HowToReleasePostMavenization

DEPRECATED! This doc refers to the releases already end of life. For current releases, please see HowToRelease.

This page is prepared for Hadoop Core committers. You need committer rights to create a new Hadoop Core release.

These instructions have been updated for Hadoop 0.23.x, 2.0 ~ 2.7.x.

Preparation

1. Bulk update Jira to unassign from this release all issues that are open non-blockers and send follow-up notification to the developer list that this was done.

1. If you have not already done so, update your @apache.org account via id.apache.org with your key; also add and commit your public key to the Hadoop repository KEYS, appending the output of the following commands:

   ```
   gpg --armor --fingerprint --list-sigs <keyid>
   gpg --armor --export <keyid>
   ```

   and publish your key at Signing Releases]. Once you commit your changes, log into `people.apache.org` and pull updates to `/www/www.apache.org/dist/hadoop/core`. For more details on signing releases, see [http://www.apache.org/dev/release-signing.html and Step-By-Step Guide to Mirroring Releases.
2. To deploy artifacts to the Apache Maven repository create `~/.m2/settings.xml`:

   ```
   <settings xmlns="http://maven.apache.org/SETTINGS/1.0.0"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation="http://maven.apache.org/SETTINGS/1.0.0
                         http://maven.apache.org/xsd/settings-1.0.0.xsd">
     <servers>
       <server>
        <id>apache.staging.https</id>
        <username>Apache username</username>
        <password>Apache password</password>
       </server>
     </servers>
   </settings>
   ```

Branching

Skip this section if this is NOT the first release in a series (i.e. release X.Y.0).

1. Notify developers on the #hadoop IRC channel that you are about to branch a release.
2. Update `CHANGES.txt` to include the release version and date (use `Unreleased` for the date if it is unknown) and remove `Trunk (unreleased changes)`.
3. Commit these changes to trunk.

   ```
   svn commit -m "Preparing for release X.Y.Z"
   ```

4. Create a branch for the release series:

   ```
   svn copy https://svn.apache.org/repos/asf/hadoop/common/trunk \
   https://svn.apache.org/repos/asf/hadoop/common/branches/branch-X.Y -m "Branching for X.Y releases"
   ```

5. Update `CHANGES.txt` to add back in `Trunk (unreleased changes)`.
   a. Update the default version in the pom files on trunk to X.Y+1.0-SNAPSHOT `mvn versions:set -DnewVersion=X.Z-SNAPSHOT`.

    b. Update the `project.version` number in `hadoop-hdfs-project/hadoop-hdfs/src/test/aop/build/aop.xml` on trunk to X.Y+1.0.
    c. Update the `symlink link` number in `hadoop-mapreduce-project/hadoop-mapreduce-client/hadoop-mapreduce-client-app/pom.xml` on trunk to X.Y+1.0.
6. Commit these changes to trunk.

```
svn commit -m "Preparing for X.Y+1.0 development"
```

Updating Release Branch

These operations take place in the release branch.

1. Check out the branch with:

```
svn co https://svn.apache.org/repos/asf/hadoop/common/branches/branch-X.Y
```

2. Update `CHANGES.txt` to include the release version and date (this change must be committed to trunk and any intermediate branches between trunk and the branch being released).
3. Generate `releasenotes.html` with release notes for this release. You generate these with:

```
python ./dev-support/relnotes.py -v $(vers)
```

   If you release includes more then one version you may add additional -v options for each version. By default the previousVersion mentioned in the notes will be X.Y.Z-1, if this is not correct you can override this by setting the --previousVer option.

    a. Update `releasenotes.html`

```
mv releasenotes.$(vers).html ./hadoop-common-project/hadoop-common/src/main/docs/releasenotes.html
```

