

Reproducible science in action

Setting up git for practicals



Google scholar

Franco Marsico

Github: <https://github.com/MarsicoFL>



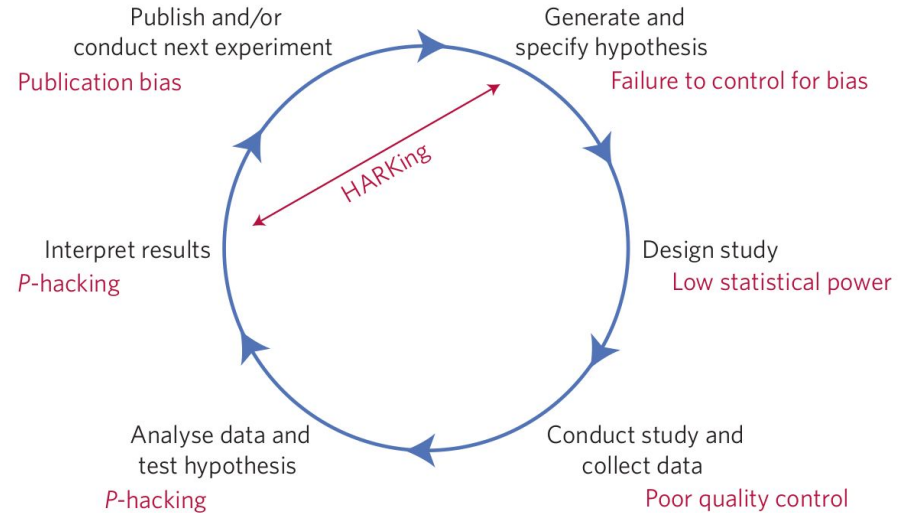
EMBO POPGEN

- What is reproducible science?
- Sharing the code
- A conceptual introduction to Git
- Using GitHub repositories
- Practical example: Setting up RStudio with GitHub
- Using git in the command line

Reproducible Science

Definition: Reproducible science refers to the ability to replicate the results of a scientific study using the same methods, data, and conditions. Its application is expected to:

- Ensures the credibility and reliability of scientific findings.
- Enhances transparency and trust in research.
- Facilitates the efficient accumulation and validation of knowledge.



Munafò, M. R., Nosek, B. A., Bishop, D. V., Button, K. S., Chambers, C. D., Percie du Sert, N., ... & Ioannidis, J. (2017). A manifesto for reproducible science. *Nature human behaviour*, 1(1), 1-9.

What we can do?

Transparency and Open Science:

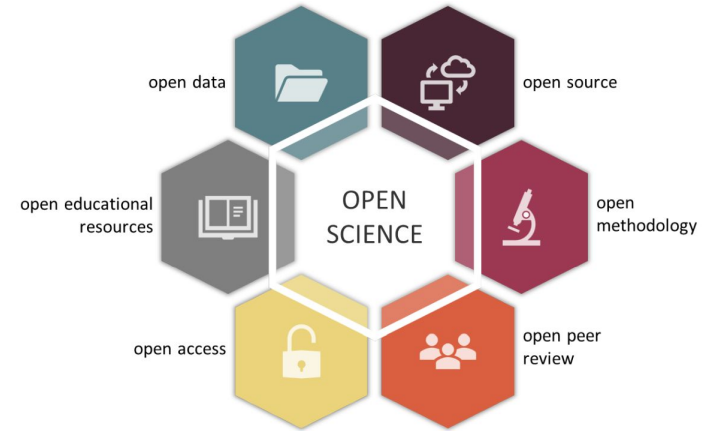
- Share data, materials, and code openly.
- Use open-access repositories and platforms like GitHub.

Blinding and Methodological Rigor:

- Implement blinding to reduce bias.
- Employ rigorous and standardized methods.

Collaboration and Team Science:

- Foster collaborative efforts and multi-site studies to enhance statistical power and generalizability.



Gallagher, R. V., Falster, D. S., Maitner, B. S., Salguero-Gómez, R., Vandvik, V., Pearse, W. D., ... & Enquist, B. J. (2020). Open Science principles for accelerating trait-based science across the Tree of Life. *Nature ecology & evolution*, 4(3), 294-303.

It is not enough to share the data and the code



Important_
Biblio



Interesting
_Results




Main_
results



Other_less_
important



Other_stuff



Paper_
final_
version



Interpretability and traceability become central

Good practices when sharing the code

Documentation:

- Provide clear and comprehensive documentation.
- Include a README file with instructions on how to use the code.

