

# Getting Started

**João Vieira & Pedro Fonseca**



**Introduction to R Programming**

April 1, 2020



- ① R and Rstudio
- ② Workflow
- ③ Packages
- ④ Settings

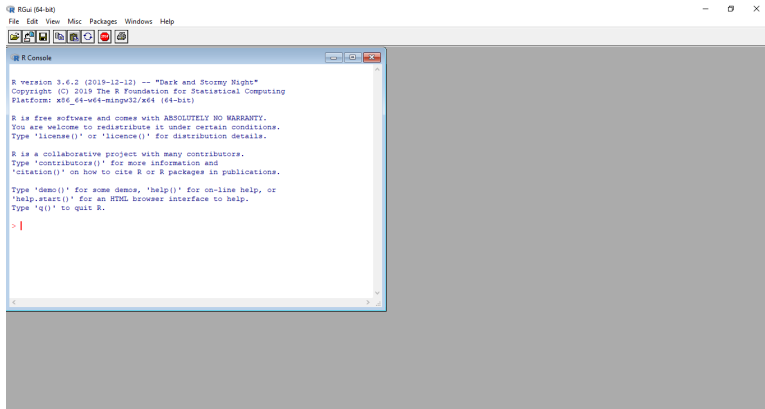
# R vs Rstudio



- R is a programming language and free software environment for statistical computing and graphics.
- RStudio is an integrated development environment (IDE) for R.
- R and Rstudio are not two different versions of the same thing.
- You can use R without using RStudio, but you can't use Rstudio without using R.

What is R?

# This is How R Looks Like



**Figure 1:** R console on windows

What is R?

# This is How R Looks Like



Figure 2: R console on MacOS

What is R?

# This is How R Looks Like



```
Terminal

R version 3.6.1 (2019-07-05) -- "Action of the Toes"
Copyright (C) 2019 The R Foundation for Statistical Computing
Platform: x86_64-pc-linux-gnu (64-bit)

R é um software livre e vem sem GARANTIA ALGUMA.
Você pode redistribuí-lo sob certas circunstâncias.
Digite 'license()' ou 'licence()' para detalhes de distribuição.

R é um projeto colaborativo com muitos contribuidores.
Digite 'contributors()' para obter mais informações e
'citation()' para saber como citar o R ou pacotes do R em publicações.

Digite 'demo()' para demonstrações, 'help()' para o sistema on-line de ajuda,
ou 'help.start()' para abrir o sistema de ajuda em HTML no seu navegador.
Digite 'q()' para sair do R.

> █
```

Figure 3: R console on Linux

What is R?

# This is How Rstudio Looks Like

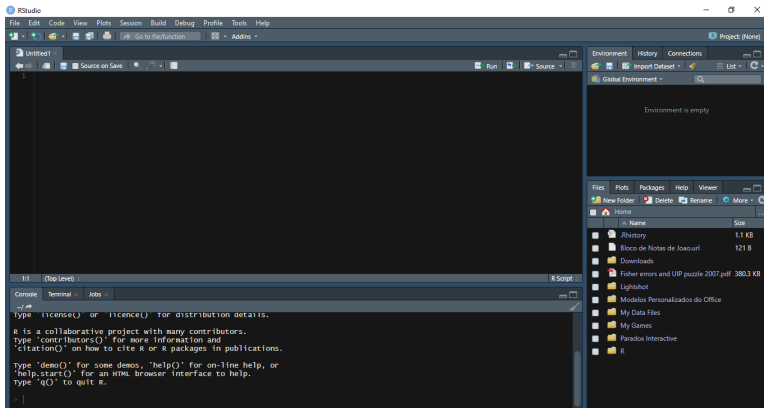


Figure 4: Rstudio on Windows and Linux

What is R?

# This is How Rstudio Looks Like

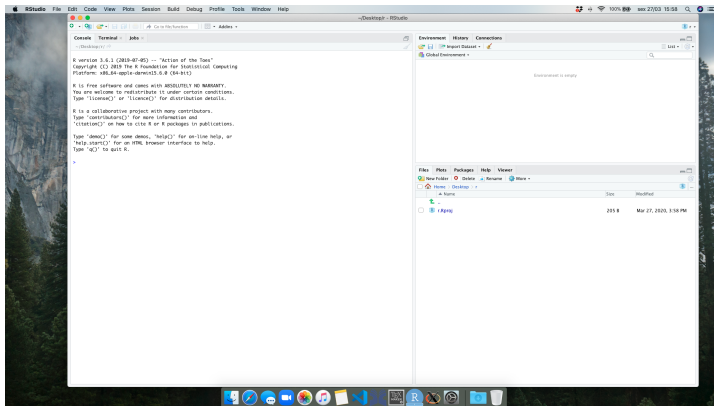


Figure 5: Rstudio on MacOS



What is R?

