

# R para análise de dados em Psicologia

Uma Introdução

Francisco Pablo Huascar Aragão Pinheiro



# Quem sou eu

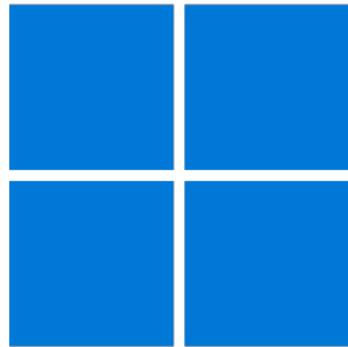


- Psicólogo
- Doutor em educação
- Professor do campus Sobral da UFC
- Recentemente, um entusiasta do R

# Por que usar o R?

- Gratuito
- Pesquisa reproduzível (para outros cientistas e o seu eu do futuro)
- Flexível
- Milhares de funções para todo tipo de análise
  - Novas funções são desenvolvidas o tempo todo ao redor do mundo

O que eu preciso saber  
para participar desse  
minicurso?



# Windows 11

- Criar e navegar entre pastas
- Abrir arquivos
- Criar arquivos etc.

# Estatística

- Medidas de tendência central e dispersão
- Anova e Teste T
- Correlação
- Regressão

# Eu tenho que aprender uma linguagem de programação para usar o R?



Sim, mas é como aprender um novo idioma: há desconforto inicial, mas, com a prática, é possível se tornar fluente!



R Studio<sup>®</sup>



R

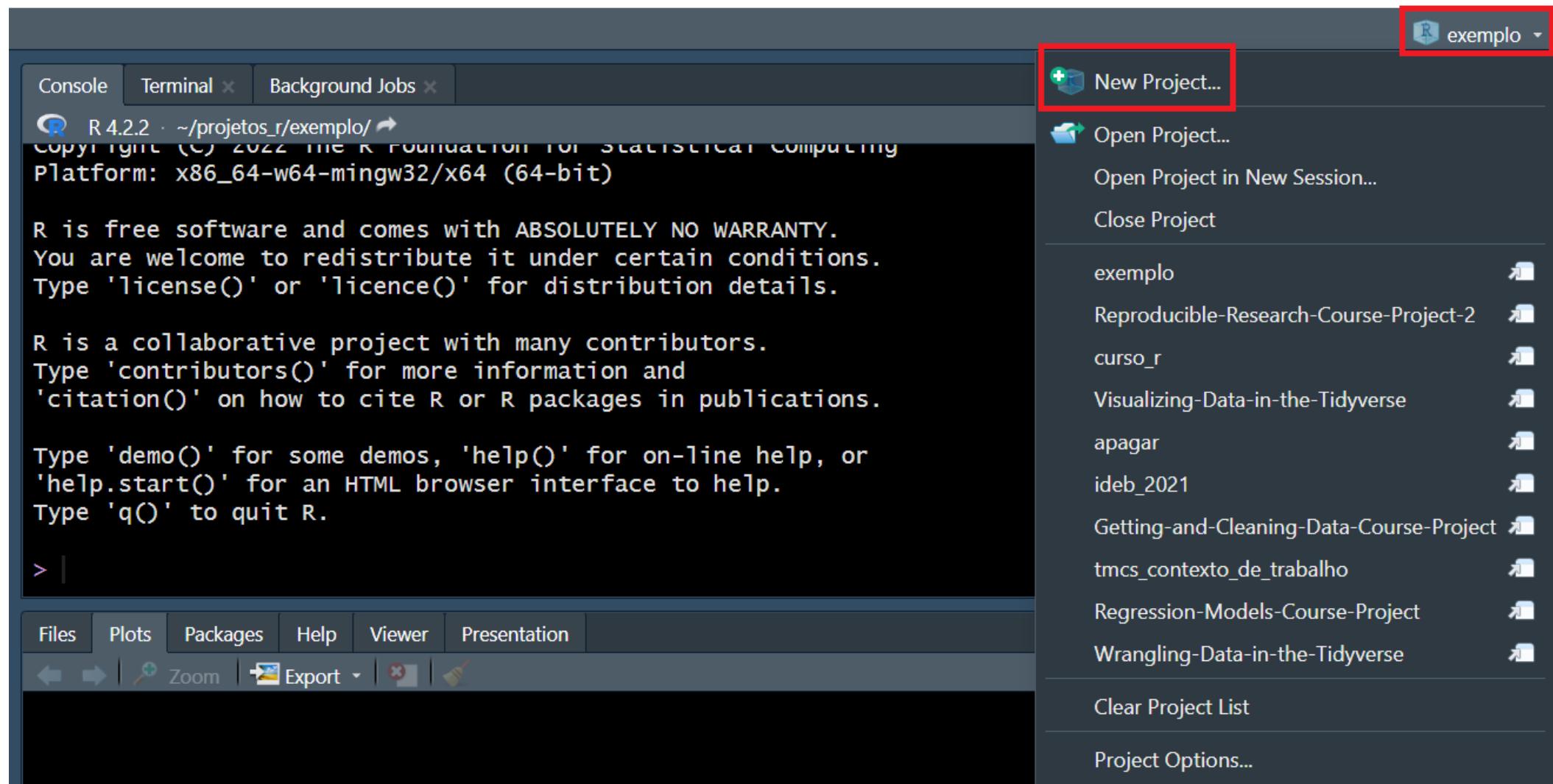
10

# R Studio



Vamos  
começar!

# Criação de um projeto - Passo 1



# Criação de um projeto - Passo 2

New Project Wizard

## 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

# Criação de um projeto - Passo 3

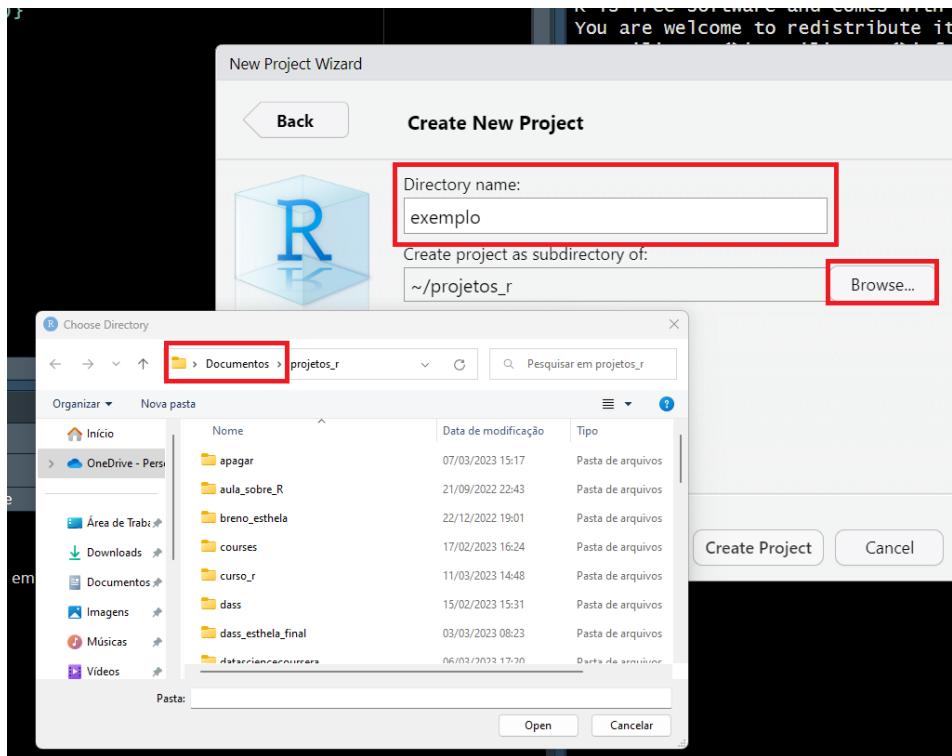
New Project Wizard

Back      **Project Type**

-  New Project >
-  R Package >
-  Shiny Application >
-  Quarto Project >
-  Quarto Website >
-  Quarto Blog >
-  Quarto Book >