Licensing:

- Choose an appropriate open-source license (e.g., MIT, GPL).

Accessibility:

- Share your code on platforms like GitHub or GitLab.

Examples and Tests:

- Include example datasets and usage examples.
- Provide tests to ensure the code works as intended.

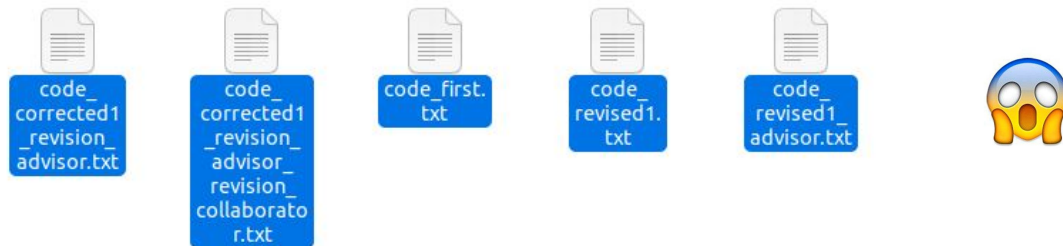
Version Control (very important for collaborations!):

- Use Git for version control to track changes and collaborate efficiently.
- Create a well-organized repository structure.



<https://docs.github.com/en>

An introduction to Git

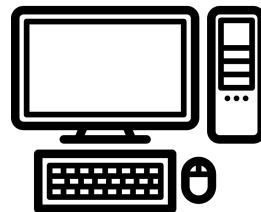


VCS: Version Control Systems are designed to organize collaborative software development

- They store all revisions of each file.
- They allow us to switch between versions and view differences.
- Each revision must be explicitly created with a message indicating the changes made.
- Distributed: Each user works independently on their own computer but can share their modifications with others.

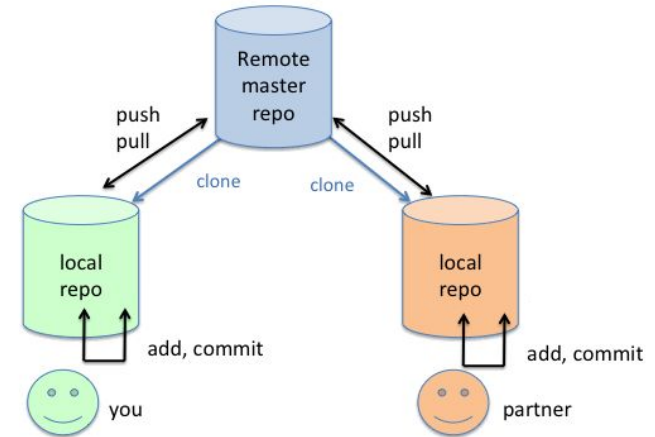
Git: Key concepts

- **Local repository:** The place where the files of a project are stored, along with all the additional information needed for version control.
- **Revision:** A snapshot of the repository at a given moment.
- **History:** A set of revisions ordered chronologically.
- **Remote repository:** A server to which Git sends changes when a push is executed.



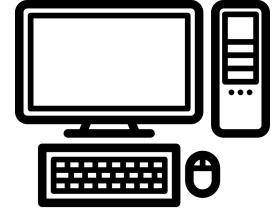
Git actions

- **add:** Adds a file to the revision.
- **commit:** Creates a new revision with a message describing changes.
- **clone:** Downloads a repository and its history.
- **push:** Shares local commits with a remote server or user.
- **pull:** Updates local repository with changes from others.
- **merge:** Combines local changes with others' changes.
- **status:** Shows the state of the local repository.

