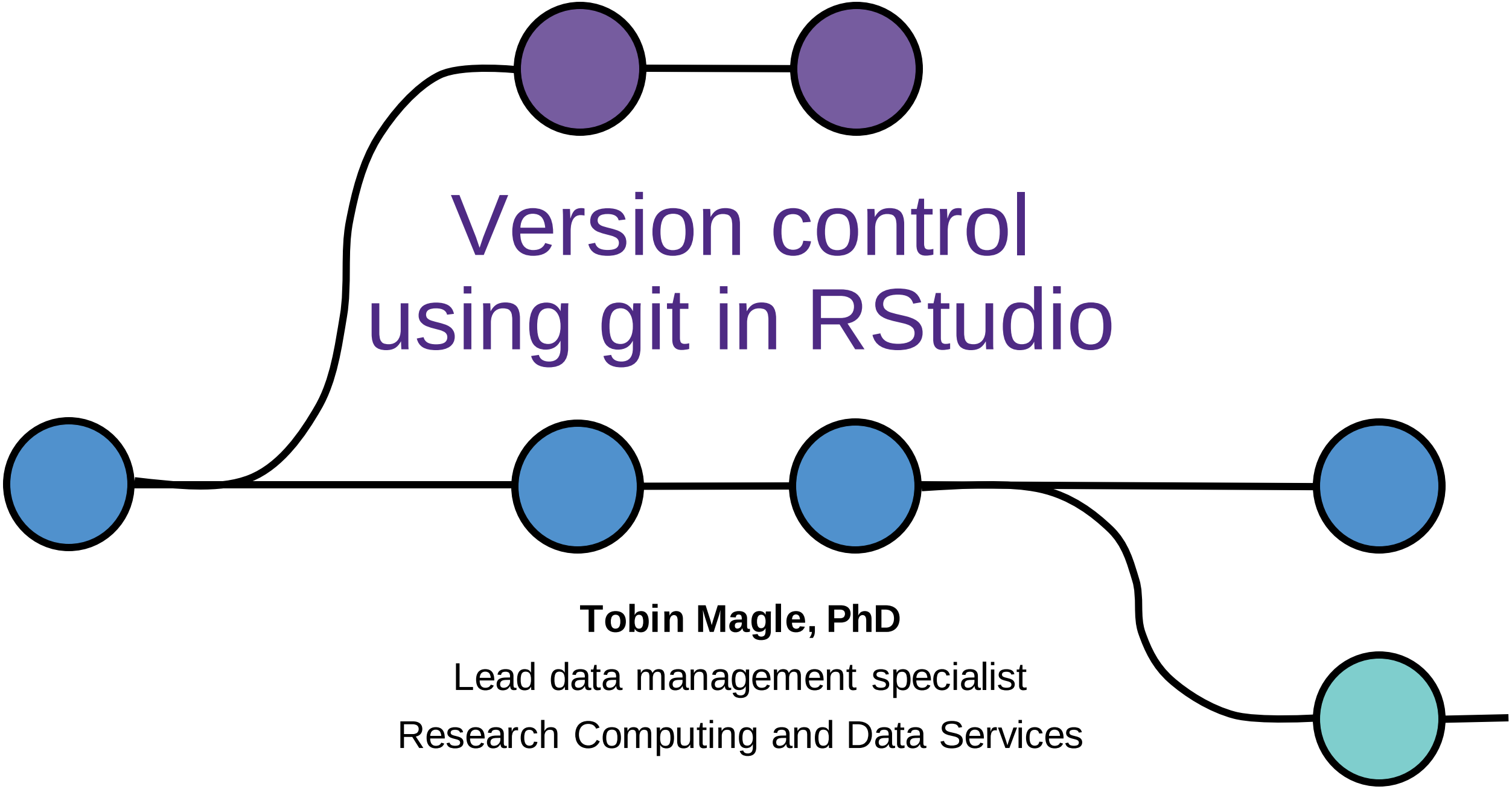


Version control using git in RStudio

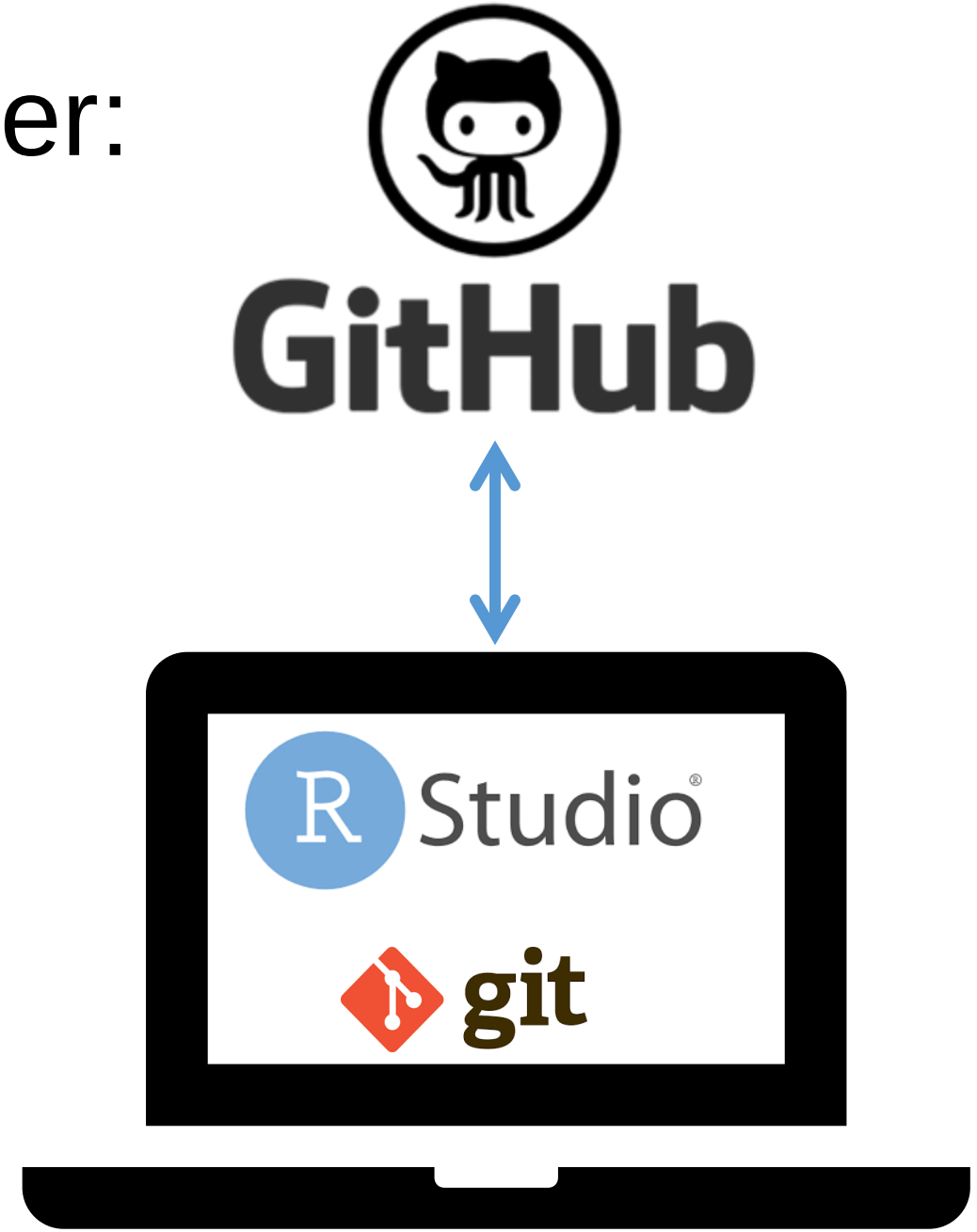
Tobin Magle, PhD

Lead data management specialist

Research Computing and Data Services

Today we're going to cover:

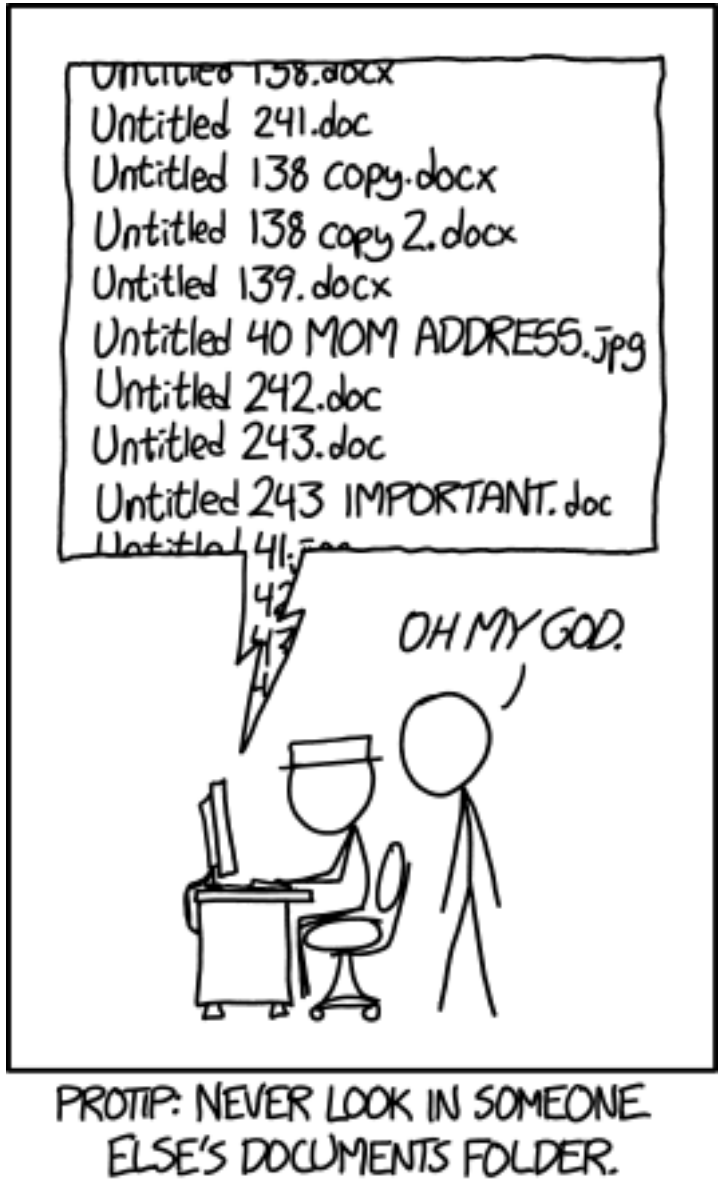
- What is **version control**?
- What is **git**?
- Version control using **RStudio**
- Backing up your data to **GitHub**



Manual Version Control

“Renaming files”

- Creates many files
- Relies on your ability to name files consistently
- Not designed for collaboration



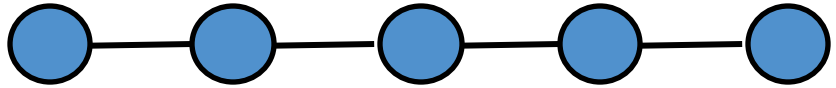
https://xkcd.com/1459/

Version Control Systems

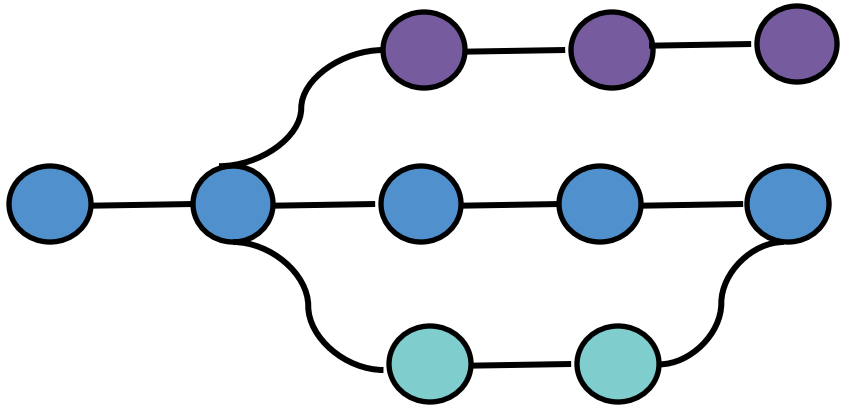
“Unlimited undo”

- One file, many versions
- Records who made what change when
- Good for **collaboration**: identifies and helps resolve conflicting changes

Solo

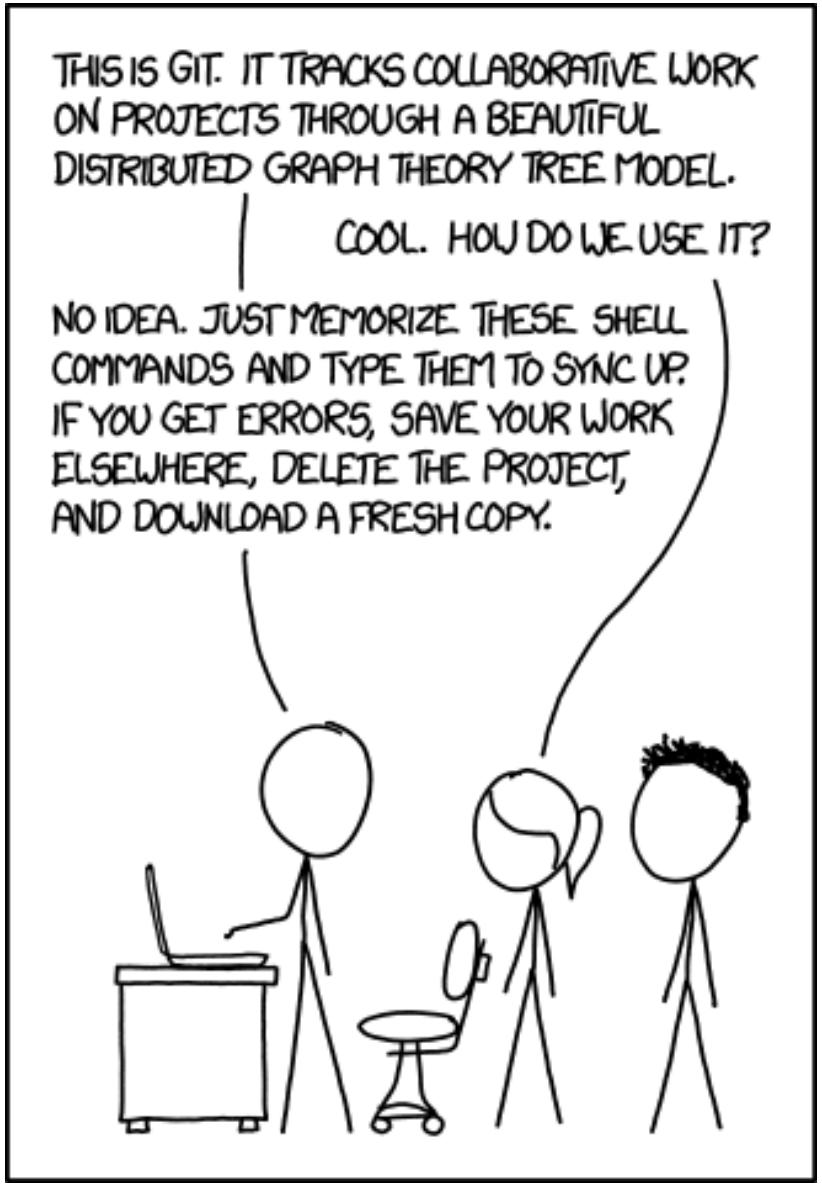


Collaborative



Today we're using **git**

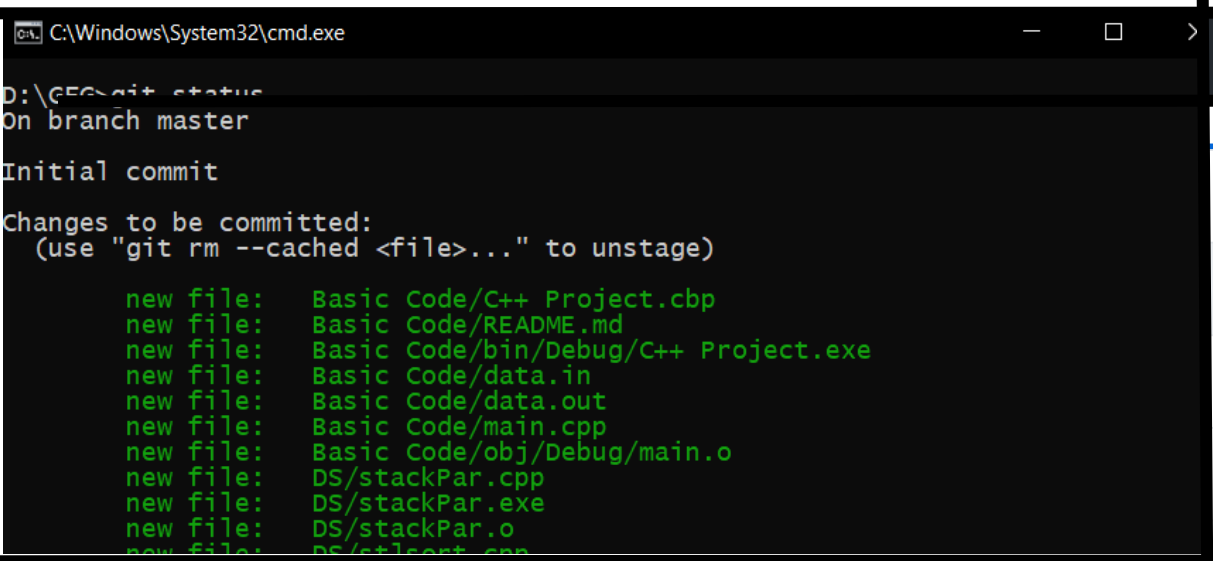
- Built for large software development projects
- Can get complicated, but the basics are simple
- Free, open source
- Many online resources/tutorials



