Overview

Data Wrangling, Session 1

Kieran Healy
Code Horizons

January 2026

Housekeeping

10:30am till 12:30pm US EST each day

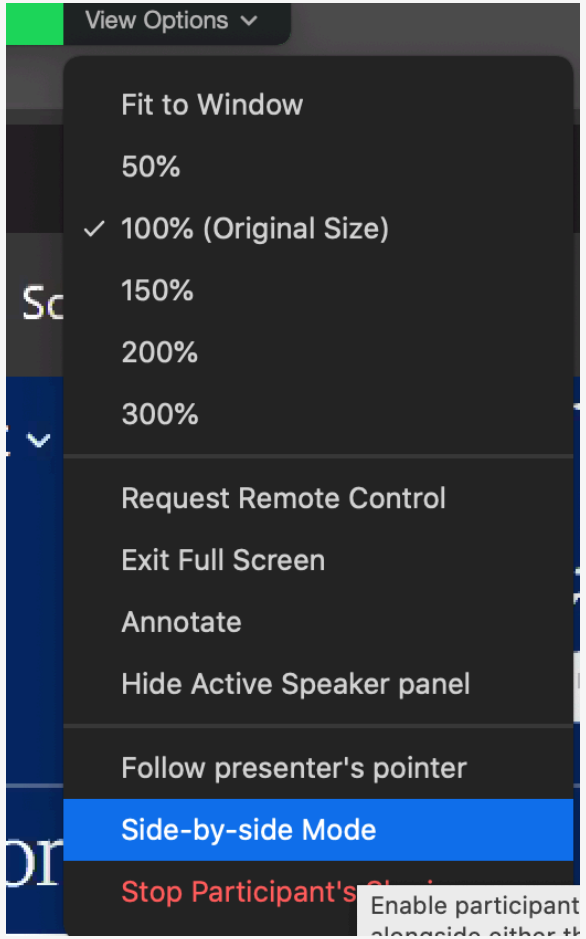
1:30pm to 3:30pm US EST each day

Use the Zoom chat to ask questions, or raise a hand with 👋

In between class sessions

slack

For a better Zoom experience



If you're watching in full-screen view and I'm sharing my screen, then from Zoom's "View options" menu *turn off* "Side-by-Side" mode.

My Setup and Yours

Talking, Slides, and Live-Coding in RStudio.

Follow along with RStudio yourself if you can.

The course packet is also an RStudio project and the place for your notes.

Goals for this first session

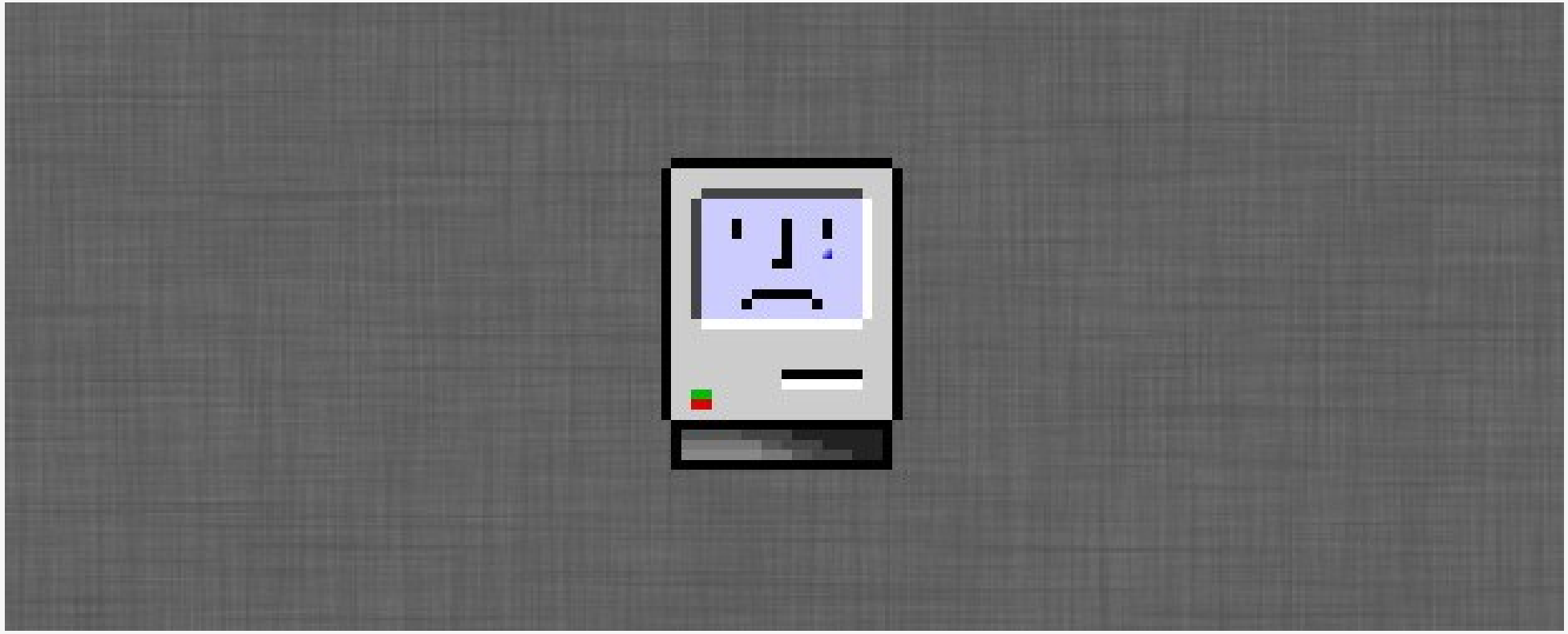
Some big-picture motivation & perspective.

Getting familiar with RStudio & its relationship to R.

Getting oriented to R and how it thinks.

DATA ANALYSIS
is mostly
DATA WRANGLING

Wrangling data is frustrating



Sad Mac

Can we make it fun?

No.

Fun data wrangling

Can we make it fun?

Fun data wrangling

No.

⇦ Not *this* much fun, at any rate

OK but can we eliminate frustration?

Frustration-free data wrangling

Also no.

OK but can we eliminate frustration?

