How can I check that all the changes in a git branch have been cherry-picked or rebased into its upstream branch?

devblogs.microsoft.com/oldnewthing/20240920-00

September 20, 2024

If you try to delete a branch that contains changes that haven't been merged to its upstream, git warns you, `the branch is not fully merged. If you are sure you want to delete it, run 'git branch -D xxx'`. Suppose you want to check that all of the changes in the branch have been cherry-picked or rebased into the upstream branch. How can you confirm this? How can you do the "If you are sure" step?

One way to do this is to merge the branch into its upstream and see if the result produces any changes.

```
# Detach the HEAD from the upstream so we can do some tinkering.
$ git checkout --detach featurebranch{@u}

# Propose a merge of the feature branch.
$ git merge featurebranch
$ ⟦ resolve any merge conflicts and commit ⟧

# See if anything would have changed.
$ git diff HEAD HEAD~1

# Return to original branch, abandon detached HEAD.
$ git checkout -
```

If the `git diff` produces no output, then everything in your `featurebranch` is already present in the upstream, so you can delete the `featurebranch`.

If the `git diff` produces output, then there is content in your `featurebranch` that is missing from the upstream, so you want to look at those changes and see if they are worth keeping.

The older the `featurebranch` is, the more likely that you'll get a merge conflict when doing a test merge with its upstream, since that gives the upstream more time to make a conflicting change to the files that you changed in `featurebranch`.