HowToRelease using maven release plugin

Setting up the signing keys

Before you do a release you'll need a signing key that is registered with apache. If you already have one, you can skip this section. Otherwise, here are the steps to do:

1. use **gpg2 --gen-key** to generate a new key. Make sure that the key size is 4096 bits, the key doesn't expire, and use CODE SIGNING KEY for the comment.
2. Now you need to register your key at http://pgp.mit.edu/ using the output of **gpg --armor --export <keyid>**
3. Add your key to the KEYS file:
    a. svn co https://dist.apache.org/repos/dist/release/zookeeper/ zookeeper_dist
    b. cd zookeeper_dist.
    c. edit the KEYS file. See the top of the KEYS file for instructions on how to edit this file
    d. svn ci
4. Upload digest to your account on id.apache.org (gpg2 --fingerprint)

Useful Links and Background:

Apache infra page: https://cwiki.apache.org//confluence/display/INFRA/Index

Apache self-service page - **now allows for forcing git repo sync**: https://selfserve.apache.org/

Important Notes

- when updating JIRA be sure to use the "batch change" operator under the "tools" menu. Disable email notifications when using batch change, subsequent to large JIRA operations (moving issues btw releases, closing JIRAs after a release, etc...) send email to the dev@ list detailing the changes.

Configuring maven

Make sure your settings.xml in ~/.m2 contain logins for apache repos, and your signing key is published and configured here.

Please follow the instructions here to encrypt your Apache credentials: https://maven.apache.org/guides/mini/guide-encryption.html

.m2 settings

```
<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>
    <!-- To publish a snapshot -->
    <server>
      <id>apache.snapshots.https</id>
      <username>YOUR_APACHE_ID</username>
      <password>YOUR_APACHE_PASSWORD</password>
    </server>

    <!-- To stage a release -->
    <server>
      <id>apache.staging.https</id>
      <username>YOUR_APACHE_ID</username>
      <password>YOUR_APACHE_PASSWORD</password>
    </server>

  <server>
           <id>apache.releases.https</id>
    <username>YOUR_APACHE_ID</username>
    <password>YOUR_APACHE_PASSWORD</password>
  </server>

 </servers>
 <profiles>
    <profile>
      <id>apache-release</id>
      <properties>
            <!-- gpg list-keys will show you your keyname ( something like 00A5F21E) -->
            <gpg.keyname>YOUR_KEYNAME</gpg.keyname>
                  <gpg.passphrase>YOUR_KEY_PASSWORD</gpg.passphrase>
      </properties>
    </profile>
  </profiles>
</settings>
```

For a more detailed steps, check out How To Release page for Hadoop. It is used as a base.

Smoke Tests

Before the release, run the following smoke tests (at least).

- Run unit tests on different hardwares: mac, windows, linux.

Checkout a fresh new clone

Check out a fresh new copy of the repository:

```
git clone https://gitbox.apache.org/repos/asf/zookeeper.git
cd zookeeper
```

Test your docker environment

We are going to perform the full procedure using docker, this way we are sure that the binary artifacts are built using a know environment:

- Linux
- Java 8
- Maven
- OpenSSL

Enter the docker environment (this should work on Mac and on Linux).

In some environment with SELinux you have to enable "permissive mode" ("sudo setenforce Permissive")

```
dev/docker/run.sh
```

Build Java anc C sources, run tests and test GPG signature

```
export GPG_TTY=$(tty)

mvn clean install -Pfull-build -Papache-release -DskipTests
mvn dependency-check:check
```

Manage JiRA issues

- In Jira, move open issues to next version. Disable mail notifications for this bulk change. If you are in doubt as on dev@zookeeper.apache.org mailing list before proceedig

Create a branch (when you create X.Y.0 and move master to X.Y+1.0)

Create a branch for X.Y.Z (the current release candidate)

```
RC_NUM=0
BRANCH_NAME=branch-3.6
WORK_BRANCH=branch-3.6.0
RELEASE_DEVELOPMENT_VERSION=3.6.1-SNAPSHOT
RELEASE_VERSION=3.6.0
MASTER_DEVELOPMENT_VERSION=3.7.0-SNAPSHOT
TAG=release-$RELEASE_VERSION-$RC_NUM


git checkout master
git pull --rebase
git clean -xdf
mvn release:clean
mvn release:branch \
    -DbranchName=$BRANCH_NAME \
    -DdevelopmentVersion=$MASTER_DEVELOPMENT_VERSION \
    -Pfull-build
```

That command will:

- change the version in master branch
- create a new release branch

The goal "branch" in Maven Release plugin does not support "preparationGoals" and "completionGoals" so we have to to manually change the version inside the C-client

So now you have to set the version in zookeeper-client/zookeeper-client-c

In 3.7.0 we have:

- configure.ac
- CMakelist
- include/zookeeper_version.hs

Then push a commit with the fix.