Frustration-free data wrangling

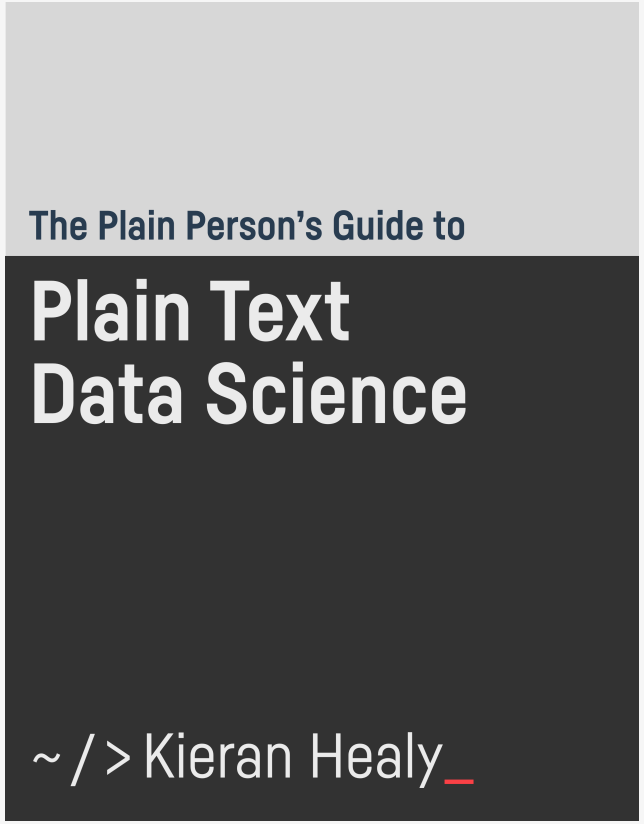
Also no.

(Sorry.)

However, we *can* make it work

Also, it's weirdly satisfying once you get into it.

We take a broadly *Plain Text* approach



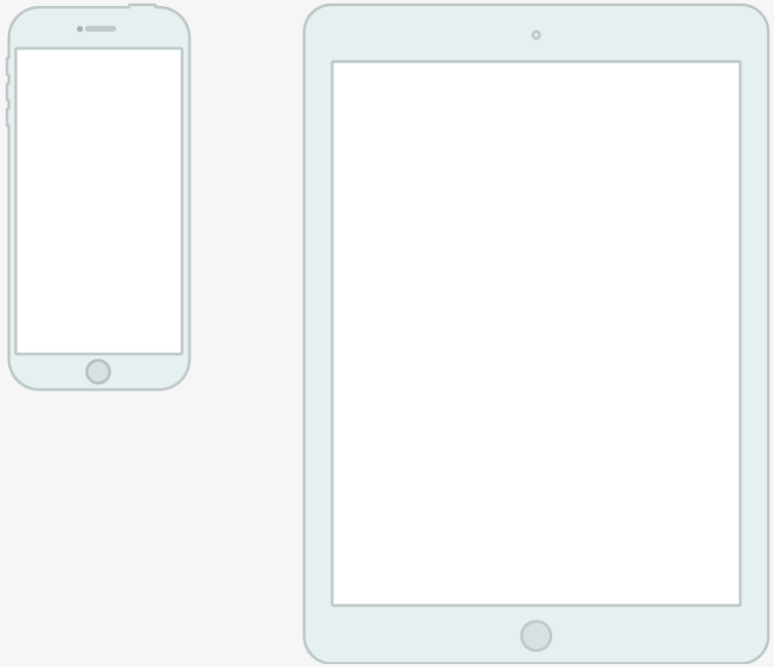
The plain person's guide

Using R and the Tidyverse can be understood within this broader context.

The same principles would apply to, e.g., using Python or similar tools.

Two revolutions in computing

Where the action is



iPhone and iPad

Touch-based user interface

Foregrounds a single application

Dislikes multi-tasking*

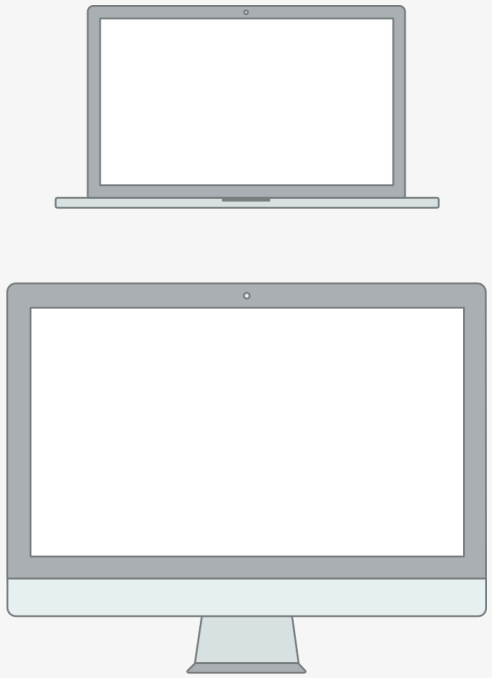
Hides the file system

“Laundry basket” model of where things are

*Multitasking

I mean, “Making different specialized applications and resources work together in the service of a single but multi-dimensional project”, not “Checking Twitter while also listening to a talk and waiting for an update from the school nurse.”

Where statistical computing lives



Desktop and laptop

Windows and pointers.

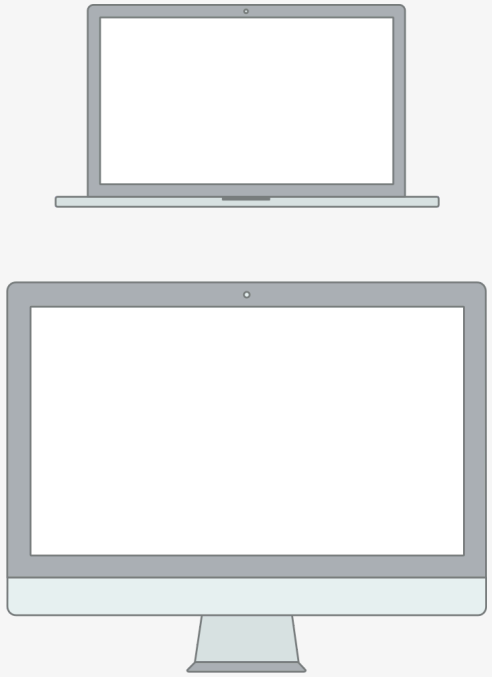
Multi-tasking, multiple windows.

Exposes and leverages the file system.

Many specialized tools in concert.