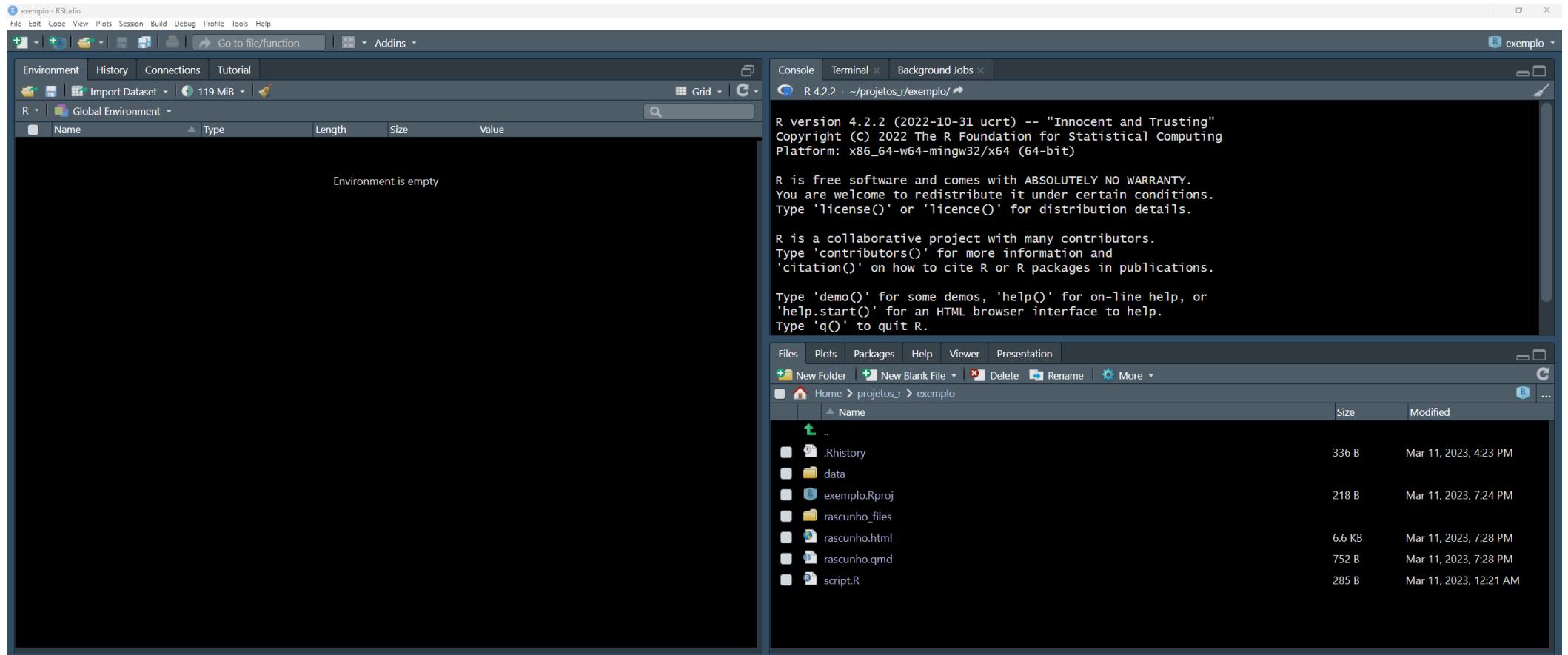
Cancel

# Criação de um projeto - Passo 4

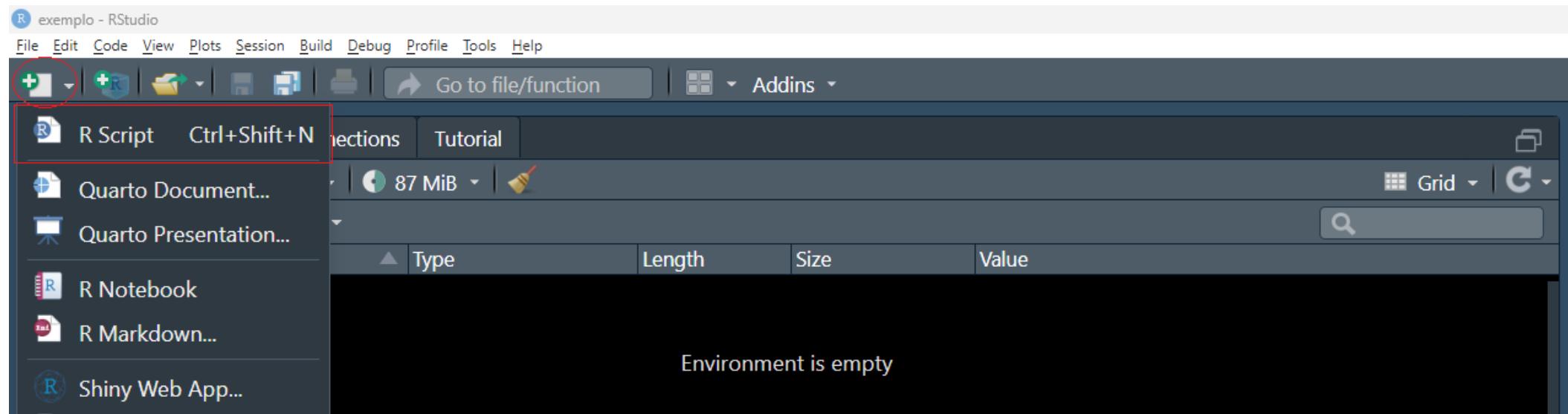


- É muito importante que todos os arquivos do projeto fiquem dentro de uma mesma pasta
- A pasta “raiz” não pode conter caracteres especiais
- Dica: crie seus projetos dentro da pasta “documentos” do Windows (ou em uma subpasta “projetos” dentro da pasta “documentos”)

