Introduction to R from Zero to Hero
Part 1
"Introduction to R and RStudio"

Isabella Gollini

isabella.gollini@ucd.ie
igollini.github.io

UCD Netsoc – 8th November 2022
All material available at: github.com/igollini/netsoc22

University College Dublin
Ireland's Global University

What will we do today?

1. Introduction to R and RStudio (Part 1)
2. RStudio Projects (Part 1)
3. Data import and handling (Part 1)
4. Data manipulation and summaries with `dplyr` (Part 2)
5. Graphics with `ggplot2` (Part 2)
6. Dynamic documents with R Markdown (Part 1 and 2)

Why R

R is a language and environment for statistical computing and graphics `https://www.r-project.org/`

- R was created by Ross Ihaka and Robert Gentlemen in the early 90s at the University of Auckland New Zealand.
- R was developed from another statistical language S that was developed at Bell Laboratories by John Chambers and colleagues.
- R is free and open-source so many people are contributing to its development.
- R is superior in many ways to existing commercial products such as SAS, SPSS or Stata.
- It is available for Windows, Mac and Linux
- R can be used online with RStudio Cloud `https://rstudio.cloud/`

What can R do?

- Data Handling
- Analysis
- Reporting
- Programming

R Ecosystem

Base

base

create R objects
summaries
math functions

recommended

statistics
graphics
example data

Contributed Packages

CRAN
cran.r-project.org

main repos
~18000 pkgs

Bioconductor
bioconductor.org

bioinformatics
>2000 pkgs

GitHub
github.com

devel pkgs
GitHub-only pkgs

IDEs for R

- It is easiest to use R via an Integrated Development Environment (IDE).
- An IDE provides a "Front End" to R which can make it a little bit easier to use.
- We will use RStudio as an IDE, though there are many others available.
- Features provided by RStudio include:
  - syntax highlighting, code completion, smart indentation
  - interactively send code chunks from editor to R
  - organise multiple scripts, help files, plots
  - search code and help files
- RStudio provides a few shortcuts to help write code in the R console go to "Help - Keyboard Shortcuts Help"

Installing R and RStudio

- We need to install R first and then RStudio.
- R can be installed from: `https://cloud.r-project.org/`
- RStudio can be installed from: `https://posit.co/download/rstudio-desktop/`
- See the `README.md` file on github for installation instructions.