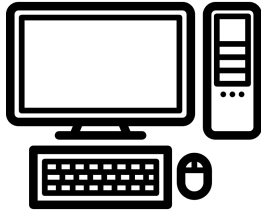


Git actions

Remote repository



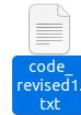
Local repository: User 2



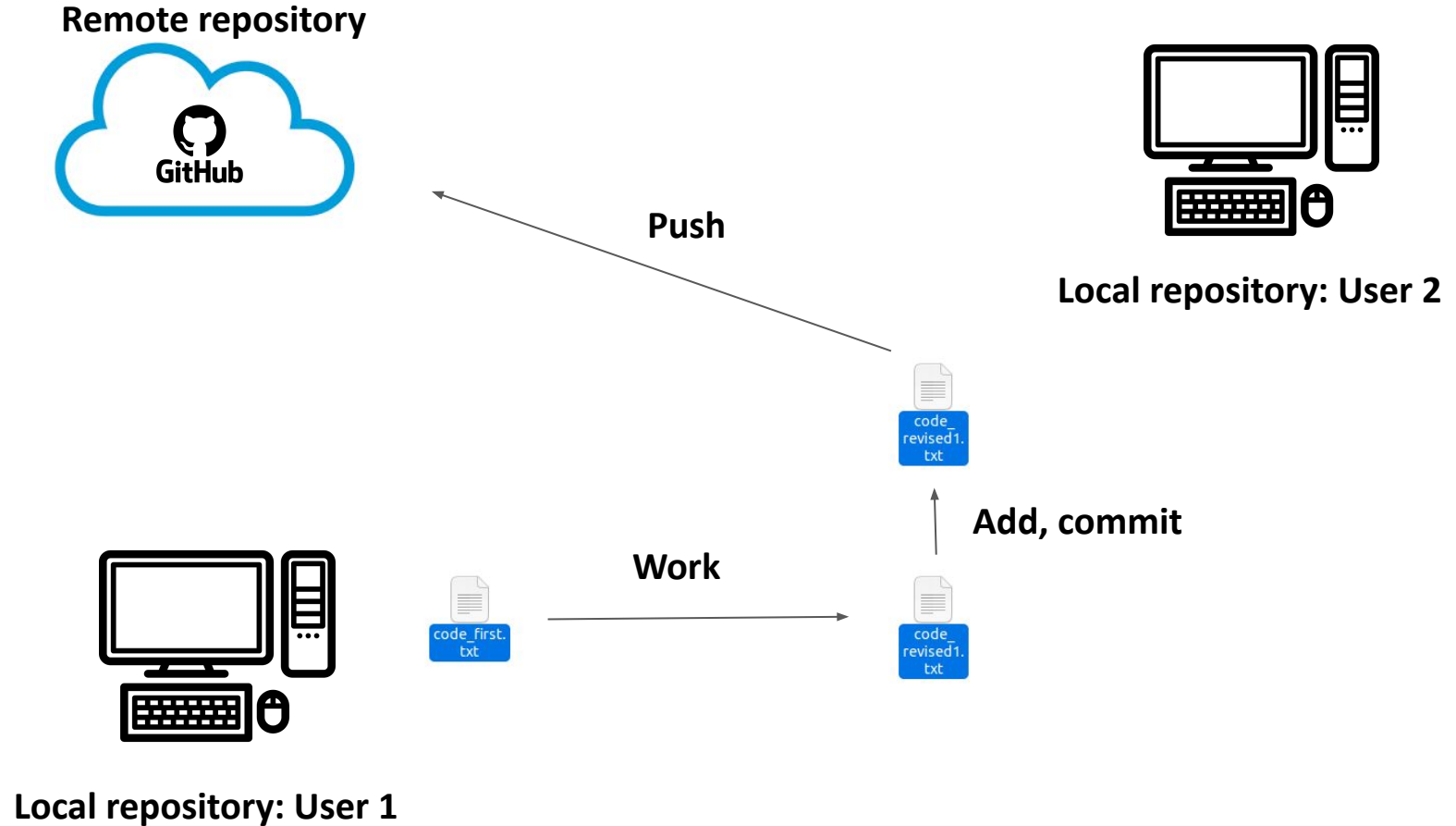
Local repository: User 1



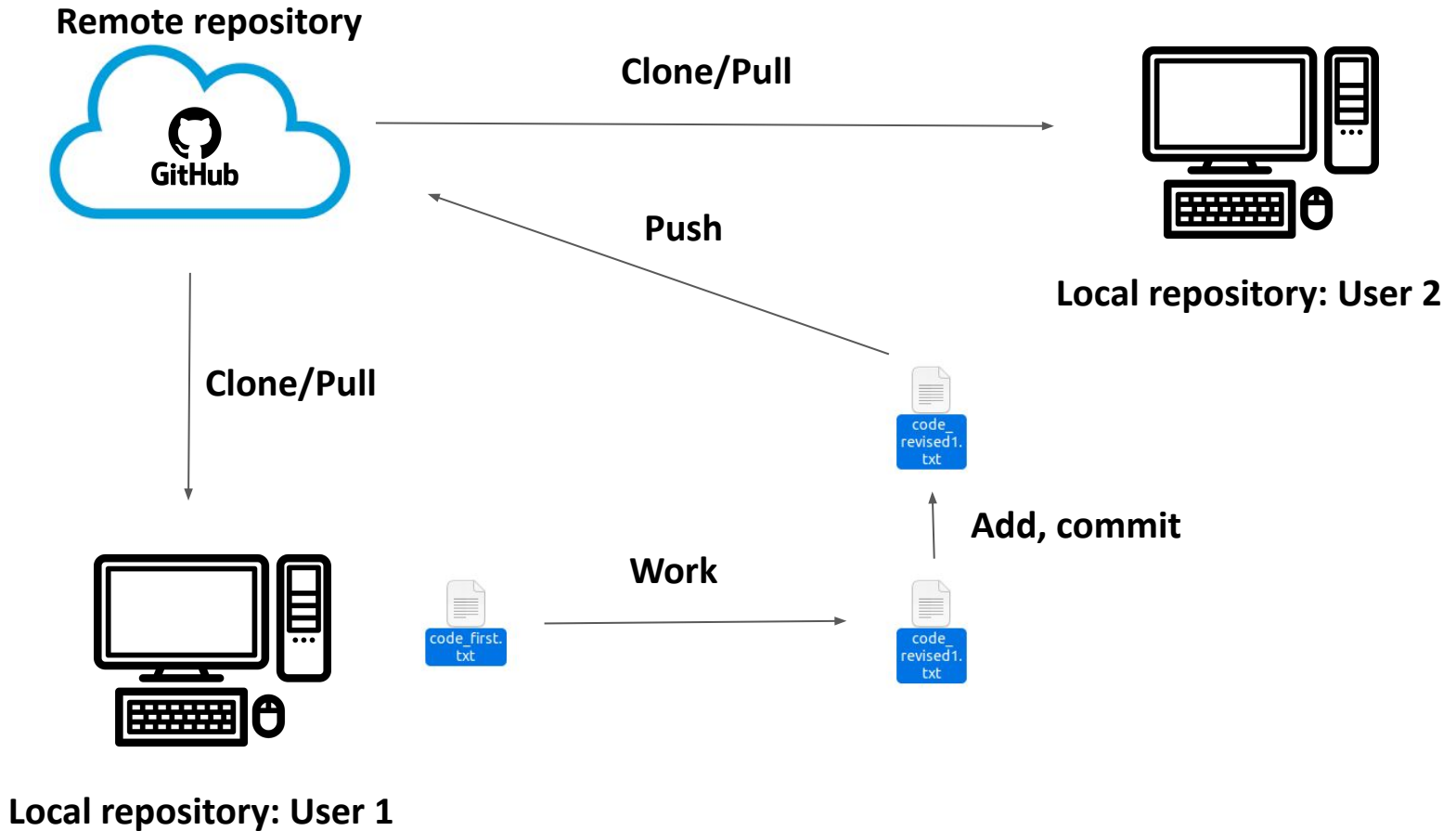
Work



Git actions



Git actions



- Several services offer Git repository hosting:
 - Provide a convenient web interface for:
 - Creating repositories
 - Managing user access
 - Viewing files
 - Comparing revisions
 - Viewing history
- Well-known services include:
 - GitHub
 - GitLab
 - Bitbucket

Repositories > New

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 *

MarsicoFL /

Great repository names are short and memorable. Need inspiration? How about **potential-octo-meme**?

Description (optional)

☒ **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](#).

☐ You are creating a public repository in your personal account.

Create repository



Using GitHub repositories

Code

Data

Code and data documentation

General documentation

The screenshot shows the GitHub repository for 'mispitools' by MarsicoFL. The repository is public and has 36 stars, 8 forks, and 12 tags. The file list on the left includes folders for 'R', 'data', and 'man', and files for 'README.md', 'UsingAll.R', 'mispitools.Rproj', and 'usingAll.R'. The right sidebar contains sections for 'About', 'Releases' (v1.2.0), 'Packages', 'Contributors', and 'Languages' (R 100.0%).