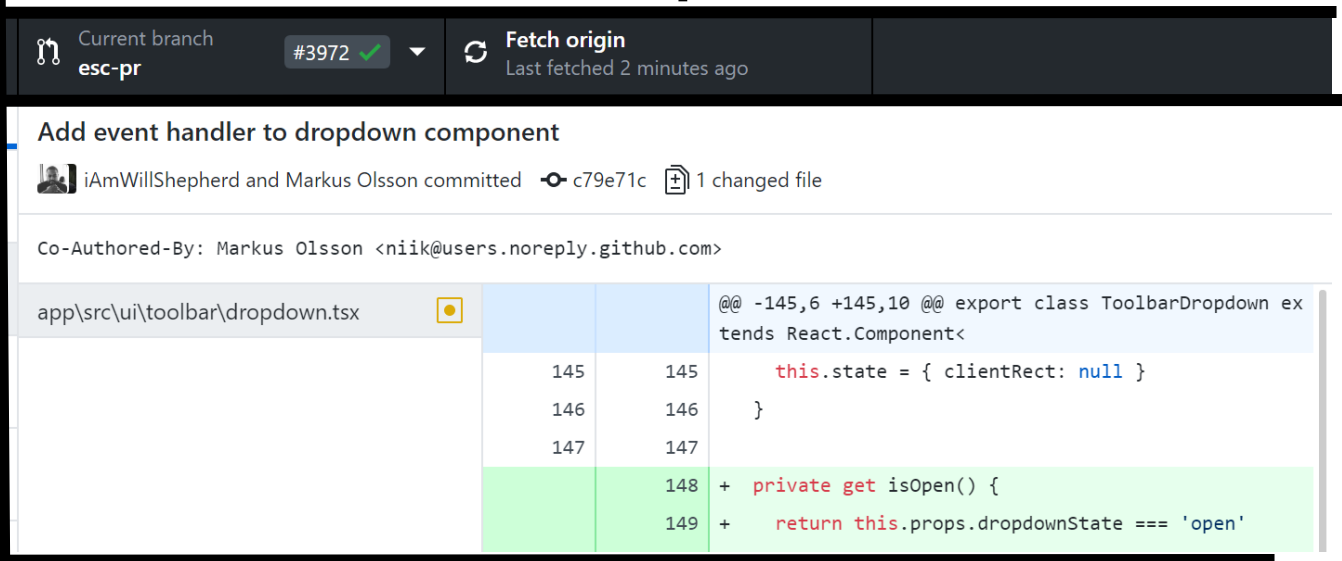
https://xkcd.com/1597/

Many ways to use Git

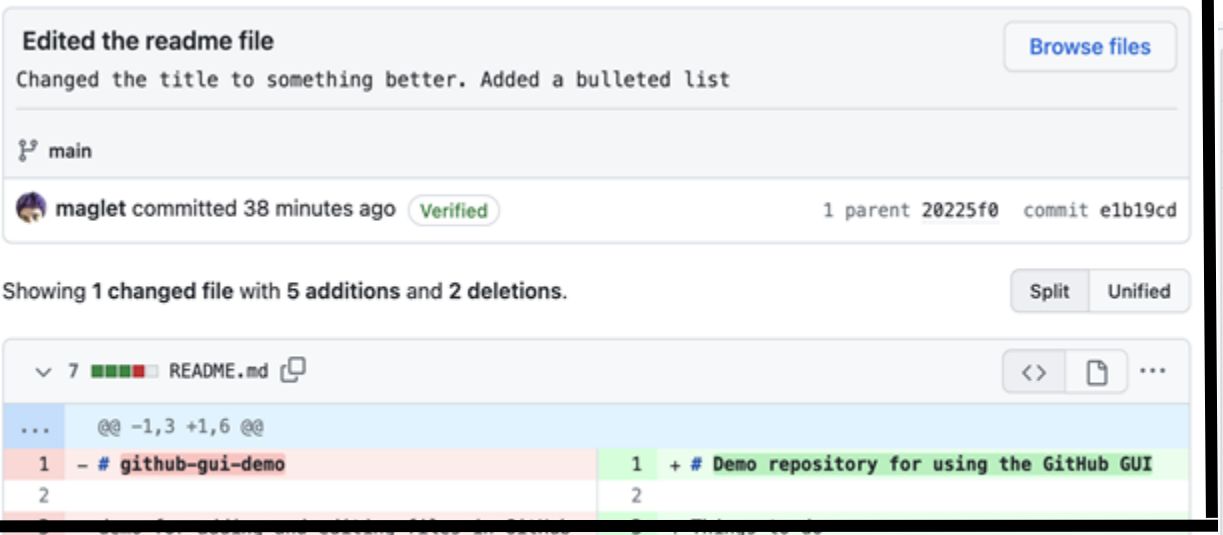
Command line



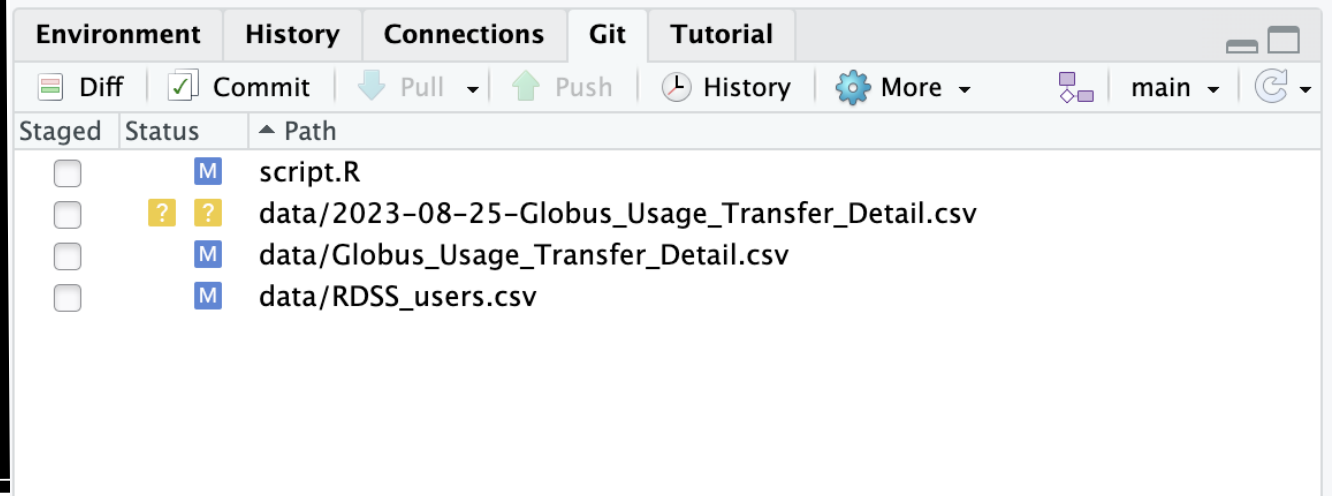
Desktop Clients



GitHub Website

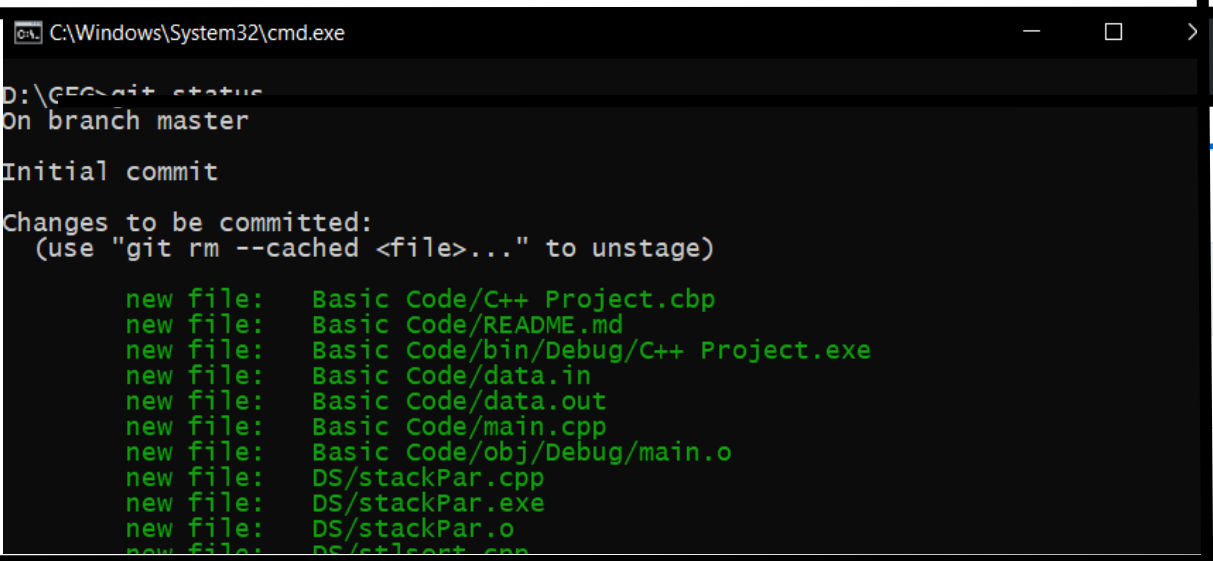


RStudio IDE

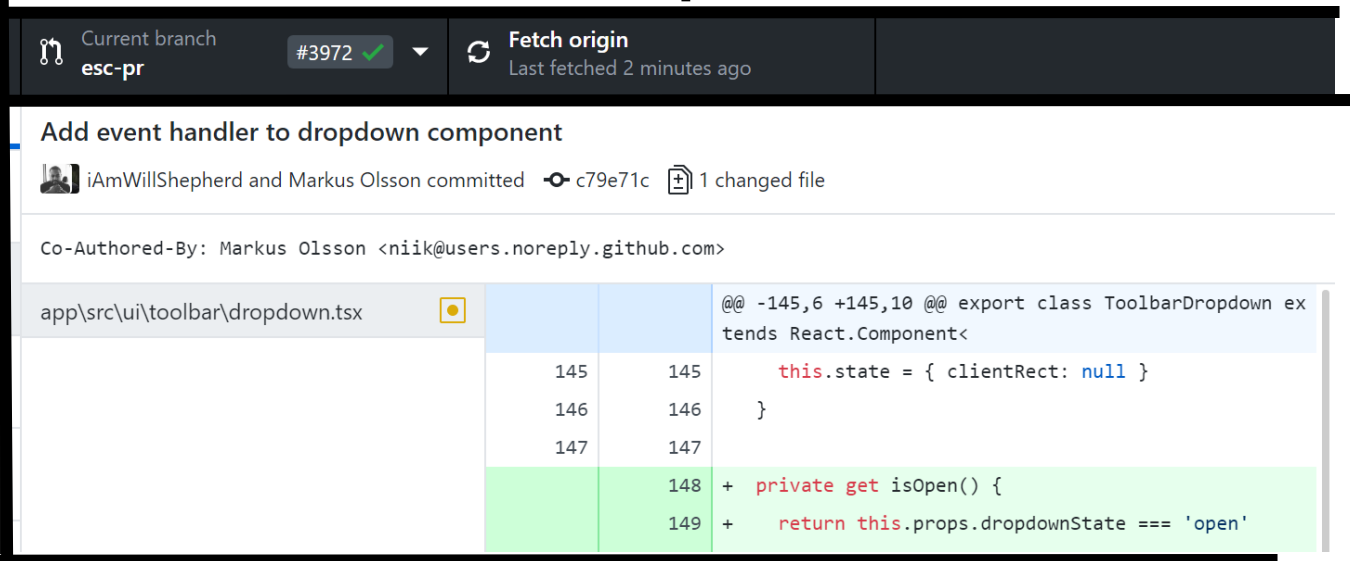


Many ways to use Git

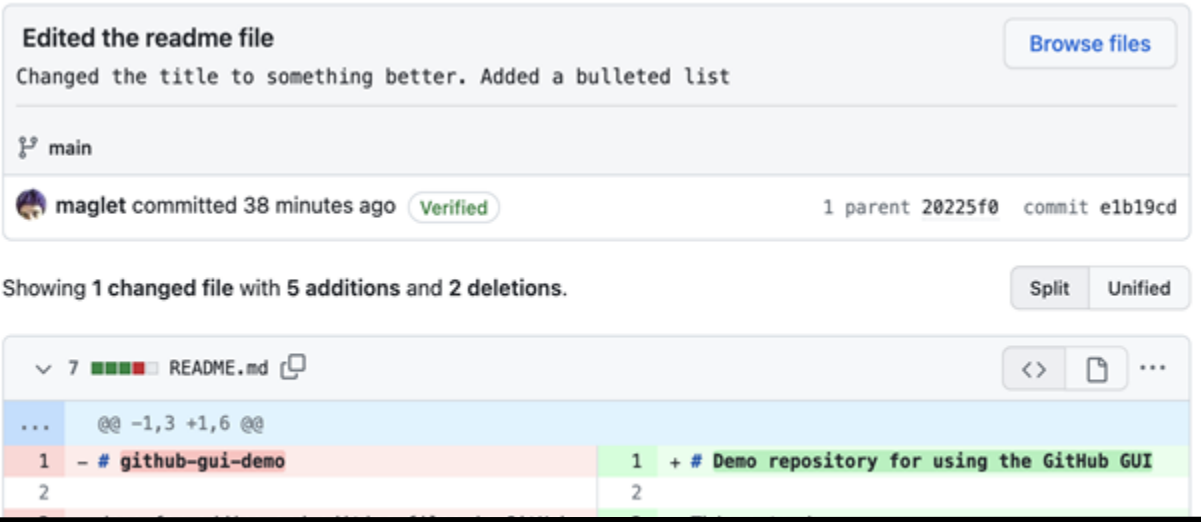
Command line



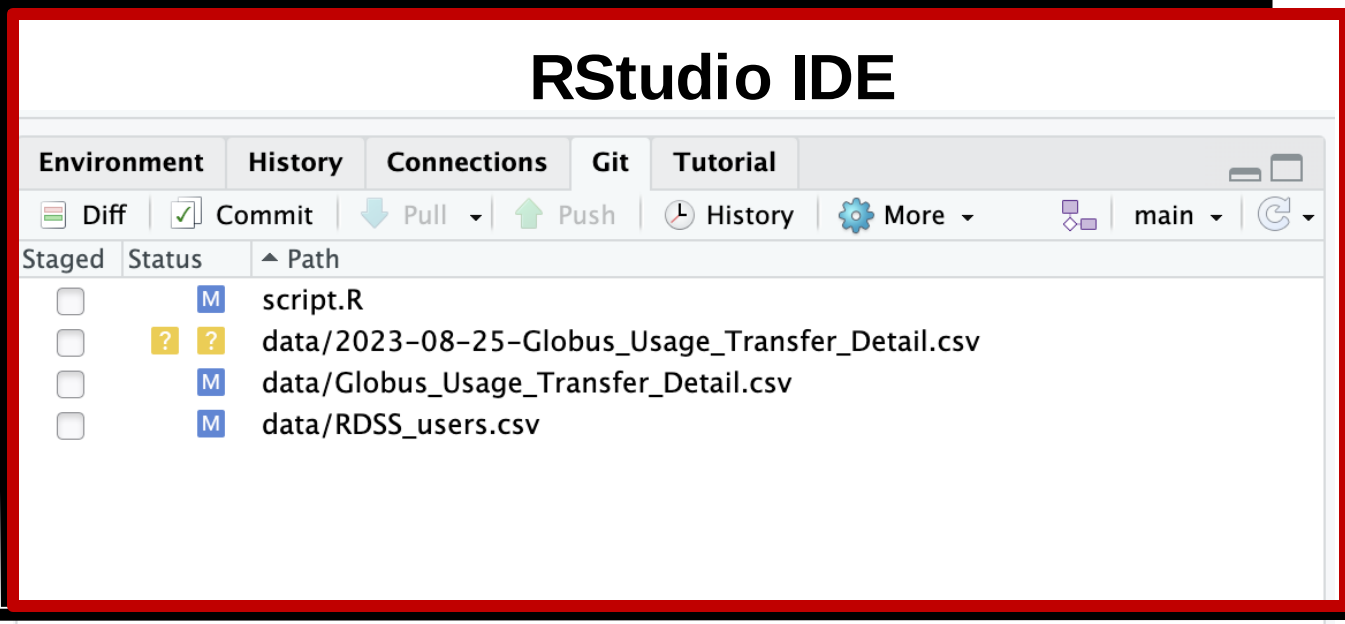
Desktop Clients



GitHub Website

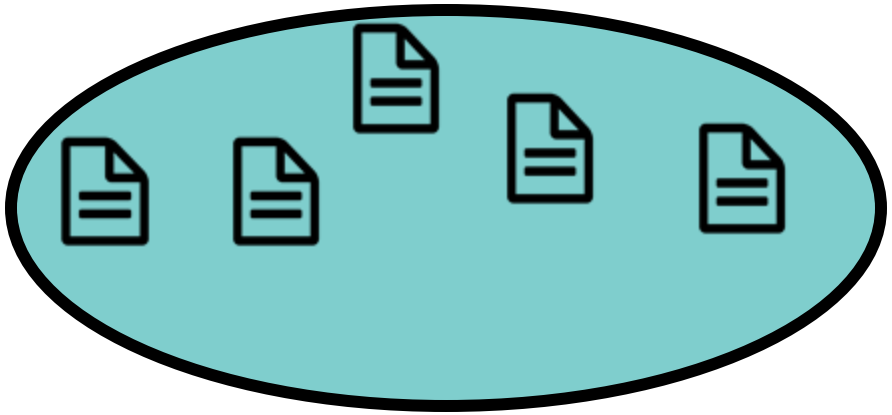


RStudio IDE



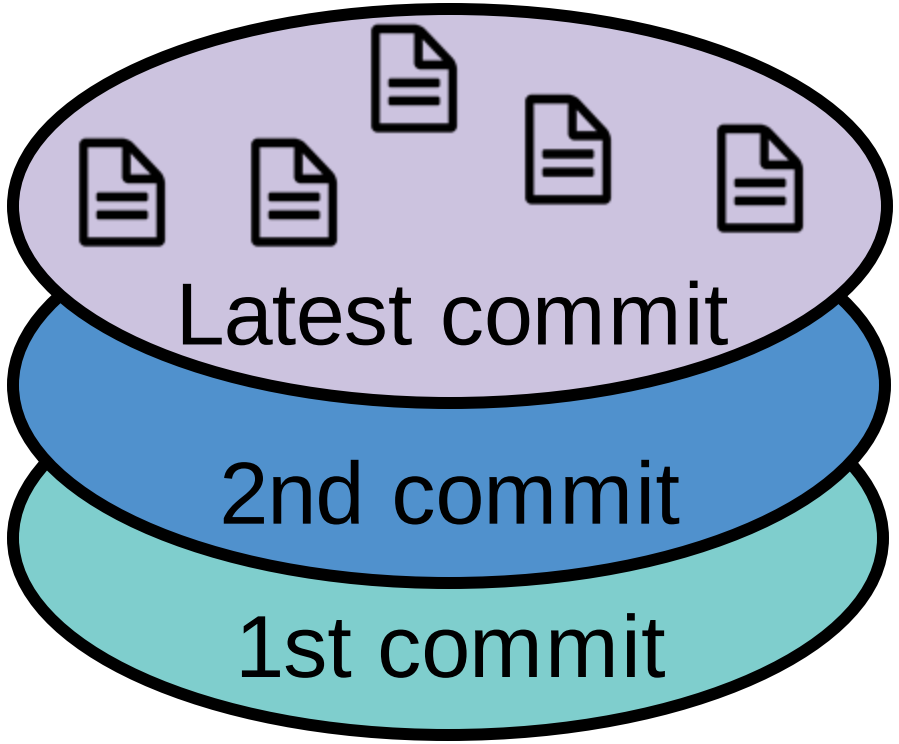
Using git creates a repository

- **Repository** – A folder that is tracked by git



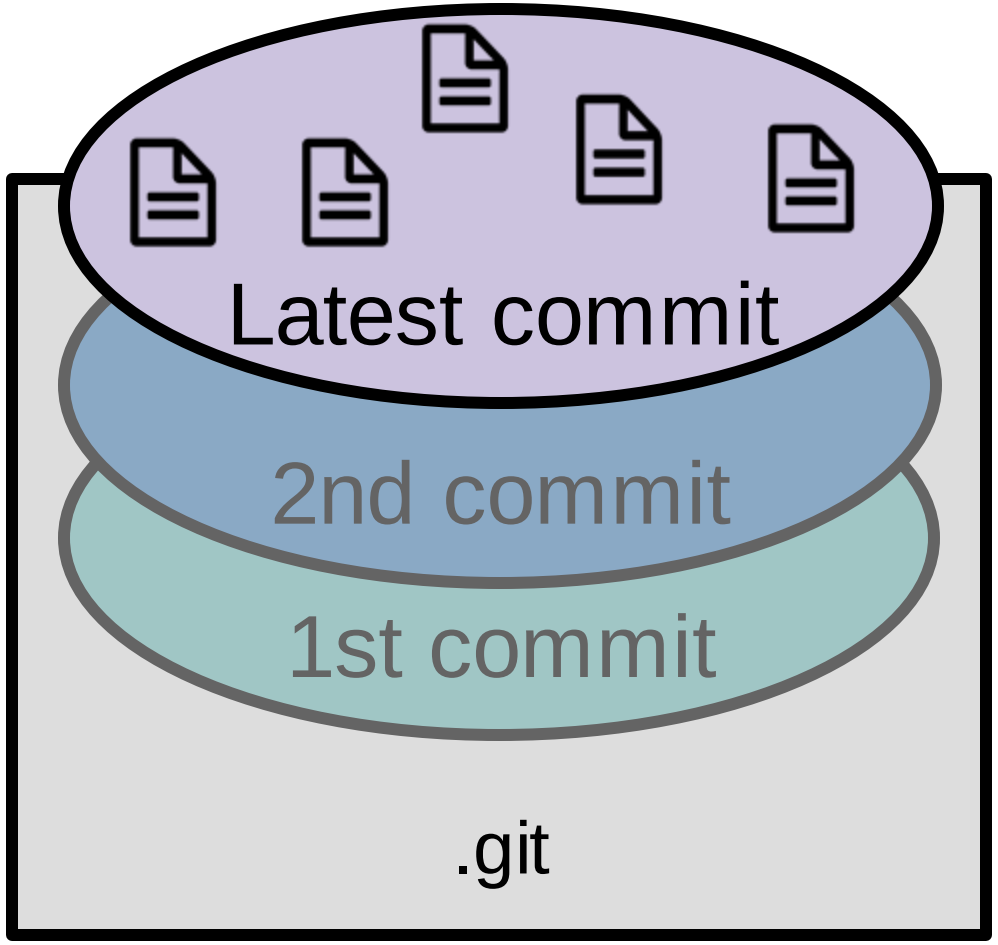
What is a repository?

- **Repository** – A folder that is tracked by git
- **Commit** – a version of what the folder looks like when it was created



Repository structure

- **Repository** – A folder that is tracked by git
- **Commit** – a version of what the folder looks like when it was created
- **Structure is hidden** (.git), you see the most recent version



RStudio workflow

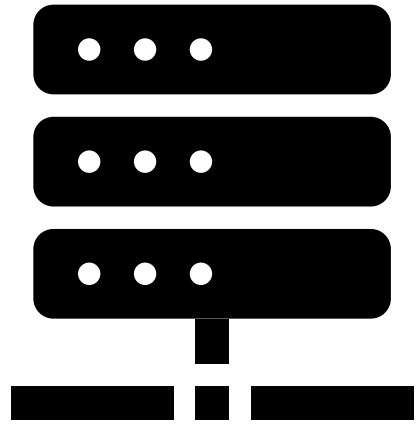
Coding happens on your computer

- Create and edit files in RStudio
- Use git version control in the RStudio interface
- Creates a local repository

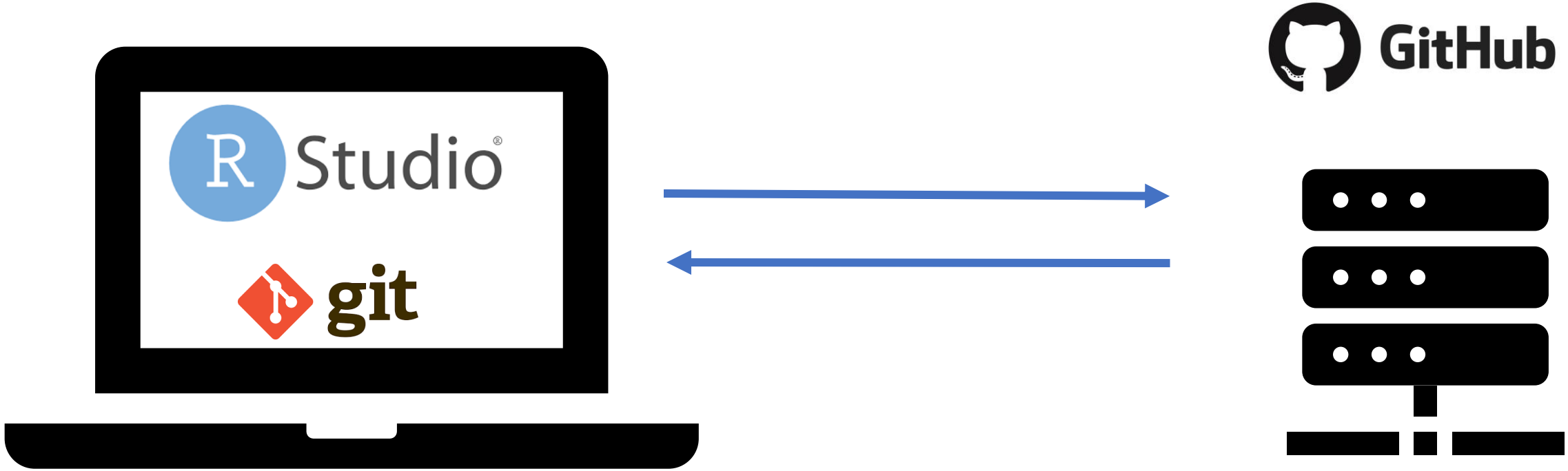
Git Workflow

You can save your work on GitHub

- Acts as a backup
- Share with colleagues
- Collaboration: Users with access can copy, modify and suggest changes to code



In practice

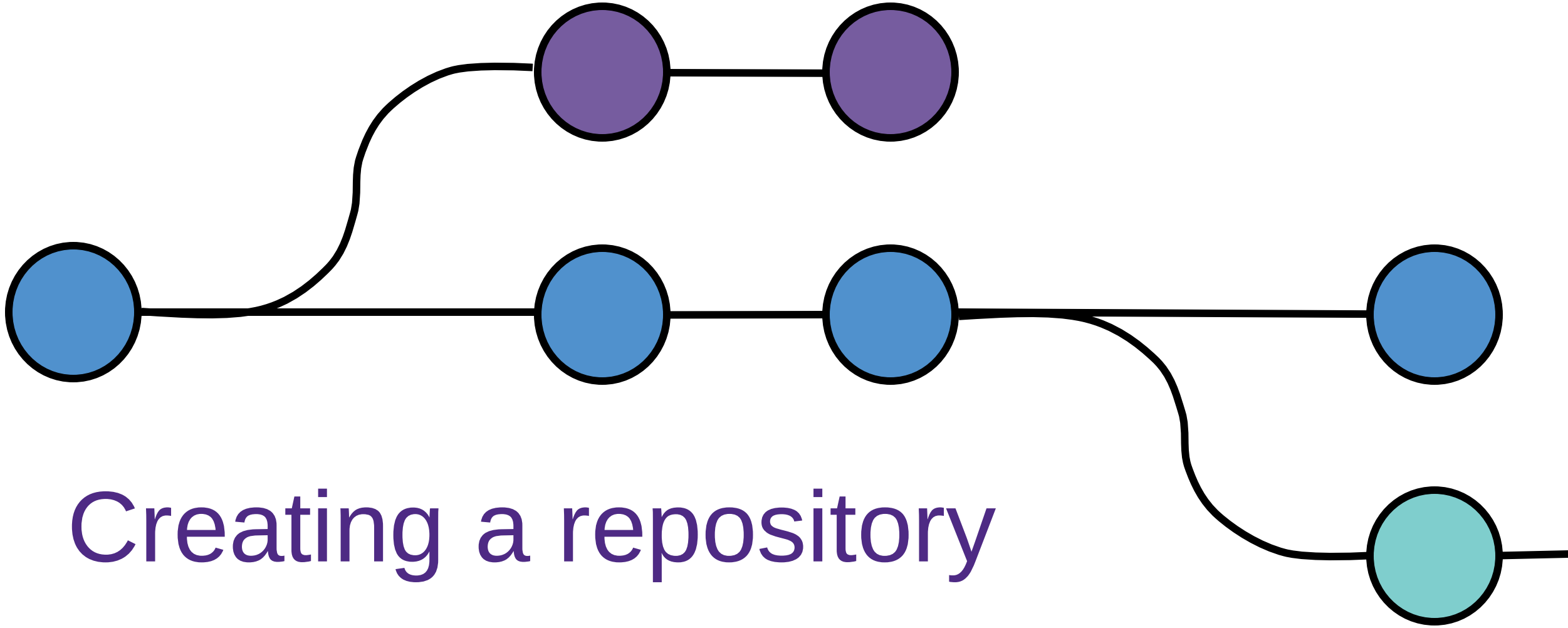


Local repository
(on your computer)