       Note that the script generates a set of notes for HDFS, HADOOP, MAPREDUCE, and YARN too, but only common is linked from the html documentation so the indavidual ones are ignored for now.
    b. Update the version number in the pom files on trunk to X.Y.N `mvn versions:set -DnewVersion=X.Z.N` where *N* is one greater than the release being made.
    c. Update the `project.version` number in `hadoop-hdfs-project/hadoop-hdfs/src/test/aop/build/aop.xml` on trunk to X.Y.N.
    d. Update the `symlink link` number in `hadoop-mapreduce-project/hadoop-mapreduce-client/hadoop-mapreduce-client-app/pom.xml` on trunk to X.Y.N.
4. Commit these changes.

```
svn commit -m "Preparing for release X.Y.Z"
```

5. If not already done, merge desired patches from trunk into the branch and commit these changes. You can find the revision numbers using `svn log CHANGES.txt` in the branch and in trunk.

```
cd branch-X.Y
svn merge -rR1:R2 ../trunk .
svn commit -m "Merge -r R1:R2 from trunk to X.Y branch. Fixes: HADOOP-A, HADOOP-B."
```

6. Run mvn rat-check:

```
mvn apache-rat:check
```

   (Look for errors in rat.txt in the appropriate maven module.)

    a. Tag the release candidate (R is the release candidate number, and starts from 0):

```
svn copy https://svn.apache.org/repos/asf/hadoop/common/branches/branch-X.Y \
https://svn.apache.org/repos/asf/hadoop/common/tags/release-X.Y.Z-rcR -m "Hadoop X.Y.Z-rcR
release."
```

Build Requirements

To build an official release, you must:

1. Use a 32-bit JVM. Currently we don't have support for 64-bit binaries in our maven builds.
2. Change refs to -SNAPSHOT in the following files:

```
hadoop-hdfs-project/hadoop-hdfs/src/test/aop/build/aop.xml
hadoop-mapreduce-project/hadoop-mapreduce-client/hadoop-mapreduce-client-app/pom.xml
```

Building the Release Candidate (RC)

1. Build the release & run unit tests. This is captured in http://svn.apache.org/viewvc/hadoop/nightly/hudsonBuildHadoopRelease.sh?view=markup.

```
# mvn clean
$ mvn clean

# set version
$ export version=0.23.1
$ mvn versions:set -DnewVersion=${version}

# make the distribution (do not use 'clean' or -Dtar, for now do not use install either since it breaks -
Psrc)
$ mvn package install -Dmaven.test.skip.exec=true
$ mvn deploy -Psign,src,native,dist -Dmaven.test.skip.exec=true -Dcontainer-executor.conf.dir=/etc/hadoop
/conf

# stage site
$ mvn site
$ mvn site:stage

# release notes
$ cp hadoop-common-project/hadoop-common/src/main/docs/releasenotes.html target/staging/hadoop-project
/hadoop-project-dist/hadoop-common

# copy CHANGES.txt
$ cp hadoop-common-project/hadoop-common/CHANGES.txt target/staging/hadoop-project/hadoop-project-dist
/hadoop-common
$ cp hadoop-hdfs-project/hadoop-hdfs/CHANGES.txt target/staging/hadoop-project/hadoop-project-dist
/hadoop-hdfs
$ mkdir target/staging/hadoop-project/hadoop-project-dist/hadoop-yarn
$ cp hadoop-yarn-project/CHANGES.txt target/staging/hadoop-project/hadoop-project-dist/hadoop-yarn
$ mkdir target/staging/hadoop-project/hadoop-project-dist/hadoop-mapreduce
$ cp hadoop-mapreduce-project/CHANGES.txt target/staging/hadoop-project/hadoop-project-dist/hadoop-
mapreduce

# copy site + javdocs
$ cp -R target/staging/hadoop-project/* hadoop-dist/target/hadoop-${version}/share/doc/hadoop/

# finally, create src/binary tarballs
$ cd hadoop-dist/target

# src tarball after copying NOTICE.txt README.txt LICENSE.txt
$ tar -xzf hadoop-${version}-src.tar.gz
$ cp ../../hadoop-common-project/hadoop-common/NOTICE.txt hadoop-${version}-src
$ cp ../../hadoop-common-project/hadoop-common/README.txt hadoop-${version}-src
$ cp ../../hadoop-common-project/hadoop-common/LICENSE.txt hadoop-${version}-src
$ tar -czf hadoop-${version}-src.tar.gz hadoop-${version}-src

# binary tarball after copying NOTICE.txt README.txt LICENSE.txt
$ cp ../../hadoop-common-project/hadoop-common/NOTICE.txt hadoop-${version}
$ cp ../../hadoop-common-project/hadoop-common/README.txt hadoop-${version}
$ cp ../../hadoop-common-project/hadoop-common/LICENSE.txt hadoop-${version}
$ tar -czf hadoop-${version}.tar.gz hadoop-${version}
```

