Installing Liferay DXP on Wildfly

Liferay DXP 7.0 bundled with Wildfly is available on the Help Center (DXP) or the Community Downloads page (Portal CE). The Wildfly bundle contains JARs, scripts, and configuration files required for deploying Liferay DXP 7.0. Copying these files from the Liferay DXP Wildfly bundle facilitates installing Liferay DXP on an existing Wildfly application server.

Whether you copy bundle files (recommended) or download and create the files, you must download these *Additional Files* for DXP or Portal CE:

- Liferay DXP WAR file
- Dependencies ZIP file
- OSGi Dependencies ZIP file

Installing Liferay DXP manually requires these basic steps:

- Installing Liferay DXP dependencies to your application server
- Configuring your application server for Liferay DXP
- Installing the Liferay DXP WAR file to your application server

Liferay Home is one folder above Wildfly's install location. *Liferay Home* refers to the folder containing your Wildfly server folder. When Liferay DXP is installed on Wildfly, the Liferay Home folder contains the Wildfly server folder as well as `data`, `deploy`, `logs`, and `osgi` folders. You'll also see the term `$WILDFLY_HOME` used in this guide. `$WILDFLY_HOME` refers to your Wildfly server folder. This folder is usually named `wildfly-[version]`.

Installing Liferay DXP Dependencies

Liferay DXP depends on many JARs that are included in the Liferay DXP Wildfly bundle. Some JARs in the bundle are not strictly required but can still be useful. If you don't have a Liferay DXP Wildfly bundle, you can download the required JARs from third-parties, as described below.

2. Download your database driver `.jar` file and copy it into the same folder. For example, for MySQL, download the MySQL Connector/J driver and put its `.jar` file into the `$WILDFLY_HOME/modules/com/liferay/portal/main` folder.

3. Download the remaining required JAR and insert it into the same folder.

    - `com.liferay.registry.api.jar`

    Be sure to remove the version number from the JAR file names or update their names where they're defined (you'll see where the `com.liferay.registry.api.jar` is defined next).

4. Create the file `module.xml` in the `$WILDFLY_HOME/modules/com/liferay/portal/main` folder and insert the following contents:

```
<?xml version="1.0"?>

<module xmlns="urn:jboss:module:1.0" name="com.liferay.portal">
    <resources>
        <resource-root path="com.liferay.registry.api-[version].jar" />
        <resource-root path="mysql-connector-java-[version]-bin.jar" />
        <resource-root path="portal-kernel.jar" />
        <resource-root path="portlet.jar" />
    </resources>
    <dependencies>
        <module name="javax.api" />
        <module name="javax.mail.api" />
        <module name="javax.servlet.api" />
        <module name="javax.servlet.jsp.api" />
        <module name="javax.transaction.api" />
    </dependencies>
</module>
```

    Make sure to replace `[version]` with the correct version of the MySQL JDBC driver. If you are using a different database, replace the MySQL `.jar` with the driver JAR for your database (e.g., HSQL, PostgreSQL, etc.).

5. Create an `osgi` folder in your Liferay Home folder. Then extract the OSGi ZIP file that you downloaded into the `osgi` folder.

Checkpoint:

1. At this point, you should have the following files in the `$WILDFLY_HOME/modules/com/liferay/portal/main` folder:

- `com.liferay.registry.api.jar`
- `portal-kernel.jar`
- `portlet.jar`
- a database `jar` such as the MySQL Connector.

2. The `module.xml` has listed all jars in the `<resource-root-path>` elements.

3. The `osgi` folder has the following subfolders:

- `configs`
- `core`
- `marketplace`
- `target-platform`
- `test`

Great! You have your `.jar` files ready.

Running Liferay DXP on Wildfly 10.0 in Standalone Mode vs. Domain Mode

Wildfly 10.0 can be launched in either *standalone* mode or *domain* mode. Domain mode allows multiple application server instances to be managed from a single control point. A collection of such application servers is known as a *domain*. For more information on standalone mode vs. domain mode, please refer to the section on this topic in the Wildfly 10 Admin Guide. Liferay DXP fully supports Wildfly 10.0 when it runs in standalone mode but not when it runs in domain mode.

