Setting up a Project in R

mjiwa

15 August, 2022

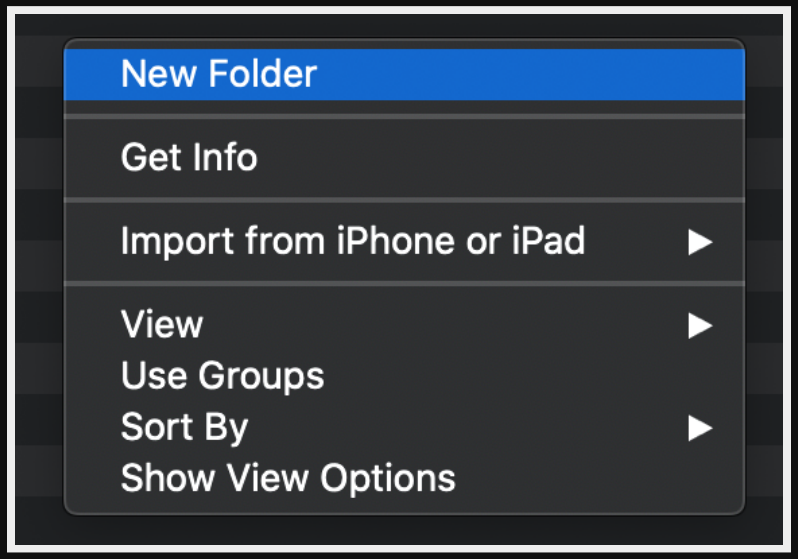
Goals

- Develop a workflow in R that is useful when working on several projects

- Stop doing this:

```
1  setwd("/files/that/only/exist/here")
2  rm(list = ls())
```

First thing's first, let's make a folder for our new project…



Now let's head to RStudio and start a new project inside that folder (File > New Project… > Existing Directory).

What is a project and why bother doing this?

Projects help you to manage your workflow in RStudio.

When you open a project (.Rproj file), RStudio will do a number of things…

- Change your working directory to the location of the Rproj file
- Restore your previously active tabs from the last time you worked on this project
- Load your command history from previous times you worked on this project

This means that, when working on multiple projects, we can segment our workflow so that we aren't constantly opening/closing files that aren't relevant to our current task.

But can't I just clear my workspace and switch tabs when switching to a different task?

Yes!

But this isn't particularly good practice.

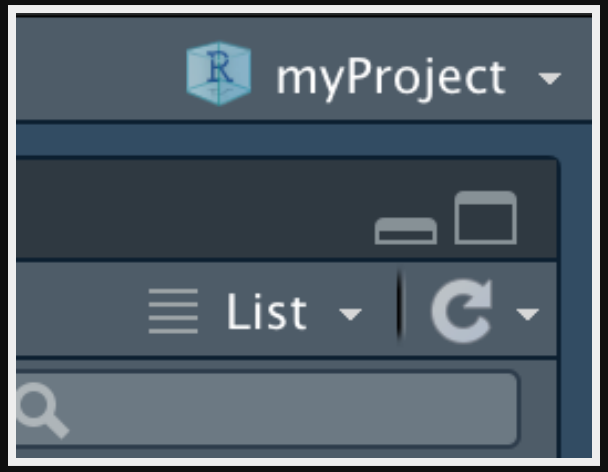
Clicking the broom icon or using `rm(list = ls())` when swapping between tasks doesn't mean your R session has been entirely reset. While the variables have gone, some other aspects may remain:

- Your current working directory
- The packages you have loaded
- Changes made to settings such as `stringsAsFactors = FALSE`, etc.

This will mean that next week, once you've restarted R, things can fail to run without a clear explanation (e.g., a script doesn't load a package it relies on, because you wrote it after loading that package elsewhere).

