R-Projects

Creating a project-oriented workflow in R

Daniela Palleschi

Tue Oct 8, 2024

Table of contents

Topics

- Project-oriented workflows

- creating an R-Project
- project-relative filepaths with the `here` package

Installation requirements

- required installations/recent versions of:
  - R
    - at least version `4.4.0`, "Puppy Cup"
    - check current version with `R.version`
    - download/update: https://cran.r-project.org/bin/macosx/
  - RStudio
    - at least 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

R-Projects

- 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 R-Project
  - creates a `.Rproj` file in a project folder
  - stores project settings
- you can have several R-Projects open simultaneously
  - and run several scripts across projects simultaneously
- most importantly, R-Projects (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 R-Project
>
> Create a new R-Project 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)

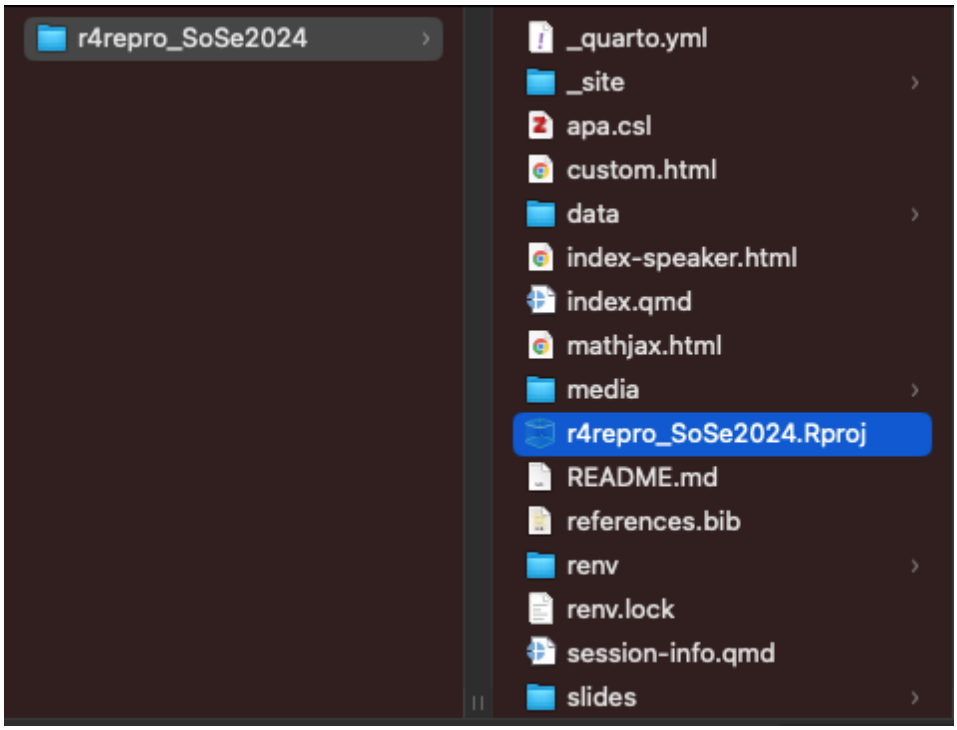

Figure 1: Double-click `.Rproj`

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

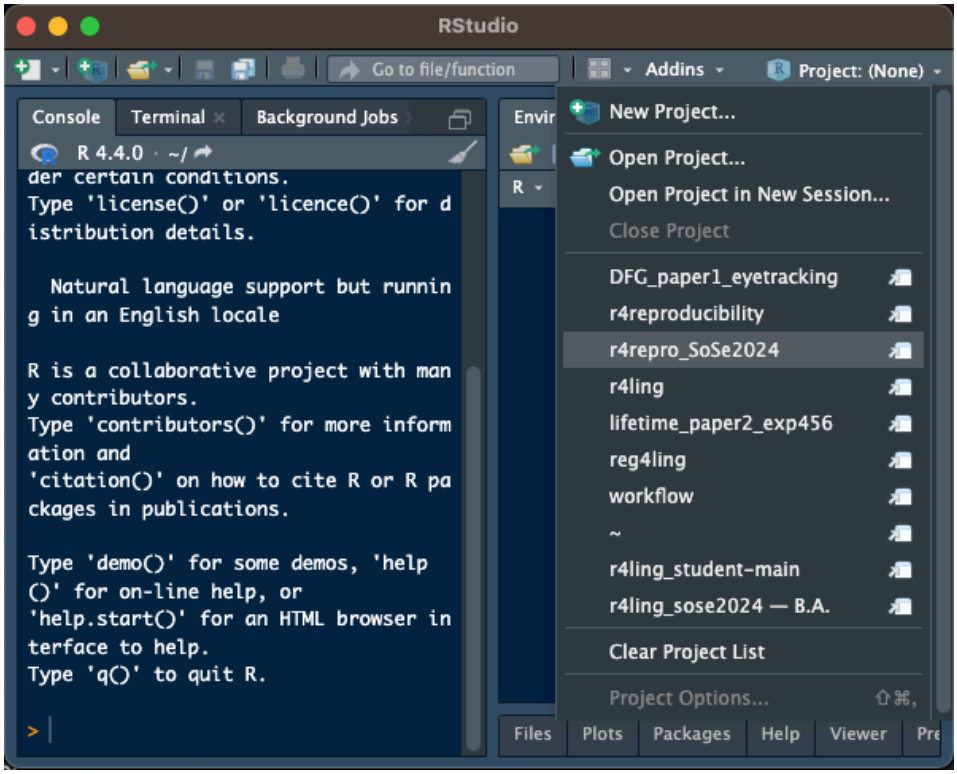

Figure 2: Open from RStudio

Global RStudio options

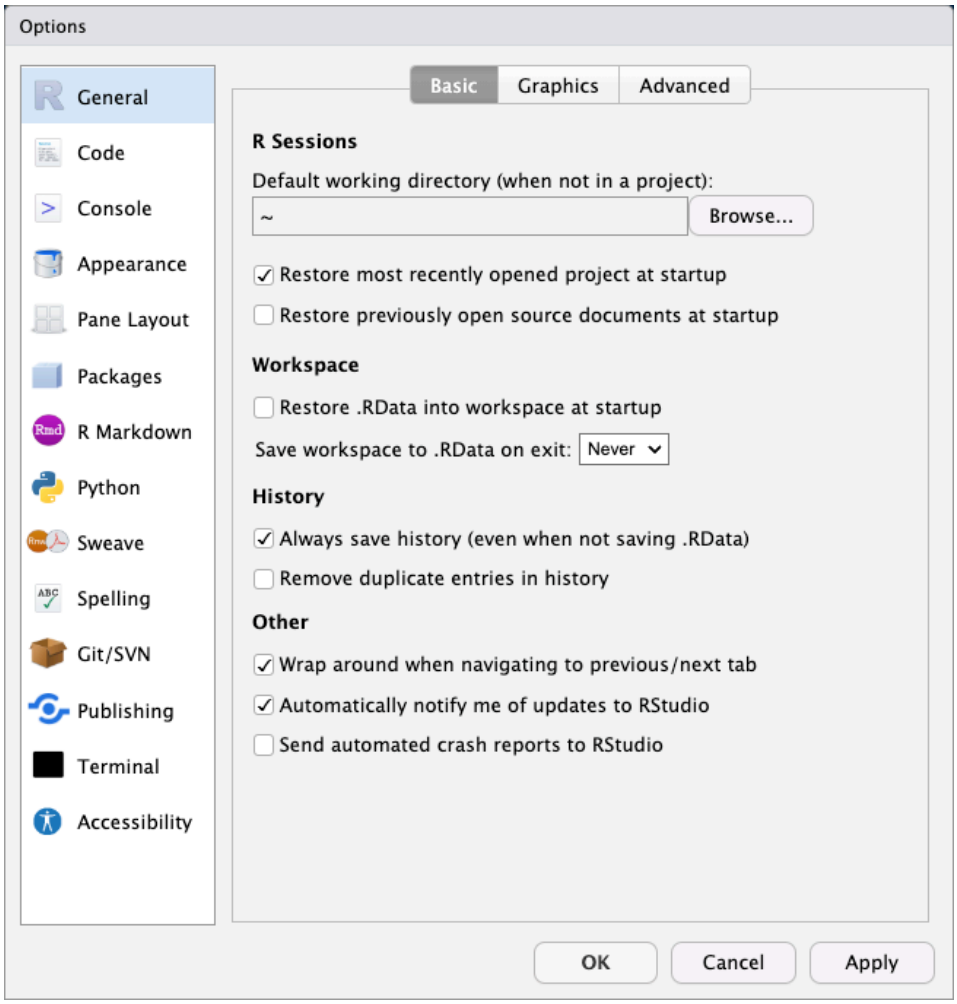

Figure 3: RStudio settings for reproducibility

- `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 R-Project

- there are a ways to check which (if any) R-Project you're in
  - there are 6 differences between Figure 4 and Figure 5
  - which is in an R-Project session?

