# RProjects

### Creating a project-oriented workflow in R

Daniela Palleschi

Thu Aug 22, 2024

## Table of contents



## Learning Objectives

Today we will…

- learn about project-oriented workflows
- create an RProject
- use project-relative filepaths with the `here` package

## Installation requirements

- required installations/recent versions of:
  - R
    - version `4.4.0`, "Puppy Cup"
    - check current version with `R.version`
    - download/update: https://cran.r-project.org/bin/macosx/
  - RStudio
    - version `2023.12.1.402`, "Ocean Storm"
    - Help > Check for updates
    - new install: https://posit.co/download/rstudio-desktop/

## Project-oriented workflow

1. Folder structure:
   - keeping everything related to a project in one place
   - i.e., contained in a single folder, with subfolders as needed
2. Project-relative working directory
   - the project folder should act as your working directory
   - all file paths should be relative to this folder

### Folder structure

- a core computer literacy skill
  - keep your Desktop as empty as possible
  - have a sensible folder structure
  - avoid mixing subfolders and files
    - i.e., if a folder contains subfolders, ideally it should not contain files

## RProjects

- in data analysis, using an IDE is beneficial
  - e.g., RStudio
- most IDEs have their own implementation of a Project
- in RStudio, this is the RProject
  - creates a `.Rproj` file in a project folder
  - stores project settings
- you can have several RProjects open simultaneously
  - and run several scripts across projects simultaneously
- most importantly, RProjects (can) centralise a specific project's workflow and file path
- to read more about R Projects, check out Section 6.2: Projects from Wickham et al. (2023; or Ch. 8 - Workflow: Projects in Wickham & Grolemund, 2016)

### Creating a new Project

- when?
  - whenever you're starting a new course or project which will use R
- why?
  - to keep all the relavent materials in one place
- where?
  - somewhere that makes sense, e.g., a folder called `SoSe2024` or `Mastersarbeit`
- how?
  - `File > New Project > New Directory > New Project > [Directory name] > Create Project`

> New RProject
>
> Create a new RProject for this workshop
>
> - `File > New Project > New Directory > New Project > [Directory name] > Create Project`

- make sure you choose a sensible location

## Opening a Project

- to open a project, locate its `.Rproj` file and double-click
- or if you're already in RStudio, you can use the `Project (None)` drop-down (top right)