Remote repository
(on the web)

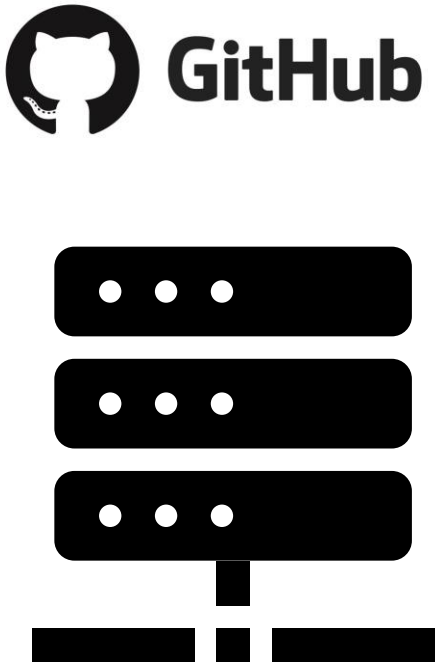
Pre-workshop Setup

- Install R installed
- Install RStudio
- Install git



Creating a repository

Workflow



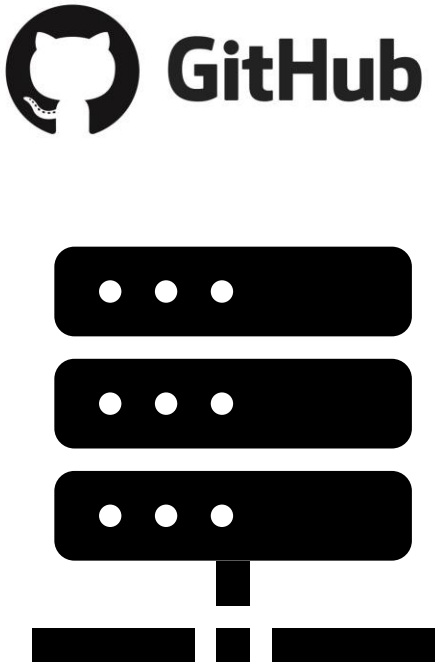
Remote repository
(on the web)

Workflow

Local

Clone

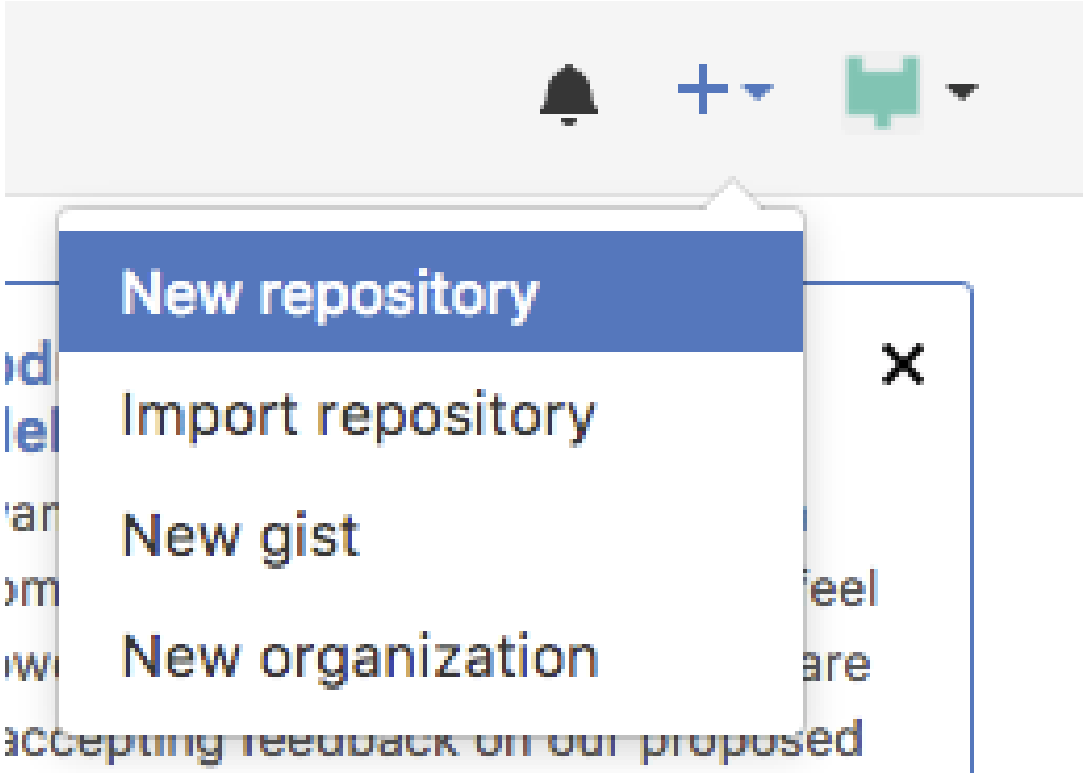
- Establish a connection
- Download remote to local



Remote

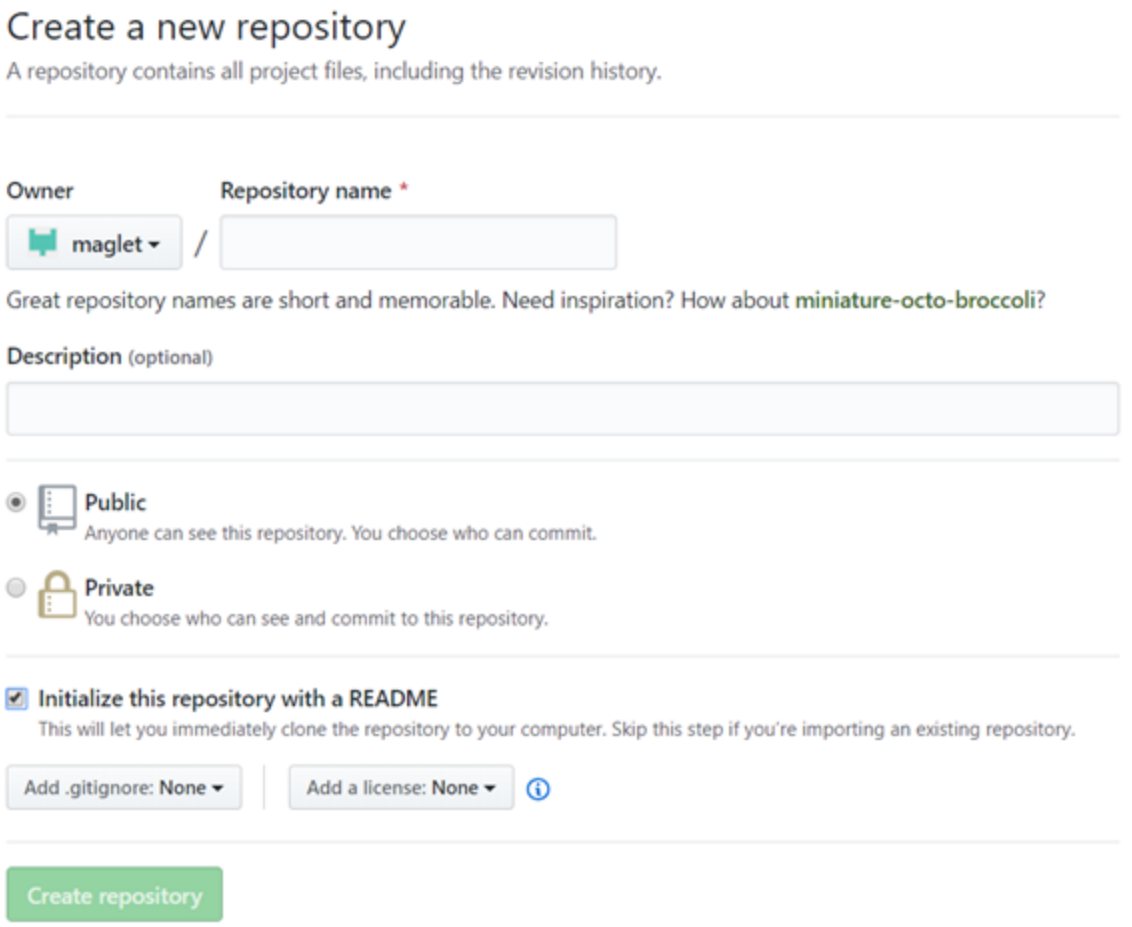
Make an empty GitHub repo

- Go to http://github.com
- Click the plus sign on the upper right hand side of the screen
- Select New repository



Make an empty repository

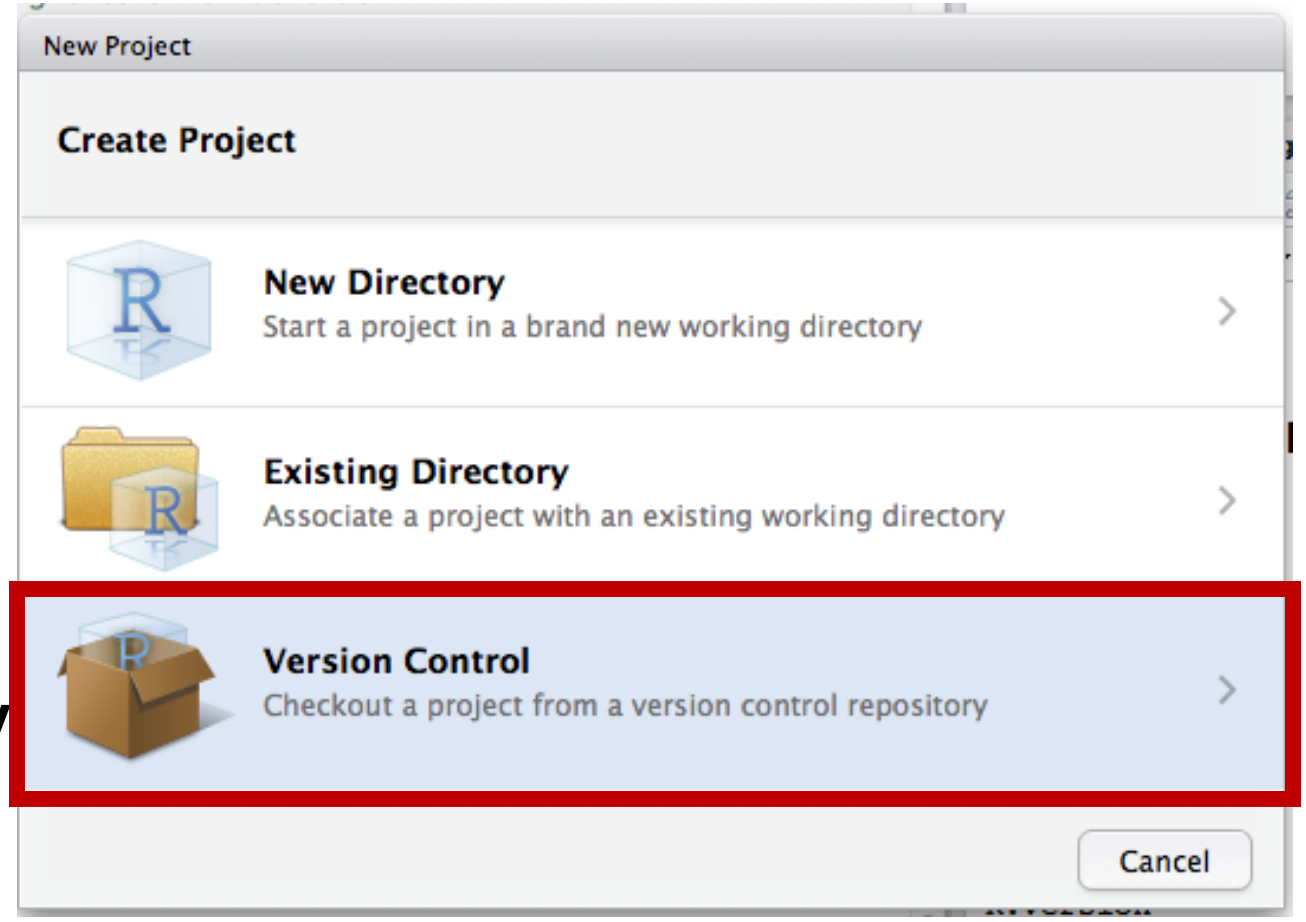
- Name your repository
- Provide a description
- Keep it public
- Initialize with a readme



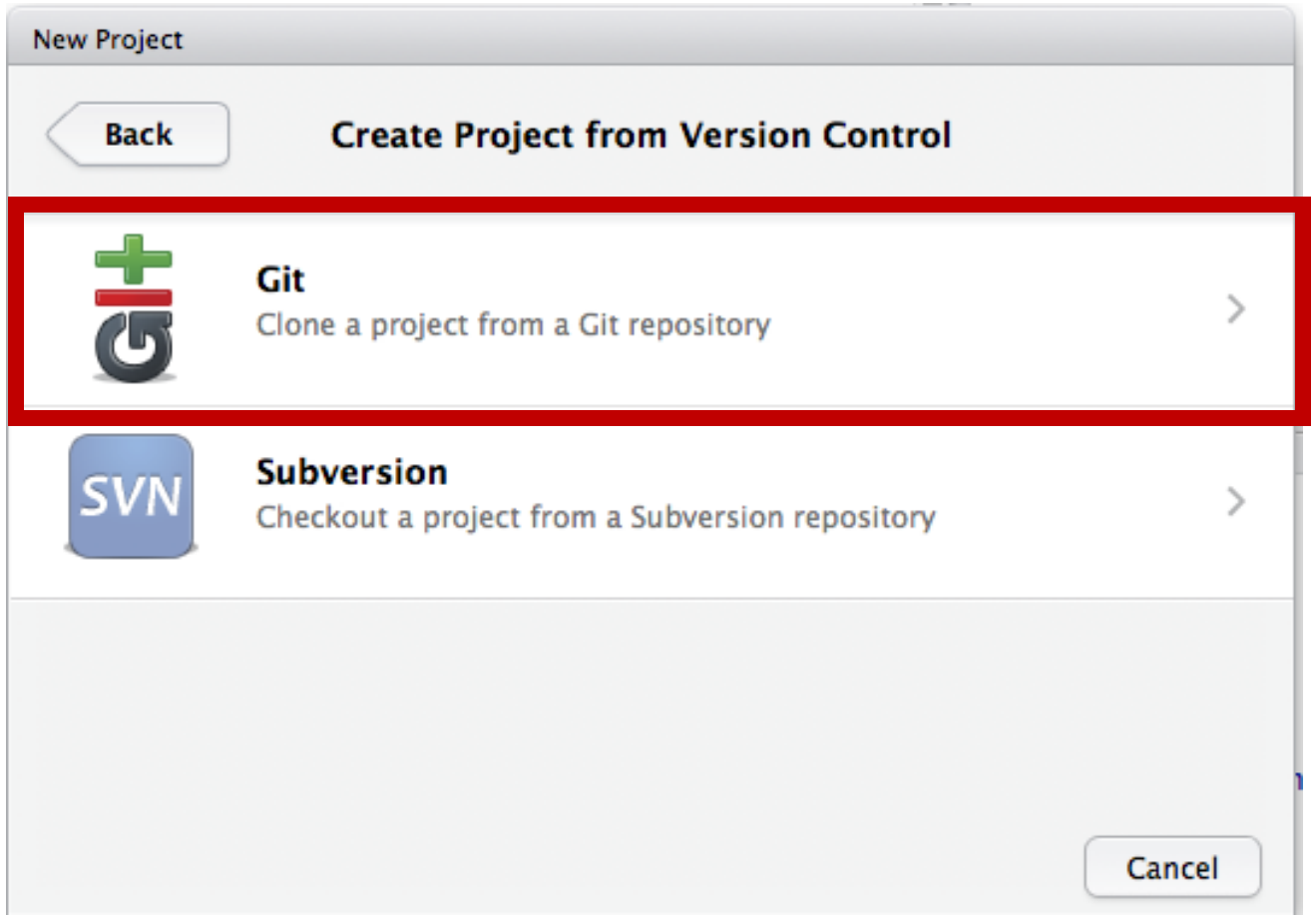
Select Version Control

File > New Project

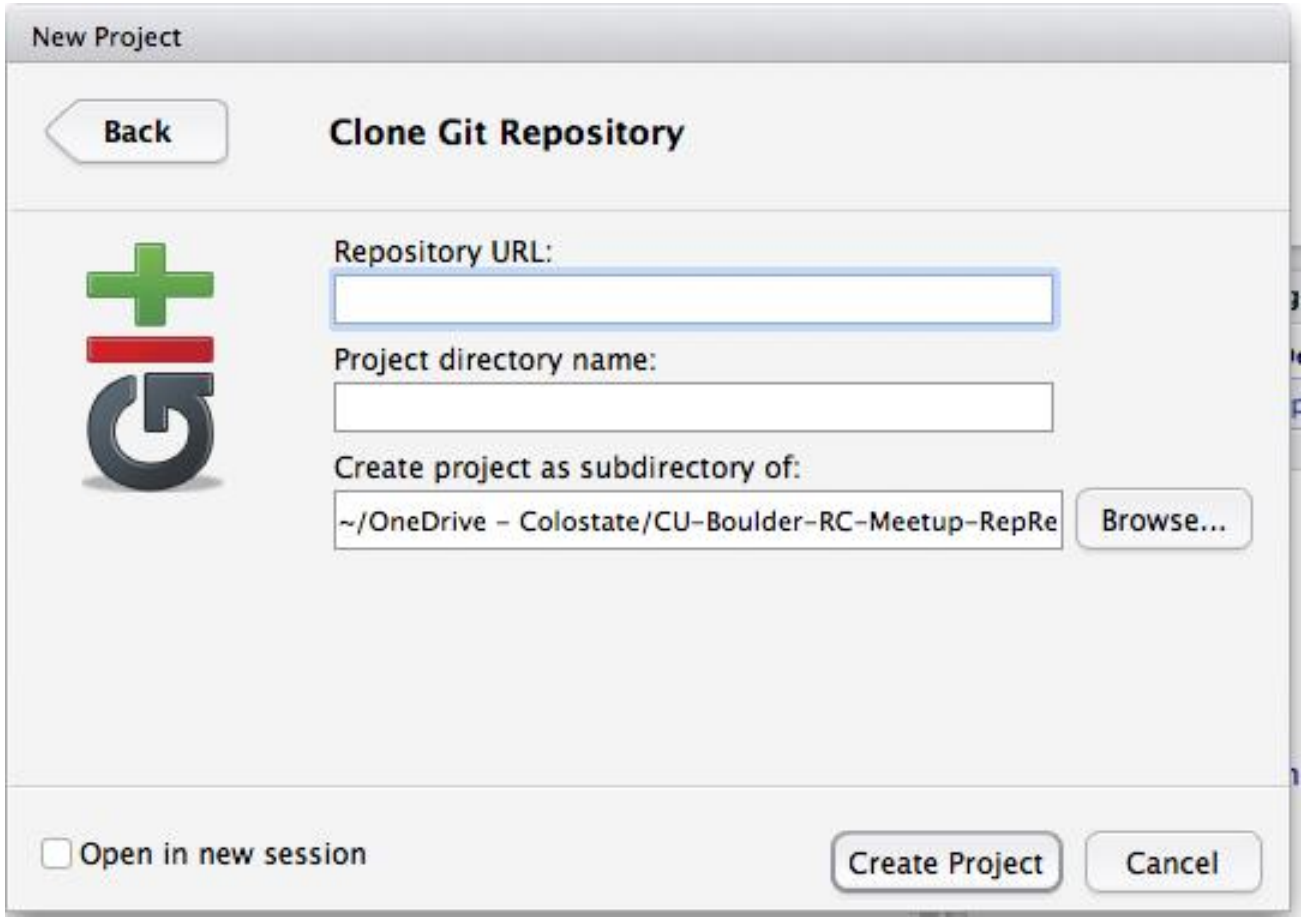
- no version control
- auto-detects existing version control
- **creates a version-controlled directory**



Select git

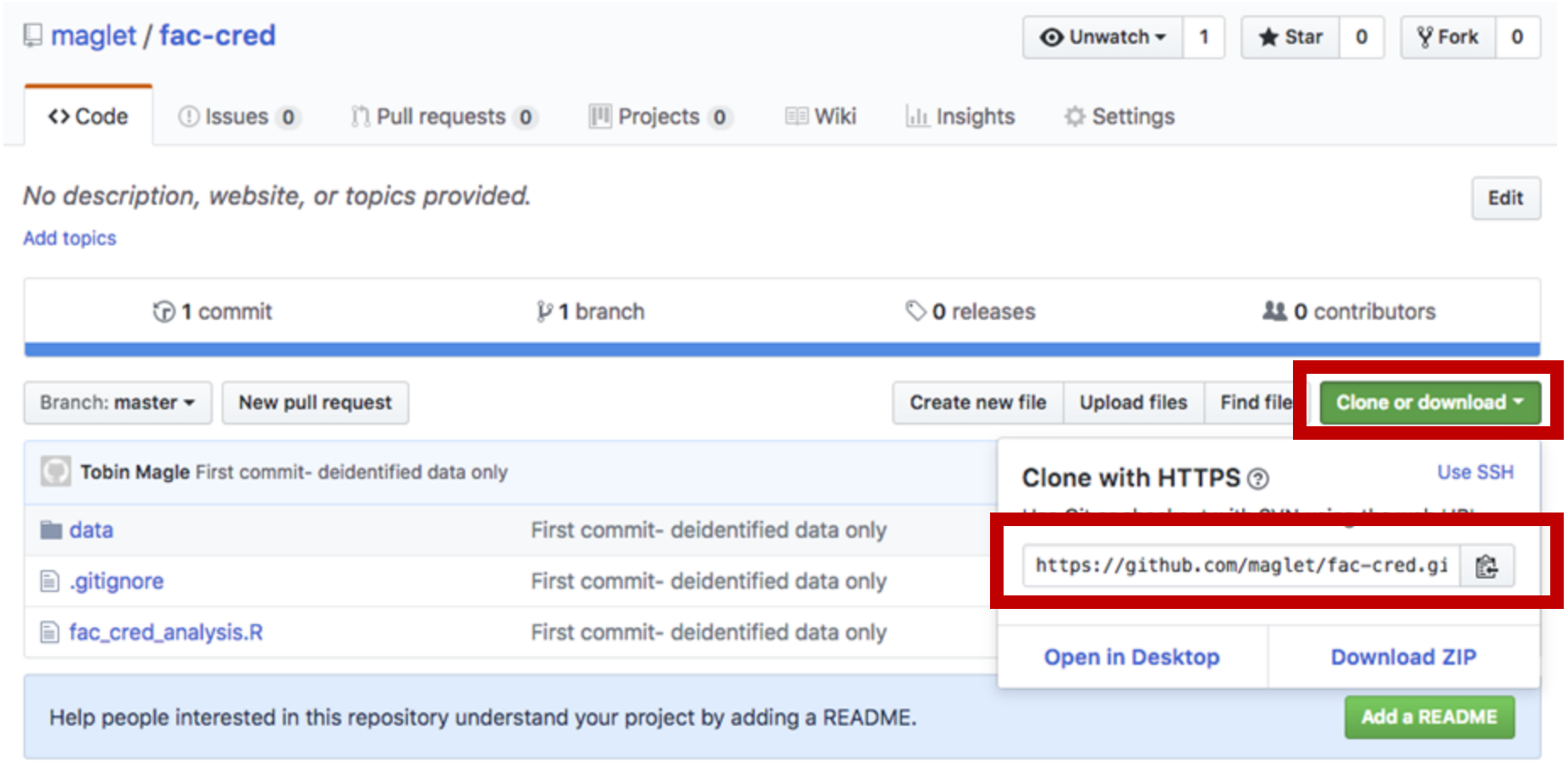

New Project
Back
Create Project from Version Control
Git
Clone a project from a Git repository
SVN
Subversion
Checkout a project from a Subversion repository
Cancel

