
# Welcome to R

Miriam Lerma

April 2023

# Index

# Today

## Your profile

- Interest in learning R (or other ways to use it).

## Goals of today

- Identify the parts of RStudio
- Be able to load and download data
- Feel comfortable using R
- Hopefully we will cover all the topics

## Pauses and questions

- There will be several pauses
- You can stop me to ask questions or
- Questions can be made at this link too

# References

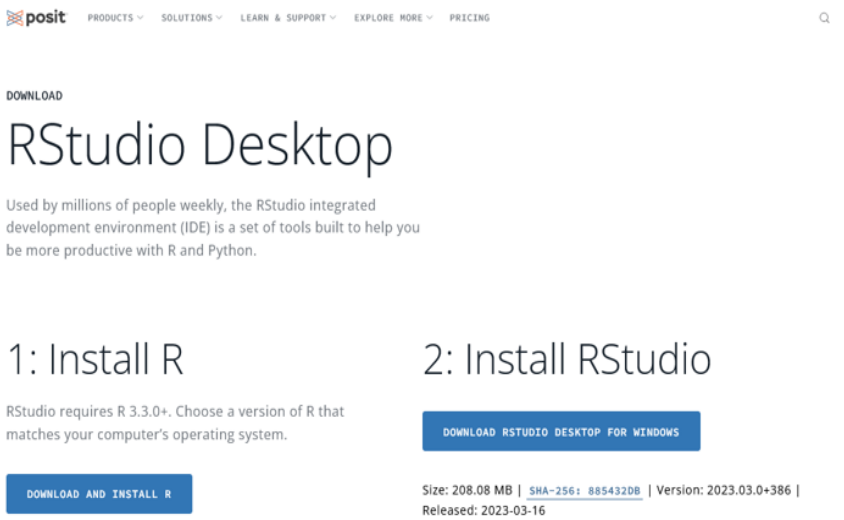
- R for Data Science
  R4DS
- Data Carpentries
  Carpentries
- From Zero to Shero by RLadies
  Zero to Hero
- Images from
  Unsplash
  Allison horst
- Other resources
  YaRrr
  Data structure
  Assignment operators
  R inferno
  Why they are not the same
  TidyTuesday


Lets begging!

# 1. Install R

## First time installing R

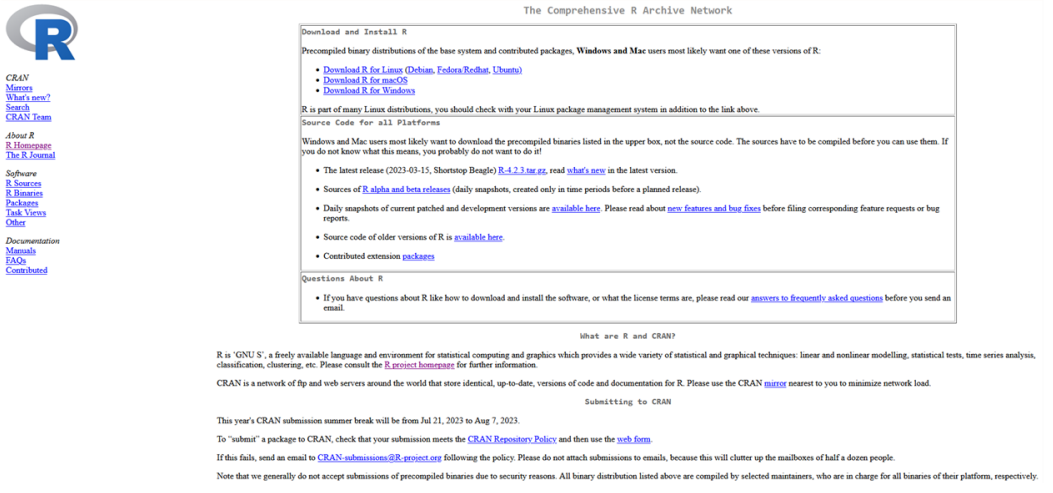
## To install R and Rstudio you can visit the page posit



# First time installing R

Once you click download and install R, you will find several links.

If you are the first time installing you likely need to choose the first link.



CRAN
Mirrors
What's new?
Search
CRAN Team

About R
R Homepage
The R Journal

Software
R Sources
R Binaries
Packages
Task Views
Other

Documentation
Manuals
FAQs
Contributed

The Comprehensive R Archive Network

Download and Install R

Precompiled binary distributions of the base system and contributed packages, **Windows and Mac** users most likely want one of these versions of R:

- Download R for Linux (Debian, Fedora/Redhat, Ubuntu)
- Download R for macOS
- Download R for Windows

R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

Source Code for all Platforms

Windows and Mac users most likely want to download the precompiled binaries listed in the upper box, not the source code. The sources have to be compiled before you can use them. If you do not know what this means, you probably do not want to do it!

- The latest release (2023-03-15, Shortstop Beagle) R-4.2.3.tar.gz, read what's new in the latest version.
- Sources of R alpha and beta releases (daily snapshots, created only in time periods before a planned release).
- Daily snapshots of current patched and development versions are available here. Please read about new features and bug fixes before filing corresponding feature requests or bug reports.
- Source code of older versions of R is available here.
- Contributed extension packages

Questions About R

- If you have questions about R like how to download and install the software, or what the license terms are, please read our answers to frequently asked questions before you send an email.

What are R and CRAN?

R is 'GNU S', a freely available language and environment for statistical computing and graphics which provides a wide variety of statistical and graphical techniques: linear and nonlinear modelling, statistical tests, time series analysis, classification, clustering, etc. Please consult the R project homepage for further information.

CRAN is a network of ftp and web servers around the world that store identical, up-to-date, versions of code and documentation for R. Please use the CRAN mirror nearest to you to minimize network load.

Submitting to CRAN

This year's CRAN submission summer break will be from Jul 21, 2023 to Aug 7, 2023.

To "submit" a package to CRAN, check that your submission meets the CRAN Repository Policy and then use the web form.

If this fails, send an email to CRAN-submissions@R-project.org following the policy. Please do not attach submissions to emails, because this will clutter up the mailboxes of half a dozen people.

Note that we generally do not accept submissions of precompiled binaries due to security reasons. All binary distribution listed above are compiled by selected maintainers, who are in charge for all binaries of their platform, respectively.