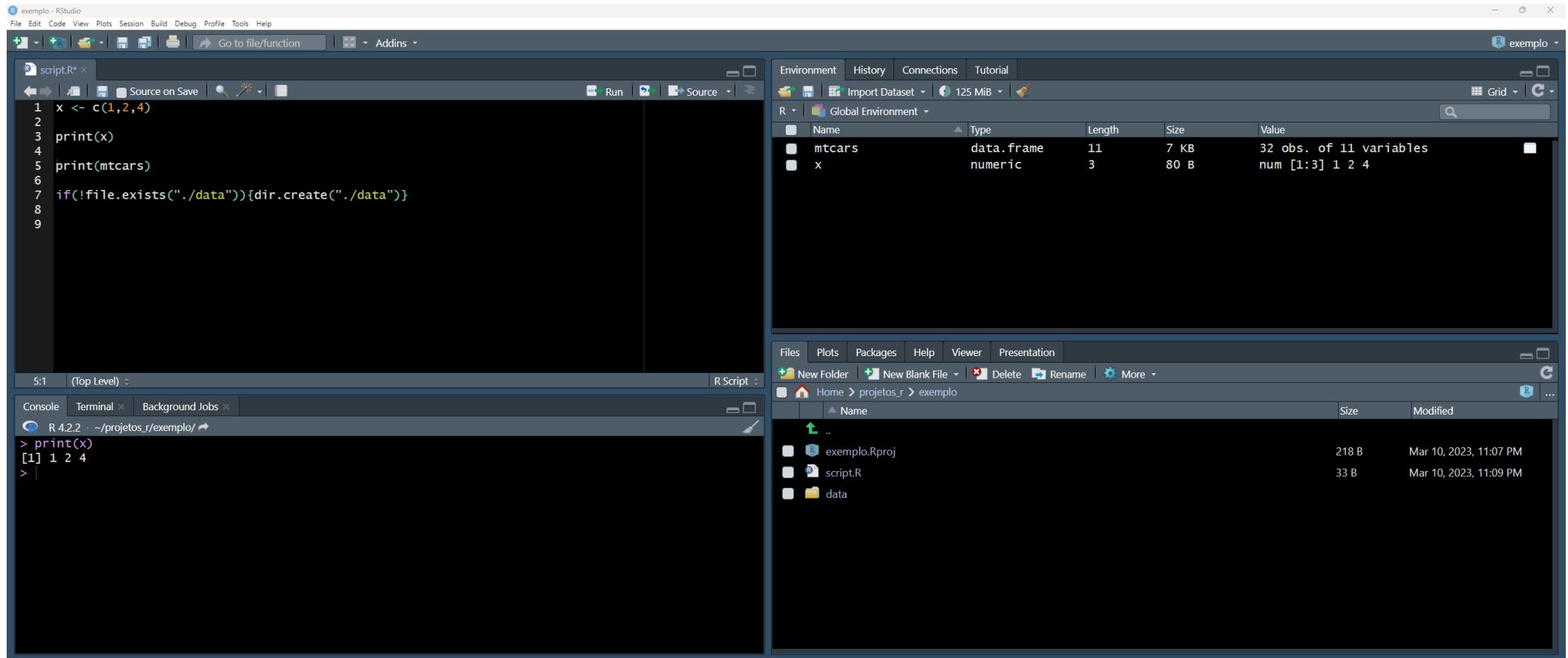
# E você vai chegar aqui



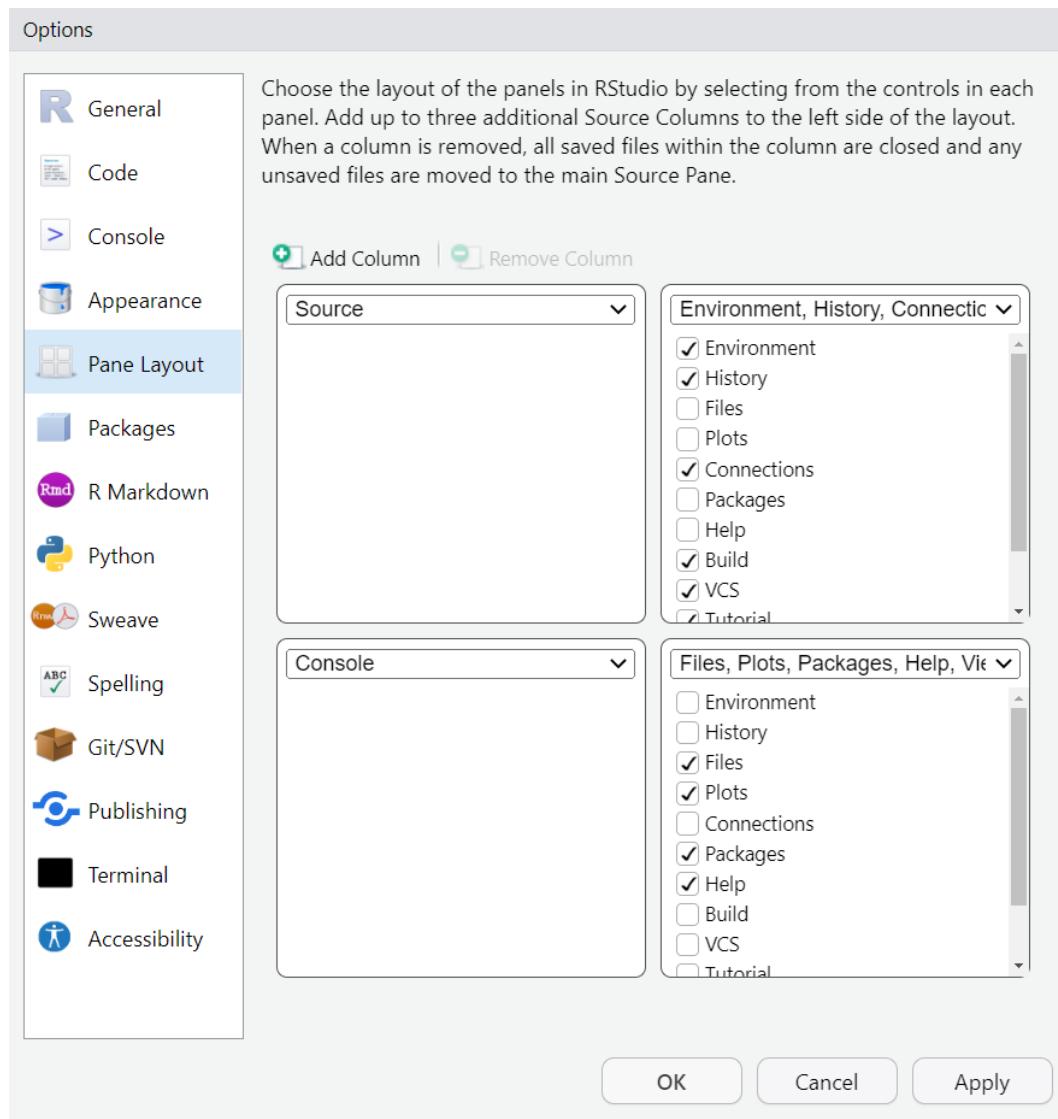
# Criando o seu primeiro Script



# Paineis do R Studio

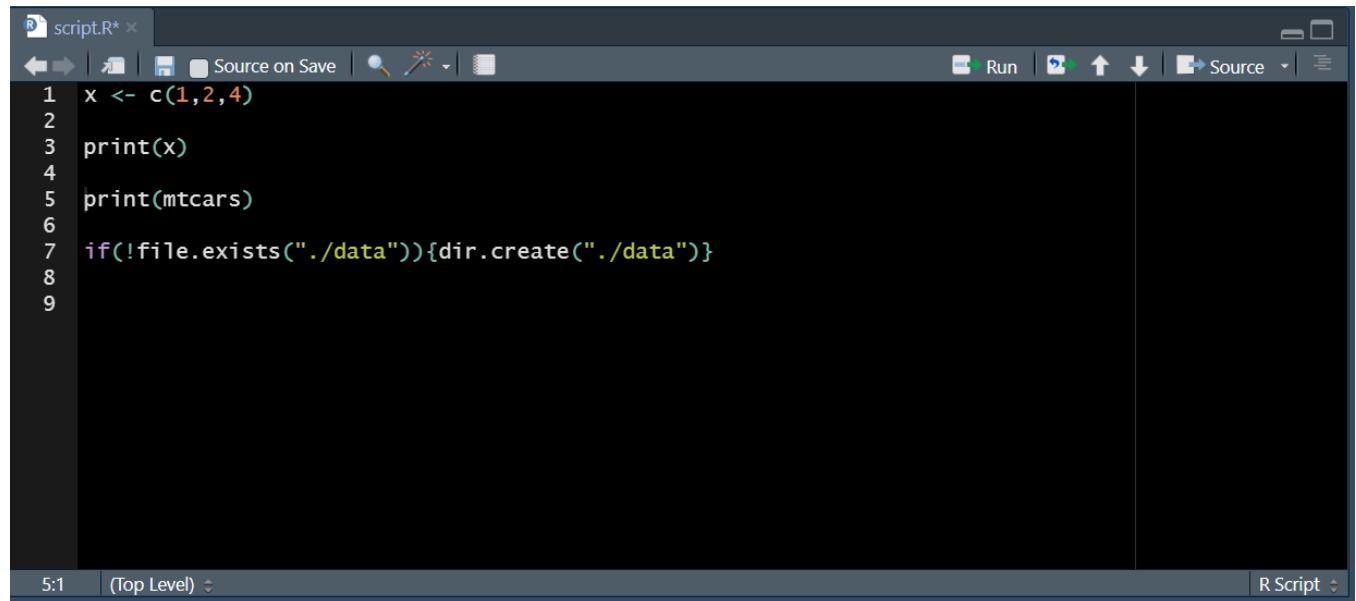


# Ajuste dos painéis



# Source Code

É onde o código é escrito. Vários tipos de arquivos podem ser utilizados:  
Scripts,  
Rmarkdown etc.

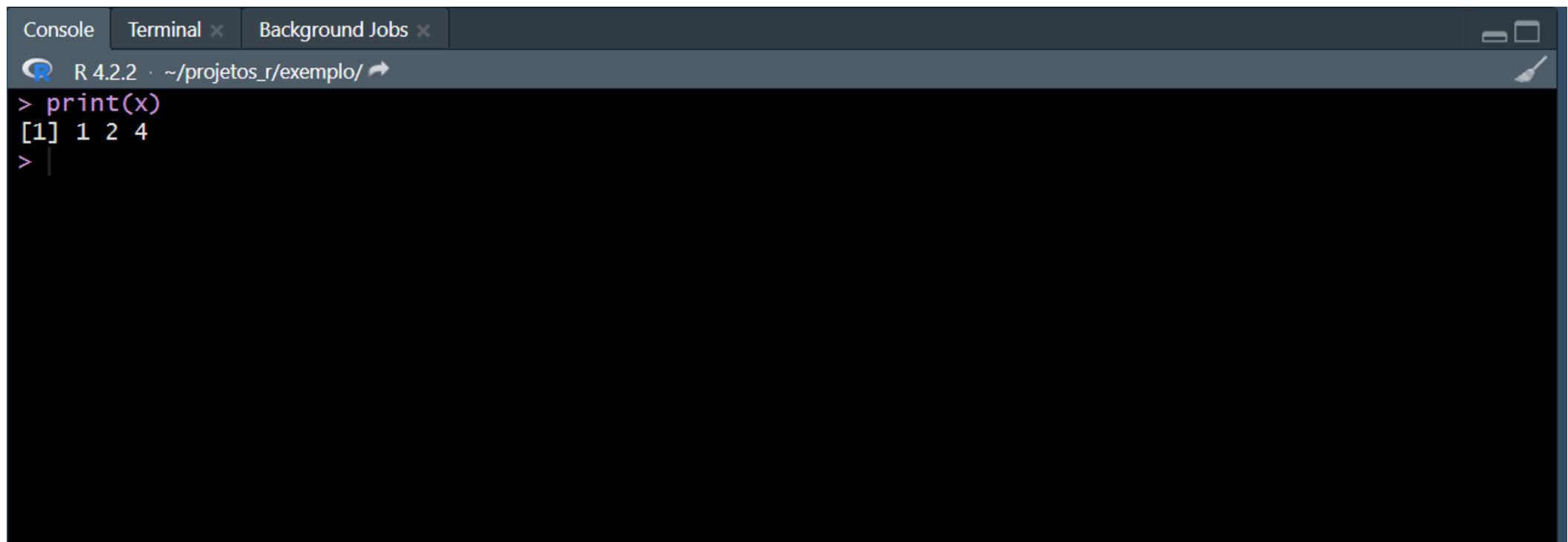


```
R script.R* x
Source on Save | Run | Source | Help
1 x <- c(1,2,4)
2
3 print(x)
4
5 print(mtcars)
6
7 if(!file.exists("./data")){dir.create("./data")}
8
9

5:1 (Top Level) R Script
```