Link to GitHub



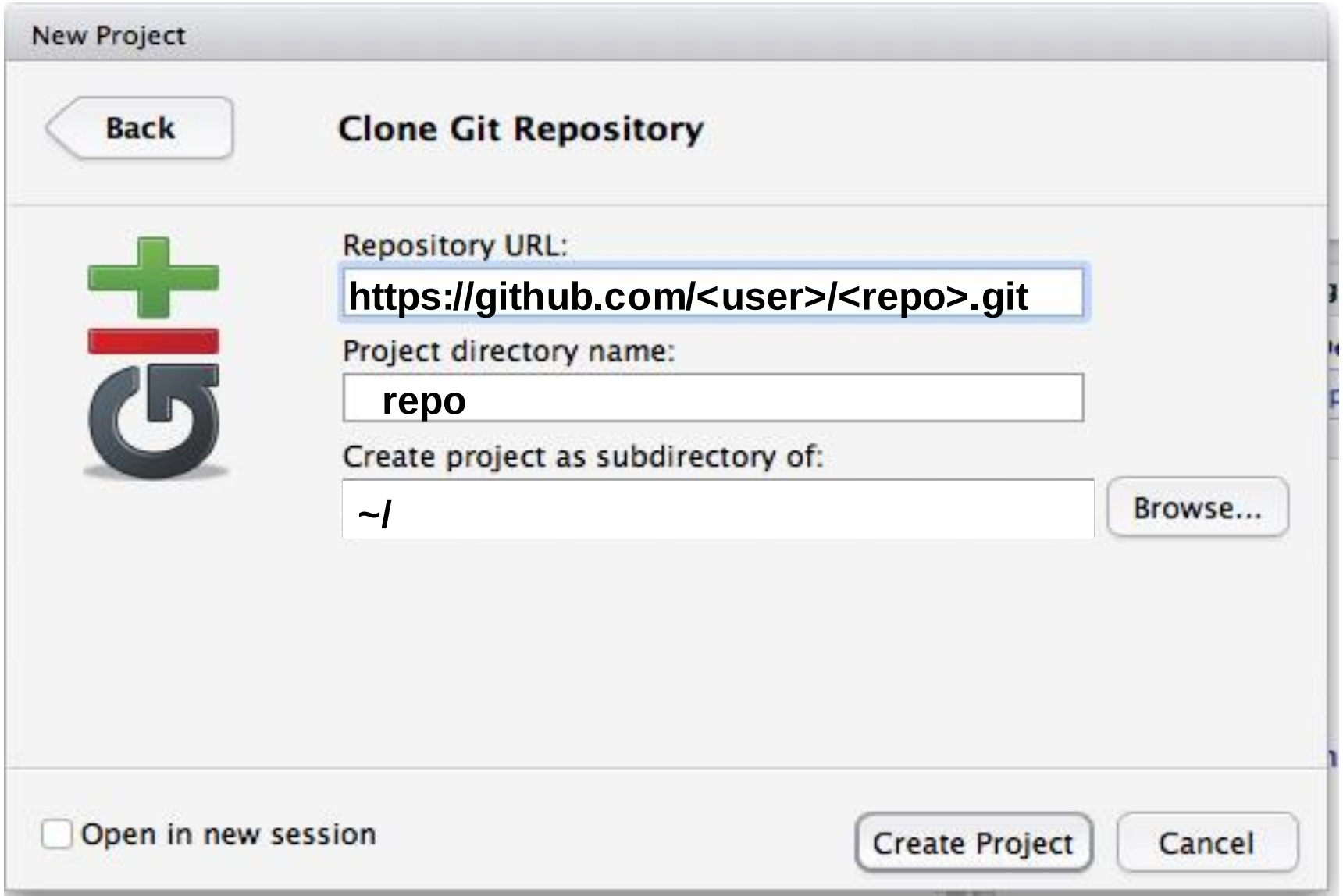
- URL from github
- New dir name
- Parent directory

Get GitHub URL



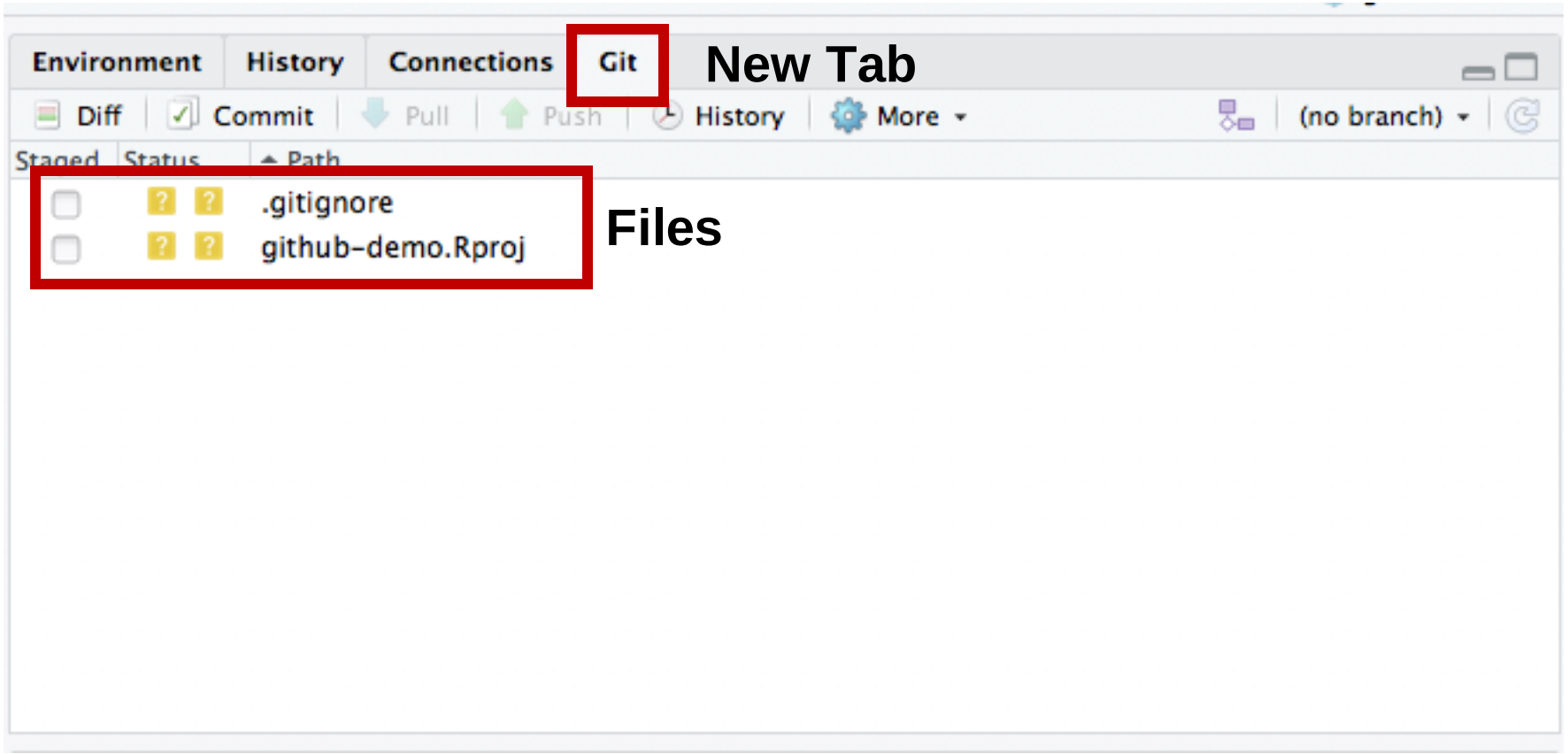
- Click
- Copy

Paste GitHub URL into RStudio



- Paste
- Match git repo name
- Pick

Upper right panel should look like this:



Untracked

Edited files recognized by git, not acted upon

?

Staging Area

Edited files to be added to your repository

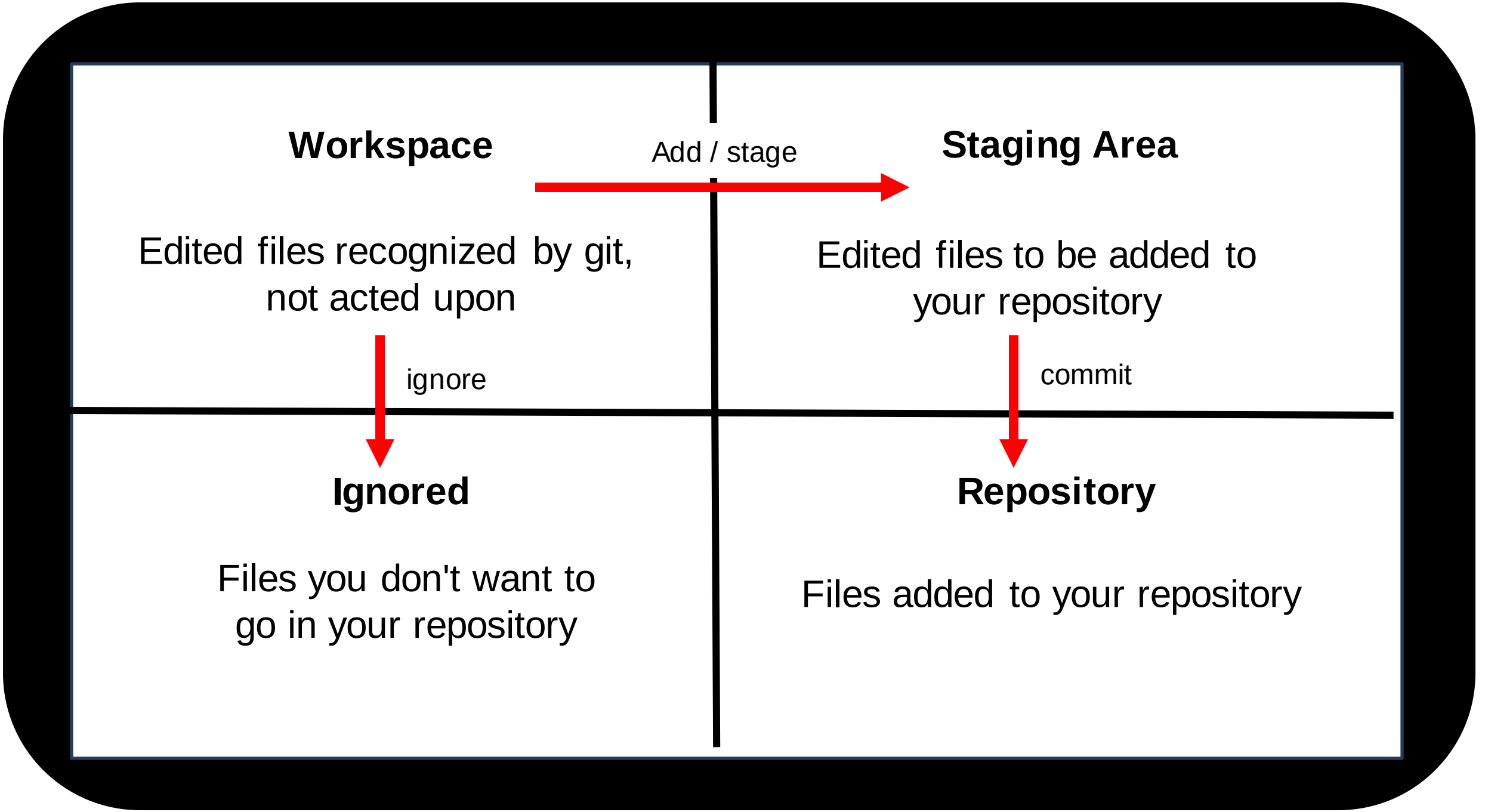
A

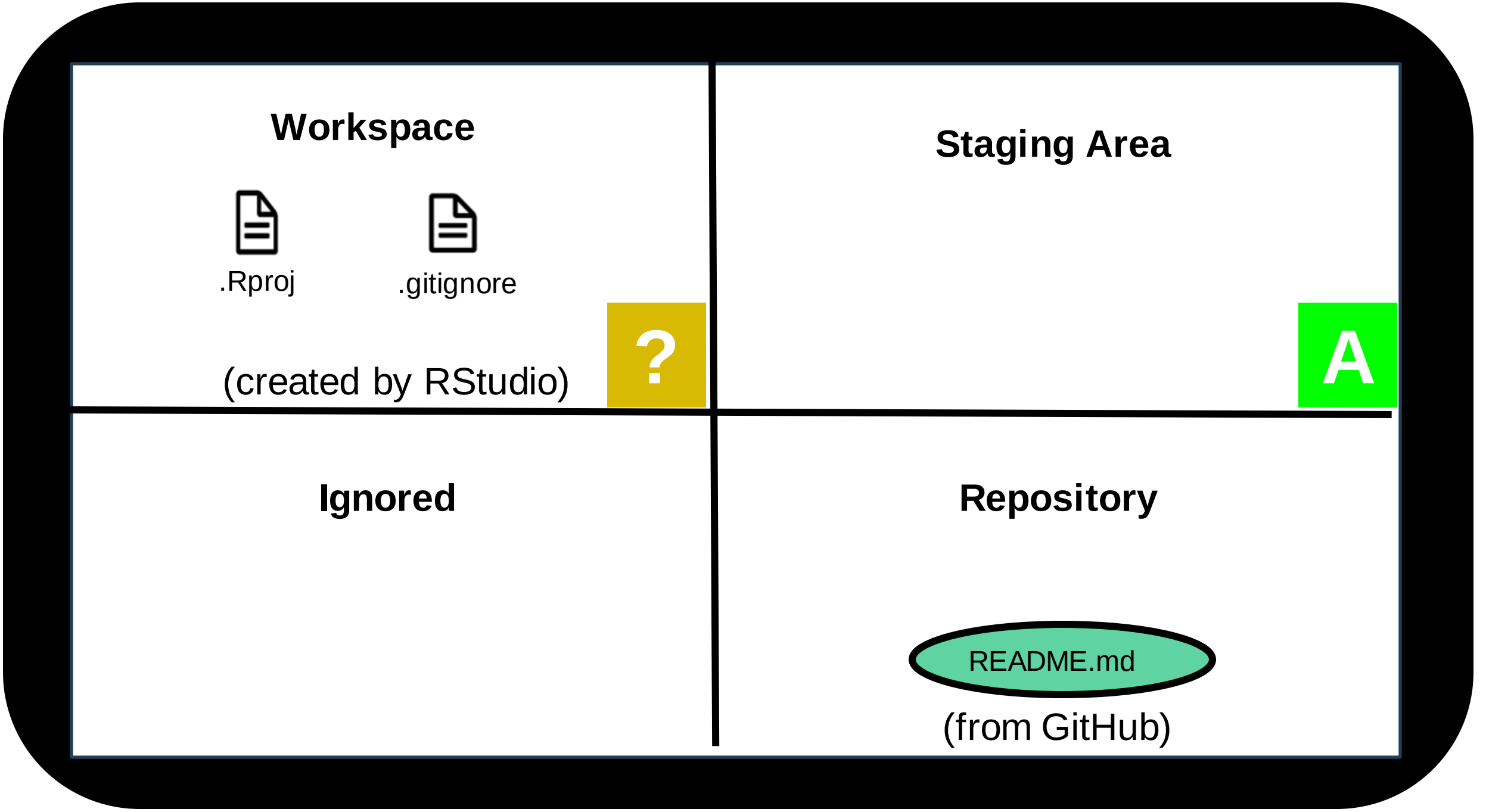
Ignored

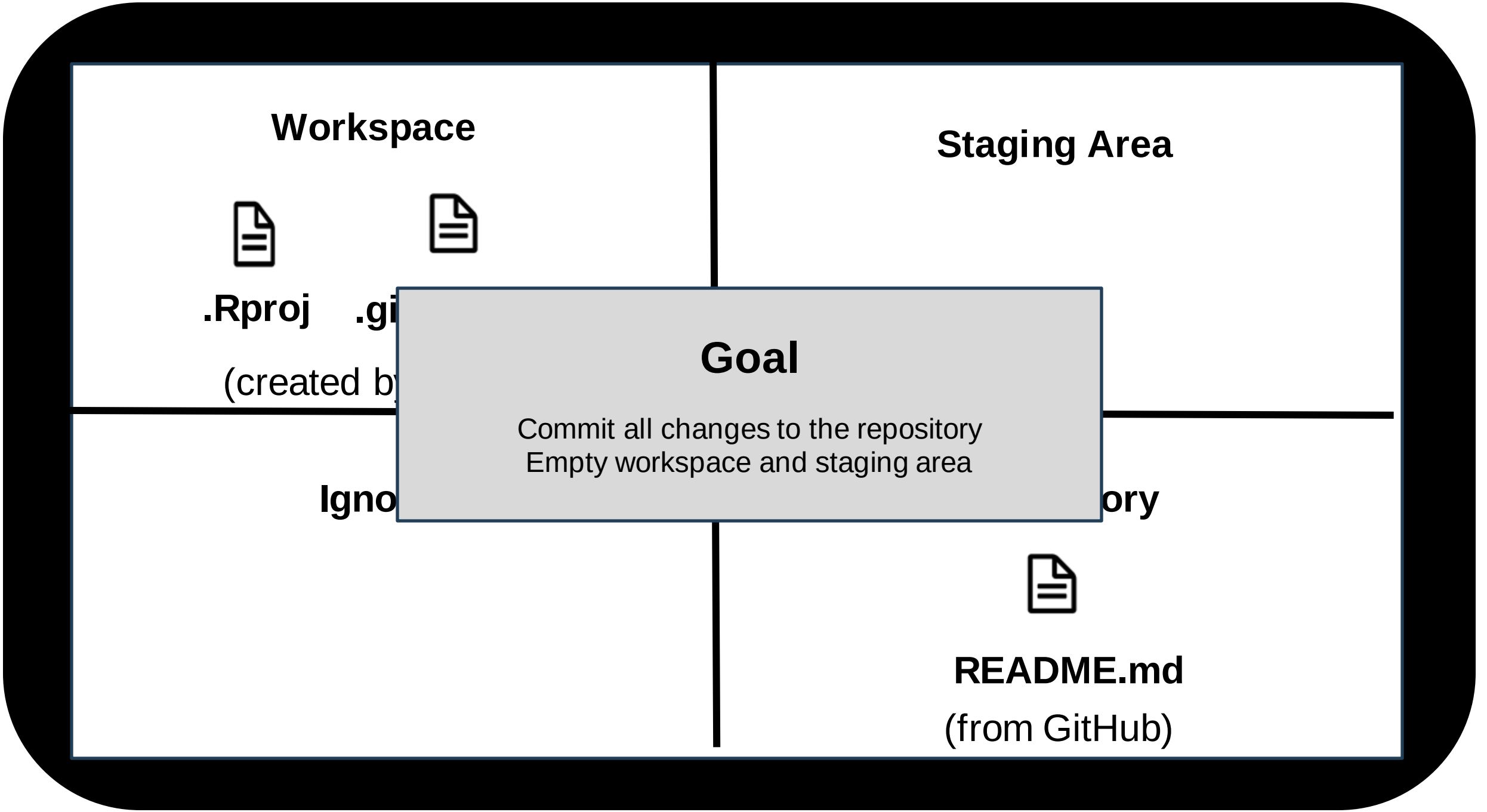
Files you don't want to go in your repository

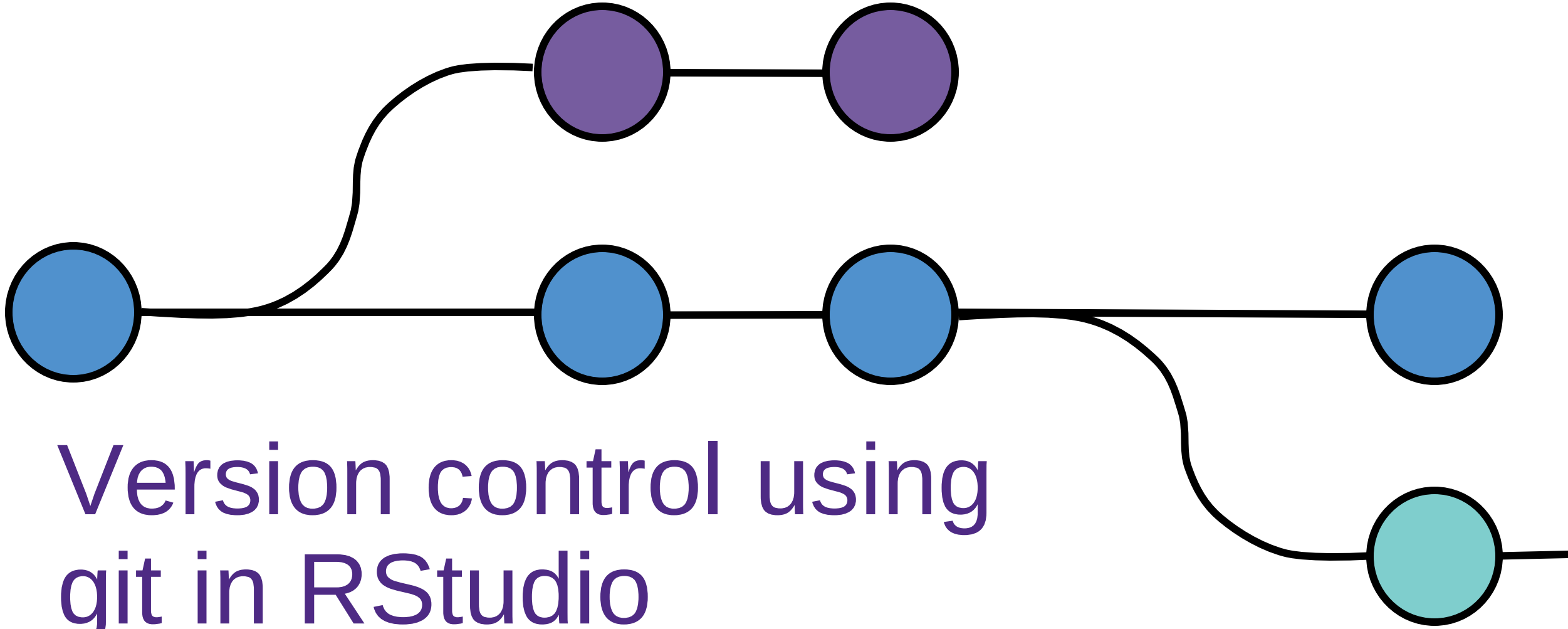
Repository

Files added to your repository


Workspace
Add / stage
Staging Area
Edited files recognized by git, not acted upon
Edited files to be added to your repository
ignore
commit
Ignored
Repository
Files you don't want to go in your repository
Files added to your repository


