Publishing Maven Artifacts With Ant And Ivy

Publishing Maven Artifacts to a Nexus Repository Using Ant and Ivy

This page currently is a draft for the Ant specific parts of the ASF Nexus Guide

Prepare your build

Usually your normal build process will already create the artifacts you want to publish (typically jars) but you may need to PGP-sign them the same way you sign your normal distribution artifacts. The artifacts are expected to follow the naming scheme *artifactId-version*.jar.

In addition you will need a minimal POM for your jar. If you are already using Ivy, you can use the `makepom` task to create one from your `ivy.xml`, otherwise see the Apache Maven project's documentation for "minimal" and the Apache Compress Antlib's POM for an example. If you are publishing multiple jars you may consider adding a parent POM to encapsulate the common information; see the Maven documentation, an example might be Ant's parent POM used for the several jars that make up an Ant release.

Users that use your project's jars from the Maven repository rather than using your "normal" distributions will likely want additional artifacts containing the source files or javadocs matching your jars in files named *artifactId*~~*version*-sources.jar and *artifactId*~~*version*-javadoc.jar respectively. Don't forget to sign those jars as well if you provide them.

Create Minimal Ivy Files for your Project

If you are not already using Ivy in your project you'll need to create a minimal `ivy.xml` file for your project. The syntax is described in Ivy's documentation. Since you will only use it to publish your artifacts, all you need to provide are the *organization*, *module* and *revision* definitions and an entry for each artifact you want to publish; see Ant's ivy.xml for a minimal version.

organization and *module* combined must match your Maven *groupId*.

You will need `artifact` elements for your jar as well as the POM and any PGP signature file. You don't need artifacts for your checksum files (if you create any) since Nexus will create MD5 and SHA1 checksums on the fly anyway.

If you are publishing source or javadoc jars as well, you'll need to provide something similar to Maven's classifier. You do so by adding an extra attribute to each artifact element that lives outside of Ivy's XML namespace and referencing this element in your `ivysettings.xml` file (see below). An example which uses this approach is the Compress Antlib's ivy.xml. It contains additional information and -sources as well as -javadoc artifacts.

Alternatively you can specify the -sources and -javadoc artifacts inside your `publish` task rather than your `ivy.xml` file. If you use Ivy 2.2.0 or later you can also configure it to PGP-sign you artifacts and no longer need to specify your signatures as artifacts. Ivy's own ivy-settings.xml configures Ivy to sign artifacts and the publish task inside the upload-nexus target declares the POM as well as -sources and -javadoc jars as additional artifacts.

Configure Ivy to Use Nexus

If you are already using Ivy you may need to adapt your `resolvers` configuration by adding an `url` resolver for Nexus and referencing that in a `module` matching your `ivy.xml`.

In general you'll mostly need to adapt the ivysettings.xml file used by Ant by using the same values for *organization* and *name* on the `module` element that you used in your `ivy.xml` file (where *name* on the `module` element in `ivysettings.xml` corresponds to *module* in `ivy.xml`).

Ant's `ivysettings.xml` uses Ant properties for the authentication information and Nexus' URL which will be expanded by the `ivy:configure` task. It also shows how to use the *classifier* extra attribute.

Uploading the Artifacts

Uploading involves three Ivy tasks.

1. `ivy:configure` uses your `ivysettings.xml` file to configure Ivy (what else?).
2. `ivy:resolve` reads your `ivy.xml` and doesn't do much beyond that if you are only using Ivy to upload your artifacts.
3. Finally `ivy:publish` publishes the artifcats to Nexus.

An example build file combining those steps that expects you to provide the authentication information via the command line (i.e. `ant -Dupload.user=YOUR-ASF_ID -Dupload.password=YOUR-PASSWORD`) can be found here: http://svn.apache.org/repos/asf/ant/antlibs/common/trunk/upload.xml

Continuing with the Release Process

Once the artifacts have been uploaded to Nexus, there isn't anything specific to Ant or Maven that needs to be done. You should now follow the guidelines provided in http://www.apache.org/dev/publishing-maven-artifacts.html#common