It is now time to create CI jobs for the new branch, if you do not have permissions to edit CI configuration just send an email to dev@zookeeper.apache.org

Preparation for minor release, say 3.6.1

If you are releasing a version that is not the first one of a main branch just checkout the release branch

```
RC_NUM=0
BRANCH_NAME=branch-3.6
WORK_BRANCH=branch-3.6.1
RELEASE_DEVELOPMENT_VERSION=3.6.2-SNAPSHOT
RELEASE_VERSION=3.6.1
TAG=release-$RELEASE_VERSION-$RC_NUM

git checkout $BRANCH_NAME
git pull
git push --dry-run
```

Create a work branch for the release

If you are releasing a version that is not the first one of a main branch just checkout the release branch

```
RC_NUM=0
BRANCH_NAME=branch-3.6
WORK_BRANCH=branch-3.6.0
RELEASE_DEVELOPMENT_VERSION=3.6.1-SNAPSHOT
RELEASE_VERSION=3.6.0
TAG=release-$RELEASE_VERSION-$RC_NUM

git checkout $BRANCH_NAME
git checkout -b $WORK_BRANCH
git pull
git push --set-upstream origin $WORK_BRANCH
git status
# you are now in release-x.y.z branch
```

Create release notes on release branch

Update the copyright years in NOTICE.txt if it's outdated.

Update **zookeeper-docs/src/main/resources/markdown/releasenotes.md** with release notes for this release. You can get the HTML by following the "Release Notes" link for the relevant release on thehttps://issues.apache.org/jira/projects/ZOOKEEPER?selectedItem=com.atlassian.jira.jira-projects-plugin%3Arelease-page&status=unreleased tab in Jira. Note that you need to exclude the won't fix or invalid tickets.
Hints for editing releasenotes.md (using vim) - you can skip 'c' parameter, it tells vim to ask for all replacement:

```
:%s/<h2>\s\+/## /gc
:%s/<\/h2>//gc
:%s/<[\/]*ul>\n//gc
:%s/<li>\[<a href='\(.*\)'>\(ZOOKEEPER-[0-9]\+\)<\/a>\] - \+/* [\2](\1) - /gc
:%s/<\/li>\n//gc
```

Push these changes to the $WORK_BRANCH (release-3.6.0) and to the $RELEASE_BRANCH (branch-3.6) (**no review is needed at this point**)

In case you have to run more than one RC you have to update the release notes every time and push them to those two branches.

Please make sure that the entire history of the actual release linue is present in the release notes. For example in case of release 3.6.3, the notes from 3.6.0 to 3.6.3 are included.

Build and check that the release branch is in good shape

Run spotBugs, checkstyle, rat and OWASP checks:

```
git checkout $WORK_BRANCH
git clean -xdf
# perform quality checks
mvn clean apache-rat:check -DskipTests -Pfull-build -Papache-release
mvn clean install checkstyle:check spotbugs:check -DskipTests -Pfull-build -Papache-release
mvn dependency-check:check
```

Perform a release on branch (for major and minor release) (remove SNAPSHOT and create tag)

```
git checkout $WORK_BRANCH
git clean -xdf

mvn release:prepare \
   -DautoVersionSubmodules=true  \
   -DreleaseVersion=$RELEASE_VERSION \
   -Dtag=$TAG  \
   -DdevelopmentVersion=$RELEASE_DEVELOPMENT_VERSION \
   -Darguments='-DskipTests=true -Pfull-build,apache-release' \
   -Pfull-build,apache-release
```

see details in ~~ZOOKEEPER-3635~~ - Use Docker and Maven Release Plugin to prepare ZooKeeper releases CLOSED

The command above perform the following activities:

- removes -SNAPSHOT in all of the pom.xml files (the version is RELEASE_VERSION)
- build the project (even the C client and contrib packages)
- ensures the correct version in C client files (configure.ac, CMakelink.txt, zookeeper_version.h) **!! The command will not update the version in zookeeper_version.h, it has to be manually updated !!**
- created a signed tag
- bump the branch to the next minor version (RELEASE_DEVELOPMENT_VERSION)