Underneath, it's the 1970s, UNIX, and the command-line.

Plain-Text Tools for Data Analysis



Desktop and laptop

Better than they've ever been!

Free! Open! Powerful!

Friendly community! Many resources!

But grounded in a UI paradigm that is increasingly far away from the everyday use of computing devices

So why do we use these tools?


CONTROL

Control, not Productivity

Productivity is great and everything, but not why we do all this.

The most important thing is to be able to *confidently know and clearly show what it was that you did* in the service of doing your work properly.

“Office” vs “Engineering” approaches

Questions

What is “real” in your project?

What is the final output?

How is it produced?

How are changes managed?

Different Answers

Office model

Formatted documents are real.

Intermediate outputs are cut and pasted into documents.

Changes are tracked inside files.

Final output is often in the same format you've been working in, e.g. a Word file, or perhaps a PDF.

Different Answers

Office model

Formatted documents are real.

Intermediate outputs are cut and pasted into documents.

Changes are tracked inside files.

Final output is often in the same format you've been working in, e.g. a Word file, or perhaps a PDF.

Engineering model

Plain-text files are real.

Intermediate outputs are produced via code, often inside documents.

Changes are tracked outside files.

Final outputs are assembled programatically and converted to a desired output format.

Different strengths and weaknesses

Office model

Everyone knows Word, Excel, or Google Docs.

“Track changes” is powerful and easy.

Hm, why can’t I remember how I made this figure?

Where did this table of results come from? Where did my file go?

`Paper_Final_edits_FINAL_kh-1a.docx`

Different strengths and weaknesses

Office model

Everyone knows Word, Excel, or Google Docs.

“Track changes” is powerful and easy.

Hm, why can’t I remember how I made this figure?

Where did this table of results come from? Where did my file go?

`Paper_Final_edits_FINAL_kh-1a.docx`

Engineering model

Plain text is universally portable.

Push button, recreate analysis.

Gaaah, Why can’t I make R do this simple thing?

This version control stuff is a pain.

```
Object of type 'closure' is
not subsettable
```

Each generates solutions to its problems

Office model

Make a suite of applications that know about each other.

Put everything on a cloud server, not your computer.

Just rely on search to find stuff in the pile.

Allow users to treat documents like plain-text (e.g. Markdown).

Put more advanced functions back in, somewhere.

Engineering model

Try to guess what the user wants; offer hints.

Continuously analyze code for errors.

Reintroduce GUI elements for e.g. Version Control.

Allow users treat their code more like a formatted document.

Have an LLM propose, edit, or directly write code.