R Commands

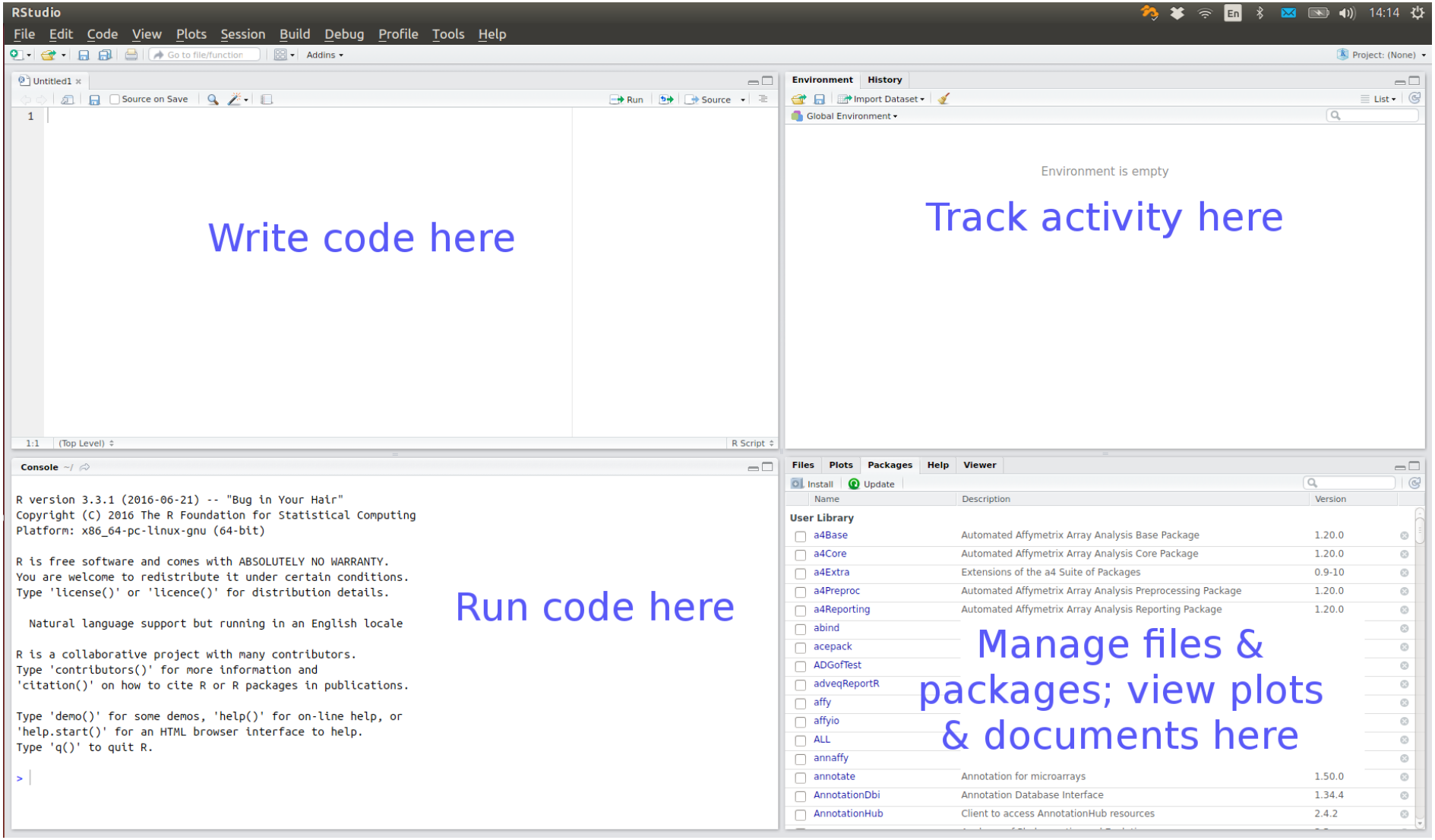
We can type commands directly into the R console:

```
3 + 4
?"+" # look up help for "+". You need quotes!
x <- 3 + 4 # store 3 + 4 into the object x
x # print the object R
y <- log(x) # store the natural log of x in the object y
log(x) -> y # same as before using -> instead of <-
y = log(x) # same as before using the = instead of <-
3 == 4 # == is the comparison operator for equal
3 != 4 # != is the comparison operator for not equal
?log # look up help for the function log
ls() # list of objects in the current workspace
rm(x) # remove the object x
```

Posit/RStudio

- RStudio (now Posit since Nov 2022) was founded in 2009 with the vision of creating high quality open-source software for data scientists.
- It did focus on R, in particualr **RStudio IDE**, **Shiny**, and **tidyverse**.
- *"Posit is not about pivoting from R to Python. It's about broadening and embracing the Python community as well as the R community."*, Hadley Wickham Chief Scientist, Posit
- More about the change `https://posit.co/blog/rstudio-is-now-posit/`

RStudio IDE


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Untitled1
Source on Save
Run
Source
Write code here
Environment
History
Import Dataset
Global Environment
Environment is empty
Track activity here
Console
R version 3.3.1 (2016-06-21) -- "Bug in Your Hair"
Copyright (C) 2016 The R Foundation for Statistical Computing
Platform: x86_64-pc-linux-gnu (64-bit)
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.
Natural language support but running in an English locale
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.
Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.
Run code here
Files
Plots
Packages
Help
Viewer
Install
Update
Name
Description
Version
User Library
a4Base Automated Affymetrix Array Analysis Base Package 1.20.0
a4Core Automated Affymetrix Array Analysis Core Package 1.20.0
a4Extra Extensions of the a4 Suite of Packages 0.9-10
a4Preproc Automated Affymetrix Array Analysis Preprocessing Package 1.20.0
a4Reporting Automated Affymetrix Array Analysis Reporting Package 1.20.0
abind
acepack
ADGofTest
adveqReportR
affy
affyio
ALL
annaffy
annotate Annotation for microarrays 1.50.0
AnnotationDbi Annotation Database Interface 1.34.4
AnnotationHub Client to access AnnotationHub resources 2.4.2
Manage files & packages; view plots & documents here

RStudio Features

Features provided by RStudio include:

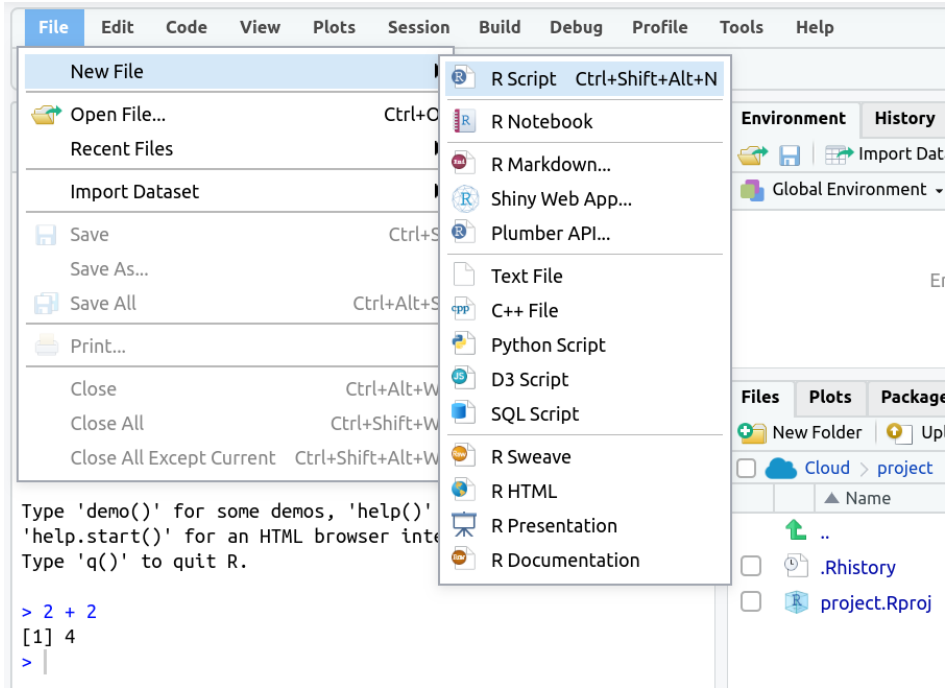
- syntax highlighting, code completion, smart indentation
- interactively send code chunks from editor to R
- organise multiple scripts, help files, plots
- search code and help files

RStudio Shortcuts from the R Console

RStudio provides a few shortcuts to help write code in the R console go to `Help - Keyboard Shortcuts Help`

R Scripts

Rather than typing commands individually in the console window, it is often more useful to keep a record of everything you have run.
You can store commands in the R Script in the source window and run these en bloc or line by line.



Text files saved with a `.R` suffix are recognised as R code.

More R Commands

```
data() # find out what standard data sets there are
plot(iris) # plot Fisher's iris data
head(iris, 4) # print the first 4 rows of the iris data
View(iris) # view the iris dataset on the viewer
summary(iris,  # summaries of the iris dataset up to the 2nd digit
        digits = 2) # you can split the command in two lines
sum(3, 4) # do the sum of 3 and 4
log(sum(1:10)) # Sum the numbers from 1 to 10 and then takes the natural log
```

Data Structures

Data structures are the building blocks of code. In R there are four main types of structure:

- vectors and factors
- matrices and arrays
- lists
- data frames

Vectors

A single number is a special case of a numeric vector. Vectors of length greater than one can be created using the concatenate function, `c`.

```
x <- c(1, 3, 6)
```

The elements of the vector must be of the same type: common types are numeric, character and logical

```
y <- c("red", "yellow", "green")
z <- c(TRUE, FALSE)
```

Missing values (of any type) are represented by the symbol `NA`.

Data Frames

Data sets are stored in R as **data frames**. These are structured as a list of objects, typically vectors, of the same length

```
str(iris)
```

```
# 'data.frame': 150 obs. of  5 variables:
#  $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ..
#  $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1
#  $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.
#  $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.
#  $ Species     : Factor w/ 3 levels "setosa","versicolor",..
```

Here `Species` is a factor, a special data structure for categorial variables.

Introduction to functions

We can create the function oddcount which counts the number of odd integers in a vector:

```
oddcount <- function(x) { # x is the input
  k <- 0 # Set k to be 0
  for(n in x) {
    # %% finds remainder on division
    if(n %% 2 == 1) k <- k + 1
    }
  return(k) # k is the output
  }
```

What answers do you get when you run the following commands?

```
oddcount(c(1,3,5))
oddcount(c(1,2,3,7,9))
```

