Release_HOWTO

Introduction

This page is prepared and maintained for/by Nutch committers. You need committer rights to create a new Nutch release.

Release HOW TO Maintenance

Before you begin...

It has become tradition for the Release Manager (RM) to take due care to ensure that this release management HOW TO us kept accurate.

If you fulfill the role of Release Manager, please take the time to carefully review, update, remove, or edit this documentation such that it remains a canonical, reproducible resource for the Nutch community.

Thank you 🙂

Prerequisites

- Apache Maven (most recent stable release)
- Apache Ant (most recent stable release)
- **apache-release** profile present in your local **~/.m2/settings.xml.** An example can be found below **N.B.** It is important that you pay close attention to the contents of this file. Numerous issues now exist with the ant-maven-task plugin so a couple of workarounds exist in this guide.

~/.m2/settings.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<settings
  xsi:schemaLocation='http://maven.apache.org/SETTINGS/1.0.0 http://maven.apache.org/xsd/settings-1.0.0.xsd'
  xmlns='http://maven.apache.org/SETTINGS/1.0.0'
  xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'>
  <servers>
    <server>
      <id>apache.snapshots.https</id>
      <username>${username}</username>
      <password>${password}</password>
    </server>
    <server>
      <id>apache.staging.https</id>
      <username>${username}</username>
      <password>${password}</password>
    </server>
    <server>
      <id>apache.releases.https</id>
      <username>${username}</username>
      <password>${password}</password>
    </server>
    <server>
      <id>gpg.passphrase</id>
      <passphrase>${password}</passphrase>
    </server>
    <server>
      <username>${username}</username>
      <password>${password}</password>
      <id>central</id>
    </server>
    <server>
      <username>${username}</username>
      <password>${password}</password>
      <id>snapshots</id>
    </server>
  </servers>

  <mirrors>
    <mirror>
      <id>central.mirror</id>
      <url>https://repo1.maven.org/maven2/</url>
      <mirrorOf>central</mirrorOf>
    </mirror>
  </mirrors>

  <profiles>
    <profile>
      <id>apache-release</id>
      <activation>
        <activeByDefault>true</activeByDefault>
      </activation>
      <properties>
        <mavenExecutorId>forked-path</mavenExecutorId>
        <gpg.executable>gpg</gpg.executable>
        <gpg.keyname>${keyname}</gpg.keyname>
        <!--gpg.passphrase>${password}</gpg.passphrase-->
      </properties>
    </profile>
  </profiles>

  <activeProfiles>
    <activeProfile>apache-release</activeProfile>
  </activeProfiles>