Workspace
.Rproj
.gitignore
(created by RStudio)
?
Staging Area
A
Ignored
Repository
README.md
(from GitHub)

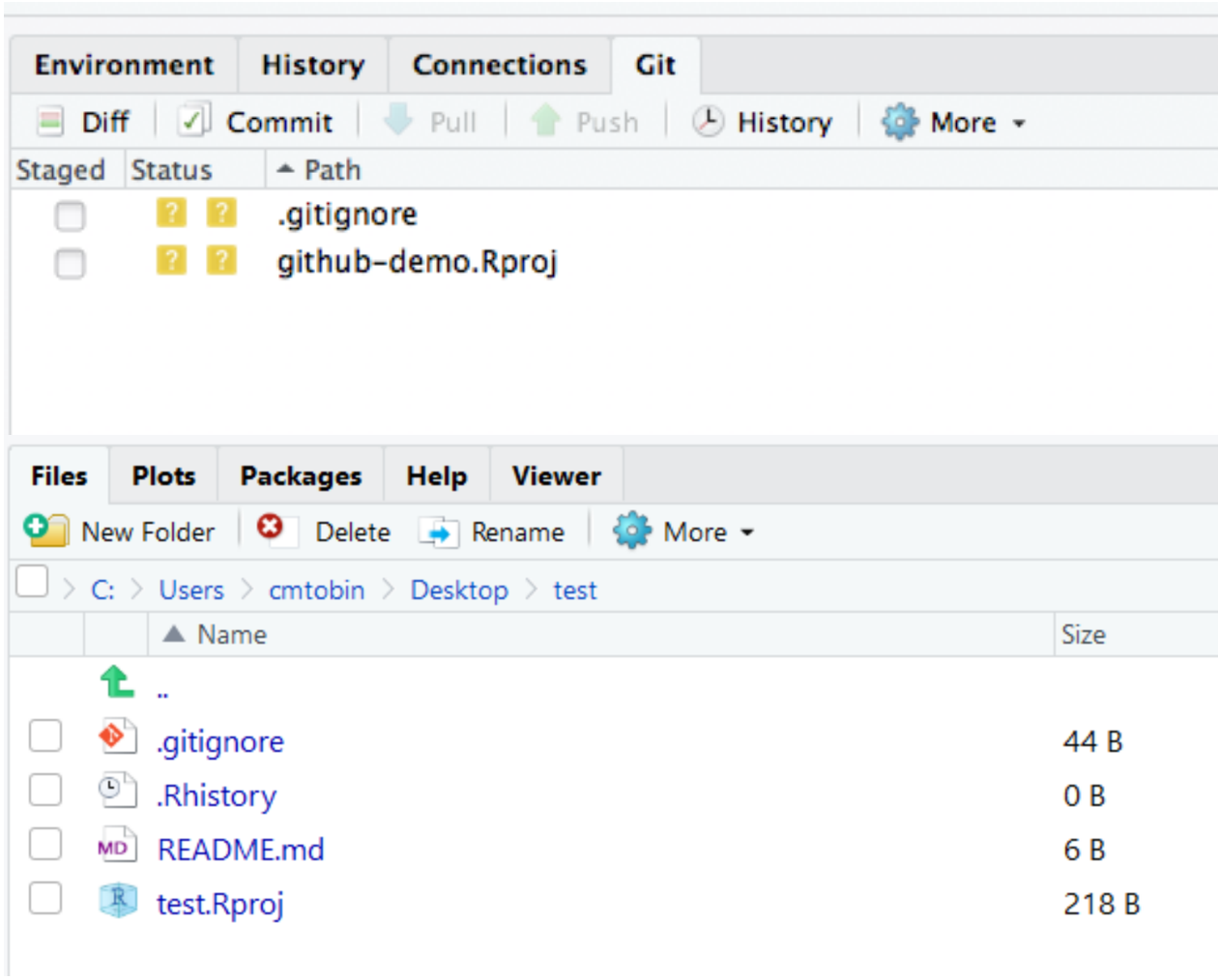

Workspace
Staging Area
.Rproj
.gi
(created b
Goal
Commit all changes to the repository
Empty workspace and staging area
Igno
ory
README.md
(from GitHub)



Version control using git in RStudio

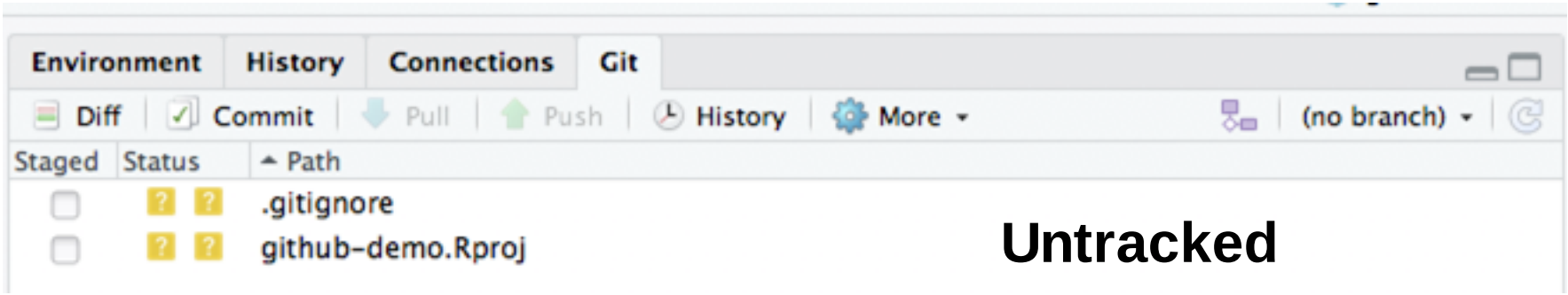
Setup results

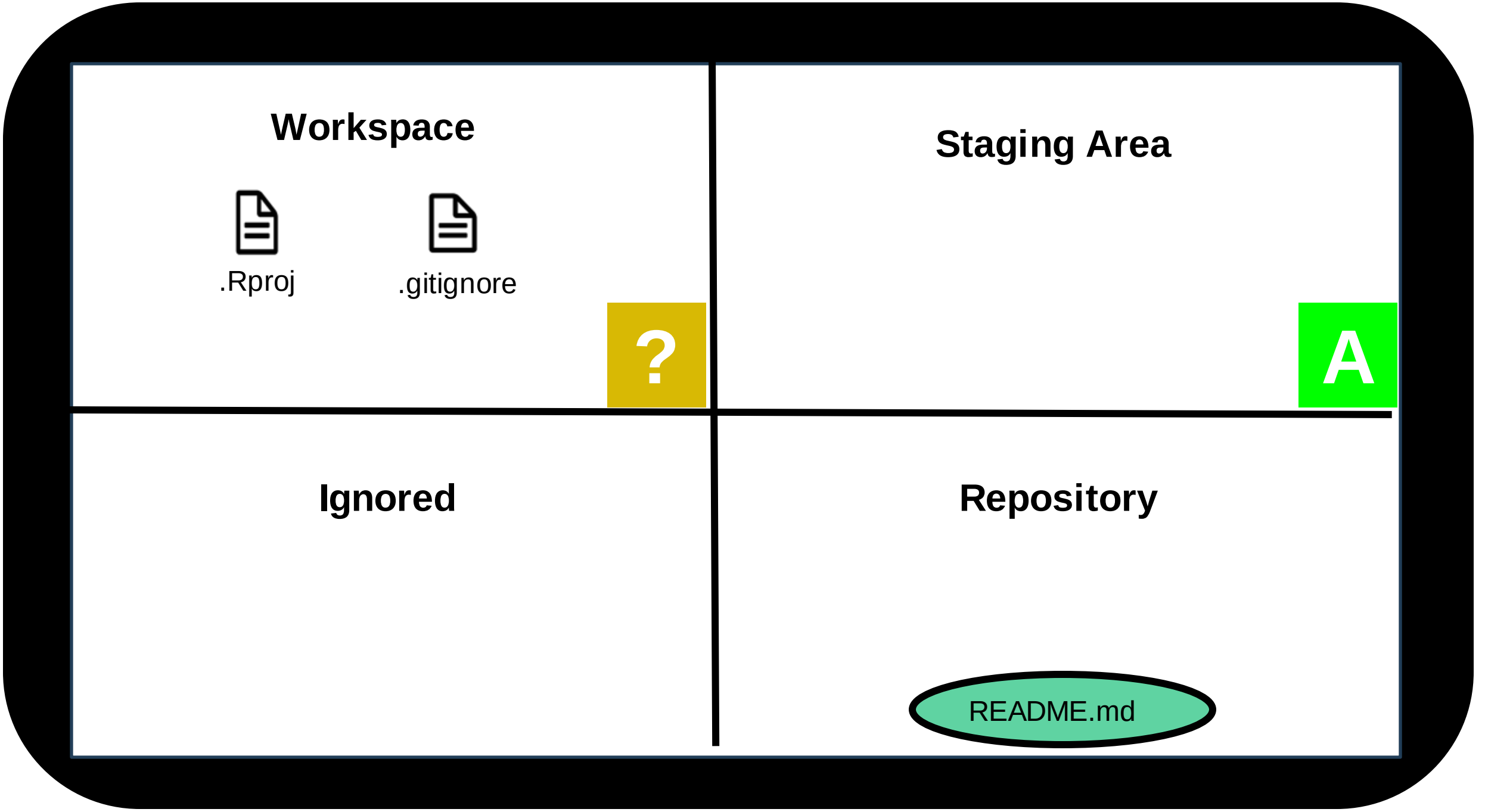
- Downloaded files from Github
  - README.md
- New files added to your project folder
  - .gitignore, .Rproj, .Rhistory
- Ignore files
  - .Rhistory, .Rproj.user, .Rdata, .Ruserdata



Git tab

- Only files with uncommitted changes appear
- README.md was already in the repository (committed)
- .Rhistory is being ignored (will address this later)

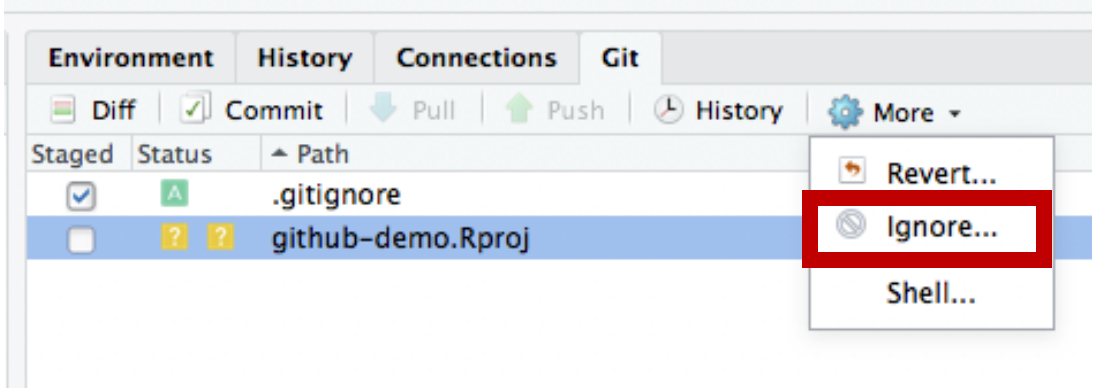



Workspace
.Rproj
.gitignore
?
Staging Area
A
Ignored
Repository
README.md

Ignoring files

- Git by default “sees” all files in its directory unless you tell it not to

- Add the filename to the .gitignore file to stop tracking



Types of files git works best with

“Good” formats (text based)

- Documents (.txt, .tex, .rtf, .Rmd)
- Tabular Data (.csv)
- Source code (.R, .py, .c, .sh)

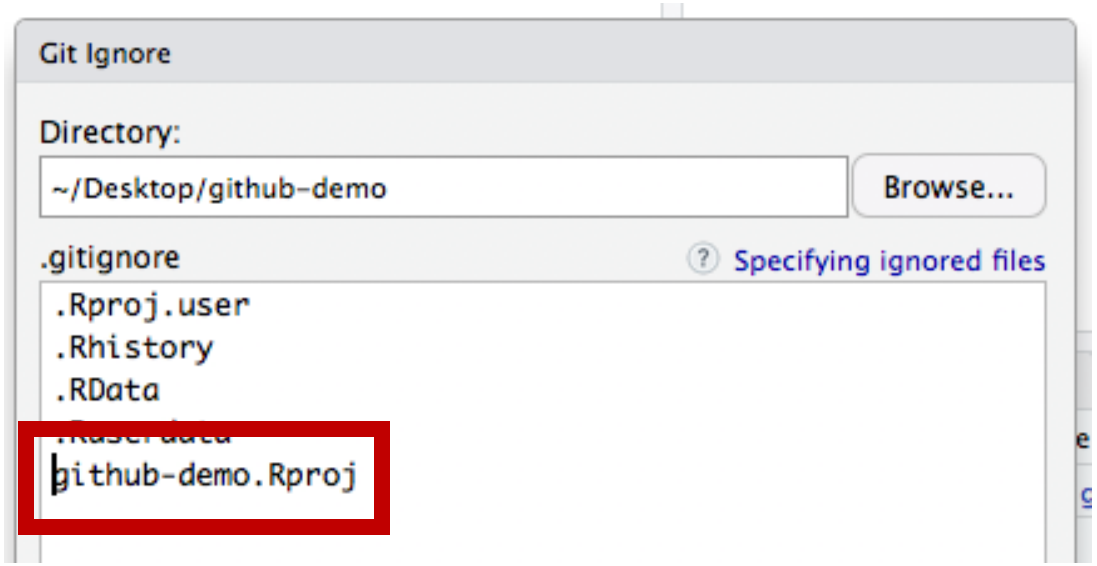
“Bad” formats (binary files)

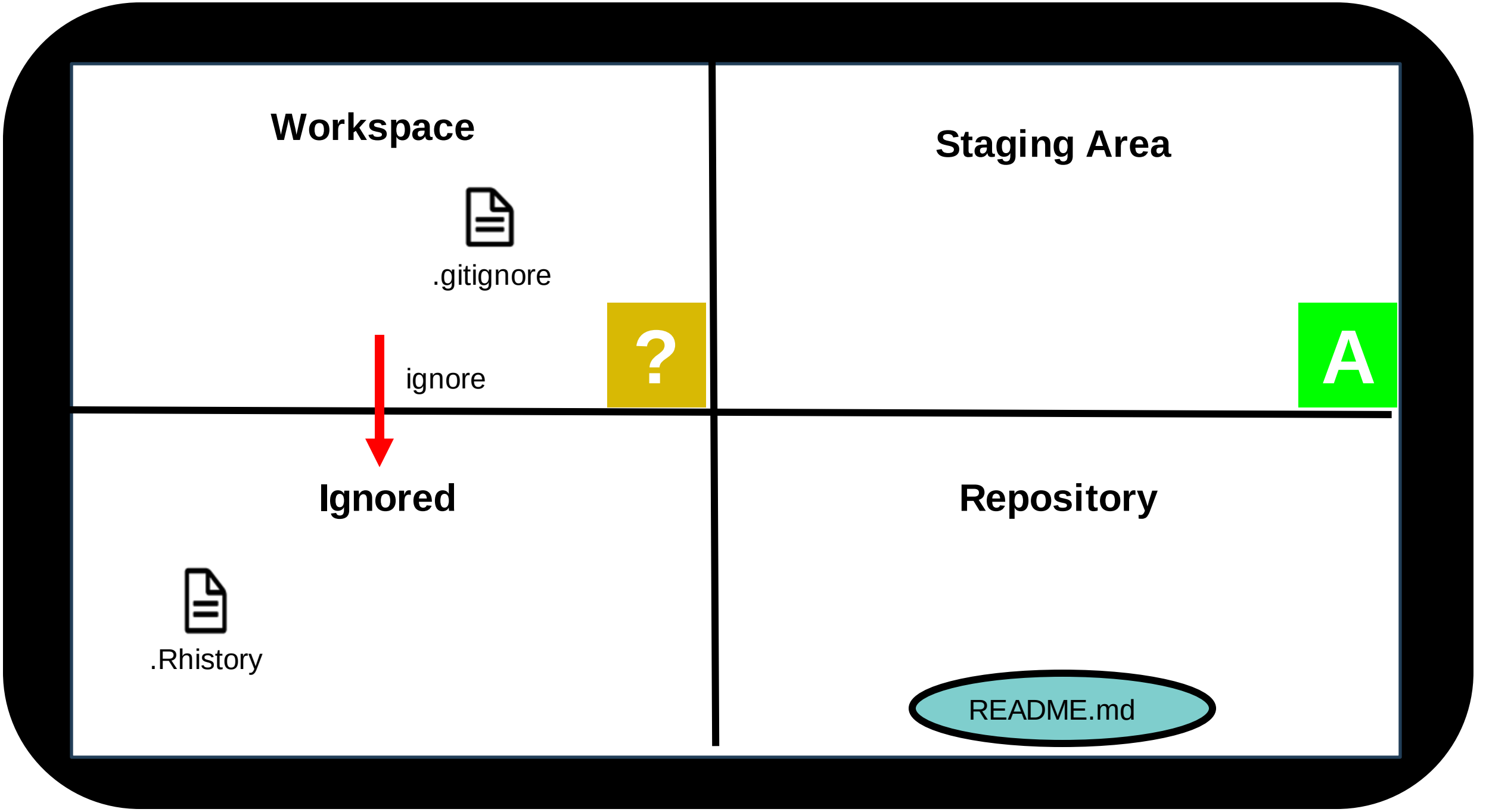
- Documents (.docx, .pdf, .ppt)
- Excel spreadsheets (.xlsx)
- Media (.jpg, .mp3, .mp4)
- Databases (.mdb, .sqlite)

https://www.nayuki.io/page/what-are-binary-and-text-files

Ignoring files

- Added the R.proj file to the ignore list

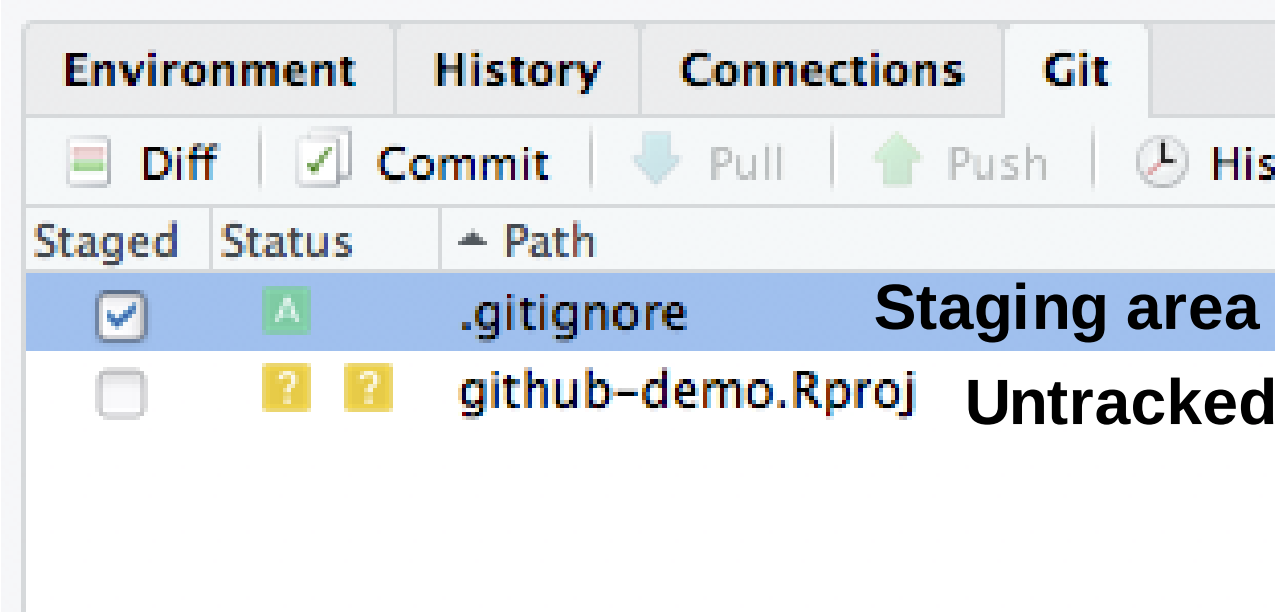


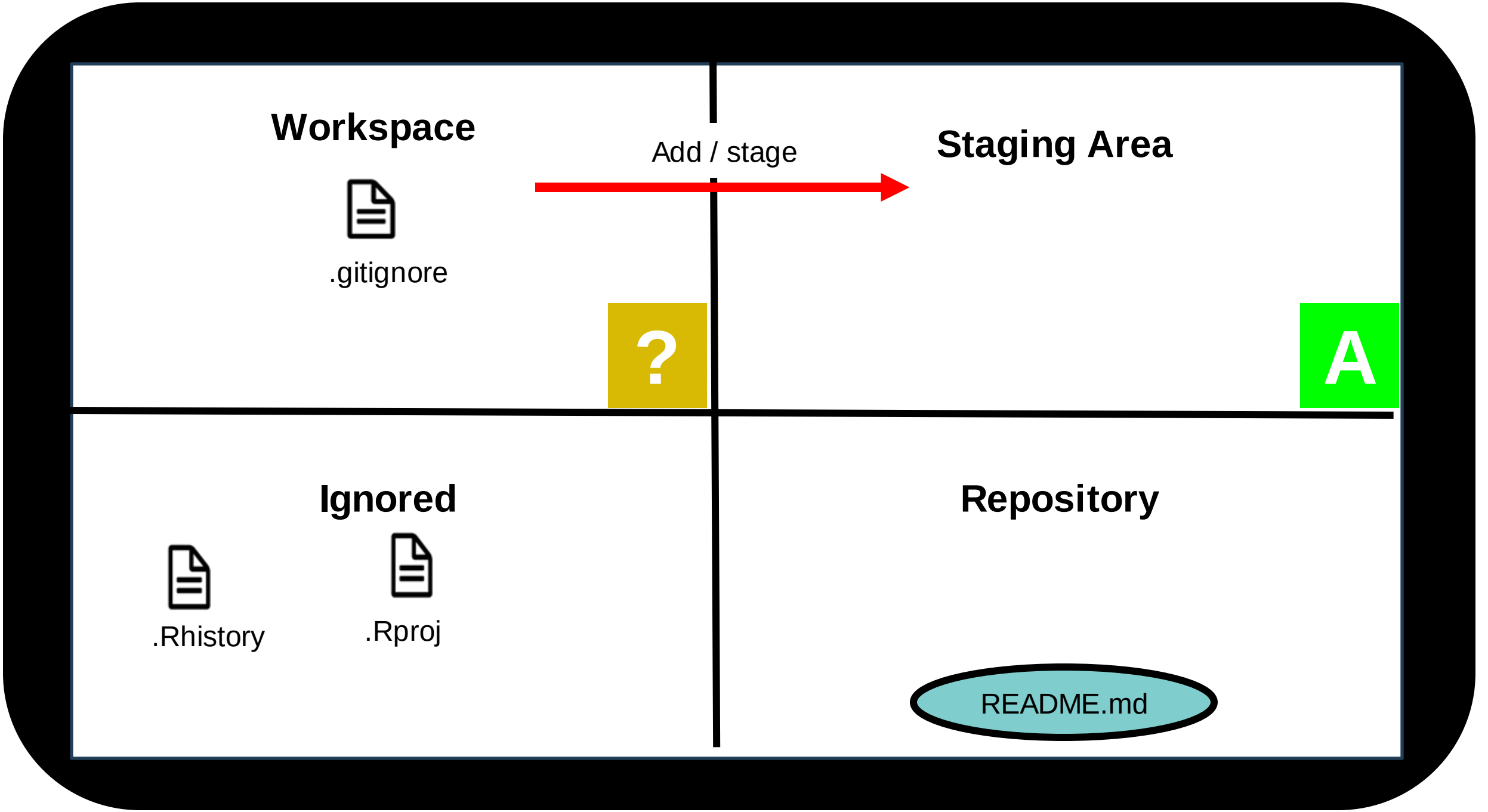

Workspace
Staging Area
.gitignore
?
A
ignore
Ignored
Repository
.Rhistory
README.md