# Update R

If you already have R and R-Studio

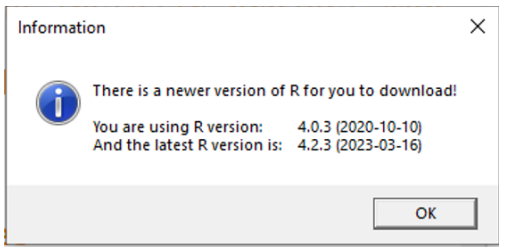
Check which version of R you are using:

```
sessionInfo()
```

To see if you need to update you can use:

```
install.packages("installr")
library(installr)
updateR()
```

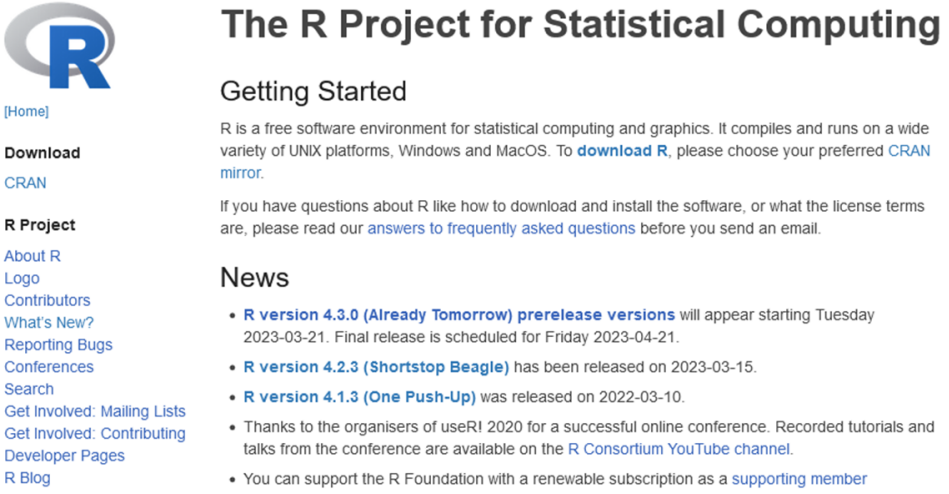
## Do you need an update?



# Update R

If you opt to update R, one way to do it is going to the installation page

You will likely select >4.3 (in 2023)



[Home]

**Download**

CRAN

**R Project**

About R
Logo
Contributors
What's New?
Reporting Bugs
Conferences
Search
Get Involved: Mailing Lists
Get Involved: Contributing
Developer Pages
R Blog

## The R Project for Statistical Computing

### Getting Started

R is a free software environment for statistical computing and graphics. It compiles and runs on a wide variety of UNIX platforms, Windows and MacOS. To **download R**, please choose your preferred CRAN mirror.

If you have questions about R like how to download and install the software, or what the license terms are, please read our answers to frequently asked questions before you send an email.

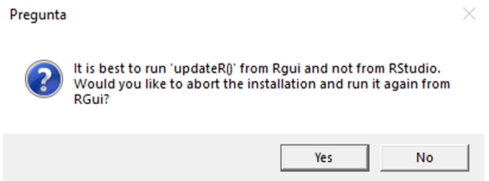
### News

- **R version 4.3.0 (Already Tomorrow) prerelease versions** will appear starting Tuesday 2023-03-21. Final release is scheduled for Friday 2023-04-21.
- **R version 4.2.3 (Shortstop Beagle)** has been released on 2023-03-15.
- **R version 4.1.3 (One Push-Up)** was released on 2022-03-10.
- Thanks to the organisers of useR! 2020 for a successful online conference. Recorded tutorials and talks from the conference are available on the R Consortium YouTube channel.
- You can support the R Foundation with a renewable subscription as a supporting member

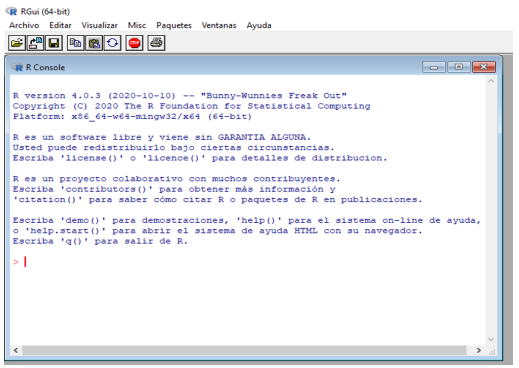
# Update R

If you decide to update your R, its better to do it in the Rgui and not R-Studio

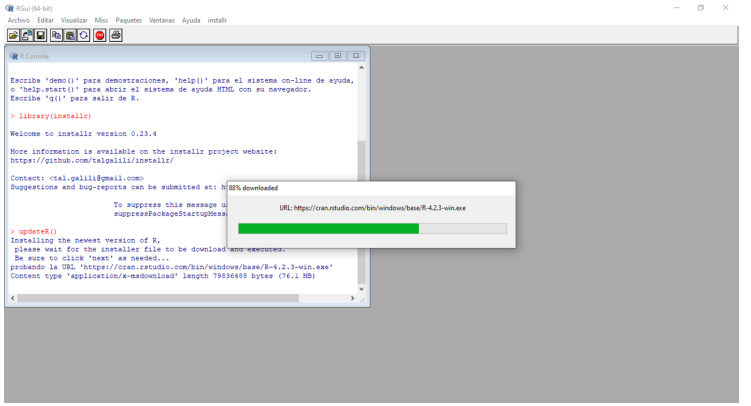
## Open Rgui




## Once in the Rgui, write updateR()

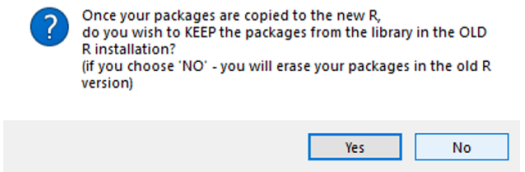



## Give it some time



# Update R

Select YES if you already had R packages installed in your previous version, and do not want to install each package one by one again.




By March 2023 4.2.3 was the newest version of R.

```
R version 4.2.3 (2023-03-15 ucrt) -- "Shortstop Beagle"
Copyright (C) 2023 The R Foundation for Statistical Compu
ting
Platform: x86_64-w64-mingw32/x64 (64-bit)
```

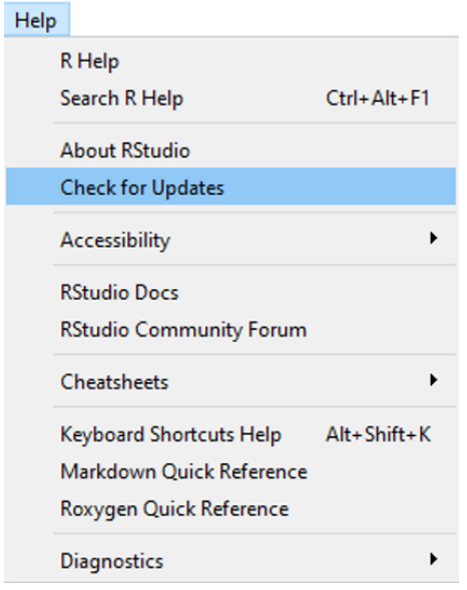
**Did you notice the difference?**

# Update R-Studio

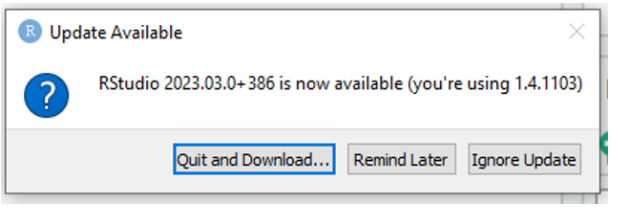
One thing is updating R and another one to update R-Studio

If you already have Rstudio and want to see if you need to update you can go to:

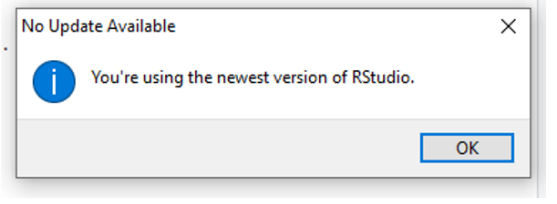
## Help>Check for updates

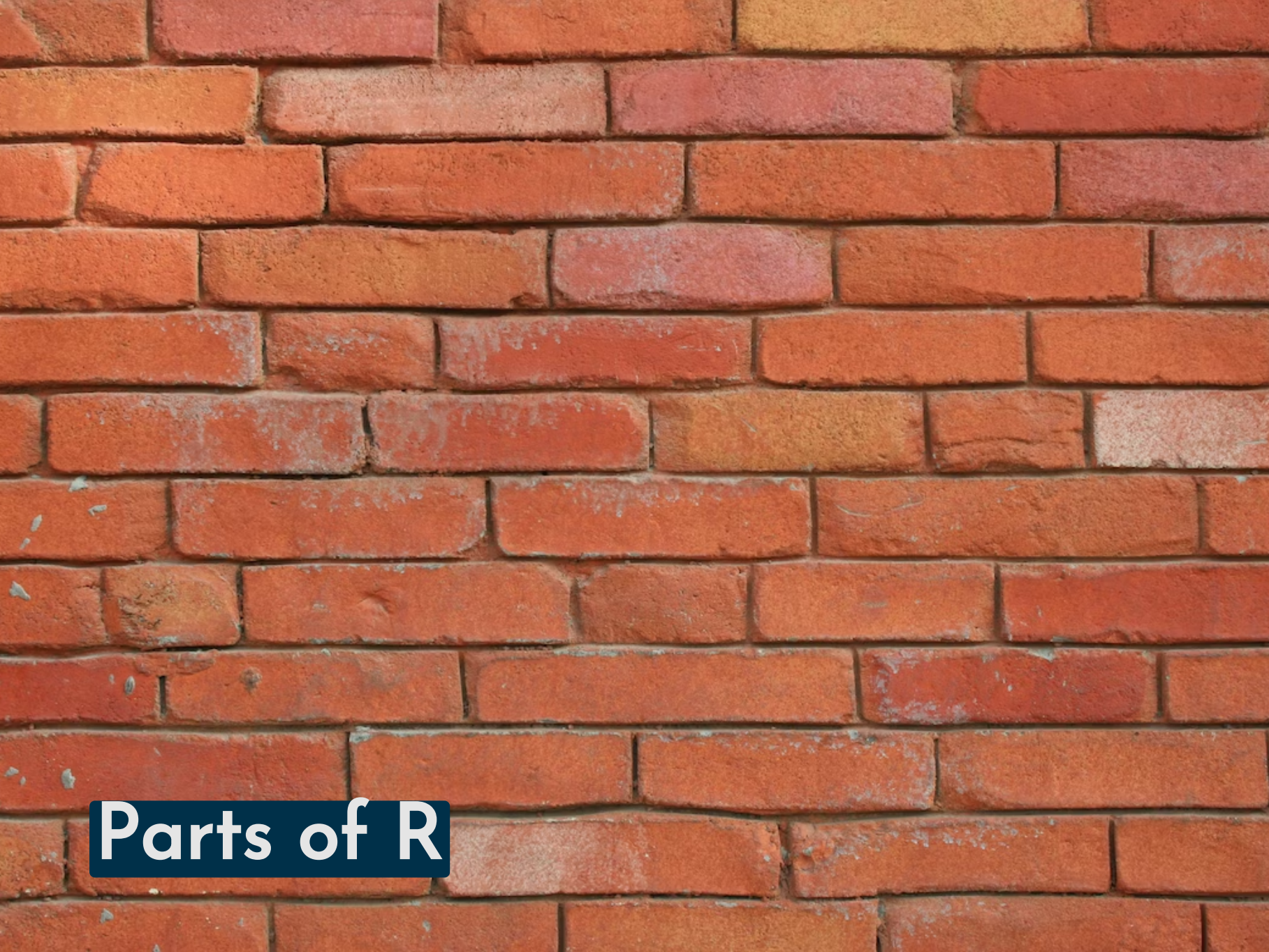



## Outdated



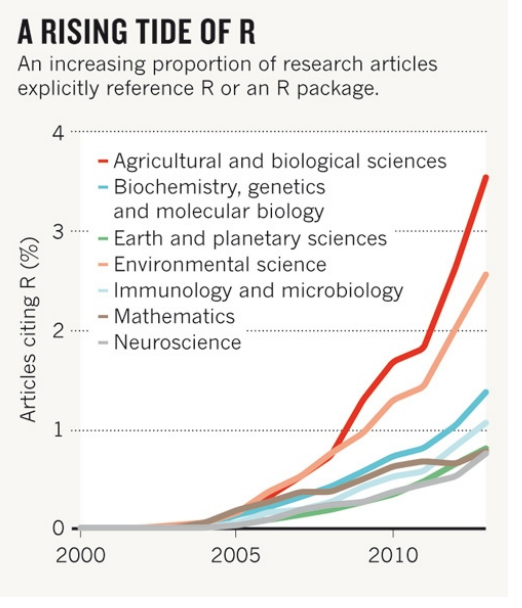

## Up to date




Parts of R

# 2. What is R?

R is a programming language developed for statistical analyses by the Ross Ihaka and Robert Gentleman (New Zealand) in 1996.




Sylvia Tippmann

# 2.2 Why R?

## Advantages

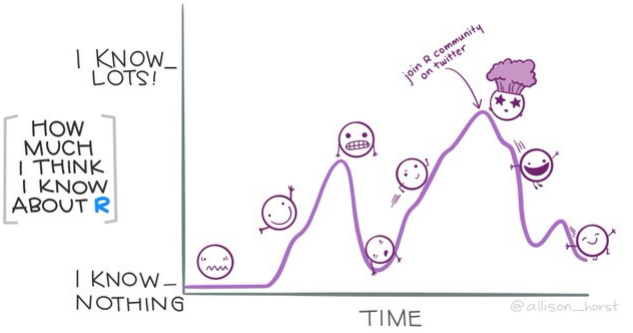
- Its free
- A lot of tools
- Flexible data visualization

## For science, is reproducible

- You can save all the steps you followed, making it verifiable
- Instead of clicking on buttons the instructions can be save in sequence, helping save time and reduce errors

