Log4j 2 Release Process

Before beginning your first release, take the following preparatory steps:

1. Make sure you have Java 8 as the default JDK.
2. Run `mvn clean install` on the project to make sure it will build cleanly.
3. Run `mvn -P rat -DskipTests clean verify` to verify all the licenses are valid.
4. Login to https://repository.apache.org using your ASF credentials. Click on your username in the upper-right-hand corner and click Profile. Make sure your email address is correct. If not, contact Infra to get it corrected. You will be unable to get the necessary notifications until it's corrected.
5. Make sure you have a 4096-bit RSA PGP key pair for code signing. The public key should be published in a public repository, preferably http://keyserver.ubuntu.com. Also make sure the public key is published to https://www.apache.org/dist/logging/KEYS.
6. Configure this key as your git signing key for release tags. `git config user.signingkey 0x1234567812345678`
7. Make sure your `user.name` and `user.email` git config values match the name and email of this GPG key.

    Edit ~/.m2/settings.xml to add the corresponding private key and its passphrase to your settings. Also add the Apache Subversion server credentials and Sonatype repository credentials to your settings (this is your ASF credentials). You'll want to blank out the passphrase after each release and fill it back in prior to each release so that it isn't compromised. Alternatively, you can encrypt your passwords with a master password.

    ```xml
    <profiles>
    <profile>
    <id>apache-release</id>
    <properties>
    <gpg.keyname>ABC123FC</gpg.keyname>
    <gpg.passphrase>thisIsMyPassphrase</gpg.passphrase>
    <gpg.executable>gpg2</gpg.executable><!-- Use this if executable differs from "gpg" -->
    </properties>
    </profile>
    </profiles>
    <activeProfiles>
    <activeProfile>apache-release</activeProfile>
    </activeProfiles>
    ...
    <servers>
    <server>
    <id>svn.apache.org</id>
    <username>myUsername</username>
    <password>myPassword</password>
    </server>
    <server>
    <id>apache.releases.https</id>
    <username>myUsername</username>
    <password>myPassword</password>
    </server>
    <server>
    <id>apache.snapshots.https</id>
    <username>myUsername</username>
    <password>myPassword</password>
    </server>
    </servers>
    ```

Once you've done this, take the following steps to perform a release:

1. Edit pom.xml and change the Log4j ReleaseVersion property to the version you are releasing.
3. Update src/changes/announcement.vm
4. Run `mvn -P release-notes generate-resources` to create the release notes.
5. Update src/changes/changes.xml with the release date.
6. Run `git add, git commit, and git push` to commit the RELEASE-NOTES.md file that was just generated along with the other files that were modified.
7. Verify whether the bug with the Maven DOAP plugin still exists. If it does configure the build computer to use Oracle Java 1.8.0_202-b08.
8. Run `mvn site` followed by

    ```
    mvn site:stage -DstagingDirectory=$HOME/log4j
    ```

    where $HOME is your home directory and verify that the site looks good.
9. Run

to start the release. Enter the password to your signing key when prompted.

10. Login to [http://repository.apache.org](http://repository.apache.org) using your ASF credentials. Select "Staging Repositories" then check the org.apache.logging repository and close it.

11. Check out the release tag that was created via the Maven release plugin using `git checkout tags/tagname`.

12. Run `mvn site` in the tag directory. When that completes run `mvn site:stage -DstagingDirectory=$HOME/log4j` where $HOME is your home directory.


14. Create the preview web site:
   b. Create a directory matching the target release version. Unzip the site zip file into that directory. Unlink the 2.x symlink and link it to the newly created directory.
   c. Commit and push the web site preview.

   a. Generate the sha512 checksum of the bin and source archive files using

   ```
   sha512sum apache-log4j-2.x.x-xxx.zip > apache-log4j-2.x.x-xxx.zip.sha512
   ``

   or

   ```
   gpg --print-md SHA512 apache-log4j-2.x.x-xxx.zip > apache-log4j-2.x.x-xxx.zip.sha512
   ```

16. Generate the release vote email:
   a. Create the email addressed to dev@logging.apache.org. Sending the email to the PMC is not necessary.
   b. Copy the changes in the release from `RELEASE-NOTES.txt` into the email.
   c. Provide a link to the tag, web site on [http://people.apache.org](http://people.apache.org), and the artifacts in the Nexus repository.
   d. Provide the command to download all the artifacts:

   ```
   wget -e robots=off --cut-dirs=3 -r -p --no-check-certificate $LINK
   ```

   where $LINK is the URL to the repository you just closed (plus the org/apache/logging/log4j/ path appended).

17. If the release vote fails proceed as described in the section below, otherwise if it passes:
   a. Create a new (immutable) tag named `rel/n.n.n` from the log4j-n.n.n-rcn tag by changing to the changing to the directory containing the source for the release and performing

   ```
   git tag [--local-user userId] -s rel/n.n.n -m "Release n.n.n of Log4j"
   ```

   where --local-user is optional, followed by

   ```
   git push --tags
   ```

18. The following steps must be performed by a PMC member:
   b. Create the release directory under the log4j directory.
   c. Move all the distribution artifacts from the distribution dev location to that directory.
   d. Perform an `svn add` of the release directory.
   e. Commit the release to Subversion.
   f. Login to [reporter.apache.org](https://reporter.apache.org) and add the release version and date.

19. The following steps can then be taken by whomever started the release:
   a. Delete the core-its project from the Nexus repository.
   b. Release the remaining artifacts in the Nexus Repository.

20. Wait 12-24 hours (or after the distribution artifacts have been propagated to the mirrors and the Maven artifacts have been propagated to the Central Repository), and then the following steps must be performed by a PMC member:
   a. Perform `svn delete` on the previous release directory for the same version under the log4j directory ([https://dist.apache.org/repos/dist/release/logging/log4j](https://dist.apache.org/repos/dist/release/logging/log4j)).
   b. Commit the delete to Subversion.
   c. See [ManagingTheWebSite](https://dist.apache.org/repos/dist/dev/logging/log4j/) regarding sub-projects.
20. In your log4j web site repo checkout the asf-site branch. Perform `git checkout asf-site` then `git rebase asf-staging` and finally `git push`. Wait 5 to 10 minutes and verify the web site is live.

21. After the website is updated, send the release announcement email. This should be sent out to dev@logging.apache.org, log4j-user@logging.apache.org, private@logging.apache.org, and general@logging.apache.org.

22. Send the release announcement to announce@apache.org using your apache.org email address. This can be combined with the previous step as a Cc field as long as the To field is the dev@ list.

If the release fails before sending the vote email:

1. Login to https://repository.apache.org using your ASF credentials. Select "Staging Repositories" then check the org.apache.logging repository and drop it.
2. Revert any changes that have not been committed.
3. Restart the release process as the same release candidate.

If the release fails after sending the vote email:

1. Login to https://repository.apache.org using your ASF credentials. Select "Staging Repositories" then check the org.apache.logging repository and drop it.
2. Rename the release tag in Git to add rcn to the end of the tag.
3. Restart the release process as a new release candidate.