A First Look at RStudio

Modern Plain Text Computing

Week 01

Kieran Healy
kieran.healy@duke.edu

August 2025

Getting to know R and RStudio

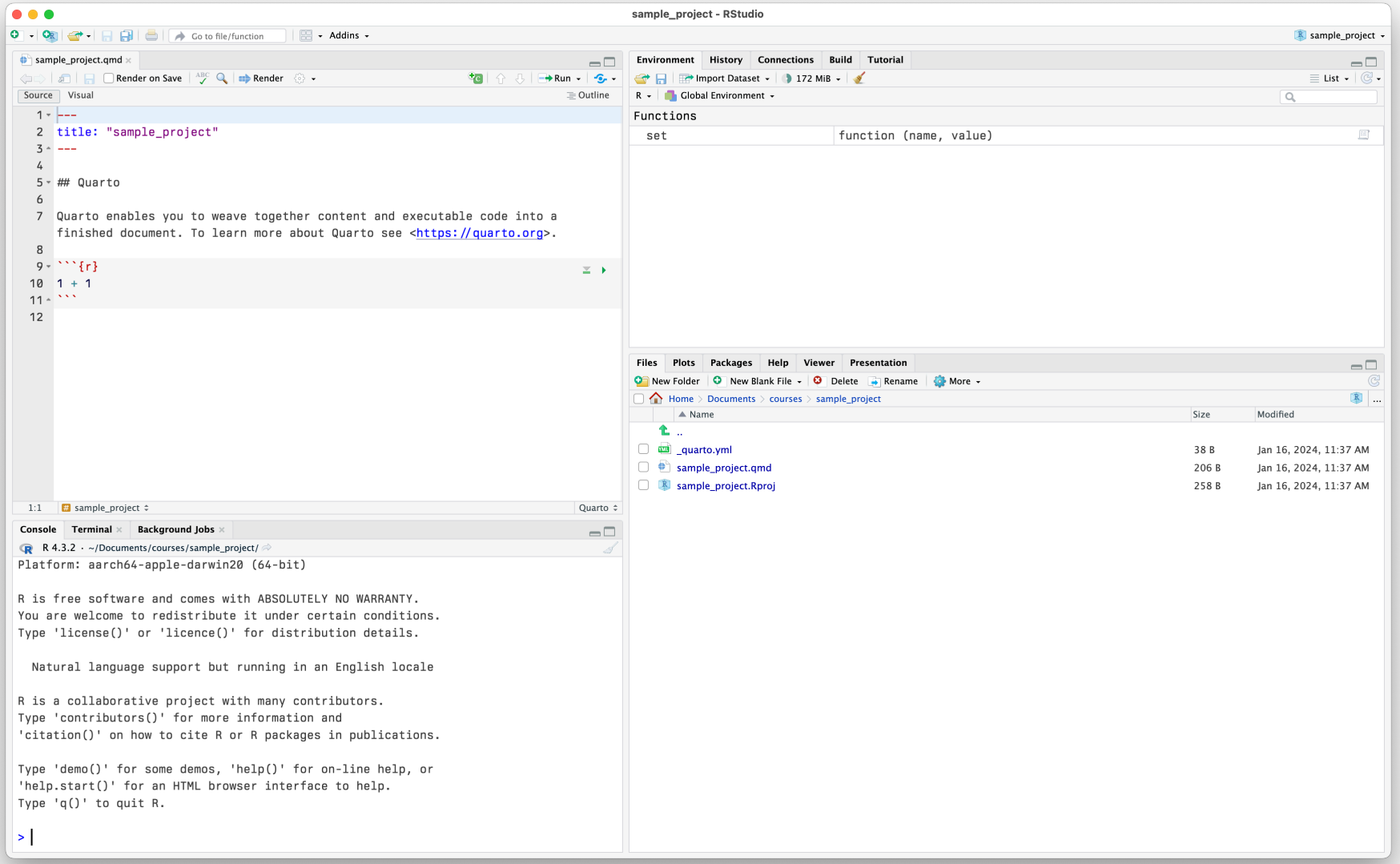
We want to do our work reproducibly


R
Studio®