Spot the differences

Show the differences

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 online_cleaned.csv dataset from the GitHub or OSF repo from Ćwiek et al. (2021)
  - *or*, move a dataset of your own to this folder

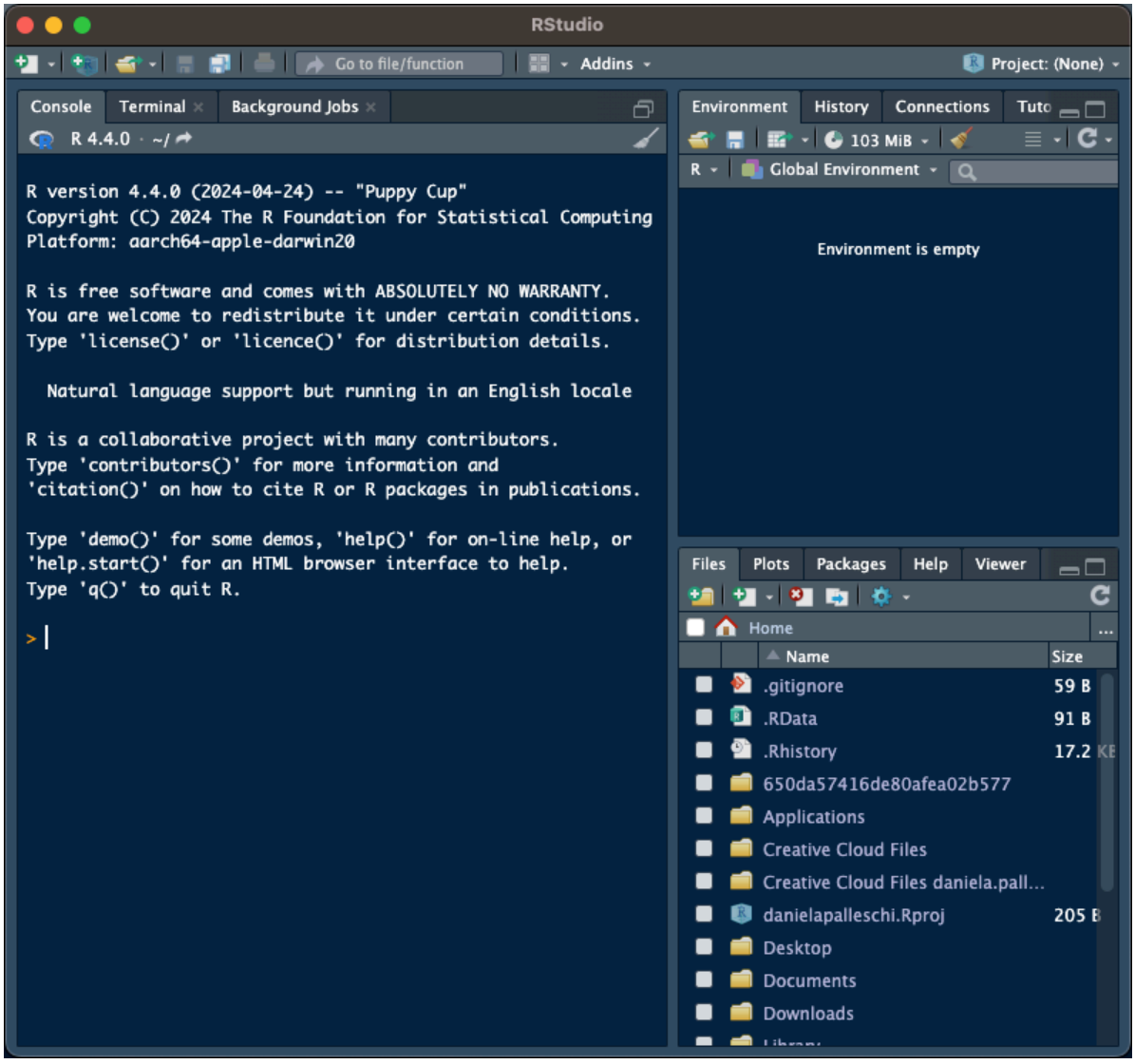

Figure 4: RStudio Session A

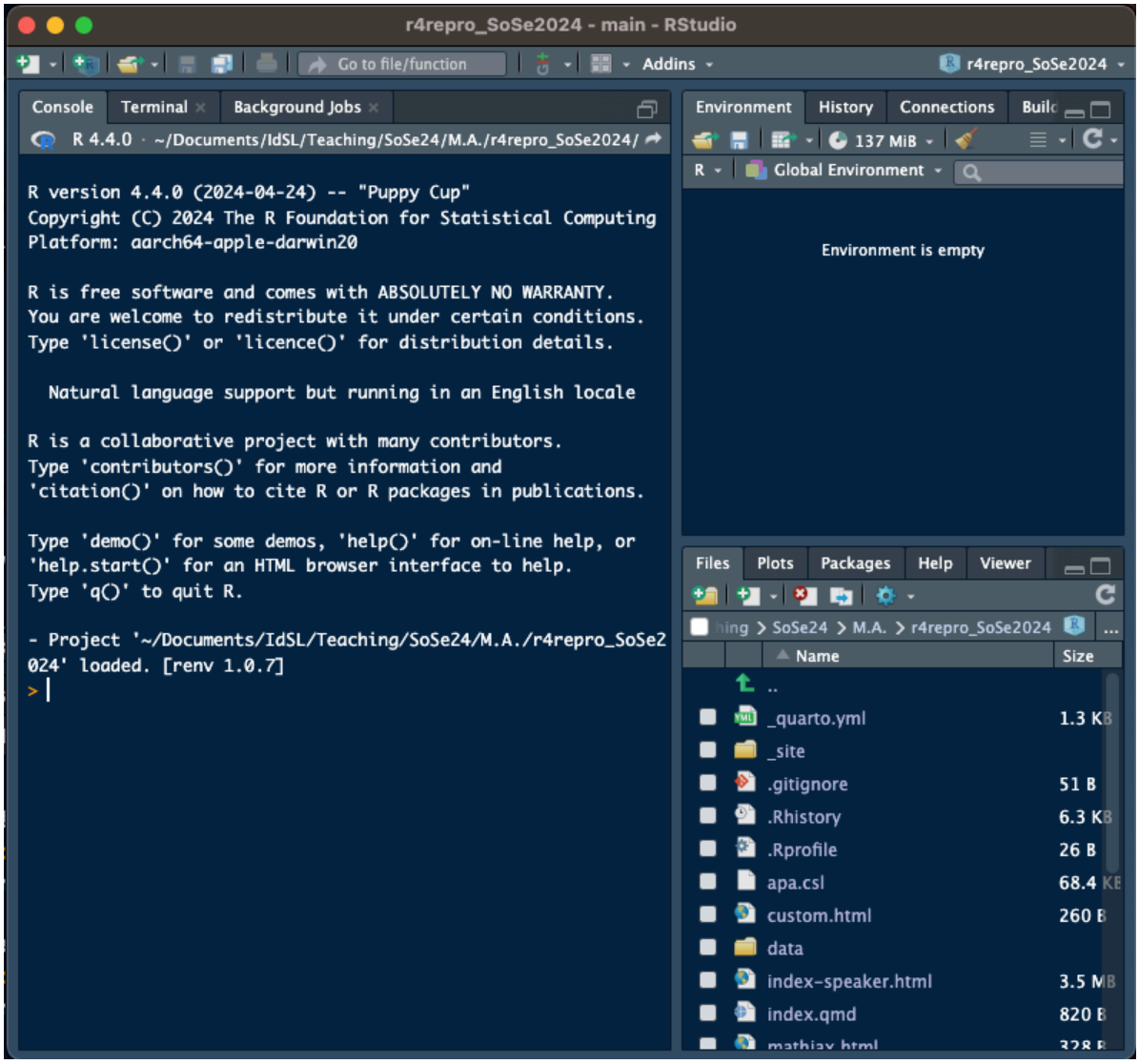

Figure 5: RStudio Session B

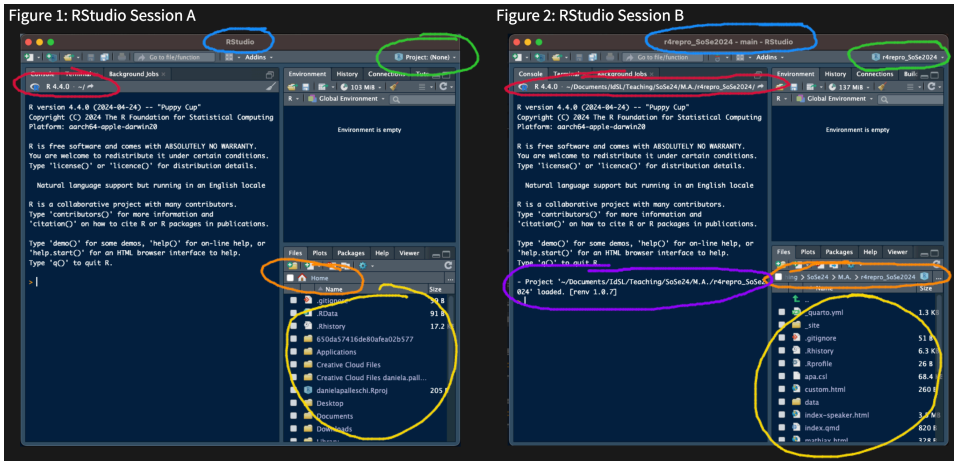

Figure 6: How to tell if you're in a project

- save the file as `cwiek_2021-online_cleaned.csv`
- description of data collection:

  In an online experiment with listeners of 25 different languages (from nine language families), participants listened to the 90 vocalizations (three for each of the 30 meanings), and for each, guessed its intended meaning from six written alternatives

  – Ćwiek et al. (2021)

- you could also download the data directly from GitHub in R:

```
write.csv(
  file = "data/cwiek_2021-online_cleaned.csv",
  read.csv("https://raw.githubusercontent.com/bodowinter/iconicity_challenge/refs/heads/maste
  )
```

`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 script:

1. `File > New File >` Choose your preferred script type
2. Save it in your `scripts/` folder: `File > Save as...`

Load in the data

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