You can run Liferay DXP on Wildfly 10.0 in domain mode, but this method is not fully supported. In particular, Liferay DXP's hot-deploy does not work, since Wildfly 10.0 cannot deploy non-exploded `.war` files in domain mode. Instead, `.war` files are in the `domain/data/content` directory. Deployments are only possible using the command line interface. This prevents many Liferay DXP plugins from working as intended. For example, JSP hooks don't work on Wildfly 10.0 running in domain mode, since Liferay DXP's JSP override mechanism relies on the application server reloading customized JSP files from the exploded plugin `.war` file location. Other plugins, such as service or action hooks, should still work properly since they don't require Wildfly to access anything (such as JSP files) from an exploded `.war` file on the file system.

Please refer to the chapter of this guide on Configuring Liferay DXP for High Availability for information on setting up a Liferay DXP cluster.

Configuring Wildfly

Now you'll make some adjustments in your configuration to support using Liferay DXP.

You can specify the Wildfly server instance's configuration in the XML file `$WILDFLY_HOME/standalone/configuration/standalone.xml`. You must also make some modifications to your configuration and startup scripts found in the `$WILDFLY_HOME/bin/` folder. Lastly, you'll need to make some modifications in your `$WILDFLY_HOME/modules/`. You'll begin with making changes to `standalone.xml`.

Make the following modifications to `standalone.xml`:

1. In the `<jsp-config>` tag, set the Java VM compatibility for Liferay source and class files. They are compatible with Java 8 by default.

```
<jsp-config development="true" source-vm="1.8" target-vm="1.8" />
```

2. Locate the closing `</extensions>` tag. Directly beneath that tag, insert the following system properties:

```
<system-properties>
    <property name="org.apache.catalina.connector.URI_ENCODING" value="UTF-8" />
    <property name="org.apache.catalina.connector.USE_BODY_ENCODING_FOR_QUERY_STI
</system-properties>
```

3. Add a timeout for the deployment scanner by setting `deployment-timeout="360"` as seen in the excerpt below.

```
<subsystem xmlns="urn:jboss:domain:deployment-scanner:2.0">
    <deployment-scanner deployment-timeout="360" path="deployments" relative-to='
</subsystem>
```

4. Add the following JAAS security domain to the security subsystem `<security-domains>` defined in element `<subsystem xmlns="urn:jboss:domain:security:1.2">`.

```
            <login-module code="com.liferay.portal.kernel.security.jaas.PortalLoginM
        </authentication>
    </security-domain>
```

5. Remove the following tags (if necessary):

    - `<location name="/" handler="welcome-content"/>`
    - `<extension module="org.jboss.as.weld"/>`
    - `<subsystem xmlns="urn:jboss:domain:weld:2.0"/>`
    - `<subsystem xmlns="urn:jboss:domain:weld:3.0"/>`
6. Find the `<jsp-config/>` tag and insert the `development="true"` attribute into the tag. Once finished, the tag should look like the following:

```
<jsp-config development="true" />
```

Checkpoint:

Before continuing, verify the following properties have been set in the `standalone.xml` file:

1. A new `<system-property>` has been created.
2. The `<deployment-timeout>` has been set to `360` .
3. A new `<security-domain>` has been created.
4. Four tags have been removed.
5. `<jsp-config development>` has been set to `true` .

Now it's time for some changes to your configuration and startup scripts.

Make the following modifications to your standalone domain's configuration script file `standalone.conf` (`standalone.conf.bat` on Windows) found in your `$WILDFLY_HOME/bin/` folder.