## Disadvantages

- Has a learning curve
- It is like learning a new language

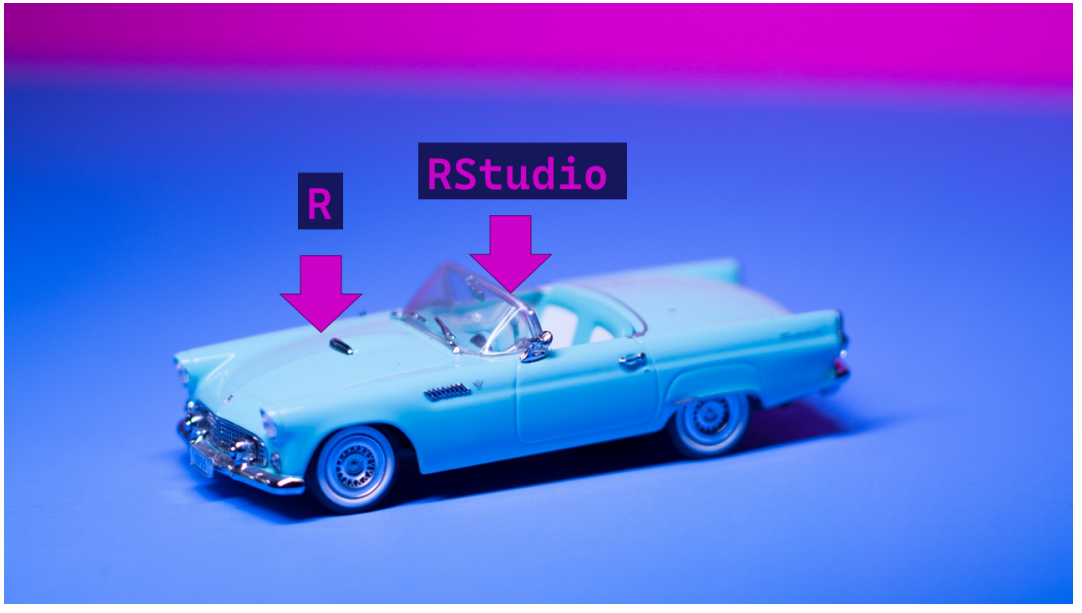


# 2.3. R vs RStudio.

What are the differences between R and RStudio?

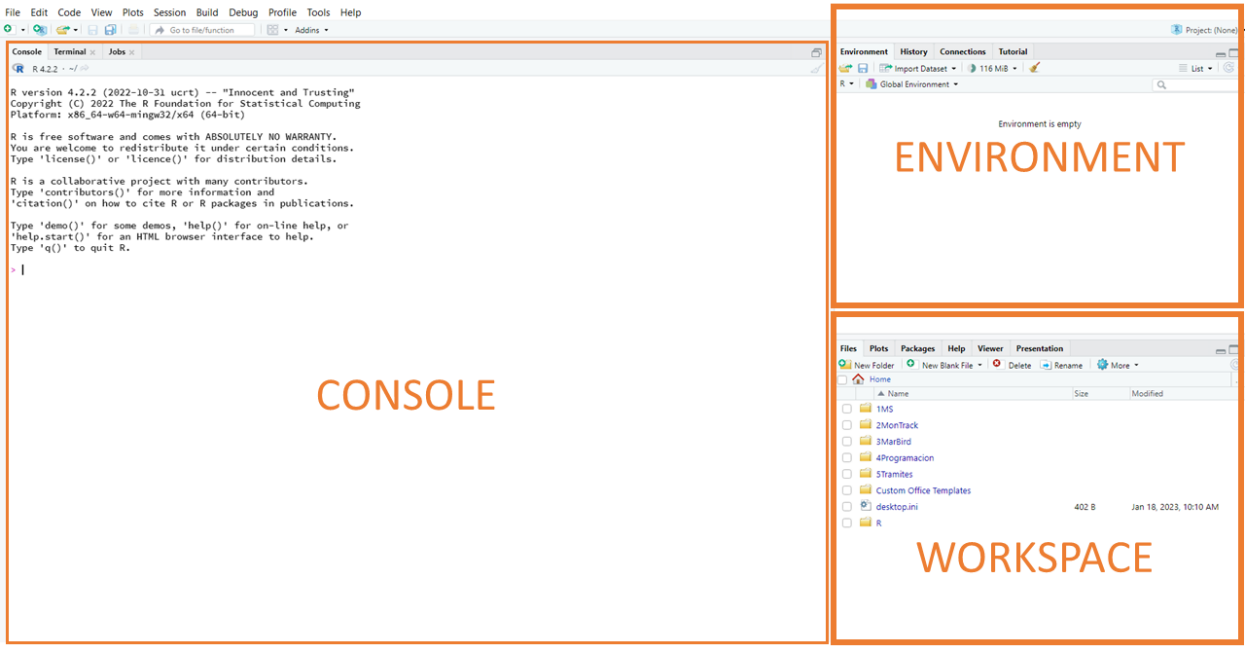
**R** is the programming language.
**RStudio** is an IDE: Integrated Development Environment.

R is the engine and RStudio is the steering wheel



# 2.4 Using R

First time opening RStudio looks like this:

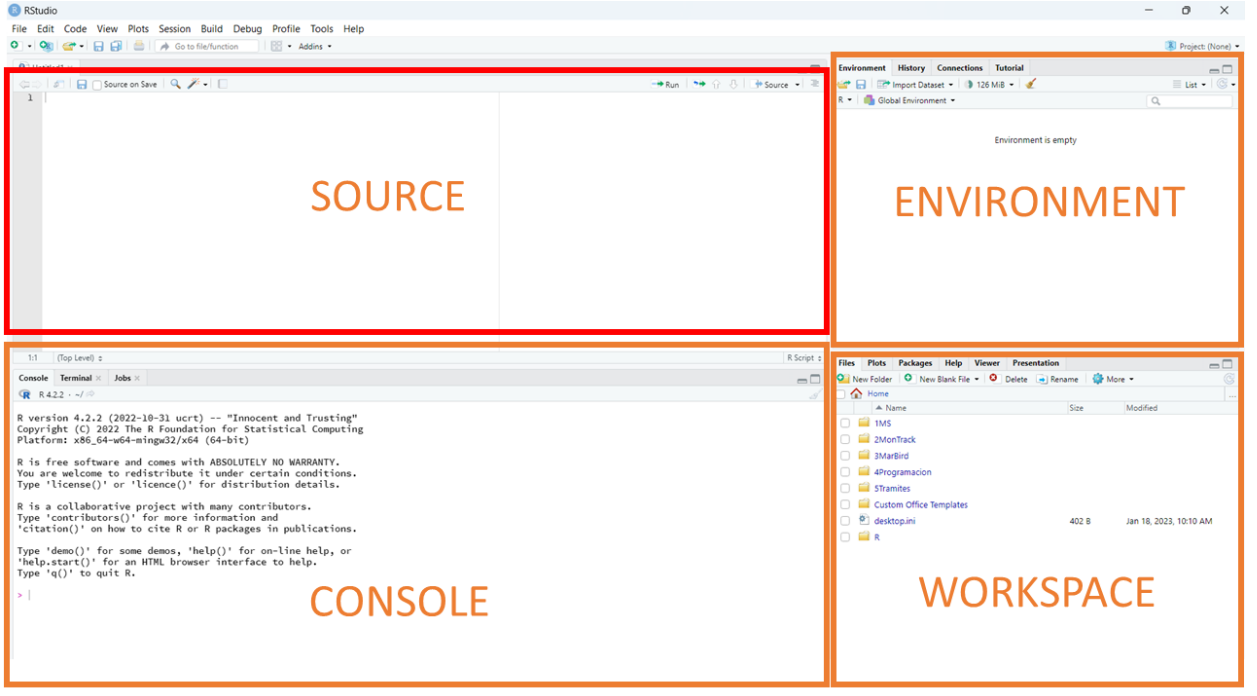



To have a code editor:

Open:
File > New File > Rscript

# 2.4. Using RStudio.

In the **source** section the code can be edited and saved.