Preparing the release candidate (prepare stage artifacts for VOTE)

- dUsing the same directory in which you run "release:prepare" perform these are steps
    - Maven Release Plugin uses release.properties files prepared by release:prepare
    - It checks out a new copy of the repository at gitsha pointed by $TAG (like release-3.6.0-0) inside zookeeper/target/checkout
- There is no need to build the C client or "contrib" packages

  ```
  mvn release:perform -Darguments='-DskipTests=true -Papache-release' -Papache-release
  ```

- You should have the SIGNED jar files and also the tarballs built. The two tarball is available at:
    - target/checkout/zookeeper-assembly/target/apache-zookeeper-X.Y.Z-bin.tar.gz
    - target/checkout/target/apache-zookeeper-X.Y.Z-source-release.tar.gz
- This command also staged all of the maven artifacts to repository.apache.org

Self validate the Release candidate

Check that release file looks ok - e.g. install it and run examples from tutorial.

- untar the release artifact in a test directory
- try to build the sources with

```
mvn clean install -DskipTests -Pfull-build
```

- validate the contents of the staging Maven repository
- **test the staged binaries with a JDK8 runtime**

Push branches and release tag to upstream

Push the following:

- Work branch
- Release branch
- Release tag

```
git push
git push origin $WORK_BRANCH:$BRANCH_NAME
git push origin $TAG
```

Close the Maven Repository

- Enter Apache Nexus and do the following:
    1. Click on Log In in the upper right corner. Log in using your apache user name and password.
    2. In the left navigation pane, select Staging Repositories.
    3. Identify the release candidate you just pushed, by your user name (in parentheses as part of the "Repository" name) and the "Created On" date. Click on the check box to the left of your Repository name to select it. (If you accidentally click on the Repository name itself, another tab will pop open. If so, just close it.)
    4. Click the Close button above the Repository names. This makes your release candidate available at the Staging level.
    5. If you have previously staged an older release candidate with the same version number, and it is still showing in the Repository list, you must select and Drop the old one now.
    6. Confirm that your new release candidate is visible at https://repository.apache.org/content/groups/staging/org/apache/zookeeper/zookeeper/ with correct file modification dates.

Stage source and binaries to your own Apache home (old procedure)

Copy release files to a public place and ensure they are readable. Note that home.apache.org only supports SFTP, so this may be easier with a graphical SFTP client like Nautilus, Konqueror, etc.

Stage source and binaries to Apache dev repo

Copy release files to a public place and ensure they are readable.

```
# Staged artifacts built by "release:perform" are in target/checkout

ZKHOME=where you have your work directory "zookeeper"
svn co https://dist.apache.org/repos/dist/dev/zookeeper/ zookeeper_dev
cd zookeeper_dev
mkdir zookeeper-$RELEASE_VERSION-candidate-$RC_NUM
cd zookeeper-$RELEASE_VERSION-candidate-$RC_NUM
cp $ZKHOME/target/checkout/target/*.tar.gz* .
cp $ZKHOME/target/checkout/zookeeper-assembly/target/*-bin.tar.gz* .
rm *.asc.sha512

mkdir website
cp -r $ZKHOME/target/checkout/zookeeper-docs/target/html/* website/
cd ..
svn add zookeeper-$RELEASE_VERSION-candidate-$RC_NUM
svn ci -m "Add ZooKeeper $RELEASE_VERSION release candidate $RC_NUM"
```

Call for VOTE

- Call for a release vote on **dev** (note dev@ and not user@, the user list is for discussion of released software only) at zookeeper.apache.org. Here is a sample email (from 3.4.6 release):

```
Subject: [VOTE] Apache ZooKeeper release 3.8.0 candidate 0


This is a release candidate for 3.8.0.

It is a major release and it introduces a lot of new features, most notably:
- Migration of the logging framework from Apache Log4j1 to LogBack
- Read Key/trust store password from file (and other security related
improvements)
- Restored support for OSGI
- Reduced the performance impact of Prometheus metrics
- Official support for JDK17 (all tests are passing)
- Updates to all the third party dependencies to get rid of every known CVE.

The full release notes is available at:

https://issues.apache.org/jira/secure/ReleaseNote.jspa?projectId=12310801&version=12349587

*** Please download, test and vote by February 28th 2022, 23:59 UTC+0. ***

Source files:
https://dist.apache.org/repos/dist/dev/zookeeper/zookeeper-3.8.0-candidate-1/

Maven staging repo:
https://repository.apache.org/content/repositories/orgapachezookeeper-1073/

The release candidate tag in git to be voted upon: release-3.8.0-1
https://github.com/apache/zookeeper/tree/release-3.8.0-1

ZooKeeper's KEYS file containing PGP keys we use to sign the release:
https://www.apache.org/dist/zookeeper/KEYS

The staging version of the website is:
https://dist.apache.org/repos/dist/dev/zookeeper/zookeeper-3.8.0-candidate-1/website/index.html


Should we release this candidate?



Release Manager Signature
```

Cancelling a Release Candidate

In case you get a -1 you have to work together with the community in order to see if that vote is a real blocker for the release or not.

