Getting Started with Git

BU-PRO Summer 2025

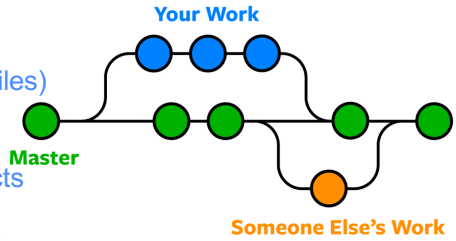
Based on slides originally created by Maria Gamez and Hayden Swanson, BU-PRO 2022
With procedural contributions by Isaul Garcia, BU-PRO 2023

Git Overview

Git is a version control system that lets you manage and keep track of your source code history.

Git is software used for

- Tracking changes in any set of files (best for text files)
- Coordinating work among other programmers
- Working collaboratively
- Handles everything from small to very large projects
- Managing products using simple commands



GitHub Website

GitHub is a cloud-based hosting service that lets you manage git repositories. There are others like GitLab.

Github:

- Hosts repositories where you can collaborate with others
- Or just keep VCS backups of your work
- Saves your project files and helps interaction and communication between other programmers
- Create and save new projects
- Contribute to an existing repository
- View and create pull requests

How git works

Creating repositories or projects where you can create changes and create edits while working independently or collaboratively.

- You can observe changes made by you or others to a specified files or codes and why those changes were made
- Can make debugging code easier by tracking changes
- Basic git commands allow you to create new projects, files and new edits
- A list of the basic commands are
    - **git init:** create a new local repository.
    - **git clone "GitHub URL" :** downloads a git repository from GitHub.
    - **git commit:** to create a change to a file

There are many other basic git commands that help with the collaborative aspect of creating edits, sharing and collaboratively working on projects.

Basic git commands

- **Creating a repository**
  - Create a “repository” (project) with a git hosting tool
  - git init
- **Cloning a repository**
  - Copy the repository to your local machine
  - git clone “URL”
- **Commit**
  - Add a file to your local repository and “commit” to save the changes
  - git commit
- **Push**
  - Push your changes to your main branch
  - git push
- **Other**
  - git status
  - git add .
  - git log --oneline
  - git push
- **Branching**
  - A branch from the master branch allows you to make modifications without changing your original branch (master branch)
  - git branch <branch name>
  - git branch –r (to list remote branches)
  - git branch –a (to list all branches)
- **Git Checkout**
  - to move to another branch in the repository
  - To checkout a forked branch:
    git checkout <branch name>
- **Pull Request**
  - when you want to add your changes, needs to be approved then can be merged
  - Deletes branch when successfully merged
  - git pull

Let’s GIT our hands dirty

Do you have Git Installed?
https://git-scm.com/downloads

Start on your device

- Open terminal or Git CMD window
- $ git config --global user.name “My Name”
- $ git config --global user.email myemail@domain.edu
- Create/browse to a directory for the project
- $ git init (creates repository in current directory)
- Create or edit a text file (Notepad, vim, Emacs…)
- $ git status (shows status of project)
- $ git add . (adds all file in current directory to project)
- $ git commit –m “message for log”
- $ git log (prints log of all past commits/changes)
- OR $ git log --oneline (more compact version)

Exercise

- Create a project
- Create text file (add words)
- Commit
- Make new changes
- Commit
- Repeat

Text Editors

Vim (purist-- like 'less' environment)
- i (insert), esc (escape editing), :wq (wite and quit)

Emacs (a little more user friendly?)
Notepad/Wordpad/TextEdit

Can undo changes or roll back the clock

RESET
$ Git reset [commit-ID] (7 dig alphanumeric code from log --oneline)

This doesn't change the file—just removes later commits from log
If want to undo all changes since that commit add "hard" flag
$ Git reset [commit-ID] –hard (careful—esp when working in team)

REVERT
Only undoes changes you made in that commit—not changes others have made
$ Git revert [commit-ID]
This will undo current commit, ie revert to previous
Does not delete commit, like reset does--Creates new commit

Branches– creating

Might want to make a ‘copy’ to experiment with.
$ git branch -a (lists existing branches)
$ git branch test (creates branch named ”test”)
$ git branch –a (lists again and now should see new one)
$ git checkout test (“checks out” new branch to work on it)
$ git branch –a (listing should now show which one is active)
$ git branch (or tells which branch you are on)