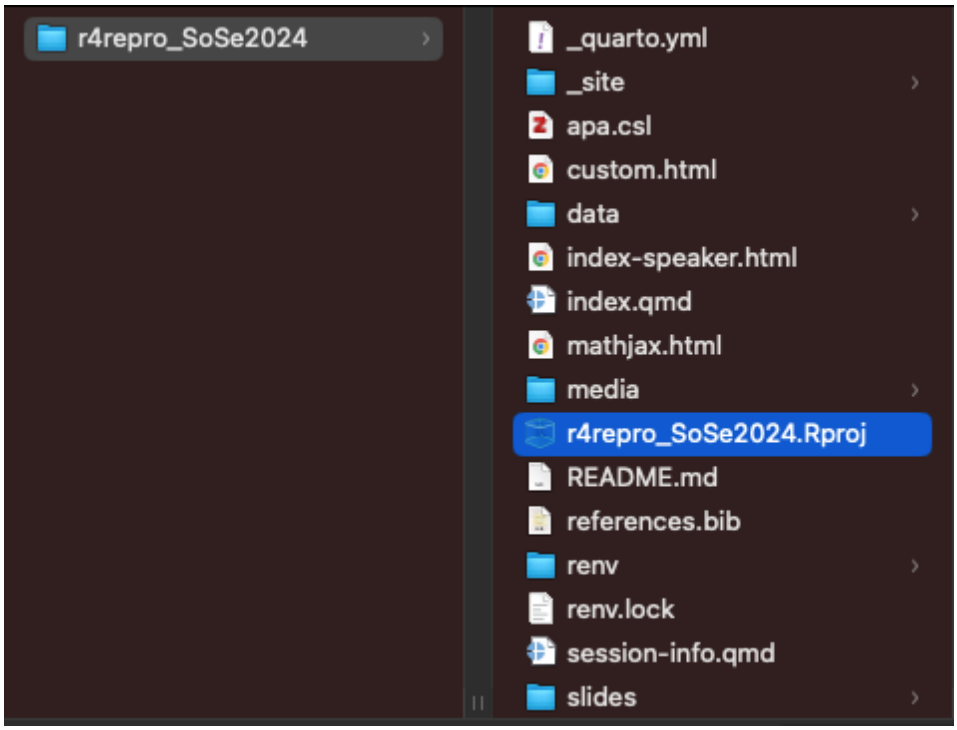



Figure 1: Double-click `.Rproj`

## Adding a README file

- `File > New File > Markdown File` (*not* R Markdown!)
  - add some text describing the purpose of this project
  - include your name, the date
  - use Markdown formatting (e.g., `#` for headings, `*italics*`, `**bold**`)
- save as `README.md` in your project directory

