CS400

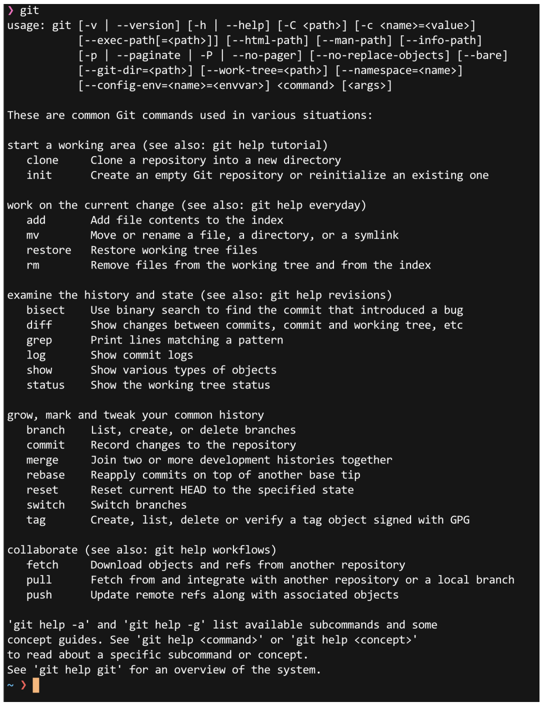
Git Setup

Walkthrough

Install Git

1. **MacOS, Google Compute Engine Virtual Machine, Linux Users** You should already have Git installed on your computer. You can open the “Terminal” app and type in *“git”* to check this. It should display the following (picture on the right):

   If it says “command not recognized”, you can follow this link for MacOS: https://git-scm.com/download/mac. On Ubuntu Linux, you can type “*sudo apt update && sudo apt install git*”.

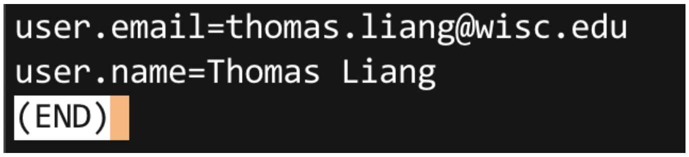


```
❯ git
usage: git [-v | --version] [-h | --help] [-C <path>] [-c <name>=<value>]
           [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
           [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--bare]
           [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
           [--config-env=<name>=<envvar>] <command> [<args>]

These are common Git commands used in various situations:

start a working area (see also: git help tutorial)
   clone     Clone a repository into a new directory
   init      Create an empty Git repository or reinitialize an existing one

work on the current change (see also: git help everyday)
   add       Add file contents to the index
   mv        Move or rename a file, a directory, or a symlink
   restore   Restore working tree files
   rm        Remove files from the working tree and from the index

examine the history and state (see also: git help revisions)
   bisect    Use binary search to find the commit that introduced a bug
   diff      Show changes between commits, commit and working tree, etc
   grep      Print lines matching a pattern
   log       Show commit logs
   show      Show various types of objects
   status    Show the working tree status

grow, mark and tweak your common history
   branch    List, create, or delete branches
   commit    Record changes to the repository
   merge     Join two or more development histories together
   rebase    Reapply commits on top of another base tip
   reset     Reset current HEAD to the specified state
   switch    Switch branches
   tag       Create, list, delete or verify a tag object signed with GPG

collaborate (see also: git help workflows)
   fetch     Download objects and refs from another repository
   pull      Fetch from and integrate with another repository or a local branch
   push      Update remote refs along with associated objects

'git help -a' and 'git help -g' list available subcommands and some
concept guides. See 'git help <command>' or 'git help <concept>'
to read about a specific subcommand or concept.
See 'git help git' for an overview of the system.
~ ❯
```

2. **Windows Users** You will need to install Git from this link: https://git-scm.com/download/win . Afterwards, you can navigate to the “Terminal” or “Powershell” app and type in “git”. If it says “command not recognized”, log out or restart your computer. Otherwise, it should display the same output as the above image.