Prompt > and Continuation prompt +

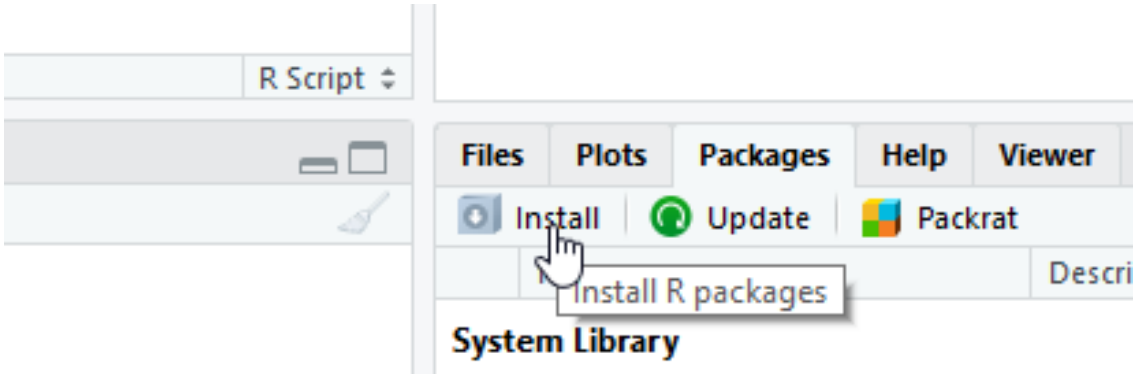
- When the symbol > appears at the start of a line is called `prompt`.
  - It appears when R is ready to receive a command
- When the symbol + appears at the start of a line is called **continuation prompt**.
  - It appears when the expression is written in multiple lines.
  - If it appears inadvertently it is possible to stop it either completing the command, or pressing ESC (Windows and Mac) or Ctrl-C (Unix).

Install Packages

Most day-to-day work will require at least one contributed package. CRAN packages can be installed by using the function `install.packages`. For example:

```
install.packages("ggplot2")
```

Or from the **Packages** tab:



Load the package

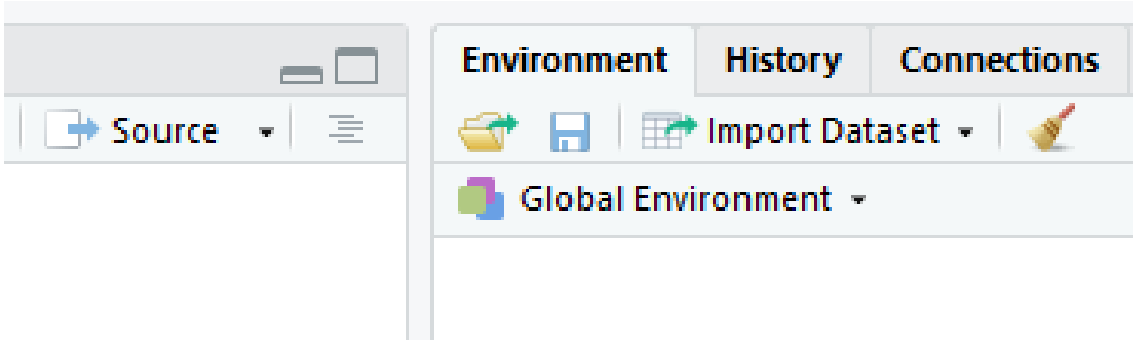
To use an installed package in your code, you must first load it from your package library.

```
library(ggplot2)
```

Sometimes an RStudio feature will require a contributed package. A pop-up will ask permission to install the package the first time, after that RStudio will load it automatically.

Data Input via Import Dataset

Using the **Import Dataset** dialog in RStudio



we can import files stored locally or online in the following formats:

- `.txt`/`.csv` via `read_delim`/`read_csv` from **readr**.
- `.xlsx` via `read_excel` from **readxl**.
- `.sav`/`.por`, `.sas7bdat` and `.dta` via `read_spss`, `read_sas` and `read_stata` respectively from **haven**.

Most of these functions also allow files to be compressed, e.g. as `.zip`.

Tibbles

The functions used by *Import Dataset* return data frames of class `"tbl_df"`, aka **tibbles**. The main differences are:

	data.frame	**tibble**
Printing (default)	Whole table	10 rows; columns to fit Prints column type
Subsetting	`dat[, 1]`, `dat$X1`, `dat[[1]]` all return vector	`dat[,1]` returns tibble `dat$X1`, `dat[[1]]` return vector
Strings	Converted to factor (default)	Left as character
Variable names	Made *syntactically valid* e.g. `Full name` -> `Full.name`	Left as is use e.g. `` dat$`Full name` ``

17

Data Input via Code

The **rio** package provides a common interface to the functions used by *Import Dataset* as well as many others.

