

# Git & GitHub Step-by-Step

Instructions for ORD Workshop | Basel, 02 June 2023

# Install Git

Full instructions are here: <https://github.com/git-guides/install-git>

Option 1: Install using an installer package (requires admin access)

- follow links on the website above for your OS (Windows, Mac or Linux)

Option 2: Install using Homebrew (for MacOS, requires admin access)

- `brew install git`

Option 3: Download GitHub Desktop <https://desktop.github.com/> - installs Git as well

- best option if you don't have admin access on your machine

Mac OS Notes:

Terminal may say xcode-select command line developer tools are required for git. This is a lie. You can skip installing the command line developer tools and use the installer package instead.

If you get a warning that the software package can't be opened because it's from an unidentified developer, go to **System Preferences > Security & Privacy > General** > find the warning about git package and click **Open Anyway**, then **Open**. I promise it's safe.

After installation, you may need to close your Terminal window and open a new window to have the command 'git version' show the version number.

# Set up GitHub

Create GitHub account

<https://github.com/> -> Sign Up

Notes:

- Choose the free plan.
- You can select “collaborative coding” as your use case.

Optional:

- Set up an SSH key for connecting to GitHub (not required for the workshop but very helpful for regular use). <https://docs.github.com/en/authentication/connecting-to-github-with-ssh>

# Configure Git

## Via Command Line

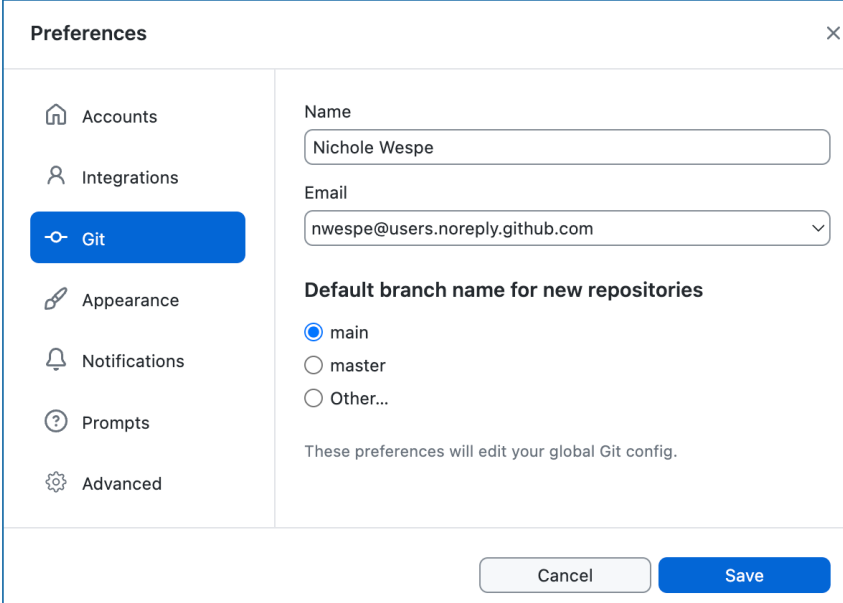
Substitute italicized text with your information

```
git config --global user.name "Your Name"
```

```
git config --global user.email "your.email@domain.com"
```

## Via GitHub Desktop application

**GitHub Desktop > Settings > Git**



The screenshot shows the 'Preferences' window in GitHub Desktop, with the 'Git' tab selected. The left sidebar contains icons for Accounts, Integrations, Git (highlighted), Appearance, Notifications, Prompts, and Advanced. The main area on the right is titled 'Name' and 'Email'. The 'Name' field contains 'Nichole Wespe'. The 'Email' field is a dropdown menu showing 'nwespe@users.noreply.github.com'. Below these fields, the section 'Default branch name for new repositories' has three radio button options: 'main' (selected), 'master', and 'Other...'. A note at the bottom states 'These preferences will edit your global Git config.' At the bottom right, there are 'Cancel' and 'Save' buttons.