# Console

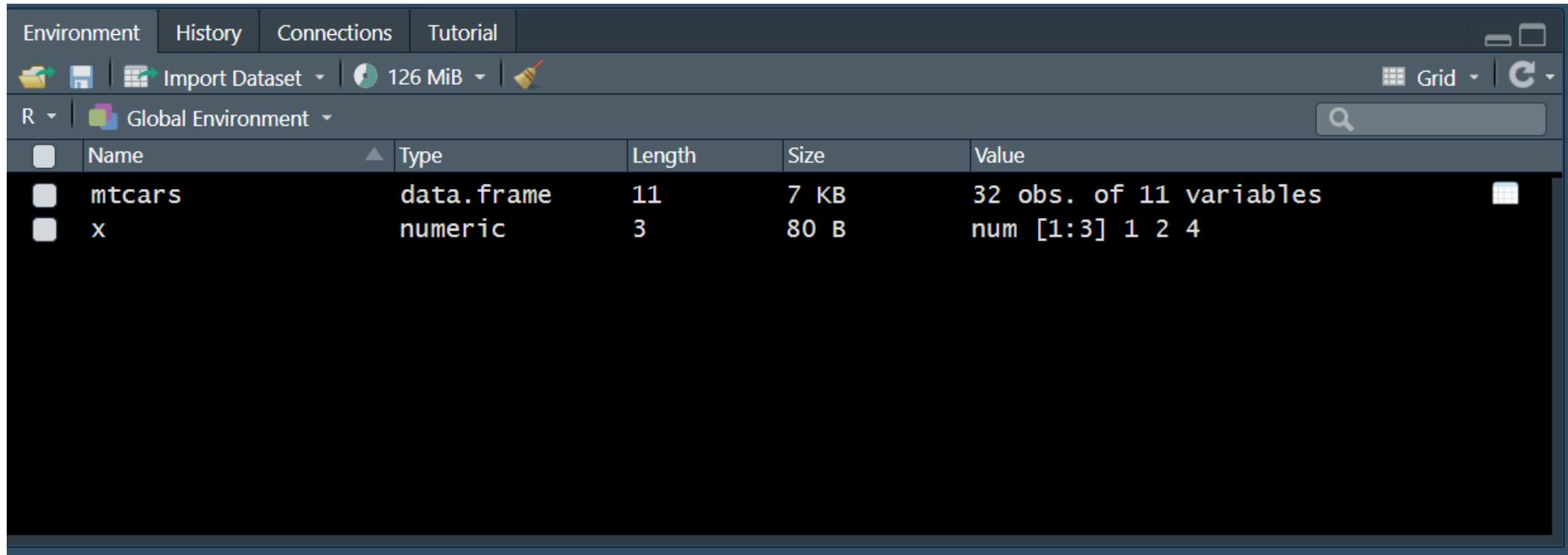
Mostra a saída do que é feito no source code. Também pode ser usado para escrever códigos, mas não é possível salvá-los



The screenshot shows the RStudio interface with the 'Console' tab selected. The title bar indicates 'R 4.2.2 · ~/projetos\_r/exemplo/'. The console window displays the following R session:

```
R 4.2.2 · ~/projetos_r/exemplo/
> print(x)
[1] 1 2 4
> |
```

# Environment/History



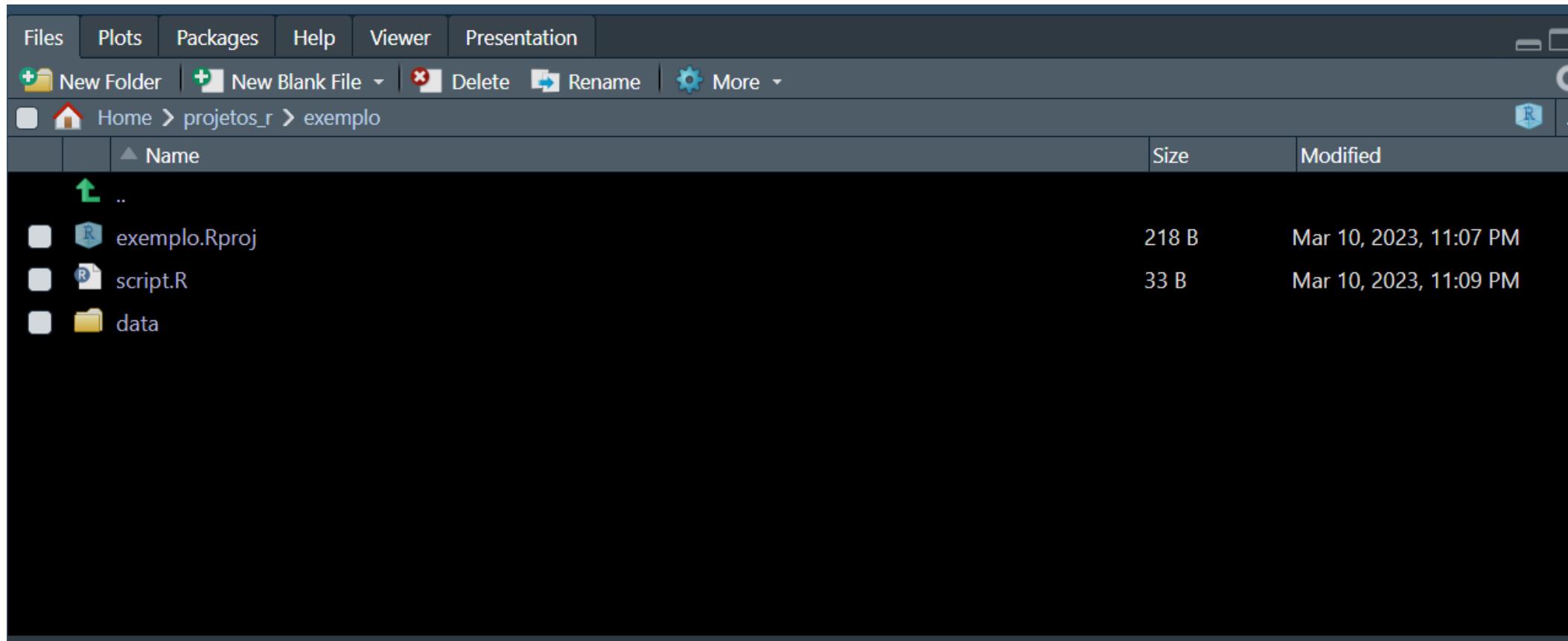
The screenshot shows the RStudio interface with the 'Environment' tab selected. The global environment contains two objects: 'mtcars' (a data frame with 32 observations and 11 variables) and 'x' (a numeric vector with 3 elements). The 'Global Environment' dropdown is open, and the 'Grid' view is selected.