2. Check that release file looks ok - e.g. install it and run examples from tutorial.
    a. Generate the checksums of the release file.

```
$ gpg --print-mds hadoop-${version}-src.tar.gz > hadoop-${version}-src.tar.gz.mds
$ gpg --print-mds hadoop-${version}.tar.gz > hadoop-${version}.tar.gz.mds
```

3. Sign the release

```
$ gpg --armor --output hadoop-${version}-src.tar.gz.asc --detach-sig hadoop-${version}-src.tar.gz
$ gpg --armor --output hadoop-${version}.tar.gz.asc --detach-sig hadoop-${version}.tar.gz
```

4. Copy release files to a public place.

```
ssh people.apache.org mkdir public_html/hadoop-X.Y.Z-candidate-0
scp -p hadoop-${version}.tar.gz* people.apache.org:public_html/hadoop-${version}-candidate-0
```

5. Log into Nexus, select `Staging` from the left navigation pane, right-click on the pushed repository, and `close` the release.
    a. Call a release vote on common-dev at hadoop.apache.org.

Publishing

In 7 days if the release vote passes, the release may be published.

1. Tag the release:

```
svn move https://svn.apache.org/repos/asf/hadoop/common/tags/release-X.Y.Z-rcR \
https://svn.apache.org/repos/asf/hadoop/common/tags/release-X.Y.Z -m "Hadoop X.Y.Z release."
```

2. Copy release files to the distribution directory and make them writable by the hadoop group.

```
ssh people.apache.org
cp -pr public_html/hadoop-${version}-candidate-0 /www/www.apache.org/dist/hadoop/core/hadoop-${version}
cd /www/www.apache.org/dist/hadoop/core
chgrp -R hadoop hadoop-${version}
chmod -R g+w hadoop-${version}
```

3. The release directory usually contains just two releases, the most recent from two branches, with a link named 'stable' to the most recent recommended version.

```
ssh people.apache.org
cd /www/www.apache.org/dist/hadoop/core
rm -rf hadoop-${version}; rm stable
ln -s hadoop-${version} stable
```

4. In Nexus, effect the release of artifacts by right-clicking the staged repository and select `Release`
5. Wait 24 hours for release to propagate to mirrors.
    a. Prepare to edit the website.

```
svn co  https://svn.apache.org/repos/asf/hadoop/common/site/main
```

6. Update the documentation links in `author/src/documentation/content/xdocs/site.xml`.
7. Update the release news in `author/src/documentation/content/xdocs/releases.xml`.
    a. Copy the new release docs to svn and update the `docs/current` link, by doing the following:

```
tar xvf /www/www.apache.org/dist/hadoop/core/hadoop-${version}/hadoop-${version}.tar.gz
cp -rp hadoop-${version}/share/doc/hadoop publish/docs/r${version}
rm -r hadoop-${version}
rm current
ln -s r${version} current
svn add publish/docs/r${version}
```

8. Regenerate the site, review it, then commit it.

```
ant -Dforrest.home=/usr/local/forrest -Djava5.home=/usr/local/jdk1.5
firefox publish/index.html
svn commit -m "Updated site for release X.Y.Z."
```

9. Send announcements to the user and developer lists once the site changes are visible.
    a. In Jira, ensure that only issues in the "Fixed" state have a "Fix Version" set to release X.Y.Z.
    b. In Jira, "release" the version. Visit the "Administer Project" page, then the "Manage versions" page. You need to have the "Admin" role in Hadoop Core's Jira for this step and the next.
    c. In Jira, close issues resolved in the release. Disable mail notifications for this bulk change.

See Also

- Apache Releases FAQ