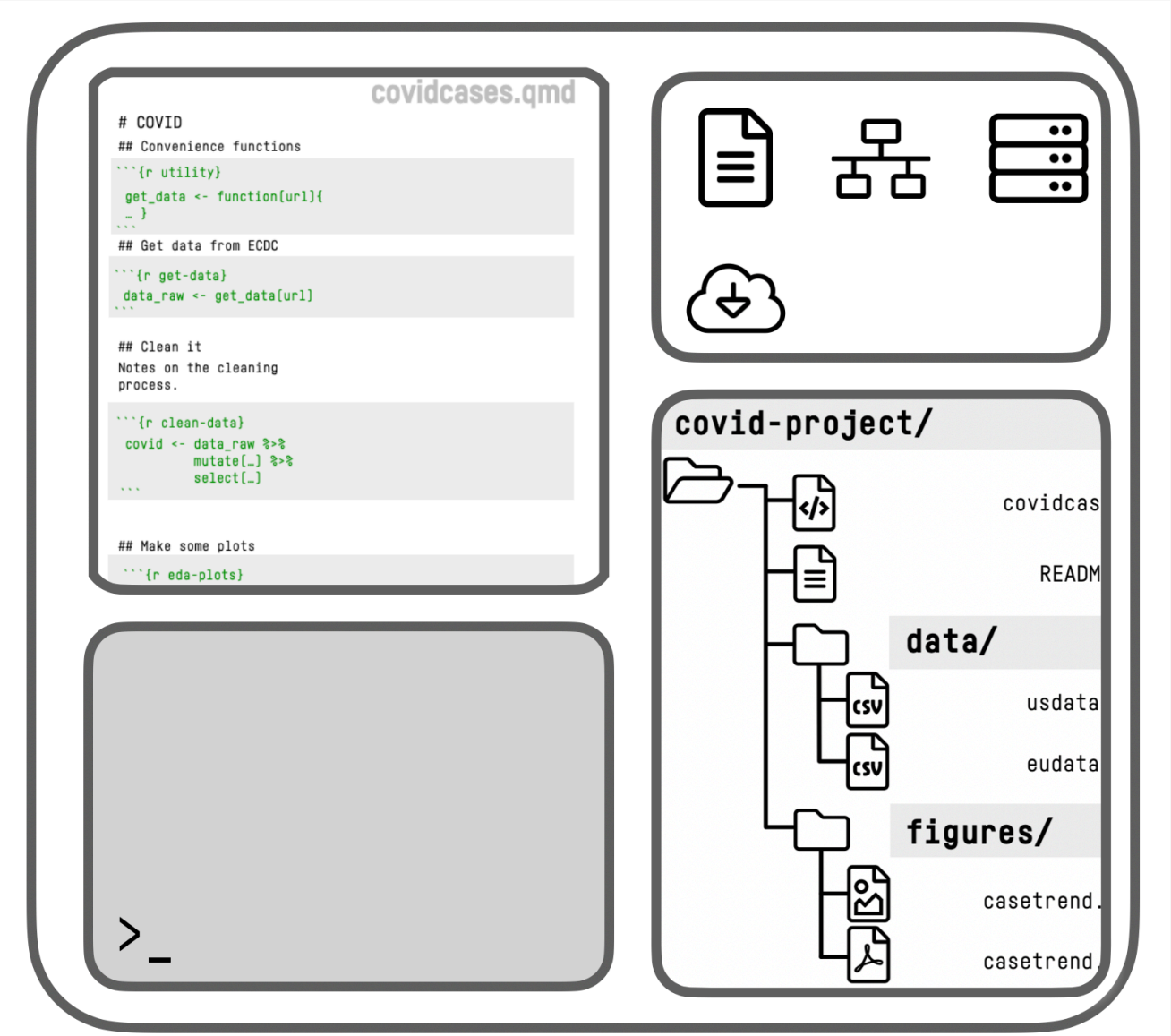
The RStudio IDE

An Integrated Development Environment

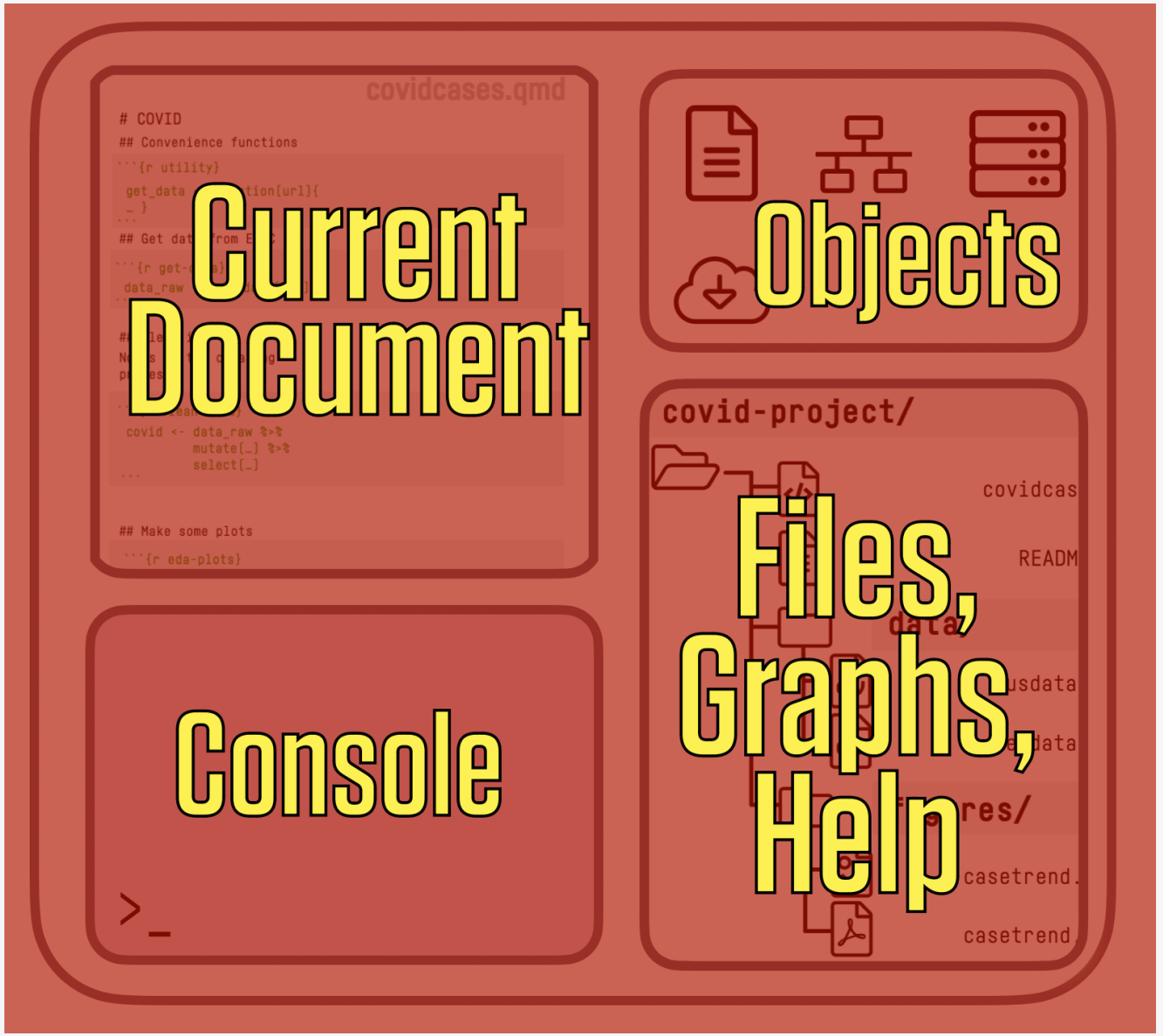
An IDE for Meals



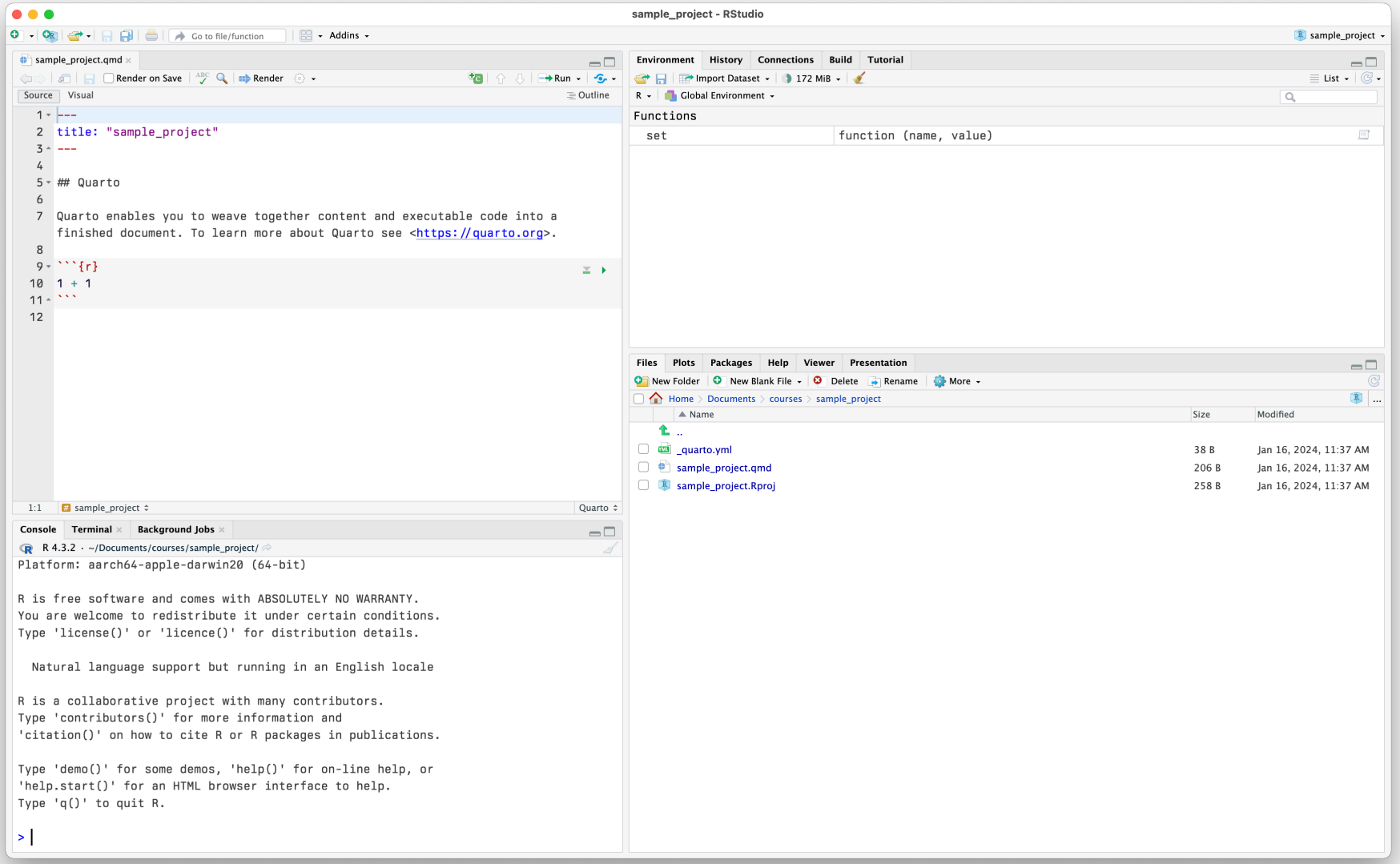
RStudio at startup with an empty sample project