In case it is a real blocker we have to fix the problem and cherry-pick the fix to $BRANCH_NAME and to $WORK_BRANCH.

Send an explicit email in answer to the VOTE thread and state that current RC VOTE is cancelled.

"Drop" the Maven Staging Repository at repository.apache.org.

Then when you are okay restart from "Perform a release on branch" step.

In order to create another release candidate, you need to revert the latest two commits on the work and the release branch.

```
git revert <commit id for "Prepared $RELEASE_DEVELOPMENT_VERSION-SNAPSHOT">
git revert <commit id for "Prepared $RELEASE_VERSION">
```

This must be done *before* cherry-picking the required commits from the master branch. It's also possible to hard reset the latest commit (the SNAPSHOT one), but keep the other intact, because you don't want to delete the previous RC tag.

Rollback

If you messed up something before publishing a release candidate, feel free to reset the branches to the commit before those "Prepared" commits. You can also delete the RC tag locally and remotely if needed.

Once your work branch is in a good shape, you can force push it.

```
git tag -d $TAG                     # delete locally
git push --delete origin $TAG       # delete remotely
git push --force
```

Publishing

- Once three PMC members have voted for a release, it may be published.

1. Answer to the VOTE email thread with a final RESULT email like this one:
2. ```
   Subject: [RESULT] [VOTE] Apache ZooKeeper release 3.6.0 candidate 0

   I'm happy to announce that we have unanimously approved this release.

   There are 7 approving votes, 5 of which are binding:

   - approver 1...
   - approver 2...(binding)
   - approver 3...

   There are no disapproving votes.

   I will promote the artifacts and complete the release procedure.

   Thanks to every one who contributed to this great release !

   Release Manager Signature
   ```
3. Compare and sync the WORK_BRANCH with the RELEASE_BRANCH, check RELEASE NOTES and Maven Version in pom.xml and in the C client:
4. Update the version in $BRANCH_NAME (like branch-3.6), the Maven Release plugin pushed it to the WORK_BRANCH, **you should cherry pick it** (actually they are 2 commits, one to remove SNAPSHOT and one to bump the version) but in case of conflict (if the branch diverged) you can do it with this command but you will have to manually update the C client files

   ```
   git checkout $BRANCH_NAME
   git pull
   mvn release:update-versions -DdevelopmentVersion=$RELEASE_DEVELOPMENT_VERSION -Pfull-build
   # verify that we are committing only pom.xml files
   git status
   git diff
   git add .
   git commit -m "Start ZooKeeper $RELEASE_DEVELOPMENT_VERSION"
   git push
   ```
5. Tag the release. Do it from the release branch and push the created tag to the remote repository, drop the $WORK_BRANCH:

   ```
   git checkout $TAG
   # create a signed tag
   RELEASE_TAG=release-$RELEASE_VERSION
   git tag -s $RELEASE_TAG -m "ZooKeeper $RELEASE_VERSION release."
   git branch -D $WORK_BRANCH
   git push -d origin $WORK_BRANCH
   # push the newly created release tag to the remote repo (it should be "origin" if you cloned a
   frash new copy of apache repo).
   git push origin $RELEASE_TAG
   ```
6. Check in release files to the distribution server
   ```