# Rstudio Cloud



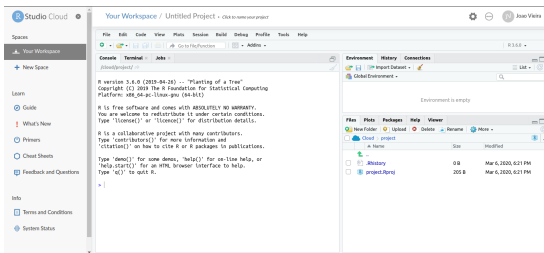
- If you don't want to install R and RStudio:
  - 1 Go to [RStudio Cloud](#).
  - 2 Create an account and login.
  - 3 Click "New Project".

What is R?

# Rstudio Cloud



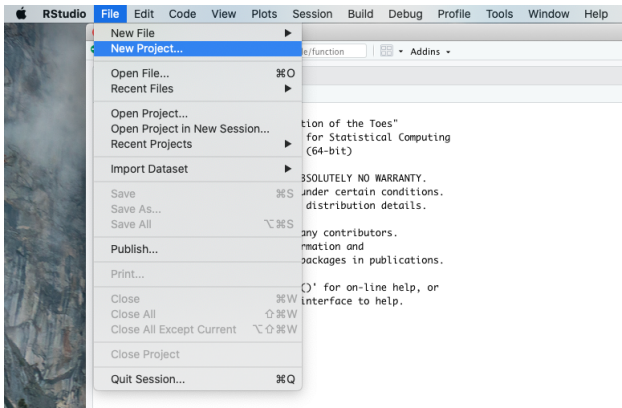
- You should then see something like this:



**Figure 6:** Rstudio Cloud

- For additional information: <https://rstudio.cloud/learn/guide>

# Your First Project



# Your First Project



## New Project

### Create Project



#### New Directory

Start a project in a brand new working directory



#### Existing Directory

Associate a project with an existing working directory



#### Version Control

Checkout a project from a version control repository



Cancel

# Your First Project



New Project

Back

Project Type

New Project >

R Package >

Create a new project in an empty directory

Shiny Web Application >

R Package using Rcpp >

R Package using RcppArmadillo >

R Package using RcppEigen >

Book Project using bookdown >


Cancel

# Your First Project



New Project

[Back](#) **Create New Project**



Directory name:

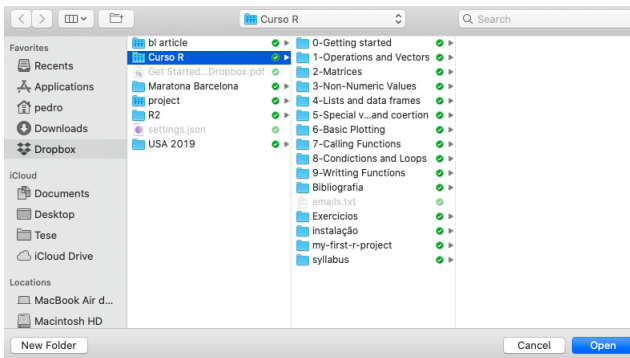
Create project as subdirectory of:  
 [Browse...](#)

☐ Create a git repository

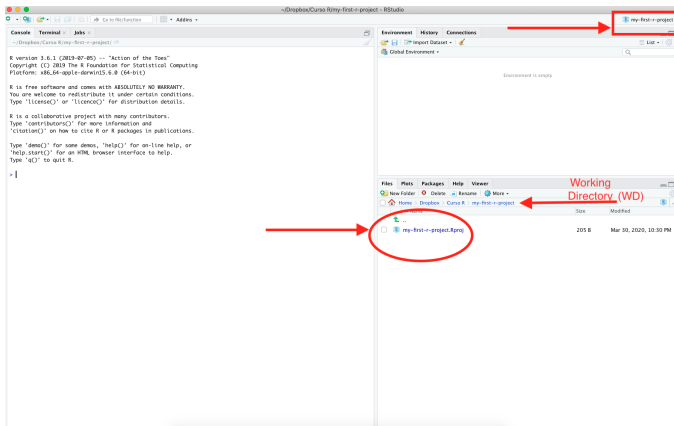
☐ Open in new session

[Create Project](#) [Cancel](#)