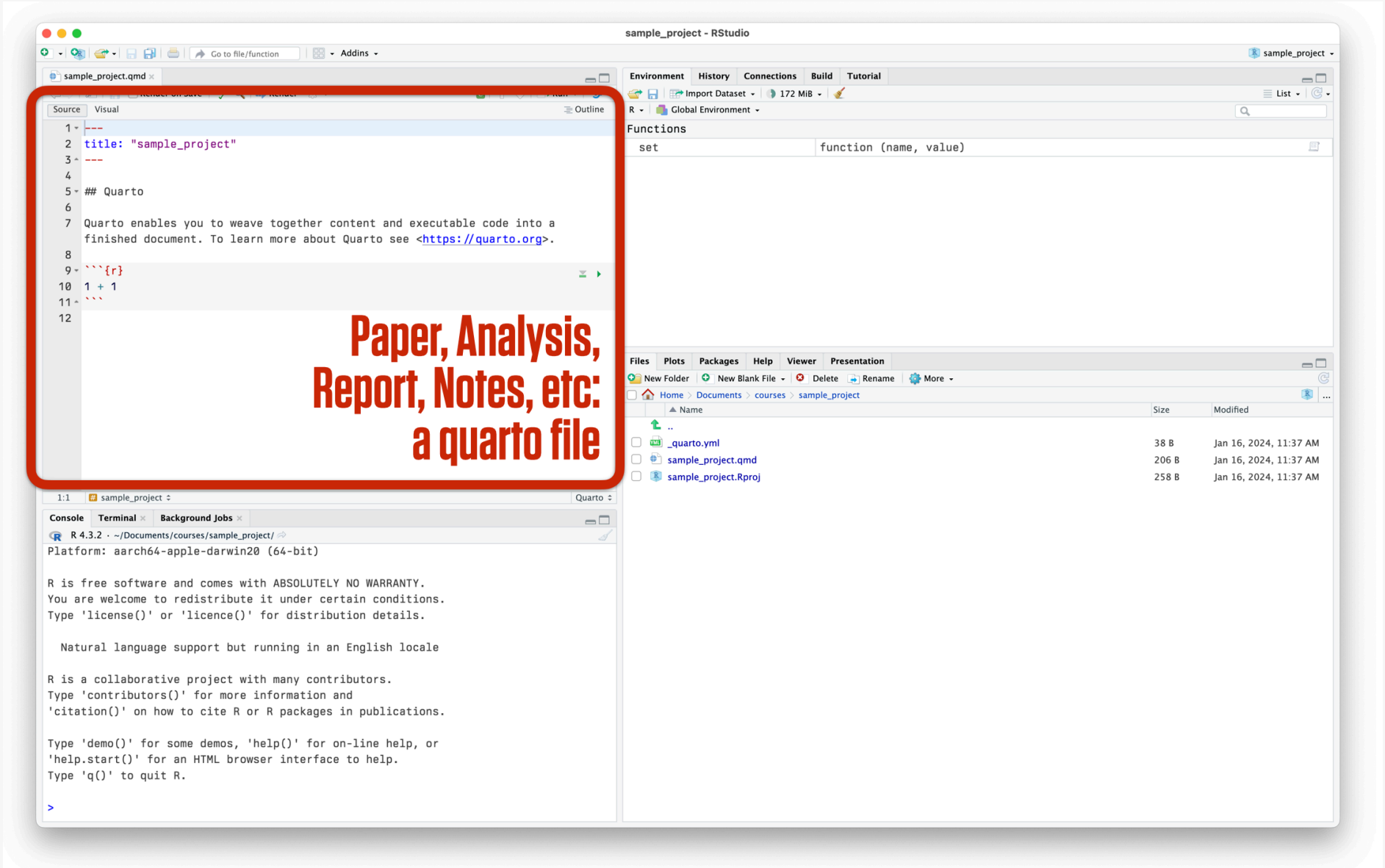
RStudio schematic overview



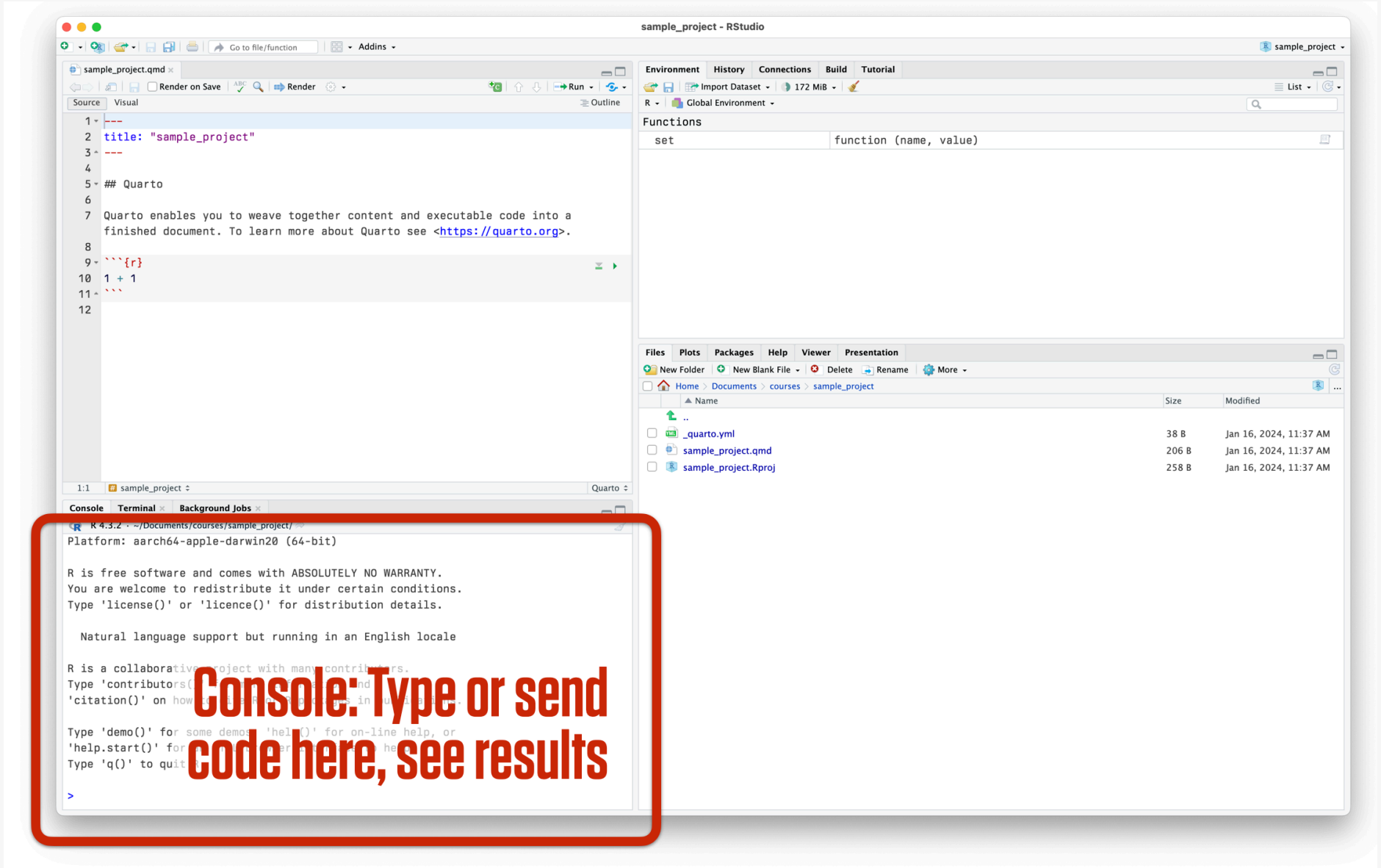
RStudio schematic overview