The data format is automatically recognised from the file extension. To read the data in as a tibble, we use the `setclass` argument.

```
library(rio)
compsci <- import("compsci.csv", setclass = "tibble")
cyclist <- import("cyclist.xlsx", setclass = "tibble")
```

See `?rio` for the underlying functions used for each format and the corresponding optional arguments, e.g. the skip argument to `read_excel` to skip a certain number of rows.

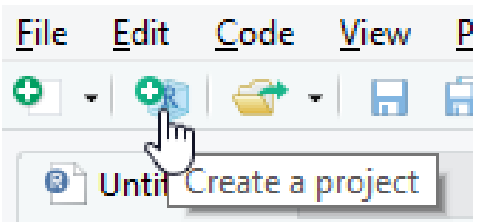
RStudio Projects

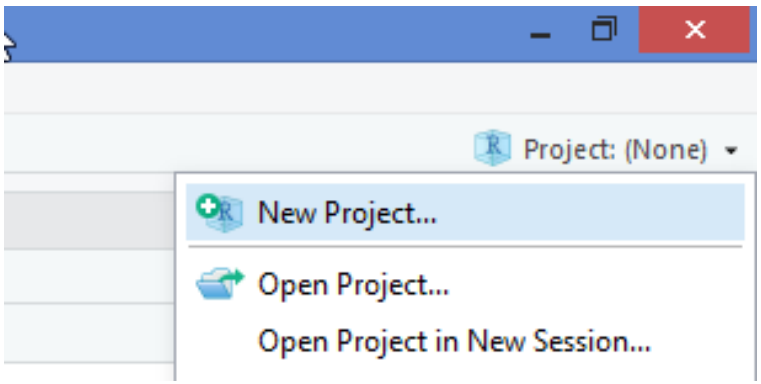
An RStudio project is a context for work on a specific project

- automatically sets working directory to project root
- has separate workspace and command history
- works well with version control via git or svn

Create a project from a new/existing directory via the *File* Menu or the *New Project* button.

Switch project, or open a different project in a new RStudio instance via the Project menu.





R Markdown Documents

R markdown documents (`.Rmd`) intersperse code chunks (R, Python, Julia, C++, SQL) with markdown text

YAML header

```
---
title: "Report"
output: html_document
---
```

Markdown text

```
## First section

This report summarises the `cars` dataset.
```

R code chunk

````
```{r summary-cars}
summary(cars)
```
````

Options can be controlled on a document or chunk level whether to show code and/or output.

Rendering

The `.Rmd` file can be rendered to produce a document (HTML, PDF, docx) integrating the code output.

Report

First section

This report summarises the `cars` dataset.

```
summary(cars)
```

```
##      speed           dist
##  Min.   : 4.0   Min.   :  2.00
##  1st Qu.:12.0   1st Qu.: 26.00
##  Median :15.0   Median : 36.00
##  Mean   :15.4   Mean   : 42.98
##  3rd Qu.:19.0   3rd Qu.: 56.00
##  Max.   :25.0   Max.   :120.00
```

Getting Started with R Markdown

1. To create a new R Markdown document; go to *File - New File - R Markdown*
   - The first time you use R Markdown on your machine you may be asked to install some R packages; if so, press `Yes`.
2. Select the type of output document you want to create.
   - You are able to produce `HTML` output on any computer.
   - To produce `pdf` output you need to have LaTeX installed.

   ```
   install.packages("tinytex")
   tinytex::install_tinytex()
   ```

3. Open the "Markdown Quick Reference": *Help - Markdown Quick Reference*

   R Markdown is a very powerful tool; an extended guide with tutorials is available on the RMarkdown website:
   `https://rmarkdown.rstudio.com/lesson-1.html`

Your Turn

1. Open `infant.Rproj`, which is an RStudio project file.
2. From the Files tab, open the `infant.Rmd` R markdown file.
3. In the chunk labelled `import-data`, write some code that will import the `infant.xlsx` file and create a tibble named `infant`.
4. Load any required packages in the `setup` chunk.
5. Run the code in the `import-data` chunk (the `setup` chunk is run automatically).
6. Use `View()` in the console to inspect the result.
7. Install the **skimr** package and use the `skim` function to summarise the data set.

Learning more/getting support

- Posit/RStudio cheatsheets (`rmarkdown`, `dplyr`, `ggplot2`) `https://posit.co/resources/cheatsheets/`
- R for Data Science (1st edition) (data handling, basic programming and modelling, R markdown) `https://r4ds.had.co.nz/`
- R for Data Science (Work in Progress 2nd edition) (data handling, basic programming and modelling, R markdown) `https://r4ds.hadley.nz/`