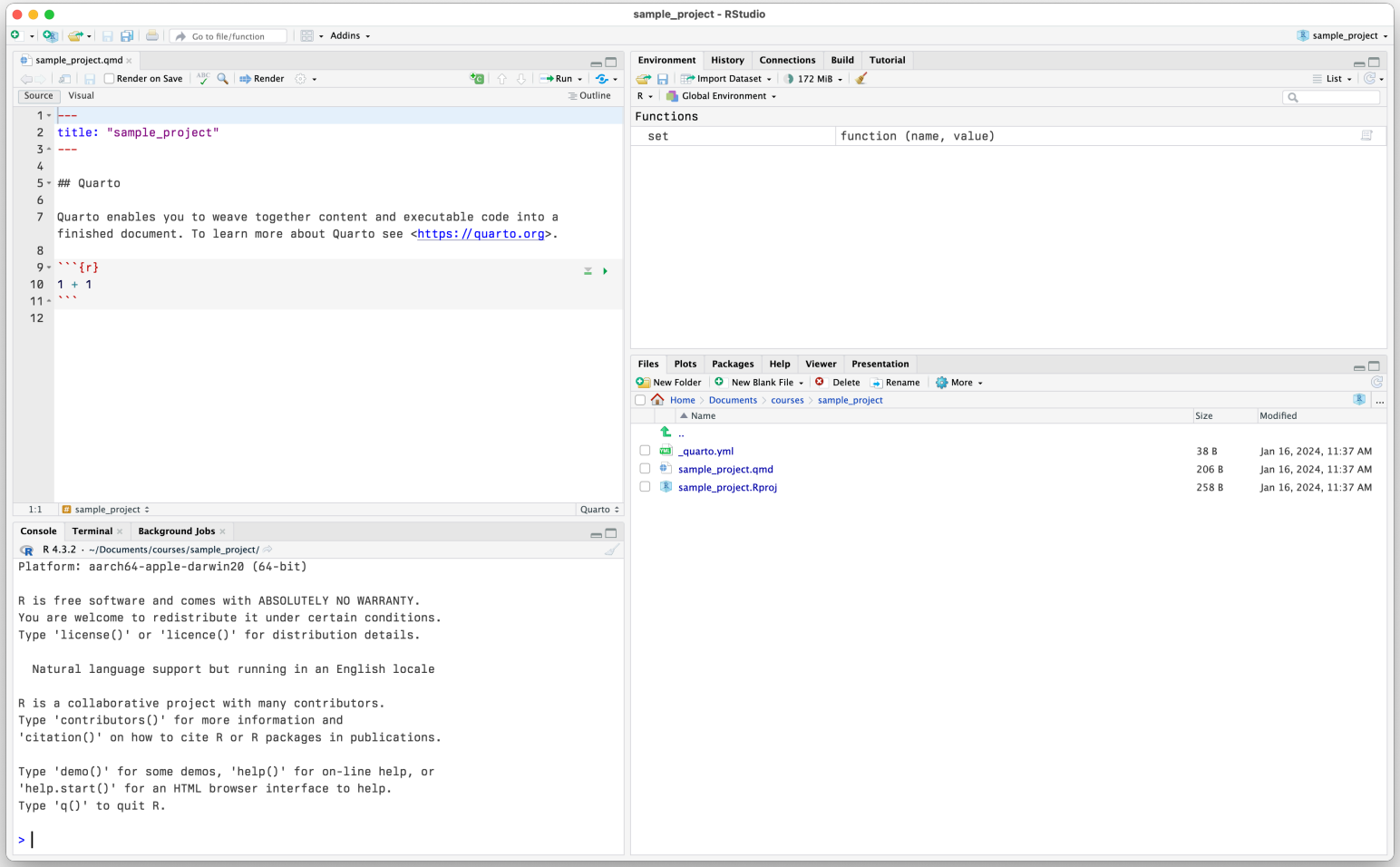
Into the Kitchen


RStudio®

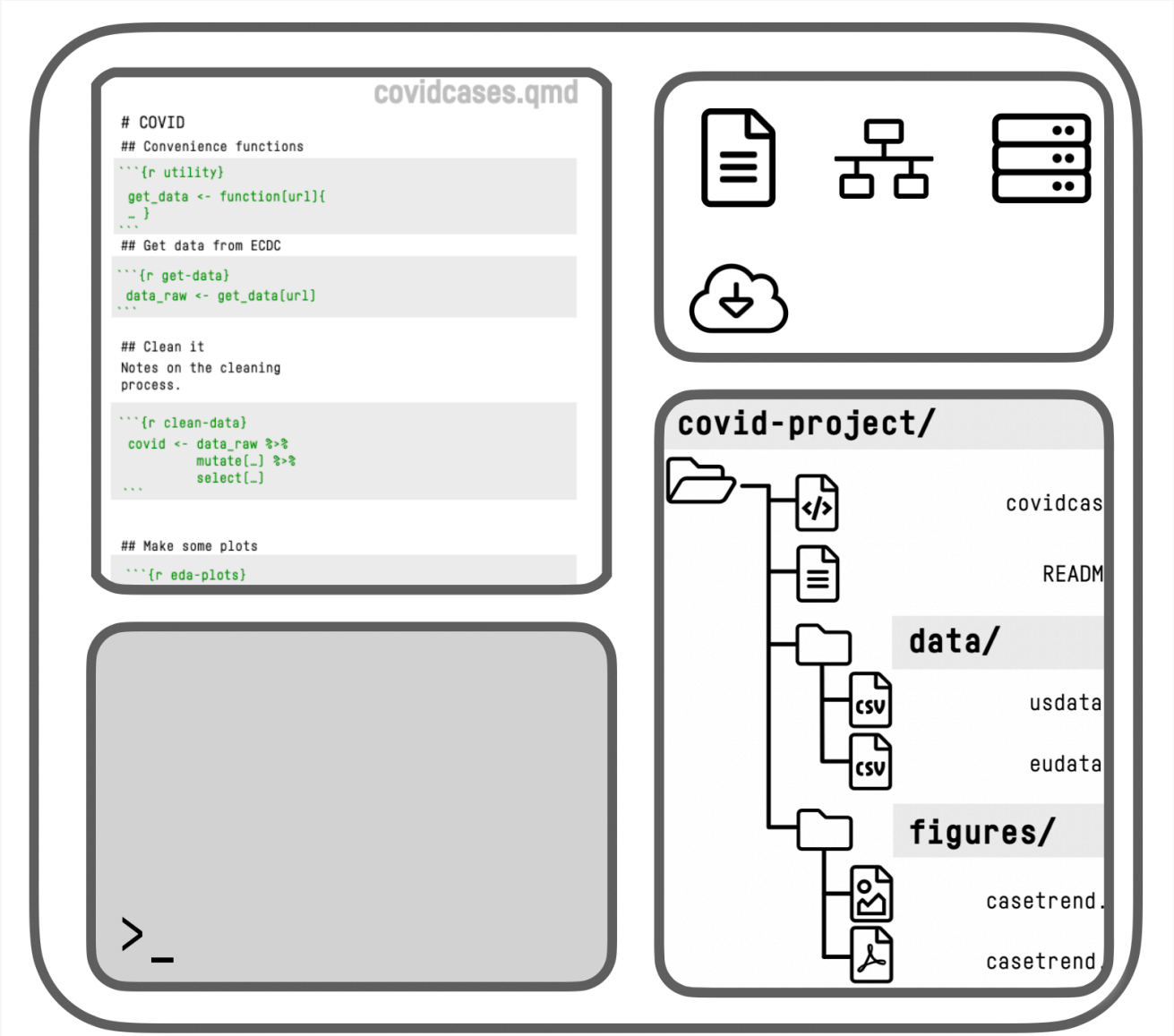
RStudio is an IDE for R

A kitchen is an IDE for Meals

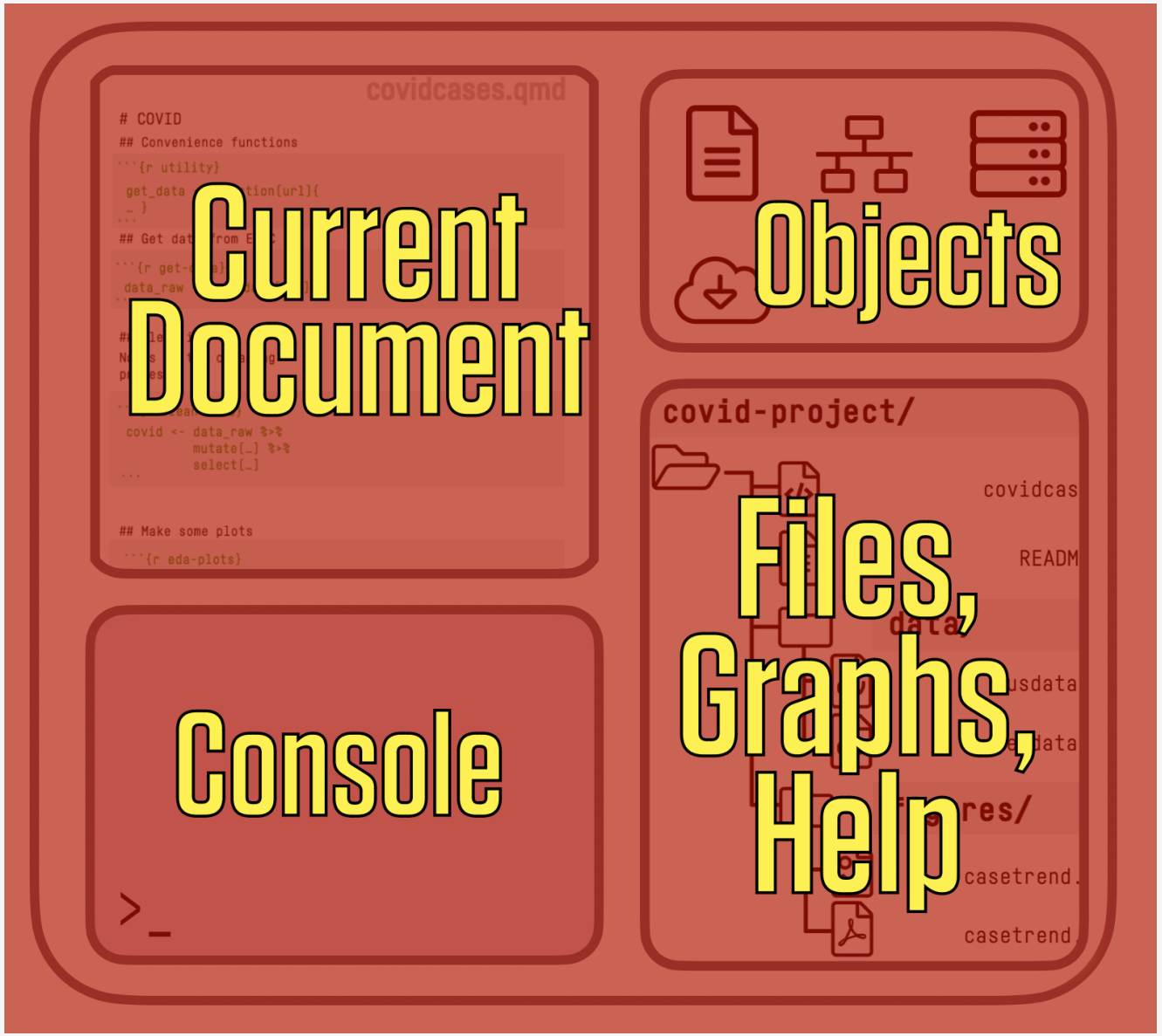
R and RStudio



RStudio at startup