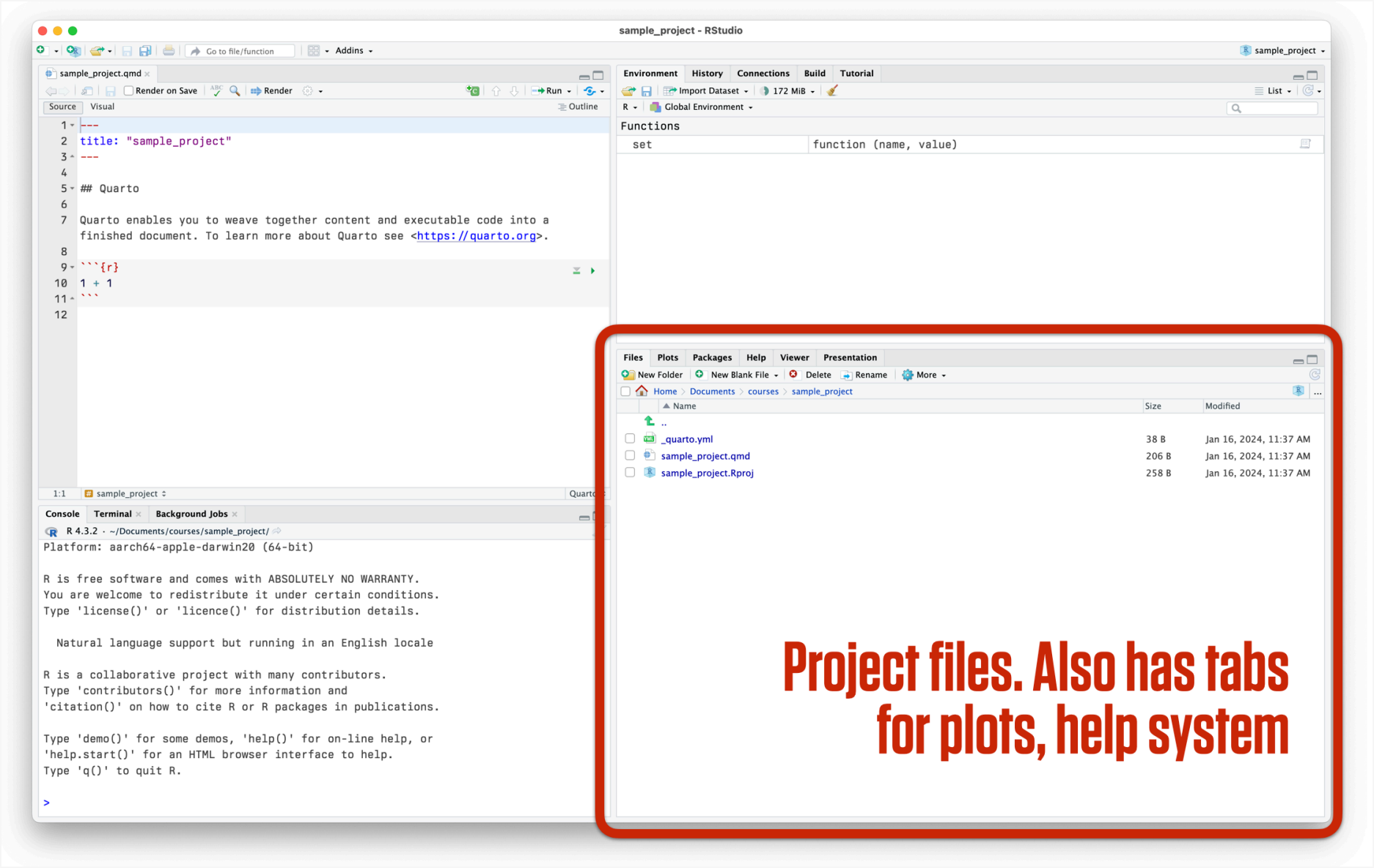
RStudio at startup



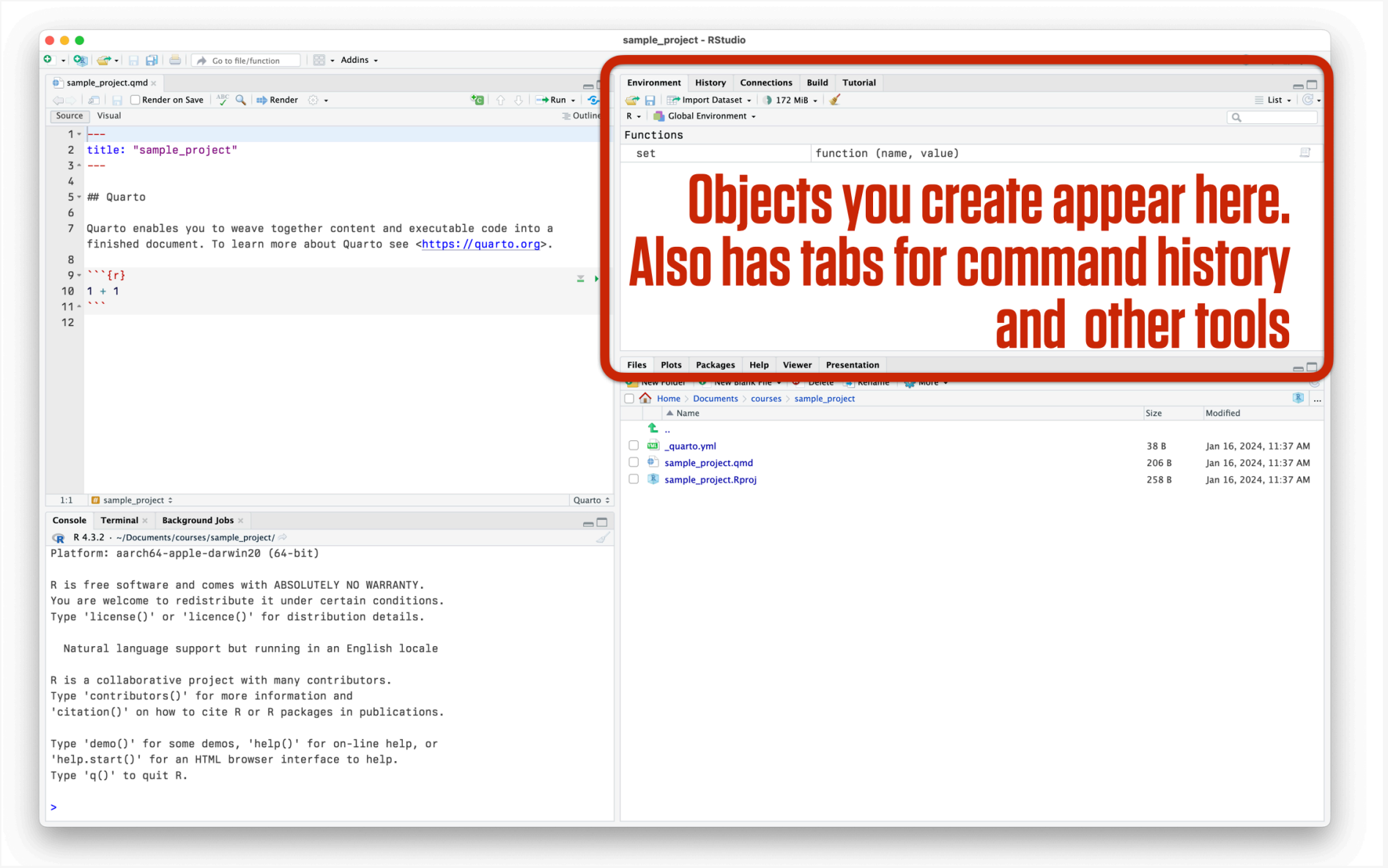
RStudio at startup



RStudio at startup