# Your First Project

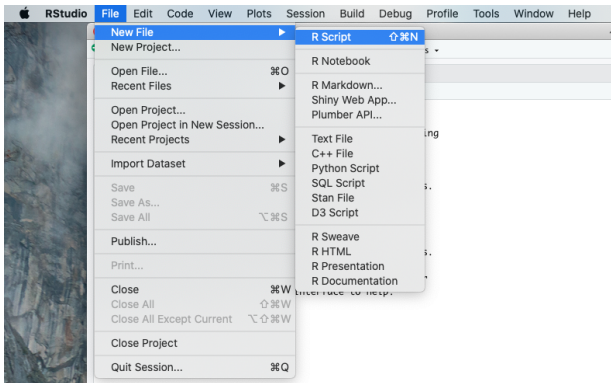


# Your First Project



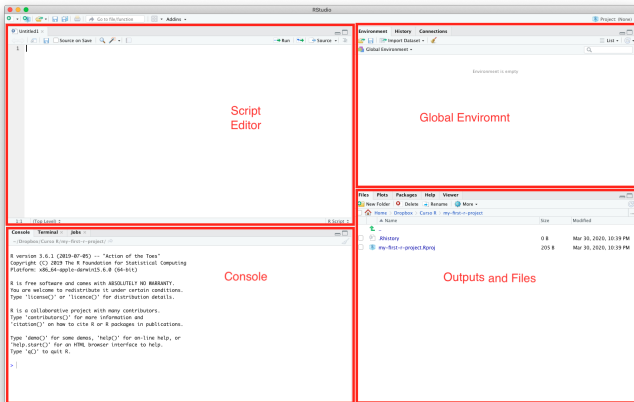


# Your First R Script



- Shortcut: Cmd+Shift+N (Mac) or Ctrl+Shift+N (Windows and Linux)

# Rstudio Panes



# Editor



- The editor is where you can edit the script that you just created. This is where you should write the code that you will learn.
- In the editor you can send text to the console, which is where your code will be run.
- In the editor you can modify, rerun and save your code at any time.
- If instead you write your code directly in the console, your code will be run but you won't be able to modify it or reuse it later.



# Some Useful Shortcuts

- To save a script use `Cmd/Ctrl + S`
- To send code from the editor to the console use `cmd/ctrl + enter`. This will run the current line of code (or the current selection of lines)
- To run the whole script use `Cmd/Ctrl + Shift + S`

# Example



The screenshot shows the RStudio interface with the following components:

- Script Editor:** Contains an R script with comments and calculations. A red box highlights the code: 

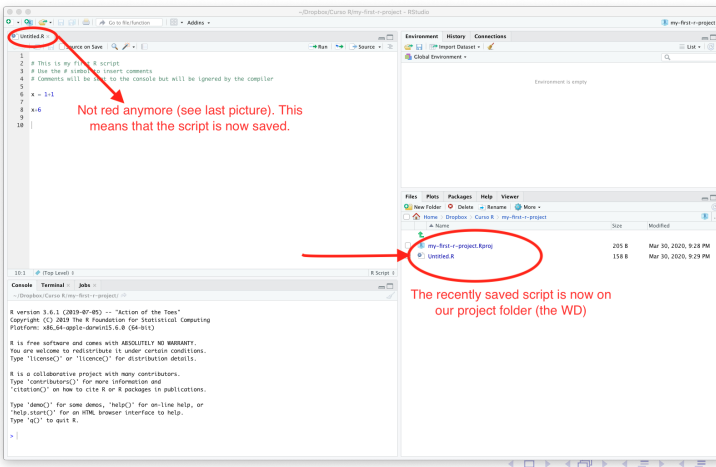
```
x = 1 + 1  
y = cos(1)
```
- Environment Pane:** Labeled "Stored values", it shows the current values of variables: 

Variable	Value
x	2
y	0.5403023
- Console:** Shows the output of the script execution: 

```
> y = cos(1)  
> y  
[1] 0.5403023  
> x = 1 + 1  
> x  
[1] 2  
> y = cos(1)  
> y  
[1] 0.5403023
```

Red arrows indicate the workflow: one arrow points from the script code to the console, labeled "Cmd/Ctrl + Enter", and another arrow points from the console output to the Environment pane.

