Apache NetBeans Release README

- Overview
- Requirements
- Branching, builds and release candidate overview.
- Preparing the Code Base and Jenkins TLP Job
- Taming JSON to Prepare Beta or Voting Candidates
    - Preparing for beta:
    - Preparing for voting candidate:
- Producing a Release Candidate
    - 1. Obtaining and preparing the files
        - 1.1 On Apache Jenkins
        - 1.2 On your computer once the build described in 1.1 above has succeeded:
        - 1.3 On the NetBeans virtual machine
    - 2. Notifications to mailing list
    - 3. Workflow
- Producing a Voting Candidate
        - 1. Obtaining and Preparing the Voting Candidate Files
            - 1.1 On Apache Jenkins:
            - 1.2 On your computer, once the build described in 1.1 above has succeeded:
        - 2. Verifying the Release by Checklist
        - 3. Publishing them in the staging area
            - Create an empty directory for the release then:
            - Publish to Staging Area
        - 4. Do the vote on the dev mailing list.
        - 5. Creating tag for the Release:
        - 6. Releasing a Release
        - 7. Updating redirect for NetBeans Distribution Update Center
        - 8. Post Release Step
            - 8.1 Git operation post release create PR for snapshot (example for 12.5)
            - 8.2 Git operation post release merge delivery to master
    - Specific Steps, Details, and Examples

Overview

Every PMC member should read the below docs at least once, several times, because the Apache NetBeans PMC members are responsible to verify and ensure releases are done in compliance with these rules:

1. See the generic Apache Release process – http://www.apache.org/dev/release-publishing.html.
2. See the generic Apache Release FAQ – http://www.apache.org/legal/release-policy.html.
3. See the ASF policy and documentation for releases: http://www.apache.org/dev/#releases.
4. See the necessary steps and requirements for the release distribution -- http://www.apache.org/dev/release-distribution.html.

Requirements

1. You must be a **PMC member** in Apache NetBeans.
2. You must have **connected** your Apache ID **to Apache NetBeans GitHub** via GitBox: https://gitbox.apache.org.
3. You must have access to https://github.com/apache/netbeans and https://github.com/apache/netbeans-jenkins-lib/.
4. You must be able to connect to Apache CI Jenkins instance builds, i.e., see **INFRA-17082** - Getting issue details... STATUS .
5. You must have a PGP KEY (http://www.apache.org/dev/release-distribution.html#sigs-and-sums) for which the public key must be here: https://dist.apache.org/repos/dist/release/netbeans/KEYS.
6. You need to have sudo access to netbeans-vm1.apache.org (Eric Barboni Neil C Smith Laszlo Kishalmi has this, at least).

Branching, builds and release candidate overview.

Two git branches are created at feature freeze for each quarterly NetBeans release.

- The **release** branch (eg. `release140`) is used for release candidate and release builds. Naming is important as it is picked up by the build system (the last zero is a legacy of the old version scheme).
- The **delivery** branch is used for collating fixes for the release.

All pull requests intended for the release are merged to delivery, and delivery is merged to both the release branch and master branches regularly. The delivery branch is transitory and should be deleted shortly after the final release.

The release team handle all merges to delivery, and syncing to other branches. This is mainly for scheduling reasons, as well as to keep an overview of fixes / check for merge issues. A wrongly timed (eg. between syncs) or unwanted (eg. spec version changes) commit merged to delivery will likely lead to a forced reset. The release team has no more say in what actually gets merged to a release than any other committer.

Syncing from delivery to master and release branches is done via pull requests. This has a number of benefits, including running CI against both targets with every merge to delivery, easy view of all changes between release candidates, and ensuring release hashes are unique to the release branch.

It is useful for all committers if we can use the same day of the week for release candidates and syncing. This has often been a Wednesday, but is up to the member(s) of the release team managing this for any release - publicise it!

Before freezing:

1. Before freezing, the plugin portal for the upcoming release should exist and dev should point to it. Maybe move everything from the previous to the new release. When we have our first release candidate, we should have plugins already in the plugin portal, to test the plugin functionality for the upcoming release. Login to https://plugins.netbeans.apache.org and create a new version NetBeans versions
2. Announce the feature freeze date on dev@ using [NOTICE] emails a week and a day before branching following up in the same thread - see / adapt example.