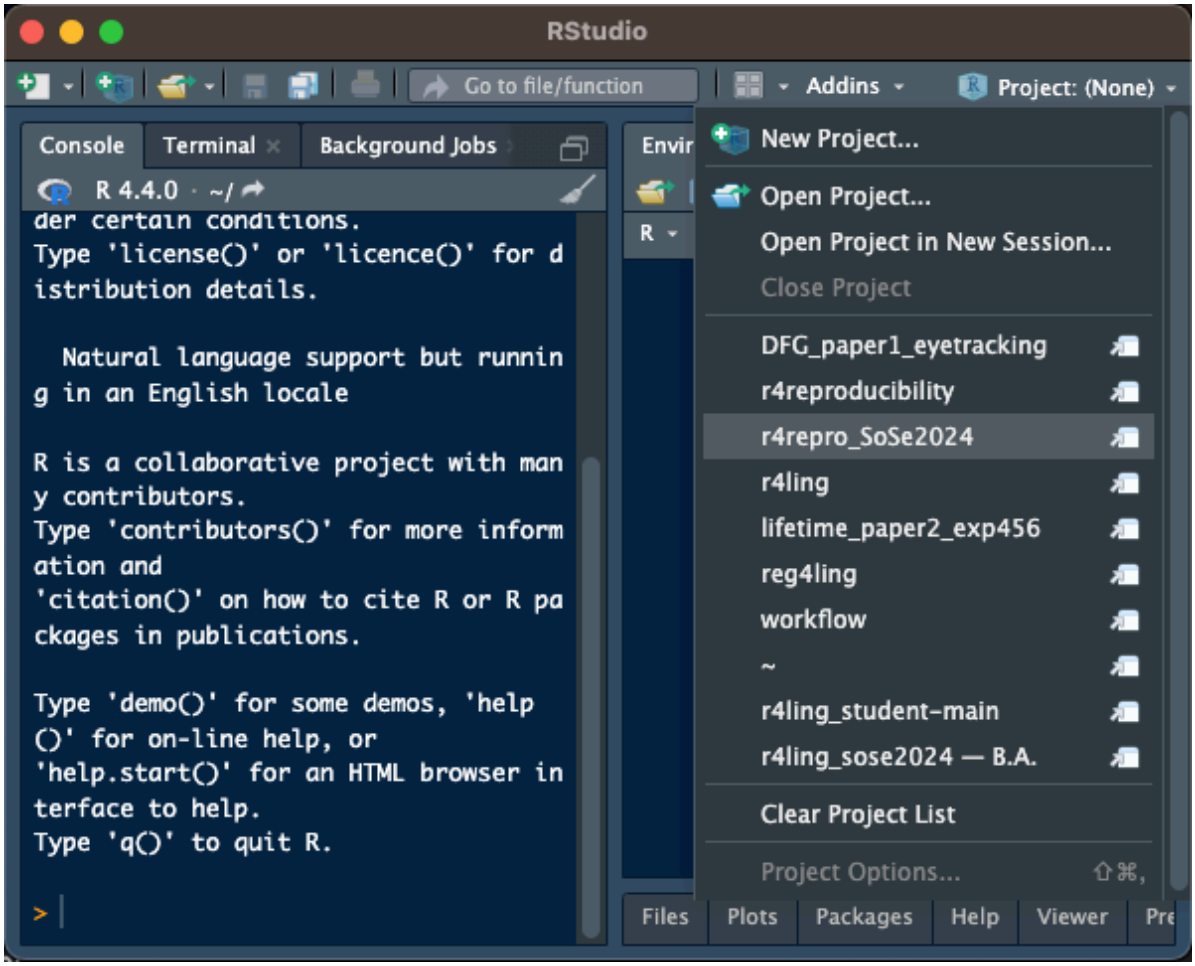



Figure 2: Open from RStudio

## Global RStudio options

- `Tools > Global Options`
  - **Workspace**: Restore .RData into workspace at startup: NO
  - Save workspace to .RData on exit: Never
- this will ensure that you are always starting with a clean slate
  - and that your code is not dependent on some pacakge or object you created in another session
- this is also how RMarkdown and Quarto scripts run
  - they start with an empty environment and run the script linearly

Global settings

Change your Global Options so that

- **Workspace**: Restore .RData into workspace at startup: NO
- Save workspace to .RData on exit: Never

## Identifying your RProject

- there are a ways to check which (if any) RProject you're in
  - there are 6 differences between xyzfig-noproject and xyzfig-project
  - which is in an RProject session?

