

# Git: Getting Started

2025-08-25

## Table of contents

|  |   |
|--|---|
| Installation & Configuration . . . . . | 1 |
| Setup Duke GitLab Account . . . . .    | 2 |
| SSH Key Authentication . . . . .       | 3 |
| Git Tutorials . . . . .                | 3 |

## Installation & Configuration

- [Git](#) - version control software
  - Note - MacOS usually comes with `git` pre-installed, so there is no need to install anything else.
- After installing `git`, please configure it: [Getting Started - First-Time Git Setup](#)
- The setup steps above should create a `~/.gitconfig` file that has some barebones configuration options. We can expand that a bit by manually editing it with the following content:

```
[user]
  name = your first and last name
  email = yournetid@duke.edu
[alias]
  lg = log --graph --pretty=format:'%Cred%h%Creset -%C(yellow)%d%Creset %s %Cgreen(%cr) %C'
[core]
  editor = nano
  autocrlf = input
  preloadindex = true
  FileMode = false
[color]
```

```

status = auto
branch = auto
interactive = auto
diff = auto
ui = auto
[push]
  default = current
  autoSetupRemote = true
[pull]
  rebase = false
[merge]
  ff = no
[init]
  defaultBranch = main

```

- The above configuration will:
  - Set your name and email address for commits
  - Set up a `git lg` alias that will show a nice graph of the commit history
  - Change the default editor for making commit messages from `vi` to `nano`. (Note, you can change this to something else.)
  - Set the default branch name to `main` (instead of the legacy `master` name). *It is important to do this for our grading scripts to work properly.*
- This is a good tutorial on using `git` within VS Code: [Using Git source control in VS Code](#)

## Setup Duke GitLab Account

While [GitHub](#) is a very popular public git repository hosting site, we will be using Duke's GitLab server. Please make sure you can log into <https://gitlab.oit.duke.edu> using your Oath2 NetID authentication.

### Warning

This is **not** gitlab.com!

### Caution

Do **not** edit files in remote git repositories using the GitLab web interface!! Only work with your repository files using your local clone of the repository.

## SSH Key Authentication

We will use SSH keys to authenticate us on GitLab to be able to `clone/push/pull` from GitLab without needing to always enter your username and password.

To setup an SSH key and add it to your GitLab profile:

### Warning

The guide below will reference `github.com`; you want to substitute `gitlab.oit.duke.edu`.

#### 1. [Generate a New SSH Key](#)

- While your private key is most secure by encrypting it with a passphrase, you will need to either enter that passphrase everytime the key is used, or you will need to add it to your local credential manager (i.e., `keychain`) or `ssh-agent` to keep it unlocked.
- If you are just using this SSH key for this class, you can use an empty passphrase and not need to worry about needing to deal with the password management.

#### 2. [Add your SSH key to your GitLab Profile](#)

### Warning

The guide above will reference `gitlab.com`; you want to substitute `gitlab.oit.duke.edu`.

## Git Tutorials

We are going to cover `git` extensively in class, but it can really help to review some tutorials ahead of time to get familiar with the nomenclature and high-level overview of the workflow. If you have never used `git` before, I would recommend reviewing the following tutorials:

- [Git Immersion](#)
- [Learn Git Branching](#)
- [Think Like \(a\) Git](#)
- [Git Tutorial for Beginners: Learn Git in 1 Hour](#)

More comprehensive `git` documentation can be found at <https://git-scm.com/doc>.