R Version Control

Vincent Arumadri

2025-11-10

R Version Control with Git/GitHub

Why you need to start using Git/GitHub?

Git/GitHub enables continuous software/code development by tracking code changes (**version control**) to the **initial commit**, tracking who made the changes and enabling code collaboration. Version control also helps to keep your code safe in case your laptop or hard disk with your code is lost or damaged.

Data privacy on GitHub

Project repositories on GitHub can either be **private or public**. A **public repository** is visible to anyone who visits that repository while a **private repository** is only visible to the user and to their invited collaborators (for private repositories where collaboration is needed).

1. Care must be taken by the user to ensure that only approved/final code and data are added to a public repository to avoid breach of data privacy rules.

2. Care must be taken by the user to ensure that before making a private repository public, all data and code do not breach any data privacy rules.

Necessary steps

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. To verify Git installation on your machine, **follow the steps here**. **NB: only available for Windows x64 and ARM64**
   - **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

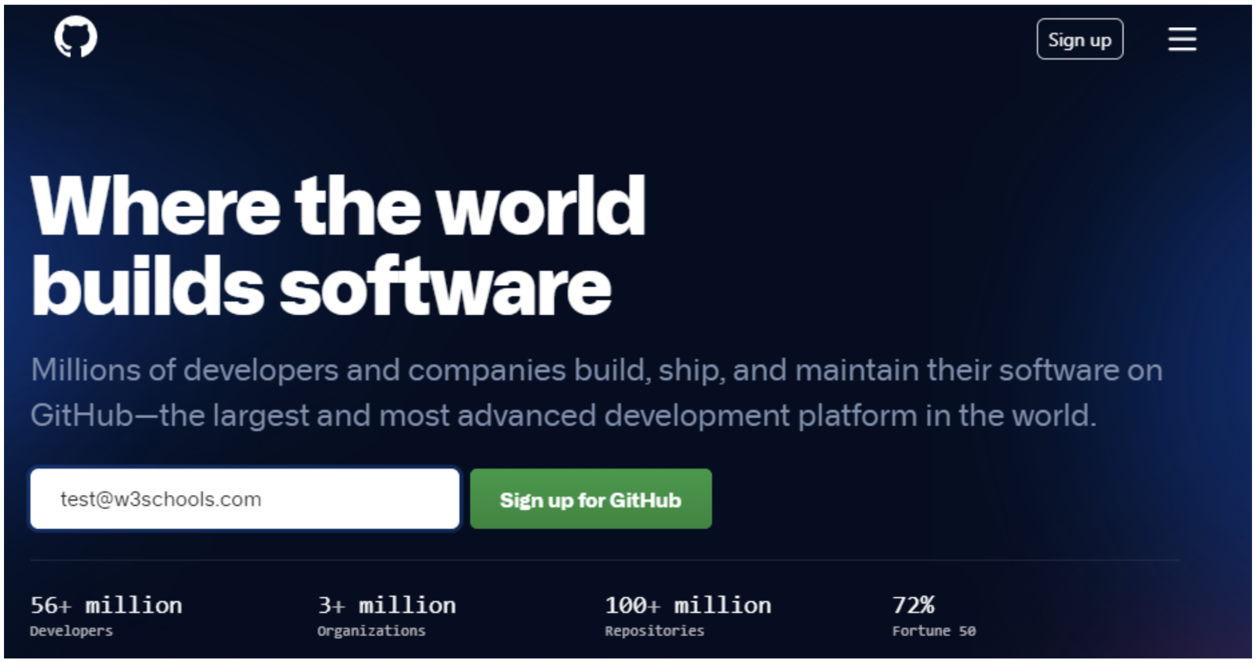
4. **GitHub Desktop** – for version control and collaboration

   - **Windows and macOS**: You can download GitHub Desktop here

Getting started

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

NB: Use a personal email account to sign up if you want to have access to your repositories in the future when you leave a given workplace. Using a work email account to sign up may be necessary for some projects, however, be aware that you may lose access to your repositories in the future when you leave that workplace.