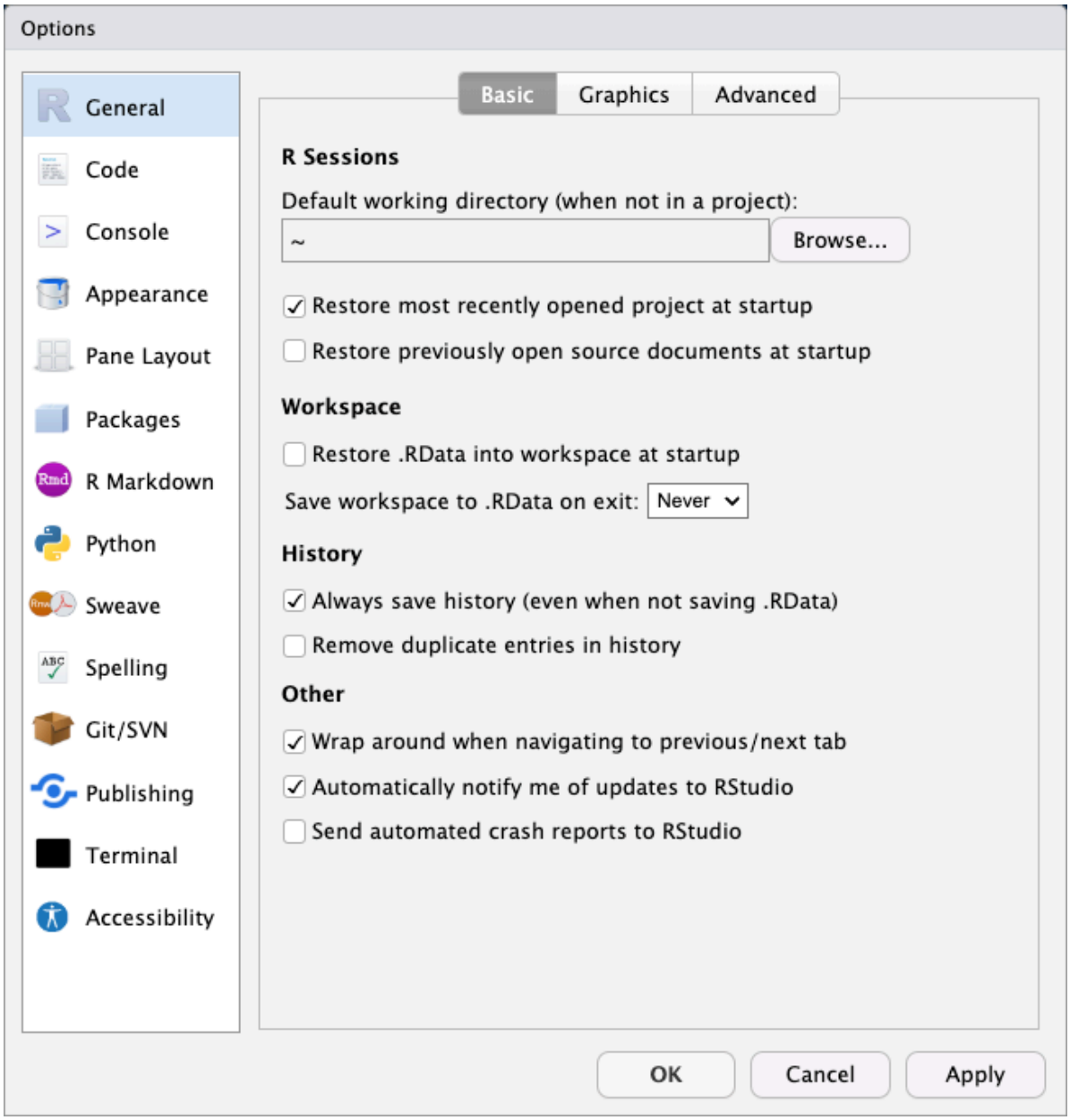



Figure 3: RStudio settings for reproducibility

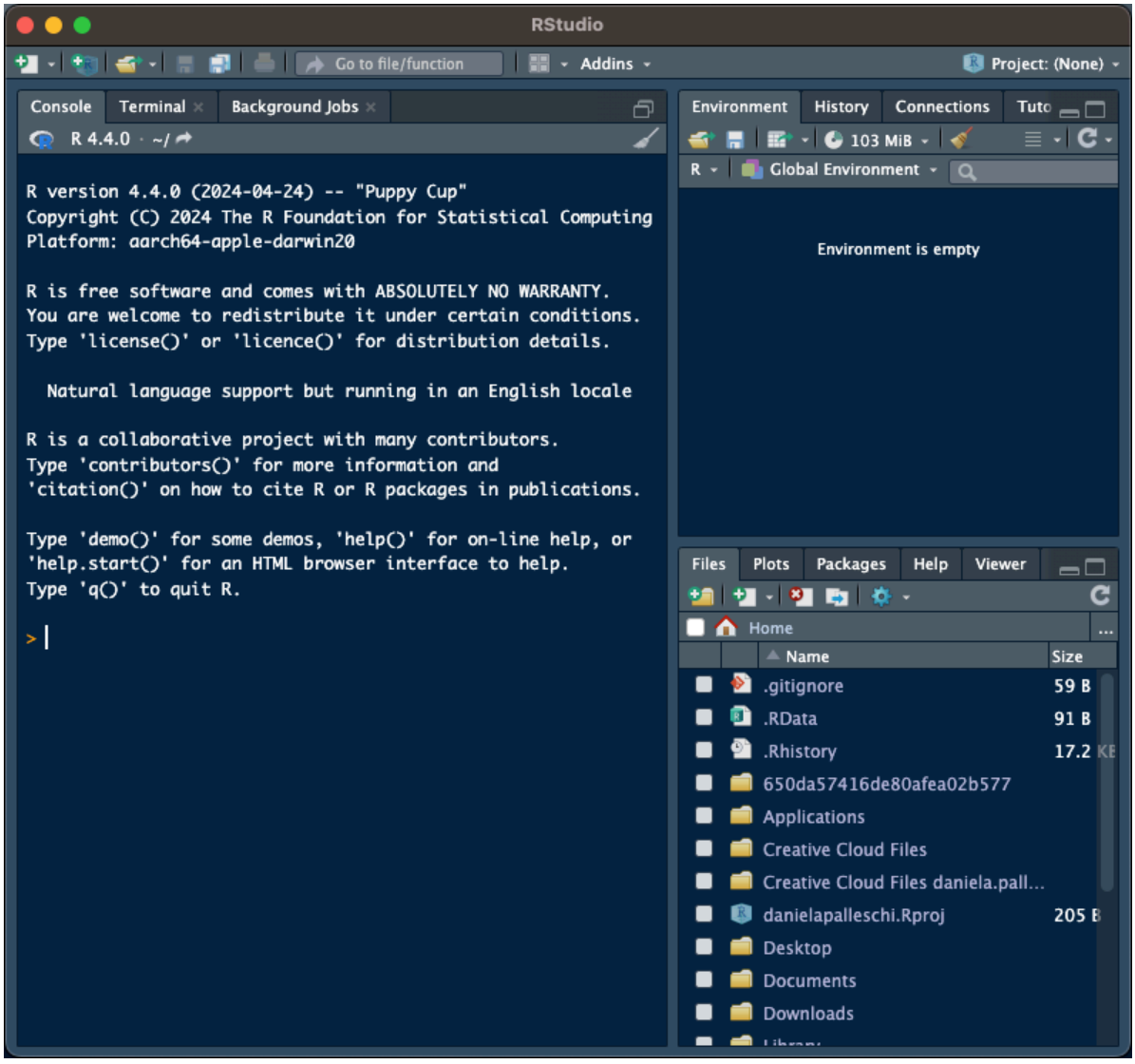



Figure 4: RStudio Session A

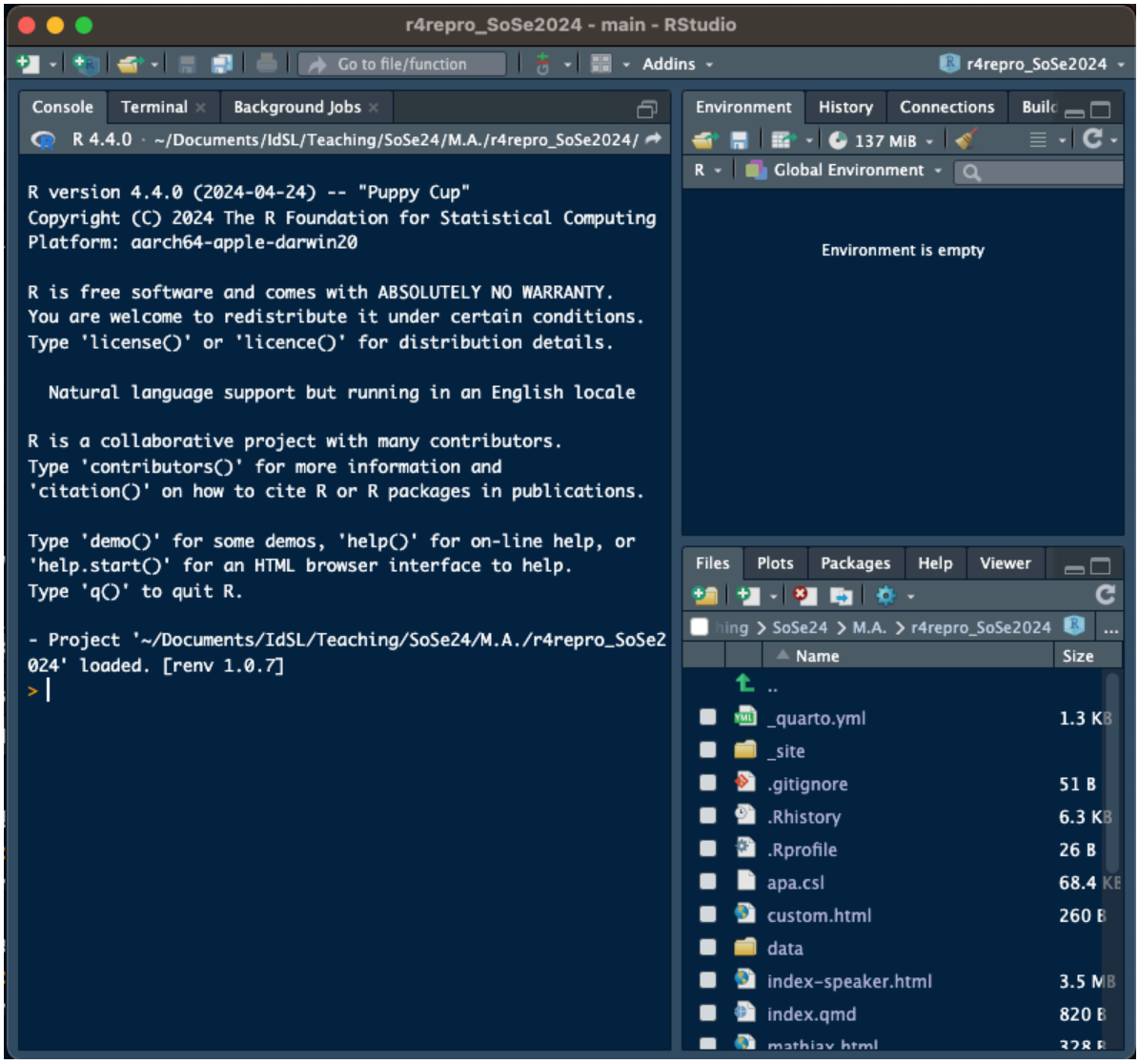



Figure 5: RStudio Session B

