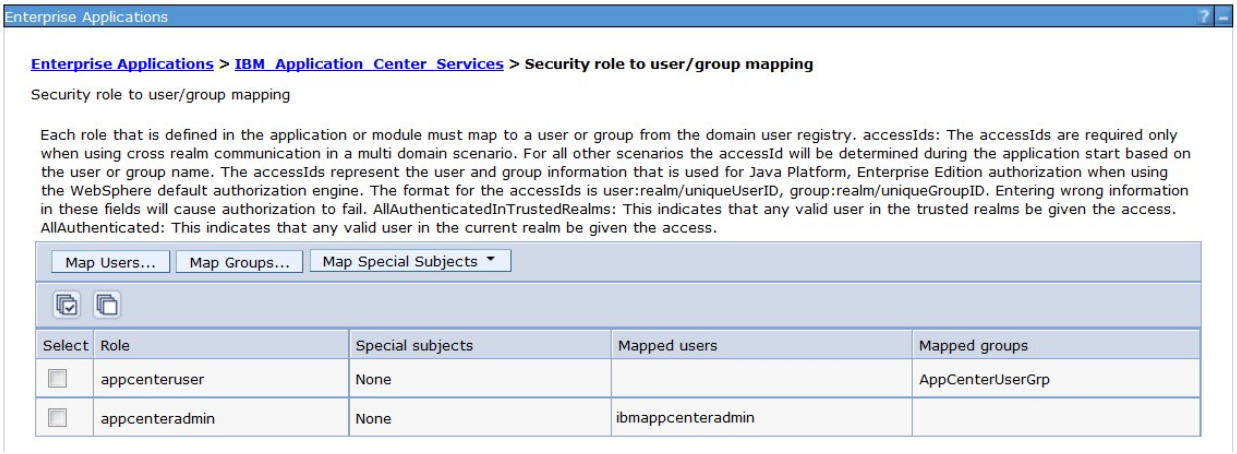
Configure security by mapping the Application Center Java™ EE roles to a set of users for both web applications.

You define the basics of user configuration in the WebSphere Application Server console. Access to the console is usually by this address: `https://localhost:9043/ibm/console/`.

1. Select **Security → Global Security**.
2. Select **Security Configuration Wizard** to configure users.
   You can manage individual user accounts by selecting **Users and Groups → Manage Users**.
3. If you deployed WAR files, map the **appcenteruser** and **appcenteradmin** roles to a set of users as follows:
   - Select **Servers → Server Types → WebSphere application servers**.
   - Select the server.
   - In the Configuration tab, select **Applications → Enterprise applications**.



   - Select **IBM_Application_Center_Services**.
   - In the Configuration tab, select **Details → Security role to user/group mapping**.



- Perform the necessary customization.
- Click **OK**.
- Repeat the steps to map the roles for the console web application; select **IBM_Application_Center_Console**.
- Click **Save** to save the changes.

4. If you deployed an EAR file, map the appcenteruser and appcenteradmin roles to a set of users as follows:
   - Select **Applications → Application Types → WebSphere application servers**.
   - Click **AppCenterEAR**.
   - In the **Detail Properties** section, click **Security role to user/group mapping**.
   - Customize as necessary.
   - Click **OK**.
   - Click **Save**.

Configuring the Java EE security roles on WebSphere Application Server Liberty profile

Configure the Java™ EE security roles of the Application Center and the data source in the **server.xml** file.

To configure the security roles, you must edit the **server.xml** file. In the `<application-bnd>` element of each `<application>` element, create two `<security-role>` elements. One `<security-role>` element is for the **appcenteruser** role and the other is for the **appcenteradmin** role. Map the roles to the appropriate user group name **appcenterusergroup** or **appcenteradmingroup**. These groups are defined through the `<basicRegistry>` element. You can customize this element or replace it entirely with an `<ldapRegistry>` element or a `<safRegistry>` element.

Then, to maintain good response times with a large number of installed applications, for example with 80 applications, you should configure a connection pool for the Application Center database.

1. Edit the **server.xml** file. For example:

```
<security-role name="appcenteradmin">
  <group name="appcenteradmingroup"/>
</security-role>
<security-role name="appcenteruser">
  <group name="appcenterusergroup"/>
</security-role>
```

   You must include this example in the following location:
   - If you deployed WAR files, in the `<application-bnd>` element of each `<application>` element: the **appcenterconsole** and **applicationcenter** applications.
   - If you deployed an EAR file, in the `<application-bnd>` element of the **applicationcenter** application.

   Replace the `<security-role>` elements that have been created during installation for test purposes.

```
<basicRegistry id="appcenter">
  <user name="admin" password="admin"/>
  <user name="guest" password="guest"/>
  <user name="demo" password="demo"/>
  <group name="appcenterusergroup">
    <member name="guest"/>
    <member name="demo"/>
  </group>
  <group name="appcenteradmingroup">
    <member name="admin" id="admin"/>
  </group>
</basicRegistry>
```

This example shows a definition of users and groups in the `basicRegistry` of WebSphere Application Server Liberty. For more information about configuring a user registry for WebSphere Application Server Liberty profile, see Configuring a user registry for the Liberty profile (http://www-01.ibm.com/support/knowledgecenter/SSD28V_8.5.5/com.ibm.websphere.wlp.core.doc/ae/twlp_sec_registries.html).

2. Edit the **server.xml** file to define the `AppCenterPool` size.

```
<connectionManager id="AppCenterPool" minPoolSize="10" maxPoolSize="40"/>
```

3. In the `<dataSource>` element, define a reference to the connection manager:

```
<dataSource id="APPCNTR" jndiName="jdbc/AppCenterDS" connectionManagerRef="AppCenterPool">
...
</dataSource>
```

Configuring the Java EE security roles on Apache Tomcat

You must configure the Java™ EE security roles for the Application Center on the Apache Tomcat web application server.

1. In the Apache Tomcat web application server, you configure the roles of **appcenteruser** and **appcenteradmin** in the **conf/tomcat-users.xml** file. The installation creates the following users:

```
<user username="appcenteradmin" password="admin" roles="appcenteradmin"/>
<user username="demo" password="demo" roles="appcenteradmin"/>
<user username="guest" password="guest" roles="appcenteradmin"/>
```

2. You can define the set of users as described in the Apache Tomcat documentation, Realm Configuration HOW-TO (http://tomcat.apache.org/tomcat-7.0-doc/realm-howto.html).

Managing users with LDAP

Use the Lightweight Directory Access Protocol (LDAP) registry to manage users.

LDAP is a way to centralize the user management for multiple web applications in an LDAP Server that maintains a user registry. It can be used instead of specifying one by one the users for the security roles **appcenteradmin** and **appcenteruser**.

If you plan to use an LDAP registry with the Application Center, you must configure your WebSphere Application Server or your Apache Tomcat server to use an LDAP registry to authenticate users.

In addition to authentication of users, configuring the Application Center for LDAP also enables you to use LDAP to define the users and groups who can install mobile applications through the Application Center. The means of defining these users and groups is the Access Control List (ACL).

Since IBM Worklight V6.0, use the JNDI environment entries for defining LDAP configuration properties.

Expert users could configure the application servers to use LDAP authentication by using the methods that were documented in releases before IBM Worklight V6.0.

Jump to

LDAP with WebSphere Application Server V8.x

LDAP authentication is based on the federated repository configuration. ACL management configuration of the Application Center uses the Virtual Member Manager API.

You must configure LDAP based on the federated repository configuration. The stand-alone LDAP registry is not supported.

Several different repositories, LDAP and non-LDAP, can be configured in the federated repository.

For information about configuring federated repositories, see the WebSphere Application Server V8.0 (http://ibm.biz/knowctr#/SSEQTP_8.0.0/as_ditamaps/welcome_base.html) user documentation or the WebSphere Application Server V8.5 (http://ibm.biz/knowctr#SSEQTP_8.5.5/as_ditamaps/was855_welcome_base_dist_iseries.html) user documentation, depending on your version.

Configuration of the Application Center for ACL management with LDAP

Some configuration details of ACL management are specific to the Application Center, because it uses the Virtual Member Manager (VMM) API.

The Application Center refers to these VMM attributes for users:

- **uid** represents the user login name.
- **sn** represents the full name of the user.

- For groups, the Application Center refers only to the VMM attribute **cn**.

If VMM attributes are not identical in LDAP, you must map the VMM attributes to the corresponding LDAP attributes.

- Configuring LDAP authentication for WebSphere Application Server V8.x
- Configuring LDAP ACL management for WebSphere Application Server V8.x

Configuring LDAP authentication for WebSphere Application Server V8.x

You can configure LDAP based on the federated repository configuration only. This procedure shows you how to use LDAP to define the roles appcenteradmin and appcenteruser in WebSphere Application Server V8.x.

1. Log in to the WebSphere Application Server console.
2. Select **Security → Global security** and verify that administrative security and application security are enabled.
3. In the "User account repository" section, select **Federated repositories**.
4. Click **Configure**.
5. Add a repository and configure it.
    - Click **Add Base entry to Realm**.
    - Specify the value of **Distinguished name of a base entry that uniquely identifies entries in the realm** and click **Add Repository**.
    - Select **LDAP Repository**.
    - Give this repository a name and enter the values that are required to connect to your LDAP server.
    - Under **Additional Properties**, click **LDAP entity types**.
    - Configure the **Group**, **OrgContainer**, and **PersonAccount** properties. These configuration details depend on your LDAP server.
6. Save the configuration, log out, and restart the server.
7. If you deployed WAR files, in the WebSphere Application Server console, map the security roles to users and groups.
    - In the **Configuration** tab, select **Applications → WebSphere Enterprise applications**.
    - Select **IBM*Application*Center_Services**.
    - In the **Configuration** tab, select **Details → Security role to user/group mapping**.
    - For **appcenteradmin** and **appcenteruser** roles, select **Map groups**. This selection enables you to select users and groups inside the WebSphere user repository, including LDAP users and groups. The selected users can access the Application Center as **appcenteradmin** or **appcenteruser**. You can also map the roles to **Special Subjects** “All authenticated in application realm” to give everyone in the WebSphere user repository, including everyone registered in the LDAP registry, access to the Application Center.
8. Repeat step 7 for **IBM*Application*Center_Console**.
   Make sure that you select **IBM*Application*Center_Console** in step 7.b instead of **IBM*Application*Center_Services**.
9. If you deployed an EAR file, in the WebSphere Application Server console, map the security roles to users and groups.
    - Click **Applications → Application Types → WebSphere enterprise applications**.
    - From the list of applications, click **AppCenterEAR**.
    - In the **Detail Properties** section, click **Security role to user/group mapping**.
    - For **appcenteradmin** and **appcenteruser** roles, select **Map groups** or **Map users** to select users or groups inside the WebSphere user repository, including LDAP users and groups.

   The selected users can access the Application Center as **appcenteradmin** or **appcenteruser**. You can also map the roles to **Special Subjects** “All authenticated in application realm” to give access to the Application Center to everyone in the WebSphere user repository, including everyone registered in the LDAP registry.
10. Click **Save** to save your changes.

Configuring LDAP ACL management for WebSphere Application Server V8.x

To configure ACL with LDAP, you define three properties: **uid**, **sn**, and **cn**. These properties enable the login name and the full name of users and the name of user groups to be identified in the Application Center. Then you enable ACL management with VMM. You can configure LDAP based on the federated repository configuration only.

1. Log in to the WebSphere Application Server console.
2. Select **Security → Global security**.
3. In the **User account repository** section, select **Configure**.
4. Select your LDAP repository entry.
5. Under **Additional Properties**, select **LDAP attributes** (WebSphere Application Server V8.0) or **Federated repositories property names to LDAP attributes mapping** (WebSphere Application Server V8.5).
6. Select **Add → Supported**.
7. Enter these property values:

- For **Name** enter your LDAP login attribute.
- For **Property** name enter **uid**.
- For **Entity types** enter the LDAP entity type.
- Click **OK**.

8. Select **Add → Supported**.
   - For **Name** enter your LDAP attribute for full user name.
   - For **Property** name enter **sn**.
   - For **Entity types** enter the LDAP entity type.
   - Click **OK**.

9. Select **Add → Supported** to configure a group name:
   - For **Name** enter the LDAP attribute for your group name.
   - For **Property** name enter **cn**.
   - For **Entity types** enter the LDAP entity type.
   - Click **OK**.
10. Enable ACL management with LDAP:
    - Select **Servers → Server Types → WebSphere application servers**.
    - Select the appropriate application server.
      In a clustered environment you must configure all the servers in the cluster in the same way.
    - In the **Configuration** tab, under **Server Infrastructure**, click the **Java and Process Management** tab and select **Process definition**.
    - In the **Configuration** tab, under **Additional Properties**, select **Java Virtual Machine**,
    - In the **Configuration** tab, under **Additional Properties**, select **Custom properties**.
    - Enter the required property-value pairs in the form. To enter each pair, click **New**, enter the property and its value, and click **OK**.

Property-value pairs:

- ibm.appcenter.ldap.vmm.active = true
- ibm.appcenter.ldap.active = true
- ibm.appcenter.ldap.cache.expiration.seconds = delay*in*seconds

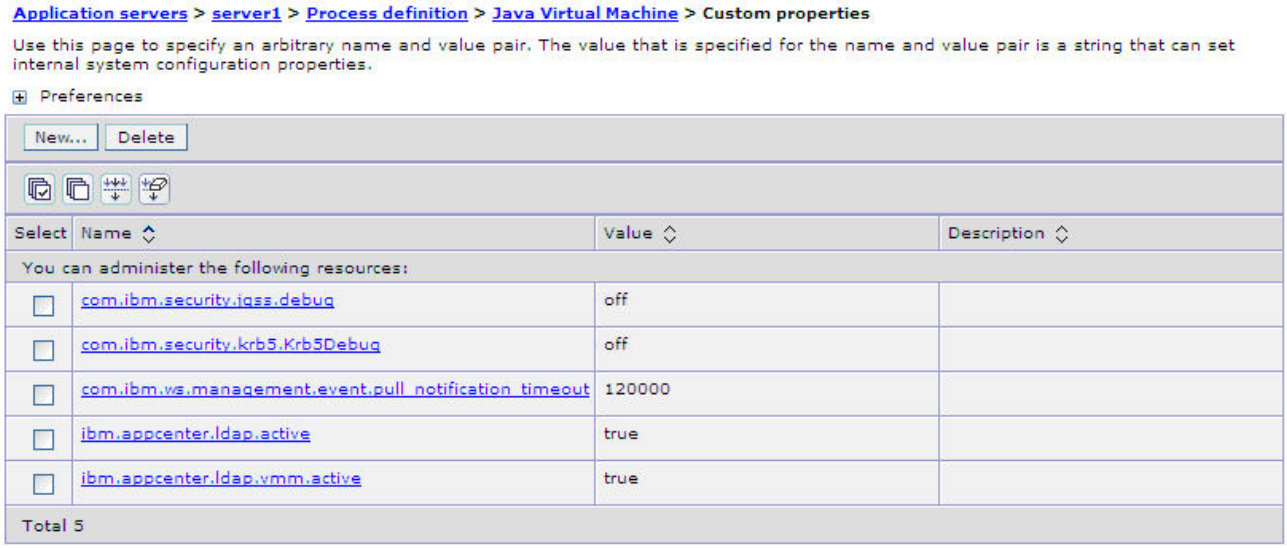
- Enter the delay in seconds before the LDAP cache expires. If you do not enter a value, the default value is 86400, which is equal to 24 hours.

Changes to users and groups on the LDAP server become visible to the Application Center after a delay, which is specified by **ibm.appcenter.ldap.cache.expiration.seconds**. The Application Center maintains a cache of LDAP data and the changes become visible only after the cache expires. By default, the delay is 24 hours. If you do not want to wait for this delay to expire after changes to users or groups, you can call this command to clear the cache of LDAP data:

```
acdeploytool.sh -clearLdapCache -s serverurl -c context -u user -p password
```

See Using the stand-alone tool to clear the LDAP cache (../../../appcenter/command-line/#using-the-stand-alone-tool-to-clear-the-ldap-cache) for details.

The following figure shows an example of custom properties with the correct settings.



What to do next

1. Save the configuration and restart the server.
2. To use the VMM API, you must assign the **ldMgrReader** role to the users who run the VMM code, or to the group owners of these users. You must assign this role to all users and groups who have the **appcenteruser** or **appcenteradminroles**.
3. In the **was_home\bin** directory, where **was_home** is the home directory of your WebSphere Application Server, run the **wsadmin** command.
4. After connecting with the WebSphere Application Server administrative user, run the following command:

   ```
   $AdminTask mapIdMgrGroupToRole {-roleName ldMgrReader -groupId your_LDAP_group_id}
   ```

5. Run the same command for all the groups mapped to the **appcenteruser** and **appcenteradminroles**. For individual users who are not members of groups, run the following command:

   ```
   $AdminTask mapIdMgrUserToRole {-roleName ldMgrReader -userId your_LDAP_user_id}
   ```

   You can assign the special subject "All Authenticated in Application's Realm" as roles for appcenteruser and appcenteradmin. If you choose to assign this special subject, ldMgrReader must be configured in the following way:

   ```
   $AdminTask mapIdMgrGroupToRole {-roleName ldMgrReader -groupId ALLAUTHENTICATED}
   ```

6. Enter **exit** to end **wsadmin**.

LDAP with Liberty profile

Use LDAP to authenticate users and to define the users and groups who can install mobile applications with the Application Center by using the JNDI environment.

Using LDAP with Liberty profile requires you to configure LDAP authentication and LDAP ACL management.

- Configuring LDAP authentication for the Liberty profile
- Configuring LDAP ACL management (Liberty profile)

Configuring LDAP authentication for the Liberty profile

You can configure LDAP authentication of users and groups in the **server.xml** file by defining an LDAP registry or, since WebSphere Application Server Liberty profile V8.5.5, a federated registry that uses several LDAP registries. Then, you map users and groups to Application Center roles. The mapping configuration is the same for LDAP authentication and basic authentication.

1. To open the **server.xml** descriptor file, enter **{server.config.dir}/server.xml**
2. Insert one or several LDAP registry definitions after the `<httpEndpoint>` element. Example for the LDAP registry:

```
<ldapRegistry baseDN="o=ibm.com" host="employees.com" id="Employees"
        ldapType="IBM Tivoli Directory Server" port="389" realm="AppCenterLdap"
        recursiveSearch="true">
 <idsFilters
    groupFilter="(&amp;(cn=%v)(|(objectclass=groupOfNames)(objectclass=groupOfUniqueNames))) " id="Employees"
    userFilter="(&amp;(emailAddress=%v)(objectclass=ibmPerson))"
    groupMemberIdMap="ibm-allGroups:member;ibm-allGroups:uniqueMember"
    userIdMap="*:emailAddress"/>
</ldapRegistry>
```

   For information about the parameters that are used in this example, see the WebSphere Application Server V8.5 (http://ibm.biz/knowctr#SSEQTP_8.5.5/as_ditamaps/was855_welcome_base_dist_iseries.html) user documentation.
3. Insert a security role definition after each Application Center application definition.
   - If you deployed WAR files: **applicationcenter** and **appcenterconsole**
   - If you deployed an EAR file: **applicationcenter**

   Group names unique within LDAP

   This sample code shows how to use the group names **ldapGroupForAppcenteruser** and **ldapGroupForAppcenteradmin** when they exist and are unique within LDAP.

```
<application-bnd>
   <security-role name="appcenteruser" id="appcenteruser">
     <group name="ldapGroupForAppcenteruser" />
   </security-role>
   <security-role name="appcenteradmin" id="appcenteradmin">
     <group name="ldapGroupForAppcenteradmin" />
   </security-role>
</application-bnd>
```

   Group names not unique within LDAP

   This sample code shows how to code the mapping when the group names are not unique within LDAP. The groups must be specified with the **access-id** attribute. The **access-id** attribute must refer to the realm name that is used to specify the LDAP realm. In this sample code, the realm name is **AppCenterLdap**. The remainder of the **access-id** attribute specifies one of the LDAP groups named **ldapGroup** in a way that makes it unique.

```
<application-bnd>
   <security-role name="appcenteruser" id="appcenteruser">
     <group name="ldapGroup"
          id="ldapGroup"
          access-id="group:AppCenterLdap/CN=ldapGroup,OU=myorg,
                DC=mydomain,DC=AD,DC=myco,DC=com"/>
   </security-role>
   ...
</application-bnd>
```

   If applicable, use similar code to map the **appcenteradmin** role.

Configuring LDAP ACL management (Liberty profile)

You enable ACL management after you configure LDAP and map users and groups to Application Center roles. Only the simple type of LDAP authentication is supported.

To be able to define JNDI entries, the following feature must be defined in the **server.xml** file:

```
<feature>jndi-1.0</feature>
```

Add an entry for each property in the `<server>` section of the **server.xml** file. This entry should have the following syntax:

```
<jndiEntry jndiName="JNDI_property_name" value="property_value"/>
```

Where:

- **JNDI_property_name** is the name of the property you are adding.
- **property_value** is the value of the property you are adding.

Property	Description
ibm.appcenter.ldap.active	Set to true to enable LDAP; set to false to disable LDAP.
ibm.appcenter.ldap.federated.active	Since WebSphere Application Server Liberty profile V8.5.5: set to true to enable use of the federated registry; set to false to disable use of the federated registry, which is the default setting.
ibm.appcenter.ldap.connectionURL	LDAP connection URL.
ibm.appcenter.ldap.user.base	Search base of users.
ibm.appcenter.ldap.user.loginName	LDAP login attribute.
ibm.appcenter.ldap.user.displayName	LDAP attribute for the user name to be displayed, for example, a person's full name.
ibm.appcenter.ldap.group.base	Search base of groups.
ibm.appcenter.ldap.group.name	LDAP attribute for the group name.
ibm.appcenter.ldap.group.uniquemember	LDAP attribute that identifies the members of a group.
ibm.appcenter.ldap.user.groupmembership	LDAP attribute that identifies the groups to which a user belongs.
ibm.appcenter.ldap.group.nesting	Management of nested groups: if nested groups are not managed, set the value to false.
ibm.appcenter.ldap.user.filter	LDAP user search filter for the attribute of user login name. Use %v as the placeholder for the login name attribute. This property is only required when LDAP users and groups are defined in the same subtree; that is, when the properties **ibm.appcenter.ldap.user.base** and **ibm.appcenter.ldap.group.base** have the same value.
ibm.appcenter.ldap.displayName.filter	LDAP user search filter for the attribute of user display name. Use %v as the placeholder for the display name attribute. This property is only required when LDAP users and groups are defined in the same subtree; that is, when the properties **ibm.appcenter.ldap.user.base** and **ibm.appcenter.ldap.group.base** have the same value.
ibm.appcenter.ldap.group.filter	LDAP group search filter. Use %v as the placeholder for the group attribute. This property is only required when LDAP users and groups are defined in the same subtree; that is, when the properties **ibm.appcenter.ldap.user.base** and **ibm.appcenter.ldap.group.base** have the same value.
ibm.appcenter.ldap.security.sasl	The value of the security authentication mechanism when the LDAP external SASL authentication mechanism is required to bind to the LDAP server. The value depends on the LDAP server; usually, it is set to "EXTERNAL".
ibm.appcenter.ldap.security.binddn	Property that identifies the distinguished name of the user permitted to search the LDAP directory. Use this property only if security binding is required.
ibm.appcenter.ldap.security.bindpwd	Property that identifies the password of the user who is allowed to search the LDAP directory. Use this property only if security binding is required. The password can be encoded with the "Liberty profile securityUtility" tool. Run the tool and then set the value of this property to the encoded password generated by the tool. The supported encoding types are xor and aes. Edit the Liberty profile server.xml file to check whether the classloader is enabled to load the JAR file that decodes the password.

Property	Description
ibm.appcenter.ldap.cache.expiration.seconds	elay in seconds before the LDAP cache expires. If no value is entered, the default value is 86400, which is equal to 24 hours. Changes to users and groups on the LDAP server become visible to the Application Center after a delay, which is specified by **ibm.appcenter.ldap.cache.expiration.seconds**. The Application Center maintains a cache of LDAP data and the changes only become visible after the cache expires. By default, the delay is 24 hours. If you do not want to wait for this delay to expire after changes to users or groups, you can call this command to clear the cache of LDAP data: `acdeploytool.sh -clearLdapCache -s serverurl -c context -u user -p password` See Using the stand-alone tool to clear the LDAP cache (../../../appcenter/command-line/#using-the-stand-alone-tool-to-clear-the-ldap-cache) for details.
ibm.appcenter.ldap.referral	Property that indicates whether referrals are supported by the JNDI API. If no value is given, the JNDI API will not handle LDAP referrals. Possible values are: • ignore: ignores referrals found in the LDAP server. • follow: automatically follows any referrals found in the LDAP server. • throw: causes an exception to occur for each referral found in the LDAP server.

See JNDI properties for Application Center for a complete list of LDAP properties that you can set.

Example of setting properties for ACL management with LDAP
This example shows the settings of the properties in the server.xml file required for ACL management with LDAP.

```
<jndiEntry jndiName="ibm.appcenter.ldap.active" value="true"/>
<jndiEntry jndiName="ibm.appcenter.ldap.connectionURL" value="ldap://employees.com:636"/>
<jndiEntry jndiName="ibm.appcenter.ldap.user.loginName" value="uid"/>
<jndiEntry jndiName="ibm.appcenter.ldap.user.base" value="dc=ibm,dc=com"/>
<jndiEntry jndiName="ibm.appcenter.ldap.group.base" value="dc=ibm,dc=com"/>
<jndiEntry jndiName="ibm.appcenter.ldap.user.displayName" value="sn"/>
<jndiEntry jndiName="ibm.appcenter.ldap.group.name" value="cn"/>
<jndiEntry jndiName="ibm.appcenter.ldap.group.uniquemember" value="uniqueMember"/>
<jndiEntry jndiName="ibm.appcenter.ldap.user.groupmembership" value="ibm-allGroups"/>
<jndiEntry jndiName="ibm.appcenter.ldap.cache.expiration.seconds" value="43200"/>
<jndiEntry jndiName="ibm.appcenter.ldap.security.sasl" value="'EXTERNAL'"/>
<jndiEntry jndiName="ibm.appcenter.ldap.referral" value="'follow'"/>
<jndiEntry jndiName="ibm.appcenter.ldap.user.filter" value="'(&amp;(uid=%v)(objectclass=inetOrgPerson))'"/>
<jndiEntry jndiName="ibm.appcenter.ldap.user.displayName.filter" value="'(&amp;(cn=%v)(objectclass=inetOrgPerson))'"/>
<jndiEntry jndiName="ibm.appcenter.ldap.group.filter" value="'(&amp;(cn=%v)(|(objectclass=groupOfNames)(objectclass=groupOfUniqueNames)))'"/>
```

LDAP with Apache Tomcat

Configure the Apache Tomcat application server for LDAP authentication and configure security (Java™ Platform, Enterprise Edition) in the web.xml file of the Application Center.

To configure ACL management of the Application Center, configure LDAP for user authentication, map the Java™ EE roles of the Application Center to the LDAP roles, and configure the Application Center properties for LDAP authentication. Only the simple type of LDAP authentication is supported.

- Configuration of LDAP authentication (Apache Tomcat)
- Configuring LDAP ACL management (Apache Tomcat)

Configuration of LDAP authentication (Apache Tomcat)

Define the users who can access the Application Center console and the users who can log in with the mobile client by mapping Java™ Platform, Enterprise Edition roles to LDAP roles.

To configure ACL management of the Application Center, follow this process:

- Configure LDAP for user authentication.
- Map the Java Platform, Enterprise Edition (Java EE) roles of the Application Center to the LDAP roles.
- Configure theApplication Center properties for LDAP authentication.

Restriction: Only the simple type of LDAP authentication is supported.

You configure the Apache Tomcat server for LDAP authentication and configure security (Java™ Platform, Enterprise Edition) in the web.xml file of theApplication Center Services web application (**applicationcenter.war**) and of the Application Center Console web application (**appcenterconsole.war**).

LDAP user authentication
You must configure a **JNDIRealm** in the **server.xml** file in the `<Host>` element. For more information about configuring a realm, see the

Realm Component on the Apache Tomcat website.

Example of configuration on Apache Tomcat to authenticate against an LDAP server
This example shows how to configure user authentication on an Apache Tomcat server by comparing with the authorization of these users on a server enabled for LDAP authentication.

```
<Host appBase="webapps" autoDeploy="true" name="localhost" unpackWARs="true">
 ...
<Realm className="org.apache.catalina.realm.JNDIRealm"
      connectionURL="ldap://bluepages.ibm.com:389"
      userSubtree="true"
      userBase="ou=bluepages,o=ibm.com"
      userSearch="(emailAddress={0})"
      roleBase="ou=ibmgroups,o=ibm.com"
      roleName="cn"
      roleSubtree="true"
      roleSearch="(uniqueMember={0})"
      allRolesMode="authOnly"
      commonRole="appcenter"/>
 ...
</Host>
```

The value of **connectionURL** is the LDAP URL of your LDAP server.

The **userSubtree**, **userBase**, and **userSearch** attributes define how to use the name that is given to the Application Center in login form (in the browser message box) to match an LDAP user entry.

In the example, the definition of **userSearch** specifies that the user name is used to match the email address of an LDAP user entry.

The basis or scope of the search is defined by the value of the **userBase** attribute. In LDAP, an information tree is defined; the user base indicates a node in that tree.

Set the value of **userSubtree** to true; if it is set to **false**, the search runs only on the direct child nodes of the user base. It is important that the search penetrates the subtree and does not stop at the first level.

For authentication, you define only the **userSubtree**, **userBase**, and **userSearch** attributes. The Application Center also uses Java EE security roles. Therefore, you must map LDAP attributes to some Java EE roles. These attributes are used for mapping LDAP attributes to security roles:

- **roleBase**
- **roleName**
- **roleSubtree**
- **roleSearch**

In this example, the value of the **roleSearch** attribute matches all LDAP entries with a **uniqueMember** attribute whose value is the **Distinguished Name (DN)** of the authenticated user.

- The **roleBase** attribute specifies a node in the LDAP tree below which the roles are defined.
- The **roleSubtree** attribute indicates whether the LDAP search should search the entire subtree, whose root is defined by the value of **roleBase**, or only the direct child nodes.
- The **roleName** attribute defines the name of the LDAP attribute.
- The **allRolesMode** attribute specifies that you can use the asterisk (*) character as the value of **role-name** in the **web.xml** file. This attribute is optional.
- The **commonRole** attribute adds a role that is shared by all authenticated users. This attribute is optional.

Mapping the Java EE roles of the Application Center to LDAP roles
After you define the LDAP request for the Java EE roles, you must change the **web.xml** file of the Application Center Services web application (**applicationcenter.war**) and of the Application Center Console web application (**appcenterconsole.war**) to map the Java EE roles of **appcenteradmin** and **appcenteruser** to the LDAP roles.

These examples, where LDAP users have LDAP roles, called **MyLdapAdmin** and **MyLdapUser**, show where and how to change the web.xml file. Replace the names **MyLdapAdmin** and **MyLdapUser** with the roles that are defined in your LDAP. Modify the following files:

- **tomcat_install_dir/webapps/appcenterconsole/WEB-INF/web.xml**
- **tomcat_install_dir/webapps/applicationcenter/WEB-INF/web.xml**

The security-role-ref element in the JAX_RS servlet

```
<servlet>
   <servlet-name>...</servlet-name>
   <servlet-class>...</servlet-class>
      <init-param>
         ...
      </init-param>
      <load-on-startup>1</load-on-startup>
      <security-role-ref>
         <role-name>appcenteradmin</role-name>
         <role-link>MyLdapAdmin</role-link>
      </security-role-ref>
      <security-role-ref>
         <role-name>appcenteruser</role-name>
         <role-link>MyLdapUser</role-link>
      </security-role-ref>
</servlet>
```

The security-role element

```
<security-role>
   <role-name>MyLdapAdmin</role-name>
</security-role>
<security-role>
   <role-name>MyLdapUser</role-name>
</security-role>
```

The auth-constraint element
After you edit the **security-role-ref** and the **security-role** elements, you can use the roles that are defined in the **auth-constraint** elements to protect the web resources. Edit these roles for the **appcenteradminConstraint** element in both the **web.xml** file of both **appcenterconsole** and **applicationcenter**, and for the **appcenteruserConstraint** element in the **appcenterconsole web.xml** file.

```
<security-constraint>
      <display-name>appcenteradminConstraint</display-name>
      <web-resource-collection>
         ...
      </web-resource-collection>
      <auth-constraint>
         <role-name>MyLdapAdmin</role-name>
      </auth-constraint>
      <user-data-constraint>
         ...
      </user-data-constraint>
</security-constraint>
```

and

```
<security-constraint>
      <display-name>appcenteruserConstraint</display-name>
      <web-resource-collection>
         ...
      </web-resource-collection>
      <auth-constraint>
         <role-name>MyLdapUser</role-name>
      </auth-constraint>
      <user-data-constraint>
         ...
      </user-data-constraint>
</security-constraint>
```

Configuring LDAP ACL management (Apache Tomcat)

Use LDAP to define the users and groups who can install mobile applications with the Application Center by defining the Application Center LDAP properties through JNDI.

To configure LDAP ACL management of the Application Center; add an entry for each property in the `<context>` section of the IBM Application Center Services application in the server.xml file. This entry should have the following syntax:

```
<Environment name="JNDI_property_name" value="property_value" type="java.lang.String" override="false"/>
```

Where:

- **JNDI_property_name** is the name of the property you are adding.
- **property_value** is the value of the property you are adding.

Property	Description
ibm.appcenter.ldap.active	Set to true to enable LDAP; set to false to disable LDAP.
ibm.appcenter.ldap.federated.active	Since WebSphere Application Server Liberty profile V8.5.5: set to true to enable use of the federated registry; set to false to disable use of the federated registry, which is the default setting.
ibm.appcenter.ldap.connectionURL	LDAP connection URL.
ibm.appcenter.ldap.user.base	Search base of users.
ibm.appcenter.ldap.user.loginName	LDAP login attribute.
ibm.appcenter.ldap.user.displayName	LDAP attribute for the user name to be displayed, for example, a person's full name.
ibm.appcenter.ldap.group.base	Search base of groups.
ibm.appcenter.ldap.group.name	LDAP attribute for the group name.
ibm.appcenter.ldap.group.uniquemember	LDAP attribute that identifies the members of a group.
ibm.appcenter.ldap.user.groupmembership	LDAP attribute that identifies the groups to which a user belongs.
ibm.appcenter.ldap.group.nesting	Management of nested groups: if nested groups are not managed, set the value to false.
ibm.appcenter.ldap.user.filter	LDAP user search filter for the attribute of user login name. Use %v as the placeholder for the login name attribute. This property is only required when LDAP users and groups are defined in the same subtree; that is, when the properties **ibm.appcenter.ldap.user.base** and **ibm.appcenter.ldap.group.base** have the same value.
ibm.appcenter.ldap.displayName.filter	LDAP user search filter for the attribute of user display name. Use %v as the placeholder for the display name attribute. This property is only required when LDAP users and groups are defined in the same subtree; that is, when the properties **ibm.appcenter.ldap.user.base** and **ibm.appcenter.ldap.group.base** have the same value.
ibm.appcenter.ldap.group.filter	LDAP group search filter. Use %v as the placeholder for the group attribute. This property is only required when LDAP users and groups are defined in the same subtree; that is, when the properties **ibm.appcenter.ldap.user.base** and **ibm.appcenter.ldap.group.base** have the same value.
ibm.appcenter.ldap.security.sasl	The value of the security authentication mechanism when the LDAP external SASL authentication mechanism is required to bind to the LDAP server. The value depends on the LDAP server; usually, it is set to "EXTERNAL".
ibm.appcenter.ldap.security.binddn	Property that identifies the distinguished name of the user permitted to search the LDAP directory. Use this property only if security binding is required.
ibm.appcenter.ldap.security.bindpwd	Property that identifies the password of the user who is allowed to search the LDAP directory. Use this property only if security binding is required. The password can be encoded with the "Liberty profile securityUtility" tool. Run the tool and then set the value of this property to the encoded password generated by the tool. The supported encoding types are xor and aes. Edit the Liberty profile server.xml file to check whether the classloader is enabled to load the JAR file that decodes the password.

Property	Description
ibm.appcenter.ldap.cache.expiration.seconds	elay in seconds before the LDAP cache expires. If no value is entered, the default value is 86400, which is equal to 24 hours. Changes to users and groups on the LDAP server become visible to the Application Center after a delay, which is specified by **ibm.appcenter.ldap.cache.expiration.seconds**. The Application Center maintains a cache of LDAP data and the changes only become visible after the cache expires. By default, the delay is 24 hours. If you do not want to wait for this delay to expire after changes to users or groups, you can call this command to clear the cache of LDAP data: `acdeploytool.sh -clearLdapCache -s serverurl -c context -u user -p password` See Using the stand-alone tool to clear the LDAP cache (../../../appcenter/command-line/#using-the-stand-alone-tool-to-clear-the-ldap-cache) for details.
ibm.appcenter.ldap.referral	Property that indicates whether referrals are supported by the JNDI API. If no value is given, the JNDI API will not handle LDAP referrals. Possible values are: • ignore: ignores referrals found in the LDAP server. • follow: automatically follows any referrals found in the LDAP server. • throw: causes an exception to occur for each referral found in the LDAP server.

See JNDI properties for Application Center for a complete list of LDAP properties that you can set.

The example shows properties defined in the **server.xml** file.

```
<Environment name="ibm.appcenter.ldap.active" value="true" type="java.lang.String" override="false"/>
<Environment name="ibm.appcenter.ldap.connectionURL" value="ldaps://employees.com:636" type="java.lang.String" override="false"/>
<Environment name="ibm.appcenter.ldap.user.base" value="dc=ibm,dc=com" type="java.lang.String" override="false"/>
<Environment name="ibm.appcenter.ldap.user.loginName" value="uid" type="java.lang.String" override="false"/>
<Environment name="ibm.appcenter.ldap.user.displayName" value="cn" type="java.lang.String" override="false"/>
<Environment name="ibm.appcenter.ldap.user.groupmembership" value="ibm-allGroups" type="java.lang.String" override="false"/>
<Environment name="ibm.appcenter.ldap.group.base" value="dc=ibm,dc=com" type="java.lang.String" override="false"/>
<Environment name="ibm.appcenter.ldap.group.name" value="cn" type="java.lang.String" override="false"/>
<Environment name="ibm.appcenter.ldap.group.uniquemember" value="uniquemember" type="java.lang.String" override="false"/>
<Environment name="ibm.appcenter.ldap.cache.expiration.seconds" value="43200" type="java.lang.String" override="false"/>
<Environment name="ibm.appcenter.ldap.security.sasl" value="EXTERNAL" type="java.lang.String" override="false"/>
<Environment name="ibm.appcenter.ldap.security.referral" value="follow" type="java.lang.String" override="false"/>
<Environment name="ibm.appcenter.ldap.user.filter" value="(&amp;(uid=%v)(objectclass=inetOrgPerson))" type="java.lang.String" override
="false"/>
<Environment name="ibm.appcenter.ldap.user.displayName.filter" value="(&amp;(cn=%v)(objectclass=inetOrgPerson))" type="java.lang.Stri
ng" override="false"/>
<Environment name="ibm.appcenter.ldap.group.filter" value="(&amp;(cn=%v)(|(objectclass=groupOfNames)(objectclass=groupOfUniqueNa
mes)))" type="java.lang.String" override="false"/>
```

Configuring properties of DB2 JDBC driver in WebSphere Application Server

Add some JDBC custom properties to avoid DB2 exceptions from a WebSphere Application Server that uses the IBM DB2 database.

When you use WebSphere Application Server with an IBM DB2 database, this exception could occur:

```
Invalid operation: result set is closed. ERRORCODE=-4470, SQLSTATE=null
```

To avoid such exceptions, you must add custom properties in WebSphere Application Server at the Application Center data source level.

1. Log in to the WebSphere Application Server administration console.
2. Select **Resources → JDBC → Data sources → Application Center DataSource name → Custom properties** and click **New**.
3. In the **Name** field, enter **allowNextOnExhaustedResultSet**.
4. In the **Value** field, type **1**.
5. Change the type to **java.lang.Integer**.
6. Click **OK**.
7. Click **New**.
8. In the **Name** field, enter **resultSetHoldability**.
9. In the **Value** field, type **1**.
10. Change the type to **java.lang.Integer**.
11. Click **OK** and save your changes.

Managing the DB2 transaction log size

When you upload an application that is at least 40 MB with IBM MobileFirst Foundation Application Center console, you might receive a transaction log full error.

The following system output is an example of the **transaction log full** error code.

```
DB2 SQL Error: SQLCODE=-964, SQLSTATE=57011
```

The content of each application is stored in the Application Center database.

The active log files are defined in number by the **LOGPRIMARY** and **LOGSECOND** database configuration parameters, and in size by the LOGFILSIZ database configuration parameter. A single transaction cannot use more log space than **LOGFILSZ * (LOGPRIMARY + LOGSECOND) * 4096 KB**.

The `DB2 GET DATABASE CONFIGURATION` command includes information about the log file size, and the number of primary and secondary log files.

Depending on the largest size of the MobileFirst application that is deployed, you might need to increase the DB2 log space.

Using the `DB2 update db cfg` command, increase the **LOGSECOND** parameter. Space is not allocated when the database is activated. Instead, the space is allocated only as needed.

Defining the endpoint of the application resources

When you add a mobile application from the Application Center console, the server-side component creates Uniform Resource Identifiers (URI) for the application resources (package and icons). The mobile client uses these URI to manage the applications on your device.

To manage the applications on your device, the Application Center console must be able to locate the Application Center REST services and to generate the required number of URI that enable the mobile client to find the Application Center REST services.

By default, the URI protocol, host name, and port are the same as those defined in the web application server used to access the Application Center console; the context root of the Application Center REST services is **applicationcenter**. When the context root of the Application Center REST services is changed or when the internal URI of the web application server is different from the external URI that can be used by the mobile client, the externally accessible endpoint (protocol, host name, and port) of the application resources must be defined by configuring the web application server. (Reasons for separating internal and external URI could be, for example, a firewall or a secured reverse proxy that uses HTTP redirection.)

The following figure shows a configuration with a secured reverse proxy that hides the internal address (192.168...). The mobile client must use the external address (**appcntr.net**).

Endpoint properties

Property name	Purpose	Example
ibm.appcenter.services.endpoint	This property enables the Application Center console to locate the Application Center REST services. The value of this property must be specified as the external address and context root of the applicationcenter.war web application. You can use the asterisk (*) character as wildcard to specify that the Application Center REST services use the same value as the Application Center console. For example: ///:*/appcenter means use the same protocol, host, and port as the Application Center console, but use appcenter as context root. This property must be specified for the Application Center console application.	https://appcntr.net:443/applicationcenter
ibm.appcenter.proxy.protocol	This property specifies the protocol required for external applications to connect to the Application Center.	https
ibm.appcenter.proxy.host	This property specifies the host name required for external applications to connect to the Application Center.	appcntr.net
ibm.appcenter.proxy.port	This property specifies the port required for external applications to connect to the Application Center.	443

Jump to

- Configuring the endpoint of application resources (full profile)
- Configuring the endpoint of the application resources (Liberty profile)
- Configuring the endpoint of the application resources (Apache Tomcat)

Configuring the endpoint of application resources (full profile)

For the WebSphere Application Server full profile, configure the endpoint of the application resources in the environment entries of the Application Center services and the Application Center console applications. The procedure differs depending on whether you deployed WAR files or an EAR file.

If you deployed WAR files

Follow this procedure when you must change the URI protocol, host name, and port used by the mobile client to manage the applications on your device. Since IBM Worklight V6.0, you use JNDI environment entries.

For a complete list of JNDI properties, see JNDI properties for Application Center.

1. Log in to the WebSphere Application Server console.
2. Select **Applications → Application Types → WebSphere enterprise applications**.
3. Click **IBM Application Center Services**.
4. In the **Web Module Properties** section, select **Environment entries for Web modules**.
5. Assign the appropriate values for the following environment entries:
    - For **ibm.appcenter.proxy.host**, assign the host name.
    - For **ibm.appcenter.proxy.port**, assign the port number.
    - For **ibm.appcenter.proxy.protocol**, assign the external protocol.
    - Click **OK** and save the configuration.
6. Select **Applications → Application Types → WebSphere enterprise applications**.
7. Click **IBM Application Center Console**.
8. In the **Web Module Properties** section, select **Environment entries for Web modules**.
9. For **ibm.appcenter.services.endpoint**, assign the full URI of the Application Center REST services (the URI of the **applicationcenter.war** file).
    - In a scenario with a firewall or a secured reverse proxy, this URI must be the external URI and not the internal URI inside the local LAN.
    - You can use the asterisk (*) character as wildcard to specify that the Application Center REST services use the same value as the Application Center console.

    For example: `*://*:*/appcenter` means use the same protocol, host, and port as the Application Center console, but use appcenter as the context root.
10. Click **OK** and save the configuration.

If you deployed an EAR file

1. Log in to the WebSphere Application Server console.
2. Select **Applications → Application Types → WebSphere enterprise applications**.
3. Click **AppCenterEAR**.
4. In the **Web Module Properties** section, select **Environment entries for Web modules**.
5. Assign the appropriate values for the following environment entries:
    - For **ibm.appcenter.proxy.host**, assign the host name.
    - For **ibm.appcenter.proxy.port**, assign the port number.
    - For **ibm.appcenter.proxy.protocol**, assign the external protocol.
6. For **ibm.appcenter.services.endpoint**, assign the full URI of the Application Center REST services (the URI of the **applicationcenter.war** file).
    - In a scenario with a firewall or a secured reverse proxy, this URI must be the external URI and not the internal URI inside the local LAN.
    - You can use the asterisk (*) character as wildcard to specify that the Application Center REST services use the same value as the Application Center console.

    For example: `*://*:*/appcenter` means use the same protocol, host, and port as the Application Center console, but use appcenter as the context root.
7. Click OK and save the configuration.

Configuring the endpoint of the application resources (Liberty profile)

For the Liberty profile, configure the endpoint of the application resources through the JNDI environment.

Since IBM Worklight V6.0, follow this procedure when you must change the URI protocol, host name, and port used by the Application Center client to manage the applications on your device.

Edit the **server.xml** file. To be able to define JNDI entries, the `<feature>` element must be defined correctly in the **server.xml** file:

```
<feature>jndi-1.0</feature>
```

Add an entry for each property in the `<server>` section of the **server.xml** file. This entry should have the following syntax:

```
<jndiEntry jndiName="JNDI_property_name" value="property_value"/>
```

Where:

- **JNDI_property_name** is the name of the property that you are adding.
- **property_value** is the value of the property that you are adding.