Name	Type	Length	Size	Value
mtcars	data.frame	11	7 KB	32 obs. of 11 variables
x	numeric	3	80 B	num [1:3] 1 2 4

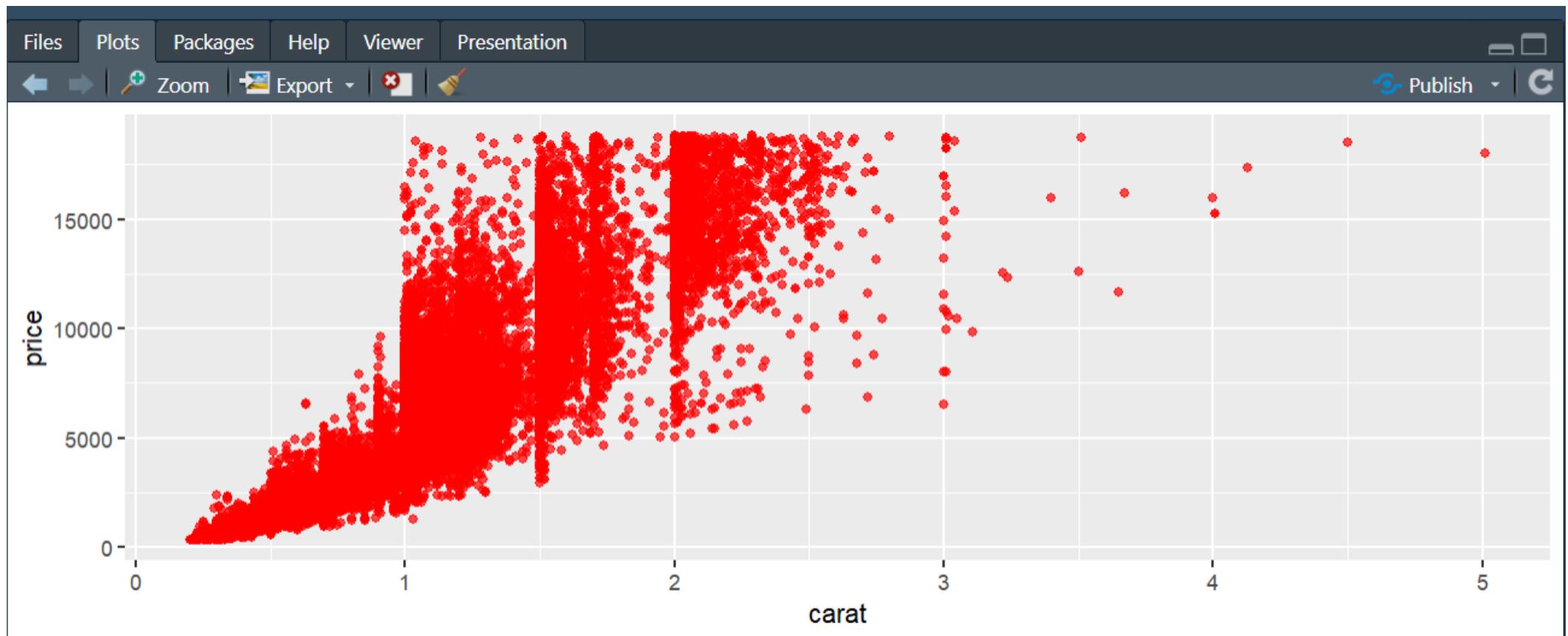
No environment estão os diversos objetos que são criados pelo código: vetores, dataframes (bancos de dados), tibbles etc.

# File/Plots/Packages/Help

Neste painel são mostrados os arquivos do projeto, os gráficos produzidos, a ajuda e informações sobre os pacotes (instalar, ativar, desativar etc).



# File/Plots/Packages/Help



# Cheat Sheets

**Keyboard Shortcuts**

	Windows/Linux	Mac
<b>RUN CODE</b>	Ctrl+arrow-up Esc Ctrl+L	Cmd+arrow-up Esc Ctrl+L
Interrupt current command		
Clear console		
<b>NAVIGATE CODE</b>		
Go to File/Function	Ctrl+	Ctrl+
<b>WRITE CODE</b>		
Attempt completion	Tab or Ctrl+Space	Tab or Ctrl+Space
Insert <- (assignment operator)	Alt+-	Option+-
Insert %-% (pipe operator)	Ctrl+Shift+M	Cmd+Shift+M
(Un)Comment selection	Ctrl+Shift+C	Cmd+Shift+C
<b>MAKE PACKAGES</b>		
Load All (devtools)	Ctrl+Shift+I	Cmd+Shift+I
Test Package (Desktop)	Ctrl+Shift+T	Cmd+Shift+T
Document Package	Ctrl+Shift+D	Cmd+Shift+D

**DOCUMENTS AND APPS**

- Knit Document (knitr)
- Insert chunk (Sweave & Knitr)
- Run from start to current line

**MORE KEYBOARD SHORTCUTS**

- Keyboard Shortcuts Help
- Show Command Palette

**Share Projects**

**R Markdown**

This is an R Markdown document. R Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents.

**Run Remote Jobs**

Run R on remote clusters (Kubernetes/Slurm) via the Job Launcher.

**R Studio Workbench**

WHY RSTUDIO WORKBENCH?

Extend the open source server with a commercial license, support, and more:

- open and run multiple R sessions at once
- tune your resources to improve performance
- administrative tools for managing user sessions
- collaborate real-time with others in shared projects
- switch easily from one version of R to a different version
- integrate with your authentication, authorization, and audit practices
- work in the RStudio IDE, JupyterLab, Jupyter Notebooks, or VS Code

Download a free 45-day evaluation at [www.rstudio.com/products/workbench/evaluation/](http://www.rstudio.com/products/workbench/evaluation/)

**Keyboard Shortcuts Reference**

Search for keyboard shortcuts with Tools > Keyboard Shortcuts or Alt+Option+Shift+K

**Source Navigation**

Source navigation features include:

- Create a New R Script
- Create a new R Markdown document
- Open Shiny, R Markdown, knitr, Sweave, LaTeX, Rd files and more in Source Pane

**Code Completion**

Search for keyboard shortcuts with Tools > Show Command Palette or Ctrl/Cmd + Shift + P

**Visual Editor**

Block format

Check spelling

Render output

Output format

Choose output location

Insert code chunk

Jump to previous chunk

Jump to next chunk

Run selected lines

Publish to server

Show file outline

Back to Source Editor (front page)

Heading 2

B I U Knit

Lists and block quotes

Links

Citations

Images

More formatting

Insert blocks, citations, equations, and special characters

Insert tables

R Markdown Including Plots

File outline

Insert and edit tables

Share Project with Collaborators

Select R Version

**R Markdown**

This is an R Markdown document. R Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents.

{r cars}  
summary(cars)

R Markdown 2

Jump to chunk or header

Set knitr chunk options

Run this and all previous code chunks

Run this code chunk

Run launcher jobs

Monitor launcher jobs

Launch a job

Run launcher jobs remotely

Run this code chunk

CC BY SA Posit Software, PBC • info@posit.co • posit.co • Learn more at [rstudio.com](http://rstudio.com) • Font Awesome 5.15.3 • RStudio IDE 1.4.1717 • Updated: 2021-07

**RStudio IDE :: CHEAT SHEET**

**Documents and Apps**

Check Spelling output

Render Choose Configure Insert Input renderers code chunk Publish to server

Navigation backwards/ forwards

Open in new window Save Find and replace Compile notebook Run selected code

**Source Editor**

Run previous code Run code Show file outline Visual Editor reverse sides

Run previous and all previous code chunks Run this code chunk Run this code chunk Set knitr chunk options

Run this function Set file outline options

Access markdown guide at Help > Markdown Quick Reference See reverse side for more on Visual Editor

RStudio recognizes that files named app.R, server.R, ui.R, and global.R belong to a shiny app

Active shared collaborators

Name of current project

Share Project with Collaborators

Select R Version

Start new R Session In current project Close R Session in project

**Tab Panes**

Import data with wizard History of past commands to run/copy Manage external databases View memory usage Rutorials

Open file without Echo or as a Local Job Show file outline without Echo or as a Local Job

Multiple environments with Alt + mouse drag Choose environment to display from list of parent environments

Code diagnostic that appear in the margin. Hover over diagnostic symbols for details.

Data Values

Functions

Display objects by type

Display objects by type

View in data viewer

View function source code

**Version Control**

Stage Commit Push/Pull View History branch

Staged staged files to remote

Open shell to type commands

Show file diff to view file differences

**Debug Mode**

Use debug(), browser(), or a breakpoint and execute your code to open the debugger mode.

Launch debugger to examine the origin of error that called before the error occurred

Open traceback to examine functions that it called before the error occurred

Console Terminal Jobs

= helloWorld() > Error

Show Traceback Run with Debug

Click next to line number to add/remove a breakpoint.