### Spot the differences

### Show the differences

Figure 1: RStudio Session A

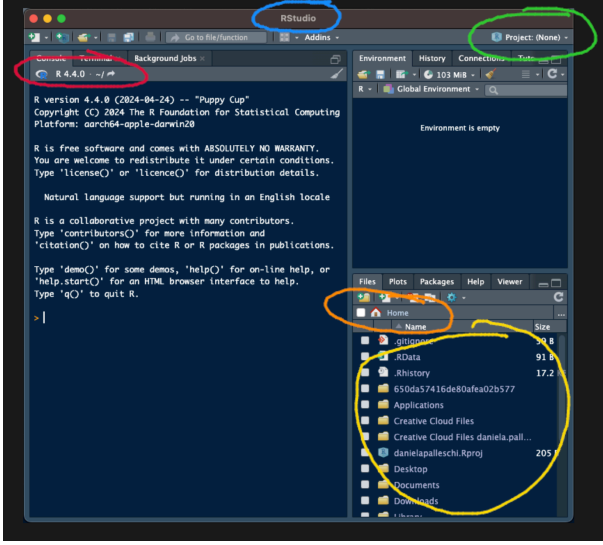
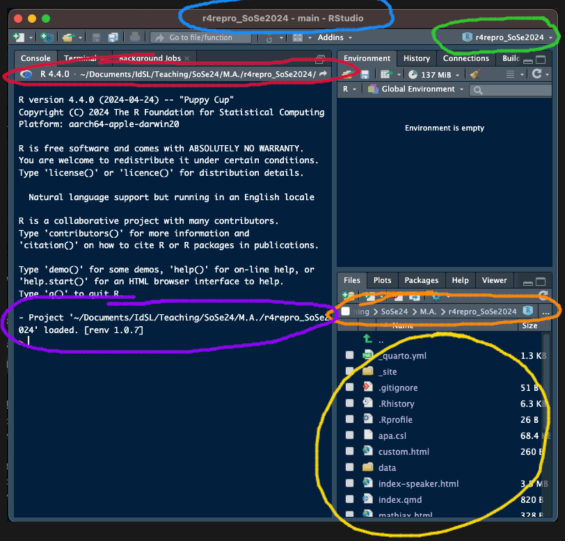



Figure 2: RStudio Session B




## Folder structure

- some folders you'll typically want to have:
  - `data`: containing your dataset(s)
  - `scripts` (or `analyses`, etc.): containing any analysis scripts
  - `manuscript`: containing any write-ups of your results
  - `materials`: containing relevant experiment materials (e.g., stimuli)
- let's just create the first 2 (`data` and `scripts`)