RStudio schematic overview

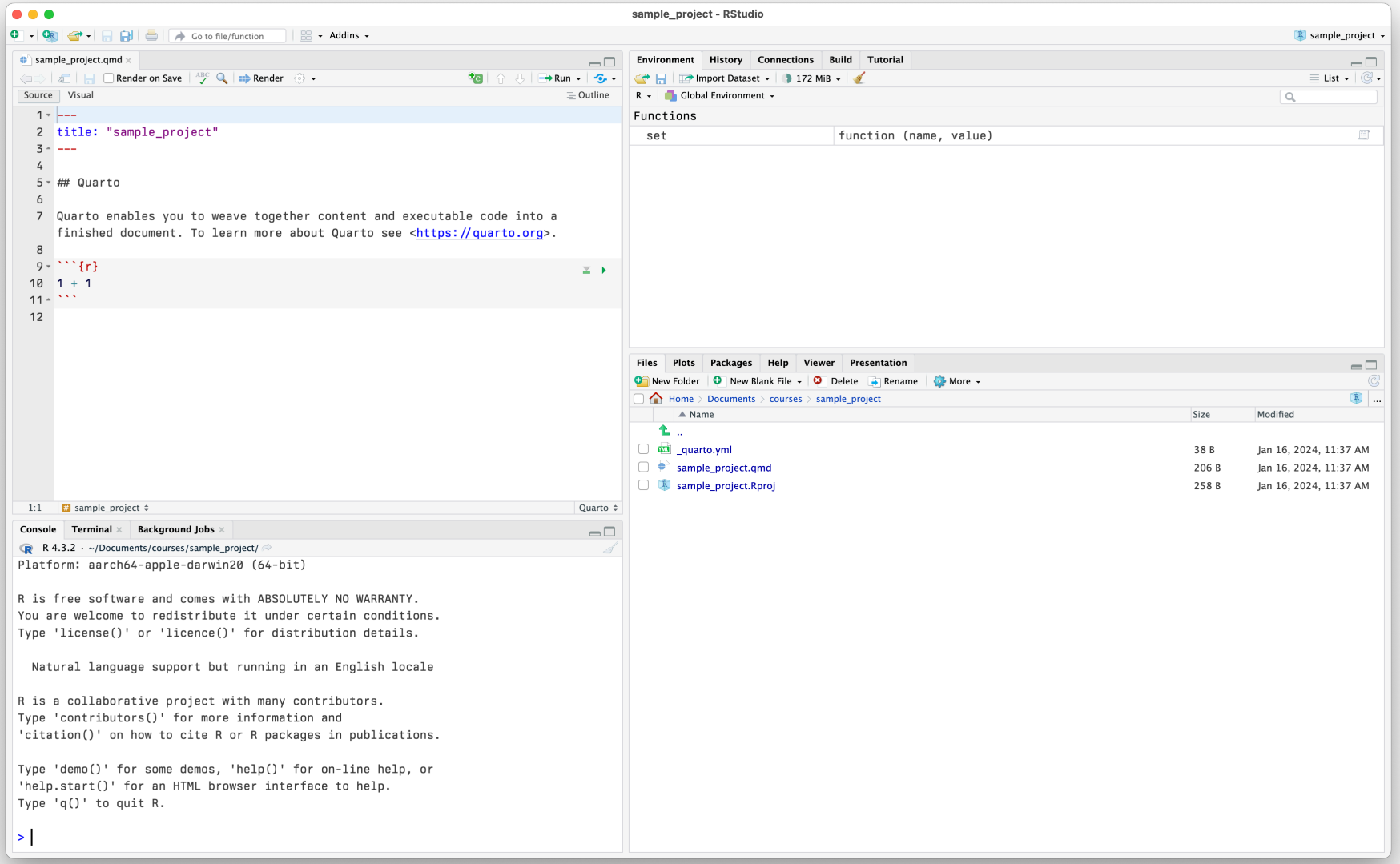


RStudio schematic overview

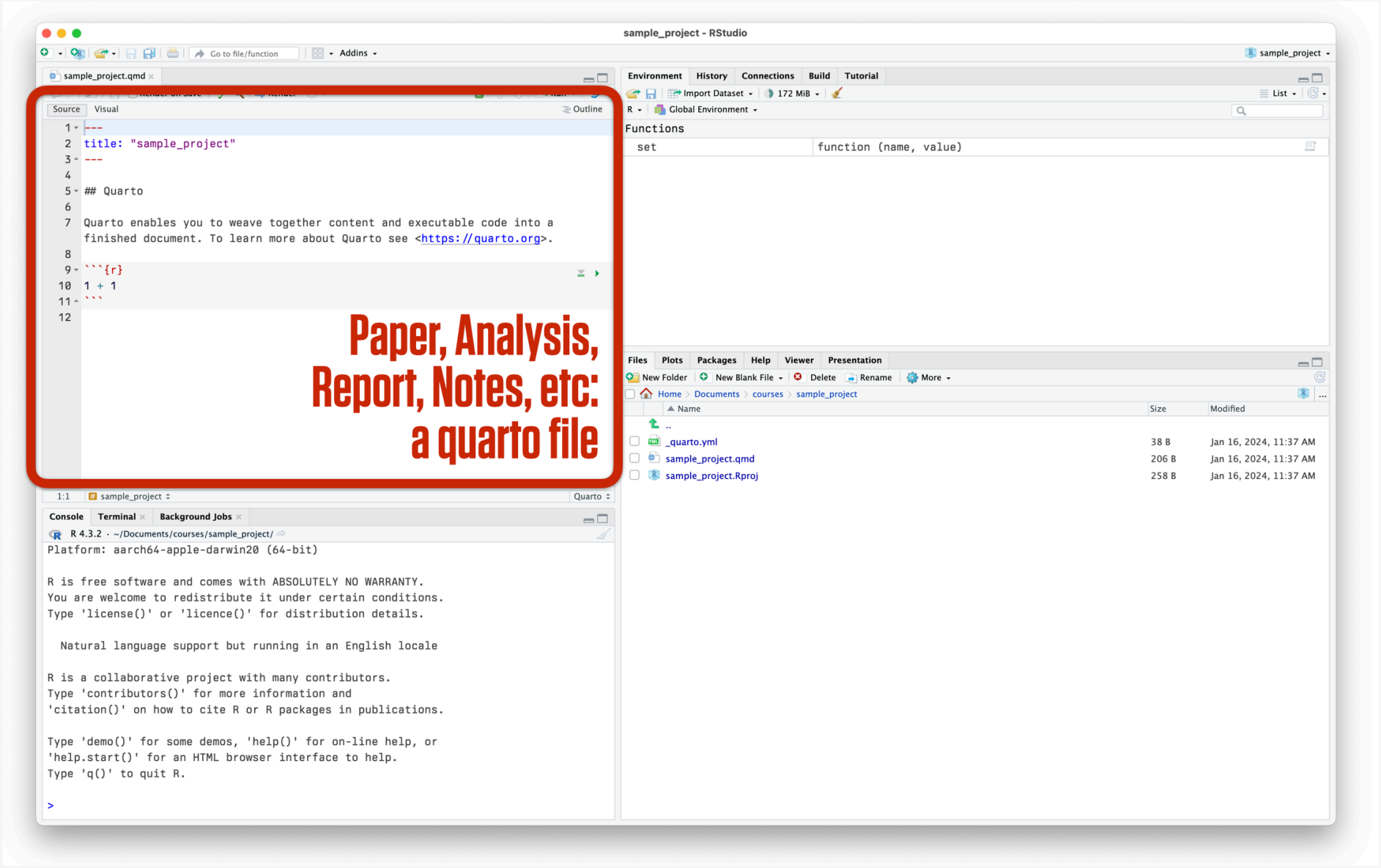
Think in terms of Data + Transformations, written out as code, rather than a series of point-and-click steps