Property	Description
ibm.appcenter.services.endpoint	The URI of the Application Center REST services. In a scenario with a firewall or a secured reverse proxy, this URI must be the external URI and not the internal URI inside the local LAN.
ibm.appcenter.proxy.protocol The protocol of the application resources URI. This property is optional. It is only needed if the protocol of the external and of the internal URI are different.	
ibm.appcenter.proxy.host	The host name of the application resources URI.
ibm.appcenter.proxy.port	The port of the application resources URI. This property is optional. It is only needed if the protocol of the external and of the internal URI are different.

For a complete list of LAPD properties that you can set, see JNDI properties for Application Center.

Example of setting properties for configuring the endpoint

This example shows the settings of the properties in the **server.xml** file required for configuring the endpoint of the application resources.

```
<jndiEntry jndiName="ibm.appcenter.services.endpoint" value=" https://appcntr.net:443/applicationcenter" />
<jndiEntry jndiName="ibm.appcenter.proxy.protocol" value="https" />
<jndiEntry jndiName="ibm.appcenter.proxy.host" value="appcntr.net" />
<jndiEntry jndiName="ibm.appcenter.proxy.port"  value=" 443"/>
```

You can use the asterisk (*) character as wildcard to specify that the Application Center REST services use the same value as the Application Center console. For example: `*://*:*/appcenter` means use the same protocol, host, and port as the Application Center console, but use **appcenter** as context root.

Configuring the endpoint of the application resources (Apache Tomcat)

For the Apache Tomcat server, configure the endpoint of the application resources in the **server.xml** file.

Since IBM Worklight V6.0, follow this procedure when you must change the URI protocol, host name, and port used by the Application Center client to manage the applications on your device.

Edit the **server.xml** file in the conf directory of your Apache Tomcat installation.
Add an entry for each property in the `<context>` section of the corresponding application. This entry should have the following syntax:

```
<Environment name="JNDI_property_name" value="property_value" type="property_type" override="false"/>
```

Where:

- **JNDI_property_name** is the name of the property you are adding.
- **property_value** is the value of the property you are adding.
- **property_type** is the type of the property you are adding.

Property	Type	Description
ibm.appcenter.services.endpoint	java.lang.String	The URI of the Application Center REST services (applicationcenter.war). In a scenario with a firewall or a secured reverse proxy, this URI must be the external URI and not the internal URI inside the local LAN.

Property	Type	Description
ibm.appcenter.proxy.protocol	java.lang.String	The protocol of the application resources URI. This property is optional. It is only needed if the protocol of the external and of the internal URI are different.
ibm.appcenter.proxy.host	java.lang.String	The host name of the application resources URI.
ibm.appcenter.proxy.port	java.lang.Integer	The port of the application resources URI. This property is optional. It is only needed if the protocol of the external and of the internal URI are different.

For a complete list of JNDI properties that you can set, see JNDI properties for Application Center.

Example of setting server.xml properties for configuring the endpoint

This example shows the settings of the properties in the **server.xml** file required for configuring the endpoint of the application resources.

In the `<context>` section of the Application Center console application:

```
<Environment name="ibm.appcenter.services.endpoint" value="https://appcntr.net:443/applicationcenter" type="java.lang.String" override="false"/>
```

You can use the asterisk (*) character as wildcard to specify that the Application Center REST services use the same value as the Application Center console. For example: `*://*:*/appcenter` means use the same protocol, host, and port as the Application Center console, but use appcenter as context root.

In the `<context>` section of the Application Center services application:

```
<Environment name="ibm.appcenter.services.endpoint" value="https://appcntr.net:443/applicationcenter" type="java.lang.String" override="false"/>
<Environment name="ibm.appcenter.proxy.protocol" value="https" type="java.lang.String" override="false"/>
<Environment name="ibm.appcenter.proxy.host" value="appcntr.net" type="java.lang.String" override="false"/>
<Environment name="ibm.appcenter.proxy.port"  value="443" type="java.lang.Integer" override="false"/>
```

Configuring Secure Sockets Layer (SSL)

Learn about configuring SSL for the Application Center on supported application servers and the limitations of certificate verification on mobile operating systems.

You can configure the Application Center with SSL or without SSL, **unless** you intend to install applications on iOS devices. For iOS applications, you must configure the Application Center server with SSL.

SSL transmits data over the network in a secured channel. You must purchase an official SSL certificate from an SSL certificate authority. The SSL certificate must be compatible with Android and iOS. Self-signed certificates do not work with the Application Center.

When the client accesses the server through SSL, the client verifies the server through the SSL certificate. If the server address matches the address that is filed in the SSL certificate, the client accepts the connection. For the verification to be successful, the client must know the root certificate of the certificate authority. Many root certificates are preinstalled on Android and iOS devices. The exact list of pre-installed root certificates varies between versions of mobile operating systems.

For information about the mobile operating system versions that support its certificates, consult the SSL certificate authority.

If the SSL certificate verification fails, a normal web browser requests confirmation to contact an untrusted site. The same behavior occurs when you use a self-signed certificate that was not purchased from a certificate authority. When mobile applications are installed, this control is not performed by a normal web browser, but by operating system calls.

Some versions of Android, iOS, and Windows Phone operating systems do not support this confirmation dialog in system calls. This limitation is a reason to avoid self-signed certificates or SSL certificates that are not suited to mobile operating systems. On Android, iOS, and Windows Phone operating systems, you can install a self-signed CA certificate on the device to enable the device to handle system calls regarding this self-signed certificate. This practice is not appropriate for Application Center in a production environment, but it can be suitable during the testing period. For details, see Managing and installing self-signed CA certificates in an Application Center test environment below.

Jump to

Configuring SSL for WebSphere Application Server full profile

Request a Secure Sockets Layer (SSL) certificate and process the received documents to import them into the keystore.
This procedure indicates how to request an SSL certificate and import it and the chain certificate into your keystore.

1. Create a request to a certificate authority; in the WebSphere administrative console, select **Security → SSL certificate and key management → Key stores and certificates → keystore → Personal certificate requests → New**, where **keystore** identifies your keystore.

   The request is sent to the certificate authority.
2. When you receive the SSL certificate, import it and the corresponding chain certificate into your keystore by following the instructions provided by the certificate authority. In the WebSphere administrative console, you can find the corresponding option in **Security → SSL certificate and key management → Manage endpoint security configurations → node SSL settings → Key stores and certificates → keystore → Personal certificates → certificate → Receive a certificate from a certificate authority**.

   Where:
   - **node SSL settings** shows the SSL settings of the nodes in your configuration.
   - **keystore** identifies your keystore.
   - **certificate** identifies the certificate that you received.
3. Create an SSL configuration. See the instructions in the user documentation that corresponds to the version of the WebSphere Application Server full profile that supports your applications.

You can find configuration details in the WebSphere administrative console at **Security → SSL certificate and key management → Manage endpoint security configurations → SSL Configurations**.

Configuring SSL for Liberty profile

Create a keystore, import the Secure Socket Layer (SSL) certificate, and edit the server.xml file to configure SSL on Liberty profile. Follow the steps in this procedure to configure SSL on Liberty profile.

1. Create a keystore for your web server; use the **securityUtility** with the **createSSLCertificate** option. See Enabling SSL communication for the Liberty profile (http://www-01.ibm.com/support/knowledgecenter/SSAW57_8.5.5/com.ibm.websphere.wlp.nd.doc/ae/twlp_sec_ssl.html?cp=SSAW57_8.5.5%2F1-3-11-0-4-1-0) for more information.
2. Import the SSL certificate and the corresponding chain certificate into your keystore by following the instructions provided by the certificate authority.
3. Enable the ssl-1.0 Liberty feature in the **server.xml** file.

   ```
   <featureManager>
     <feature>ssl-1.0</feature>
   </featureManager>
   ```
4. Add the keystore service object entry to the server.xml file. The **keyStore** element is called **defaultKeyStore** and contains the keystore password. For example:

   ```
   <keyStore id="defaultKeyStore" location="/path/to/myKeyStore.p12"
       password="myPassword" type="PKCS12"/>
   ```
5. Make sure that the value of the **httpEndpoint** element in the **server.xml** file defines the httpsPort attribute. For example:

   ```
   <httpEndpoint id="defaultHttpEndpoint" host="*" httpPort="9080" httpsPort="9443" >
   ```
6. Restart the web server. Now you can access the web server by `https://myserver:9443/...`

Configuring SSL for Apache Tomcat

Create a keystore, import the Secure Socket Layer (SSL) certificate, and edit the **conf/server.xml** file to define a connector for SSL on Apache Tomcat. Follow the steps in this procedure to configure SSL on Apache Tomcat. See SSL Configuration HOW-TO (http://tomcat.apache.org/tomcat-7.0-doc/ssl-howto.html) for more details and examples of configuring SSL for Apache Tomcat.

1. Create a keystore for your web server. You can use the Java™ **keytool** command to create a keystore.

   ```
   keytool -genkey -alias tomcat -keyalg RSA -keystore /path/to/keystore.jks
   ```
2. Import the SSL certificate and the corresponding chain certificate into your keystore by following the instructions provided by the certificate authority.
3. Edit the **conf/server.xml** file to define a connector to use SSL. This connector must point to your keystore.

   ```
   <Connector port="8443" protocol="HTTP/1.1" SSLEnabled="true"
           maxThreads="150" scheme="https" secure="true"
           clientAuth="false" sslProtocol="TLS"
           keystoreFile="/path/to/keystore.jks"
           keystorePass="mypassword" />
   ```

4. Restart the web server. Now you can access the web server by `https://myserver:8443/...`

Managing and installing self-signed CA certificates in an Application Center test environment

Use self-signed certificate authority (CA) certificates in test environments to install applications with Application Center on a mobile device from a secured server.

Uploading or deleting a certificate
When you install the Application Center mobile client from OTA (the bootstrap page), the device user must upload and install the self-signed CA file before the Application Center mobile client is installed.

When you use Application Center for a test installation, the administrator might not have a real Secure Sockets Layer (SSL) certificate available. You might want to use a self-signed CA certificate. Such certificates work if they get installed on the device as root certificate. As an administrator, you can easily distribute self-signed CA certificates to devices.

The following procedure focuses mostly on the iOS and Android environments. Support for X.509 certificates comes from the individual mobile platforms, not from IBM MobileFirst Foundation. For more information about specific requirements for X.509 certificates, see the documentation of each mobile platform.

Managing self-signed certificates: in your role of administrator of Application Center, you can access the list of registered self-signed CA certificates to upload or delete certificates.

1. To display Application Center settings, click the gear icon Icon to access Application Center settings.
2. To display the list of registered certificates, select **Self Signed Certificates**.
3. Upload or delete a certificate.
    - To upload a self-signed CA certificate, in the Application Center console, click **Upload a certificate** and select a certificate file.

    Note: The certificate file must be in PEM file format. Typical file name suffixes for this type of file are **.pem, .key, .cer, .cert**. The certificate must be a self-signed one, that is, the values of the **Issuer** and **Subject** fields must be the same. And the certificate must be a CA certificate, that is, it must have the X509 extension named **BasicConstraint** set to **CA:TRUE**.

- To delete a certificate, click the trash can icon on the right of the certificate file name in the list.

Installing a self-signed CA certificate on a device
Registered self-signed CA certificates are available through the bootstrap page at `http://hostname:portnumber/appcenterconsole/installers.html`.

Where:

- **hostname** is the name of the server that hosts the Application Center console.
- **portnumber** is the corresponding port number.

1. Click the **SSL Certificates** tab.
2. To display the details of a certificate, select the appropriate registered certificate.
3. To download and install the certificate on the device, click **Install**.

JNDI properties for Application Center

You can configure some JNDI properties for Application Center.

Property	Description
appcenter.database.type	The database type, which is required only when the database is not specified in appcenter.jndi.name.
appcenter.jndi.name	The JNDI name of the database. This parameter is the normal mechanism to specify the database. The default value is java:comp/env/jdbc/AppCenterDS.
appcenter.openjpa.ConnectionDriverName	The fully qualified class name of the database connection driver class. This property is needed only when the database is not specified in appcenter.jndi.name.
appcenter.openjpa.ConnectionPassword	The password for the database connection. Set this property only when the database is not specified in appcenter.jndi.name.
appcenter.openjpa.ConnectionURL	The URL for the database connection driver class. Set this property only when the database is not specified in appcenter.jndi.name.
appcenter.openjpa.ConnectionUserName	The user name or the database connection. Set this property only when the database is not specified in appcenter.jndi.name.

Property	Description
ibm.appcenter.apns.p12.certificate.isDevelopmentCertificate	Set this property to true to specify whether the certificate that enables Application Center to send push notifications about updates of iOS applications is a development certificate. Set the property to **false** if it is not a development certificate. See Configuring the Application Center server for connection to Apple Push Notification Services (../../../appcenter/push-notifications/#configuring-the-application-center-server-for-connection-to-apple-push-notification-services).
ibm.appcenter.apns.p12.certificate.location	The path to the file of the development certificate that enables Application Center to send push notifications about updates of iOS applications. For example, **/Users/someUser/someDirectory/apache-tomcat/conf/AppCenter*apns*dev_cert.p12**. See Configuring the Application Center server for connection to Apple Push Notification Services (../../../appcenter/push-notifications/#configuring-the-application-center-server-for-connection-to-apple-push-notification-services).
ibm.appcenter.apns.p12.certificate.password	The password of the certificate that enables Application Center to send push notifications about updates of iOS applications is a development certificate. See Configuring the Application Center server for connection to Apple Push Notification Services (../../../appcenter/push-notifications/#configuring-the-application-center-server-for-connection-to-apple-push-notification-services).
ibm.appcenter.forceUpgradeDBTo60	The database design was changed starting from IBM Worklight version 6.0. The database is automatically updated when the Application Center web application starts. If you want to repeat this update, you can set this parameter to **true** and start the web application again. Later you can reset this parameter to **false**.
ibm.appcenter.gcm.signature.googleapikey	The Google API key that enables Application Center to send push notifications about updates for Android applications. For example, AIxaScCHg0VSGdgfOZKtzDJ44-oi0muUasMZvAs. See Configuring the Application Center server for connection to Google Cloud Messaging (../../../appcenter/push-notifications/#configuring-the-application-center-server-for-connection-to-google-cloud-messaging).
ibm.appcenter.ios.plist.onetimeur	Specifies whether URLs stored in iOS plist manifests use the one-time URL mechanism without credentials. If you set this property to true, the security level is medium, because one-time URLs are generated with a cryptographic mechanism so that nobody can guess the URL but do not require the user to log in. Setting this property to false provides maximal security, because the user is then required to log in for each URL. However, requesting the user to log in multiple times when you install an iOS application can degrade the user experience. See Installing the client on an iOS mobile device (../../../appcenter/mobile-client/#installing-the-client-on-an-ios-mobile-device).
ibm.appcenter.ldap.active	Specifies whether Application Center is configured for LDAP. Set this property to true to enable LDAP or to false to disable LDAP. See Managing users with LDAP.
ibm.appcenter.ldap.cache.expiration.seconds	The Application Center maintains a cache of LDAP data and the changes become visible only after the cache expires. Specify the number of seconds during which an entry in the LDAP cache is valid. Set this property to a value greater than 3600 (1 hour) to reduce the amount of LDAP requests. If no value is entered, the default value is 86400, which is equal to 24 hours. If you need to clear the cache of LDAP data manually, enter this command: `acdeploytool.sh -clearLdapCache -s serverurl -c context -u user -p password`. See Using the stand-alone tool to clear the LDAP cache (../../../appcenter/command-line/#using-the-stand-alone-tool-to-clear-the-ldap-cache).
ibm.appcenter.ldap.connectionURL	The URL to access the LDAP server when no Virtual Member Manager (VMM) is used. See Configuring LDAP ACL management (Liberty profile) and Configuring LDAP ACL management (Apache Tomcat).

Property	Description
ibm.appcenter.ldap.federated.active	Specifies whether Application Center is configured for LDAP with federated repositories. Since WebSphere Application Server Liberty profile V8.5.5., set this property to true to enable use of the federated registry. Set this property to false to disable use of the federated registry, which is the default setting. See Managing users with LDAP.
ibm.appcenter.ldap.group.base	The search base to find groups when you use LDAP without Virtual Member Manager (VMM). See Configuring LDAP ACL management (Liberty profile) and Configuring LDAP ACL management (Apache Tomcat).
ibm.appcenter.ldap.group.filter	LDAP group search filter. Use **%v** as the placeholder for the group attribute. This property is only required when LDAP users and groups are defined in the same subtree; that is, when the properties **ibm.appcenter.ldap.user.base** and **ibm.appcenter.ldap.group.base** have the same value.
ibm.appcenter.ldap.group.name	The group name attribute when you use LDAP without Virtual Member Manager (VMM). See Configuring LDAP ACL management (Liberty profile) and Configuring LDAP ACL management (Apache Tomcat).
ibm.appcenter.ldap.group.nesting	Specifies whether the LDAP contains nested groups (that is, groups in groups) when you use LDAP without Virtual Member Manager (VMM). Setting this property to false speeds up LDAP access because the groups are not searched recursively. See Configuring LDAP ACL management (Liberty profile) and Configuring LDAP ACL management (Apache Tomcat).
ibm.appcenter.ldap.group.uniquemember	Specifies the members of a group when you use LDAP without Virtual Member Manager (VMM). This property is the inverse of ibm.appcenter.ldap.user.groupmembership. See Configuring LDAP ACL management (Liberty profile) and Configuring LDAP ACL management (Apache Tomcat).
ibm.appcenter.ldap.referral	Specifies whether referrals are supported by the JNDI API. If no value is specified, the JNDI API does not handle LDAP referrals. Here are the possible values: • ignore: Ignores referrals that are found in the LDAP server. • follow: Automatically follows any referrals that are found in the LDAP server. • throw: Causes an exception to occur for each referral found in the LDAP server.
ibm.appcenter.ldap.security.binddn	The distinguished name of the user that is allowed to search the LDAP directory. Use this property only if security binding is required.
ibm.appcenter.ldap.security.bindpwd	The password of the user that is permitted to search the LDAP directory. Use this property only if security binding is required. The password can be encoded with the Liberty profile securityUtility tool. Run the tool and then set the value of this property to the encoded password that is generated by the tool. Edit the Liberty profile server.xml file to check whether the classloader is enabled to load the JAR file that decodes the password. See Configuring LDAP ACL management (Apache Tomcat).
ibm.appcenter.ldap.security.sasl	Specifies the security authentication mechanism when the LDAP external SASL authentication mechanism is required to bind to the LDAP server. The value depends on the LDAP server and it is typically set to EXTERNAL. When this property is set, security authentication is required to connect to LDAP without Virtual Member Manager (VMM). See Configuring LDAP ACL management (Liberty profile) and Configuring LDAP ACL management (Apache Tomcat).

