Adding JARs to the Geronimo repository

{scrollbar}

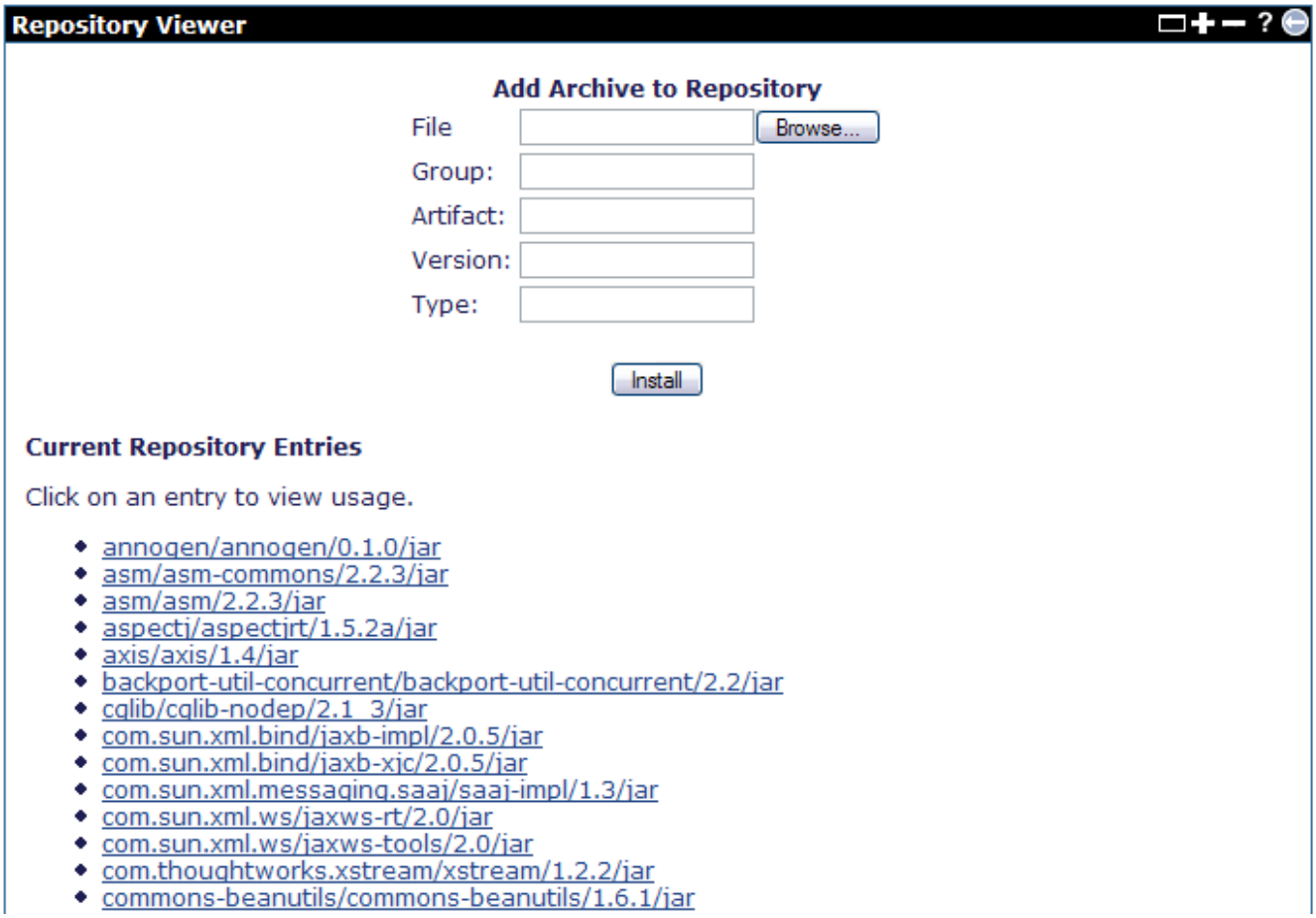
INLINE

The server keeps common Java libraries in a repository using the same directory structure and naming conventions prompted by the Apache Maven project. If your Java EE asset depends on a library that is not already in the repository, you can update the repository to include the new library from console or using deployer command, and define the dependency in your Java EE asset's deployment plan.

In this section, we'll show you how to do it via the Admin Console.

Update the repository with new library

For adding archives or jars to the repository the **Repository Viewer** portlet is available by selecting **Server > Repository** on the **Console Navigation** menu on the left hand side. The **Repository Viewer** portlet illustrated in the following figure displays the artifacts installed in the server's repository. The layout of the repository is the same as that used by Apache Maven making it possible to easily copy files over.



Here are the meaning for each field on **Repository Viewer** portlet:

- *File* is actual location of the archive.
- *group* is a group identifier, typically, the name of the open source project (for example **commons-logging** or **log4j**), or the directory tree matching the Java package prefix (for example **org/apache/derby**) of the organization that supplies the library.
- *Artifact* is the file name prefix of the library.
- *Version* is the version identifier of the library in the file, allowing the repository to hold multiple versions of the same library without ambiguity
- *Type* is the type of file, typically, **jar**.

Define the dependency in the deployment plan

In order to avoid problems with trying to load the same native library more than once per jvm lifetime, you need to put code that loads native libraries (such as many db drivers) into a classloader plugin that can be shared among all the datasource plugins that need the classes.

To use an artifact in an application, let's take the first one from the list as an example, you will need to add a **<dependency>** element under **<dependencies>** in the **<environment>** element in the application's deployment plan. Here is an excerpt of how a deployment plan would look like:

xmlsolid <environment> ... <dependencies> ... <dependency> <groupId>annogen</groupId> <artifactId>annogen</artifactId> <version>0.1.0</version> <type>jar</type> </dependency> </dependencies> </environment>

Besides using the console to add an archive into Geronimo repository, you can do the same thing using command `deploy`.