Our starting data + our code is what's "real" in our projects, not the final output or any intermediate objects

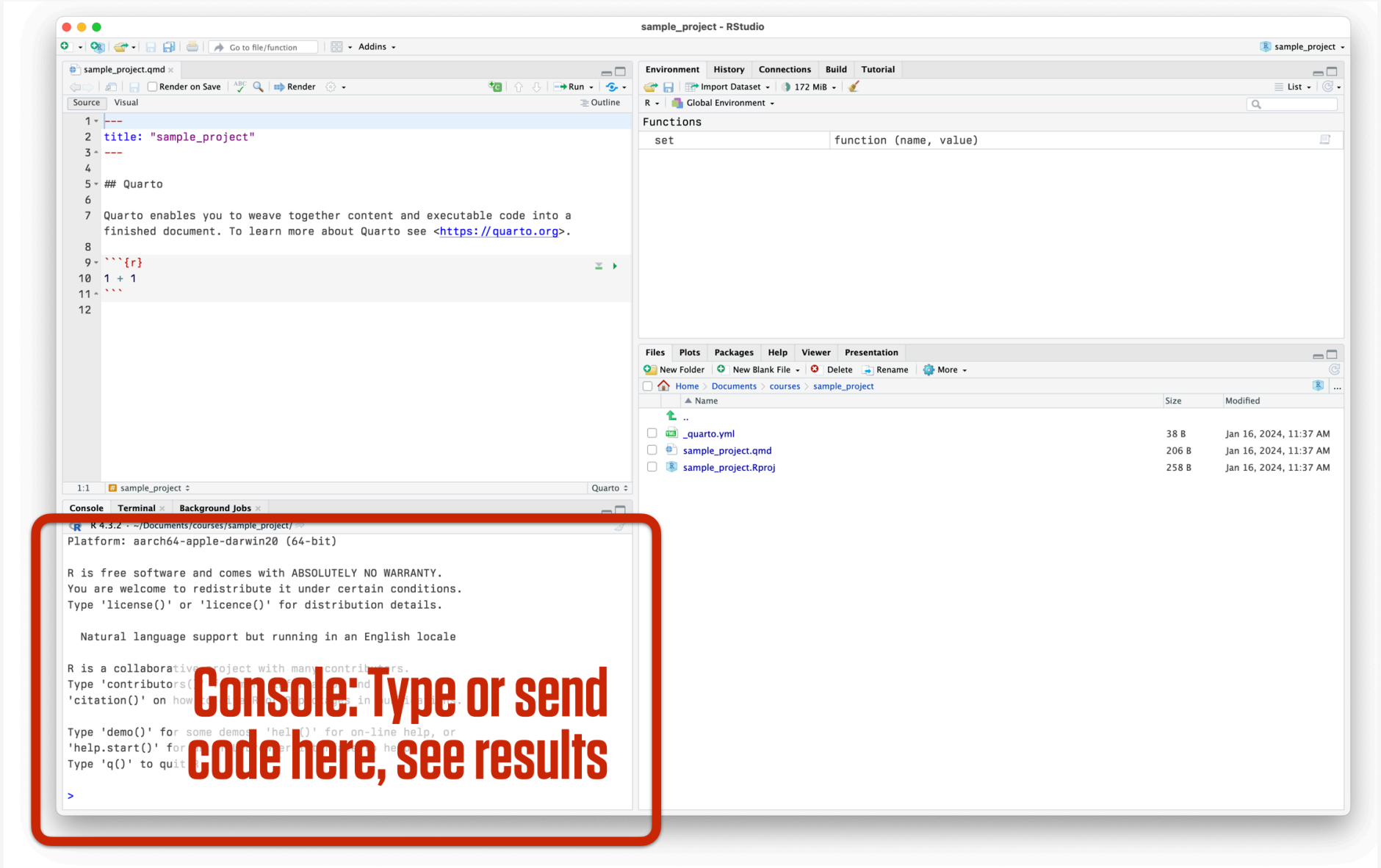
The IDE is the thing that helps us keep track of and control over the code we write and the outputs we produce.