7.
```
# copy locally the staged artifacts and the website
svn co https://dist.apache.org/repos/dist/dev/zookeeper/ zookeeper_dev
cd zookeeper_dev
svn rm https://dist.apache.org/repos/dist/dev/zookeeper/zookeeper-$RELEASE_VERSION-
candidate-$RC_NUM/website -m "Remove staged website for ZooKeeper ${RELEASE_VERSION}rc${RC_NUM}"

svn move https://dist.apache.org/repos/dist/dev/zookeeper/zookeeper-$RELEASE_VERSION-
candidate-$RC_NUM https://dist.apache.org/repos/dist/release/zookeeper/zookeeper-$RELEASE_VERSION -
m "Release Apache ZooKeeper $RELEASE_VERSION"
```

    a. After checking in the release, you'll receive an automated email from reporter.apache.org with a link requesting additional details about the release. This form must be completed by a PMC member. If you are a PMC members, follow the link and complete the form. If you are not a PMC member, forward the email to dev@zookeeper.apache.org and ask for assistance from a PMC member.
8. 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.

```
svn co https://dist.apache.org/repos/dist/release/zookeeper zookeeper_dist
cd zookeeper_dist
svn rm zookeeper-A.B.C # apache folks don't like old releases hanging around - they are available
in the archive if people need access
rm stable; rm current
ln -s zookeeper-A.B.D stable
ln -s zookeeper-A.B.D current
svn ci -m 'Updating links'
```

9. Release the Maven artifacts on Apache Nexus:
    a. Click on 'Staging repositories'.
    b. Select the repository corresponding to the release candidate.
    c. Click on the 'Release' button and follow instructions.
10. Wait 24 hours for release to propagate to mirrors.
11. Prepare to edit the website. ZooKeeper uses Jekyll/Markdown with gitpubsub. See WebSiteSetup for general instructions and tool setup /prerequisites.

```
git clone -b website https://gitbox.apache.org/repos/asf/zookeeper.git
```

12. Copy the new release documentation into the _released_docs directory

```
cd _released_docs
mkdir r$RELEASE_VERSION
cd r$RELEASE_VERSION
tar xvf $ZKHOME/target/checkout/zookeeper-assembly/target/apache-zookeeper-$RELEASE_VERSION-bin.
tar.gz --strip-components 2 "apache-zookeeper-$RELEASE_VERSION-bin/docs"
rm apache-zookeeper-$RELEASE_VERSION-bin.tar.gz
# Update the "current" doc pointer if necessary (next 2 lines)
cd ..
rm current
ln -s r$RELEASE_VERSION current
git add r$RELEASE_VERSION current
cd ..
```

13. Change directory to **src/main/resources/markdown**
14. Update the release news in **releases.md**
15. Update the nav panel in **html/header.html** and **documentation.md** to include X.Y.Z
16. Stage changes to git

```
git add releases.md documentation.md html/header.html
```

17. Regenerate the site, review it, then commit it. (install Jekyll/pygments if you haven't already - see the README.md in website branch)

```
mvn clean install
cp -RP _released_docs target/html/doc
# at this point verify that the site is working properly - open _site/index.html
# once you are happy move on to the next step...
git commit -m "Updated website content for release X.Y.Z."

git push origin website
```

18. Deploy: you are now ready to deploy the changes to the public website using gitpubsub. Using the same repo as the previous step:

```
git checkout asf-site
rm -fr content
mv target/html content
git add content
# verify that the content of "content" looks proper. open content/index.html in a browser and
verify, etc...
# verify git staging looks proper - remember, we added the release docs to content/doc/rX.Y.Z and
that will also be added here
git commit -m "Website update for release X.Y.Z"

git push origin asf-site
```

    If the apache zookeeper website does not update, check if the commit is mirrored into github.com/apache/zookeeper asf-site branch
19. Send announcements to the **user and developer** lists once the site changes are visible. Here is a sample email:

```
Subject: [ANNOUNCE] Apache ZooKeeper X.Y.Z

The Apache ZooKeeper team is proud to announce Apache ZooKeeper version X.Y.Z

ZooKeeper is a high-performance coordination service for distributed
applications. It exposes common services - such as naming,
configuration management, synchronization, and group services - in a
simple interface so you don't have to write them from scratch. You can
use it off-the-shelf to implement consensus, group management, leader
election, and presence protocols. And you can build on it for your
own, specific needs.

For ZooKeeper release details and downloads, visit:
https://zookeeper.apache.org/releases.html

ZooKeeper X.Y.Z Release Notes are at:
https://zookeeper.apache.org/doc/rX.Y.Z/releasenotes.html

We would like to thank the contributors that made the release possible.

Regards,

The ZooKeeper Team
```

20. In Jira, ensure that only issues in the "Fixed" state have a "Fix Version" set to release X.Y.Z.
21. In Jira, "release" the version. Visit the "Administer Project" page, then the "Manage versions" page. You need to have the "Admin" role in ZooKeeper's Jira for this step and the next.
22. In Jira, close issues resolved in the release. Disable mail notifications for this bulk change.