After freezing, on the freeze date or shortly thereafter:

1. Change the current milestone on all remaining open pull request to the next milestone (use bulk edit in the PR table).
2. Create the delivery and release branches from master. Announce feature freeze and branching - see / adapt example.
3. At the time of branching, update plugin center and update center redirects https://github.com/apache/netbeans-antora/blob/main/supplemental-ui/.htaccess.
4. Clone the update center on the NetBeans VM, see also Update centre .htaccess.
5. Increment module spec versions on master for the next release using `ant increment-spec-versions`. **This should be the first commit in master after branching**. See PR after NB14 branch.
6. Add the metadata about the release to `netbeansrelease.json`, including a milestone for `rc1` using the hash from the release branch (eg. `git show`). This will be the last hash shared with any other branch.
7. Trigger a build for the release branch in the Jenkins NetBeansTLP job. A build may have started automatically but will likely not have picked up the latest metadata - if so, stop and restart. Build with parameters – 'installers', 'vscode plugin', and 'push to nightly'. For every release candidate, each of these need to be selected, generating multiple binaries ('installers' will generate three, two Linux, deb and rpm, and one Windows, because Mac installer has to be generated on a Mac).
8. Once the build is complete, download the source and binary zip artefacts, and check build, fixes, etc. locally. **Lock the Jenkins build** to stop it being cleaned up, and announce the release candidate with links to the artefacts - see / adapt example.
9. Add a pinned GitHub discussion with links to the release candidate: https://github.com/apache/netbeans/discussions/6581, share it on social networks beyond the dev list.
10. Modify the https://github.com/apache/netbeans/blob/master/.github/ISSUE_TEMPLATE/netbeans_bug_report.yml to have XXX Release Candidate in section label: Apache NetBeans version and XXX-1 in section Did this work correctly in an earlier version?

After the build is complete (weekly, bringing fixes into the delivery branch for the release, syncing them, triggering weekly release candidates):

1. Over the following week, review and merge any inbound fixes to `delivery`.
2. **As soon as there is any change in delivery**, open two pull requests to sync branches - *Sync delivery to releaseXXX for XX-rcN* (see example) and *Sync delivery to master after XX-rcN* (see example) Use label release process.
3. If there are no merged fixes and no open major / critical issues for this release, consider whether it's time to move to a vote. See also the criteria described at: Pull requests for delivery.
4. If / when ready to trigger another release candidate, check and merge the sync PR to `releaseXXX`. Add a milestone based on the git hash of the sync PR merge commit to `netbeansrelease.json`.
5. Repeat from step 1, making sure to merge the other sync PR to master **after** verifying and announcing the release candidate. Do not sync to master if there are problems with the release candidate build - fix in `delivery` and resync / rebuild.

Moving towards a release vote:

1. The release voting candidate **must** be built off the same git hash as the last release candidate - edit the existing milestone in `netbeansrelease.json`.
2. Instead of adding another hash, change what was the last release candidate to have, with version "vote:1", etc, see https://github.com/apache/netbeans-jenkins-lib/blob/master/meta/netbeansrelease.json#L1074
3. Do the build again, this time, only selecting parameters 'installers' and 'vscode'. Release build should never be pushed to nightlies.

Preparing the Code Base and Jenkins TLP Job

1. Create a branch **release<version>** in https://github.com/apache/netbeans. <version> looks like 113 for Apache NetBeans 11.3 full name of branch is release113
2. Create a milestone on github for next iteration. (to help triage)
3. Create a section for **release<version>** in https://github.com/apache/netbeans-jenkins-lib/blob/master/meta/netbeansrelease.json (after the copy paste, read carefully to match date, change version, change position.
4. Check on Jenkins that item appears for **release<version>** in https://ci-builds.apache.org/job/Netbeans/job/netbeans-TLP/ (all previous branches should be present).
5. On freeze date create delivery branch to allow PR that will be resynch with release<version>
6. Normal development will continue on master but before merging normal PR do a PR to increment spec version
   `ant increment-spec-versions`

Taming JSON to Prepare Beta or Voting Candidates

https://github.com/apache/netbeans-jenkins-lib/blob/master/meta/netbeansrelease.json

For each **release** section you will have **milestones** section. This section will help branding a special Apache NetBeans milestone.

If nothing is set in the section, it will release a dev version of current branch, e.g., 11.3-dev.

Preparing for beta:

Milestone section key are commit hash. If you want a commit hash to form a beta1 you need to add milestone entry (do not forget position):

"<commit hash>":{"version": "beta1","position": "1"}

Branding will contain beta information.

Preparing for voting candidate:

Milestone section key are commit hash. If you want a commit hash to form a beta1 you need to add milestone entry (do not forget position):

"<commit hash>":{"vote": "1","position": "2"}

Branding will contain release information.