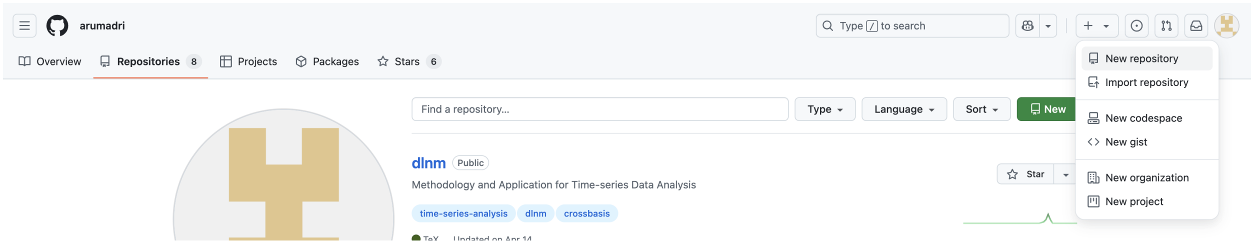
Where the world builds software

Millions of developers and companies build, ship, and maintain their software on GitHub—the largest and most advanced development platform in the world.

56+ million	3+ million	100+ million	72%
Developers	Organizations	Repositories	Fortune 50

Figure 1: GitHub account setup

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



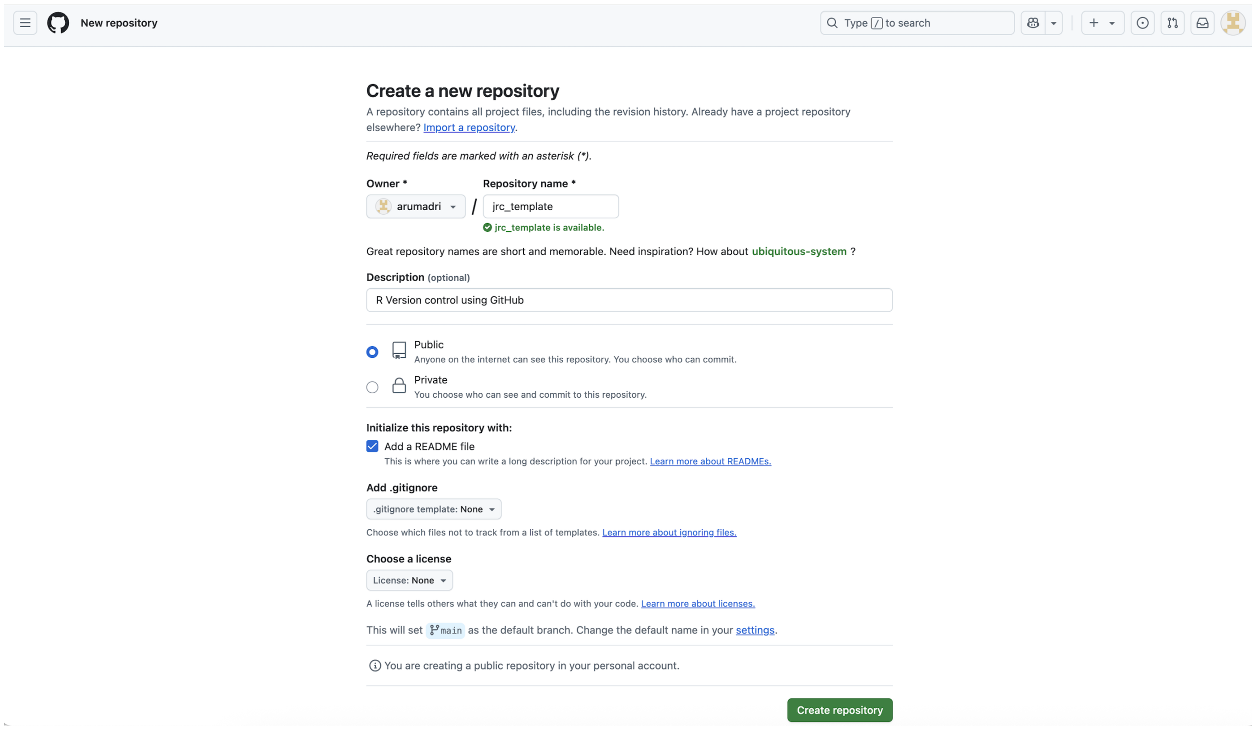
dlnm Public

Methodology and Application for Time-series Data Analysis

time-series-analysis dlnm crossbasis

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

NB: It's essential to check "Add a README file during the step up of your repository.