Highlighted line shows where execution has passed

**Package Development**

Create a new package with File > New Project > New Directory > R Package

Enable roxygen documentation with Tools > Project Options > Build Tools

Roxygen guide at Help > Roxygen Quick Reference

See package information in the Build Tab

Install package and restart R

Run devtools::load\_all() and reload changes

Start launcher job

Selected by submission time

factR Running Local 0.09

sleepy\_X Succeeded 11:22 AM Local 0.41

sleepy\_X idle KubernetesX Waiting 0.00

Run launcher jobs

Monitor launcher jobs

Launch a job

Run launcher jobs remotely

Run this code chunk

**RStudio IDE :: CHEAT SHEET**

RStudio opens plots in a dedicated Plots pane

Navigate recent plots window Export plot Delete plot Delete all plots

Home page of Shiny app Search within help file Search for help file

Viewer pane displays HTML content, such as Shiny apps, R Markdowns, and Interactive visualizations

Search for helpful links Publish to shinyapps.io Refresh RStudio

Run commands in environment where execution has passed

Examine variables. Select function in executing environment

Console Terminal Jobs

= Next Continue Stop

Step through code one line at a time

Step into and out of functions to run

Resume execution

Quit debug mode

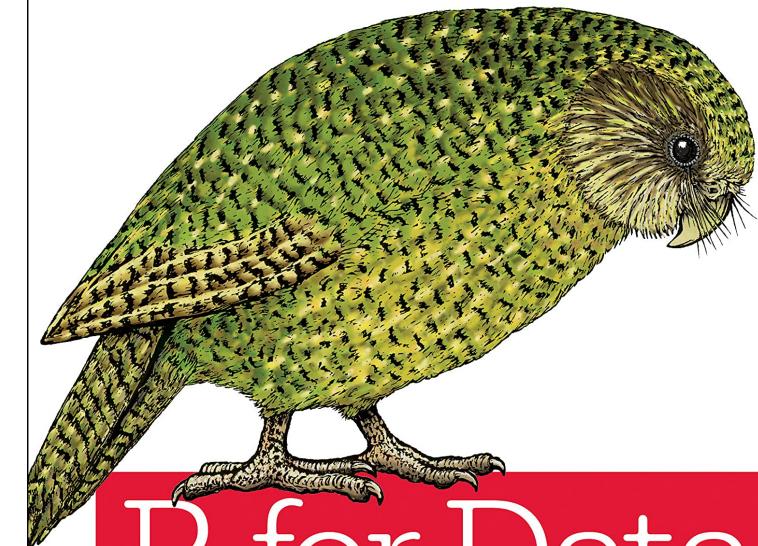
CC BY SA Posit Software, PBC • info@posit.co • posit.co • Learn more at [rstudio.com](http://rstudio.com) • Font Awesome 5.15.3 • RStudio IDE 1.4.1717 • Updated: 2021-07



# R for Data Science

Livro base para o mini curso

O'REILLY®



# R for Data Science

IMPORT, TIDY, TRANSFORM, VISUALIZE, AND MODEL DATA

Hadley Wickham &  
Garrett Grolemund

# R for Data Science

- Neste [link](#) você pode acessar a primeira edição do livro:

<https://r4ds.had.co.nz/>

- E [aqui](#) você pode acessar a segunda edição que ainda está sendo finalizada:

<https://r4ds.hadley.nz/>

# 3 Fluxo de trabalho: básico

# 3.1 Noções básicas de codificação

## Execução de códigos no Script

- Atalho de teclado: ctrl + enter



A screenshot of the RStudio interface showing a script editor window titled "Untitled1\*". The window contains the following R code:

```
1 library(tidyverse)
2
3 data("mtcars")
4
5 mtcars |>
6   select(mpg, cyl) |>
7   filter(mpg > 22)
```

The "Run" button in the toolbar is highlighted with a red box. The toolbar also includes other buttons for back, forward, search, and source.

# Cálculos matemáticos básicos

```
1 10+10
```

```
[1] 20
```

```
1 10/2
```

```
[1] 5
```

```
1 10*2+5
```

```
[1] 25
```

```
1 10-3
```

```
[1] 7
```

```
1 10^2
```

```
[1] 100
```

```
1 sqrt(100)
```

```
[1] 10
```

# Exercícios

- Divida 250 por 10
- Eleve 11 ao quadrado
- Multiplique 10 por 2 e some 5
- Subtraia 10 de 20 e divida por 10
- Some 40 e 50 e multiplique por 10

# Respostas

1  $250/10$

[1] 25

1  $11^2$

[1] 121

1  $10*2+5$

[1] 25

1  $(20-10)/10$

[1] 1

1  $(40+50)/10$

[1] 9

# Criação de objetos com o operador de atribuição <-

```
1 x <- 3  
2 x
```

```
[1] 3
```

```
1 y <- 10*10  
2 y
```

```
[1] 100
```

- Os valores de `x` ou `y` não são impressos; apenas armazenados.
- Para visualizar o valor de um `objeto`, execute-o no script.

# Todas as instruções de atribuição têm o mesmo formato

```
1 nome_do_objeto <- "valor"  
2  
3 nome_do_objeto  
  
[1] "valor"
```

- Ao ler esse código, diga: “o nome do objeto obtém valor”
- Atalho de teclado do R Studio: Alt + “-” (sinal de menos)

# Exercícios

Crie os seguintes objetos:

- a com valor 2
- b com valor 10
- c com valor 5
- d com valor  $10^2$

# Respostas

```
1 a <- 2  
2 a
```

```
[1] 2
```

```
1 b <- 10  
2 b
```

```
[1] 10
```

```
1 c <- 5  
2 c
```

```
[1] 5
```

```
1 d <- 10^2  
2 d
```

```
[1] 20
```

# As operações podem ser realizadas entre objetos

```
1 a*b
```

```
[1] 20
```

```
1 a+b
```

```
[1] 12
```

```
1 a/b
```

```
[1] 0.2
```

```
1 a-b
```

```
[1] -8
```

# Exercícios

- Divida **c** por **d** e some **a**
- Some **a** e **b** e divida por **c**
- Multiplique **a** por **b** e some **d**
- Some **c** com **d** e divida por **a**

# Respostas

```
1 (c/d) + a
```

```
[1] 2.25
```

```
1 (a+b)/c
```

```
[1] 2.4
```

```
1 (a*b)+d
```

```
[1] 40
```

```
1 (c+d)/a
```

```
[1] 12.5
```

# Exercício desafio

Para encontrar as soluções para uma equação do formato  $ax^2 + bx + c$ , use a equação quadrática:

$$x = \frac{-b \pm \sqrt{(b^2 - 4ac)}}{2a}.$$

Quais as duas soluções para  $2x^2 - x - 4 = 0$ ?

# Resposta

```
1 # Definindo os coeficientes da equação
2 a <- 2
3
4 b <- -1
5
6 c <- -4
7
8 # Calculando as soluções da equação
9 x1 <- (-b + sqrt(b^2 - 4*a*c)) / (2*a)
10
11 x2 <- (-b - sqrt(b^2 - 4*a*c)) / (2*a)
12
13 x1
```

```
[1] 1.686141
```

```
1 x2
```

```
[1] -1.186141
```

# É possível combinar múltiplos elmentos em um vetor com c() - “concatenar”

```
1 z <- c(1,2,10,12)  
2 z
```

```
[1] 1 2 10 12
```

```
1 w <- c(3,5,6,8)  
2 w
```

```
[1] 3 5 6 8
```

# Aritimética básica nos vetores é aplicada a cada elemento do vetor

```
1 z
```

```
[1] 1 2 10 12
```

```
1 z^2
```

```
[1] 2 4 20 24
```

```
1 w
```

```
[1] 3 5 6 8
```

```
1 w^3
```

```
[1] 9 15 18 24
```

# Também é possível realizar operações com vetores

```
1 z+w
```

```
[1] 4 7 16 20
```

```
1 z-w
```

```
[1] -2 -3 4 4
```

```
1 z-w^2
```

```
[1] -8 -23 -26 -52
```

# Exercícios

- Crie os vetores `z <- c(1,2,10,12)` e `w <- c(3,5,6,8)`
- Eleve `z` ao quadrado
- Some 10 a `w`
- Divida `z` por 3
- Multiplique `z` e `w`
- Divida `z` por `w`

# Respostas

```
1 z <- c(1,2,10,12)  
2 z
```

```
[1] 1 2 10 12
```

```
1 w <- c(3,5,6,8)  
2 w
```

```
[1] 3 5 6 8
```

```
1 z^2
```

```
[1] 1 4 100 144
```

```
1 w+10
```

```
[1] 13 15 16 18
```

```
1 z/3
```

```
[1] 0.3333333 0.6666667 3.3333333 4.0000000
```

```
1 z+w
```

```
[1] 4 7 16 20
```

```
1 z/w
```

```
[1] 0.3333333 0.4000000 1.6666667 1.5000000
```

# Exercícios

- Crie o objeto `peso` com os seguintes valores: 80.2, 56.3, 70.5 e 60.3
- Crie o objeto `altura` com os seguintes valores: 1.75, 1.60, 1.65 e 1.72
- Assuma que os valores acima estão, respectivamente, em quilogramas e em metros
- Sabendo que o Índice de Massa Corpórea (IMC) é obtido dividindo o peso (em kg) pela altura (em metros) ao quadrado, crie um objeto `imc` com os valores do IMC de cada caso.

# Respostas

```
1 peso <- c(80.2, 56.3, 70.5, 60.3)
2 peso
```

```
[1] 80.2 56.3 70.5 60.3
```

```
1 altura <- c(1.75, 1.60, 1.65, 1.72)
2 altura
```

```
[1] 1.75 1.60 1.65 1.72
```

```
1 imc <- peso/altura^2
2 imc
```

```
[1] 26.18776 21.99219 25.89532 20.38264
```

# Dica 1

- O R diferencia MAIÚSCULAS de minúsculas (case sensitive)

```
1 a <- letters  
2  
3 A
```

Error: object 'A' not found

```
1 a <- letters  
2  
3 a
```

```
[1] "a" "b" "c" "d" "e" "f" "g" "h" "i" "j" "k" "l" "m" "n" "o" "p" "q" "r"  
"s"  
[20] "t" "u" "v" "w" "x" "y" "z"
```

# Dica 2

- Cuidado com erros de digitação

```
1 data("cars")
2
3 carz
```

Error: object 'carz' not found

# Cars

```
1 data("cars")  
2  
3 cars
```

	speed	dist
1	4	2
2	4	10
3	7	4
4	7	22
5	8	16
6	9	10
7	10	18
8	10	26
9	10	34
10	11	17
11	11	28
12	12	14
13	12	20
14	12	24
15	12	20

## 3.2 Comentários

R irá ignorar qualquer texto após # para essa linha.

```
1 # Criação de um vetor com os 5 primeiros números pares  
2  
3 pares <- c(2, 4, 6, 8, 10)  
4  
5 pares
```

```
[1] 2 4 6 8 10
```

```
1 # Elevando os pares ao quadrado  
2  
3 (pares)^2
```

```
[1] 4 16 36 64 100
```

# Uso dos comentários: Demarcar seções do no código

Código Gráfico

```
1 # Obtenção dos dados
2
3 penguins <- penguins
4
5 # Gráficos
6
7 penguins |>
8   filter(!is.na(sex)) |>
9   ggplot(aes(flipper_length_mm,
10           bill_length_mm,
11           color = sex,
12           size = body_mass_g)) +
13   geom_point(alpha = 0.5) +
14   facet_wrap(~species)
```

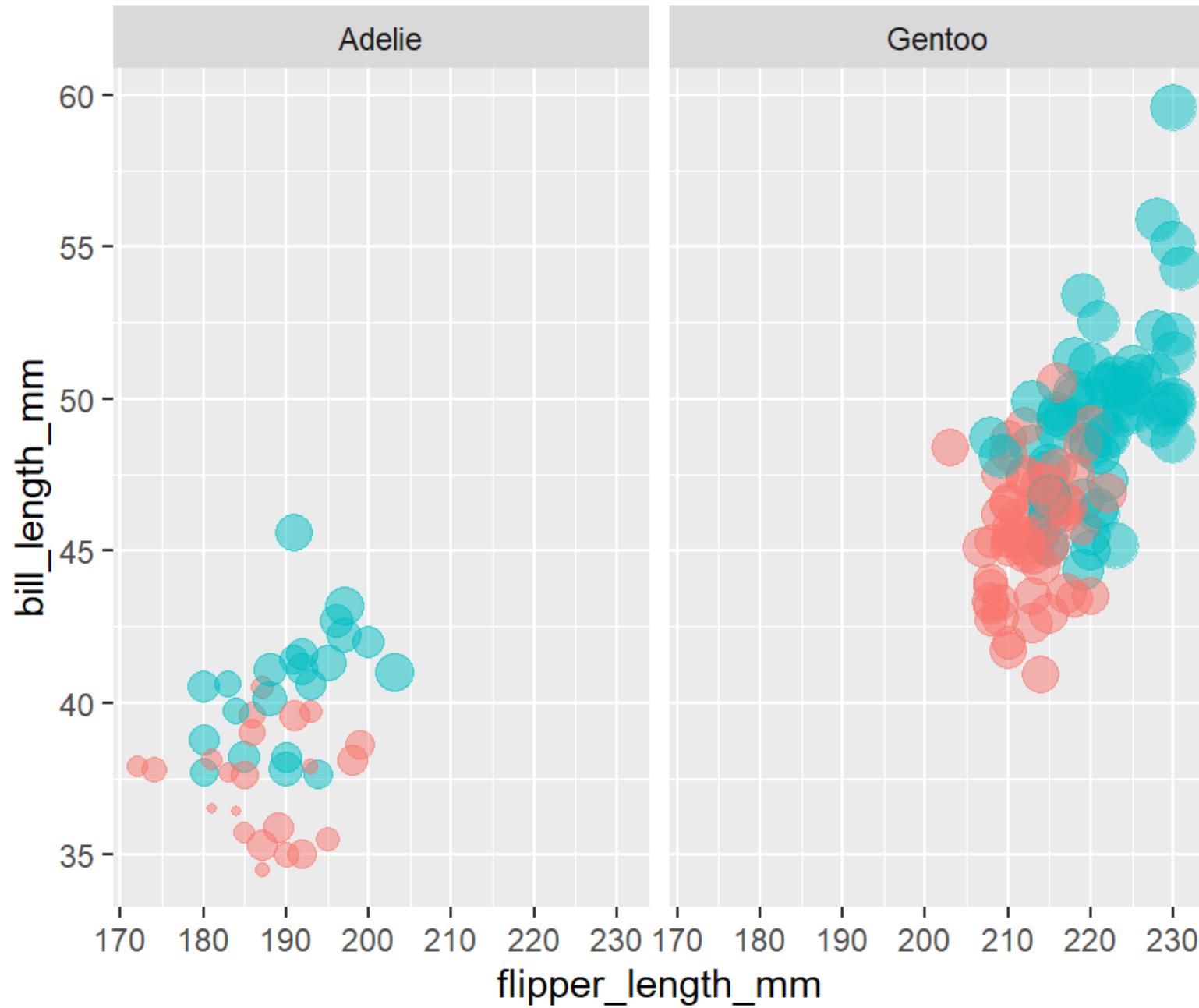
# Outro uso dos comentários: explicações

- Use comentários para explicar o porquê do seu código, não o como ou o quê
- O que e como do seu código são sempre possíveis de descobrir
- Descobrir por que algo foi feito é muito mais difícil

# Sem explicações para a equipe

```
1 penguins |>
2   filter(island == "Biscoe",
3         !is.na(sex)) |>
4   ggplot(aes(flipper_length_mm,
5             bill_length_mm,
6             color = sex, size =
7             body_mass_g)) +
8   geom_point(alpha = 0.5) +
9   facet_wrap(~species)
```

