Patches

How to apply a PATCH

Depending on the origin of a patch, the best way to apply it is slightly different.

Important: All patches require a corresponding issue filed in JIRA before it can be added to the source repository.

Important: When applying a patch (or pull-request) it is your responsibility to check that all requirements for contributions to the ASF are met. This includes especially that a patch has to be submitted to JIRA or one of the projects mailing lists.

Patch from JIRA

Applying a patch attached to an issue in JIRA is quite straight foreward:

1. Download the patch and apply it to the source:
   `git apply MARMOTTA-123.patch`
2. Examine the changes introduced by the patch. Make sure the code compiles and does what it claims in the issue. Check that
   a. all LICENSE headers are in place (maybe you want to run `mvn apache-rat:check`),
   b. all newly introduced dependencies are covered in the N&L files (remember that some dependencies are bundled and shipped with the binary release packages!)
   Add you changes to the commit: `git add <file>`
3. If the changes comply with both, your personal and the projects quality standards, add the author of the patch as a *contributor* in the parent-pom.
   Add the parent pom to the commit: `git add parent/pom.xml`
4. Commit all changes. Make sure to include
   a. the author of the patch,
   b. the issue numer
   to the commit message: `git commit -m "MARMOTTA-123: applied patch by John Doe" -m "Submitted by: John Doe <john.doe.at.null.org>" --signoff --edit -S`
   The merge commit will be signed with your GPG-Key.

Pull-Request from Github

Pull requests from via github should be included slightly different that "plain patches" to keep the original authorship of the changes.

Also pull requests on github require a corresponding inssue in JIRA **with a patch attached**. The patch can be downloaded directly from github by adding a *.patch* extension to the pull-request url. The patch has to be attached by the author.

Use the following steps to apply a pull-requests from github.

1. Add the forked github repository the pull-requests origins from:
   `git remote add <github-user> https://github.com/<github-user>/marmotta.git` you can also restrict the remote to a specific branch:
   `git remote add -t <remote-branch> <github-user> https://github.com/<github-user>/marmotta.git`
2. Fetch the changes
   `git fetch <github-user>`
3. Merge in the changes telling where the change comes from and tell github to close the pull request:
   `git merge --no-commit --no-ff --log -m "MARMOTTA-123: Merged patch from <github-user> on github, closes #1234" -m "Obtained from: <github-user> <https://github.com/<github-user>/>" <github-user>/<remote-branch>`
4. Examine the changes introduced by the patch. Make sure the code compiles and does what it claims in the pull request. Check that
   a. all LICENSE headers are in place (maybe you want to run `mvn apache-rat:check`),
   b. all newly introduced dependencies are covered in the N&L files (remember that some dependencies are bundled and shipped with the binary release packages!)
   Add all changes to the merge-commit: `git add <file>`
5. If the changes comply with both, your personal and the projects quality standards, add the author of the pull-request as a *contributor* in the parent-pom. Add the change in the parent-pom to the merge-commit: `git add parent/pom.xml`
6. Commit the merge
   `git commit --signoff -S` The merge commit will be signed with your GPG-Key.