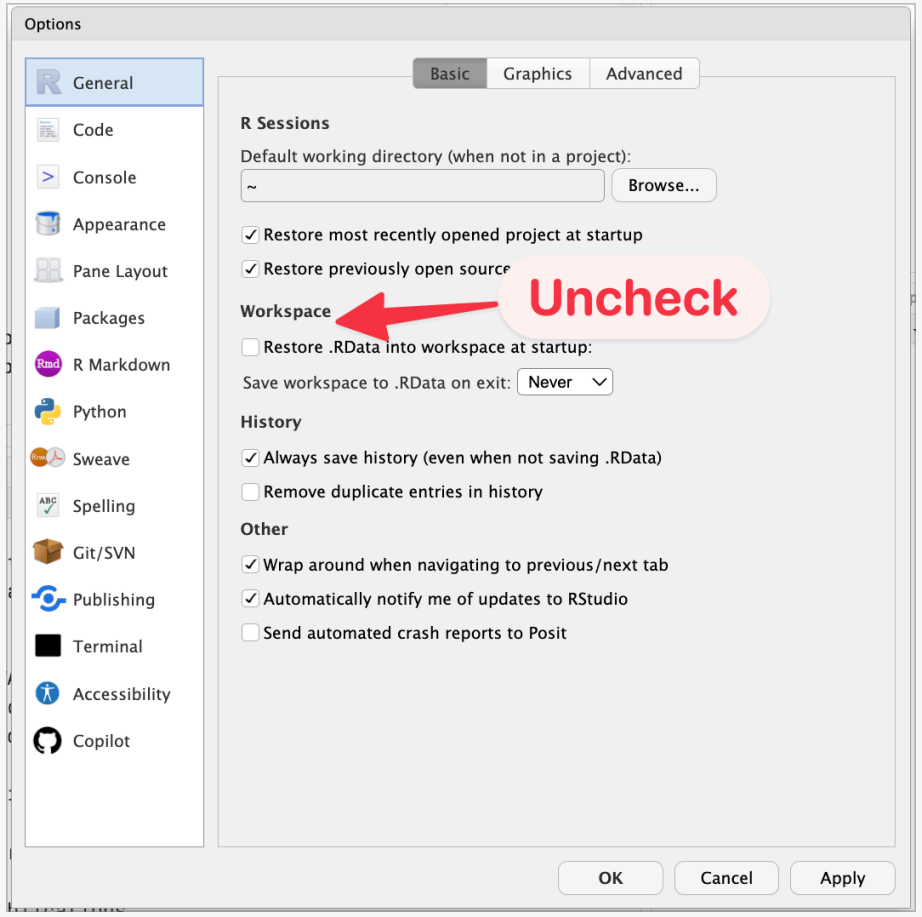


RStudio at startup



RStudio at startup

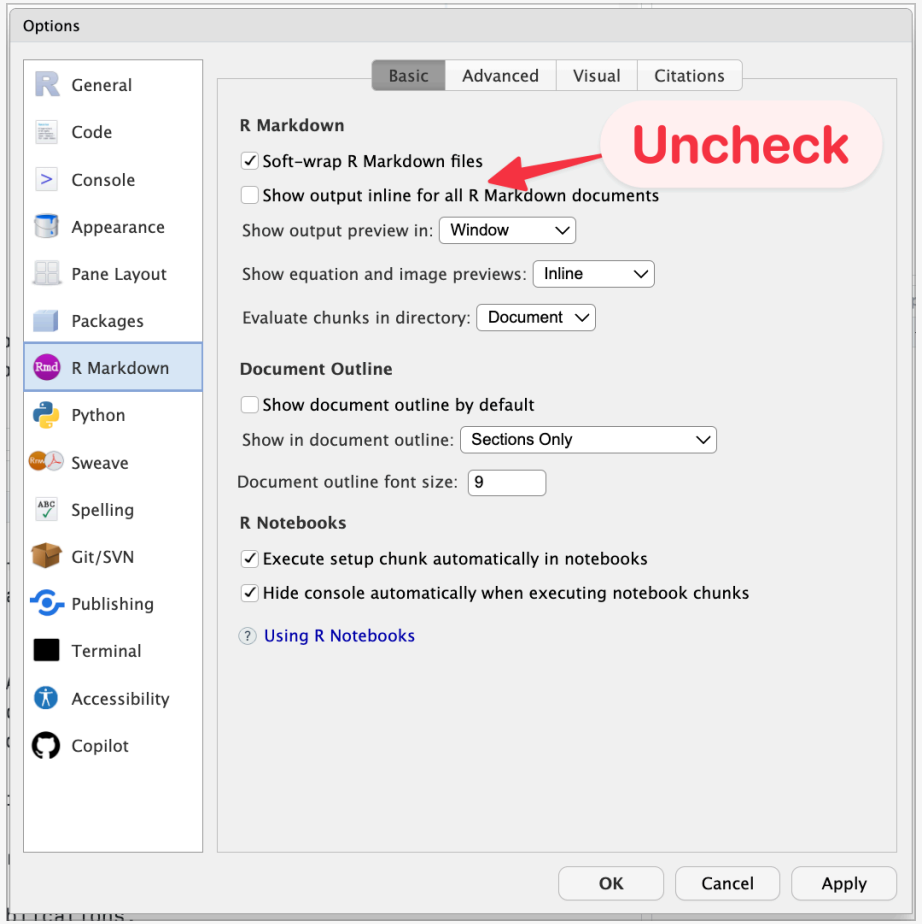
Your code is what's real in your project