These modifications change the following options:

- Set the file encoding
- Set the user time-zone

Make the following edits as applicable to your operating system:

On Windows, comment out the initial `JAVA_OPTS` assignment as demonstrated in the following line:

```
rem set "JAVA_OPTS=-Xms64M -Xmx512M -XX:MetaspaceSize=96M -XX:MaxMetaspaceSize=256
```

Then add the following `JAVA_OPTS` assignment one line above the `:JAVA_OPTS_SET` line found at end of the file:

```
set "JAVA_OPTS=%JAVA_OPTS% -Dfile.encoding=UTF-8 -Djava.net.preferIPv4Stack=true -
```

On Unix, merge the following values into your settings for `JAVA_OPTS` , replacing any matching attributes with the ones found in the assignment below:

```
JAVA_OPTS="$JAVA_OPTS -Dfile.encoding=UTF-8 -Djava.net.preferIPv4Stack=true -Dsecm
```

Important: For Liferay DXP to work properly, the application server JVM must use the `GMT` time zone and `UTF-8` file encoding.

Make sure you replace the `$WILDFLY_HOME` references with the appropriate directory. You'll notice some Java security options. You'll finish configuring the Java security options in the *Security Configuration* section.

Note: If you plan on using the IBM JDK with your Wildfly server, you'll need to complete some additional steps. First, navigate to the `$WILDFLY_HOME/modules/com/liferay/portal/main/module.xml` file and insert the following dependency within the `<dependencies>` element:

```
<module name="ibm.jdk" />
```

Then navigate to the `$WILDFLY_HOME/modules/system/layers/base/sun/jdk/main/module.xml` file and insert the following path names inside the `<paths>...</paths>` element:

```
<path name="com/sun/crypto" />
<path name="com/sun/crypto/provider" />
<path name="com/sun/image/codec/jpeg" />
```

The added paths resolve issues with portal deployment exceptions and image uploading problems on a Liferay DXP instance running on Wildfly 10.0.x.
Checkpoint:

At this point, you'll have finished configuring the application server's JVM settings.

1. The file encoding, user time-zone, preferred protocol stack have been set in the `JAVA_OPTS` in the `standalone.conf.bat` file.

2. The default amount of memory available has been increased.

3. If using IBM's JDK, the `sun crypto` properties have been set in the `$WILDFLY_HOME/modules/system/layers/base/sun/jdk/main/module.xml` file.

The prescribed script modifications are now complete for your Liferay DXP installation on Wildfly. Next you'll configure mail and the database.

Database Configuration

If you want Wildfly to manage your data source, follow the instructions in this section. If you want to use the built-in Liferay DXP data source, you can skip this section.

Modify `standalone.xml` and add your data source and driver in the `<datasources>` element of your data sources subsystem.

1. First, add your data source inside the `<datasources>` element.

```
<datasource jndi-name="java:jboss/datasources/ExampleDS" pool-name="ExampleDS" e
    <connection-url>jdbc:mysql://localhost/lportal</connection-url>
    <driver>mysql</driver>
    <security>
        <user-name>root</user-name>
        <password>root</password>
    </security>
</datasource>
```

Note: If you'd like to change your datasource `jndi-name` to something different, you'll need to also edit the `datasource` element in the `<default-bindings>` tag.

2. Add your driver to the `<drivers>` element also found within the `<datasources>` element.

```
<drivers>
    <driver name="mysql" module="com.liferay.portal"/>
</drivers>
```

Your final data sources subsystem should look like this:

```
<subsystem xmlns="urn:jboss:domain:datasources:1.0">
    <datasources>
        <datasource jndi-name="java:jboss/datasources/ExampleDS" pool-name="Exampl
            <connection-url>jdbc:mysql://localhost/lportal</connection-url>
            <driver>mysql</driver>
            <security>
                <user-name>root</user-name>
                <password>root</password>
            </security>
        </datasource>
        <drivers>
            <driver name="mysql" module="com.liferay.portal"/>
        </drivers>
    </datasources>
</subsystem>
```

Now that you've configured your data source, the mail session is next.

Mail Configuration

If you want Wildfly to manage your mail session, use the following instructions. If you want to use the built-in Liferay DXP mail session, you can skip this section.