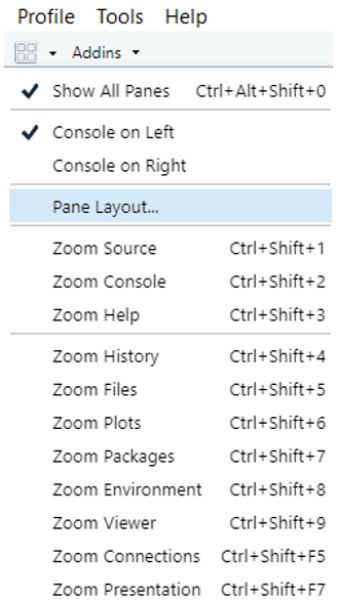
# 2.4. Using RStudio.

To change the order of the panes, you can change the pane layout

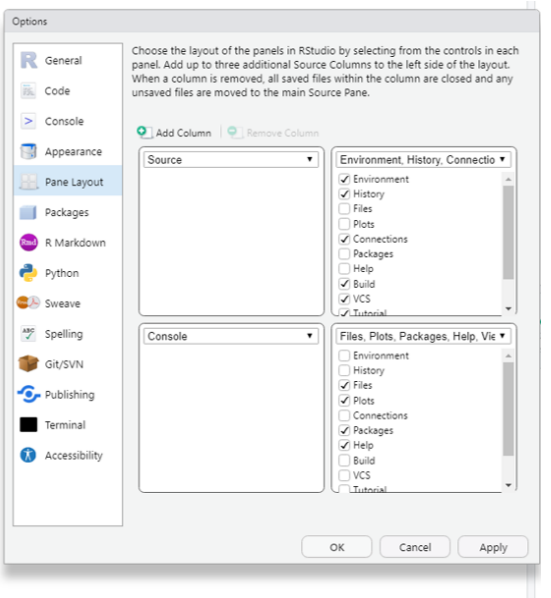
Click on the squares >

Or in Tools> Global Options

select **Pane Layout**




Select your preferred order



# 2.4 Using RStudio

Testing the console

R is like a calculator

- Write in the console

```
3*4
```

- Write in the editor

```
3*4
```

In the editor: ctrl + enter

- Now write in the editor, dont forget to click ctrl+enter

```
Result<-3*4
```

What happend in the **environment**?

# 2.5. Data types

Some data types:

- Numeric

```
class(1.4)
```

```
[1] "numeric"
```

- Characters

```
class("Miriam")
```

```
[1] "character"
```

# 2.5. Data types

Why if we use quotes ""?

```
class("1")
```

```
[1] "character"
```

- Logical

```
3 == 4
```

```
## [1] FALSE
```

```
3 < 5
```

```
## [1] TRUE
```

There are many others, that would be relevant later on.

Source: Data structures by the Carpentries

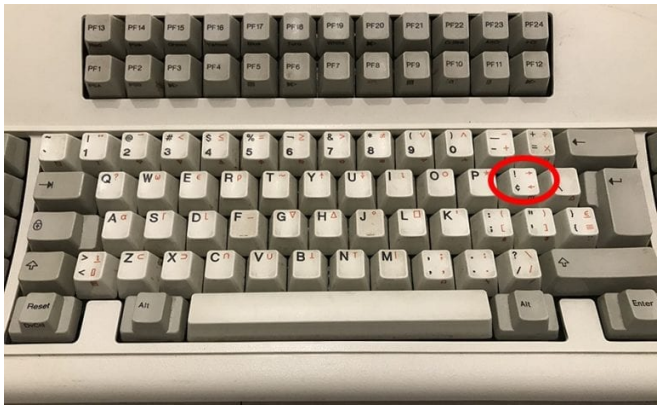
# 2.6. Data frame

## Assign symbol

```
<-
```

This symbol is frequently use, and is important that you find it in your keyboard as soon as possible.

- The equal = might behave differently than the assign symbol
  See here: Assignment operators, R inferno, Why they are not the same



# 2.6. Data frame

Lets use the arrow symbol.

```
Names<-c("Miriam","Jette","Kai","Nele","Jana","Kristin","Karena","Nico",
class(Names)
```

```
## [1] "character"
```

```
Names<-as.data.frame(Names)
class(Names)
```

```
## [1] "data.frame"
```

Now in the **environment** click twice and look at your data frame.

# 2.6. Data frame

Assign vectors to a column or check a column contents

The symbol $ is in R

```
$
```

Lets create a new column

```
Names$LastName<-c("G","A","T","D","L","S","H",NA,"M","L")
```

```
head(Names,5)
```

```
   Names LastName
1 Miriam        G
2  Jette        A
3    Kai        T
4   Nele        D
5   Jana        L
```

# 2.7. Force classes

## Force column type

```
my_number<-'1'
```

```
class(my_number)
```

```
## [1] "character"
```

```
my_number<-as.numeric(my_number)
```

```
class(my_number)
```

```
## [1] "numeric"
```

# 2.7. Force classes

## Our favorite class

```
my_timestamp<-'2015-06-12 05:00:00'
```

```
class(my_timestamp)
```

```
## [1] "character"
```

```
my_timestamp <- as.POSIXct(strptime(my_timestamp, "%d/%m/%Y %H:%M:%S "),
```

```
class(my_timestamp)
```

```
## [1] "POSIXct" "POSIXt"
```

Timestamp should be a "POSIXct" "POSIXt" for R to understand the are date and time

# 2.8. Code editor

To add text in the **editor** we use the symbol **#** (hashtag)

```
#Here you created a data frame with names, but there is NA
```

R will ignore the text that has a # but these are very important for us to make annotations.

It is a **good practice** to write comments. Either for us to share the script, or for you to remember what you were doing.

This is specially important if you get corrections, or if you want to repeat the analyses.


Errors

# 2.9. Type of errors

Errors are part of life.

R very often is trying to tell us how to solve the error.

- For example:

```
3 = 4
```

Error in 3 = 4 :left-hand side to assignment (do_set)

Other messages that might be red or orange are:

- Warnings, I will do it, but Im not responsible of what you are doing.
- Messages, hey I just want to let you know this happend.