Tools > Global Options > General

Uncheck "Restore .RData into workspace at startup"

Consider not showing output inline



Tools > Global Options > RMarkdown

Uncheck “Show output inline for all R Markdown documents”

Things an IDE brings together

A *text editor* for writing code and documents

A *console* or REPL (Read-Eval-Print Loop) for running code interactively

A *terminal* to talk to the operating system

A *debugger* to help find problems in your code

A *file manager* to navigate your project

A *version control interface* to manage changes to your code

A *viewer* for plots, tables, and other outputs

An *inspector* to see what's in your environment

RStudio is just one IDE

Positron (a newer IDE for R and Python)

VS Code (a general-purpose IDE that supports many languages)

JupyterLab (an IDE for Python notebooks)

Text editors like **Vim**, **Emacs**, and **Sublime Text** can be configured to work like IDEs, too.

An IDE is not required

You can write R code in any text editor and run it from the command line.

You can do the other things an IDE does with separate tools, via the command line or other applications.

However, IDEs make your life easier.

They also don't impose much structure on your projects. If you want to switch to another, or to no IDE at all, you can do so without much trouble.

Fundamentally …

A core set of interdependent tasks that require doing things with the computer.

A modular set of tools that do specific things, built around a text editor, a console, and plain-text configuration or code files.

Some method of organizing and orchestrating these tools.

Writing documents

Use Quarto to produce and reproduce your work

Where we want to end up

Where we want to end up

Covid Cases

Kieran Healy

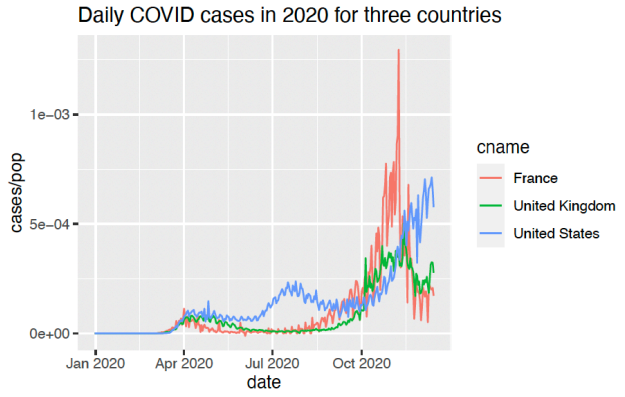
Introduction

We will look at some data from the `covdata` package.

cname	cases
France	2376852
United Kingdom	1849403
United States	16256754

A little graph

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.



PDF out

Where we want to end up

Covid Cases

AUTHOR
Kieran Healy

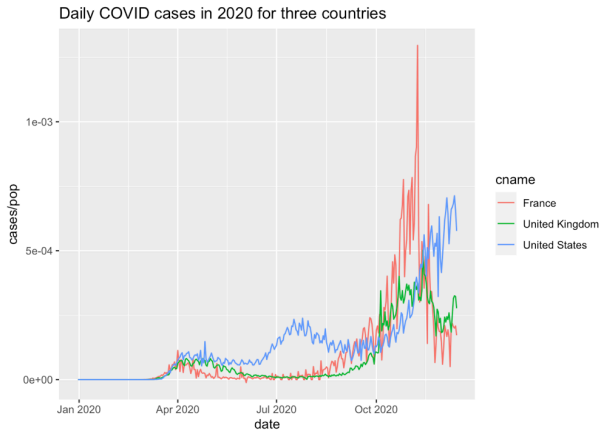
Introduction

We will look at some data from the `covdata` package.

cname	cases
France	2376852
United Kingdom	1849403
United States	16256754

A little graph

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.



HTML out

Where we want to end up

Covid Cases

Kieran Healy

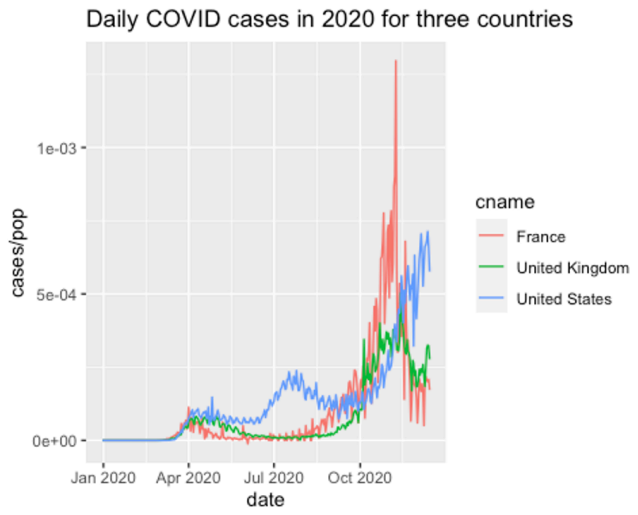
Introduction

We will look at some data from the `covdata` package.

cname	cases
France	2376852
United Kingdom	1849403
United States	16256754

A little graph

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.



Word out

How to get there?

```
# COVID                                    covidcases.R
# Get data from ECDC
# FIXME Write a fn to
# do this
data_raw ← read_csv(url)

# Clean it
# Notes on the cleaning
# process.

 covid ← data_raw ▷
          mutate(…) ▷
          select(…)



# Make a plot
 covid ▷
      ggpplot(…) +
      geom_line(…) ▷
 ggsave()
```

We could write an **R script** with some notes inside, using it to create some figures and tables, paste them into our paper, which we write separately.

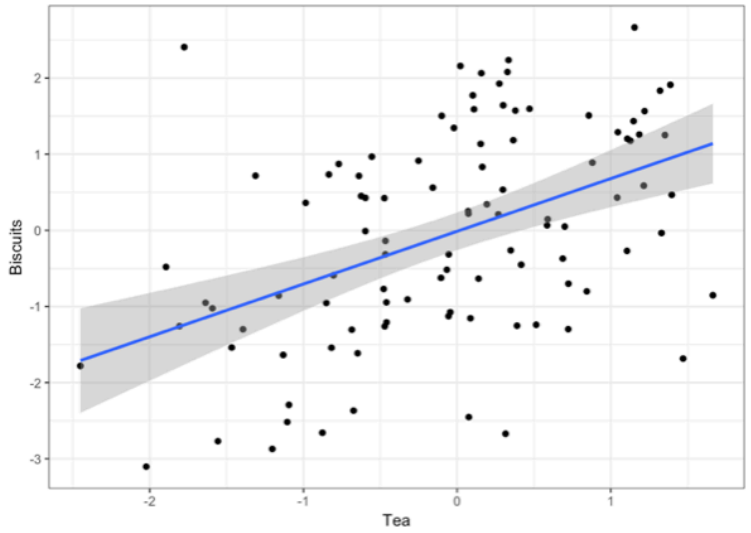
This will work fine. In fact, the more complex our projects get, the more likely it is we will write more standalone code like this. It will also look less and less like a single all-in-one-breath script and more like a structured collection R files that combine to do many things.