</settings>
```

Prepping the Release Candidate

1. Create a new release in JIRA. If you do not already have these privileges ask your PMC Chair.
2. Push off all open issues to the next release; any critical or blocker issues should be resolved on mailing list. Discuss any issues that you are unsure of on the mailing list.
3. Create a branch branch-x.x, from now on, use the branch created above.
4. Update version numbers (from X.Y-SNAPSHOT to X.Y) for release in:
    - conf/nutch-default.xml - **http.agent.version** property
    - default.properties - **version** property and **year** property
    - src/bin/nutch - **version number** in echo "nutch X.Y"
5. in default.properties update the links to Javadocs of important dependencies (eg. Hadoop)
6. Update CHANGES.md with release date and (if needed) add additional changelog entries (from Jira Report). It's also good practice to include a link to the Jira report.
7. Check if documentation needs an update. Although this may be a huge task at any given time, any minor contribution is better than nothing at all.
8. Commit all these changes to the branch you are releasing.
9. To avoid that "forgotten" files in your development environment are packaged, make a **clean** checkout for the release branch or tag:

    ```
    cd ...
    git clone --branch branch-x.x https://github.com/apache/nutch.git branch-x.x
    cd branch-x.x
    ```

10. Run unit tests.
    ant test
11. Do basic test to see if release looks ok - e.g. install it and run example from tutorial.
12. Run the docker container
13. Get hold of the most recent version of maven-ant-tasks-2.X.X.jar and put it in the **$NUTCH_HOME/ivy** directory
14. Set the **${MVN_HOME}** environment variable on the machine you are executing the release from. This must point to the Apache Maven home directory.
15. Ensure you have an **apache-release** profile in your local **~/.m2/settings.xml** as detailed in the prerequisites above
16. Set the **${MAVEN_GPG_PASSPHRASE}** environment variable on the machine you are executing the release from. This must be the GPG passphrase associated with the key you are using to sign the release artifacts.
17. Execute **ant -lib ivy deploy** from **$NUTCH_HOME**, this will sign the Maven artifacts (sources, javadoc, .jar) and send them to a Apache Nexus staging repository. Details of how to set this up can be found here. **N.B.** Ensure that you have an **apache-release** profile contained within ~/.m2 /settings.xml
18. Once you've read, and are happy with the staging repository, close it.
19. Remove the following artifacts
    - **$NUTCH_HOME/ivy/maven-ant-tasks-2.X.X.jar**
    - **$NUTCH_HOME/pom.xml**
    - **$NUTCH_HOME/pom.xml.asc**
    - **$NUTCH_HOME/target**
20. Tag the release candidate.
    git tag -a release-X -m "Apache Nutch X RC#X Tag"
21. Push it to the remote host.
    git push origin release-X
22. Run the packaging tasks. The generated artifacts can be found in **$NUTCH_HOME/dist**. If you experience issues during this stage you may need to prune/delete **~/.ivy2/cache/***

    a. **ant zip-bin && ant tar-bin && ant zip-src && ant tar-src**
23. Check out the release management area, create a new directory for the release and copy CHANGES.md there

    a. **svn co https://dist.apache.org/repos/dist/dev/nutch/ nutch_staging**
    b. **mkdir nutch_staging/${release_version}**
    c. **cp CHANGES.md nutch_staging/${release_version}**
24. Sign all of the generated artifacts - Step-By-Step Guide to Signing Releases ' - Consider using Chris Mattmann's Apache Utility Scripts. After the signing each release package must be accompanied by the *.asc and the *.sha512 signature file.
25. Copy all of the release artifacts to the staging directory and commit to the SVN server
    a. **cp $NUTCH_HOME/dist/*.tar.gz* $NUTCH_HOME/dist/*.zip* nutch_staging/${release_version}/**
    b. **svn add nutch_staging/${release_version}**
    c. **svn ci -m "Stage Apache Nutch ${release_version} RC#1"**
26. Make sure your pgp key is listed in the Nutch KEYS file located at http://www.apache.org/dist/nutch/KEYS or better yet, use https://id.apache.org/ and add your PGP there, and it will then appear in Apache Nutch's Automatically Generated Keys
27. Create and open a VOTE thread on user@ and dev@nutch.apache.org. The VOTE must pass with 3 +1 binding VOTE's before any release can take place. A VOTE thread usually takes the form

> *title = [VOTE] Apache Nutch X.XX Release*
>
> *Hi user@, dev@nutch,*
>
> *A first candidate for the Nutch X.XX release is available at [0] where accompanying SHA512 and ASC signatures can also be found.*
>
> *Information on verifying releases can be found at [1].*

> *The release candidate comprises a .zip and tar.gz archive of the sources in [2] and complementary binary distributions. In addition, a staged maven repository is available at [3].*
>
> *The X.XX release report is available at ${Jira release report URL included in CHANGES.md} [4]*
>
> *Please vote on releasing this package as Apache Nutch X.XX. The vote is open for at least the next 72 hours and passes if a majority of at least three +1 Nutch PMC votes are cast.*
>
> *[ ] +1 Release this package as Apache Nutch X.XX.*
>
> *[ ] -1 Do not release this package because… Cheers, ${yourname} P.S. Here is my +1.*
>
> *[0] https://dist.apache.org/repos/dist/dev/nutch/X.XX/*
>
> *[1] https://nutch.apache.org/download/#verify-releases*
>
> *[2] https://git-wip-us.apache.org/repos/asf?p=nutch.git;a=tag;h=a8ef2997d14eb7af95dcafee379d54b31f89dd1a*
>
> *[3] https://repository.apache.org/content/repositories/orgapachenutch-XXXX*
>
> *[4] ...*

Once the 72 hour period expires it is time to close the VOTE thread on user@ and dev@ with a RESULT thread. This should simply state the outcome of VOTE'ing (including how many binding VOTE's were received. Finally it should included whether the VOTE passed and if the released can be made.

In the instance where the VOTE does not pass, the release manager should roll back all of the work above as well as **DROP** the staging artifacts.

Making the release

1. Return to the NEXUS staging repository. Login, and **RELEASE** the Nutch repository.
2. Move the artifacts from the release management area to the release area as follows:

```
svn mv https://dist.apache.org/repos/dist/dev/nutch/$release.version https://dist.apache.org/repos/dist
/release/nutch/$release.version --message "Release Apache Nutch $release.version"
```

3. wait until the release is visible on the release page
4. update the Nutch website (see nutch-site README how to)
    a. add the new release info to the doap.rdf file (content/doap.rdf), and double check for any other updates that should be made to the doap file as well if it hasn't been updated in a while. If this is the case please see here
    b. replace the folder content/documentation/javadoc/apidocs/ with the Java API doc folder of the current version
    c. update the version numbers on the javadoc page (content/documentation/javadoc/index.md) and the wiki FrontPage
    d. update the links to release packages on the downloads page (content/download.md)
    e. write a short announcement about the new release in the news section (content/news/)
    f. build the Nutch website, verify the changes and deploy the site
5. Release the version on Jira
6. (if necessary) prepare announcements regarding CVEs resolved with this release of Nutch, see the Apache security howtos
7. Send announcements to the user and developer lists as well as announce@apache.org
8. Finally, don't forget to remove the previous release from the dist.apache.org release directory to reduce the load on ASF mirrors, see when-to-archive and NUTCH-1742.

Preparing for new development

1. Update version numbers to **A.B-SNAPSHOT** (assuming A.B is next release number) in:
    - conf/nutch-default.xml - **http.agent.version** property
    - default.properties - **version** property and **year** property
    - src/bin/nutch - **version number** in echo "nutch X.Y"
2. Update CHANGES.md with header for new changes.
3. Ensure that a new version in JIRA exists for development snapshots (A.B). If this is not there then create one.