Exercise: mini-Code Review

R-Project template

1. Download the R-Project template at https://osf.io/ctmwj/
2. Open (or switch to) `rproject-template.Rproj`
3. Inspect the folder structure and the files.
4. Look at the `scripts/` folder. Is it clear which scripts should be run first?
5. Try running `02-visualisation.R` first. Do you encounter any problems?

`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 .
>
> — Jenny Bryan

Figure 7: Illustration by Allison Horst

- `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
- trying to use somebody else's (or your former) folder path will result in a warning message like:

```
Error in setwd("/Users/danielapalleschi/Documents/R/rproject-template") :
cannot change working directory
```

The benefit of `here()`

- uses the top-level directory of your Project as the working directory
  - meaning we never need to specify the path to our project folder relative to our current higher-level folder structure
- can separate folder names with a comma
  - meaning it doesn't matter if the original code was written on a Mac or a Windows machine

`here`

In your R Project, load the `cwiek_2021-online_cleaned.csv` data using `here`

1. Install `here` (if needed; 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

Listing 1 `In the Console`

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

`here::here()`

- install package
- load package and call the `here` function

```
# load package
library(here)

# read in data
df_icon <- read.csv(here("data", "cwiek_2021-online_cleaned.csv"))
```

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

```
# read in data without loading here
df_icon <- read.csv(here::here("data", "cwiek_2021-online_cleaned.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 R-Project.
3. Re-open the project. Can you re-run the script?

Topics

- Project-oriented workflows
- creating an R-Project
- project-relative filepaths with the `here` package

References

Bryan, J., Hester, J., Pileggi, S., & Aja, D. E. (n.d.). *What They Forgot to Teach You About R*. Retrieved May 6, 2024, from https://rstats.wtf/

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/

Ćwiek, A., Fuchs, S., Draxler, C., Asu, E. L., Dediu, D., Hiovain, K., Kawahara, S., Koutalidis, S., Krifka, M., Lippus, P., Lupyan, G., Oh, G. E., Paul, J., Petrone, C., Ridouane, R., Reiter, S., Schümchen, N., Szalontai, Á., Ünal-Logacev, Ö., … Perlman, M. (2021). Novel vocalizations are understood across cultures. *Scientific Reports*, *11*(1), 10108. https://doi.org/10.1038/s41598-021-89445-4

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.".