But to begin—and in fact for quite a long time after—we can also do things a little differently, by taking a more *notebook-based* approach. For many use-cases this will be better.

We can make this ...

1. Lorem Ipsum

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do **eiusmod tempor** incididunt ut labore et dolore magna aliqua. Ut enimad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.



Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

… by writing this

```
# Lorem Ipsum

Lorem ipsum dolor sit amet, consectetur adipisicing elit,
sed do *eiusmod tempor* incididunt ut labore et dolore magna
aliqua. Ut enimad minim veniam, quis nostrud exercitation
ullamco laboris nisi ut aliquip ex ea commodo consequat.

library(ggplot2)
tea <- rnorm(100)
biscuits <- tea + rnorm(100, 0, 1.3)
data <- data.frame(tea, biscuits)
p <- ggplot(data, aes(x = tea, y = biscuits)) +
    geom_point() +
    geom_smooth(method = "lm") +
    labs(x = "Tea", y = "Biscuits") + theme_bw()
print(p)

Duis aute irure dolor in reprehenderit in voluptate velit esse
cillum dolore eu fugiat nulla pariatur. Excepteur sint
occaecat cupidatat non proident, sunt in culpa qui officia
deserunt mollit anim id est laborum.
```

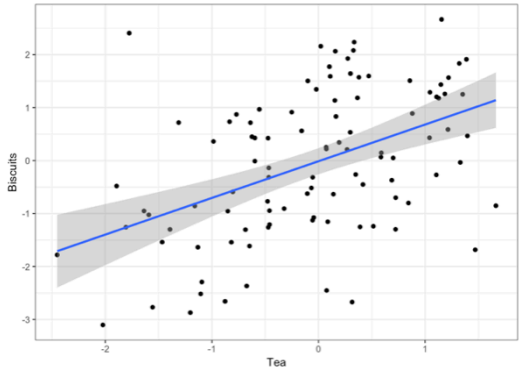
The code gets replaced by its output

This way of doing things is called a *Literate Programming* or *Notebook* approach.

```
# Lorem Ipsum

Lorem ipsum dolor sit amet, consectetur adipisicing elit,
sed do *eiusmod tempor* incididunt ut labore et dolore magna
aliqua. Ut enimad minim veniam, quis nostrud exercitation
ullamco laboris nisi ut aliquip ex ea commodo consequat.

```
```
library(ggplot2)
tea <- rnorm(100)
biscuits <- tea + rnorm(100, 0, 1.3)
data <- data.frame(tea, biscuits)
p <- ggplot(data, aes(x = tea, y = biscuits)) +
    geom_point() +
    geom_smooth(method = "lm") +
    labs(x = "Tea", y = "Biscuits") + theme_bw()
print(p)
```
```

Duis aute irure dolor in reprehenderit in voluptate velit esse
cillum dolore eu fugiat nulla pariatur. Excepteur sint
occaecat cupidatat non proident, sunt in culpa qui officia
deserunt mollit anim id est laborum.
```

1. Lorem Ipsum

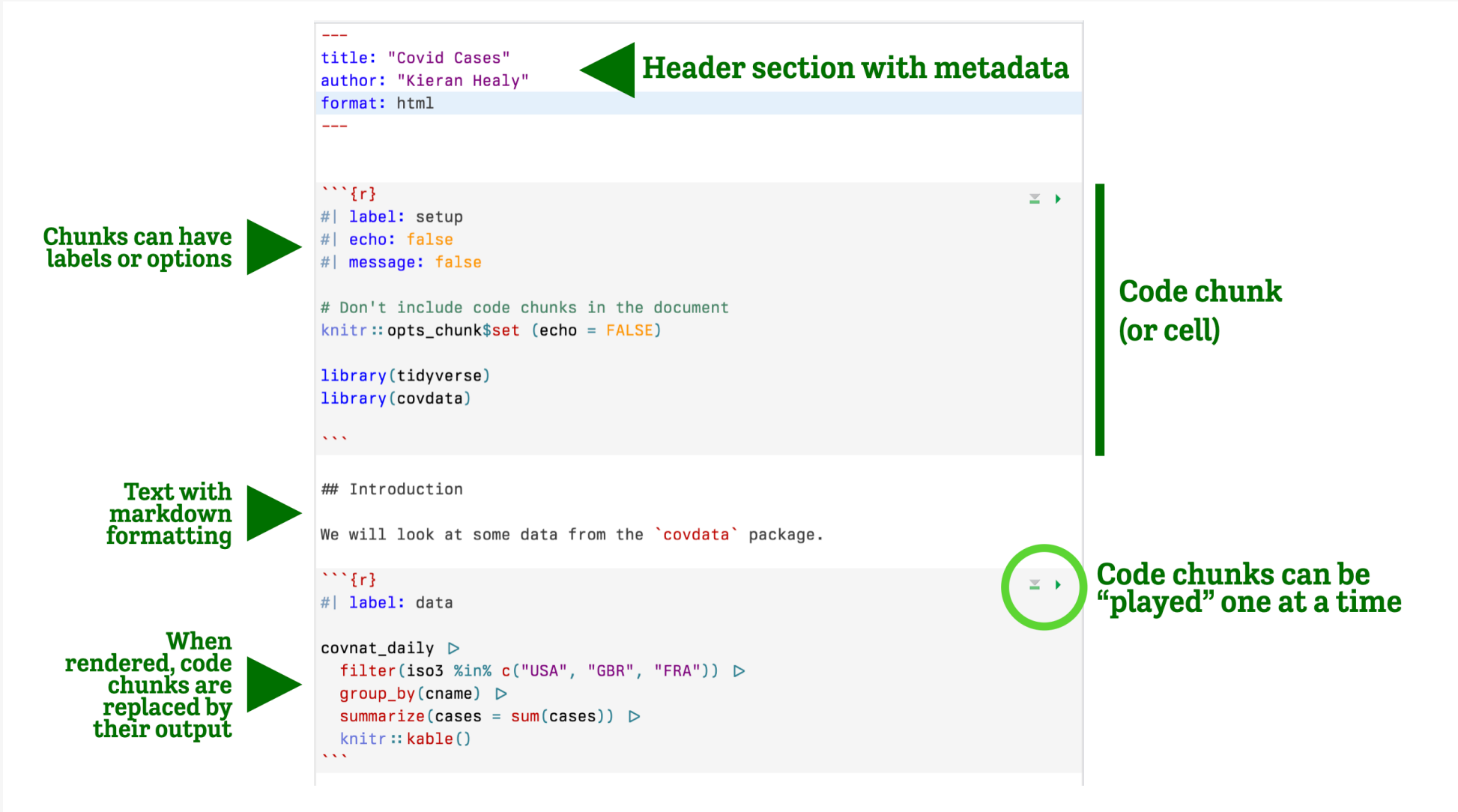
Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do **eiusmod tempor** incididunt ut labore et dolore magna aliqua. Ut enimad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.