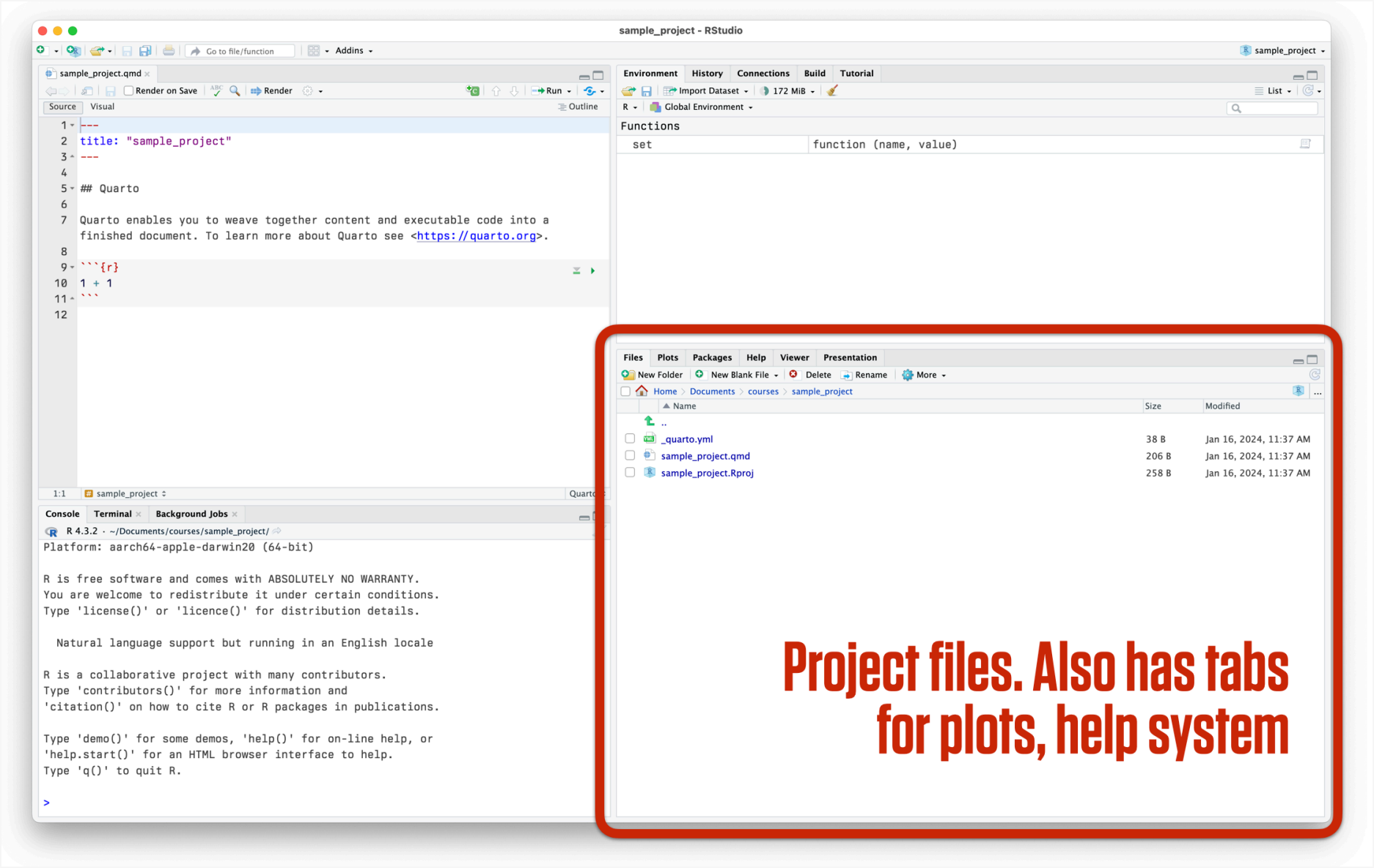
RStudio at startup



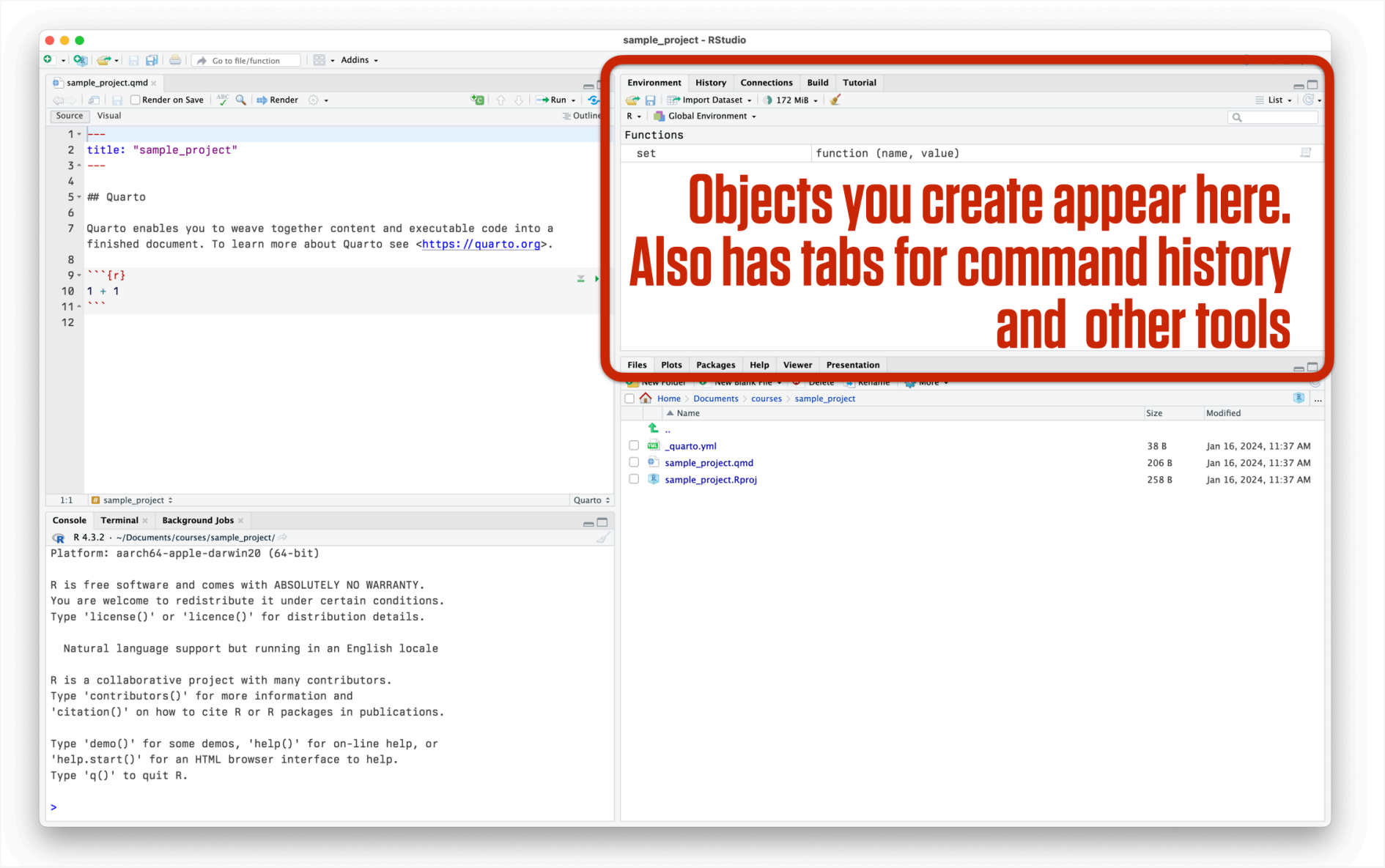
RStudio at startup



RStudio at startup

