Fedora Release Process

This document is intended to be used and kept up to date by the *Fedora Release Manager*. It details the steps necessary to perform an official release of Fedora.

Before Release Day

Verify release privileges

To make sure release day goes smoothly, you should ensure that:

- You have an account with commit access for the fedora project on github. As a committer, you should already have this level of access.
- You have an account with edit privileges on the duraspace.org Confluence wiki.
- You have an oss.sonatype.org account and have requested to be given permission to publish to the org.fcrepo groupId by adding a comment to the Fedora Sonatype Hosting Ticket
- You have project configuration privileges on JIRA (you'll see an error here if you don't)
- Your maven settings (`~/.m2/settings.xml`) includes the following:

```
<settings>
  ...
  <servers>
    ...
    <server>
      <id>sonatype-nexus-snapshots</id>
      <username>your-sonatype-id</username>
      <password>your-sonatype-pwd</password>
    </server>
    <server>
      <id>sonatype-nexus-staging</id>
      <username>your-sonatype-id</username>
      <password>your-sonatyp-pwd</password>
    </server>
    <server>
      <id>github</id>
      <username>your-github-id</username>
      <password>your-github-pwd</password>
    </server>
  </servers>
  ...
</settings>
```

Note about encrypted passwords

Encrypted passwords work for the plugin that references the sonatype-nexus passwords, but NOT the one that uses github. To avoid a cryptic error, enter your github password in plaintext.

Ensure you have a trusted code signing key

- create one if you haven't before
- ensure that it's listed within the committer keys

Prepare and distribute test plan

The test plan should also be ready prior to code freeze.

It should include:

- Which platform/configuration combinations will be tested
- Which automated tests will be run, and by whom
- Which manual tests will be run, and by whom
- Which service compatibility tests (external search, external triplestore) will be run, and by whom
- Instructions on how testers will report on test results

Create release branches and begin final test phase

Create a release candidate branch, release testing wiki page and notify developers to test.

Variable release number

```
RC_VERSION=4.5.1
```

Using the above variables, complete the below `git` commands for each module being released.   The modules to be released are shown in the Fedora modules release plan.   A new page may need to be added for this release.   When the steps below are complete for a specific module, change the color to green, so that others know what's been done.

NOTE: The value of `RC_VERSION` will vary for each release.

```
git checkout <main -or- maintenance-branch>
git pull
git push origin <main -or- maintenance-branch>:${RC_VERSION}-RC
```

Release candidate branches **CANNOT** have the same *version* property as the branch you are branching from (ie main or n.x-maintenance branch, depending on whether this is a major, minor, or patch release) in the *pom.xml* file. The versions on the original branch will need to be incremented at the same time you create the release branches. You will need to pull in another community member to create a pull request with the version change, or another committer if you are going to create the pull requests yourself.

In this case, you can use the "versions-maven-plugin" to update project version numbers. See versions-maven-plugin documentation. For example:

```
mvn versions:set -DnewVersion=${RC_VERSION}-SNAPSHOT
```

Verify that the previous SNAPSHOT version is no longer anywhere in the code base. Verify that the code base builds. Commit the changes.

Tag the release candidate branch

```
git checkout ${RC_VERSION}-RC
git tag -a "<artifact-id>-${RC_VERSION}-RC-1" -m "<artifact-id>-${RC_VERSION}-RC-1"
git push origin --tags
```

Then we must merge the pull requests to increment the version numbers to the main branch.

Example: During the 4.5.1 release our branches should have the version `4.5.1-SNAPSHOT`, this will be incremented on the main branch to `4.5.2-SNAPSHOT`

Optional - Deploy Snapshot Artifacts

If the release candidate is coming off of a "maintenance" branch instead of main, it is possible that snapshot artifacts have not been deployed to the Sonatype snapshot repository. If this is the case, Travis will fail to build.

You can check if the snapshot artifacts exist by looking for each module. For example:

```
https://oss.sonatype.org/content/repositories/snapshots/org/fcrepo/fcrepo-http-commons/
```

If the snapshot artifacts do not exist, you can deploy them to Sonatype with the following command:

```
mvn -DaltDeploymentRepository=sonatype-nexus-snapshots::default::https://oss.sonatype.org/content/repositories
/snapshots/ deploy -DskipTests
```

Build resources

These build scripts may be of use for building the set of release candidate modules.

Once a release has been created, it should be uploaded to GitHub as a Pre-Release.   A Pre-Release should be created for fcrepo. The name should be "Release Candidate 1 - RC_VERSION"

Update online resources

If any online resources have been modified or added to during the release, these must be updated.

Release Day

Determine which modules will be released.

Create a new tab in the Fedora modules release plan spreadsheet.

Notes:

1. *Follow the release order of the spreadsheet plan*
2. *Some projects need pom.xml dependency version properties to be updated prior to release (e.g. fcrepo-camel-toolbox).*

Variables Used

These variables will be used in the examples that follow. The exact values of `$ORG, $REPO, $CURR and $NEXT` will vary depending on which module and version is being released.

```
ORG=fcrepo
REPO=fcrepo
CURR=6.0.0-alpha-2
NEXT=6.0.0-SNAPSHOT
```

Github Release - part 1

Perform a clean checkout of the code from Github and prepare the release.

```
git clone git@github.com:$ORG/$REPO.git
cd $REPO
git checkout -b ${CURR}-RC origin/${CURR}-RC # or the release branch if named differently
mvn release:clean
```

If release:clean fails, you may need to revert the RC commit with `git revert HEAD`. If the parent snapshot is not available, build an old version of fcrepo to populate it locally.

Resolve dependencies and set main versions to `$CURR` and dev versions to `$NEXT`

```
mvn release:prepare -DreleaseVersion=$CURR -DdevelopmentVersion=$NEXT -DautoVersionSubmodules=true -
DpushChanges=false
```

Your GPG passphrase may not be masked in terminal.

If you have more than one personal key on your GPG keyring, you can specify the correct key by adding

```
-Darguments=-Dgpg.keyname=<Your Key ID>
```

to the above *mvn* command.

Inspect/Verify local updates:

```
git diff HEAD~1
git diff HEAD~2 HEAD~1
```

These diffs should only contain changes of version numbers (from ${CURR}-SNAPSHOT to $CURR or $CURR to $NEXT) or occasionally HEAD to the current tag name ($REPO-$CURR)

Remove your local copies of Fedora artifacts to be sure of a clean build, and build the release.

```
rm -rf ~/.m2/repository/org/fcrepo
git checkout $REPO-$CURR # detached head state i.e. > git checkout fcrepo-6.4.0
mvn clean install
```

Up until this point, all of the changes made are strictly in your local repository and working directory. From this point on, the changes you make will be visible to the world and in some cases difficult to back-out of.

Push the changes to Github.

```
git push origin --tags # mvn task relies on the tag, make sure it does not collide with a branch name
```

Sonatype Release

Release the build artifacts to the Sonatype repository.

```
mvn release:perform -DperformRelease -Dgoals=deploy
```

As before, your GPG passphrase may not be masked in terminal.

If you have more than one personal key on your GPG keyring, you can specify the correct key by adding

```
-Darguments=-Dgpg.keyname=<Your Key ID>
```

to the above *mvn* command.

Troubleshooting

If you get a warning like:

[ERROR] Provider message:
[ERROR] The git-clone command failed.
[ERROR] Command output:
[ERROR] Cloning into '/working_directory/fcrepo/fcrepo-storage-ocfl/target/checkout'...
[ERROR] git@github.com: Permission denied (publickey).
[ERROR] fatal: Could not read from remote repository.
[ERROR]
[ERROR] Please make sure you have the correct access rights
[ERROR] and the repository exists.

You may need to generate a SSH key, associate it with your Github account and with your local ssh client. More instructions are available on Github
Point of no return

The following steps, once completed are final. They cannot be undone, revoked or altered. Only proceed if you've completed all the above steps and are absolutely certain the release is ready for publication.

- Go to https://s01.oss.sonatype.org and log in
- Click Staging Repositories in left navigation
- Search for "fcrepo" in upper right search box (project will not have $REPO in title)
- Select repository and verify that $REPO is present in the Content tab
- Click Close, then Refresh, then Release

This will publish the artifacts to the Sonatype releases repository and start the process of syncing them with Maven Central, which may take several hours. When finished, they'll be available at https://repo1.maven.org/maven2/org/fcrepo/.

Docker Release (fcrepo only)

From the cloned fcrepo repository home run the following:

```
git clone git@github.com:fcrepo-exts/fcrepo-docker.git
cd fcrepo-docker
DOCKER_PASSWORD=<password> DOCKER_USERNAME=<username> ./build-and-push-to-dockerhub.sh ../fcrepo-webapp/target
/fcrepo-webapp-${CURR}.war latest ${CURR}
```

Github Release - part 2

- Go to `https://github.com/fcrepo/$REPO/releases/tag/fcrepo-$CURR`
- Click Edit tag, and update title to "Release $CURR"
- Attach fcrepo-webapp-$CURR binaries and checksums that have been published to Maven Central to the Github release
    - e.g. https://repo1.maven.org/maven2/org/fcrepo/fcrepo-webapp/$CURR/
- Build the fcrepo-webapp-$CURR-jetty-console.jar for the release using

  ```
  mvn clean install -Pone-click -pl fcrepo-webapp
  ```

- Create checksums for the fcrepo-webapp-$CURR-jetty-console and attach the binary and checksums to the Github release.
    - Note: The checksum files should be of the format "[checksum] [filename]" (MacOSX's md5 requires the use of the -r argument to produce the correct format. I.e. `md5 -r fcrepo-webapp-5.0.2-jetty-console.jar >> fcrepo-webapp-5.0.2-jetty-console.jar.md5`).

      ```
      jarPath=fcrepo-webapp/target/fcrepo-webapp-$CURR-jetty-console.jar
      md5 -r ${jarPath} > ${jarPath}.md5
      shasum ${jarPath}> ${jarPath}.sha1
      ```

- Click Publish Release

Github Pages

Update the Github Pages documentation:

```
mvn site-deploy -DskipTests
```

fcrepo pages will be visible at `http://docs.fcrepo.org/site/$CURR/$REPO/`

Other module pages will be located at: `$ORG.github.io/$REPO/site/$CURR/fcrepo/$REPO`

For fcrepo/fcrepo and fcrepo-exts/fcrepo-camel, manually add links to the current releases. The easiest way to do this is to search for an old version number and copy/update for the current release.

Troubleshooting

Error creating blob: API rate limit exceeded

Github only allows a certain number of requests per hour. Once that number is hit you'll have to wait an hour before resuming your operation. The site documentation may exhaust this limit several times.

If you get the following error:

> *Error creating blob: You have triggered an abuse detection mechanism and have been temporarily blocked from calling the API. Please retry your request again later. (403)*

You may consider using the patched version of site-maven-plugin: https://github.com/github/maven-plugins/commit/d4ccf887098b18e9a23b7316ecf96348a2c73d0a

Or a `405` error or a 502 error:

You may consider using the patched version of site-maven-plugin: https://github.com/github/maven-plugins/commit/09b282544a1208be63bd012c2cdfd4d58e3f2d22

If you use two factor authentication with Github and have a **Personal Access Token** setup for Maven. Ensure that this token has the **repo** and **user:email** permissions.

If you get the following error:

```
Failed to execute goal com.github.github:site-maven-plugin:0.12:site (github) on project fcrepo-module-auth-
rbacl: Error creating commit: Invalid request.
[ERROR]
[ERROR] For ‘properties/name’, nil is not a string.
[ERROR] For ‘properties/name’, nil is not a string. (422)
```

`You will need to ensure that the "Name" field of your github profile is not null. Fix it by going to github.com and updating your profile.`

Error creating blob: cannot retry due to server authentication, in streaming mode

This is an authentication error, check your password or token in your Maven settings.xml file. If you use 2-factor you can create a new token with the following permissions *notifications, public_repo, repo:status, repo_deployment, user:email*

Push Release Branch to Maintenance

The release branch has changes made since code freeze. It also contains the update to the version numbers for future development.

```
git checkout ${CURR}-RC # this is your local copy of the release branch
git log
```

Ensure that your commit history matches the release branch's commit history, except for the two additional commits.