File/Folder	Commit Message	Time Ago
R	Cleaning and preparing new version	4 months ago
README_files/figure-markdown_github	Add files via upload	10 months ago
data	Delete data/YRI_summarized_ibd_datatxt.txt	4 months ago
man	Cleaning and preparing new version	4 months ago
.Rbuildignore	Update .Rbuildignore	10 months ago
.gitignore	nuevo ingreso	4 years ago
DESCRIPTION	Cleaning and preparing new version	4 months ago
LICENSE.md	Add files via upload	4 years ago
NAMESPACE	Cleaning and preparing new version	4 months ago
README.md	Update README.md	3 months ago
UsingAll.R	Create UsingAll.R	last month
mispitools.Rproj	Add files via upload	4 years ago
usingAll.R	Add files via upload	3 months ago

About

'mispitools' is an open-source package written in the R statistical language. It consists of a collection of decision-making tools designed for conducting missing person searches.

Releases (12)

v1.2.0 Latest
on Aug 16, 2024

+ 11 releases

Packages

No packages published
[Publish your first package](#)

Contributors (9)

Languages

R 100.0%

Path

Copies



Practical example: Setting up RStudio with GitHub

1 - Setting Up the key



Guide for Setting Up Git and GitHub with RStudio

EMBO POPGEN 2024

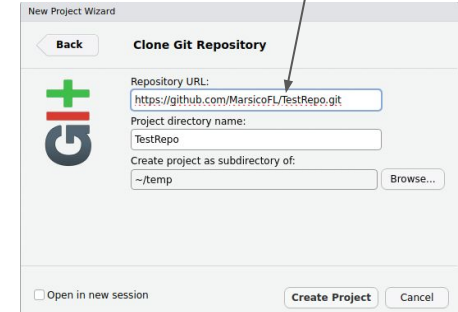
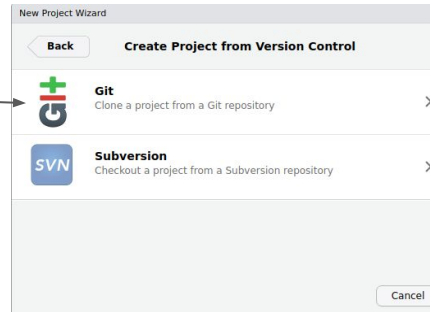
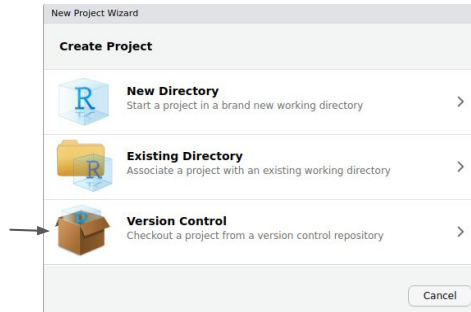
This guide provides a step-by-step process to set up R, RStudio, Git, and GitHub with RStudio. It includes instructions for installing R, RStudio, and Git, configuring Git with RStudio, creating a personal access token for GitHub, and verifying the setup.

https://github.com/ColonnaLab/EMBO_popgen/blob/main/popgen2024/Franco_Marsico/Tutorial_RStudio_Github.pdf

2 - Open a new git project

RStudio: File > New Project

Your github repo



Practical example: Setting up RStudio with GitHub

3 - Open and modify files

Panel showing git actions

The screenshot shows the RStudio interface with the following components:

- Source Editor:** Displays the content of `README.md`.

```
1 # TestRepo
2 This repo is just for testing purposes.
3 This is a line from RStudio.
4 Test2.
5 Test3.
6 Test4. From Windows.
7
8 Test5. For ppt
9
```
- Console:** Shows the R prompt and output of the `license()` command.

```
R 4.1.2 ~ /temp/TestRepo/
Usted puede redistribuirlo bajo ciertas circunstancias.
Escriba 'license()' o 'licence()' para detalles de distribucion.

R es un proyecto colaborativo con muchos contribuyentes.
Escriba 'contributors()' para obtener más información y
'citation()' para saber cómo citar R o paquetes de R en publicaciones.

Escriba 'demo()' para demostraciones, 'help()' para el sistema on-line de ayuda,
o 'help.start()' para abrir el sistema de ayuda HTML con su navegador.
Escriba 'q()' para salir de R.

> |
```
- Git Panel:** Shows the status of the repository. The `README.md` file is marked as modified (M).