Producing a Release Candidate

1. Obtaining and preparing the files

1.1 On Apache Jenkins

1. You have to set up json file milestone content.
2. Go to https://ci-builds.apache.org/job/Netbeans/job/netbeans-TLP/
3. Check that the build on **release<version>** and verify artefacts they should be branded with beta in their name. Cancel and relaunch build otherwise
4. Wait for the job to be finished.
5. Keep the build

1.2 On your computer once the build described in 1.1 above has succeeded:

1. Download and extract the build artifacts.
2. Check the SHA512 checksums:

```
find . -name '*.sha512' -execdir sha512sum --check '{}' \;
#alternative
for z in $(find . -name '*.sha512'); do cd $(dirname $z); sha512sum ./$(basename $z) --check --quiet; cd
- >/dev/null; done
```

1.3 On the NetBeans virtual machine

1. Prepare apidoc folder according to json file:

```
sudo su -
cd /var/www/bits.netbeans.org
mkdir **version**
```

2. Notifications to mailing list

1. Send a mail to netcat and dev mailing lists.

Notice email

```
<SUBJECT>
[NOTICE] Apache NetBeans XX release candidate YY available for testing

<BODY>
Hi,
 the nth Release candidate for NetBeans XX is ready.

The NetBeans XX-rcYY artifacts are here:
https://ci-builds.apache.org/job/Netbeans/job/netbeans-TLP/job/netbeans/job/XX/BNUM/artifact/dist/

You will found linux/windows installer and vsix file too

Link to the binary zip:
https://ci-builds.apache.org/job/Netbeans/job/netbeans-TLP/job/netbeans/job/XX/BNUM/artifact/dist/netbeans
/netbeans-XX-rcYY-bin.zip

SHA512:
<INSERT SHA512>

The sources are here:
https://ci-builds.apache.org/job/Netbeans/job/netbeans-TLP/job/netbeans/job/XX/BNUM/artifact/dist/netbeans
/netbeans-XX-rcYY-source.zip

SHA512:
<INSERT SHA512>

If you're a committer adding an issue, or helping triage an issue (please do!), add the milestone and/or
priority labels as appropriate.
Use priority:high for should be fixed before release, priority:critical for must be fixed before we can release.

**The following rules are applied to pull requests from now until release:**

PR's intended to be included in the XX release :
 - Limited to fixes (no need for a ticket!)
 - Base on the delivery branch.
 - Mark with NBXX milestone (we'll monitor - no need to add us all as reviewers!).
 - Will be merged by the release team.
 - Will be assessed against bug priorities - please use the priority:high and priority:critical labels here too.

PR's with features for NBXX+1 :
 - Base on the master branch.
 - Will be reviewed and merged in the usual way.
 - If possible stay away from big refactoring.
 - If possible do not overlap with fixes for XX (delivery will be merged to master weekly).


Thank you for your contributions!

Best regards,
RM
```

3. Workflow

1. Repeat Release Candidate until consensus is reached
2. If consensus to release is reached you can produce a Voting Candidate

Producing a Voting Candidate

Producing a voting is close to producing a release candidate. Branding will change to match release branding. You will have to sign all artefacts

1. Obtaining and Preparing the Voting Candidate Files

1.1 On Apache Jenkins:

1. You have to set up json file milestone content.
2. Go to the https://ci-builds.apache.org/job/Netbeans/job/netbeans-TLP/
3. Check that the build on **release<version>** and verify artefacts they should be branded with like a voting candidate. Cancel and relaunch build otherwise
4. Wait for the job to be finished.
5. Keep the build

1.2 On your computer, once the build described in 1.1 above has succeeded:

1. Download and extract the build artifacts, specifically, "(all files in ZIP)" for "dist/netbeans" and "dist/netbeans-platform". This gets us the IDE, source and binary, NBM modules, and platform source and binary.
2. Check the SHA512 checksums:

```
find . -name '*.sha512' -execdir sha512sum --check '{}' \;
#alternative
for z in $(find . -name '*.sha512'); do cd $(dirname $z); sha512sum ./$(basename $z) --check --quiet; cd
- >/dev/null; done
```

3. Sign the Release Files

```
export GPG_EMAIL="YOU@apache.org"
find . -name '*.zip' -exec gpg -u $GPG_EMAIL --armor --output {}.asc --detach-sign {} \;
# Stop here if nbms are not going to be published
find . -name '*.nbm' -exec gpg -u $GPG_EMAIL --armor --output {}.asc --detach-sign {} \;
find . -name '*.gz' -exec gpg -u $GPG_EMAIL --armor --output {}.asc --detach-sign {} \;
# nbm licenses
find . -name '*.license' -exec gpg -u $GPG_EMAIL --armor --output {}.asc --detach-sign {} \;
# nbm/updates.xml and nbm/tasks.jar
find . -name '*.jar' -exec gpg -u $GPG_EMAIL --armor --output {}.asc --detach-sign {} \;
find . -name '*.xml' -exec gpg -u $GPG_EMAIL --armor --output {}.asc --detach-sign {} \;
```