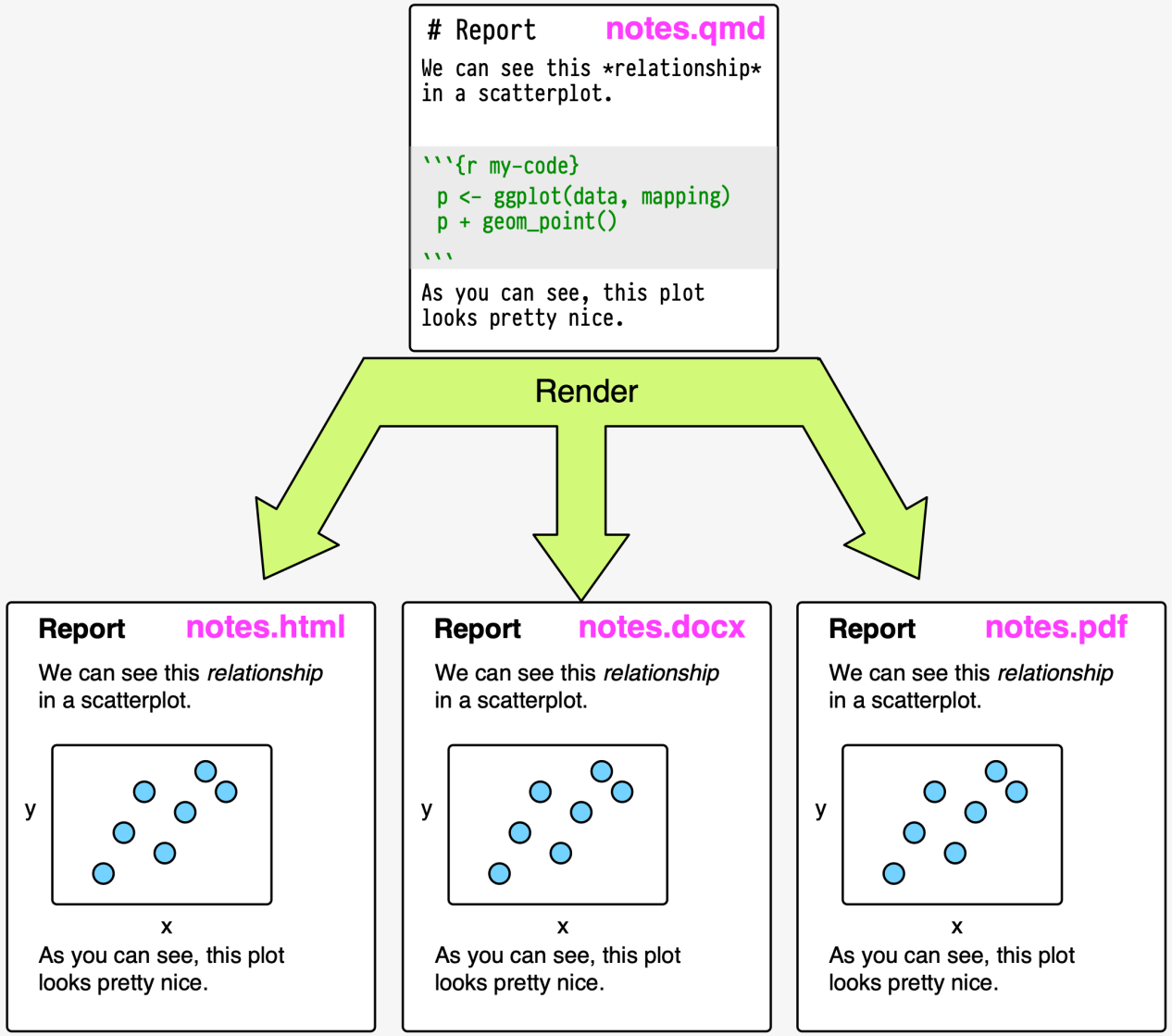
Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

````
---
title: "Covid Cases"
author: "Kieran Healy"
format: html
---



```{r}
#| label: setup
#| echo: false
#| message: false

# Don't include code chunks in the document
knitr::opts_chunk$set (echo = FALSE)

library(tidyverse)
library(covdata)

```


## Introduction

We will look at some data from the `covdata` package.

```{r}
#| label: data

covnat_daily ▷
  filter(iso3 %in% c("USA", "GBR", "FRA")) ▷
  group_by(cname) ▷
  summarize(cases = sum(cases)) ▷
  knitr::kable()
```
````

Quarto document



Quarto document annotated



This approach has its limitations, but it's *very* useful and has many benefits.

Strengths and weaknesses

Notebooks work smoothly when

- Your document or report is small and self-contained.
- Your analysis is quick or lightweight.
- You are making slides.
- You are making a lot of similar reports from a template.
- You regularly refer to calculated items in the text of your analysis.

Sidenote: In Practice

Even when things get complicated, notebook-style documents like Quarto files are great as component parts of larger projects. The more complex your project, the less likely it will straightforwardly fit into a *single* notebook. The same is true of script-based approaches. The more complex a project, the more you will break it up into smaller, more tractable pieces. You will *re-factor* it, as programmers say, and make it more *modular*.

You may find yourself, for example, splitting parts of a complex document up into different pieces. The pieces will contain code that cleans and pre-processes data, runs analyses, and produces some outputs. You can then incorporate those into a Quarto document indirectly. Not by copying and pasting them, but by pointing to those outputs and making use of them to make your tables and figures, and so on.

Basic markdown summary

Desired style	Use the following Markdown annotation
Heading 1	`# Heading 1`
Heading 2	`## Heading 2`
Heading 3	`### Heading 3` (Actual heading styles will vary.)
Paragraph	Just start typing
Bold	`**Bold**`
Italic	`*Italic*`
Images	`[Alternate text for image](path/image.jpg)`
Hyperlinks	`[Link text](https://www.visualizingsociety.com/)`
Unordered Lists	
- First	`- First`
- Second.	`- Second`
- Third	`- Third`
Ordered Lists	
1. First	`1. First`
2. Second.	`2. Second`
3. Third	`3. Third`
Footnote.[1]	`Footnote[^notelabel]`
[1]The note's content.	`[^notelabel] The note's content.`

The right frame of mind

This is like learning how to drive a car, or how to cook in a kitchen … or learning to speak a language.

After some orientation to what's where, you will learn best by *doing*.

Software is a pain, but you won't crash the car or burn your house down.

Create a new Quarto Project

File > New Project > New Directory > Quarto Project

Choose a location for a folder named `mptc`

Initialize a git repository

Check the “Open in new session” box

Click “Create Project”

Create a new Quarto Document

Once the project opens, create a new file (File > New File > Quarto Document)

Choose “Document” and “HTML” as the output format

Save the file as `scratch.qmd` in the `mptc` folder

For now

Get used to editing text files in RStudio and rendering them to HTML or PDF.

Don’t worry about writing any code at this point. Treat it as a way to take notes for the class.

In the same way that you should be reading around and beyond the assigned readings for other classes, you should also be exploring R and RStudio on your own. Try to do things; see what happens.