

Gitting started with Git

Why Git?

- Code organization
- Sharing code with peers / working with peers on projects
- Backups of all code is VITAL
- Commit history helps you track down bugs
- Can be a great addition to your resume!
- Good for organizing projects with groups
 - automatic merging and easy comparing between versions of code is helpful for collaborations
- Easy to use on multiple machines (e.g. personal computer & server)

Examples of GitHub repos

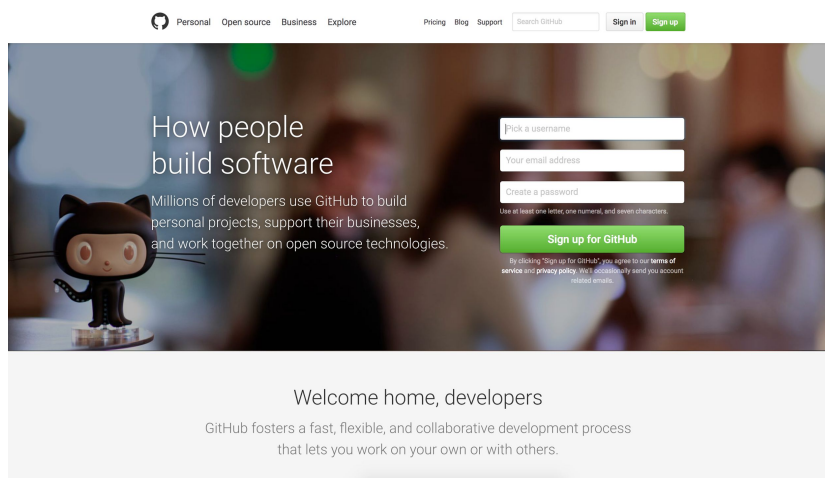
- Examples will demonstrate
 - issue tracking
 - stats & graphs on uploads / traffic / commit history
 - commit history
 - file diff for tracking changes

What you will not learn

- Collaborating with peers requires deciding on a workflow:
 - <https://www.atlassian.com/git/tutorials/comparing-workflows>
- Git is excellent for versioning & maintaining “development” and “main” branches. This is used when you are working on a project that has users in mind (i.e. TensorFlow)
- Pull requests are used as a mechanism for collaborators to contribute to a project:
 - <https://www.atlassian.com/git/tutorials/making-a-pull-request>
- Git is a version control system. GitHub is a website (**or GUI**) for interacting with Git repositories. There are many alternatives to GitHub, which we will not discuss.
- Git does not require a hosting service (e.g. GitHub) and can be set up locally using command line tools.
- Merging code with conflicts and automatic merging

Creating a GitHub repository

You should have already created a GitHub account! First you need to sign in.



The screenshot shows the GitHub homepage. At the top, there is a navigation bar with links for Personal, Open source, Business, Explore, Pricing, Blog, and Support. A search bar and Sign in / Sign up buttons are also present. The main content area features a blurred background image of people. On the left, the GitHub Octocat logo is visible. The text reads: "How people build software" and "Millions of developers use GitHub to build personal projects, support their businesses, and work together on open source technologies." On the right, there is a sign-up form with fields for "Pick a username", "Your email address", and "Create a password". Below these fields is a green "Sign up for GitHub" button. A small note below the button states: "Use at least one letter, one numeral, and seven characters." At the bottom of the sign-up form, it says: "By clicking 'Sign up for GitHub', you agree to our terms of service and privacy policy. We'll occasionally send you account related emails." Below the sign-up form, there is a section titled "Welcome home, developers" with the text: "GitHub fosters a fast, flexible, and collaborative development process that lets you work on your own or with others."

Personal Open source Business Explore Pricing Blog Support Search GitHub Sign in Sign up

How people build software

Millions of developers use GitHub to build personal projects, support their businesses, and work together on open source technologies.

Pick a username

Your email address

Create a password

Use at least one letter, one numeral, and seven characters.

Sign up for GitHub

By clicking "Sign up for GitHub", you agree to our terms of service and privacy policy. We'll occasionally send you account related emails.

Welcome home, developers

GitHub fosters a fast, flexible, and collaborative development process that lets you work on your own or with others.

GitHub Demo

- New 'Learning Python' repository
 - .gitignore
- Clone new repository locally
 - Same steps to clone someone else's repository!
- Create a file, push it to the repository
 - git status (moving items to .gitignore)
- Edit the file, make a commit
 - git add
 - git commit -m
 - git push
- Review commits
 - git log
- Pull down changes made from another computer
 - Edit README.md, then do a git pull
- Modify file & push it up again
 - git diff

A few key concepts explained

- ‘[git](#)’ is open-source software for managing a [version-controlled code repository](#). There are [alternatives](#) to git, but git has the largest community.
- ‘[github](#)’ is a website for interfacing with git repositories and hosting them online. Again, there are [alternatives](#) to github, but github is the most popular.
- You can use `git clone` to set up a local *mirror* of a repository. This command will set up a git repository on your local machine that can sync with a *remote* repository. The remote repository can exist online (e.g. one you create on github.com), on a server, or on another one of your personal machines.

A few key concepts explained

- tracking and staging files
 - For untracked files, using the `git add` command informs your repository that you wish to start tracking a file
 - For files that are already being tracked, `git add` is used to inform the repository that you would like to *stage* this file for a commit.
 - Here we describe one command to do two actions, although really they are the same action. You can `git add` an untracked file, which says “I want you to track this file, and I want to stage it for a commit.” or you can `git add` a tracked file, which says “I want you to stage this *already tracked* file for a commit.”. Either way, `git add` is telling the repository that we want to stage a file to be committed.

A few key concepts explained

- Committing changes

- Once you have staged a file (via `git add`), your next step is to perform the commit. These two steps are typically done immediately following one another, and for the most basic use cases could probably be wrapped up into the same command. However, as you get more comfortable with git you will find that there are times when you want to *stage* a file but not *commit* the file right away.
- Using `git commit -m "message"` will tell your *local* repository that you want to make a log of all changes made in all of the staged files. The associated message is meant to be a summary of the changes you have made. Git will do the work from here - it will determine what has changed in your staged files from the last time you made a commit to those files. It will then update the local repository to include your latest changes and associate those changes with your given message.

Git resources

- Excellent set of tutorials on the git website:
<https://git-scm.com/book/en/v2/Getting-Started-Summary>
- Several useful tutorials: <https://www.atlassian.com/git/tutorials>
- Another guide / tutorial site: <http://rogerdudler.github.io/git-guide/>
- Getting started with git: <https://guides.github.com/activities/hello-world/>
- Connecting multiple remote repositories:
<https://git-scm.com/book/en/v2/Git-Basics-Working-with-Remotes>
- How-to & why for git branching:
<http://nvie.com/posts/a-successful-git-branching-model/>
- git website: <https://git-scm.com/>