HowToRelease using git - OUTDATED! Used for branch-3.4

For a how-to for branch-3.5, see the guide here.

For a how-to for branch-3.6+, see the guide here.

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.

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 mailing list that you are about to branch a release.
2. Create a branch for the release series:

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

# create new branch-X.Y from master branch
git checkout -b branch-X.Y master

# push the new branch to remote repo
git push <remote> branch-X.Y
```

3. Checkout master branch.

```
git checkout master
```

4. Update the default version in **build.xml** on master to X.Y+1.0-SNAPSHOT.
5. Update the version number in **src/c/configure.ac** and **src/c/include/zookeeper_version.h** to be X.Y+1.0.
6. Commit these changes to master.

```
# check for modified files
git status

# add individual files one by one
git add build.xml
git add src/c/configure.ac
git add src/c/include/zookeeper_version.h

# commit and push the changes to remote repo
git commit -m "Preparing for X.Y+1.0 development"
git push <remote> master
```

Updating the release branch

These operations take place in the release branch:

1. Check out the branch with:

```
git clone -b branch-X.Y https://gitbox.apache.org/repos/asf/zookeeper.git
```

2. If you are creating a point release (almost always the case) then follow this step. The exception to this is typically only when creating a new release candidate because the previous candidate failed during voting (rc0 failed and you are creating rc1+) – in which case skip this step.

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

```
git branch branch-X.Y.Z
git push <remote> branch-X.Y.Z
```

   Update the version number in **build.xml**, **src/c/configure.ac**, **src/c/CMakeLists.txt**, and **src/c/include/zookeeper_version.h** to the next logical SNAPSHOT version (e.g. X.Y.Z to X.Y.(Z+1)-SNAPSHOT)

   Commit this change.

```
# check for modified files
git status

# add modified files one by one
git add <modified files>

# commit and push the changes to remote repo branch-X.Y
git commit -m "Prepare for the next release: update the version to X.Y.(Z+1)-SNAPSHOT"
git push <remote> branch-X.Y
```

3. Checkout the active point release branch - this is where you will be creating the release candidate

```
git checkout branch-X.Y.Z
```

4. If not already done, cherry-pick desired commits into branch-X.Y.Z. If cherry-picks are done from other than branch-X.Y ensure that those changes are also committed to branch-X.Y. If there are conflicts, then it is preferable to produce a new patch for this branch, review it separately and commit it via JIRA. Please refer HowToContribute and Committing changes page for pushing changes to the project.

```
# The -x option records the source commit, and reuses the original commit message
git cherry-pick -x <commit-hash>
```

5. Do a final pass regeneration of docs by running "ant -Dforrest.home=<path to Forrest> clean docs". If there are any modifications, then they'll show up under the docs folder. If there are any changes, then commit them. **However, DO NOT commit changes in docs/releasenotes.html or docs/releasenotes.pdf.** (Release notes are handled in a separate step.) Release notes should be regenerated each time a documentation patch is committed. This final pass before the release is intended to catch mistakes in case this step was missed at commit time.

6. 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 the https://issues.apache.org/jira/browse/ZOOKEEPER?report=com.sourcelabs.jira.plugin.portlet.releases:releases-projecttab#selectedTab=com.atlassian.jira.plugin.system.project%3Aroadmap-panel tab in Jira. Note that you need to exclude the won't fix or invalid tickets.
7. Update the version number in **build.xml** to be "X.Y.Z" (i.e. remove SNAPSHOT designator)
8. Update the version number in **src/c/configure.ac**, **src/c/CMakeLists.txt**, and **src/c/include/zookeeper_version.h** to be "X.Y.Z"
9. Update the copyright years in NOTICE.txt if it's outdated.
10. Commit these changes.

```
# check for modified files
git status

# add modified files one by one
git add <modified files>

# commit and push the changes to remote repo branch-X.Y
git commit -m "Preparing for release X.Y.Z"
git push <remote> branch-X.Y.Z
```

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

```
# create a signed tag
git tag -s release-X.Y.Z-rcR -m "ZooKeeper X.Y.Z-rcR release."

# push the newly created rc tag to the remote repo.
git push <remote> release-X.Y.Z-rcR
```

Building

1. Build the release & run unit tests.

```
ant test package tar
```

⚠️ Don't skip the "test" part and don't just do "ant package tar"! See here for why.

2. Untar the generated zookeeper-X.Y.Z.tar.gz and do the following:
    a. Sign the toplevel zookeeper-X.Y.Z.jar

    ```
    gpg --armor --output zookeeper-X.Y.Z.jar.asc --detach-sig zookeeper-X.Y.Z.jar
    ```

    b. Sign each file in dist-maven

    ```
    for i in dist-maven/*.jar dist-maven/*.pom; do gpg --armor --output $i.asc --detach-sig $i; done
    ```

3. Tar/GZ the release directory as zookeeper-X.Y.Z.tar.gz

```
tar czf zookeeper-X.Y.Z.tar.gz zookeeper-X.Y.Z
```

4. Generate the checksums for the release artifact. See this page for latest Apache guidelines.

