Workflow

There's a famous blog post about workflows in R[1] about a talk Jenny Bryan gave that included this slide:

> 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 🔥.
>
> If the first line of your R script is
>
> ```
> rm(list = ls())
> ```
>
> I will come into your office and SET YOUR COMPUTER ON FIRE 🔥.

1. yes, R blog posts can be famous

The problem with `setwd()`

- `setwd()` changes the working directory, leading to potential issues in collaboration and reproducibility
  - You and I don't have the same file structure!
  - For example, my current working directory is

```
1  getwd()
```

- It's also really annoying to move around files and folders, even if it's just you using them

R Projects

- R projects provide a structured and organized way to work on projects in R
- R projects encapsulate all project-related files and settings into a single directory
- RStudio makes it easy to work with R projects

R Projects

```
my-project/
├─ my-project.Rproj
├─ README
├─ data/
│   ├── raw/
│   └── processed/
├─ R/
├─ results/
│   ├── tables/
│   ├── figures/
│   └── output/
└─ docs/
```

- An `.Rproj` file is mostly just a placeholder. It remembers various options, and makes it easy to open a new RStudio session that starts up in the correct working directory. You never need to edit it directly.
- A README file can just be a text file that includes notes for yourself or future users.
- I like to have a folder for raw data – which I never touch – and a folder(s) for datasets that I create along the way.

Benefits of R Projects

1. **Isolation**: Each project has its own workspace, separate from other projects
2. **Reproducibility**: Projects ensure that code and data are self-contained and portable
3. **Collaboration**: Projects facilitate collaboration by sharing the entire project directory

Creating an R Project

1. Open RStudio and go to **File > New Project**, or click on the projects button in the upper-right corner of RStudeio.
2. Choose a project location (New Directory, Version Control, Existing Directory).
3. Specify the project directory and create the project.
4. Choose the project type (e.g., regular project, Quarto website, Bookdown book)

The `{here}` Package

- The `{here}` package simplifies file path management within R projects.
- It provides a consistent and reliable way to reference files within the project directory.

Benefits of the `{here}` Package

1. **Simplicity** : Avoids the need for manual file path manipulation with `setwd()` and `paste()` functions
2. **Flexibility** : Works seamlessly even when scripts are run from different locations
3. **Portability** : Ensures that file paths work consistently across different systems

Using `{here}` in your project

1. Install the `{here}` package: `install.packages("here")`.
2. Load the `{here}` package in your R script: `library(here)`.
3. Use the `here()` function to reference files within your project directory.

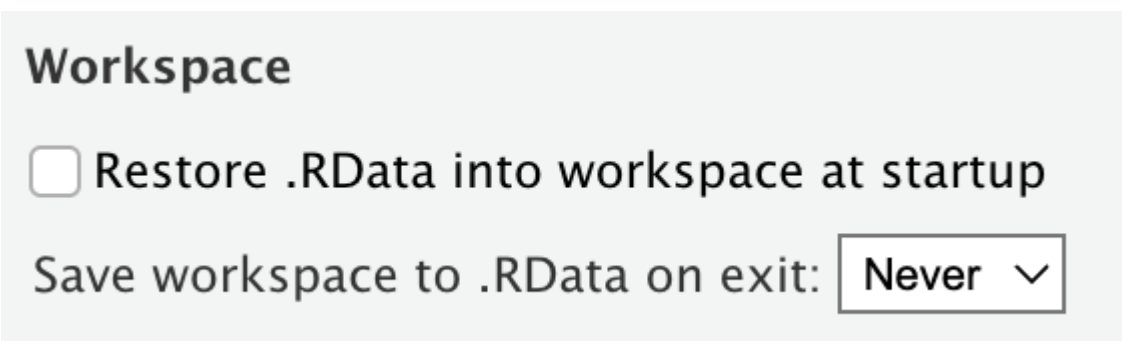
I actually like to use `here::here()` rather than loading the package with `library(here)` every time.

Using `{here}` in your project

- Construct file paths with reference to the top directory holding your `.Rproj` file.
- `here::here("data", "raw", "data.csv")` for me, here, becomes `"/Users/l.smith/Documents/Presentations/reproducible-epi-SER/data/raw/data.csv"`
- But if I send you my code to run, it will become whatever file path *you* need it to be, as long as you're running it within the R Project.

Start fresh

- `rm(list = ls())` removes objects from your environment, but it doesn't unload the packages
- You can easily create errors of the type `Error in... could not find function...` because your code runs just fine until you restart R and realize you never included the right `library()` calls
- Restart your R session early and often as you write code!



Conclusion

- Embrace the project-oriented workflow in R for better organization, reproducibility, and collaboration
- Use R projects to encapsulate project-related files and settings
- The `{here}` package helps with reliable and portable file path management

Additional Resources

- Jenny Bryan's blog post: Project-oriented workflow
- Another blog post by Malcolm Barrett
- `{here}` package documentation: