How to merge a Log4j pull request from GitHub

When a developer submits a Pull Request (PR) to the Apache Log4j project on GitHub, you can follow these steps to accept and merge that PR into the Log4j code base.

Step-by-step guide

Step #1: Carefully review the PR

Make sure that the code for the PR looks good and includes JUnit tests that prove that the code works.

Check which branch the PR is targeting and make sure you want the code to go into that branch

If the PR is large or adds new files, then make sure the person who submitted the request has an Apache ICLA on file. Ask the contributor to fill out this form and follow the instructions on the form to sent it in to Apache: https://www.apache.org/licenses/icla.txt

Check the code to ensure that it does not bring in any code or dependencies with licenses more restrictive than the Apache Software License (ASLv2). For example, we cannot bring in any code or dependencies that are under GPL or LGPL license.

Step #2: Configure your Git client to map GitHub PRs to ref

Add the github remote location to your Git config for a specific Logging Services project. If you already have a remote for GitHub then add the two lines that mention "refs" to it.

cd into the .git directory inside the project.

Add the following to the file named "config".

```
[remote "github"]
url = https://github.com/apache/logging-log4j2
fetch = +refs/heads/*:refs/remotes/github/*
fetch = +refs/pull/*/head:refs/remotes/github/pr/*
```

NOTE: For the rest of this guide we will assume that Apache Git is the remote named "origin" and GitHub is the remote named "github".

STEP #3: Fetch the latest PR refs from GitHub

Use git's fetch command to pull in the latest PR refs from GitHub.

```
git fetch github
```

STEP #4: Checkout the PR code

Now you can fetch the code for the Pull Request like so:

```
git checkout pr/<pull request number>
```

STEP #5: Merge the PR into the desired branch

First checkout the desired branch and make sure it is up to date, for example:

```
git checkout master
git pull origin
```

Next, use the merge command to merge the code into the target branch. And make sure to include a "This closes #" message so that Apache's GitHub integration feature will close the PR. For example:

```
git merge -m "Closes #<Pull Request Number>" pr/<pull request number>
```

Perform a build and verify that it passes.

STEP #5: Push the code

Push the code to Apache Git.

```
git push origin
```

Thank the contributor for their code.