Make changes to your file or files, then…
$ git add . (or *)
$ git commit -m ”my message”
$ git log --oneline

Branches– merging

Once ready to merge changes with main branch again…
$ git branch -a
$ git checkout main (move back to the main branch)
$ git branch -a

If using remote repository with other users, first pull
$ git pull origin main (pull to make sure have latest)

$ git merge test (merges locally)

If using remote—now push to remote repos
$ git push origin main (performs merge on remote)

If done with a branch, should remove it
$ git branch -d test (deletes locally)

Getting Ready for Remote Repos

Go to github.com and create account or log in.
Browse to Settings under Profile picture
Then “SSH and GPG keys”

Set up ssh key to remotely access github from scc.
On your device...
$ ssh-keygen -t ed25519 -C "your_email@example.com"
(keep default location and put in a passphrase you will remember!)

Now we are gonna get the PUBLIC KEY that you just created:
$ cat ~/.ssh/id_ed25519.pub (cd to this directory if necessary first)
Copy text that is printed to screen.
Now back to github-- click “New SSH key” and paste it in.

To put local repository on GitHub

Create new empty project on github
(With no readme, license or ignore files)

Copy remote repository ssh location (Green “code” button)

Now set up a remote connection on local machine
$ git remote origin add [paste ssh location here] (creates remote object called ‘origin’)
$ git remote -v (will list remote connections that have been set up)
$ git push origin main

Now view in browser--try editing file on browser and committing
And pull locally
$ git pull origin main
Edit and push (remember pull first if shared repository)
Etc.

To start from a remote repository on GitHub

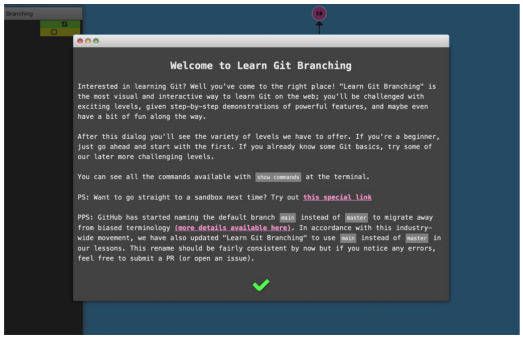
To grab an existing remote repository from start…
Navigate to directory you want project to be in
It will put a directory in current path with name of project—files will be in there
Don't need to do a git init
$ git clone [path to remote repository]
Make some edits
$ git add .
$ git commit –m "commit message"
$ git push origin main (perform pull first if shared repository)

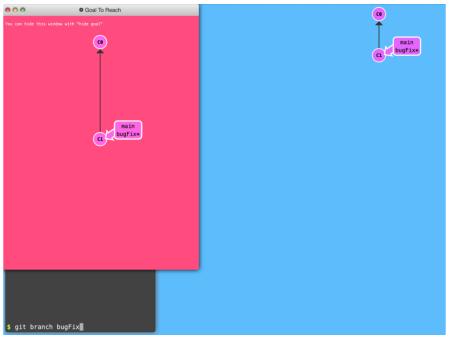
Note $ git fetch downloads changes on remote without merging
Compared to $ git pull, which also merges

Learn Git Branching

Get started with learning the basic commands of git and how the branches, cloning and forking commands work visually using the interactive git website:

https://learngitbranching.js.org





References

1) https://www.atlassian.com/git
2) https://github.com/physiksgamez
3) https://hackernoon.com/understanding-git-fcffd87c15a3
4) https://www.freecodecamp.org/news/git-reverting-to-previous-commit-how-to-revert-to-last-commit/
5) https://docs.github.com/en/migrations/importing-source-code/using-the-command-line-to-import-source-code/adding-locally-hosted-code-to-github
6) https://www.youtube.com/watch?v=nCI4DxAF3Ak
7) https://www.youtube.com/watch?v=pZGZRjlS2fY
8) https://gist.github.com/brandon1024/14b5f9fcfd982658d01811ee3045ff1e

Online Tutorial

- Go to learngitbranching.js.org
- Do the first 3 activities under the Main tab
- Then do the first several activities under the Remote tab

Play with local and remote

- Try creating local repository, making and committing changes
- Try adding to github, making changes and pull/push
- Try creating a repository on github, clone it, then change and push