

R Version Control

Vincent Arumadri

2025-07-08

R Version Control with Git/GitHub

Git/GitHub enables continuous software/code development by tracking code changes to the **initial commit**, tracking who made the changes and enabling code collaboration. Here I will go through the basics on how to step-up **Git/GitHub** for version control with **R/RStudio**. Although the focus is on R/RStudio, the same procedure can be followed by those who use other softwares for data analysis e.g **Python**, **LaTeX**, **Julia** etc.

Required programs

To start off, ensure you have the following programs installed:

1. **Git** – the base program for tracking code changes.
 - **Windows:** You can download Git here. Double-click the downloaded .exe file to start the installation process.
 - **macOS:** First you need to install **Homebrew**. Copy and paste the Homebrew download link from (<https://brew.sh/>) to the Terminal and run.

If you already have Homebrew installed, open Terminal and type:

```
brew install git
```

2. **R** – the base R program
 - **Windows:** You can download R here
 - **macOS:** You can download R here
3. **RStudio** – IDE (integrated development environment) for R
 - **Windows and macOS:** You can download RStudio here
4. **GitHub Desktop** – for version control and collaboration
 - **Windows and macOS:** You can download GitHub Desktop here

Getting started

1. **Sign up for GitHub** – Got to GitHub and create a free account:

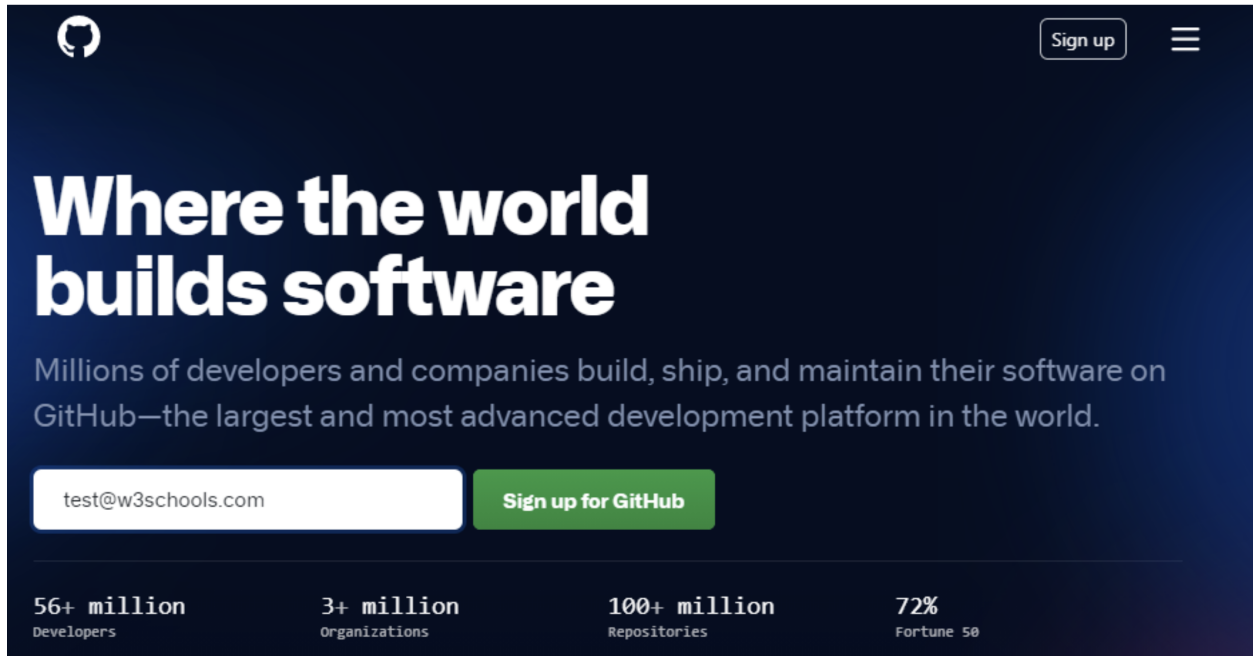
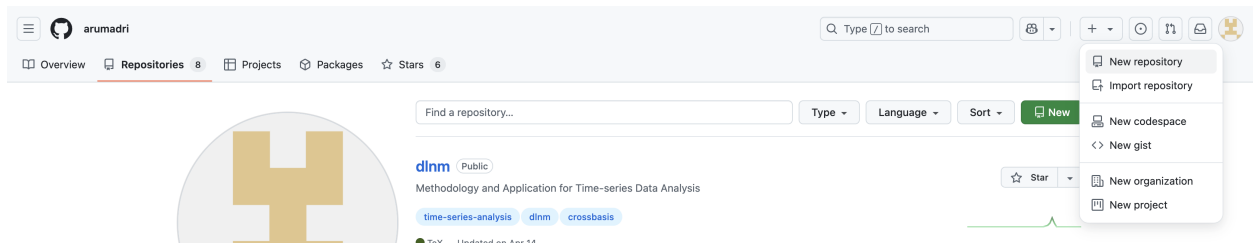


Figure 1: GitHub account setup

2. **Create a Repository** After signing in, click the + button to create a new repository:



3. **Fill in the repository details** (name, description, public/private, etc.) and click Create repository:

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository](#).