# 2.9. The most common errors

Whats going on here

Remove the # and find the error.

```
#c("Miriam","Bety","Angel","Denise""Pamela")
```

- Missing a comma

```
#c("Alvaro","Gabriela","Juan",,"Lisset")
```

- Too many commas

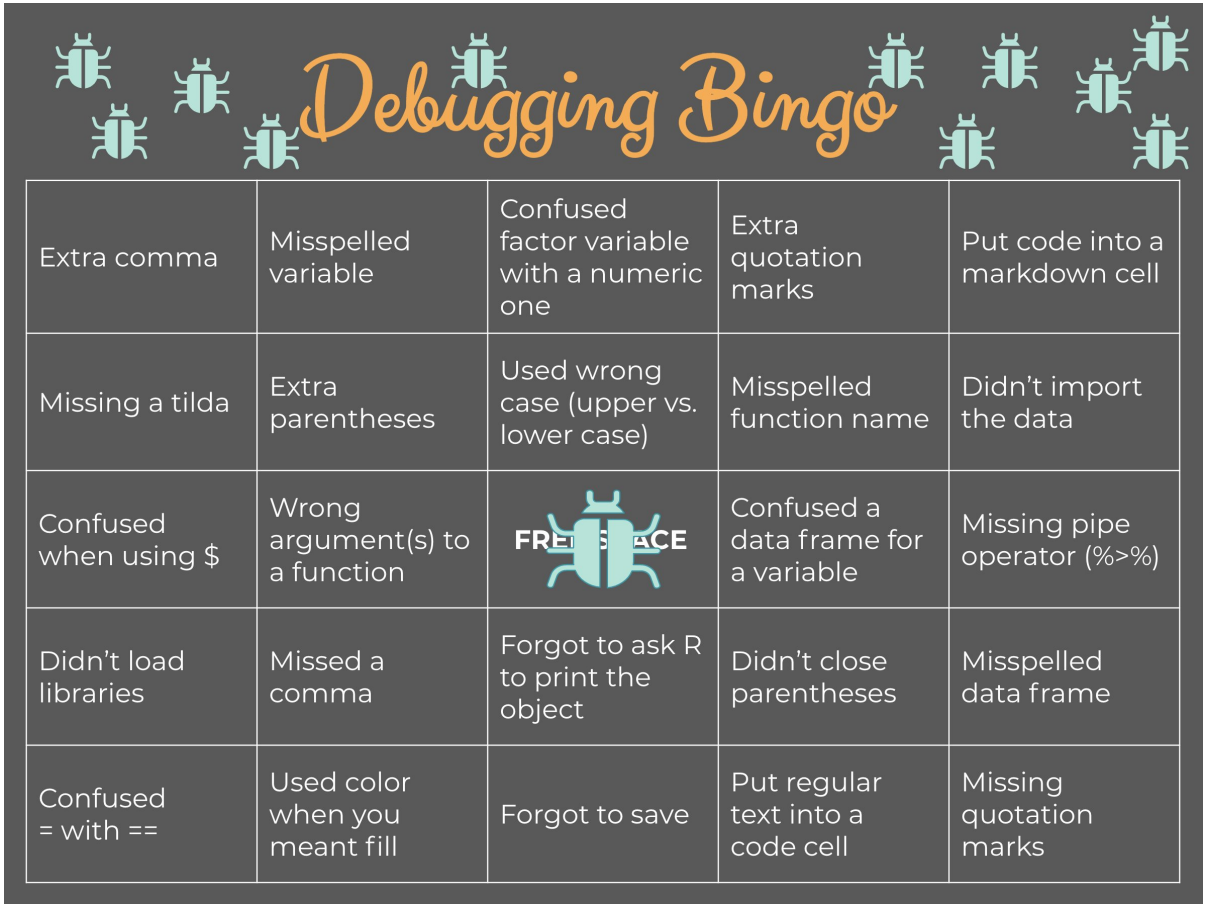
```
#c("Majo","Irma","Paulina"
```

- Missing a parenthesis

```
#c("Majo","Irma,"Paulina")
```

- Missing a quote (")

# 2.9. Debugging bingo




Source: Dr. Ji Son

# 2.10. Name consistency

Many errors are going to be marked by a red x in the **code editor**.

But many will not.

What is wrong with these?

```
#name
```

- If our goal is to use of data frame from the environment it needs to be written exactly the same **Names**.

To reduce this source of error, is important to be consitent on how we name the objects.

Some considerations are:

- R will not take objects that start with a number (Eg."1Names").
- But the number can be at the end of the name (Eg."Names1").
- Try to be consistent E.g. ("my_names").
  E.g. ("my.names").

# Pause

- Open RStudio
- Write something in the console
- Create a data frame with characters and numbers

Until here:

- Index
- What is R
- RStudio and identify the parts
- Data types
- Errors

Next part:

- Packages
- Directories
- Read data
- Contact


# Packages

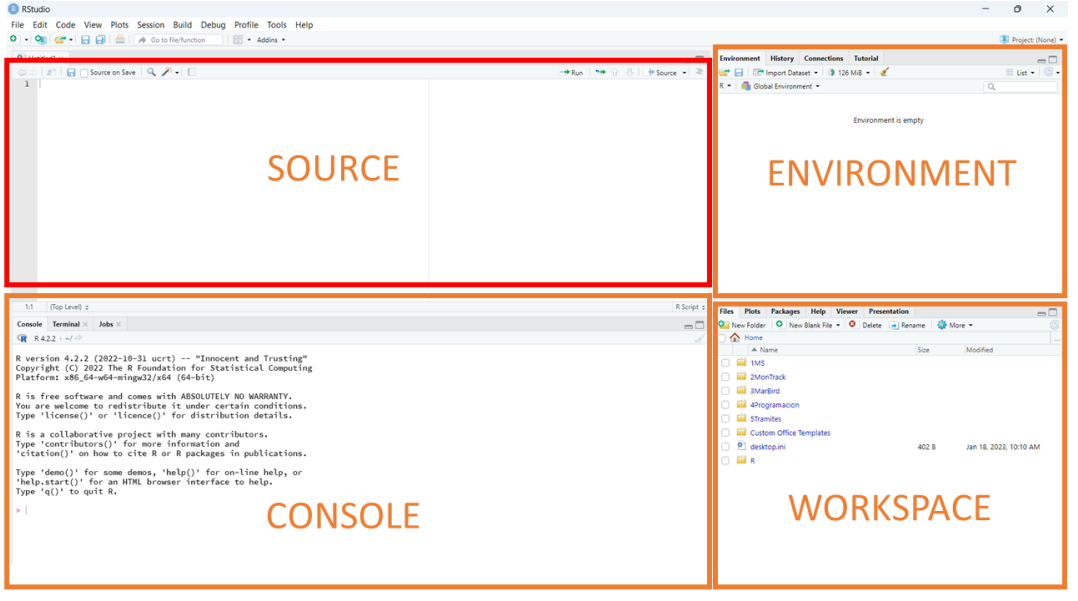
# 3.1. Packages

- Packages are a collection of functions
- Packages are installed one time, but need to be loaded every time you need them.

An analogy is that install.packages() is downloading an app, and library() is like opening the app every time you want to use it.

To install packages using the **workspace**

Go to Packages>Install and write tha name of the package



# 3.1. Packages

- To install in the **console** write: install.packages("").
  To load a package: library("").

Some packages that we will use:

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

🕐 Tidyverse is a compilation of packages, this one might take some time.

Note: It probably will give us a warning, remember that a warning might be just information. In this case is telling us what the package contains.

# 3.1. Packages

```
say(what = "Now you can download packages!", by = "cat")
```

nothing happend? We should load the library first.