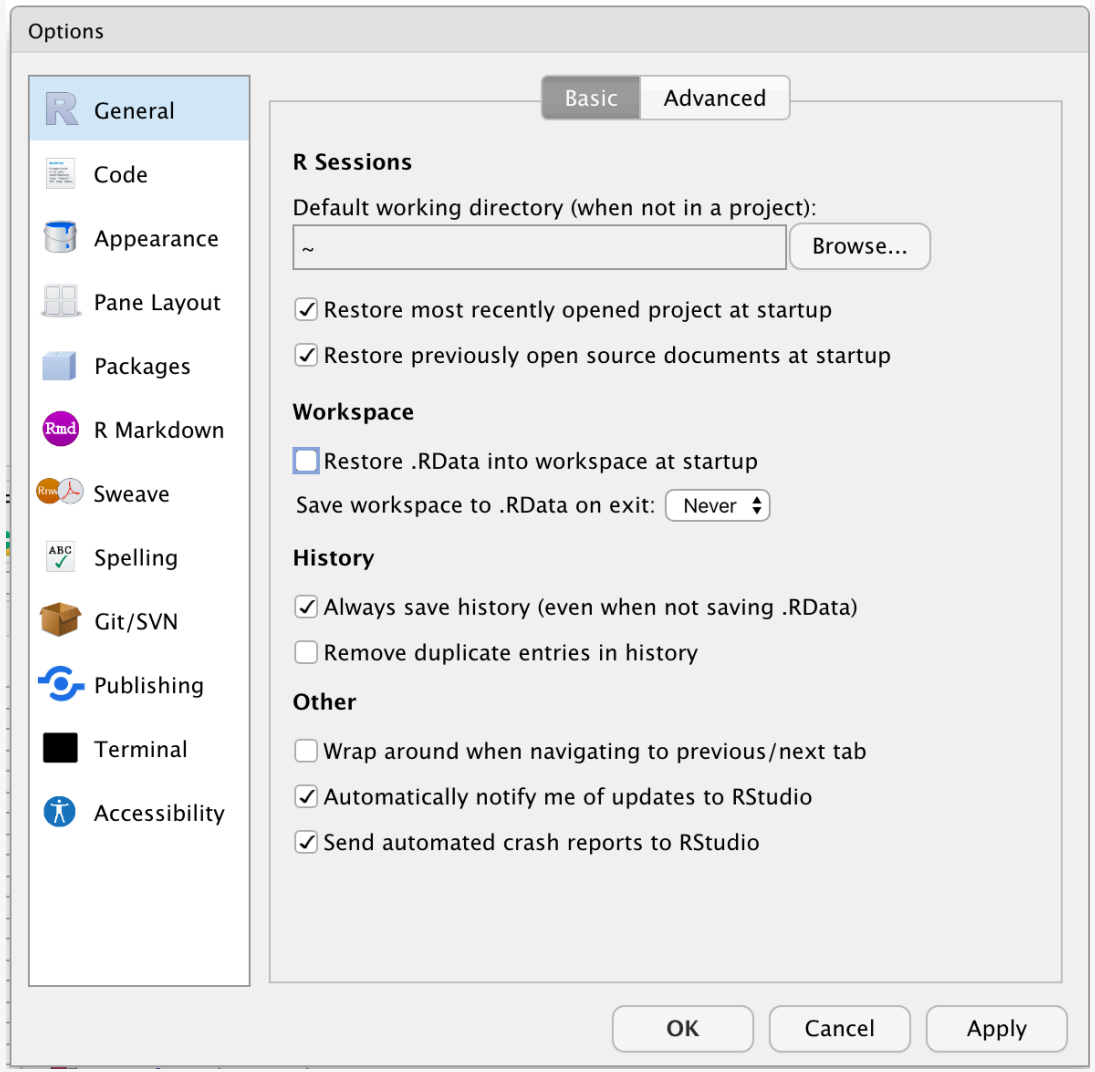


RStudio at startup



RStudio at startup

Your code is what's real in your project



Consider not showing output inline