Applications Release Process

Portals Applications (and Portals Bridges) Release Steps Overview

There are several Apache Portals Applications projects. Each project has its own independent release schedules and life cycles. All of the steps below apply to each Apache Portals Applications project releases, unless otherwise specified.

Apache Portals Bridges projects can be released in the same way described here, too. You will need to replace '/applications' by '/bridges' and the paths /names for artifacts accordingly.

Common Release Steps

1. Environment setup for releasing artifacts (same for SNAPSHOTs and releases)
    a. Increase the default Java heap available to Maven
    export MAVEN_OPTS="-Xmx256m -XX:MaxPermSize=128m"
    b. Use the JDK 1.6.
    c. Use Maven 3.0.x.
    d. Make sure the **Release Manager Configuration Guide** steps have been performed.
2. Prepare the source for release:
    a. Cleanup JIRA so the Fix Version in issues resolved since the last release includes this release version correctly.
    b. Update the text files in a working copy of the project root
        i. Review and update README file (README or README.txt) if exists and needed.
        ii. Commit any changes back to svn.
    c. Stage any Roadmap or Release landing pages on the site.
3. Checkout a clean copy of the trunk (on the specific branch if needed) to release using command line svn.
Do not use Eclipse to do the checkout. The extra dot (.) files created by Eclipse throws off the rat:check processing.
4. Verify the source has the required license headers before trying to release.
TODO: Consider to improve automation for this. For example, you can add a profile to check license headers. (e.g., *$ mvn -P pedantic verify -DskipTests*)
5. Do a dry run of the release:prepare step:
$ mvn -P apache-release release:prepare -DautoVersionSubmodules=true -DdryRun=true
The dry run will not commit any changes back to SVN and gives you the opportunity to verify that the release process will complete as expected.
You will be prompted for the following information :
    - Release version - take the default
    - SCM release tag - review the default naming and compare it with the existing svn tag structure; change it if needed.
    - New development version - take the default
    - GPG Passprhase
    If you cancel a release:prepare before it updates the pom.xml versions, then use the release:clean goal to just remove the extra files that were created.
    The Maven release plugin checks for SNAPSHOT dependencies in pom's. It will not complete the prepare goal until all SNAPSHOT dependencies are resolved.

1. Verify that the release process completed as expected
    a. The release plugin will create pom.xml.tag files which contain the changes that would have been committed to SVN. The only differences between pom.xml.tag and it's corresponding pom.xml file should be the version number(s) and svn location(s) (e.g., /trunk vs /tags).
    b. Check release.properties and make sure that the scm properties have the right version. Sometimes the scm location can be the previous version not the next version.
    c. Verify signatures (Verifying release signatures)
        - e.g., `gpg --verify target/apa-webcontent-1.3-SNAPSHOT-source-release.zip.asc target/apa-webcontent-1.3-SNAPSHOT-source-release.zip'
2. Cleanup the release prepare files again:
$ mvn -P apache-release release:clean
3. Prepare the release
    a. Run the "release:prepare" step for real this time. You'll be prompted for the same version information.
    $ mvn -P apache-release release:prepare -DautoVersionSubmodules=true
    b. Backup (zip or tar) your local release candidate directory in case you need to rollback the release after the next step is performed.
4. Perform the release
    - This step will create a maven staging repository and site for use in testing and voting.
    $ mvn -P apache-release release:perform

    - If your local OS userid doesn't match your Apache userid, then you'll have to also override the value provided by the OS to Maven for the site-deploy step to work. This is known to work for Linux, but not for Mac and unknown for Windows.
    -Duser.name=[your_apache_uid]

5. Verify the Nexus release artifacts
    a. Verify the staged artifacts in the nexus repo
        - https://repository.apache.org/index.html
        - Staging repositories (under Build Promotion) --> Name column --> org.apache.portals-XXX
        - Navigate through the artifact tree and make sure that all javadoc, sources, tests, jars, ... have .asc (GPG signature) and .md5 files. See http://people.apache.org/~henkp/repo/faq.html and http://www.apache.org/dev/release-signing.html#openpgp-ascii-detach-sig
    b. Close the nexus staging repo
        - https://repository.apache.org/index.html
        - Staging repositories (under Build Promotion) --> Name column --> org.apache.portals-XXX
        - Click checkbox for the open staging repo (org.apache.portals-XXX) and press Close in the menu bar.

