

*This document is for the hands-on part of the workshop series: Working with Command Line and Collaborating with Git*

## Part 1. Introduction to Git and GitHub

### Slides

Please download or view slides following this link:

<https://docs.google.com/presentation/d/1QG-Ax-jETaHD0DL9wdzZcuMNdT7REQnuPCX2i9AKETI/edit?usp=sharing>

### Outline:

#### [Session 1: \[Bash\] Introduction, fundamentals](#)

- How computer interfaces have changed over time
- Start moving around your computer's folders in the terminal
- Create files, folders and placeholders
- How to search for files

#### [Session 2: \[Bash\] Working with files, intro to loops and shell scripting](#)

- Inspect and transform files
- How to write loops
- Create a runnable script

#### [Session 3: \[Git\] Introduction to Git and GitHub](#)

- Understand version control, Git, and GitHub
- Build a mental model of the Git system
- Learn and practice basic Git commands



#### [Session 4: \[Bash\] Loops recap + Bash scripting for Research tasks automation](#)

- Write scripts that process multiple files
- Run or schedule scripts for data processing or analysis
- Use AI-powered chatbots to design and write bash scripts

#### [Session 5: \[Git\] Collaborating on GitHub](#)

- Understand basic collaboration workflows using Git and GitHub
- Apply Git commands and workflows in a demo repository
- Learn best practices for collaboration with Git

### Hands-on

**Important note:** Please follow [these instructions](#) to install **Bash Shell and Git** ahead of time, especially if you are using a Windows computer. Make sure to [download the file \(cookie\\_recipes.txt\)](#) that we will be using during the workshop.

### Refresher: Manipulate files with Bash commands

Create a new directory called `cookbook` for the project in `Desktop` ; enter that directory; and check that you are in the right directory.

```
$ mkdir ~/Desktop/cookbook
$ cd ~/Desktop/cookbook/
$ pwd
$ ls -a
```

Move downloaded file to the newly created directory

```
$ mv ~/Downloads/cookie_recipes.txt .
$ ls
```

Open file to edit

```
$ open cookie_recipes.txt
```

Or

```
$ nano cookie_recipes.txt
```

## Creating Git repository

Now, let's convert this project (the `cookbook` directory) into a git repository.

Before we do that, let's make sure our git is set up properly.

```
$ pwd
$ ls -a
$ git --version    # check that git is installed, and the version installed
```

```
$ git status
"fatal: not a git repository (or any of the parent directories): .git"

$ git init
Initialized empty Git repository in /Users/luajinw/Desktop/cookbook/.git/
```

## What's happening?

```
$ ls -a
$ ls -F .git
```

```

bash-3.2$ ls -a
.          .git
..         cookie_recipes.txt
bash-3.2$ ls -F .git
COMMIT_EDITMSG  description  hooks/      info/       objects/
config         HEAD        index       logs/       refs/

```

.git is a hidden folder that is the core of the Git project. It's the **local repository** that stores all the metadata, history, configuration, and objects (files and changes) for the project.

## Tracking changes

```
$ git status
```

```

On branch main

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        cookie_recipes.txt

nothing added to commit but untracked files present (use "git add" to track)

```

At this step git repository empty, meaning that it's not tracking / indexing any files.

Now let's tell Git what files to keep track of, and commit the change.

```

$ git add cookie_recipes.txt    # add files to the staging area
$ git status

```

```

On branch main

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   cookie_recipes.txt

```

```

$ git commit -m "add cookie_recipes file"
$ git status

```

```

bash-3.2$ git status
On branch main
nothing to commit, working tree clean

```

## Examining changes

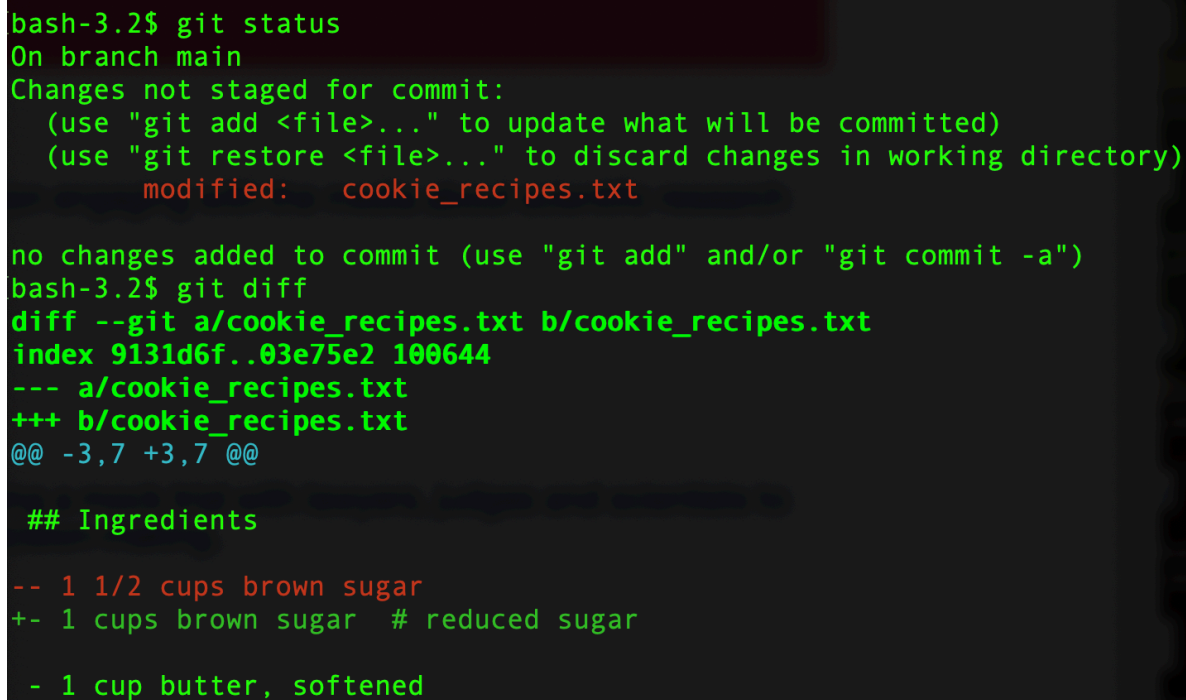
Now, let's make some changes to the `cookie_recipes.txt` file, and see how Git handles these changes.