New 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 * arumadri / Repository name * 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.

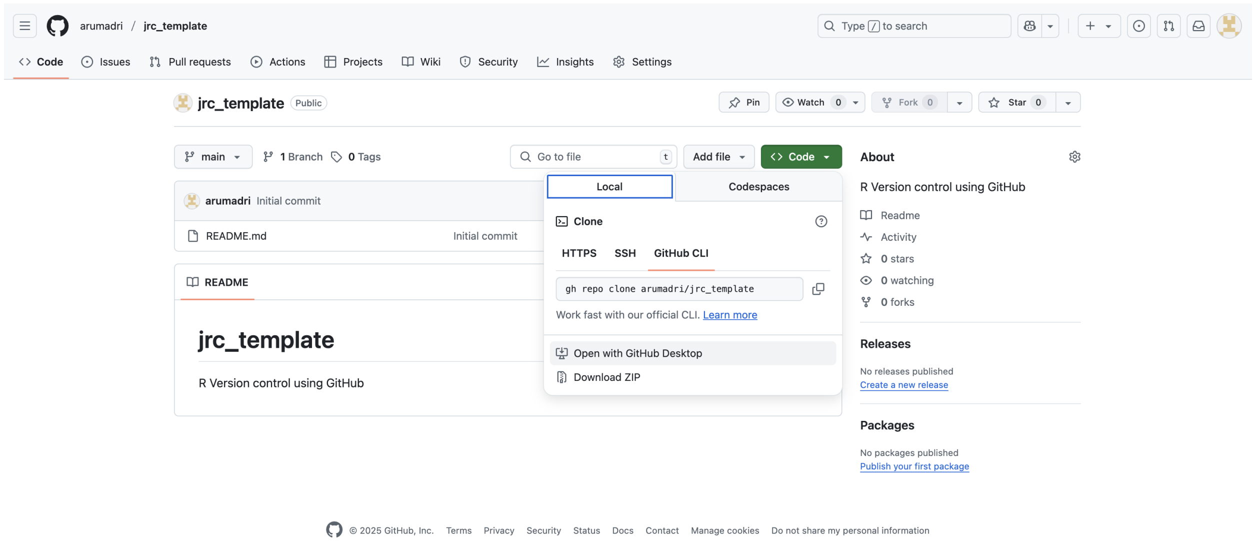
Create repository

Figure 2: GitHub new repository details

Click **Create repository**. Now you have set up your **remote repository!!**

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

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