`data/`

- do you have "raw", i.e., pre-processed data?
  - if so, you might want to create a `raw` sub-folder
  - and any other relevant sub-folders (e.g., `processed` or `tidy`)
- download the dataset from the workshop repo (from Chromý et al., 2023)
  - *or*, move a dataset of your own to this folder

`scripts/`

- try to create a single script for each "product"
  - e.g., anonymised data, 'cleaned' data, data exploration, visualisation, analyses, etc.
- you can create sub-folders as the project develops and move scripts around
  - for now, let's create a new script to take a look at our data

New script

Create a new Quarto script:

1. `File > New File > Quarto Document`
2. Add a title
3. Uncheck the `Use Visual Editor` box
4. Click `Create`
5. Save it in your `scripts/` folder: `File > Save as...`

### Load in the data

- load in the data however you normally would
  - e.g., `readr::read_csv()`

## `here`-package

- `here` package (Müller, 2020) enables file referencing
  - avoids the use of `setwd()`

### The problem with `setwd()`

If the first line of your R script is

```
setwd("C:\Users\jenny\path\that\only\I\have")
```

I will come into your office and SET YOUR COMPUTER ON FIRE .


Figure 6: Illustration by Allison Horst

— Jenny Bryan

- `setwd()` depends on your entire machine's folder structure
- `setwd()` breaks when you
  - send your project folder to a collaborator
  - make your analyses open
  - change the location of your project folder
- using slashes is also dependent on your operating system

### The benefit of `here()`

- uses the top-level directory of your project as the working directory
- can separate folder names with a comma

**`here`**

Load the dataset using `here`

1. Install `here` (e.g., `install.packages("here")`)
2. Load `here` at the beginning of your package
   - or use `here::` before calling a function
3. Use the `here()` function to load in your data
4. Inspect the dataset however you usually would (e.g., `summary()`, `names()`, etc.)
5. Save your script

### `here::here()`

- install package

**Listing 1** `In the Console`

```
install.packages("here")
```

- load package and call the `here` function

```r
# load package
library(here)

# read in data
df_data <- read.csv(here("data", "data_lifetime_pilot.csv"))
```

- or directly call the `here` function without loading the package

```r
# read in data without loading here
df_data <- read.csv(here::here("data", "data_lifetime_pilot.csv"))
```

- note that I stored the data with the prefix `df_`
  - `df` stands for dataframe
- I recommend using object-type defining prefixes for all objects in your Environment
  - e.g., `fit_` for models, `fig_` for figures, `sum_` for summaries, `tbl_` for tables, etc.

Reproduce your analysis

1. Perform some data exploration (e.g., with `names()`, `summary()`, `dplyr::glimpse()`, whatever you typically do)
2. Save your script, then close RStudio/your Rproject.
3. Re-open the project. Can you re-run the script?

## Learning objectives

Today we learned...

- learn about project-oriented workflows
- create an RProject
- establish a self-contained project environment with `here`

## References


Bryan, J., & TAs, T. S. 545. (n.d.). R Basics and workflows. In *STAT 545 Course materials*. Retrieved May 6, 2024, from https://stat545.com/

Chromý, J., Brand, J., Laurinavichyute, A., & Lacina, R. (2023). Number agreement attraction in Czech and English comprehension: A direct experimental comparison. *Glossa Psycholinguistics*, *2*(1), 1–20. https://doi.org/10.5070/G6011235

Müller, K. (2020). *Here: A Simpler Way to Find Your Files* (Version 1.0.1). https://CRAN.R-project.org/package=here

*Using RStudio Projects*. (2024, April 16). Posit Support. https://support.posit.co/hc/en-us/articles/200526207-Using-RStudio-Projects

Wickham, H., Çetinkaya-Rundel, M., & Grolemund, G. (2023). *R for Data Science* (2nd ed.). https://r4ds.hadley.nz/

Wickham, H., & Grolemund, G. (2016). *R for data science: Import, tidy, transform, visualize, and model data*. " O'Reilly Media, Inc.".