Releasing Apache Directory LDAP API

Since we are using Nexus for releases the release process is as follows (see also http://www.apache.org/dev/publishing-maven-artifacts.html#staging-maven).

1. Test the Project

```
$ mvn release:prepare -DdryRun=true
```

Be aware that this phase will ask you about the next version, and most important, for the next SCM tag :

```
...
[INFO] Checking dependencies and plugins for snapshots ...
What is the release version for "Apache Directory LDAP API"? (org.apache.directory.api:api-parent) 1.0.0-M16: :
What is the release version for "Apache Directory LDAP API I18n"? (org.apache.directory.api:api-i18n) 1.0.0-
M16: :
What is the release version for "Apache Directory LDAP API Utilities"? (org.apache.directory.api:api-util)
1.0.0-M16: :
...
What is SCM release tag or label for "Apache Directory LDAP API"? (org.apache.directory.api:api-parent) 1.0.0-
M16: :
...
```

2. Deploy a Snapshot

```
$ mvn deploy
```

This is useful to verify your settings in ~/.m2/settings.xml (Nexus password and GPG key)

3. Prepare the Release

```
$ mvn release:clean
$ mvn release:prepare
```

This creates a tag here: http://svn.apache.org/viewvc/directory/shared/tags/

4. Stage the Release

```
$ mvn release:perform
```

This deploys the release to a staging repository.

Go to https://repository.apache.org/index.html#stagingRepositories and close the staging repository.

5. Build the Site

```
$ cd target/checkout
$ mvn site
```

This creates the site.

Now, you have to sign the binary packages which are in target/checkout/distribution/target.

Use your PGP key ID (the pub key, 4096R/[XXXXXXX] where [XXXXXXX] is the key ID)

You can get the keys by typing :

1. gpg --list-keys

6. Publish Source and Binary Distribution Packages

First of all, create a new directory on people.apache.org/public_html to store the pacckages :

```
$ ssh people.apache.org
$ mkdir public_html/ldap-api-<version>
$ exit
```

Then copy the packages :

```
$ cd distributions/target
$ scp apache-ldap-api-<version>-* people.apache.org:public_html/ldap-api-<version>/
```

Update your index.html file on people.apach.org/public_html to make the packages visible. Here is an example of possible content :

```
<h2>Last Directory LDAP API 1.0.0-M16 tarballs</h2>
  <a href="ldap-api-1.0.0-M16/apache-ldap-api-1.0.0-M16-bin.tar.gz">apache-ldap-api-1.0.0-M16-bin.tar.gz</a><br
/>
  <a href="ldap-api-1.0.0-M16/apache-ldap-api-1.0.0-M16-bin.tar.gz.asc">apache-ldap-api-1.0.0-M16-bin.tar.gz.
asc</a><br/>
  <a href="ldap-api-1.0.0-M16/apache-ldap-api-1.0.0-M16-bin.tar.gz.md5">apache-ldap-api-1.0.0-M16-bin.tar.gz.
md5</a><br/>
  <a href="ldap-api-1.0.0-M16/apache-ldap-api-1.0.0-M16-bin.tar.gz.sha1">apache-ldap-api-1.0.0-M16-bin.tar.gz.
sha1</a><br/>
  <a href="ldap-api-1.0.0-M16/apache-ldap-api-1.0.0-M16-bin.zip">apache-ldap-api-1.0.0-M16-bin.zip</a><br/>
  <a href="ldap-api-1.0.0-M16/apache-ldap-api-1.0.0-M16-bin.zip.asc">apache-ldap-api-1.0.0-M16-bin.zip.asc<
/a><br/>
  <a href="ldap-api-1.0.0-M16/apache-ldap-api-1.0.0-M16-bin.zip.md5">apache-ldap-api-1.0.0-M16-bin.zip.md5<
/a><br/>
  <a href="ldap-api-1.0.0-M16/apache-ldap-api-1.0.0-M16-bin.zip.sha1">apache-ldap-api-1.0.0-M16-bin.zip.sha1<
/a><br/>
  <a href="ldap-api-1.0.0-M16/apache-ldap-api-1.0.0-M16-src.tar.gz">apache-ldap-api-1.0.0-M16-src.tar.gz</a><br
/>
  <a href="ldap-api-1.0.0-M16/apache-ldap-api-1.0.0-M16-src.tar.gz.asc">apache-ldap-api-1.0.0-M16-src.tar.gz.
asc</a><br/>
  <a href="ldap-api-1.0.0-M16/apache-ldap-api-1.0.0-M16-src.tar.gz.md5">apache-ldap-api-1.0.0-M16-src.tar.gz.
md5</a><br/>
  <a href="ldap-api-1.0.0-M16/apache-ldap-api-1.0.0-M16-src.tar.gz.sha1">apache-ldap-api-1.0.0-M16-src.tar.gz.
sha1</a><br/>
  <a href="ldap-api-1.0.0-M16/apache-ldap-api-1.0.0-M16-src.zip">apache-ldap-api-1.0.0-M16-src.zip</a><br/>
  <a href="ldap-api-1.0.0-M16/apache-ldap-api-1.0.0-M16-src.zip.asc">apache-ldap-api-1.0.0-M16-src.zip.asc<
/a><br/>
  <a href="ldap-api-1.0.0-M16/apache-ldap-api-1.0.0-M16-src.zip.md5">apache-ldap-api-1.0.0-M16-src.zip.md5<
/a><br/>
  <a href="ldap-api-1.0.0-M16/apache-ldap-api-1.0.0-M16-src.zip.sha1">apache-ldap-api-1.0.0-M16-src.zip.sha1<
/a><br/>
  <a href="ldap-api-1.0.0-M16/apache-ldap-api-1.0.0-M16.pom">apache-ldap-api-1.0.0-M16.pom</a><br/>
  <a href="ldap-api-1.0.0-M16/apache-ldap-api-1.0.0-M16.pom.asc">apache-ldap-api-1.0.0-M16.pom.asc</a><br/>
```