First, use nano or any other text editor to change sugar amount from 1 1/2 cups to 1 cup. Save changes.

```
$ nano cookie_recipes.txt
$ cat cookie_recipes.txt
```

```
$ git status
$ git diff
```

git diff shows the differences between the current state of the file and the most recently saved version

A terminal window with a dark background and green text. The output of 'git status' shows 'On branch main' and 'Changes not staged for commit: (use "git add <file>..." to update what will be committed) (use "git restore <file>..." to discard changes in working directory) modified: cookie\_recipes.txt'. The output of 'git diff' shows a comparison between 'a/cookie\_recipes.txt' and 'b/cookie\_recipes.txt' at index 9131d6f..03e75e2 100644. It highlights changes in the 'Ingredients' section: '- 1 1/2 cups brown sugar' is replaced by '+ 1 cups brown sugar # reduced sugar', and '- 1 cup butter, softened' remains.

```
bash-3.2$ git status
On branch main
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   cookie_recipes.txt

no changes added to commit (use "git add" and/or "git commit -a")
bash-3.2$ git diff
diff --git a/cookie_recipes.txt b/cookie_recipes.txt
index 9131d6f..03e75e2 100644
--- a/cookie_recipes.txt
+++ b/cookie_recipes.txt
@@ -3,7 +3,7 @@

  ## Ingredients

-- 1 1/2 cups brown sugar
+- 1 cups brown sugar # reduced sugar

- 1 cup butter, softened
```

```
$ git add .
$ git commit -m "reduce sugar content for the kids"
```

## Exploring history

```
$ git status
$ git log
```

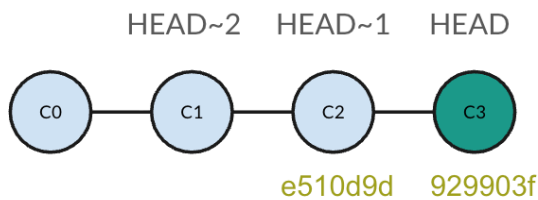
`git log` lists all commits made to a repository in reverse chronological order.

```
bash-3.2$ git log
commit 929903f06d85b17c49bc86a4c0786daa46350ff7 (HEAD -> main)
Author: Huajin Wang <23021719+huajinw@users.noreply.github.com>
Date: Mon Feb 9 13:39:31 2026 -0500

    reduce sugar content for the kids

commit e510d9d8c042b076836f21c90f95575b45e93354
Author: Huajin Wang <23021719+huajinw@users.noreply.github.com>
Date: Sat Feb 7 22:06:11 2026 -0500

    add cookie_recipes file
```



```
$ git diff HEAD~1 cookie_recipes.txt
```

```
$ git diff e510d9 cookie_recipes.txt
```

You can use `checkout` to restore to a previous commit.

```
$ git checkout HEAD~1 cookie_recipes.txt
$ cat cookie_recipes.txt
$ git status
```

```
bash-3.2$ git status
On branch main
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   cookie_recipes.txt
```

## Caution: Don't Lose Your HEAD

Don't forget to include the file name when you checkout, otherwise Git would think that you want to move the HEAD position.

```
$ git checkout HEAD~1 # Caution! this line leads to detached head.
```

To recover / reattached the head:

```
$ git switch -  
or  
$ git checkout main
```

```
bash-3.2$ git checkout HEAD~1  
Note: switching to 'HEAD~1'.  
  
You are in 'detached HEAD' state. You can look around, make experimental  
changes and commit them, and you can discard any commits you make in this  
state without impacting any branches by switching back to a branch.  
  
If you want to create a new branch to retain commits you create, you may  
do so (now or later) by using -c with the switch command. Example:  
  
    git switch -c <new-branch-name>  
  
Or undo this operation with:  
  
    git switch -  
  
Turn off this advice by setting config variable advice.detachedHead to fa  
lse  
  
HEAD is now at e510d9d add cookie_recipes file
```