Group1: Git Exercise

Preview

Suppose you are Lucy and your classmate is Charlie. Your goal is to produce a `names.txt` file on GitHub with the following contents:

```
role,NetID,LastName,FirstName
boss,lucy123,Van Pelt,Lucy
employee,cbrown,Brown,Charlie
```

Broadly, these are the three steps:

- Acting as a boss, make a first draft of `names.txt` with your information and get it to GitHub:
  `boss,lucy123,Van Pelt,Lucy`
- Since you're the boss, direct an employee (your classmate Charlie) to add his information in a second draft and get it to GitHub:
  `boss,lucy123,Van Pelt,Lucy`
  `employee,cbrown,Brown,Charlie`
- Then, as boss, realize the file should have a header, so make a third draft and get it to GitHub:
  `role,NetID,LastName,FirstName`
  `boss,lucy123,Van Pelt,Lucy`
  `employee,cbrown,Brown,Charlie`

Meanwhile, you should also play the employee role for another student. The surface goal is to make `names.txt`. The deeper goal is to practice version control and collaboration with git and GitHub.

1. Acting as a boss, make a tiny git repository.
   (a) Start emacs and open a shell buffer via `M-x shell`
   (b) Make a directory in which to practice via `mkdir ~/gitExample`
   (c) Change to the new directory via `cd ~/gitExample`
   (d) Configure git as follows (in the first line, use your name, and in the second, use your email address):

   ```
   git config --global user.name "Your Name"
   git config --global user.email "Your email address"
   git config --global core.editor emacs
   git config --list
   ```

   (e) Create an empty git repository in your directory: `git init`
   (f) Create a file `names.txt` (which will become a roster of people) by revising the following command to use your NetID, last name, and first name and then running the command:
   `echo "boss,NetID,LastName,FirstName" > names.txt`
   (g) Stage the new file for tracking: `git add names.txt`

(h) Check the status of your repository: `git status`
(You should see `names.txt` in the output.)

(i) Save a snapshot of your file; that is, commit your changes with a descriptive message:
`git commit -m "Create names.txt with a boss line."`
`git branch -M main` (rename branch from "master" to "main")

2. Put your repository on GitHub.com.

(a) Access a GitHub account (whose account name we'll refer to as ID):

- If you have a GitHub account, you may use it.
- If you do not have a GitHub account (or you want a new one), create one:
  i. Visit `https://github.com`
  ii. Click "Sign up"
  iii. Follow the instructions. (I chose a free account. I rejected receiving emails.)

(b) On GitHub, create a repository with name `gitExample`. The new account dialog gives an opportunity, or you can use the "+ > New repository" menu in the upper right corner.

i. Choose "Public," not "Private."
ii. Leave unchecked "Add a README file," "Add .gitignore," and "Choose a license."
iii. Click "Create repository."

(c) In your `gitExample` directory, to tie your local respository to GitHub, run
`git remote add origin git@github.com:ID/gitExample.git`
(after changing `ID` to your GitHub ID).
(Note: To make changes to `origin` after setting it, run `git remote set-url origin <URL>`, where `<URL>` is the new URL you want to use.)

(d) Run `git remote -v` to confirm the previous command worked.

(e) Set up SSH (Secure Shell; to allow push and pull commands without authenticating):

i. Generate new SSH keys by running:
`ssh-keygen`
(and hit Enter to accept the default answers to the three questions).
This creates two files:

- `~/.ssh/id_rsa.pub`, a *public key* that you can give to anybody (including GitHub) to encrypt (or "lock") a message to you;
- `~/.ssh/id_rsa`, a *private key* that only you can use to decrypt (or "unlock") a message to you.

ii. Use the Terminal (not emacs) to run `cat ~/.ssh/id_rsa.pub` and copy its output to the clipboard. (Do not use emacs here because it displays a backslash (\) before continuing a too-long line on the next line.)

iii. At GitHub, click the top right (profile picture) menu and choose "Settings," click "SSH and GPG keys" in the left margin, click "New SSH key," paste your key, and click "Add SSH key."

(f) Finally, push your local repository to GitHub: `git push origin main`

(Note: If you don't want your password to be visible in the emacs shell, run this command in a new Terminal window instead.)

3. Add two collaborators to your repository at GitHub while recording who your boss is:

   (a) Collect GitHub IDs for two collaborators:

      i. TA's GitHub ID: _____

      ii. (pretend) employee's GitHub ID: _____ (my neighbor to my right)

   (b) For use later, record your boss's GitHub ID here. My (pretend) boss is _____ (my neighbor to my left).

   (c) On GitHub, click on your `gitExample` repository, then "Settings" (on the upper right), and "Collaborators" (in the left margin) to get to "Manage access".

      i. Click the green ~~"Invite a collaborator"~~ "Add People" button.

      ii. In the "Search by username, full name, or email address" box, enter your two collaborators' GitHub IDs (TA, employee).

      iii. Click "Add collaborator".

4. Acting as an employee, contribute to your boss's project by making a one-line addition.

   (a) Copy your boss's repository from GitHub into a directory called `gitBoss`:

   `git clone git@github.com:bossID/gitExample.git ~/gitBoss`

   (where `bossID` is your *boss*'s GitHub ID)

   (b) Change to your boss's repository directory:

   `cd ~/gitBoss`

   (c) Run `ls` (list files) and `cat names.txt` (write the file to the standard output) and `git status` to verify that `names.txt` is in the repository.

   (d) Append your information to your boss's `names.txt` by running

   `echo "employee,NetID,LastName,FirstName" >> names.txt`

   (after replacing `NetID`, `LastName`, and `FirstName` with your information).

   (e) Add the modified file to the (boss's) stage:

   `git add names.txt`

   (f) Commit the change to the local (boss) repository:

   `git commit -m "Add a line to boss's file."`

   (g) Pull the GitHub repository of your "boss" (to avoid wiping out changes made by others while you were working with the repository):

   `git pull origin main`

   Fix conflicts if necessary (there should not be any conflicts today).

   (h) Push your change to your boss's GitHub repository:

   `git push origin main`

5. Act as a boss again.

   (a) Return to your first repository: `cd ~/gitExample`

   (b) Update your local repository to receive your employee's change: `git pull origin main`

   (c) Check your `names.txt` file to see that is has two lines, your `boss` line and your employee's `employee` line. (If it isn't there yet, try again soon.)

   (d) Add a header line to `names.txt` by adding this line at the beginning of the file:

   `role,NetID,LastName,FirstName`

   (This time, "`role,NetID,LastName,FirstName`" should be written literally. Do *not* substitute your information.)

   (e) Add your changes, commit them with a descriptive message and push them to your GitHub repository.

6. What to turn in:

   (a) Submit your GitHub URL to Canvas's Group1git assignment. (The URL is https://github.com/ID/gitExample with your ID.)

   (b) Submit your boss's GitHub URL to Canvas's Group1gitBoss assignment.

   Confirm that your GitHub repository files are correct.

   Confirm that your name is in `names.txt` at your boss's GitHub repository.