Required fields are marked with an asterisk (*).

Owner * **Repository name ***

arumadri / jrc_template

jrc_template is available.

Great repository names are short and memorable. Need inspiration? How about ubiquitous-system ?

Description (optional)

R Version control using GitHub

☒ **Public**
Anyone on the internet can see this repository. You choose who can commit.

☐ **Private**
You choose who can see and commit to this repository.

Initialize this repository with:

☒ **Add a README file**
This is where you can write a long description for your project. [Learn more about READMEs](#).

Add .gitignore

.gitignore template: None

Choose which files not to track from a list of templates. [Learn more about ignoring files](#).

Choose a license

License: None

A license tells others what they can and can't do with your code. [Learn more about licenses](#).

This will set `main` as the default branch. Change the default name in your [settings](#).

You are creating a public repository in your personal account.

Create repository

Figure 2: GitHub new repository details

Now you have set up your **remote repository**!!

4. Clone remote repository to local machine (laptop/desktop)

Inside the repository click the <>Code button and select **Open with GitHub Desktop** in the drop down menu.

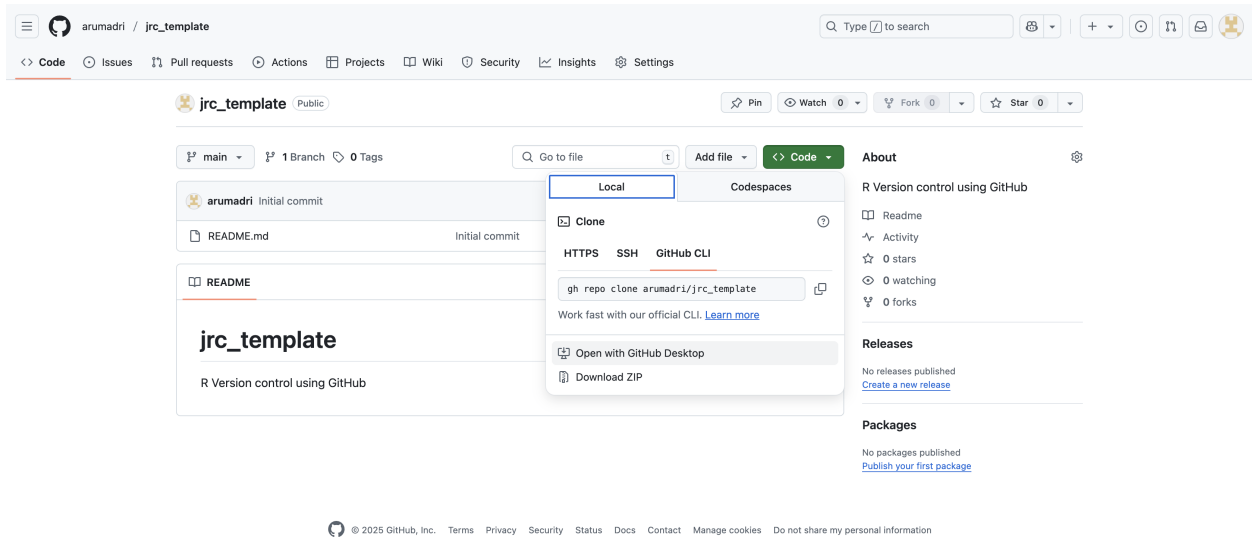


Figure 3: Clone remote to local

5. Select path for local repository

Clone a Repository

GitHub.com GitHub Enterprise **URL**

Repository URL or GitHub username and repository
(hubot/cool-repo)

Local Path

< > jrc_template

Name	Date Modified	Size	Kind
README.md	Today at 13:47	47 bytes	Markdown File

6. Add or (save your new r scripts/files) to this folder

This will initialize the **commit** and **push** prompt in GitHub Desktop to commit and push the changes respectively to the **remote repository**.

These prompts enable changes made to the folder (adding new files) or files (new edits to code) on the **local** machine to be **committed and pushed** to the **remote** for tracking and hence **version control**.

Here I add new files to the **local** folder.

<

>

jrc_template

These changes appear on **GitHub Desktop** ready to be **committed** and **pushed** to the **remote**.