1. Changing from SNAPSHOT version to release version. Something like **[maven-release-plugin] prepare release $REPO-$CURR**
2. Changing from release version to next development version. Something like **[maven-release-plugin] prepare for next development iteration**

If this appears correct, you can push your release branch on to the maintenance branch.

```
git push origin ${CURR}-RC:${CURR}-maintenance
```

Because there are no changes to main after code freeze and all bug fixes are on the ${CURR}-RC branch, this will operate as a fast-forward merge.

Lyrasis wiki documentation updates

Current, in-progress Fedora Repository Documentation wiki: Fedora 6.x Documentation

At the very minimum, update the following:

- Documentation
- Downloads
- Releases
- Fedora Upgrade Notes

Note the version of Java against which the release was built.

New Wiki Space

1. If this is a new major or minor point release, copy the current, in-progress documentation to create the release wiki, then update accordingly. Mark the new pages as current, and update the pages in the previous documentation to indicate they are out-of-date.
    a. Add the following header to the previous major release wiki space (see: Space Tools -> Look and Feel -> Sidebar, header and footer)

    ```
    {note:title=Old Release}
    This documentation covers an old version of Fedora. Looking for another version? [See all
    documentation|FF:Documentation].
    {note}
    ```

    b. Add the following header to the new release wiki space

    ```
    {tip:title=Current Release}
    This documentation covers the current version of Fedora. Looking for another version? [See all
    documentation|FF:Documentation].
    {tip}
    ```

    Make sure the license and copyright information is up-to-date with this release.
2. Update permissions on new wiki space to be like the previous space
3. Add "fedora" category to new wiki space
    a. Space Tools Overview
4. Add logo
    a. Sidebar Space Details
5. Remove default space pages
6. Set space "Home Page"
    a. Space Tools Space Details
7. Ensure `noformat` and `code` macros were copied successfully (this is a bug in the Confluence "copy space" utility)
    a. e.g: First Steps
8. Update Documentation wiki page

Close the release in Jira, and create the next one

1. Go to the release management page.
2. For the release you just made (there should be an open package icon to the left of it)
    a. click the gear icon on the left and choose "release"
    b. enter the date you finished the release
    c. the package should be closed
    d. if you are not already listed as the release manager in the description, enter "Release Manager: your-name-here"
3. Create the next release
    a. enter a name (ie, Fedora 4.x.y) in the form at the top, and click Add. If the release manager is known, enter that in the note.

Update the Fedora Repository site

Fedora Repository site (Drupal): http://www.fedorarepository.org/

- Update Download page based on Downloads page in wiki
- Update News on front page with release information
- Update Documentation page with link to current release documentation

Last sanity checks :

1. Assuming a fcrepo4 release: Download and run the fcrepo-webapp-$CURR-jetty-console.jar
2. Make sure that the checksums of the artifacts published in the github release page match those in sonatype.
3. If we're talking about a maintenance release, make sure that all commits that went into the maintenance release are also on main (where appropriate)

Announce release

Let Danny Bernstein know that the release is complete and can be announced.

Ontologies

Ontologies are released on their own schedule and do not need to be released. But they should be tagged with the current version when a release is performed, to make it easy to identify the version of the ontology that each release was using.

```
CURR=4.5.1
ORG=fcrepo4
REPO=fcrepo-webac-ontology
git clone git@github.com:$ORG/$REPO
cd $REPO
git tag -a "$REPO-$CURR" -m "$REPO-$CURR" # except fcrepo4/ontology should be fcrepo-ontology-$CURR
git push --tags
```