Category	Field	Value
User Information	Name	Nichole Wespe
	Email	nwespe@users.noreply.github.com
	Default branch name for new repositories	main

# Clone a Repo

<https://github.com/nwespe/ord-workshop-2023>

## Via Command Line

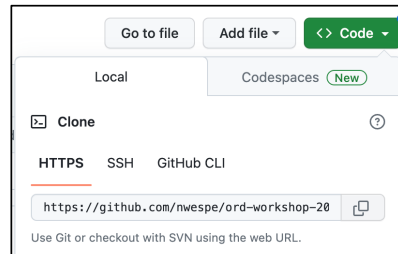
1. Go to repo on GitHub and copy URL (HTTPS or SSH)
2. In terminal app, navigate to where you want the repo to be and run command:

HTTPS:

```
git clone https://github.com/nwespe/ord-workshop-2023.git
```

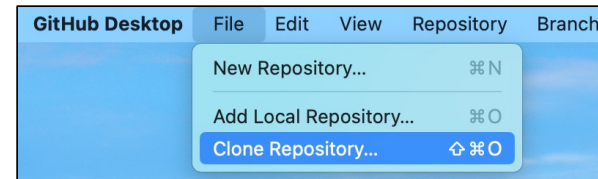
SSH:

```
git clone git@github.com:nwespe/ord-workshop-2023.git
```

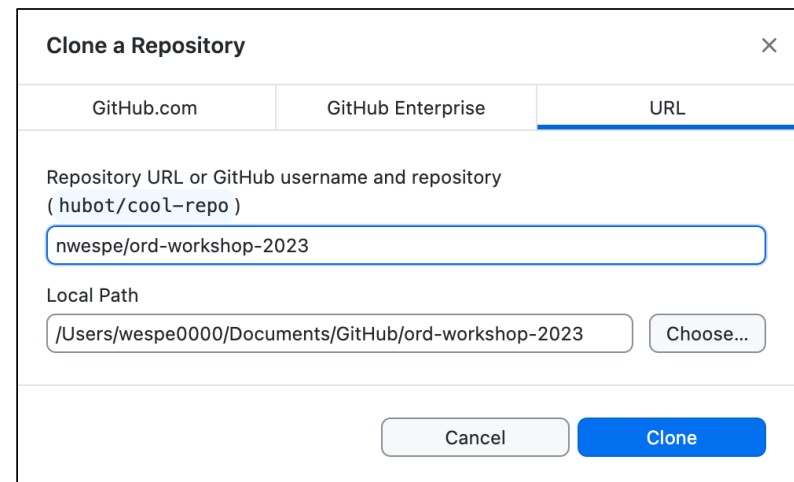


## Via GitHub Desktop

1.



2.



# Checking repo status

Tells you what branch you are on and what files have changed

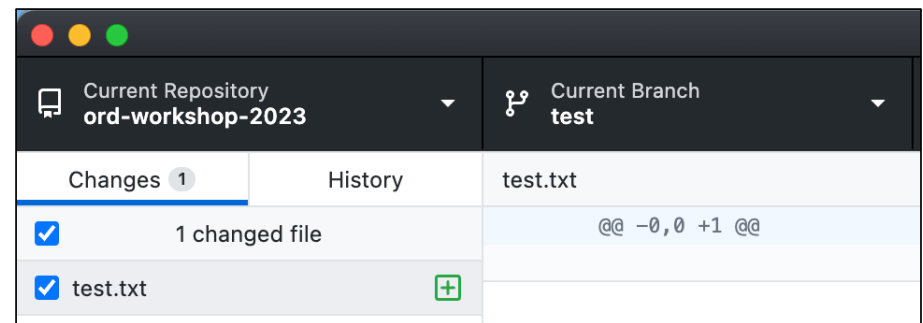
Via Command Line

```
git status
```

Example output:

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

Via GitHub Desktop



The app always shows which branch you are on, which files have changed and what changes were made.

The selected files will be added to git when you make a commit.