# Example (cont.)

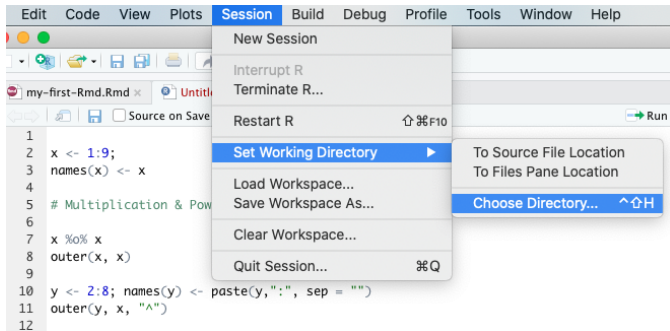




# What is a Working Directory?

- An active R session always has an associated working directory associated. R will use the working directory by default to:
  - ▶ Search for files
  - ▶ Save outputs (tables, plots, etc)

# Setting the Working Directory



**Figure 7:** Setting the working directory





# Setting the Working Directory

- You can also get or set the working directory in R's console:
  - ▶ Getting the working directory
    - > `getwd()`
  - ▶ Changing the working directory
    - > `setwd("/folder1/folder2/folder3/")`
      - The problem with commands like this is that such paths will only exist on your computer
      - Solution: Rstudio projects



# Advantages of Rstudio Projects

- Rstudio projects are self-contained.
- They put together all the files that are relevant for a particular project (article, book, research project) in the same folder.
- The project's working directory always points to that folder by default
- Rstudio projects can be moved around on your computer or onto other computers and will still “*just work*”. No directory changes are needed.
- If you need to create additional folders or start moving around parts of you project around dont use the *setwd* function. It is safer to reference the full path.

# Packages



- The more specialized functions and data sets are available on packages (also referred to as libraries).
  - ▶ Installing R Packages:  
`> install.packages("ggplot2", dependencies = TRUE)`
  - ▶ Loading R Packages:  
`> library("ggplot2")`
  - ▶ Updating R Packages:  
`> update.packages()` # This is rarely necessary
- Packages are developed by the R core team and also by the community of R users.
- You can develop your own packages and make them available to the community on [CRAN](https://cran.r-project.org/) (The Comprehensive R Archive Network)

# Packages



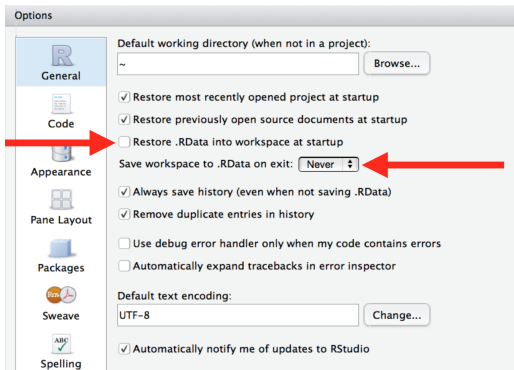
- It is typically recommend to start your scripts with the packages that you need.
- That way, if you share your code with others, they can easily see what packages they need to install.
- Note, however, that you should never include *install.packages* or *setwd* in a script that you share.
- It is very antisocial to change settings on someone else's computer!

# Settings and Appearance



- You can change Rstudio's default settings and appearance:
  - ▶ Mac: Tools – > Global Option
  - ▶ Windows and Linux: Rstudio – > preferences
- Shortcut:
  - ▶ Mac: "Cmd" + ","
  - ▶ Windows and Linux: "Ctrl" + ","

# Settings and Appearance



**Figure 8:** These are the general settings that we recommend

# Settings and Appearance

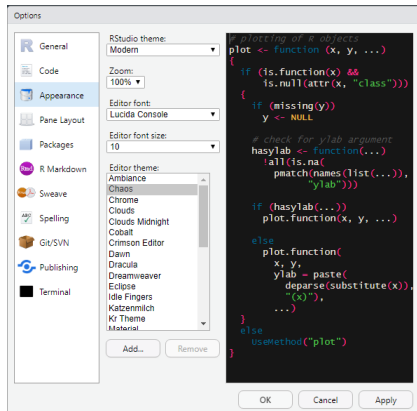


Figure 9: Changing Rstudio's appearance

# Questions?

