GIT Branching Exercise

in list Git

Labels

Description Edit

https://guides.github.com/activities/hello-world/#branch
This guide will teach you how to create and use branches and pull requests on Github. What is a branch? A branch is a duplicate of your project where you can make changes to your code in isolation. This is handy for collaboration and experimentation. Note that the default branch and the one you want to keep clean is going to be master. Every other branch is where you want to break and experiment with code. Mess up a branch? That's okay! Just nuke the branch and make a new one!

Now what is a pull request? A pull request is an automated way of merging your change from one branch into another. This will be through the Github website. A pull request will show whether you can merge your code automatically or manually. Also it's a space to discuss with your collaborators the changes to your project. Each collaborator will be able to approve or disapprove changes and leave comments.

The idea behind pull requests is that it enforces a standard operation procedure. You can now collaborate using Git with a lot less fear!

Now what the heck is a merge? Merging is the process of refactoring two different versions of a project into one. Merging will require that your files will need to change.

What you'll need Delete

0%

- ☐ 1) a Github Repo with all the collaborators invited
- ☐ 2) A local git repo pointed to the desired project
- ☐ 3) Git bash opened and pointed to the Local repo
- ☐ 4) Visual Studio (or whatever your preferred IDE is)

Add an item

Creating and Point to a Branch Delete

0%

- ☐ 1) Go to the Github Remote Repo you wish to branch.

- [ ] 2) Towards the top left click the branch drop down, type in the name of the branch you wish to create.
- [ ] 3) In Git Bash, go in and type Git branch, that will show you what branch you're on, like git status this is useful to do throughout the process to see what bash sees.
- [ ] 4) If you only see master, do a git pull to pull down info on your new branch
- [ ] 5) Test if it worked use: git branch again, note that your current branch will have a * next to it
- [ ] 6) To swap branches use: Git checkout BranchNameHere, try to check out your new branch
- [ ] 7) Right now the two branches should be identical, let's get dangerous and edit the new branch, go into visual studio and change up a line
- [ ] 8) To track your new changes, go git add . and then Git commit -m "message" as you normally would
- [ ] 9) Time to upload go git push
- [ ] 10) Now let's hop back to github, switch to your branch from the drop down towards the top left (same one as when you made it) and see if your change made it up
- [ ] 11) If not let's spend time debugging, git branch, git status, and git remote -v will all be handy here
- [ ] 12) If yes, you will see a little message pop up with a green button on the far right labeled "Compare and Pull Request" smash that button, not a gentle tap, git only respects force.

Add an item

Pull Request

Delete

0%

- [ ] 0) As you go through this process, make sure you have group mates handy to check and see which code you do and don't want. It's a good idea to sit down with everyone once a day, and one by one, merge your branches into master.
- [ ] 1) Alright we're in a new screen so let's take a moment to understand it. At the top it will have the branch you want to merge into and the branch you want to merge from. Note you can switch around which two branches you want to merge. To the right there will be a message that will tell you if you can merge automatically. If its green you're good to merge. If it's red, you'll have to do it manually but no worries, Github can help you...
- [ ] 2) Okay here let's assume you're able to auto merge, type in a comment detailing the changes and hit the green Create pull request in the bottom right corner. If not, skip to 4.
- [ ] 3) New screen! Simply hit the green button that says merge pull request and you're golden. If you're collaborating with others, you may have to wait for approval from others before you can go ahead. Congrats now you're done!

- [ ] 4) Alright so you got that red message saying you can't auto merge. Still make that pull request.
- [ ] 5) Now that you've made a pull request there will be a box that says there are conflicts that must be resolved. It will list out the offending files.
- [ ] 6) Hit the resolve conflicts button. Github will take you to each spot with a conflict.
- [ ] 7) In this new page you can now edit your code directly, you'll notice that you'll have little blocks split in two. That is where you are getting merge conflicts. You may delete one branch's code, or combine both branches, what you do will change on a case by case basis.
- [ ] 8) Flip through using the arrows in the top right to find and resolve each conflict. After you edit everything hit the mark resolved button.
- [ ] 9) That button will then turn green into a commit merge button. Hit that like it owes you lunch money.
- [ ] 9) You'll be taken back to the pull request page simply hit the green Merge Pull Request button and you're all set!
- [ ] 10) Once done merging go into git bash on your branch, and pull down the code changes from the remote.

Add an item

How to Branch in a Group

Delete

0%

- [ ] 1) one person make a repo and invite everyone else as collaborators
- [ ] 2) Each person should make their own branch.
- [ ] 3) Add, commit, and push to your branches
- [ ] 4) Come time to submit code into the master, each member should run their pull requests and merges one at a time just to be safe.
- [ ] 5) Once master has all the new changes, merge from master into your own branch
- [ ] 6) Lose something? No big deal go to commits, and find your old commits. copy paste back into your code and upload it to a branch and restart the merge process.

Add an item

Activity

Show details