Introduction to Repositories

Artifact Repositories

A repository in Maven holds build artifacts and dependencies of varying types.

There are exactly two types of repositories: **local** and **remote**:

1. the **local** repository is a directory on the computer where Maven runs. It caches remote downloads and contains temporary build artifacts that you have not yet released.
2. **remote** repositories refer to any other type of repository, accessed by a variety of protocols such as `file://` and `https://`. These repositories might be a truly remote repository set up by a third party to provide their artifacts for downloading (for example, repo.maven.apache.org (https://repo.maven.apache.org/maven2/)). Other "remote" repositories may be internal repositories set up on a file or HTTP server within your company, used to share private artifacts between development teams and for releases.

Local and remote repositories are structured the same way so that scripts can run on either side, or they can be synced for offline use. The layout of the repositories is completely transparent to the Maven user, however.

Using Repositories

In general, you should not need to do anything with the local repository on a regular basis, except clean it out if you are short on disk space (or erase it completely if you are willing to download everything again).

For the remote repositories, they are used for both downloading and uploading (if you have the permission to do so).

Downloading from a Remote Repository

Downloading in Maven is triggered by a project declaring a dependency that is not present in the local repository (or for a `SNAPSHOT`, when the remote repository contains one that is newer). By default, Maven will download from the central (https://repo.maven.apache.org/maven2/) repository.

To override this, you need to specify a `mirror` as shown in Using Mirrors for Repositories (../mini/guide-mirror-settings.html).

You can set this in your `settings.xml` file to globally use a certain mirror. However, it is common for a project to customise the repository in its `pom.xml` (../mini/guide-multiple-repositories.html) and that your setting will take precedence. If dependencies are not being found, check that you have not overridden the remote repository.

For more information on dependencies, see Dependency Mechanism (./introduction-to-dependency-mechanism.html).

Using Mirrors for the Central Repository

There are several official Central repositories (/repository/) geographically distributed. You can make changes to your `settings.xml` file to use one or more mirrors. Instructions for this can be found in the guide Using Mirrors for Repositories (../mini/guide-mirror-settings.html).

Building Offline

If you are temporarily disconnected from the internet and you need to build your projects offline, you can use the offline switch on the CLI:

```
mvn -o package
```

Many plugins honor the offline setting and do not perform any operations that connect to the internet. Some examples are resolving Javadoc links and link checking the site.

Uploading to a Remote Repository

While this is possible for any type of remote repository, you must have the permission to do so. To have someone upload to the Central Maven repository, see Repository Center (../../repository/index.html).

Internal Repositories

When using Maven, particularly in a corporate environment, connecting to the internet to download dependencies is not acceptable for security, speed or bandwidth reasons. For that reason, it is desirable to set up an internal repository to house a copy of artifacts, and to publish private artifacts to.

Such an internal repository can be downloaded using HTTP or the file system (with a `file://` URL), and uploaded to using SCP, FTP, or a file copy.

As far as Maven is concerned, there is nothing special about this repository: it is another **remote repository** that contains artifacts to download to a user's local cache, and is a publish destination for artifact releases.

Additionally, you may want to share the repository server with your generated project sites. For more information on creating and deploying sites, see Creating a Site (../mini/guide-site.html).

Setting up the Internal Repository

To set up an internal repository just requires that you have a place to put it, and then copy required artifacts there using the same layout as in a remote repository such as repo.maven.apache.org (https://repo.maven.apache.org /maven2/).

It is *not* recommended that you scrape or `rsync://` a full copy of central as there is a large amount of data there and doing so will get you banned. You can use a program such as those described on the Repository Management (../../repository-management.html) page to run your internal repository's server, download from the internet as required, and then hold the artifacts in your internal repository for faster downloading later.

The other options available are to manually download and vet releases, then copy them to the internal repository, or to have Maven download them for a user, and manually upload the vetted artifacts to the internal repository which is used for releases. This step is the only one available for artifacts where the license forbids their distribution automatically, such as several J2EE JARs provided by Sun. Refer to the Guide to coping with SUN JARs (../mini/guide-coping-with-sun-jars.html) document for more information.

It should be noted that Maven intends to include enhanced support for such features in the future, including click through licenses on downloading, and verification of signatures.

Using the Internal Repository

Using the internal repository is quite simple. Simply make a change to add a `repositories` element:

```
<project>
  ...
  <repositories>
    <repository>
      <id>my-internal-site</id>
      <url>https://myserver/repo</url>
    </repository>
  </repositories>
  ...
</project>
```

If your internal repository requires authentication, the `id` element can be used in your settings (../../settings.html#Servers) file to specify login information.

Deploying to the Internal Repository

One of the most important reasons to have one or more internal repositories is to be able to publish your own private releases.

To publish to the repository, you will need to have access via one of SCP, SFTP, FTP, WebDAV, or the filesystem. Connectivity is accomplished with the various wagons (/wagon/wagon-providers/index.html). Some wagons may need to be added as extension (/ref/current/maven-model/maven.html#class_extension) to your build.