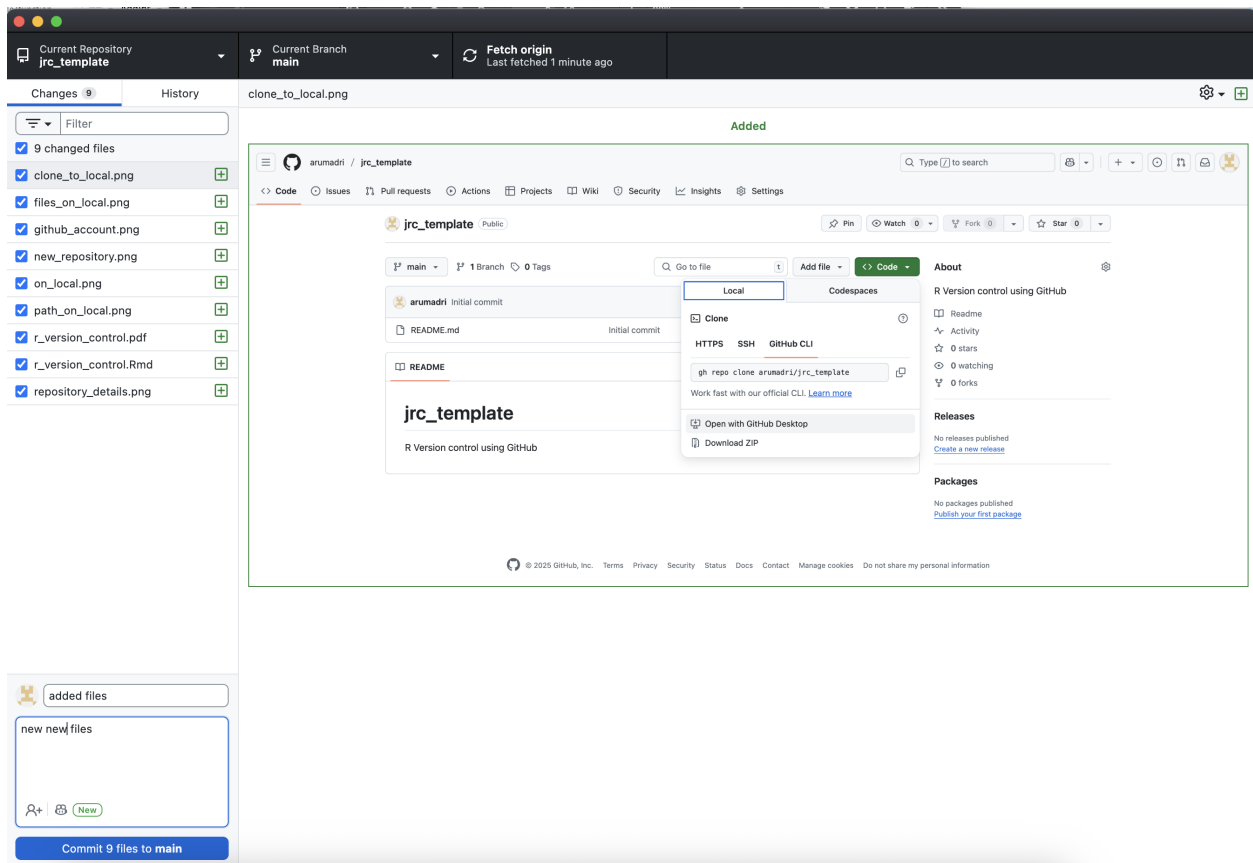
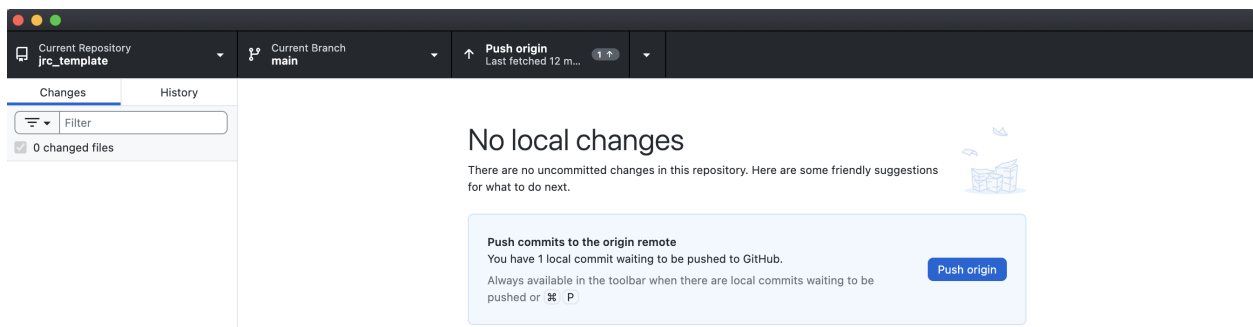


Figure 4: Commit

7. **Push** and **pull** changes to or from **GitHub**. After **committing** the changes, they are ready to be **pushed**.