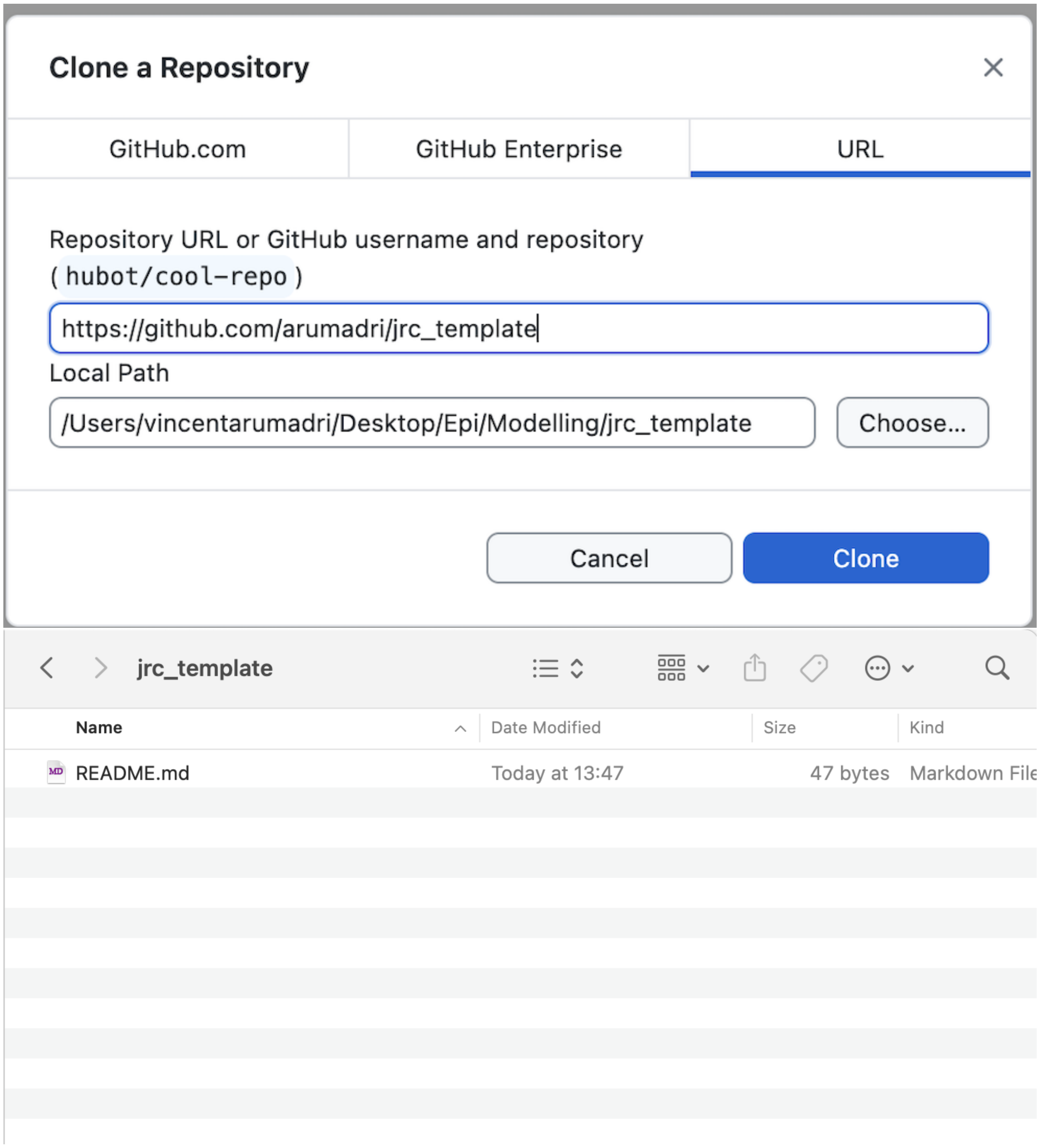


```
gh repo clone arumadri/jrc_template
```

Work fast with our official CLI. Learn more

5. Select path for local repository

NB: Care must be taken to ensure that a local path for a public repository does not include any data that should not be shared publicly. Local paths should be to new folders or current folders where no data or files will be shared publicly accidentally.