Workspace
.gitignore
?
Staging Area
A
Ignored
.Rhistory
.Rproj
Repository
README.md

Stage (Add) files

- Check boxes to add files to the “staging area”
- **Status**: Staged
- Staging area compiles changes to be committed to the repository

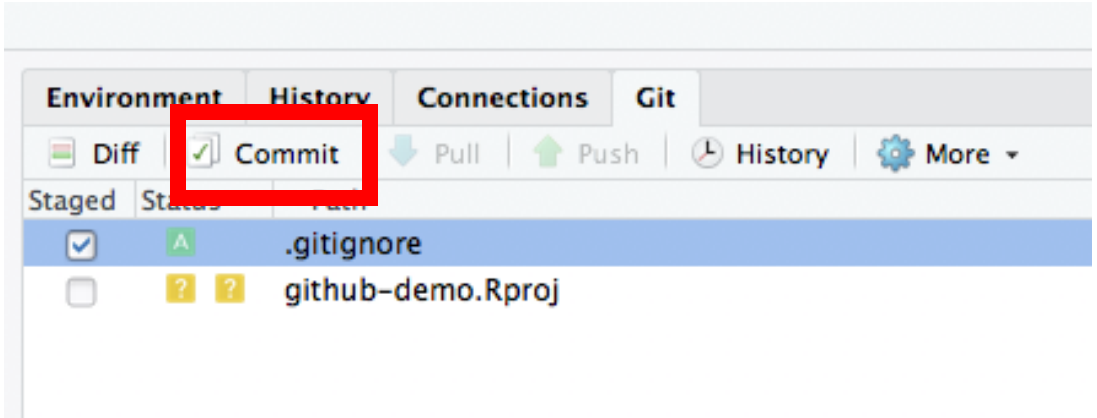



Workspace
Add / stage
Staging Area
.gitignore
?
A
Ignored
Repository
.Rhistory
.Rproj
README.md

Commit files

- Clicking “commit” adds all files from the staging area to the repository

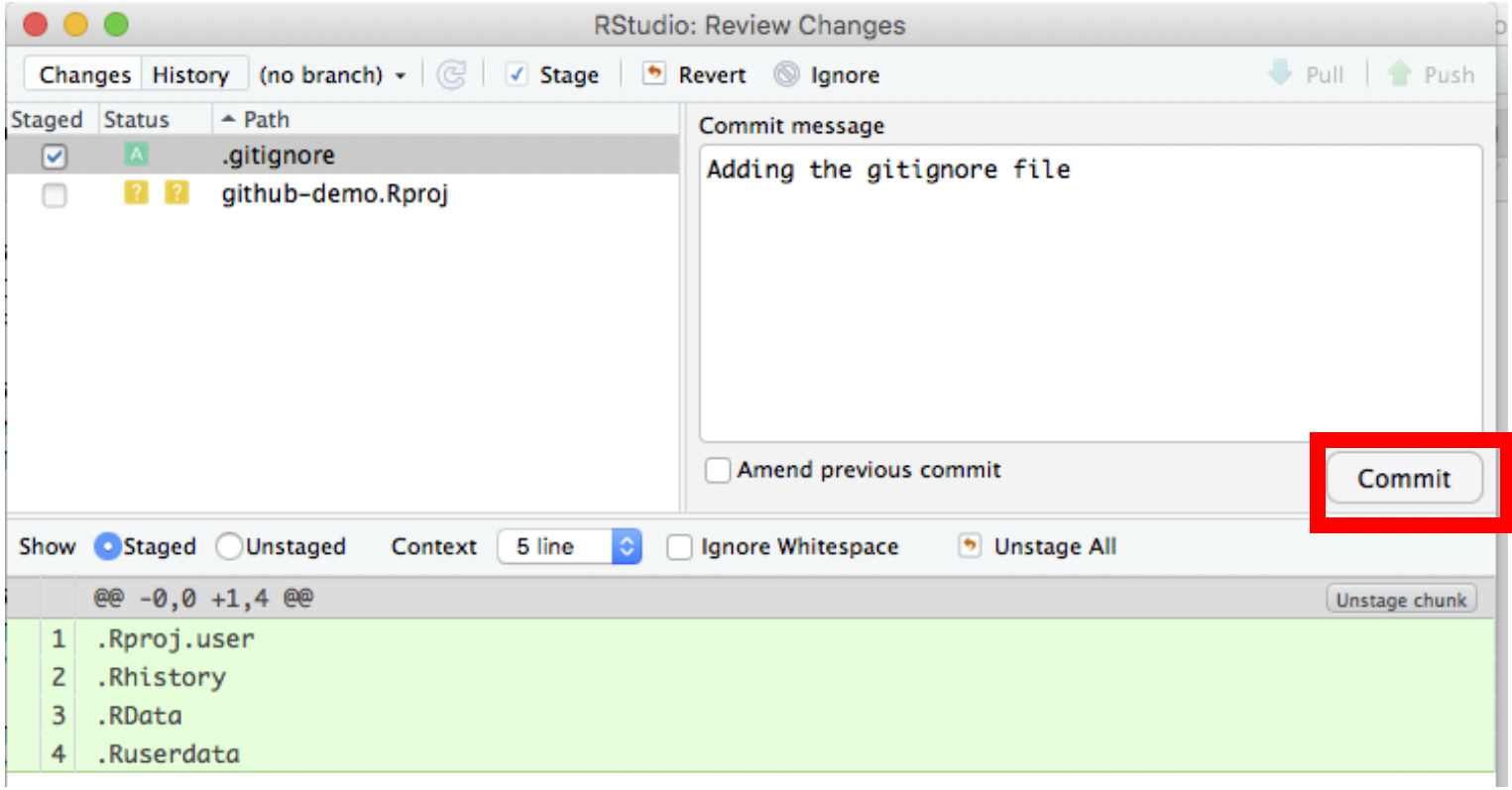
- Opens a new window

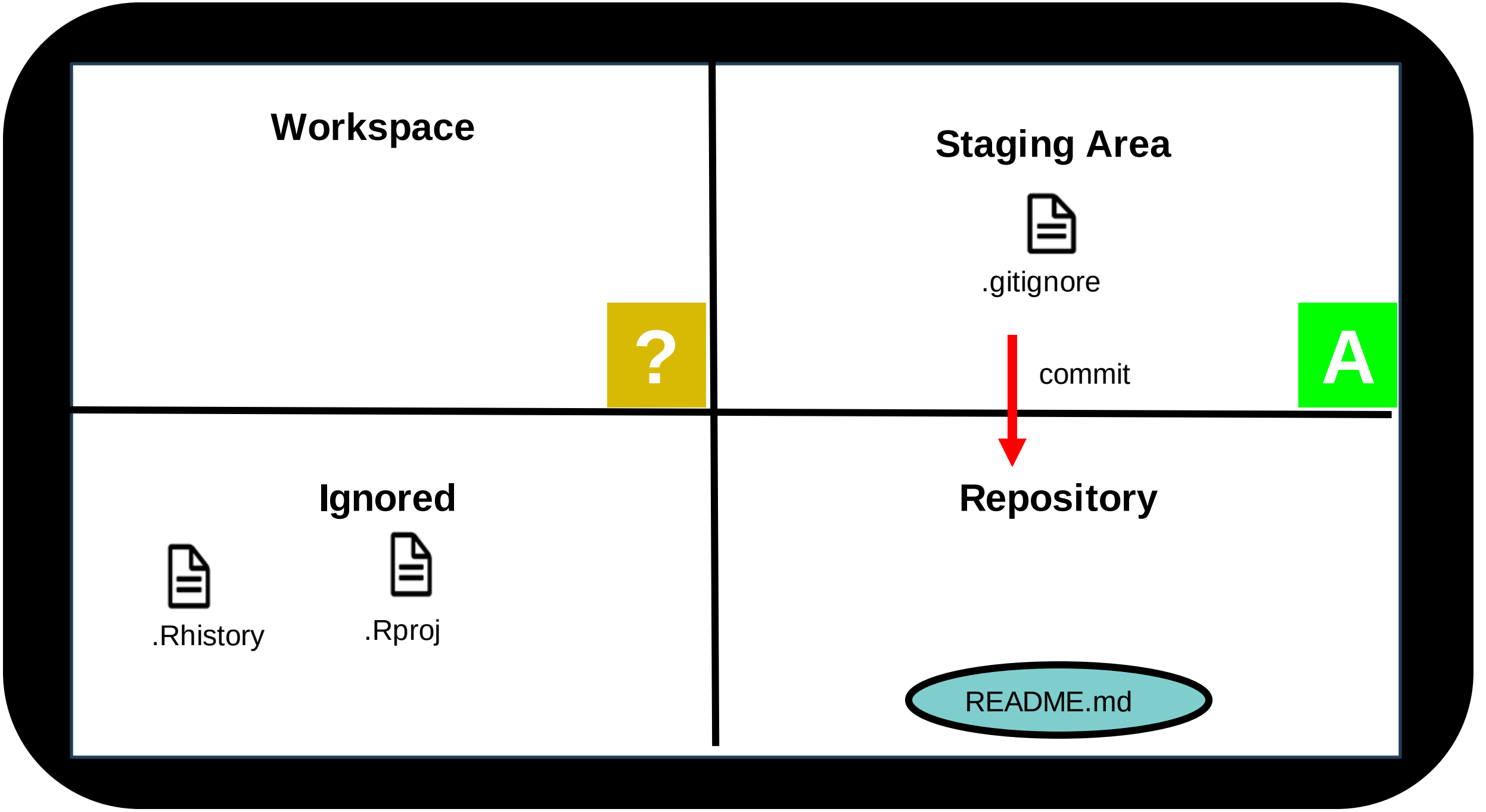


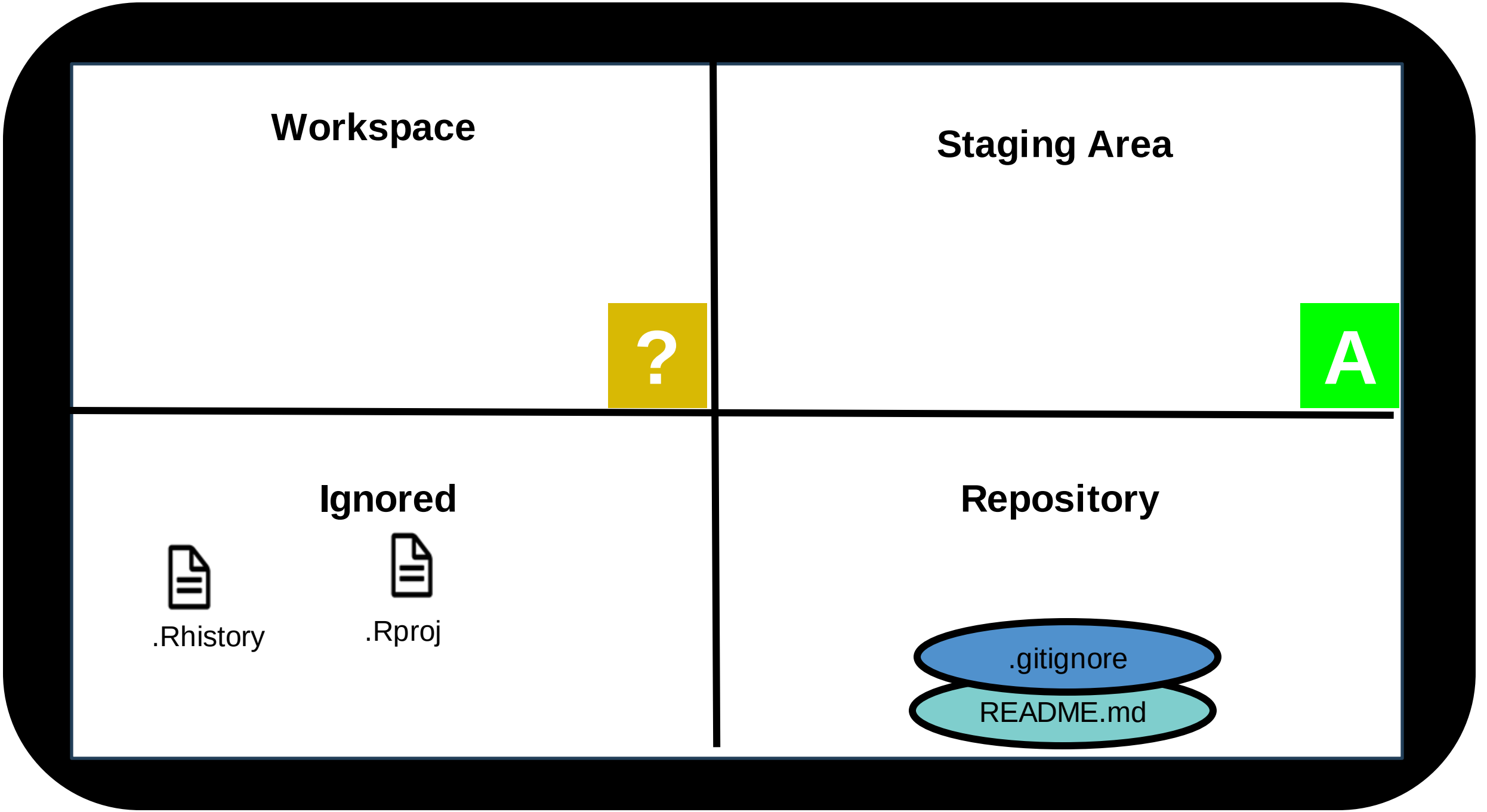
Commit message

- Add a message to say what changed in this commit

- Can see the contents of the file and what changed
  - Green = added
  - Red = removed

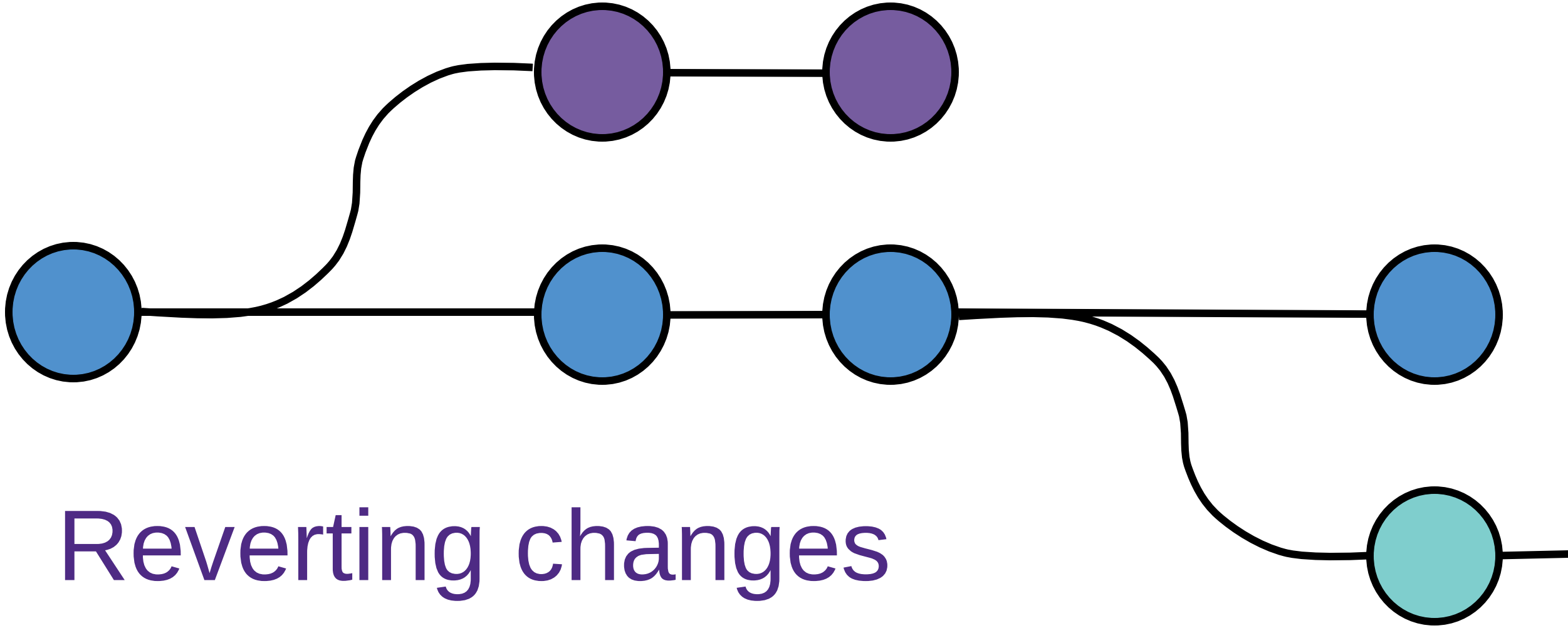



Workspace
Staging Area
.gitignore
?
commit
A
Ignored
Repository
.Rhistory
.Rproj
README.md


Workspace
?
Staging Area
A
Ignored
.Rhistory
.Rproj
Repository
.gitignore
README.md

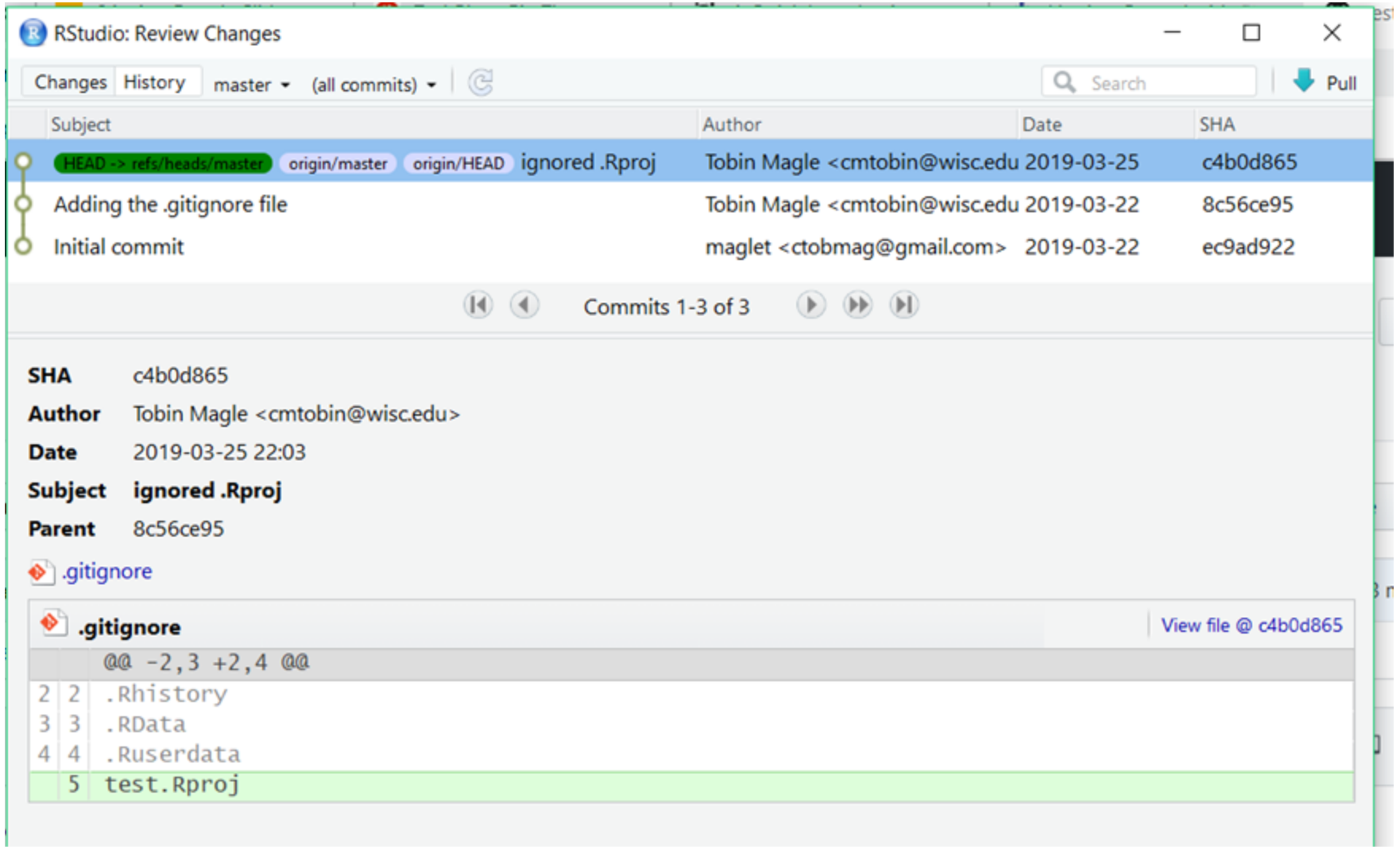
Exercise:

- Edit your README.md file
- **Stage changes: Add** the changes file to the **staging area**
- **Commit** your staging area to the **repository**



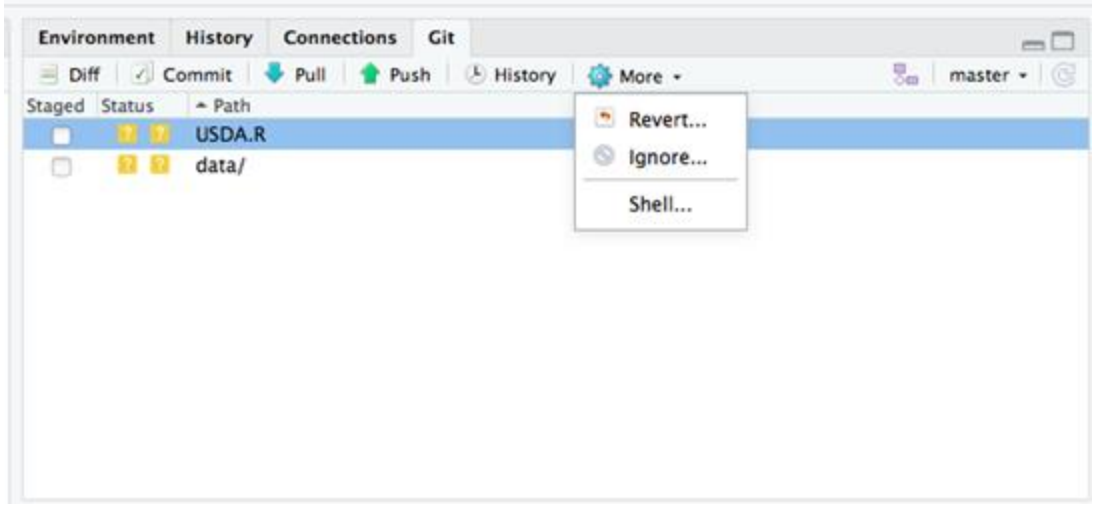
Reverting changes

History


RStudio: Review Changes
Changes
History
master
(all commits)
Search
Pull
Subject
Author
Date
SHA
HEAD -> refs/heads/master
origin/master
origin/HEAD
ignored .Rproj
Tobin Magle <cmtobin@wisc.edu
2019-03-25
c4b0d865
Adding the .gitignore file
Tobin Magle <cmtobin@wisc.edu
2019-03-22
8c56ce95
Initial commit
maglet <ctobmag@gmail.com>
2019-03-22
ec9ad922
Commits 1-3 of 3
SHA c4b0d865
Author Tobin Magle <cmtobin@wisc.edu>
Date 2019-03-25 22:03
Subject ignored .Rproj
Parent 8c56ce95
.gitignore
.gitignore
View file @ c4b0d865
@@ -2,3 +2,4 @@
.Rhistory
.RData
.Ruserdata
test.Rproj