7. Test the New Version in ApacheDS and Studio

In apacheds/pom.xml change the <*org.apache.directory.shared.version*> property, build ApacheDS, go into apacheds/service, and run *./apachds.sh* to start the server.

In studio/pom.xml change the <*org.apache.directory.shared.version*> and <*org.apache.directory.shared.validversion*> properties, build Studio, and start Studio in applications/applications_<your platform>/target/ApacheDirectoryStudio-<your platform>/<executable>. Connect to the started ApacheDS.

8. Vote

Start a 72h vote at the dev mailing list.

9. Release

If the vote succeeds LDAP API project can be released.

Go to https://repository.apache.org/index.html#stagingRepositories and release the staging repository so all artifacts are published to Maven central.

Move the distribution packages (sources and bnaries) to the dist SVN repository: https://dist.apache.org/repos/dist/release/directory/api/dist/$(version)

The best solution would be to checkout the directory in people.apache.org, to copy the packages in the right place, and to check in the changes :

```
$ ssh people.apache.org
# svn co https://dist.apache.org/repos/dist/release/directory/api/dist/ api-dist
# cd api-dist
# mkdir <version>
# cp ../public_html/ldap-api-<version>/* <version>
# svn ci <version>
...
# exit
$
```

The packages should now be available on http://www.us.apache.org/dist/directory/api/dist/<version>

10. Deploy the Javadocs and XRef

We now can deploy the generated Javadoc and cross-reference pages. They are generated in the following directory :

```
target/checkout/target/site
```

We will copy two directories :

- apidocs
- xref

> ⚠ **Staging or Production?**
>
> Those files will be stored on the production server only !!! And some extra caution ust be taken not to delete them when we will publish the staging site too...

First of all, you must checkout the two CMS store for the site : staging and revision.

```
$ cd ~/apacheds
$ svn co https://svn.apache.org/repos/infra/websites/production/directory/trunk staging
 ...
$ svn co https://svn.apache.org/repos/infra/websites/production/directory production
 ...
```

Now, you will first add the directory for the newly generated version :

```
$ cd ~/apacheds/production/content/api/gen-docs
$ mkdir <version>
$ svn add <version>
```

Then copy the generated docs :

```
$ cp -r ~/apacheds/trunks/shared/target/checkout/target/site/apidocs ~/apacheds/production/content/api/gen-docs
/<version>
$ cp -r ~/apacheds/trunks/shared/target/checkout/target/site/xref ~/apacheds/production/content/api/gen-docs
/<version>
$
```

You have to check in those directories :

```
$ svn add <version>/*
$ svn ci <version> -m "Injected <version> javadocs"
```

Now, you have to update the staging site :

extpaths.txt

This file list the file on the production site that will not be overriden by the publication of the staging site. It has to be updated

```
$ cd ~/apacheds/staging/content/
$ vi extpaths.txt
```

Add the following line :

```
...
# API
api/gen-docs/<version>
...
```

then save and check in the file

.htaccess

We also have to update this file :

```
$ cd ~/apacheds/staging/content/api/gen-docs
$ vi .htaccess
```

And update the two last lines to refer to the version you've just released :

```
RewriteRule ^latest$ <version>/
RewriteRule ^latest/(.*)$ <version>/$1
```

Save and commit the file.

11. Update the web site

You can now update the site, add a news on the front page, and publish the site.

12. Inform the world !

After 24h, you can now inform the world about the release.

Send a mail to the users and dev mailing list, and one to the announce@apacge.org.

You are done !