Clone a Repository

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

https://github.com/arumadri/jrc_template

Local Path

/Users/vincentarumadri/Desktop/Epi/Modelling/jrc_template

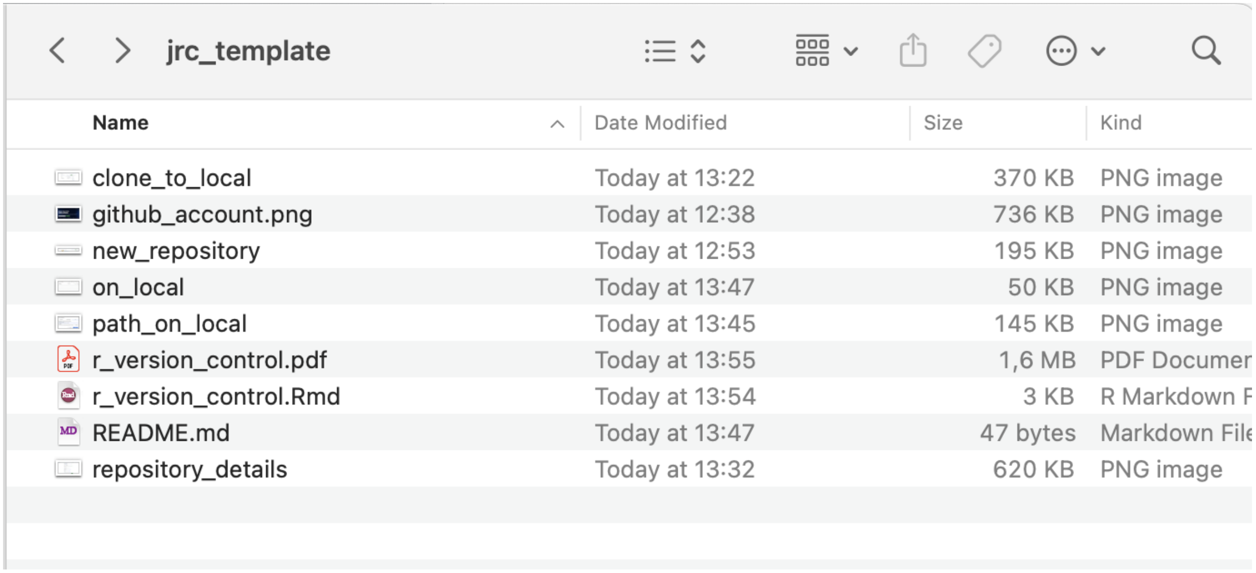
Name	Date Modified	Size	Kind
README.md	Today at 13:47	47 bytes	Markdown File

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**.

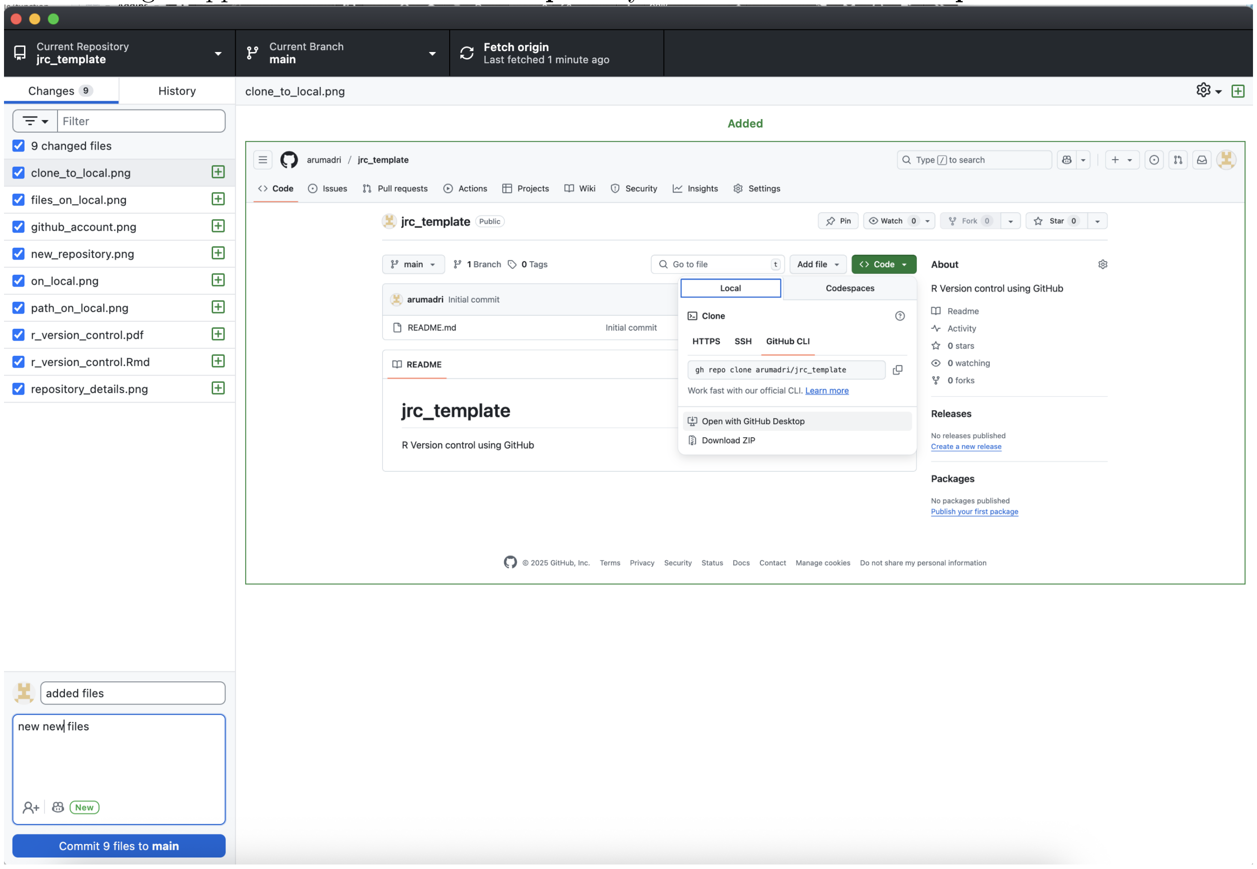
Here I add new files to the **local** folder.