Configuring your Git Username and Email

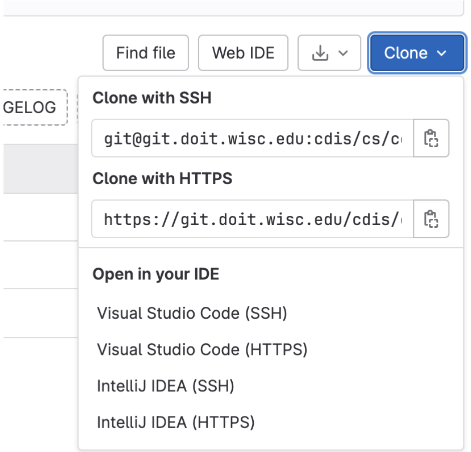
1. Git uses your Git local configuration to understand what name and email was used to commit to the repository. We will now learn how to configure this.
2. Navigate to your Git directory through the Terminal
3. Set your name: “git config --global user.name <your full name>” (Full name with GitLab)
4. Set your email: “git config --global user.email <your email>” (School email address with GitLab)
5. If you are using a different user.name and user.email across your computer, you can remove the “--global” flag to set only for the current Git repository
6. To check that you have done this correctly, type “git config --list” inside your git repository and you should see something like this amongst the text:



```
user.email=thomas.liang@wisc.edu
user.name=Thomas Liang
(END)
```

Cloning your Gitlab Repository

1. Create a new token for cloning your repository here: https://git.doit.wisc.edu/-/profile/personal_access_tokens. (Tick all the boxes) Do **NOT** delete your old token.



```
git@git.doit.wisc.edu:cdis/cs/c
https://git.doit.wisc.edu/cdis/
```

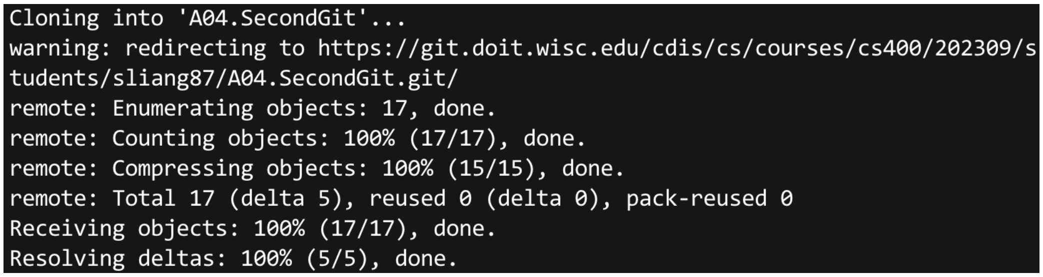
2. Once you have your token, record it so that you do not have to regenerate one every time, but keep this token private like your other passwords.
3. Find the button “Clone” on your GitLab page
4. Press the clipboard copy button on the right of “Clone with HTTPS”
5. You can paste this link anywhere at the moment as you will need to modify it.
6. In between “https://” and “git” you need to type in “oauth2:<your token>@”
7. For example, if this is my link:

   https://git.doit.wisc.edu/cdis/cs/courses/cs400/202309/students/sliang87/A04.SecondGit.git

   I need to edit it such that it becomes:

   https://oauth2:<my-token>@git.doit.wisc.edu/cdis/cs/courses/cs400/202309/students/sliang87/A04.SecondGit.git

8. Paste your modified link in the Terminal and you should get an output like this:



```
Cloning into 'A04.SecondGit'...
warning: redirecting to https://git.doit.wisc.edu/cdis/cs/courses/cs400/202309/s
tudents/sliang87/A04.SecondGit.git/
remote: Enumerating objects: 17, done.
remote: Counting objects: 100% (17/17), done.
remote: Compressing objects: 100% (15/15), done.
remote: Total 17 (delta 5), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (17/17), done.
Resolving deltas: 100% (5/5), done.
```