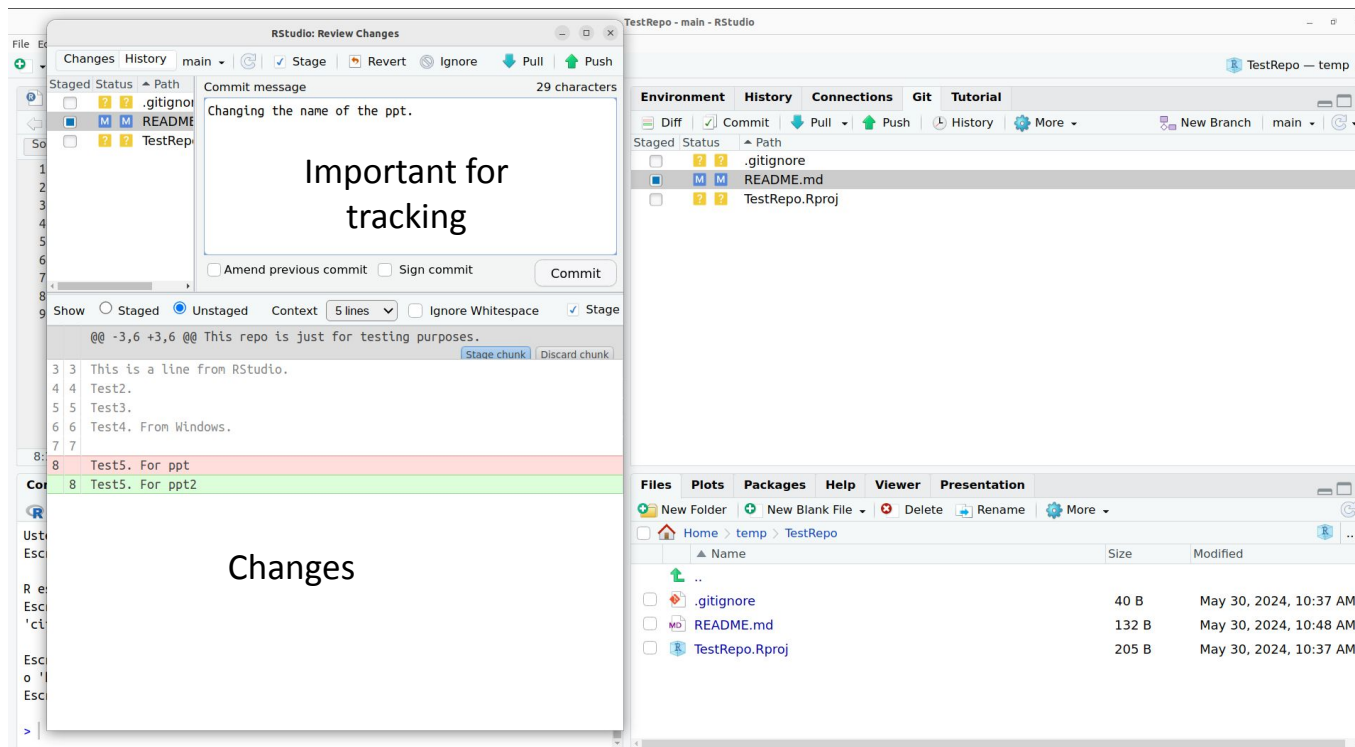
Staged	Status	Path
<input type="checkbox"/>	?	.gitignore
<input type="checkbox"/>	M	README.md
<input type="checkbox"/>	?	TestRepo.Rproj
- Files Panel:** Shows the file structure of the repository.

Name	Size	Modified
..		
.gitignore	40 B	May 30, 2024, 10:37 AM
README.md	131 B	May 30, 2024, 10:45 AM
TestRepo.Rproj	205 B	May 30, 2024, 10:37 AM

An arrow points from the text "M means that it is modified respect to the original version" to the 'M' status icon in the Git panel.