jrc_template

Name	Date Modified	Size	Kind
clone_to_local	Today at 13:22	370 KB	PNG image
github_account.png	Today at 12:38	736 KB	PNG image
new_repository	Today at 12:53	195 KB	PNG image
on_local	Today at 13:47	50 KB	PNG image
path_on_local	Today at 13:45	145 KB	PNG image
r_version_control.pdf	Today at 13:55	1,6 MB	PDF Documer
r_version_control.Rmd	Today at 13:54	3 KB	R Markdown F
README.md	Today at 13:47	47 bytes	Markdown File
repository_details	Today at 13:32	620 KB	PNG image

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



Current Repository jrc_template | Current Branch main | Fetch origin Last fetched 1 minute ago

Changes 9 | History

9 changed files
- clone_to_local.png
- files_on_local.png
- github_account.png
- new_repository.png
- on_local.png
- path_on_local.png
- r_version_control.pdf
- r_version_control.Rmd
- repository_details.png

clone_to_local.png

Added

arumadri / jrc_template

jrc_template Public

main · 1 Branch · 0 Tags

arumadri Initial commit

README.md Initial commit

README

jrc_template

R Version control using GitHub

Clone: HTTPS SSH GitHub CLI

```
gh repo clone arumadri/jrc_template
```

Work fast with our official CLI. Learn more

About

R Version control using GitHub

0 stars · 0 watching · 0 forks

Releases: No releases published

Packages: No packages published

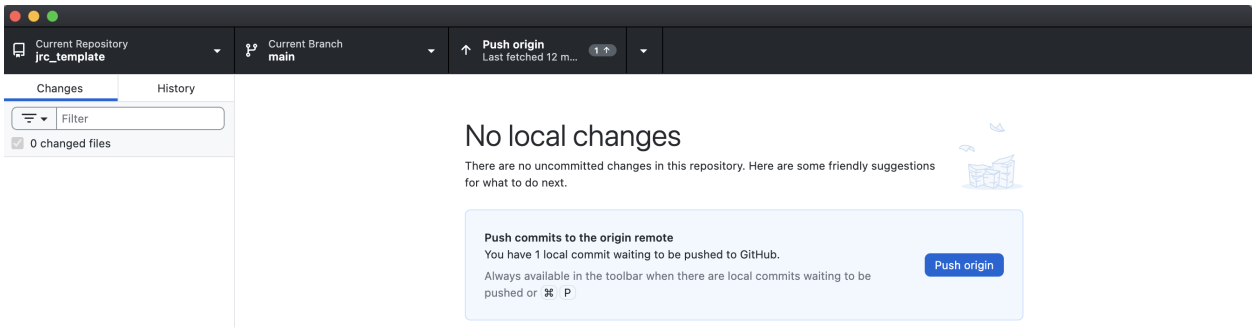
© 2025 GitHub, Inc.

Summary: added files

Description: new new files

Commit 9 files to **main**

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



No local changes

There are no uncommitted changes in this repository. Here are some friendly suggestions for what to do next.

