Source Code Control through Git

Victor Eijkhout

2022

1 Justification

Source code control packages, such as Git, are essential for synchronizing software between developers, or multiple accounts for a single developer. They also allow you to keep a history of changes, and roll them back when needed.

2 Preliminaries

Create an account on `github.com`

Pick a good name, not referring to this class, so that you can keep it for a while.

Authentication setup:

- Find the `id_rsa.pub` file (your 'public key') in your `.ssh` directory; copy the contents.
- Go to personal settings, section 'SSH keys' and add a key for the cluster. You may need one that is specific to the login node!

3 Creating a repository

1. If you have a directory with material, you can declare it to be(come) a repository:
   `git init`
2. Easier:
   2.1 Make a new repository on `github.com`
   2.2 Do `git clone` with it.

Github notes:

- On TACC machines, use `ssh` to clone a repo, not `https`
- See the point about ssh keys above.

4 Adding files

- Create a file
- Do `git status`
- Do `git add yourfile`
- Enter message: `git commit -m "this is what I did"`
- do `git push`
- Check the `github.com` page for your repository.

5 Changes to files

- Edit the file that was added to the repo
- Explore `git status` and `git diff`
- Add and commit and push again.

6 Collaboration

- Clone a repository from someone else
- (make sure you have permission to push to it)
- create a file and add/commit/push it
- The original owner can pull it.

7 Merging changes

- Start with a file that is longer than a couple of lines
- Two people edit the same file, one at the top, the other at the bottom.
- Both add/commit/push
- Do you get an error message? Pull before push.
- Are both changes visible in the file?

8 Merging conflicting changes

- Make changes on two adjacent lines
- Merging should fail
- Do a manual edit to resolve the conflict
- (Did you get some full-screen tool?)

9 Branches

Branches are good for experiments

- Create a branch
  ```
  git branch dev
  git checkout dev
  ```
- which branches do you have?
  ```
  git branch -a
  ```
  which one are you currently on?

10 working with branches

- While on the `dev` branch, make an edit to a file
- Check that the file is not edited no the main branch
- Go to the main branch, make a non-conflicting change

11 Merging

- See the difference between branches
  `git diff main dev`
- Merge while on the main branch:
  `git merge dev`
- Inspect.