Practical example: Setting up RStudio with GitHub

4 - Commit selecting modified files



After commit
this message
appears



Practical example: Setting up RStudio with GitHub

5 - Push

Git panel > Push



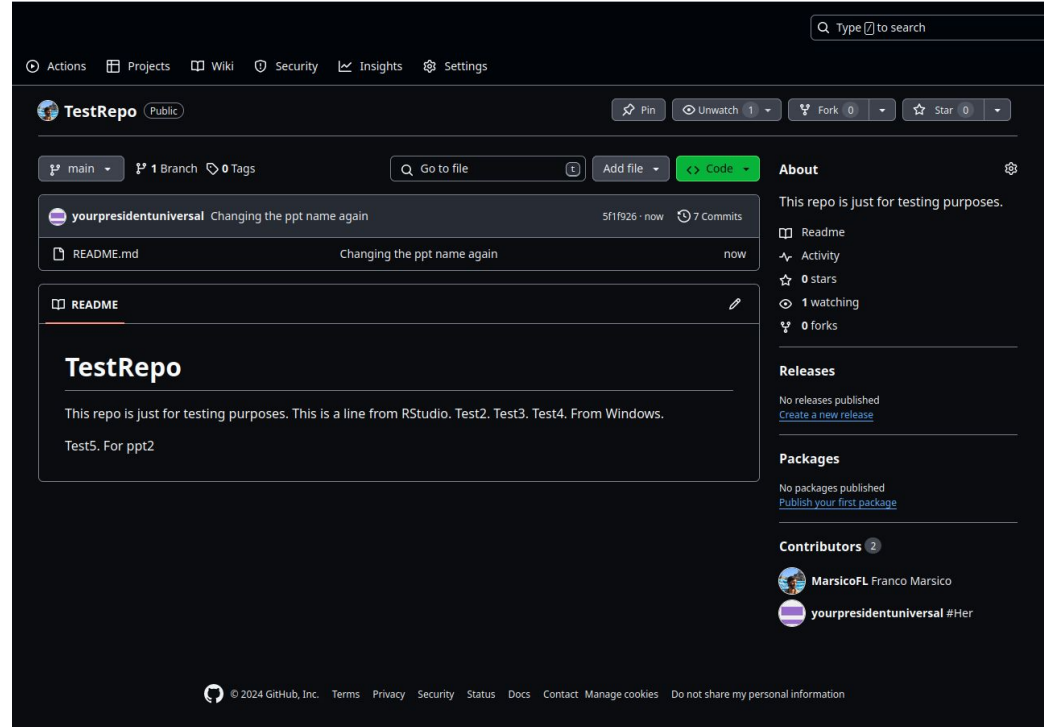
```
Git Push
Close

>>> /usr/bin/git push origin HEAD:refs/heads/main
To https://github.com/MarsicoFL/TestRepo.git
   80870da..3308c27  HEAD -> main
```

6 - Pull

If the repository is a collaborative exercise is a good practice pull (actualize with changes from other users) after push.

Changes are incorporated in the remote repository



Practical example: Setting up RStudio with GitHub

7 - History

Commit message

User

Changes

The screenshot shows the 'RStudio: Review Changes' window. At the top, there's a tab bar with 'Changes' and 'History'. The 'History' tab is active, showing a list of commits. The commit 'Checking Windows' is selected and highlighted in blue. To the left of the commit list is a vertical timeline with circular markers. Below the commit list, the details for the selected commit are shown, including the SHA, Author, Date (UTC), Subject, and Parent. At the bottom, the 'README.md' file is shown with its changes highlighted in different colors (red for deletions, green for additions).

Commit	Author	Date	SHA
Changing the name of the ppt.	YourName <your@mail>	2024-05-30	5f1f9265
Selection.	YourName <your@mail>	2024-05-30	3308c276
Checking Windows	marsicoFL <franco.lma>	2024-05-28	9ee2e666
New	marsicoFL <franco.lma>	2024-05-27	2a0cc10d
First comm	marsicoFL <franco.lma>	2024-05-27	dcdcbf37
Initial commit	Franco Marsico <francc>	2024-05-26	fa18352e

Commit Details:

- SHA:** 9ee2e66dd0f53de57fd77bc05ac1932050bbb91
- Author:** marsicoFL <franco.lmarsico@mail.com>
- Date (UTC):** 2024-05-28 01:05
- Subject:** Checking Windows
- Parent:** 2a0cc10daed0dc0b6ea8ff4cd2fde81318105185