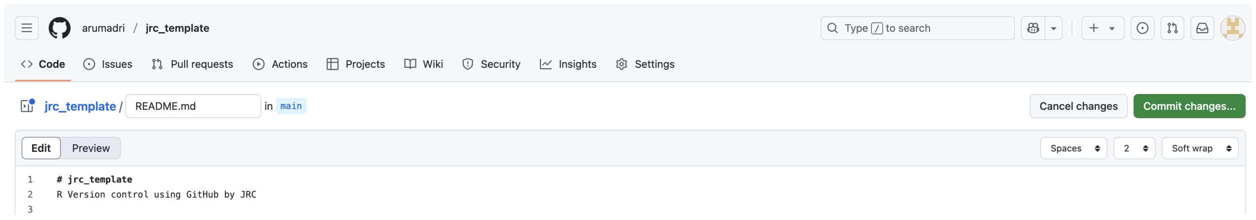
Push commits to the origin remote

You have 1 local commit waiting to be pushed to GitHub.

Always available in the toolbar when there are local commits waiting to be pushed or ⌘ P

Figure 3: Push

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 (**adding JRC contact details**) to the README file on **remote**, **commit** then **pull** them to **local** .

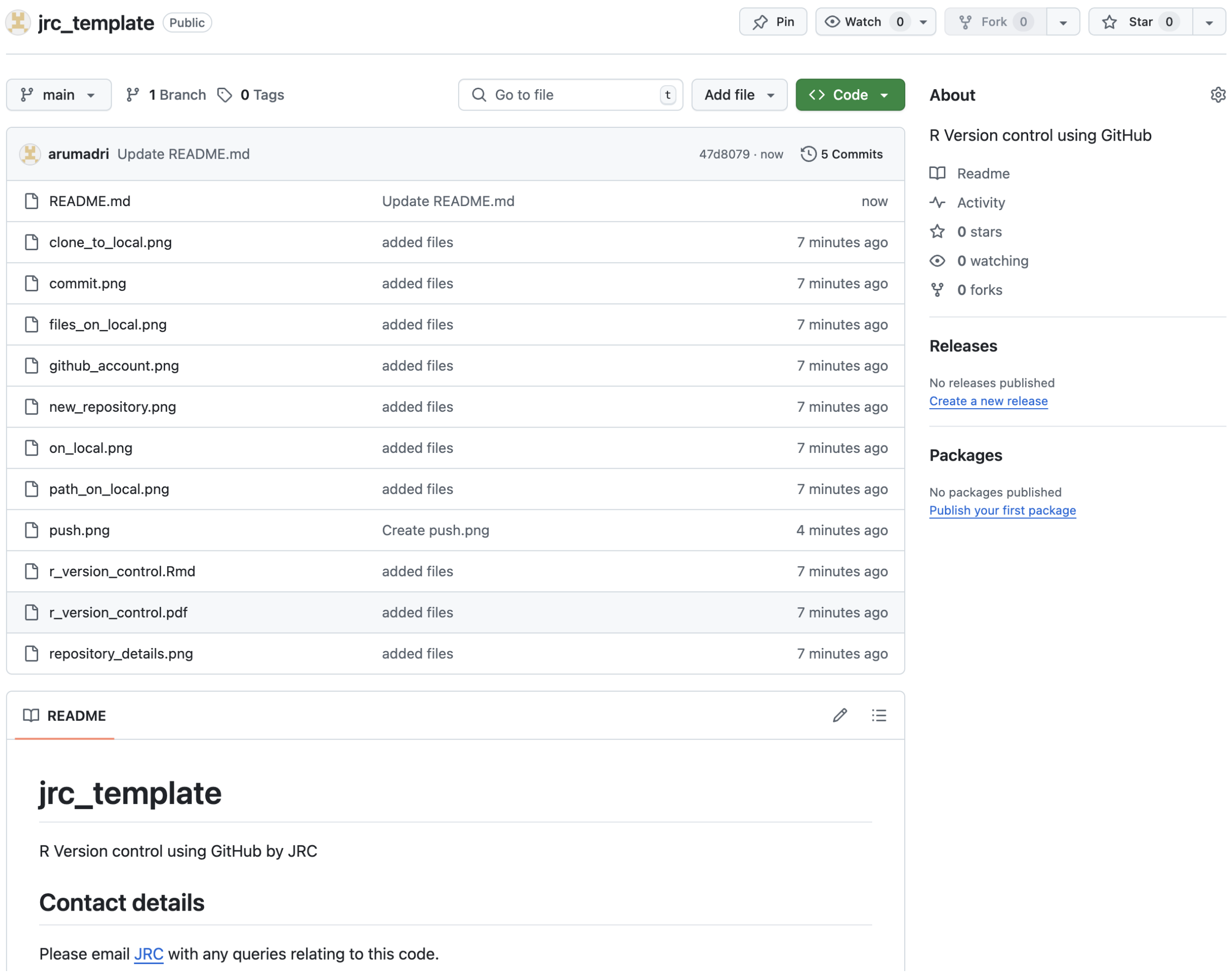


```
# jrc_template
R Version control using GitHub by JRC