# Create a new branch

Use your GitHub username as the branch name

Via Command Line

Substitute <branchname> with your username

Option 1

```
git branch <branchname>  
git switch <branchname>
```

Option 2

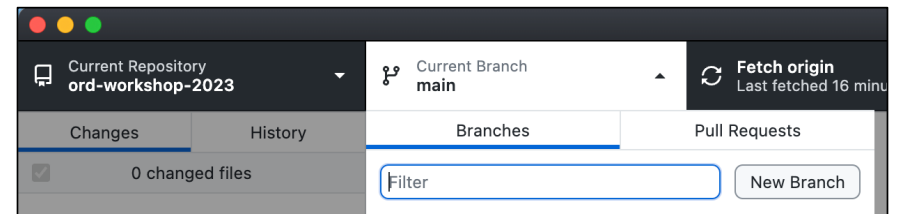
```
git switch -c <branchname>
```

Option 3

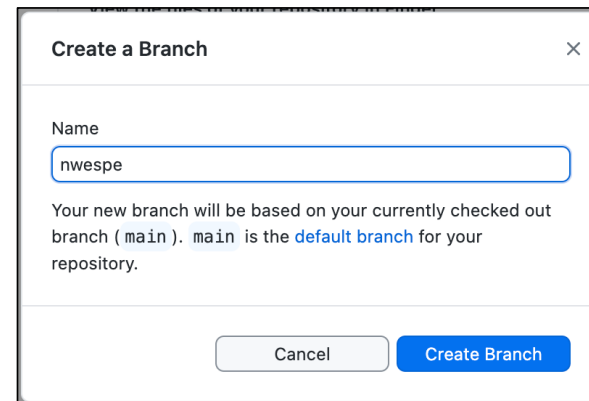
```
git checkout -b <branchname>
```

Via GitHub Desktop

1.



2.



# Make changes

- Everyone:
  - Open the README.md file in a text editor application.
  - Add your name under Participants.
- If you have a recipe file:
  - Name your file by just the recipe, in lowercase with underscores (“snake case”), e.g. “chocolate\_chip\_cookies.txt”.
  - Make sure your file includes a line at the top with your name.
  - Add your text file to the ‘recipes’ folder.
- If you don’t have a recipe file prepared but want to add one:
  - Create a text file and write your best recipe for chips and salsa.
  - Follow the instructions above for adding the recipe file. If multiple people do this, you will work together to create a consensus recipe as the branches get merged into main.



# Stage and commit changes

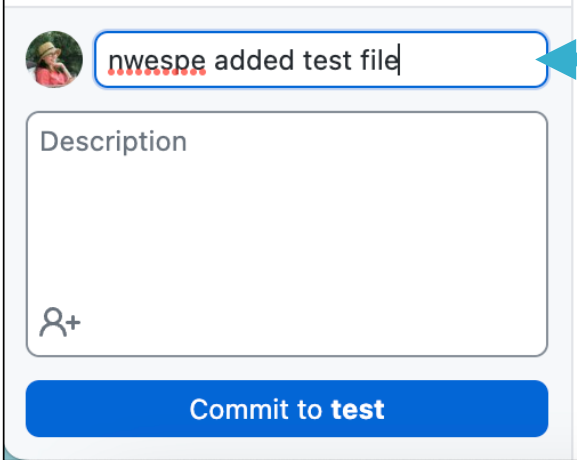
Commit message: “<your username> added recipe file and name to readme”

## Via Command Line

Substitute <filename> with your recipe filename

```
git add README.md  
git add recipes/<filename>  
git commit -m "commit message"
```

## Via GitHub Desktop



The screenshot shows the GitHub Desktop commit window. At the top left is a user profile picture. To its right is a text input field containing the commit message 'nwespe added test file'. Below this is a larger text area labeled 'Description'. At the bottom left of the description area is a plus icon with a person silhouette. At the bottom center is a blue button labeled 'Commit to test'.

Commit message  
goes here

# Set up personal access token on GitHub

- If you haven't set up SSH for connecting to GitHub, you need to get a personal access token for authenticating on the command line. Authentication is required for pushing to a remote repo.
- If you are using GitHub Desktop, you do not need to do this.
- On GitHub, go to **Settings > Developer settings** (bottom of left-side menu) > **Personal access tokens** and create a token of either type. Copy it to a text file on your laptop to save.
- In the next step ('git push'), you will enter the personal access token as your password for authentication.