Property	Description
ibm.appcenter.ldap.user.base	The search base to find users when you use LDAP without Virtual Member Manager (VMM). See Configuring LDAP ACL management (Liberty profile) and Configuring LDAP ACL management (Apache Tomcat).
ibm.appcenter.ldap.user.displayName	The display name attribute, such as the user's real name, when you use LDAP without Virtual Member Manager (VMM). See Configuring LDAP ACL management (Liberty profile) and Configuring LDAP ACL management (Apache Tomcat).
ibm.appcenter.ldap.displayName.filter	LDAP user search filter for the attribute of ibm.appcenter.ldap.user.displayName. Use %v as the placeholder for the display name attribute. This property is required only when LDAP users and groups are defined in the same subtree; that is, when the properties ibm.appcenter.ldap.user.base and ibm.appcenter.ldap.group.base have the same value.
ibm.appcenter.ldap.user.filter	LDAP user search filter for the attribute of ibm.appcenter.ldap.user.loginName. Use **%v** as the placeholder for the login name attribute. This property is required only when LDAP users and groups are defined in the same subtree; that is, when the properties ibm.appcenter.ldap.user.base and ibm.appcenter.ldap.group.base have the same value.
ibm.appcenter.ldap.user.groupmembership	Specifies the groups of a member when you use LDAP without Virtual Member Manager (VMM). This property is the inverse of ibm.appcenter.ldap.group.uniquemember. This property is optional, but if it is specified, LDAP access is faster. See Configuring LDAP ACL management (Liberty profile) and Configuring LDAP ACL management (Apache Tomcat).
ibm.appcenter.ldap.user.loginName	The login name attribute when you use LDAP without Virtual Member Manager (VMM). See Configuring LDAP ACL management (Liberty profile) and Configuring LDAP ACL management (Apache Tomcat).
ibm.appcenter.ldap.vmm.active	Set this property to true to specify that LDAP is done through Virtual Member Manager (VMM), or to false otherwise. See Configuring LDAP ACL management for WebSphere Application Server V8.x.
ibm.appcenter.ldap.vmm.adminpwd	The password when LDAP is done through Virtual Member Manager (VMM). See Configuring LDAP ACL management for WebSphere Application Server V8.x.
ibm.appcenter.ldap.vmm.adminuser	The user when LDAP is done through Virtual Member Manager (VMM). See Configuring LDAP ACL management for WebSphere Application Server V8.x.
ibm.appcenter.logging.formatjson	This property has an effect only when ibm.appcenter.logging.tosystemerror is set to true. If this property is enabled, it formats JSON responses in logging messages that are directed to System.Error. Setting this property is helpful when you debug the server.
ibm.appcenter.logging.tosystemerror	Specifies whether all logging messages are also directed to System.Error. Setting this property is helpful when you debug the server.
ibm.appcenter.openjpa.Log	This property is passed to OpenJPA and enables JPA logging. For details, see the Apache OpenJPA User's Guide (http://openjpa.apache.org/builds/1.2.2/apache-openjpa-1.2.2/docs/manual/manual.html).

Property	Description
ibm.appcenter.proxy.host	If the Application Center server is behind a firewall or reverse proxy, this property specifies the address of the host. Setting this property allows a user outside the firewall to reach the Application Center server. Typically, this property is the address of the proxy. See Defining the endpoint of the application resources.
ibm.appcenter.proxy.port	If the Application Center server is behind a firewall or reverse proxy, this property specifies the address of the host. Setting this property allows a user outside the firewall to reach the Application Center server. Typically, this property is the port of the proxy, for example 443. It is needed only if the protocol of the external URI and the protocol of the internal URI are different. See Defining the endpoint of the application resources.
ibm.appcenter.proxy.protocol	If the Application Center server is behind a firewall or reverse proxy, this property specifies the protocol (http or https). Setting this property allows a user outside the firewall to reach the Application Center server. Typically, this property is set to the protocol of the proxy. For example, appcntr.net. This property is needed only if the protocol of the external and of the internal URI are different. See Defining the endpoint of the application resources.
ibm.appcenter.proxy.scheme	This property is just an alternative name for ibm.appcenter.proxy.protocol.
ibm.appcenter.push.schedule.period.amount	Specifies the time schedule when you send push notifications of application updates. When applications are frequently changed on the server, set this property to send batches of notifications. For example, send all notifications that happened within the past hour, instead of sending each individual notification.
ibm.appcenter.push.schedule.period.unit	Specifies the unit for the time schedule when you send push notifications of application updates.
ibm.appcenter.services.endpoint	Enables the Application Center console to locate the Application Center REST services. Specify the external address and context root of the applicationcenter.war web application. In a scenario with a firewall or a secured reverse proxy, this URI must be the external URI and not the internal URI inside the local LAN. For example, https://appcntr.net:443/applicationcenter. See Defining the endpoint of the application resources.
ibm.appcenter.services.iconCacheMaxAge	Specifies the number of seconds during which cached icons remain valid for the Application Center console and the client. Application icons rarely change, therefore they are cached. Specify values larger than 600 (10 min) to reduce the amount of data transfer for the icons.
mfp.jndi.configuration	Optional. If the JNDI configuration is injected into the WAR files or provided as a shared library, the value of this property is the name of the JNDI configuration. You can also specify this value as a system property.
mfp.jndi.file	Optional. If the JNDI configuration is stored as an external file, the value of this property is the path of a file that describes the JNDI configuration. You can also specify this value as a system property.

Configuring WebSphere Application Server to support applications in public app stores

Configure WebSphere Application Server full profile and Liberty profile before access to public app stores through application links, because of the use of SSL connections.

The constraint imposed by the use of SSL connections requires the root certificates of public app stores to exist in the WebSphere truststore before you can use application links to access these public stores. The configuration requirement applies to both WebSphere Application Server full profile and Liberty profile.

The root certificate of Google play must be imported into the WebSphere truststore before you can use application links to Google play. The root certificate of Apple iTunes must be imported into the WebSphere truststore before you can use application links to iTunes.

Jump to

- Configuring WebSphere Application Server to support applications in Google play
- Configuring WebSphere Application Server to support applications in Apple iTunes

- Configuring Liberty profile when IBM JDK is used

Configuring WebSphere Application Server to support applications in Google play

Configure WebSphere Application Server to enable links in the Application Center console to access applications in Google play.

Follow this procedure to import the root certificate of Google play into the WebSphere truststore. You must import this certificate before the Application Center can support links to applications stored in Google Play.

1. Log in to the WebSphere Application Server console and navigate to **Security → SSL certificate and key management → Key stores and certificates → NodeDefaultTrustStore → Signer certificates**.
2. Click **Retrieve from port**.
3. In the **Host** field, enter **play.google.com**.
4. In the **Port** field, enter **443**.
5. In the **Alias** field, enter **play.google.com**.
6. Click **Retrieve signer information**.
7. Click **OK** and save the configuration.

Configuring WebSphere Application Server to support applications in Apple iTunes

Configure WebSphere Application Server to enable links in the Application Center console to access applications in Apple iTunes.

Follow this procedure to import the root certificate of Apple iTunes into the WebSphere truststore. You must import this certificate before the Application Center can support links to applications stored in iTunes.

1. Log in to the WebSphere Application Server console and navigate to **Security → SSL certificate and key management → Key stores and certificates → NodeDefaultTrustStore → Signer certificates**.
2. Click **Retrieve from port**.
3. In the **Host** field, enter **itunes.apple.com**.
4. In the **Port** field, enter 443.
5. In the **Alias** field, enter **itunes.apple.com**.
6. Click **Retrieve signer information**.
7. Click **OK** and save the configuration.

Configuring Liberty profile when IBM JDK is used

Configure Liberty profile to use default JSSE socket factories instead of SSL socket factories of WebSphere Application Server when IBM JDK is used.

The purpose is to configure the IBM JDK SSL factories to be compatible with Liberty profile. This configuration is required only when IBM JDK is used. The configuration does not apply for use of Oracle JDK. By default, IBM JDK uses the SSL socket factories of WebSphere Application Server. These factories are not supported by Liberty profile.

Exception when WebSphere Application Server SSL socket factories are used

If you use the IBM JDK of WebSphere Application Server, this exception could occur because this JDK uses SSL socket factories that are not supported by the Liberty profile. In this case, follow the requirements documented in Troubleshooting tips (http://www.ibm.com/support/knowledgecenter/SSD28V_8.5.5/com.ibm.websphere.wlp.core.doc/ae/rwlp_trouble.html?view=kc).

```
java.net.SocketException: java.lang.ClassNotFoundException: Cannot find the specified class com.ibm.websphere.ssl.protocol.SSLSocketFa
ctory
    at javax.net.ssl.DefaultSSLSocketFactory.a(SSLSocketFactory.java:11)
    at javax.net.ssl.DefaultSSLSocketFactory.createSocket(SSLSocketFactory.java:6)
    at com.ibm.net.ssl.www2.protocol.https.c.afterConnect(c.java:161)
    at com.ibm.net.ssl.www2.protocol.https.d.connect(d.java:36)
    at sun.net.www.protocol.http.HttpURLConnection.getInputStream(HttpURLConnection.java:1184)
    at java.net.HttpURLConnection.getResponseCode(HttpURLConnection.java:390)
    at com.ibm.net.ssl.www2.protocol.https.b.getResponseCode(b.java:75)
    at com.ibm.ws.jmx.connector.client.rest.internal.RESTMBeanServerConnection.loadJMXServerInfo(RESTMBeanServerConnection.java:1
42)
    at com.ibm.ws.jmx.connector.client.rest.internal.RESTMBeanServerConnection.<init>(RESTMBeanServerConnection.java:114)
    at com.ibm.ws.jmx.connector.client.rest.internal.Connector.connect(Connector.java:315)
    at com.ibm.ws.jmx.connector.client.rest.internal.Connector.connect(Connector.java:103)
```

Last modified on