```

The new changes (**JRC contact details**) are now added to the **remote repository** but are not yet made on the **local**.

The next step is **pulling** these changes to the **local**.



R Version control using GitHub

arumadri Update README.md 47d8079 · now 5 Commits

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

README

jrc_template

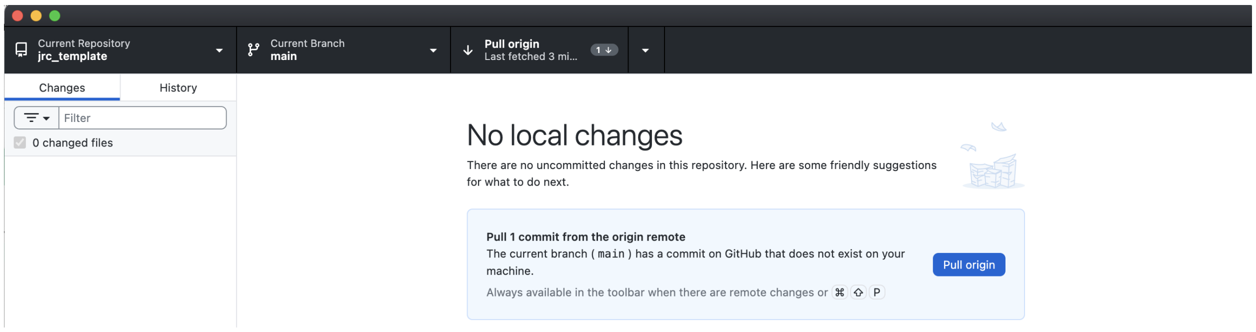
R Version control using GitHub by JRC

Contact details

Please email JRC with any queries relating to this code.

Figure 4: Committed remote

9. Within **Github Desktop**, these changes will be reflected and a prompt to **pull origin** i.e from **remote** to your **local** will appear when the application is opened while connected to the internet.



No local changes

There are no uncommitted changes in this repository. Here are some friendly suggestions for what to do next.

Pull 1 commit from the origin remote

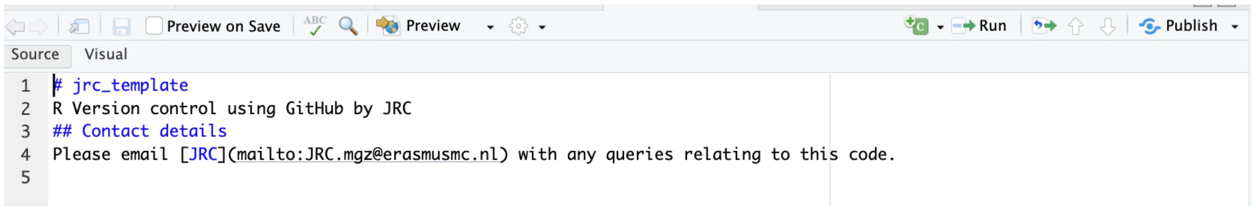
The current branch (main) has a commit on GitHub that does not exist on your machine.

Pull origin

Always available in the toolbar when there are remote changes or ⌘ ⇧ P

Figure 5: Pull to local

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



```
# jrc_template
R Version control using GitHub by JRC
## Contact details
Please email [JRC](mailto:JRC.mgz@erasmusmc.nl) with any queries relating to this code.
```

Figure 6: Commit changes on local

Support

We hope you find this helpful and start using version control to keep track of your code changes. Do not hesitate to contact the **JRC** for any support or queries relating to these steps. Good luck!