Revert

- Removes all uncommitted changes
- Includes staged changes
- Can’t revert to committed changes from R Studio

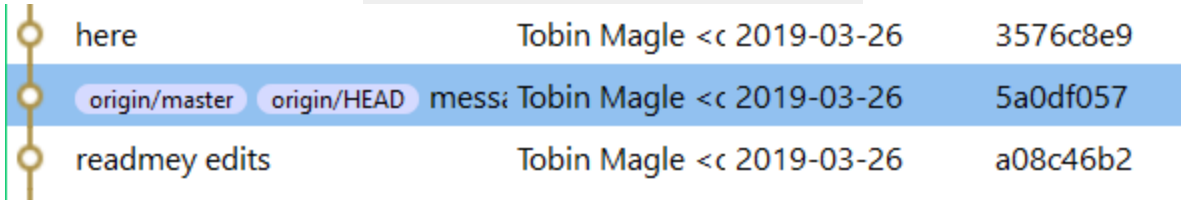


https://intro2r.com/use_git.html#undo

Go back to previous commit

- Need to use the terminal
- git checkout <commit_id> <filename>
- File will show as modified, add and commit like normal

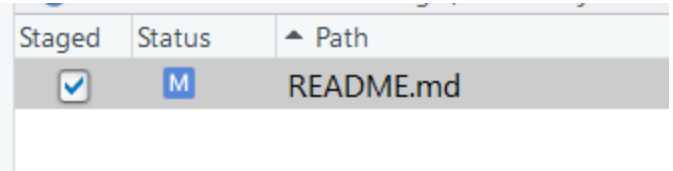
Find Commit ID



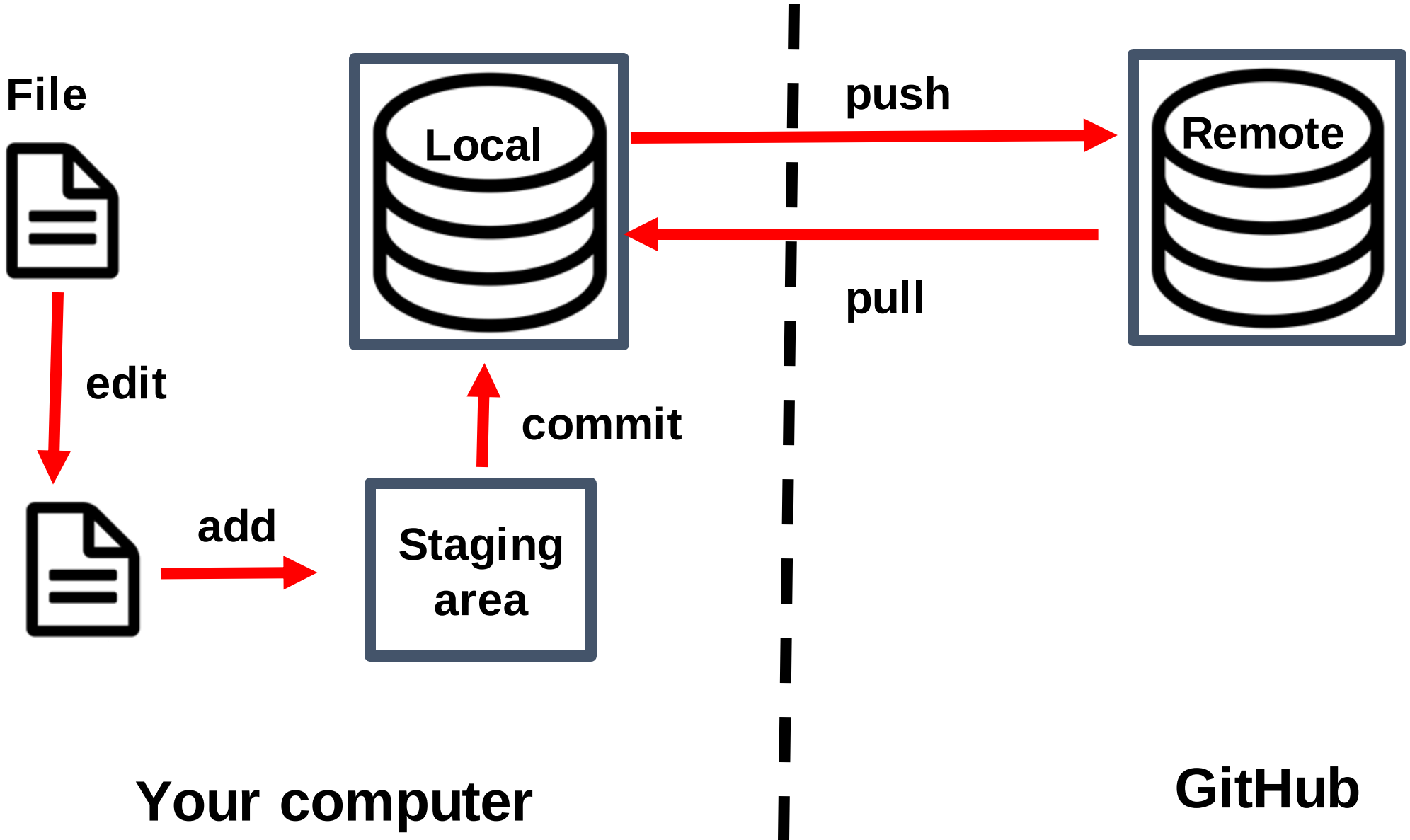
Run the checkout command in the terminal

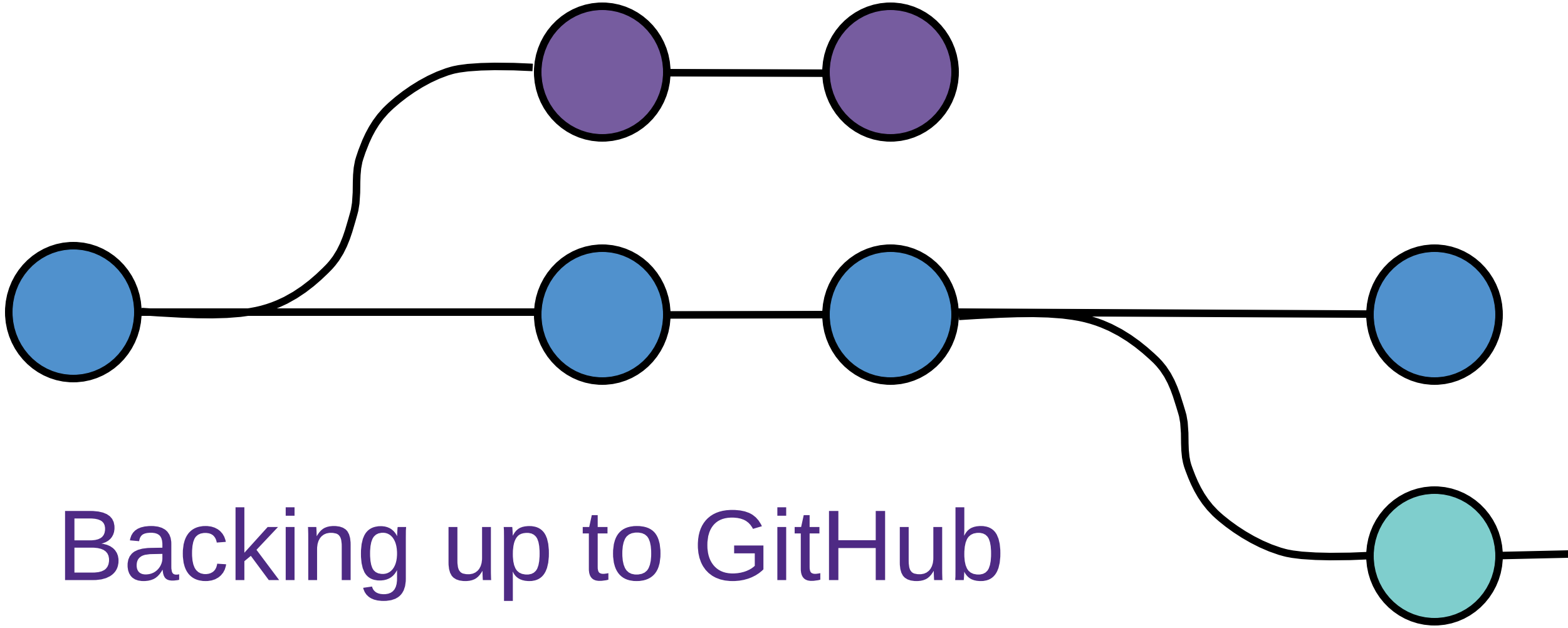
```
cmtobin@GLS-LAP8228  ~/Desktop/test2 (master)
$ git checkout 5a0df057 README.md
```

file has been modified



Git workflow

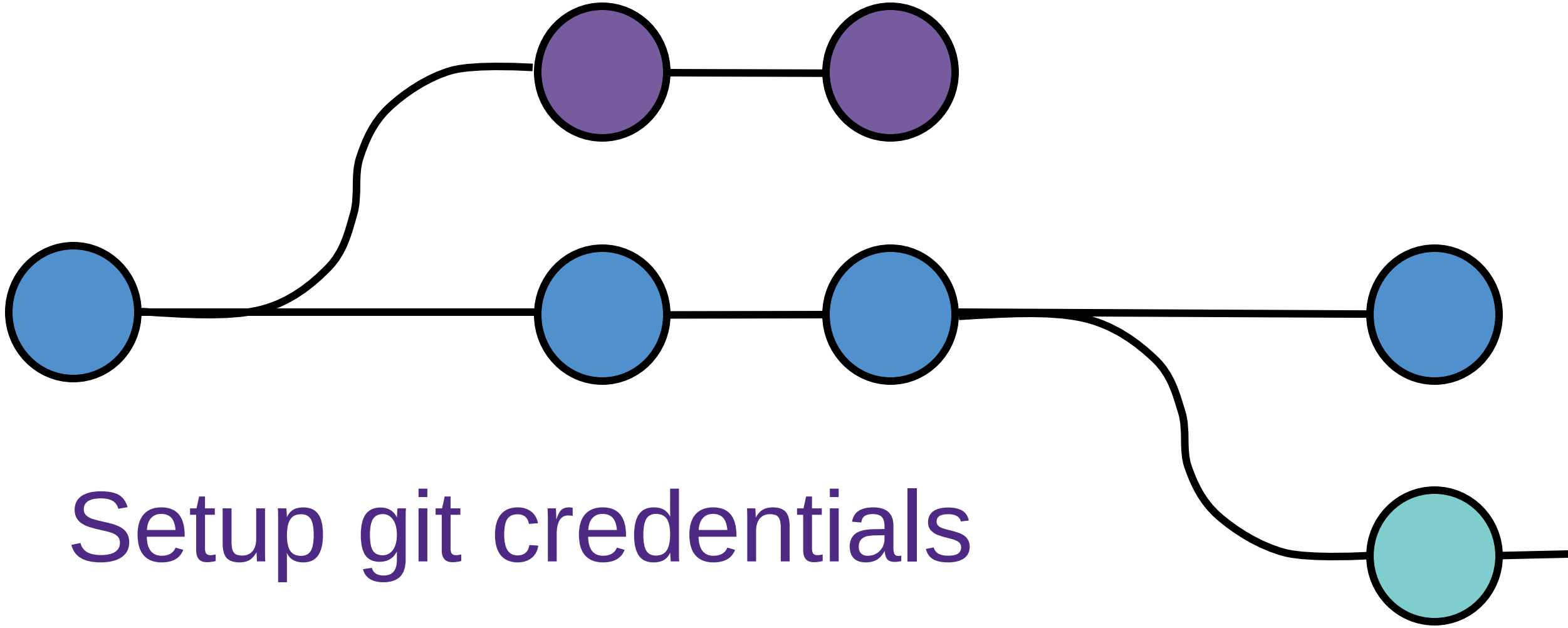




Backing up to GitHub

GitHub best practices

- Store **code and documentation**, not data
- Never store **protected or personal data** even in a private repo
- Store **credentials** (private keys or tokens) in a separate file outside of your repository
  - ie, not in your code
- Add a **personal email** to your GitHub account, so you can use it when you leave Northwestern



Setup git credentials

Setup

During the workshop

- GitHub: Create a personal access token
- RStudio:
  - Install usethis package
  - **Configure** your git user
  - Set up **git authentication**

Configure git on your computer

In RStudio console

1. Install and load `usethis`

```
install.packages("usethis")
library(usethis)
```

2. Set git username and email
   - Commit history shows username
   - **Use your GitHub account email**

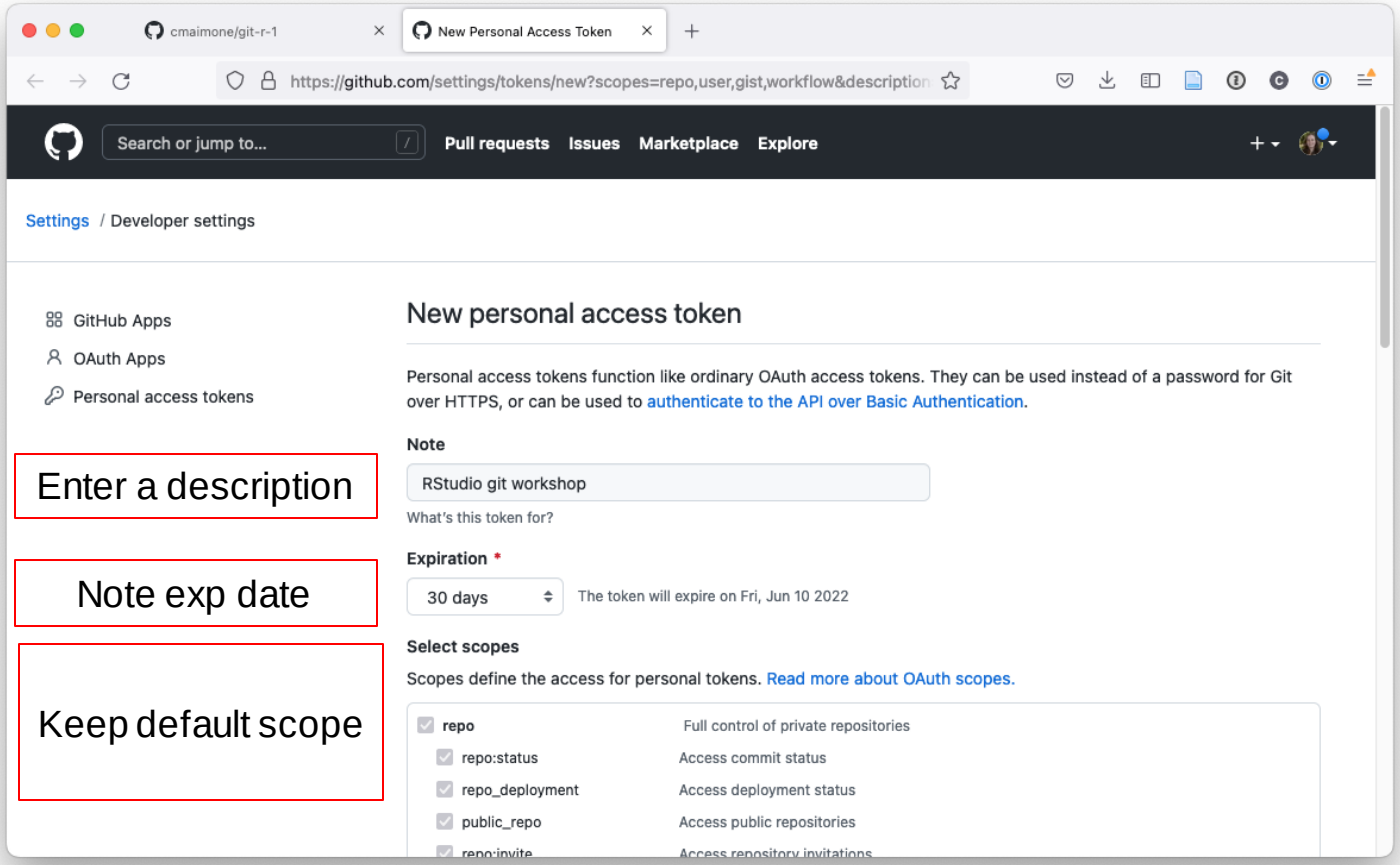
```
usethis::use_git_config(user.name="Jane Doe",
user.email="jane@example.org")
```

3. Set GitHub authentication

```
usethis::create_github_token()
```

Create GitHub Token

This page will open



Enter a description

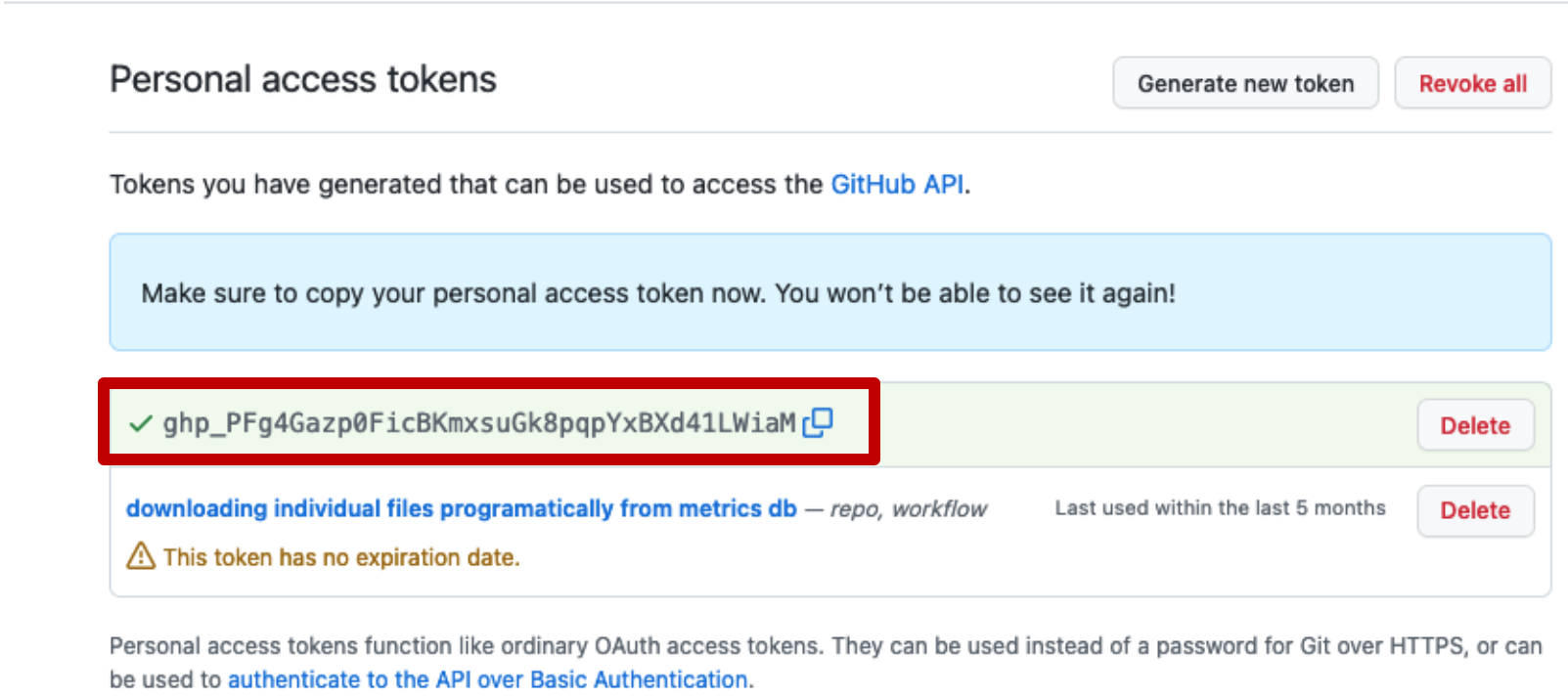
Note exp date

Keep default scope

More about scopes: https://docs.github.com/en/apps/oauth-apps/building-oauth-apps/scopes-for-oauth-apps

GitHub Personal Access Token

Copy token (it won't be visible once you leave the page)



