Community help to verify release candidates

This page explains how the community can help verify a release when a release candidate email arrives on the dev mailing list:

> *This is a vote to release Log4j x.y.z.*
>
> *Please download, test, and cast your votes on the log4j developers list.*
> *[] +1, release the artifacts*
> *[] -1, don't release because...*
>
> *The vote will remain open for as short amount as time as required to vet the release. All votes are welcome and we encourage everyone to test the release, but only Logging PMC votes are "officially" counted. As always, at least 3 +1 votes and more positive than negative votes are required.*
>
> *Changes in this release include:*
>
> *...*
>
> *Tag:*
> *a) for a new copy do "git clone https://github.com/apache/logging-log4j2.git" and then "git checkout tags/log4j-x.y.z-rc1" or just "git clone -b log4j-x.y.z-rc1 https://github.com/apache/logging-log4j2.git"*
> *b) for an existing working copy to "git pull" and then "git checkout tags/log4j-x.y.z-rc1"*

mvn clean install

Running "mvn clean install" on a system that defaults to Java 8 (java -version) you'll quickly receive this error:

```
[INFO] --- maven-toolchains-plugin:3.0.0:toolchain (default) @ log4j-api-java9 ---
[INFO] Required toolchain: jdk [ version='[9, )' ]
[ERROR] No toolchain found for type jdk
[ERROR] Cannot find matching toolchain definitions for the following toolchain types: jdk [ version='[9, )' ]
```

version='[9,)' means the build process needs access to JDK >=9. You don't need to install JDK 9. Its possible to register a JDK version that satisfies the toolchain requirement, i.e. JDK 11, without having to make changes to machine's default JDK by using Toolchains. Up to release 2.18.0 log4j shipped with several toolchain files: toolchains-sample-linux.xml, toolchains-sample-mac.xml, toolchains-sample-win.xml. In release 2.19.0 (LOG4J2-3573, LOG4J2-3520) those have files have been condensed/moved to a single file (./workflows/maven-toolchains.xml) that now relies on two environment variables:

./workflows/maven-toolchains.xml

```xml
<toolchains>
  <toolchain>
    <type>jdk</type>
    <provides>
      <version>1.8</version>
    </provides>
    <configuration>
      <jdkHome>${env.JAVA_HOME_8_X64}</jdkHome>
    </configuration>
  </toolchain>
  <toolchain>
    <type>jdk</type>
    <provides>
      <version>11</version>
    </provides>
    <configuration>
      <jdkHome>${env.JAVA_HOME_11_X64}</jdkHome>
    </configuration>
  </toolchain>
</toolchains>
```

Now Maven can be invoked using "mvn clean install -t ./workflows/maven-toolchains.xml".

mvn apache-rat:check -DskipTests

This command runs the Release Audit Tool to verify files have the necessary license header. Sometimes a unit test flakes out (doesn't wait long enough to delete a temporary file) so the RAT check will fail. Try running the test suite again or deleting the temp file then RAT should succeed.

An alternative command is "mvn -P rat -DskipTests clean verify".

Verify content on staging website

A release manager will publish content from https://github.com/apache/logging-log4j-site/tree/asf-staging to https://logging.staged.apache.org/log4j. Verify the staged website is correct.

"mvn site -DskipTests" will build the website locally based on Markdown files in /src/site. Its the slowest verification and can take over an hour on modern 2020/2021 hardware. It takes a long time because it generates documentation for all ~34 Log4j2 modules (jpa, docker, couchdb, etc). Managing the Logging Services Web Sites describes a technique for quickly generating the core files (articles.md, index.md.vm, security.md, etc) without all module HTML.

TODO: generating binary artifacts (tar.gz, bin.zip, src.tar.gz, src.zip) locally

Coming soon...

Verify signatures and file hashes

https://apache.org/info/verification.html