Git Branches

GITHUB PULL REQUEST, Branching, Merging & Team Workflow (11:20)

https://www.youtube.com/watch?v=oFYyTZwMyAg

Overview

This document describes why we branch, how to branch, and the commands we use to branch.

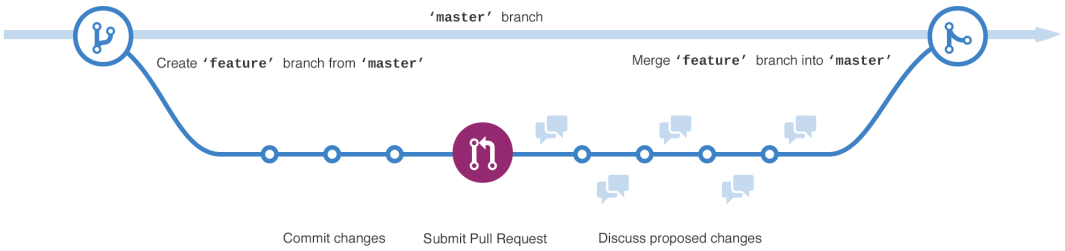
Branching

Why we branch

The Git basics documentation describes how to make changes to the github repository. Developers make changes to local copies of the repository, called branches. Branches allow developers to make changes to a local version of the repository, get those changes reviewed, and then submit the changes to the master branch. The reviewer attempts to catch any mistakes in the commit.

How to branch

Consider the repository to be the master branch, or `master`. Any changes made to newly created branches do not affect the `master`. When you create a branch, you create a copy of the repository.



https://guides.github.com/activities/hello-world/branching.png

Developers make their changes to these newly created branches, usually for a specific feature. Generally each JIRA story gets its own branch.

Before the developer pushes their branch to the master branch, they submit a `pull request`. A pull request usually requires another person to sign off for approval.

Commands to branch

Command	Function
`git branch`	list branches available
`git branch <name of branch>`	create a new branch
`git checkout <name of branch>`	swap between branches
`git checkout -b <name of branch>`	create a new branch, then immediately swap to it
`git fetch`	Download all changes from the cloned repository
`git merge`	Merge those changes with the current branch

~~`git pull`~~	Do **not** use `git pull`, use `git fetch` **then** `git merge` instead. For more information see this website's http://longair.net/blog/2009/04/16/git-fetch-and-merge/ article.

Procedure for pull request

Pretend your teams's assignment is to create documentation about how to use branches in git. You would use the following commands to execute this task.

1. Create a new branch to work on
    a. Run `git checkout -b BranchDocs` to create a new branch, and immediately switch to that branch (or run `git checkout --track -B BranchDocs origin/master` to create a branch that tracks from `master`)

> **Note:**
> You would likely call this branch by the same name as the JIRA case and description. For example, `CM-4200-branch-basics`

2. Create the document about branches, and save it in the repository's folder
    a. For example, if you ran `git clone` in your local `~/Documents` folder, then self-titled repo will be in the `~/Documents` folder
    b. You may need to move the file to the sub folder, for example

```
mv ~/Documents/BranchBasics.pdf
~/Documents/gitrepos/TestRepo/BranchBasics.pdf
```

3. Run `git checkout master`
    a. Run `git fetch` then `git merge` to ensure you are up to date
    b. Run `git checkout BranchDocs` then `git merge master` to pull any changes from `master` into your `BranchDocs` branch
4. Run `git add <filename>`, for example `git add BranchBasics.pdf`
5. Run `git commit`
6. Run `git push`
    a. You may be prompted with the following, just copy and paste the command to run:

```
git push origin BranchDocs
```

7. Navigate to your github repo, for example `https://github.com/Griffith22/TestRepo`
    a. Select the branch you were working on, for example `BranchDocs`
    b. Select *New pull request* or *Compare & Pull Request*
    c. Enter relevant information, then click *Create pull request*
    d. At this time someone else can review your work for accuracy
    e. Then they will click *Merge Pull Request,* then *Confirm merge* to approve the merge
        i. This runs the following commands, which pushes your branch's changes to `master` in the github repository:

```
# Step 1: From your project repository, bring in the changes
and test.

git fetch origin
git checkout -b BranchDocs2 origin/BranchDocs2
git merge master

# Step 2: Merge the changes and update on GitHub.

git checkout master
git merge --no-ff BranchDocs2
git push origin master
```

ii. You will get the message *You're all set—the BranchDocs branch can be safely deleted.* when the process completes.
    1. Your work in the branch is technically complete, and you may delete branch, or continue to work on relevant projects (for example, other tasks in the JIRA story)

The cPanel way

cPanel way

The following steps are the cPanel way, and I need to double check them:

1. Navigate to `https://enterprise.cpanel.net/projects/CPANEL/`
2. Click on your team's link
3. Select the branch you want to merge to
4. Click *Pull Requests* on the left hand pane
5. Click *Create pull request*
6. Choose to create the pull request from your branch, to the team's branch
    a. For example, Pull from: Garrett Griffith / CM-2016 to: cpanel-whm/code-monkeys / deprecate-mailman-scripts
7. Then click *continue*
    a. These steps need to be updated with what to do from here.