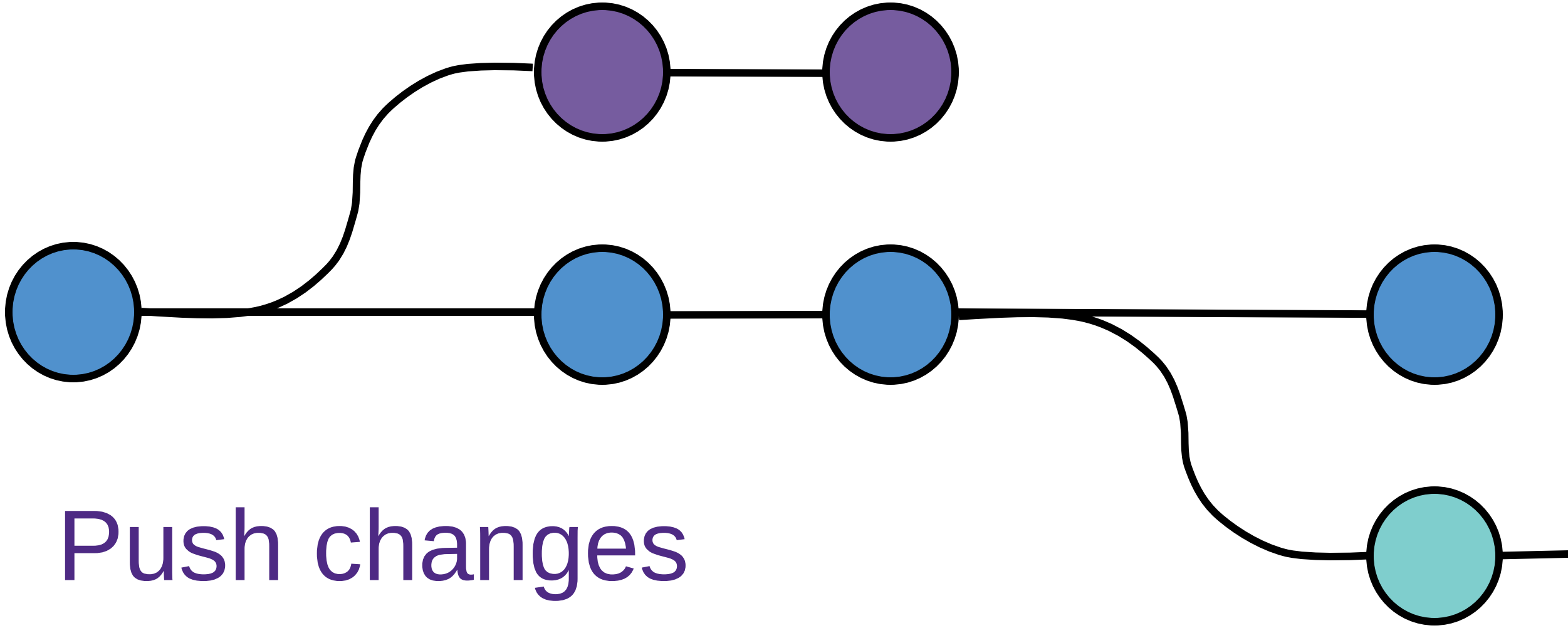
Set GitHub Credentials

In R Console:

```
gitcreds::gitcreds_set()
```

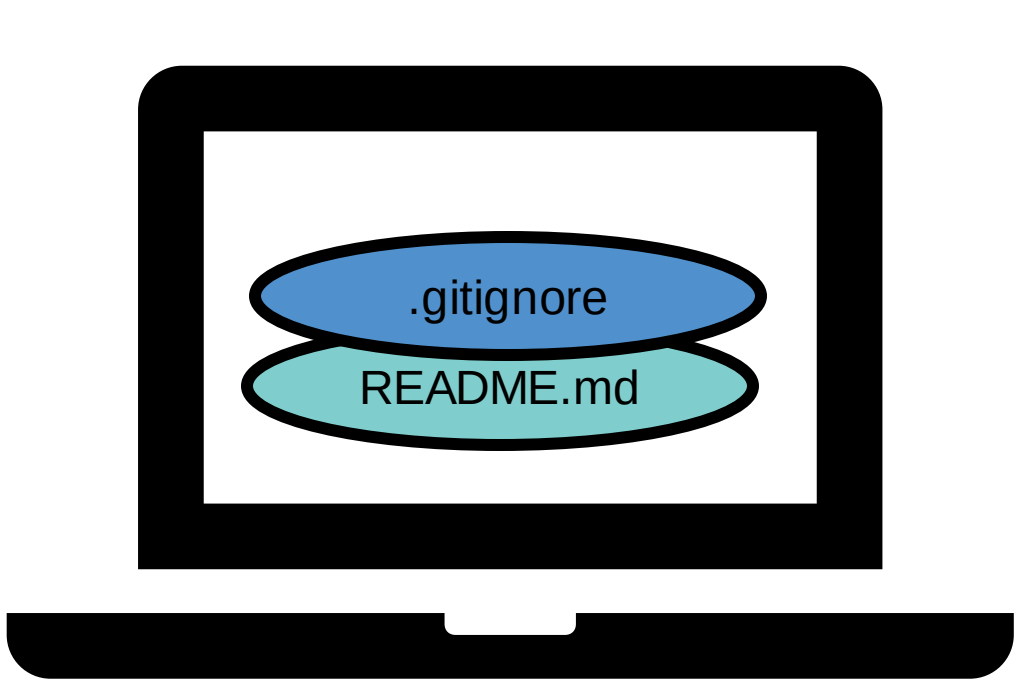
Follow the prompts

- Paste in the token when asked
- If you had existing credentials (including username/password which are no longer allowed), you may need to choose option 2: Replace these credentials
- git saves these credentials for you

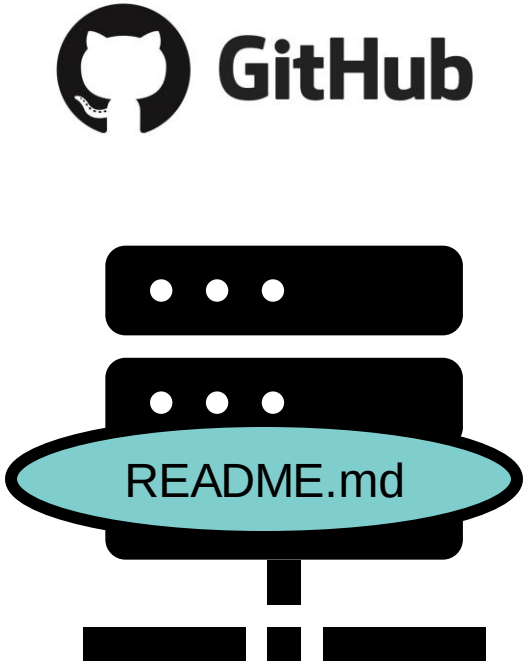


Push changes

Local changes not backed up

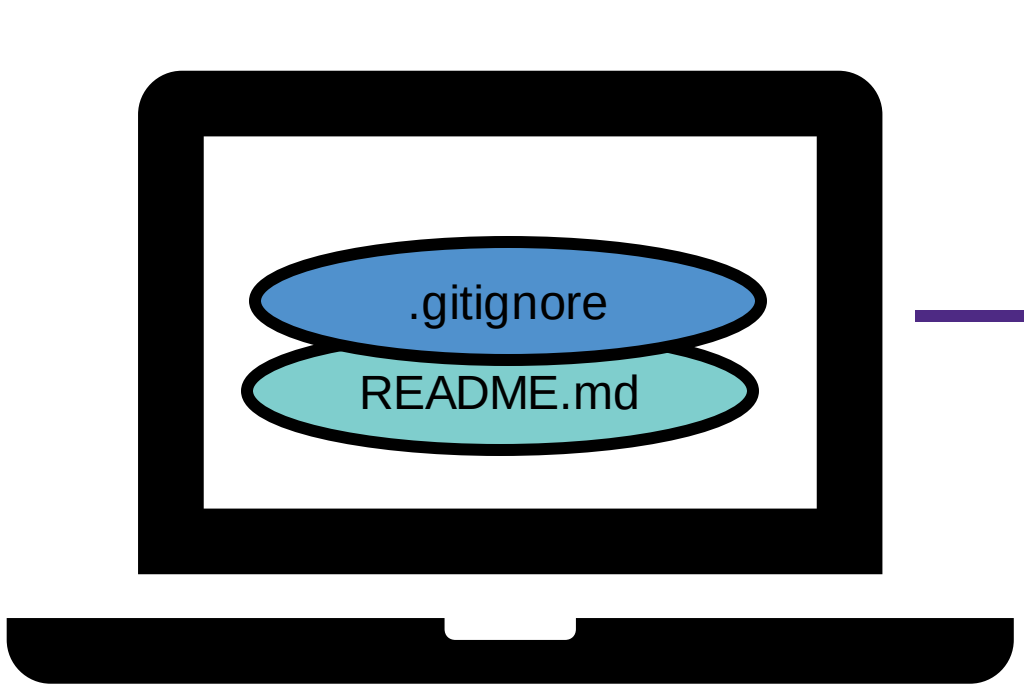


Local



Remote

Push to remote

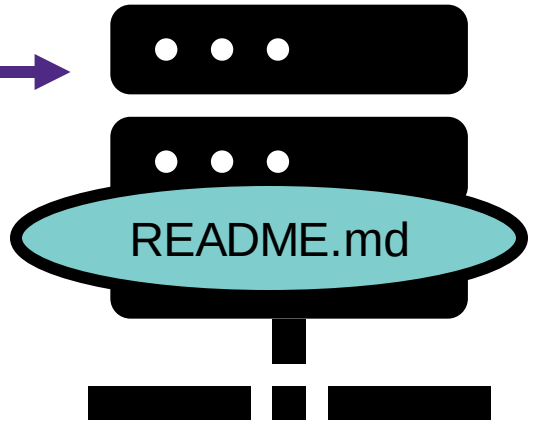


Local

Push

- Use connection established by clone
- Upload changes from local to remote repository

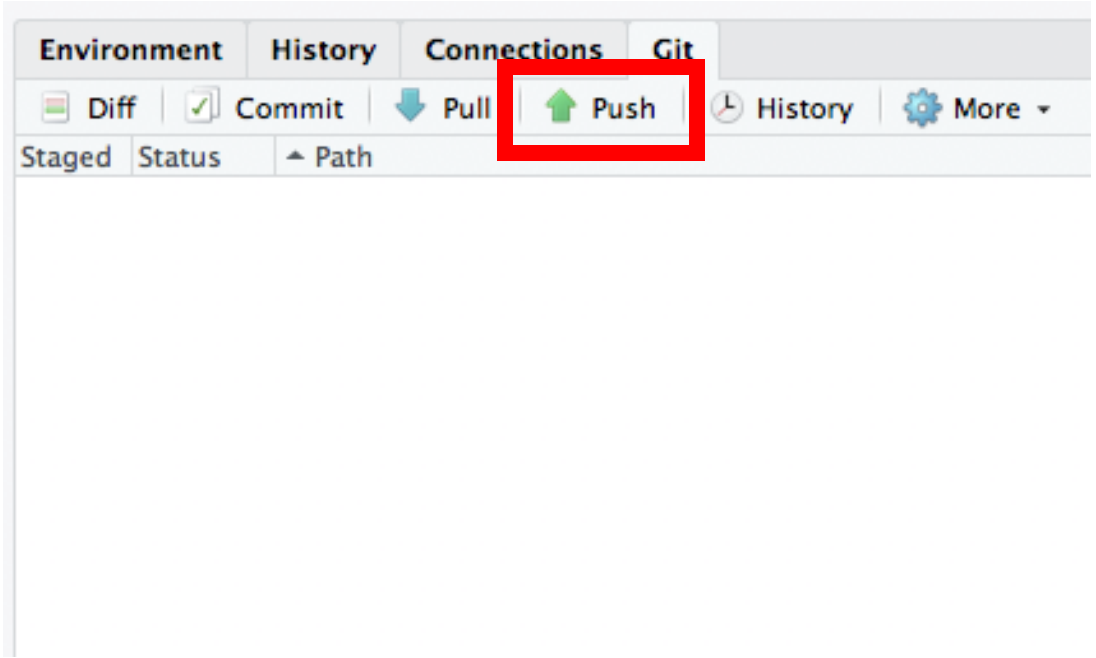
GitHub



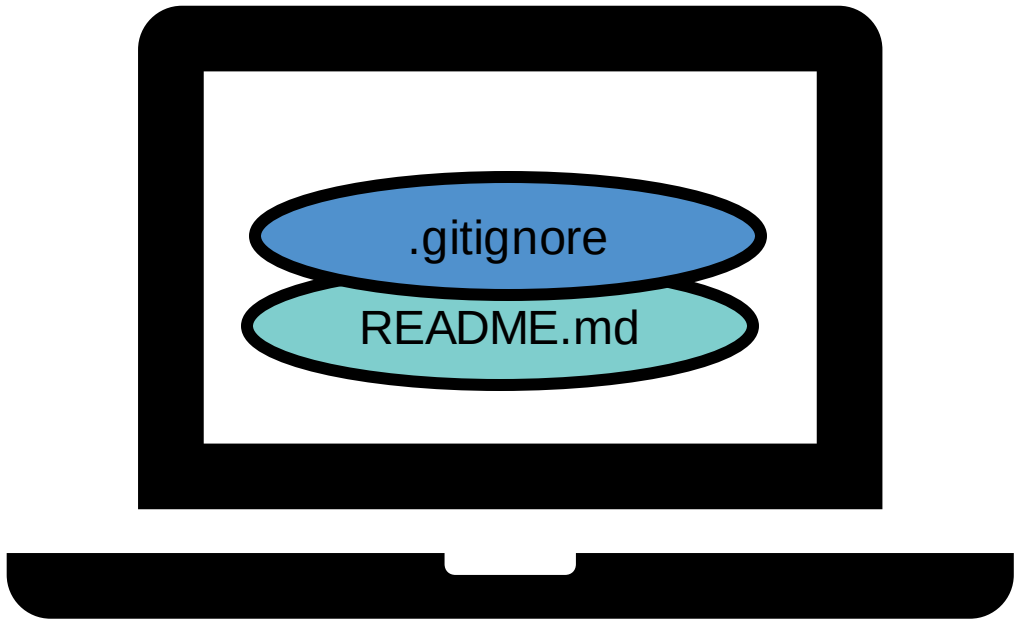
Remote

Push

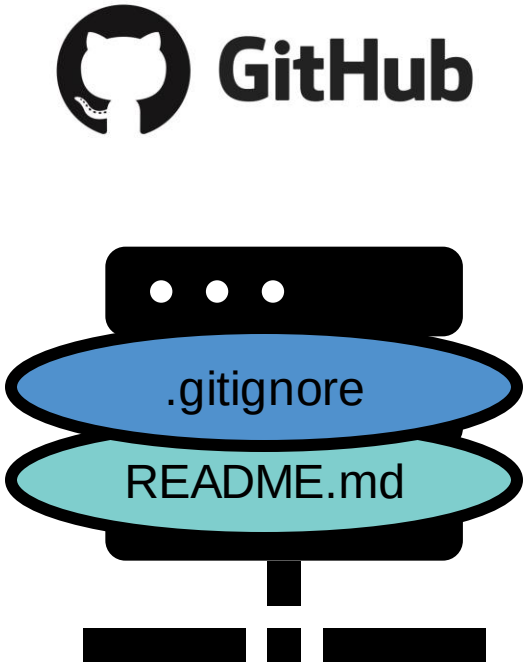
- Save local changes to remote repository
- Changes must be committed before pushing
- Requires that you have authentication set up on your machine



Changes pushed

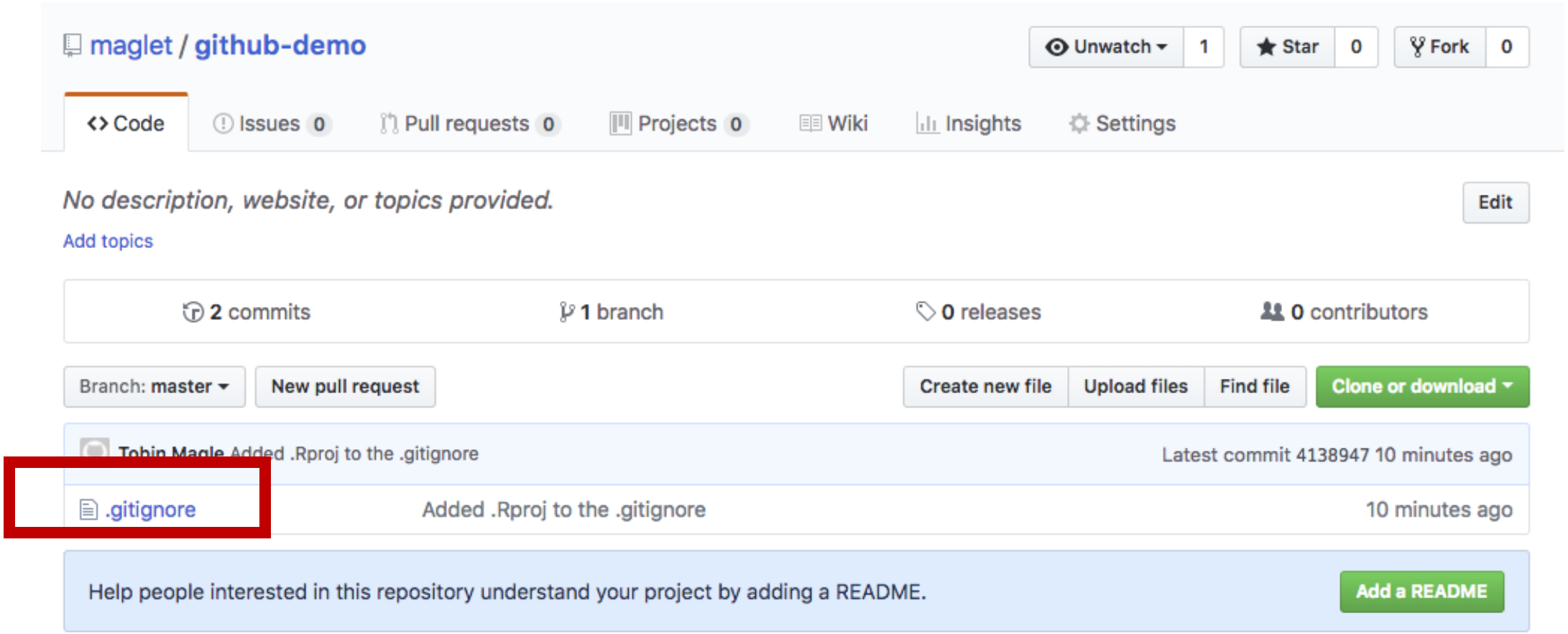


Local



Remote

Files in remote



Practice:

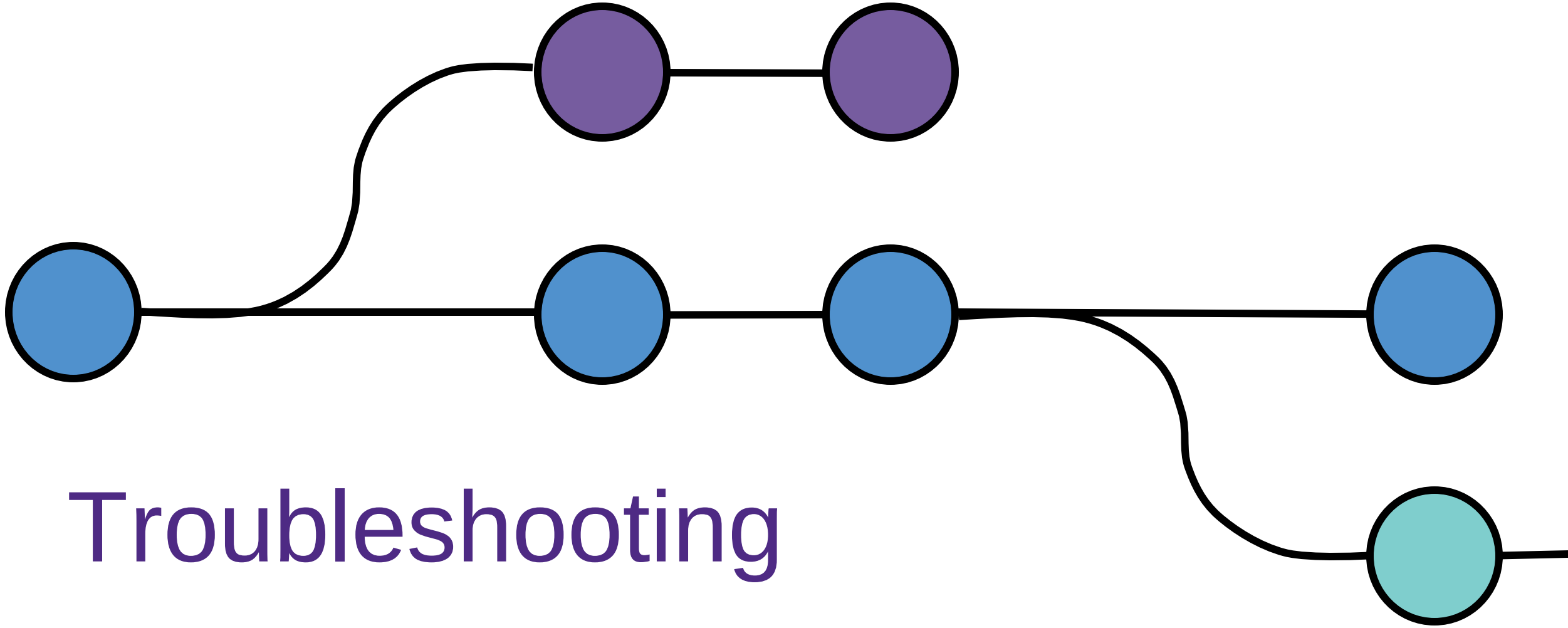
- Make a new R script
- Save it to the R project directory
- Add some R code
- Do the steps necessary to get this file into your GitHub repo

Need help?

Workshop materials: https://github.com/nuitrcs/git-RStudio

- Request a consult: bit.ly/rcdsconsult

- Other Resources
  - RCDS: Written Setup Instructions
  - RCDS: Other ways to set up Git repos in R
  - What git is and how it works: Git for Humans
  - Why you should use git: Academic benefits of using git
  - How to use git with R and RStudio: Happy git with R



Troubleshooting

Q: I don’t see the git tab
A: Check that RStudio sees git

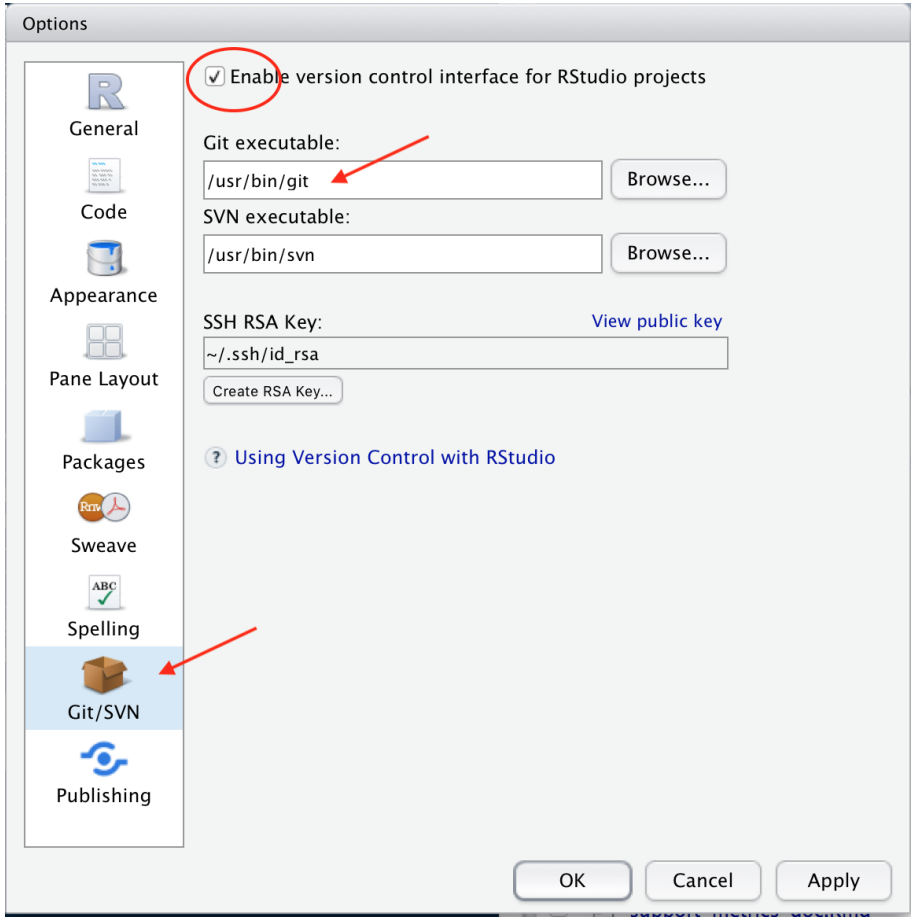
Mac: RStudio > Preferences

Windows: Tools > Global Options

Mac: type

which git

in the Terminal to get the path



RCS Help: bit.ly/rcs_data Git with R Setup Instructions: bit.ly/rcs_gitr