# Push branch to remote repo

Note: you must be added as a collaborator to the repo before pushing

## Via Command Line

```
git push --set-upstream origin  
<branchname>
```

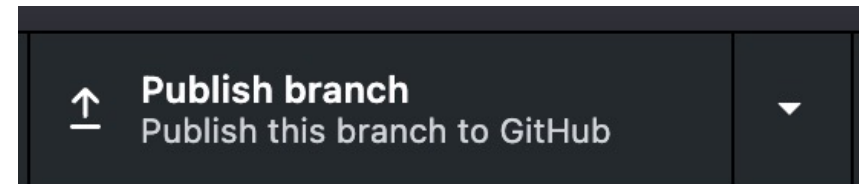
or simply

```
git push
```

if the branch has been previously pushed.

If you run the 'git push' command for a branch that hasn't been connected to the remote repo (i.e., one you created locally), it will give you a friendly error that includes the --set-upstream part of the command and then you can copy-paste the full command.

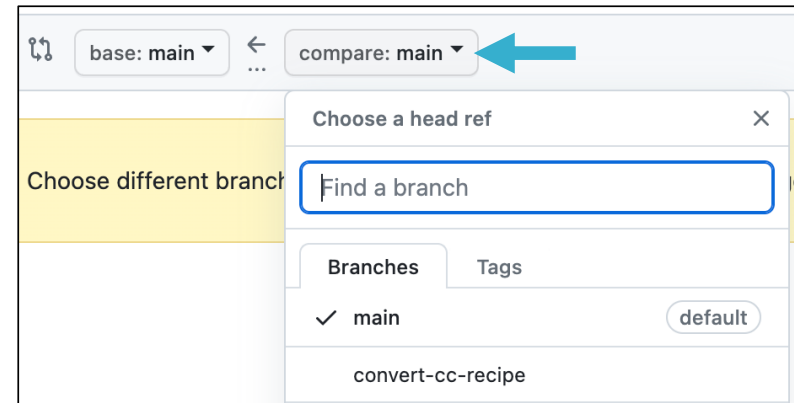
## Via GitHub Desktop



# Create a pull request on GitHub

<https://github.com/nwespe/ord-workshop-2023>

1. Go to **Pull requests** tab
2. Click **New pull request**
3. Select your branch to compare
4. Click **Create pull request**



# After PR is merged, update your local repo

## Via Command Line

### Option 1

```
git fetch
```

### Option 2

```
git pull
```

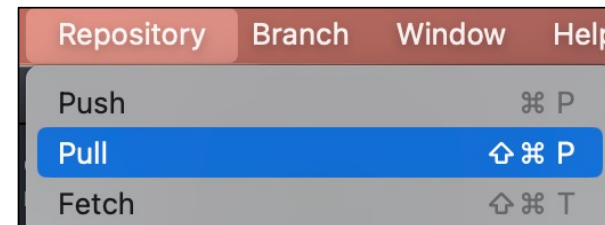
## Via GitHub Desktop

### Option 1

A dark grey button with a circular refresh icon on the left. The text "Fetch origin" is in bold, and "Last fetched 10 minutes ago" is in a smaller font below it.

**Fetch origin**  
Last fetched 10 minutes ago

### Option 2



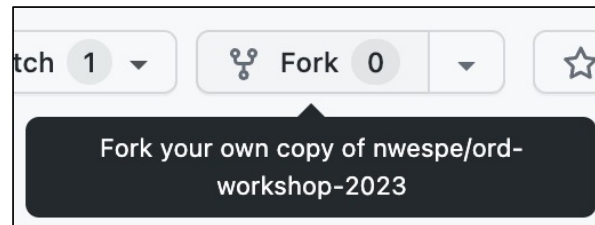
<https://www.theserverside.com/blog/Coffee-Talk-Java-News-Stories-and-Opinions/Git-pull-vs-fetch-Whats-the-difference>

# Fork the repo to your account on GitHub

<https://github.com/nwespe/ord-workshop-2023>

Wait to fork the repo until all PRs are merged into the main branch, so you only have to fork the main branch.

1. Click on Fork



2. Click

Create fork