Specify your mail subsystem in `standalone.xml` as in the following example:

```
        <smtp-server ssl="true" outbound-socket-binding-ref="mail-smtp" username="USER
    </mail-session>
</subsystem>
...
<socket-binding-group name="standard-sockets" default-interface="public" port-offset="
...
<outbound-socket-binding name="mail-smtp">
        <remote-destination host="smtp.gmail.com" port="465"/>
    </outbound-socket-binding>
</socket-binding-group>
```

You've got mail! Next, you'll make sure Liferay DXP can connect using your new mail session and database.

Configuring data sources and mail sessions

Now that your data source and mail session are set up, you need to ensure Liferay DXP can access them.

1. First, navigate to the Liferay Home folder, which is one folder above Wildfly's install location (i.e. `$WILDFLY_HOME/..`).

2. If you're using *Wildfly* to manage your data source, add the following configuration to your `portal-ext.properties` file in your *Liferay Home* to refer to your data source:

   ```
   jdbc.default.jndi.name=java:jboss/datasources/ExampleDS
   ```

   If you're using *Liferay DXP* to manage your data source, follow the instructions for using the setup wizard.

3. If you're using *Liferay DXP* to manage your mail session, this configuration is done in Liferay DXP. That is, after starting your portal as described in the *Deploy Liferay DXP* section, go to *Control Panel* → *Server Administration* → *Mail* and enter the settings for your mail session.

   If you're using Wildfly to manage your mail session, add the following configuration to your `portal-ext.properties` file to reference that mail session:

   ```
   mail.session.jndi.name=java:jboss/mail/MailSession
   ```

Security Configuration

When you're ready to begin using other people's apps from Marketplace, you'll want to protect your Liferay DXP instance and your Wildfly server from security threats. To do so, you can enable Java Security on your Wildfly server and specify a security policy to grant your Liferay DXP instance access to your server.

Remember, you set the `-Dsecmgr` and `-Djava.security.policy` Java options in the `standalone.conf.bat` file earlier in the *Configuring Wildfly* section. The `-Dsecmgr` Java option enables security on Wildfly. Likewise, the `-Djava.security.policy` Java option lists the permissions for your server's Java security policy. If you have not set these options, you'll need to do so before using Java security.

This configuration opens up all permissions. You can tune the permissions in your policy later. Create the `$WILDFLY_HOME/bin/server.policy` file and add the following contents:

```
grant {
    permission java.security.AllPermission;
};
```

For extensive information on Java SE Security Architecture, see its specification documents at http://docs.oracle.com/javase/7/docs/technotes/guides/security/spec/security-spec.doc.html. Also, see the Plugin Security and PACL tutorial to learn how to configure Liferay DXP plugin access to resources.

Deploy Liferay DXP

1. If the folder `$WILDFLY_HOME/standalone/deployments/ROOT.war` already exists in your Wildfly installation, delete all of its subfolders and files. Otherwise, create a new folder `$WILDFLY_HOME/standalone/deployments/ROOT.war`.

2. Unzip the Liferay DXP `.war` file into the `ROOT.war` folder.

3. To trigger deployment of `ROOT.war`, create an empty file named `ROOT.war.dodeploy` in your `$WILDFLY_HOME/standalone/deployments/` folder. On startup, Wildfly detects the presence of this file and deploys it as a web application.

4. Start the Wildfly application server by navigating to `$WILDFLY_HOME/bin` and running `standalone.bat` or `standalone.sh`.

« Installing Liferay DXP on Tomcat 8 Installing Liferay DXP on JBoss EAP 6.4 »

¿Fue útil este artículo?

Sí No

Usuarios a los que les pareció útil: 0 de 0

¿Necesitas más ayuda?

Abre un ticket

Contacta con nosotros

Pregunta en la Comunidad

Liferay University
Forma a tu equipo en los puntos esenciales del uso de Liferay con videos detallados y ejercicios prácticos diseñados para simular una formación presencial

Liferay Global Services
Aumenta tu conocimiento sobre el producto y el desarrollo gracias a expertos que te ayudarán a maximizar tu ROI

© 2023 LIFERAY INC. ALL RIGHTS RESERVED