4. Verify Signatures

```
find . -name '*.asc' -exec gpg --verify {} \;
```

2. Verifying the Release by Checklist

1. Go through the Apache Incubator Release Checklist
   (whether a project is in the Apache Incubator or not, these items need to be checked since they're important for all Apache releases)
2. Check the Product version:
   Start Apache NetBeans and check the **Title** and the **Help > About Dialog**
3. Check that the splash screen has been updated.
4. Check that the user directory matches the release number.
5. Check GPL License is present and need to be accepted:
   a. Start NetBeans with an empty User directory
   b. Open a Java SE project: installing nb-javac shall present GPL
   c. Start NetBeans with an empty User directory
   d. Create a new PHP project: installing Graal JS shall present GPL

3. Publishing them in the staging area

Create an empty directory for the release then:

```
svn checkout --depth immediates https://dist.apache.org/repos/dist/ apache-dist
cd apache-dist/dev && svn update --depth immediates netbeans
cd netbeans && svn update --set-depth infinity
```

Publish to Staging Area

```
svn rm netbeans-platform/* netbeans/*
mkdir netbeans-platform/$VERSION netbeans/$VERSION
# Copy the netbeans-platform and netbeans artifacts with checksums and signatures in place
svn add netbeans-platform/* netbeans/*
svn --username "<your-apache-username>" commit -m "Apache NetBeans $VERSION."
```

Note 1: In the above $VERSION should be, for example, "12.3" (and not 12.3-vc or something similar), because then when we publish later to "release", we won't need to change the name.

Note 2: The above, i.e., pushing to svn, could take several hours.

4. Do the vote on the dev mailing list.

Voting email

```
<SUBJECT>
[VOTE] Release Apache NetBeans XX

<BODY>
This is our first voting candidate for the release of Apache NetBeans XX.

Please note all requirements below for validating sources and convenience binaries before voting.

Apache NetBeans XX constitutes all clusters in the Apache NetBeans Git repository, which together provide the
NetBeans Platform (i.e., the underlying application framework), as well as all the modules that provide the
Java SE, Java EE, PHP, JavaScript and Groovy features of Apache NetBeans.

------------------------

Build artefacts are available here :

https://dist.apache.org/repos/dist/dev/netbeans/netbeans/XX/
https://dist.apache.org/repos/dist/dev/netbeans/netbeans-platform/XX/

They were built by the Jenkins pipeline :

https://ci-builds.apache.org/job/Netbeans/job/netbeans-TLP/job/netbeans/job/XX/BNUM/

------------------------

We are primarily voting on :

https://dist.apache.org/repos/dist/dev/netbeans/netbeans/XX/netbeans-XX-source.zip

SHA512 :
<INSERT SHA512>

KEYS file : https://downloads.apache.org/netbeans/KEYS

------------------------

Associated with the primary source item we have, generated with the pipeline mentioned above :

-- at https://dist.apache.org/repos/dist/dev/netbeans/netbeans/XX/

Binaries associated with the source - netbeans-XX-bin.zip as well as
update content under the nbms folder.

-- at https://dist.apache.org/repos/dist/dev/netbeans/netbeans-platform/XX/

The platform cluster build netbeans-platform-XX-bin.zip and netbeans-platform-XX-source.zip

------------------------

Maven Artefacts
```

```
The Maven artefacts for Apache NetBeans XX are ready on staging
associated to this vote.

https://repository.apache.org/content/repositories/STAGEURLID/

The version is : RELEASEXX

------------------------

Voting Requirements

Before voting +1 you are required to download the signed source code
package, compile it as provided, and test the resulting executable on
your own platform, along with also verifying that the package meets
the requirements of the ASF policy on releases -
http://www.apache.org/legal/release-policy.html#management

In particular, you should (at least) follow these steps.

1. Download the artefact to be voted on and unzip it.
2. Check that the artefact does not contain any jar files (there are
branding folders with the name *.jar).
3. Verify the cryptographic signatures, the NOTICE and LICENSE file
4. Build it using the README provided by the artefact.
5. Look in nbbuild/netbeans for the NetBeans installation created by
the build process and try running it.

In addition to checking the sources, you should check the associated
convenience binary zips, nbms and maven staging at the artefact links
above. As well as checking any artefact functions correctly, you
should check that it has been correctly signed by a PMC member, and
that the source being voted on is sufficient to build the relevant
binary.

Separate votes will be held on other convenience binaries, including
installers. Those will be dependent on this vote passing.

This vote is going to be open at least 72 hours, vote with +1, 0, and
-1 as usual. (Please justify -1)

Please mark your vote with (binding) only if you're an Apache NetBeans
PMC member to help with voting admin.

Only respond if you are going to vote, i.e., this is NOT a discussion
thread.

Apache NetBeans XX will be released if and when this vote passes.

Thank you to all contributors for all your hard work!
Thanks again
Best regards,
```