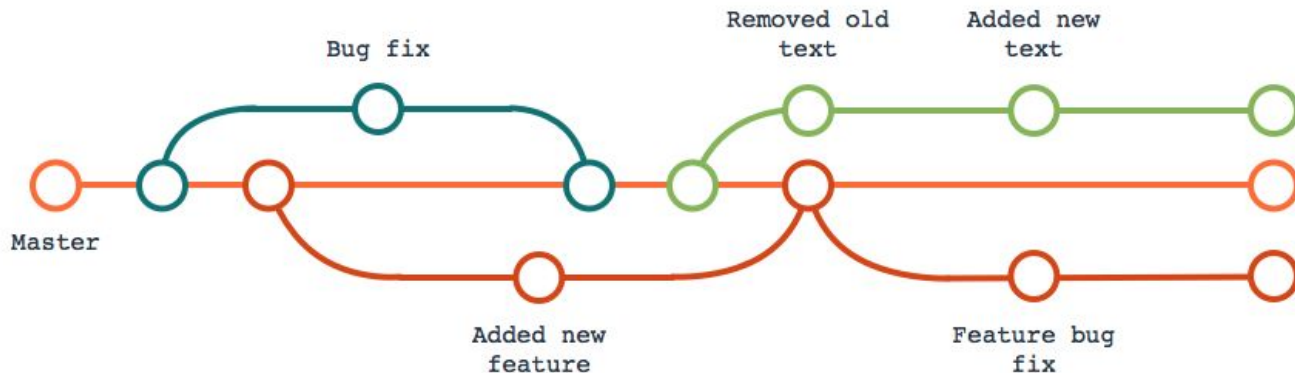
File Changes:

README.md

```
@@ -2,4 +2,5 @@
2 This repo is just for testing purposes.
3 This is a line from RStudio.
4 Test2.
5 Test3.
   No newline at end of file
5 Test3.
6 Test4. From Windows.
   No newline at end of file
```

8 - Branching

- **branch**: A fork of the repository's history.
- **merge**: The action of combining the history of two different branches.
- **master**: The main branch that always exists.
- **HEAD**: The most current revision of the branch you are on.
- **checkout**: The action to switch from one branch to another (also works with commits!).



Practical example: Setting up RStudio with GitHub

Git panel > New Branch

New Branch

Branch Name:

Remote: origin Add Remote...

☒ Sync branch with remote

Create Cancel

(LOCAL BRANCHES)

main
readme_bug
readme_bug2

(REMOTE: ORIGIN)

main
readme_bug
readme_bug2

TestRepo Public Pin Unwatch 1

readme_bug2 had recent pushes 27 seconds ago Compare & pull request

main 3 Branches 0 Tags Go to file Add file Code

yourpresidentuniversal Changing the ppt name again 5f1f926 · 12 minutes ago 7 Commits

README.md Changing the ppt name again 12 minutes ago

README

TestRepo

This repo is just for testing purposes. This is a line from RStudio. Test2. Test3. Test4. From Windows.

Test5. For ppt2

Practical example: Setting up RStudio with GitHub

Check for conflicts

Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also [compare across forks](#). Learn more about diff comparisons here.

base: main ← compare: readme_bug2 ✓ Able to merge. These branches can be automatically merged.

Add a title

branch2

Add a description

Write Preview H B I

Add your description here...

Markdown is supported | Paste, drop, or click to add files

Reviewers
No reviews

Assignees
No one—assign yourself

Labels
None yet

Projects
None yet

Milestone
No milestone

Development
Use [Closing keywords](#) in the description to automatically close issues

Helpful resources
[GitHub Community Guidelines](#)

Create pull request

1 commit 1 file changed 1 contributor

branch2 #1

MarsicoFL merged 1 commit into main from readme_bug2 now

Conversation 0 Commits 1 Checks 0 Files changed 1 +2 -0

MarsicoFL commented now

No description provided.

branch2 5495cb8

MarsicoFL merged commit cd827aa into main now

Revert

Pull request successfully merged and closed

You're all set—the readme_bug2 branch can be safely deleted.

Delete branch

Add a comment

Write Preview H B I

Add your comment here...

Markdown is supported | Paste, drop, or click to add files

Reviewers
No reviews

Assignees
No one—assign yourself

Labels
None yet

Projects
None yet

Milestone
No milestone

Development
Successfully merging this pull request may close these issues.

Notifications
Customize
Unsubscribe
You're receiving notifications because you're watching this repository.

Why use the git command line?

- The RStudio GUI is very useful for GitHub, but it may fail with other clients.
- It allows working with the terminal and not in interactive mode (useful when working in HPC).
- It provides access to a full range of commands and not only those supported by the GUI.
- Avoiding the GUI is more efficient.

<https://git-scm.com/book/en/v2/Getting-Started-About-Version-Control>

- We have introduced key aspects of reproducible science.
- Assessed the role of sharing code in reproducibility.
- Introduced central concepts of Git and GitHub.
- Set up RStudio GUI for connecting with GitHub.
- Discussed the benefits of Git command line functions.