# Sem explicações para a equipe



body\_mass\_g

- 3000
- 4000
- 5000
- 6000

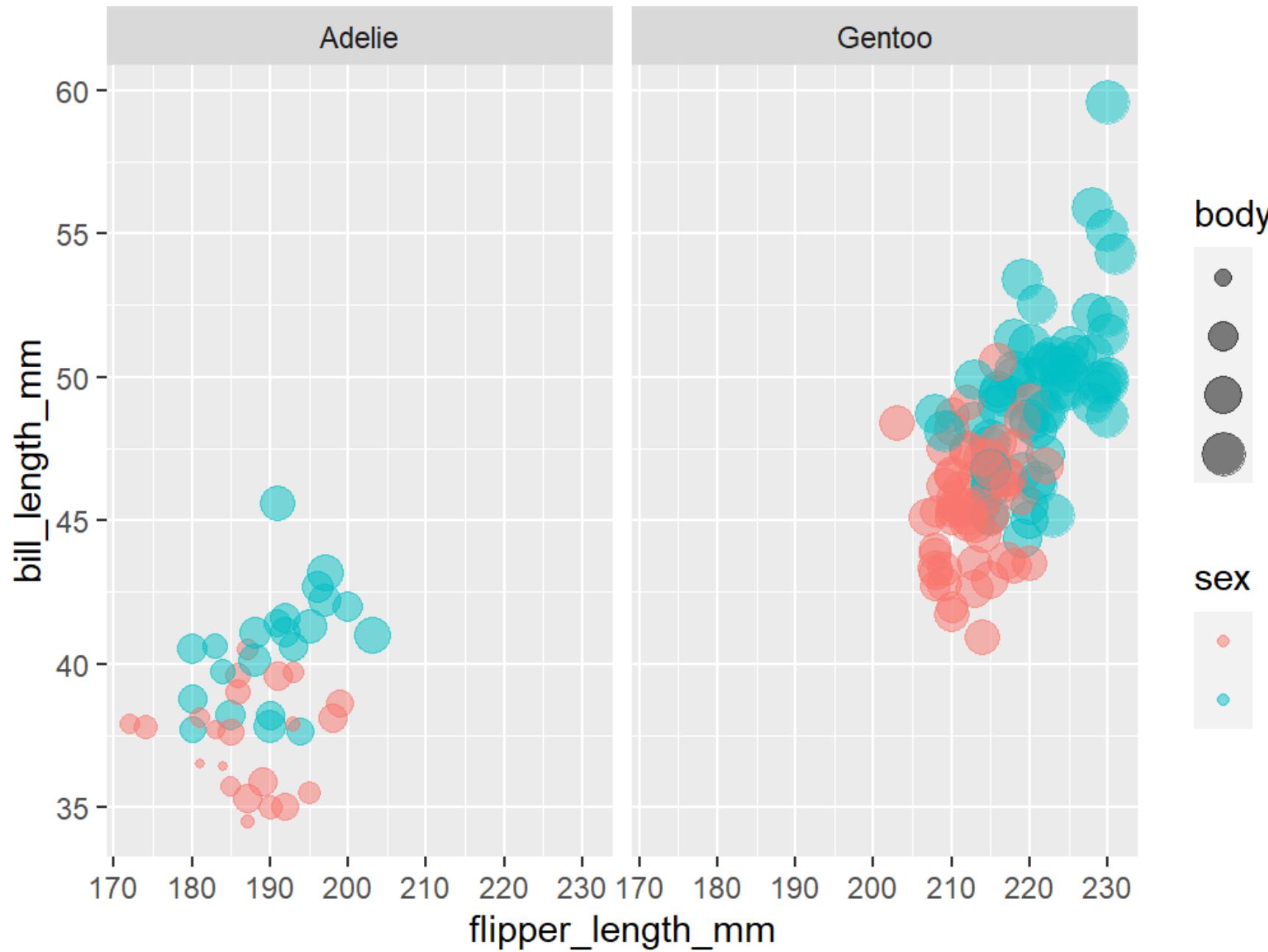
sex

- female
- male

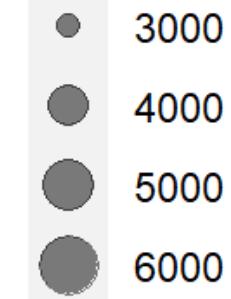
# Com explicações para a equipe

```
1 # O relatório 3 só precisava dos
2 #dados da ilha Biscoe
3 penguins |>
4   filter(island == "Biscoe",
5         !is.na(sex)) |>
6   ggplot(aes(flipper_length_mm,
7             bill_length_mm, color =
8             sex,
9             size = body_mass_g)) +
10  geom_point(alpha = 0.5) +
11  facet_wrap(~species)
```

# Com explicações para a equipe



body\_mass\_g



sex



### 3.3 O que é um nome?

- Os nomes dos objetos devem começar com uma letra
- Podem conter apenas **letras, números, \_ e .**
- Os nomes de seus objetos devem ser descritivos
- Sugerimos a seguinte convenção:
  - **letras\_minúsculas\_separadas\_com \_**

# Exemplos de nomes

- i\_use\_snake\_case
- otherPeopleUseCamelCase
- some.people.use.periods
- And\_aFew.People\_RENOUNCEconvention

# Dica: auto completar

- Digite ao menos 3 caracteres e use a tecla tab
- O R Studio vai dar opções para escolher
- A função de auto completar é nossa amiga: use sempre!

# 3.4 Usando Funções

O R possui uma grande coleção de funções integradas que são chamadas desta maneira:

```
1 nome_da_funcao(argumento1 = valor1, argumento2 = valor2, ...)
```

Ao pressionar Tab dentro dos parênteses o R irá mostrar os argumentos disponíveis

# Exemplo: Função seq()

Os dois primeiros argumentos da função seq são: from e to

```
1 # Crie uma sequência de números de 1 a 10  
2 seq(from=1, to=10)
```

```
[1] 1 2 3 4 5 6 7 8 9 10
```

Terceiro argumento: by

```
1 # Agora, o incremento da sequência é por 2  
2 seq(from=1, to=10, by = 2)
```

```
[1] 1 3 5 7 9
```

Se os valores dos argumentos estiverem na ordem da função, é possível omití-los

```
1 seq(1,10,2)
```

```
[1] 1 3 5 7 9
```

# Exemplo: Função seq()

Ao explicitar os argumentos, eles podem aparecer em qualquer ordem:

```
1 seq(by=2, to=10, from=1)
```

```
[1] 1 3 5 7 9
```

Porém, não será obtido o mesmo resultado mudando os valores de lugar sem a explicitação dos argumentos

```
1 seq(2, 10, 1)
```

```
[1] 2 3 4 5 6 7 8 9 10
```

# Como saber os argumentos e a função das funções?

```
1 ?nome_da_funcao  
2  
3 help(nome_da_funcao)  
4  
5 args(nome_da_funcao)
```

# Help

1 ?seq

The screenshot shows the RStudio interface with the 'Help' tab selected. The main area displays the documentation for the 'seq' function. The title is 'seq {base}' and the subtitle is 'R Documentation'. The page is titled 'Sequence Generation'.

**Description**

Generate regular sequences. `seq` is a standard generic with a default method. `seq.int` is a primitive which can be much faster but has a few restrictions. `seq_along` and `seq_len` are very fast primitives for two common cases.

**Usage**

```
seq(...)

## Default S3 method:
seq(from = 1, to = 1, by = ((to - from)/(length.out - 1)),
    length.out = NULL, along.with = NULL, ...)

seq.int(from, to, by, length.out, along.with, ...)

seq_along(along.with)
seq_len(length.out)
```

**Arguments**

- ... arguments passed to or from methods.
- from, to the starting and (maximal) end values of the sequence. Of length 1 unless just `from` is supplied as an unnamed argument.
- by number: increment of the sequence.
- length.out desired length of the sequence. A non-negative number, which for `seq` and `seq.int` will be rounded up if fractional.

# args()

```
1 args(rnorm)  
function (n, mean = 0, sd = 1)  
NULL
```

# Exercícios

- A partir da ajuda, descubra para que servem as funções a seguir e tente utilizá-las:
  - `ls`
  - `round`
  - `rnorm`

# Funções básicas importantes

```
1 z <- c(2, 2, 6, 11, 9, 20)  
2 sum(z)
```

```
[1] 50
```

```
1 mean(z)
```

```
[1] 8.333333
```

```
1 median(z)
```

```
[1] 7.5
```

# Funções básicas importantes

```
1 f <- c(2, 2, 6, 11, 9, 20)  
2 range(f)
```

```
[1] 2 20
```

```
1 min(f)
```

```
[1] 2
```

```
1 max(f)
```

```
[1] 20
```

```
1 quantile(f)
```

	0%	25%	50%	75%	100%
f	2.0	3.0	7.5	10.5	20.0

# Exercícios

- Use a função `rnorm` e crie um objeto chamado `mil` com mil casos com média 30 e desvio padrão 4
- Calcule as seguintes estatísticas do objeto `mil`:
  - Soma
  - Média
  - Desvio padrão
  - Mediana
  - Mínimo
  - Máximo
  - Quartis

# Resposta

```
1 mil <- rnorm(n = 100, mean = 30, sd = 4)  
2  
3 sum(mil)
```

```
[1] 3006.608
```

```
1 mean(mil)
```

```
[1] 30.06608
```

```
1 sd(mil)
```

```
[1] 4.103809
```

```
1 median(mil)
```

```
[1] 30.15771
```

```
1 min(mil)
```

```
[1] 18.41662
```

```
1 max(mil)
```

```
[1] 39.10022
```

```
1 quantile(mil)
```

	0%	25%	50%	75%	100%
18.41662	27.38667	30.15771	32.70935	39.10022	

# Compartilhar conhecimento sempre!

O código e as imagens utilizados para a construção desse slide estão disponíveis [aqui](#), assim como os slides em formato PDF:

[https://github.com/pablo-huascar/curso\\_r](https://github.com/pablo-huascar/curso_r)