Working with projects

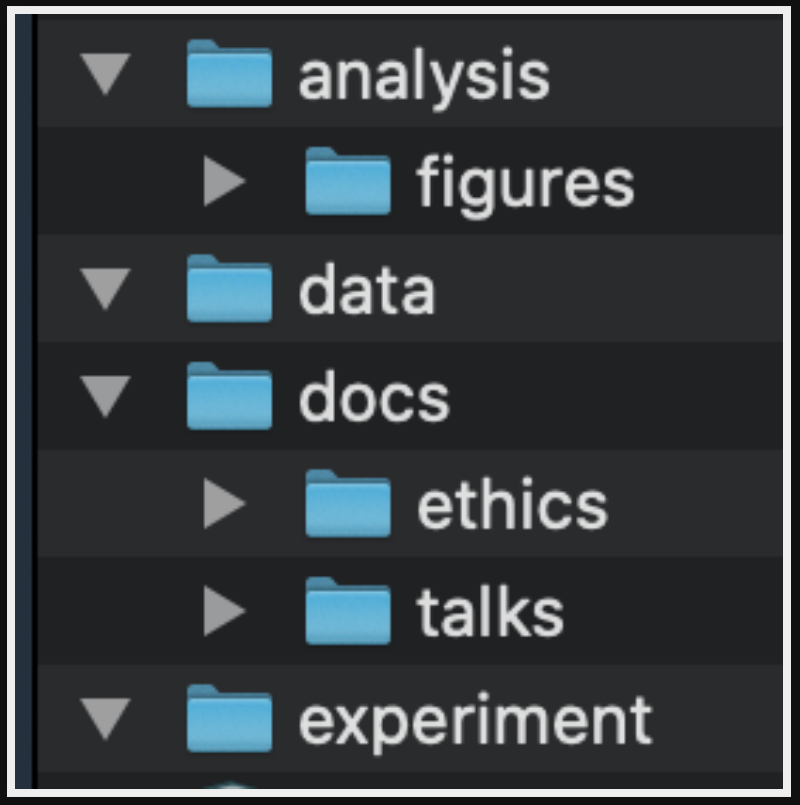
Your current project will be shown in the name of the window and in the top right-hand corner of RStudio:



Clicking on the project name will allow you to close the current project, or open a new one (in a new RStudio window, if you choose to).

Structuring the project

The file structure within your project will depend on the specific needs of the project, but most can use a skeleton that looks something like this:



File structure

- Your `experiment` folder will contain your experiment code.
- Your experiment will generate data which will be saved into the `data` folder. Alternatively, you might be importing your data into this folder from an online source.
- Your `analysis` folder will contain the scripts to load in your data and run analyses. These scripts may generate output (e.g., figures) that we can save in relevant sub-directories.
- Finally, your `docs` folder will contain all the documents relevant for your project. This could include ethics documents, PowerPoint files, relevant readings, etc.

Managing your project using Git (optional)

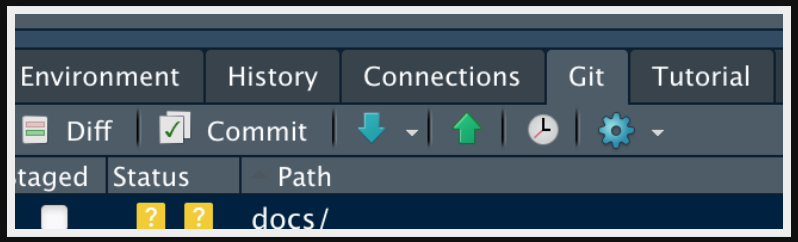
There are lots of reasons to use Git to manage your project. In short, Git offers you the ability to create and share a back-up of your file in a repository that can then be shared with supervisors and collaborators and/or accessed from different computers you may be using (e.g., acquisition computer and analysis computer).

Here is a list of potential benefits of doing this:

- Creates a back-up of your files in case you lose/damage your laptop.
- Maintains a changelog of your files so you can revert back to previous versions if something goes wrong (i.e., you no longer need to have 4 different versions of the same document named ~~~~V2_2.docx, etc.)
- Allows you to maintain consistent versions of files across multiple computers (e.g., work desktop & personal laptop).
- Enables you to work collaboratively on a project with others.

How to implement Git in your project

You can use the `usethis` package to implement Git quickly and easily in R. You can initialise a Git repository in your current working directory by running `usethis::use_git()`. After doing so (and restarting RStudio), you will have a Git tab in your RStudio environment:



From this tab, you can commit, push, and pull changes to your repository. If you don't know what that means, you may need to go through a short intro to Git course (here's one of those).

Connecting to GitHub

To connect your repository to Github, you can again use the `usethis` package, specifically by running `usethis::use_github(private = TRUE)`. You may need to complete additional steps to connect your RStudio to your Github account.

Extra point to be aware of

Any repository you create for a project involving data **should be set to private** and should be disclosed in your ethics documentation. You can make sure your repository is private by using `use_github(private = TRUE)`.

Take-away points

- Segmenting your R workflow into projects will make for a more manageable experience.
- Git is a great way of backing up and sharing your projects, and is easy to integrate with RStudio.