5. Creating tag for the Release:

Assuming the vote succeeded, we should tag the release, after voting, but not the voting candidates.

Go to your local git repository clone. Collect the git commit hash from the release build job.

```
git fetch --all
git tag -a $VERSION -m "Apache NetBeans $VERSION." $HASH
git push origin $VERSION
# or if you've disabled upstream push
git push https://github.com/apache/netbeans.git $VERSION
```

6. Releasing a Release

Assuming the vote succeeded, move the artefacts from "dev" to "release", which, on success, will automatically trigger mirroring:

```
# Go to the apache-dist directory which had been previously checked out in Step 4.
cd release && svn update --depth immediates netbeans
cd netbeans && svn update --set-depth infinity
# svn rm netbeans-platform/* netbeans/* # need to keep LTS and latest non-LTS ?
# Use svn move or copy to copy the the artifact from the staging area
svn mv ../../dev/netbeans/netbeans-platform/$VERSION netbeans-platform/$VERSION
svn mv ../../dev/netbeans/netbeans/$VERSION netbeans/$VERSION
```

1. Go to the release version directories and rename the voting candidate artifacts to have the release version in their name.
2. Adjust the name change in the checksum files (or recreate them)
3. Doublecheck the checksums with: find . -name '*.sha512' -exec sha512sum -c {} \;
4. Commit the changes from the apache-dist directory

7. Updating redirect for NetBeans Distribution Update Center

Once release is synchronized across all download mirrors it is necessary to update redirect for release modules to final location:

1. Login to NetBeans virtual machine where redirect is configured: ssh <your_apache_id>@netbeans-vm1.apache.org
2. Become root using e.g. OTP MD5 online encryption: sudo /bin/bash

```
sudo /bin/bash
cd /var/www/html/uc/
mkdir 11.1
cd 11.1/
curl https://dist.apache.org/repos/dist/release/netbeans/netbeans/11.1/nbms/updates.xml.gz -o updates.
xml.gz
echo ' RedirectMatch ^/uc/11.1/(.*)(\?.*)?$ http://www.apache.org/dyn/closer.lua?
action=download&filename=netbeans/netbeans/11.1/nbms/$1' > .htaccess
# Update the previous version UC - need to keep LTS and non-LTS!
cd ../10.0/
echo 'RedirectMatch ^/uc/10.0/(.*)(\?.*)?$ https://archive.apache.org/dist/incubator/netbeans/incubating-
netbeans/incubating-10.0/nbms/$1' > .htaccess
vim updates.xml.gz
```

3. Modify updates.xml.gz file of previous release to contain: (how to handle LTS vs non-LTS?)

```
<notification url="https://netbeans.apache.org/download/index.html">Apache NetBeans IDE 11.0 is
available!</notification>
```

   element in its <module_updates> root element.

NOTE: For accessing the NetBeans Virtual VM please see netbeans-vm.apache.org

8. Post Release Step

8.1 Git operation post release create PR for snapshot (example for 12.5)

```
git checkout release124
ant clean
ant build
ant gen-sigtests-release
git add -A
git stash
git checkout master
git checkout -b apis-nb125a1
git stash pop
git add -A
git commit -m "Snapshot of APIs as of 12.5"
# remove getPeer() calls
find . -name "*.sig" -exec sed -i '/java.awt.peer.ComponentPeer/{N;d;}' {} \;
# check git diff
git add -A
git commit --amen
```

8.2 Git operation post release merge delivery to master

(wip)

website

mail to + twitter

--

clean up old release on dist, github tag removal, free jenkins locked build

Specific Steps, Details, and Examples

See these for reference:

NetBeans 12.3 tasks: **NETBEANSINFRA-238** - Getting issue details... STATUS

NetBeans 12.1 tasks: **NETBEANSINFRA-212** - Getting issue details... STATUS

NetBeans 10 tasks: **NETBEANS-1321** - Getting issue details... STATUS