GitHub Branching and Workflows

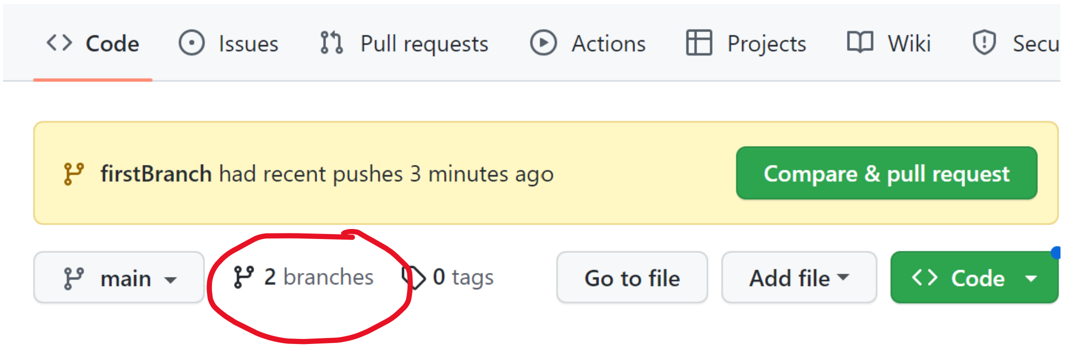
Connect to remote repo on GitHub.

1. Have a local folder initialised with git that contains a file or two (from previous lesson maybe)
2. Sign into GitHub, create a **new** Repo and name it <git-test>.
3. Copy the link to the Repo, and go to a CLI, for example Command Prompt or Git Bash.
4. To connect your local git folder to the remote repo in GitHub, type **git remote add origin <repository URL>** where <repository URL> is the link to your GitHub repo.
5. Type git push -u origin main to push your local changes to the **main** remote repository.
    a. The -u flag is used to set the upstream branch, which tells Git where to push the changes.
    b. Origin is the original location of the directory on GitHub.
    c. the main/master branch is the default branch in Git, but you can use a different branch if you prefer.

Practice 1

1. Delete your local folder from File Explorer (it is now on GitHub).
2. Go to your remote repo on GitHub, click code and copy.
3. Navigate to your git folder, using your CLI, type git clone and paste the link, enter.
4. View the files in your folder. Cloning is faster than downloading!

Branching



We are going to create 2 branches, add files to both then pull/merge them with the main branch.

1. Create a branch: git branch <firstBranch>, then **switch** to it: git checkout <firstBranch>
2. **OR** create AND switch: git checkout -b <firstBranch>
3. **OR** create AND switch git switch -c <branch-name> (*new command since version 2.0, the **-c** creates a branch)
4. Create a change, for example, make a new file.
5. Add (to stage) and commit (to local repo)…

6. Push to new branch: git push −-set-upstream origin <firstBranch>
7. Read the reply in the CLI. Go to GitHub.
8. There is a Compare & pull request waiting for you and you now have 2 branches.
9. Follow the prompts to: Open a pull request, merge a pull request, etc.

Practice 2

1. Create another new branch in your CLI, for example secondBranch and switch to it.
2. To see local branch names, run git branch
3. To see all remote branch names, run git branch -r
4. To see all local and remote branches, run git branch -a
5. To switch to the previous branch checked out: git switch -
6. Pull from the main to the new branch: git pull origin main
7. Make a change, for example add another file or update an existing file
8. Use git commands to Add, Commit, Push, then Pull & Merge.

Use the git merge - to merge to the previous branch you were working with, on to the current branch (local repo). For example, if you wanted to merge firstBranch and secondBranch into one, then push to the remote repo.

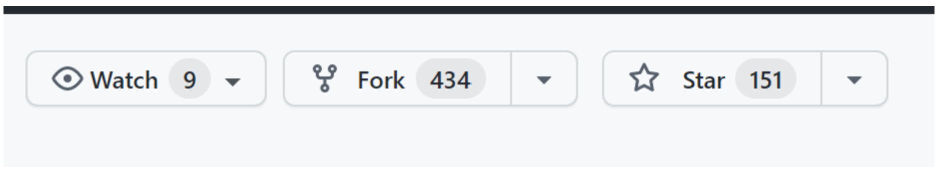
Practice 3 – Groupwork (Collaborators).

1. Break into groups of 3, create a new repo with readme.md
2. Invite collaborators (Settings)
3. Each will create a new branch named <yourname> and switch to the branch.
4. Create new files, for example index.html, style.css, script.js
5. Add, commit and push to local repo.
6. Push to remote repo.
7. Use pull request on collaborators branch, add comments.
8. Reply to comments on your branch.
9. Finally accept changes and merge.

Fork from GitHub (off another GitHub users repo)

Fork vs Clone

A fork creates a completely independent copy of Git repository.
Git clone creates a linked copy that will continue to synchronize with the target repository.



5 Steps to a Pull Request

1. Fork a Main Repository.
2. Clone the remote repo to a local folder.
3. Make Needed Changes Locally.
4. Push Local Changes to Forked Repository (Your copy on GitHub)
5. Make a Pull Request, for the original owner to Review and Comment.
6. Then they can Merge with Main Project, if they choose to.