8. Changes on the **remote** can be **pulled** to the **local**. This is especially useful in projects with multiple contributors to the code. Changes made by one person to the remote can be pulled to the local repository of another person to update their code to the latest version. Here I make some changes on the README file on **remote**, **commit** then **pull** them to **local** .

The screenshot shows the GitHub interface for the repository 'jrc_template' by user 'arumadri'. The top navigation bar includes links for Code, Issues, Pull requests, Actions, Projects, Wiki, Security, Insights, and Settings. The repository name 'jrc_template' is displayed with a 'Public' badge. Below the repository name, there are buttons for Pin, Watch (0), Fork (0), and Star (0). The main content area shows the 'README.md' file being edited. The file content is as follows:

```
1 # jrc_template
2 R Version control using GitHub by JRC
3
```

Below the file content, there is a table of file changes. The table has three columns: file name, action, and time. The changes are as follows:

File Name	Action	Time
README.md	Update README.md	now
clone_to_local.png	added files	7 minutes ago
commit.png	added files	7 minutes ago
files_on_local.png	added files	7 minutes ago
github_account.png	added files	7 minutes ago
new_repository.png	added files	7 minutes ago
on_local.png	added files	7 minutes ago
path_on_local.png	added files	7 minutes ago
push.png	Create push.png	4 minutes ago
r_version_control.Rmd	added files	7 minutes ago
r_version_control.pdf	added files	7 minutes ago
repository_details.png	added files	7 minutes ago

Below the table, there is a section for the README file. The README content is as follows:

jrc_template

R Version control using GitHub by JRC

Contact details

Please email [JRC](#) with any queries relating to this code.

On the right side of the repository page, there is a sidebar with the following sections:

- About**: R Version control using GitHub. Includes links for Readme, Activity, 0 stars, 0 watching, and 0 forks.
- Releases**: No releases published. Includes a link to [Create a new release](#).
- Packages**: No packages published. Includes a link to [Publish your first package](#).

Within **Github Desktop**, these changes will be reflected and a prompt to pull them to your **local** will appear when the application is opened while connected to the internet.

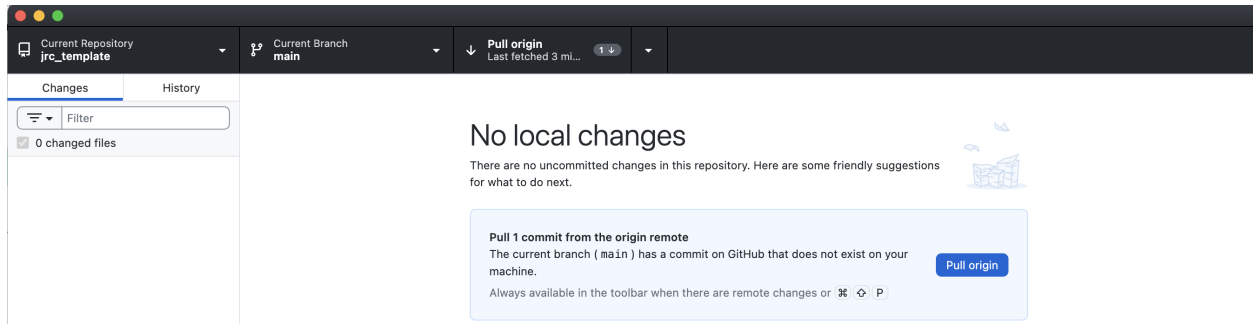


Figure 5: Pull to local

Changes made on the **remote** now **pulled** to the **local** as seen in **RStudio** here.

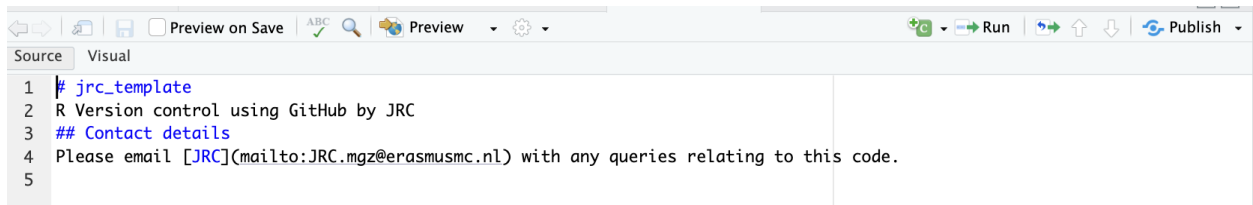


Figure 6: Commit changes on local

We hope you find this helpful and start using version control to keep track of your code changes. Do not hesitate to conduct **JRC** with any queries relating to this code. Good luck!