6. Put the release candidate up for a vote
    a. Create a VOTE email thread on dev@ to record votes as replies. (See Example VOTE e-mail message)
    b. Perform a review of the release and cast your vote. See the following for more details on Apache releases _http://www.apache.org/dev/release.html_
    c. A -1 vote does not necessarily mean that the vote must be redone, however it is usually a good idea to rollback the release if a -1 vote is received.
        - If you want to recover it from a vetoed release, refer to the OpenJPA documentation, Recovering from a vetoed release.
    d. After the vote has been open for at least 72 hours, has at least three +1 PMC votes and no -1 votes, then post the results to the vote thread by -

        - reply to the initial email and prepend to the original subject "[RESULT]" (See [Example RESULT e-mail message|.. /Example_RESULT_message])

        - Include a list of everyone who voted +1, 0 or -1.
        - References:
            - Apache Voting Process
            - Who We Are (Apache Portals Project)
7. Finalizing a release
    a. Promote the staged nexus artifacts
        - https://repository.apache.org/index.html
        - Staging repositories (under Build Promotion) --> Name column --> org.apache.portals
        - Click checkbox of the closed staging repo (org.apache.portals-XXX) and select Release.
    b. Copy the source/binary artifacts over to the distribution area (managed by svnpubsub)
        - Command line examples to distribute Portals Application artifacts (apa-webcontent-1.3 as an example below.):

```
{{{#!raw
$ svn co https://dist.apache.org/repos/dist/release/portals/applications applications
$ cd applications/sources
$ echo "Copying source artifacts..."
$ wget --no-check-certificate https://repository.apache.org/content/repositories/releases/org/apache/portals/applications/apa-webcontent/1.3/apa-webcontent-1.3-source-release.zip
$ wget --no-check-certificate https://repository.apache.org/content/repositories/releases/org/apache/portals/applications/apa-webcontent/1.3/apa-webcontent-1.3-source-release.zip.asc
$ wget --no-check-certificate https://repository.apache.org/content/repositories/releases/org/apache/portals/applications/apa-webcontent/1.3/apa-webcontent-1.3-source-release.zip.md5
$ wget --no-check-certificate https://repository.apache.org/content/repositories/releases/org/apache/portals/applications/apa-webcontent/1.3/apa-webcontent-1.3-source-release.zip.sha1
$ svn add apa-webcontent-*
$ svn commit -m "Committing Source Release for apa-webcontent-1.3"

$ cd ../binaries
$ echo "Copying binaries artifacts..."
$ wget --no-check-certificate https://repository.apache.org/content/repositories/releases/org/apache/portals/applications/apa-webcontent-war/1.3/apa-webcontent-war-1.3.war
$ wget --no-check-certificate https://repository.apache.org/content/repositories/releases/org/apache/portals/applications/apa-webcontent-war/1.3/apa-webcontent-war-1.3.war.asc
$ wget --no-check-certificate https://repository.apache.org/content/repositories/releases/org/apache/portals/applications/apa-webcontent-war/1.3/apa-webcontent-war-1.3.war.md5
$ wget --no-check-certificate https://repository.apache.org/content/repositories/releases/org/apache/portals/applications/apa-webcontent-war/1.3/apa-webcontent-war-1.3.war.sha1
$ svn add apa-webcontent-*
$ svn commit -m "Committing Binary Release for apa-webcontent-1.3"
}}}
```

1. Update the JIRA versions page to close all issues, mark the version as "released", and set the date to the date that the release was approved. You may also need to make a new release entry for the next release.
2. Announcing the release
    a. Make a news announcement on the Apache Portals and/or Apache Portals Applications homepage.
        - WAIT 24hrs after committing releases for mirrors to replicate
        - Publish updates to the download page
            - How to publish the website through svnpubsub
                - Commit changes into the SVN location of Apache Portals Maven Site documentation:
                    - https://svn.apache.org/repos/asf/portals/site/
                    - NOTE: You will probably need to edit files under applications folder mostly, but be sure to edit some pages (e.g., news page) in the root, too.
                - Build the site of Apache Portals Maven Site documentation. e.g., $ mvn site
                - Copy the generated html and resources into the workspace folder of the following svnpubsub SVN location:
                    - https://svn.apache.org/repos/asf/portals/site-live/
                - Commit your changes
    b. Make an announcement about the release on the portalapps-user@portals.apache.org, portalapps-dev@portals.apache.org, and announce@apache.org list as per the Apache Announcement Mailing Lists page) (See Example ANNOUNCE e-mail message)
        - NOTE: Because announce@apache.org list allows message posting only to senders having '@apache.org' e-mail address. Therefore, be sure that you send the announce message by your apache e-mail address.
          See Committer e-mail configuration for detail.
        - NOTE: When sending announce message to multiple lists together from your @apache.org e-mail address,
          be careful because your message can be rejected if your @apache.org e-mail address didn't susbscribe user mailinglist (portalapps-user@portals.apache.org, in this case.)
          You can either subscribe it first from your @apache.org e-mail address or send the announce message twice separately.
        - NOTE: It is not required but it is better to digitally sign the announce e-mail message by using your PGP key.