Administration Center - Settings - Maven Repositories

Settings

When **adding a new version** to analyze an Application that includes **Maven based source code**, you must tell the Console where to find the **Maven local** or **remote HTTP/S repository.** The location of the repository is crucial to ensure that any associated JAR files can be automatically discovered and that POM dependencies can also be located. See **Configuring source code delivery for Maven** for more information.

To tell Console where the repositories are located you can use this Maven panel:

- Out of the box **https://repo.maven.apache.org/maven2/** will be predefined.
- You can **remove** a repository if you do not use it (use the **trash** icon).
- You can add **additional repositories** by clicking the **ADD** button.
- You can change the order of the repositories by dragging and dropping them in to the correct position (this is the order in which the repositories are scanned for artifacts).
- **Repositories** can also be defined at individual **Application level -** see **Application - Config - Maven configuration settings**.
- Repositories discovered in the uploaded source code and any repositories defined at Application level **always take priority** over any repositories defined in this options panel.

⚠ Any repository added in this panel is **valid for all Nodes** (and therefore all applications) managed in Console. If you configure multiple large repositories in this panel, the source code delivery action can take a significant amount of time to process through all the repositories when only some may be relevant for a specific Application. To counter this, you can ALSO defined repositories at Application level (see **Application - Config - Maven configuration settings**) - these are specific to the Application and won't be valid for any other Application/Node.

Troubleshooting issues accessing remote HTTPS repositories

In certain situations, an error may be registered in the Delivery log when the AIP Node attempts to access an HTTPS repository. For example, in the log located at `delivery\{app-guid}\data\{guid}\{guid}\{guid}\DMTDeliveryReport.CastLog2:`

```
ERROR cast.dmt.engine.extractor.jee.maven.http.connectionFailed Unkown format id: cast.dmt.engine.extractor.jee.
maven.http.connectionFailed =>  %URL%="https://my.maven.repo/artifactory/maven-release/"
%MESSAGE%="sun.security.validator.ValidatorException: PKIX path building failed: sun.security.provider.certpath.
SunCertPathBuilderException: unable to find valid certification path to requested target"
javax.net.ssl.SSLHandshakeException:sun.security.validator.ValidatorException: PKIX path building failed: sun.
security.provider.certpath.SunCertPathBuilderException: unable to find valid certification path to requested
target at sun.security.ssl.Alerts.getSSLException(:0).
```

The error reported in the log is generated by the **DeliveryManagerTool-CLI.exe** tool located on the **AIP Node**. This tool uses the Java JRE delivered with AIP Core in the following location:

```
%PROGRAMFILES%\CAST\<release>\jre\
```

This error usually occurs if the remote HTTPS repository you have defined is using an SSL certificate:

- where the signing authority is not listed in the Java JRE **cacerts** file located at `%PROGRAMFILES%\CAST\<release>\jre\security\cacerts`
- that is **self-signed**

Resolving the issue involves importing the required SSL certificates into the Java JRE delivered with AIP Core - this is out of the scope of this document.