```
shasum -a 256 zookeeper-X.Y.Z.tar.gz > zookeeper-X.Y.Z.tar.gz.sha256
shasum -a 512 zookeeper-X.Y.Z.tar.gz > zookeeper-X.Y.Z.tar.gz.sha512
```

5. Sign the release

```
gpg --armor --output zookeeper-X.Y.Z.tar.gz.asc --detach-sig zookeeper-X.Y.Z.tar.gz
```

6. Check that release file looks ok - e.g. install it and run examples from tutorial.
    a. untar the release artifact in a test directory
7. 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.

```
sftp home.apache.org
> cd public_html
> mkdir zookeeper-X.Y.Z-candidate-0
> cd zookeeper-X.Y.Z-candidate-0
> put zookeeper-X.Y.Z.tar.gz*
> bye
```

8. Stage the release artifacts to the Apache Nexus repository
    a. Create ~/.m2/settings.xml introducing the values that are specific to you:

```
<settings>
  <servers>
     <server>
       <id>apache.snapshots.https</id>
       <username>XXXXXXX</username>
       <password>XXXXXXXX</password>
     </server>
     <server>
       <id>apache.staging.https</id>
       <username>XXXXXXX</username>
       <password>XXXXXXX</password>
     </server>
   </servers>
   <profiles>
     <profile>
       <id>gpg</id>
       <properties>
         <gpg.passphrase>XXXXXXX</gpg.passphrase>
       </properties>
     </profile>
   </profiles>
</settings>
```

    See http://maven.apache.org/guides/mini/guide-encryption.html for details on encrypting the passwords rather than leaving them in the clear.
    b. Run the following on the branch check out

```
ant mvn-deploy -Dstaging=true
```

    c. Enter Apache Nexus and do the following:
        i. Click on Log In in the upper right corner. Log in using your apache user name and password.
        ii. In the left navigation pane, select Staging Repositories.
        iii. 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.)
        iv. Click the Close button above the Repository names. This makes your release candidate available at the Staging level.
        v. 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.
        vi. 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.
9. 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 3.4.6 candidate 0

This is a bugfix release candidate for 3.4.6. It fixes 117 issues, including issues that
affect leader election, Zab, and SASL authentication.

The full release notes is available at:

https://issues.apache.org/jira/secure/ReleaseNote.jspa?projectId=12310801&version=12323310

*** Please download, test and vote by March 9th 2014, 23:59 UTC+0. ***

Source files:
http://people.apache.org/~fpj/zookeeper-3.4.6-candidate-0/

Maven staging repo:
https://repository.apache.org/content/groups/staging/org/apache/zookeeper/zookeeper/3.4.6/

The release candidate tag in git to be voted upon: release-X.Y.Z-rcR

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

Should we release this candidate?
```

Publishing

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

1. Tag the release. Do it from the release branch and push the created tag to the remote repository:

```
# create a signed tag
git tag -s release-X.Y.Z -m "ZooKeeper X.Y.Z release."

# push the newly created release tag to the remote repo.
git push <remote> release-X.Y.Z
```

2. Check in release files to the distribution server
3.
```
svn co https://dist.apache.org/repos/dist/release/zookeeper/ zookeeper_dist
cd zookeeper_dist
mkdir zookeeper-X.Y.Z
# copy tgz/asc/checksum files from your public_html to zookeeper-X.Y.Z
svn add zookeeper-X.Y.Z
svn ci -m "Add ZooKeeper X.Y.Z release"
```

    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.
4. 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.

```
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'
```

5. 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.
6. Wait 24 hours for release to propagate to mirrors.
7. 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
```

8. Copy the new release documentation into the _released_docs directory

```
cd _released_docs
mkdir rX.Y.Z
cd rX.Y.Z
tar xvf zookeeper-X.Y.Z.tar.gz 'zookeeper-X.Y.Z/zookeeper-docs'
mv zookeeper-X.Y.Z/zookeeper-docs/* .
rm -r zookeeper-X.Y.Z
# Update the "current" doc pointer if necessary (next 2 lines)
cd ..
rm current
ln -s rX.Y.Z current
git add rX.Y.Z current

cd ..
```

9. Update the release news in **releases.md**
10. Update the nav panel in **html/header.html** and **documentation.md** to include X.Y.Z
11. Stage changes to git

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

12. 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
```

13. 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
14. 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:
http://zookeeper.apache.org/releases.html

ZooKeeper X.Y.Z Release Notes are at:
http://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
```

15. In Jira, ensure that only issues in the "Fixed" state have a "Fix Version" set to release X.Y.Z.
16. 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.
17. In Jira, close issues resolved in the release. Disable mail notifications for this bulk change.

Smoke Tests

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

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