# 3.1. Packages

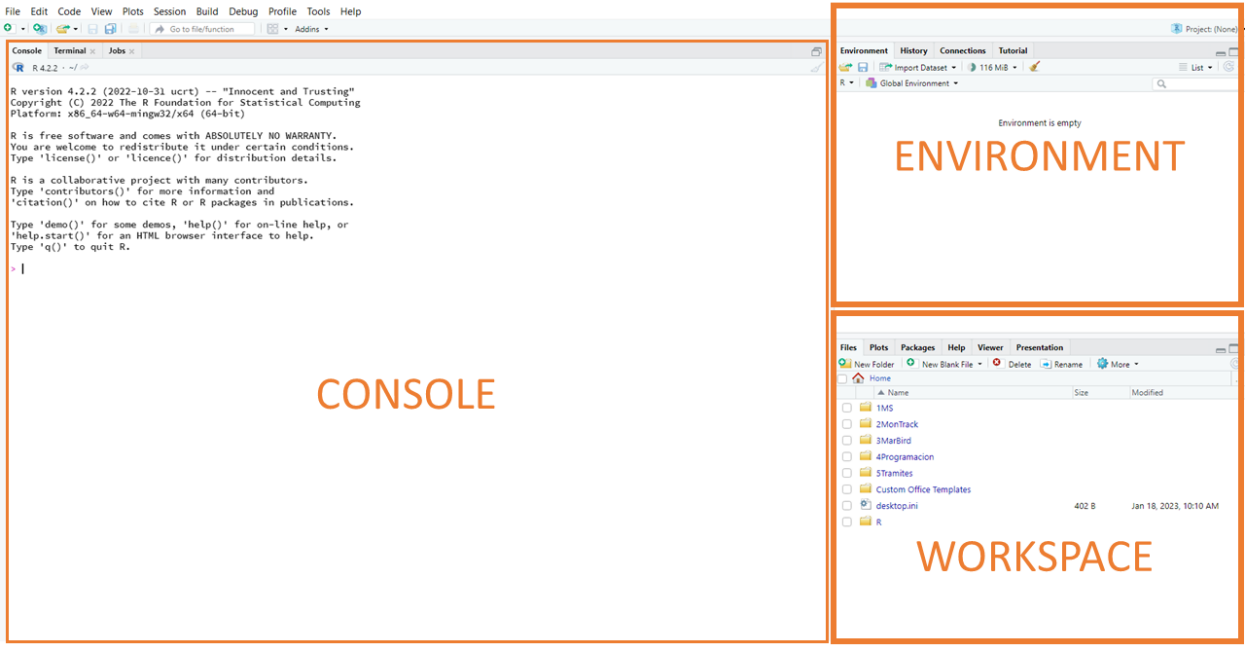
```
library("cowsay")
say(what = "Now you can download packages!", by = "cat")
```

```
 --------------
Now you can download packages!
 --------------
    \
      \
        \
            |\___/|
          ==) ^Y^ (==
            \  ^  /
             )=*=(
            /     \
            |     |
           /| | | |\
           \| | |_|/\
      jgs  //_// ___/
               \_)
```

# 3.2. Directories

Where are my files?.

By hand: Click on the **workspace** > Files > More > Set As Working directory




This is practical, but needs to be done every time that we enter R and therefore be time consuming.

# 3.2. Directories

Where I am working?

```
getwd()
```

```
setwd("paste your directory name here")
```

Note, that you will need to copy and paste the directory.

If you share your script or are working in another computer, everyone will have different directories.

# 3.2. Directories

### Where I am?

```
install.packages('here')
```

```
library(here)
```

```
here()
```

### Advantages:

- **here** allows to use the folder where we saved the script
- You can share your script and your collaborators dont need to copy and paste their directory
- If the folder name is changed or the folder is change from place it would still work

# Pause

- Open RStudio
- Install and load packages
- Check your directory

Make comments or suggestions here

Until here:

- Index
- What is R
- RStudio and identify the parts
- Data types
- Errors
- Packages
- Directories

Next part:

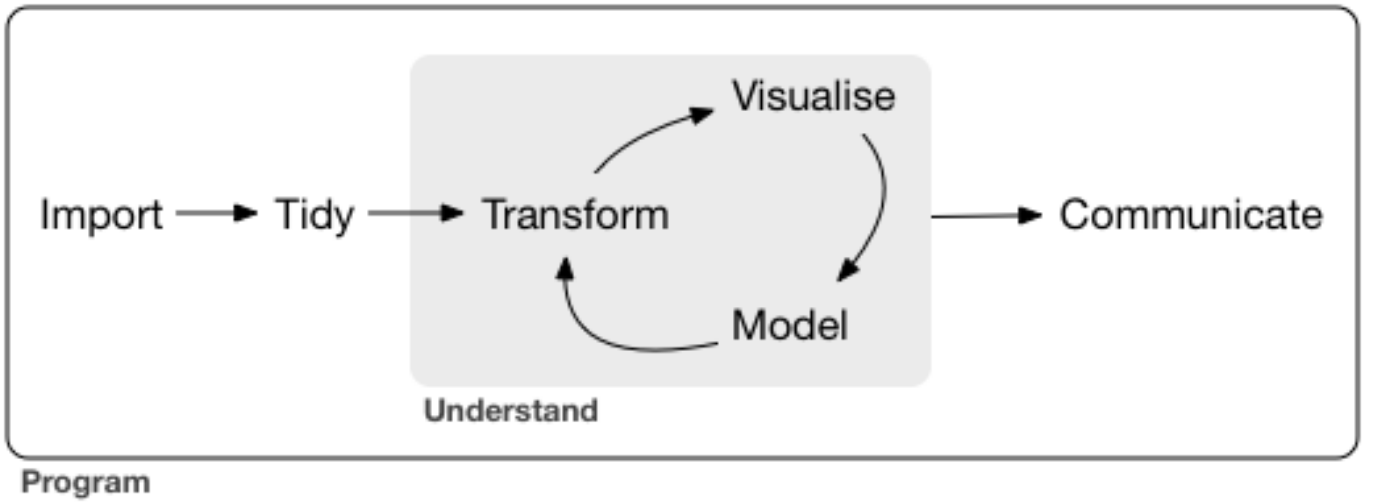
- Read data
- Contact


imation
2HD IBM
formatted formatted
1.44MB
HD
imation
12/11/03

# Read files

# 4.1. Read files

A typical R project looks like this:




Source: R4DS

# 4.1. Read files

To load data, we will use functions from the package tidyverse and the files:

- penguins1.csv
- penguins2.csv
- penguins3.txt
- penguins4.xlsx

Download here

Do you already have it installed?

```
library("tidyverse")
```

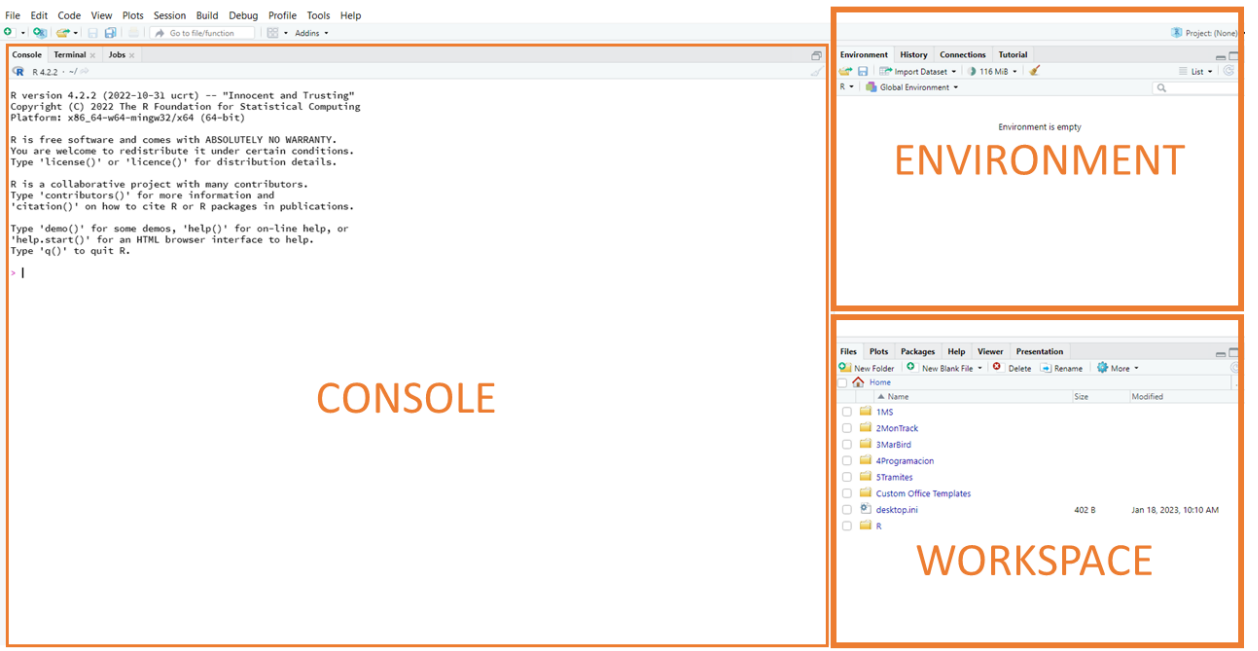
# 4.2. By hand

You can upload data by hand in your workspace

In the **environment** there is a part that says **Import Dataset**

Select the file **penguins1.csv**

Another option is to go to the **workspace** Files> Click on the file and **import data set**



# 4.2. csv format

Instead of clicks, we can write in the console or script:

```
penguins1<-read_csv("Downloads/penguins1.csv")
```

```
head(penguins1)
```

# 4.2. csv format

Now try opening **penguins2.csv**

This file instead of being separated by comas, its separated by colons ";"

Therefore, instead of using **read_csv** we will need to use **read_csv2**

To illustrate this issue, try loading the data using **read_csv**.

```
penguins2<-read_csv("Downloads/penguins2.csv")
```

```
head(penguins2)
```

Lets try now with **read_csv2**

```
penguins2<-read_csv2("Downloads/penguins2.csv")
```

```
head(penguins2)
```

# 4.3. Other formats

Click on the file **penguin3.txt**.

This one is separated by tabs.

read_tsv is for reading tab separated values.

```
penguins3<-read_tsv("Downloads/penguins3.txt")
```

```
head(penguins3)
```

# 4.4. Excel format

For loading excel data, there is a special package called **readxl**

```
library("readxl")
```

```
penguins4<- read_excel("Downloads/penguins4.xlsx")
```

```
head(penguins4)
```

# 4.5. From an url

Urls (Uniform Resource Locators) or links can also be source of data.

```
penguins5<- read_csv('https://github.com/MiriamLL/R_intro/blob/master/Do
```

Look at the first 5 rows of the data

```
head(penguins,5)
```

# 4.6. movebank

There is a package called move that can be use to access data stored in movebank.

To install:

```
install.packages('move')
```

```
library(move)
```

```
movebankLogin()
```

Add your login, if you have one, at the console.

# 4.6 movebank

It is more convenient to store your login information, but you have to be careful to not share the script with your login information.

```
loginStored <- movebankLogin(username="MiriamLerma", password="********"
```

```
my_study<-'FTZ UCN Kelp Gull Chile'
```

```
MyGull<-getMovebankLocationData(study=my_study,
                                individual_local_identifier="KEGU-noband0
                                timestamp_start="202212010000000",
                                timestamp_end="20221205000000000",
                                sensorID="GPS")
```

It might give you a warning, but for now that is not important.

# 4.7. packages with data

Data can also be stored in packages.
For example palmerpenguins

```
install.packages("palmerpenguins")
```

```
library(palmerpenguins)
penguin6<-penguins
```

```
head(penguin6)
```

# 4.7. packages with data

Moreover, packages with data are not limited to data frames.

# 4.7. packages with data

For example, the package GermanNorthSea contains shapefiles

With 6 lines of code you can plot a map (Showing just for illustration purposes)

```
# install.packages("devtools")
devtools::install_github("MiriamL
```

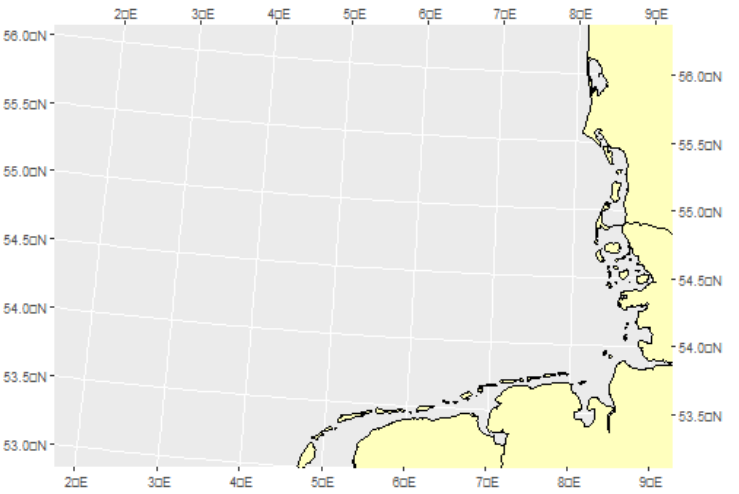
Now the package sf

```
#install.packages('sf')
library(sf)
library(ggplot2)
library(GermanNorthSea)
```

Load and plot some data

```
German_land<-GermanNorthSea::Germ
```

```
ggplot()+ geom_sf(data = German_l
  coord_sf(xlim = c(3790000,42500
                    label_axes =
```



# Pause

Load penguin data

- Using read_csv
- Using read_csv2
- Using read_tsv
- Using read_excel

There are many other options of files. Suggestions? here

Until here:

- Index
- What is R
- RStudio and identify the parts
- Data types
- Errors
- Packages
- Directories
- Read data

Next part:

- Rmd
- Contact

# Back to

- Index
- What is R
- RStudio and identify the parts
- Data types
- Errors
- Packages
- Directories
- Read data

Make comments or suggestions here

## Contact

This materials are free of use
Download the presentation here: github and webpage

Home

Index