

# Session Management

## R Intro

Gaston Sanchez

CC BY-NC-SA 4.0

R Coding Compendium

[Donate](#)

# About

## **Session Management**

In this slides we describe some important aspects about managing your session.

# Considerations of Working Directories

- ▶ **Workspace:** objects created in your current session; these can be saved in an `.RData` file when you quit R.
- ▶ **R's Console Working Directory:** the console has an associated working directory
- ▶ **Rmd's working directory:** when you open an Rmd and execute code in code chunks, they are executed in R's working directory
- ▶ When you **knit** an Rmd file, there is an associated working directory;
- ▶ Keep in mind that the working directory when running code chunks of an Rmd file, may be different from the working directory when knitting the Rmd file

# Important Files

- ▶ `.RData` file
- ▶ `.Rhistory` file in the project's main directory is loaded into the RStudio History pane (and used for Console Up/Down arrow command history).

# Project Management Basics

# Considerations

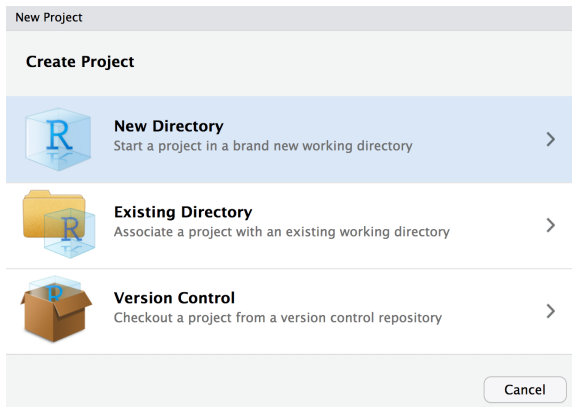
- ▶ Every “project” needs a **home**
- ▶ Projects can vary in terms of:
  - scope
  - size
  - goal(s)
  - complexity
- ▶ Although there is no “one-size-fits-all” project template, I recommend to keep a project as self contained as possible

# RStudio Projects

A great way to manage a project is by using an **RStudio Project**

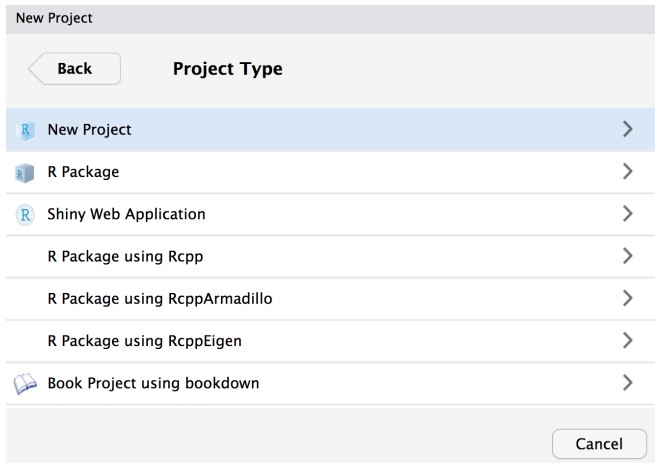
1. Go to the menu bar
2. Click on **File**
3. Select **New Project ...**
4. Choose **New project** type
5. Specify a location (directory) in your computer and give it a name

# Rstudio Project: Step 1





# Rstudio Project: Step 2




# Rstudio Project: Step 3

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

# RStudio Projects

RStudio projects make it straightforward to divide your work into multiple contexts, each with their own working directory, workspace, history, and source documents.

More information:

<https://support.rstudio.com/hc/en-us/articles/200526207-Using-Projects>

# RStudio Projects

When a project is opened within RStudio the following actions are taken:

- ▶ A new R session (process) is started
- ▶ The `.Rprofile` file in the project's main directory (if any) is sourced by R
- ▶ The `.RData` file in the project's main directory is loaded (if project options indicate that it should be loaded).
- ▶ The `.Rhistory` file in the project's main directory is loaded into the RStudio History pane (and used for Console Up/Down arrow command history).
- ▶ The current working directory is set to the project directory.
- ▶ Previously edited source documents are restored into editor tabs
- ▶ Other RStudio settings (e.g. active tabs, splitter positions, etc.) are restored to where they were the last time the project was closed.

# Open an R session

# R's Starting Message

R version 4.0.5 (2021-03-31) -- "Shake and Throw"  
Copyright (C) 2021 The R Foundation for Statistical Computing  
Platform: x86\_64-apple-darwin17.0 (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.

# R Version

- ▶ Usually, about 2 or 3 versions of R released per year
- ▶ Each version has its own name
- ▶ e.g. R version 4.0.5 -- "Shake and Throw"

# R's Starting Message

- ▶ What happens when you type `license()` or `licence()`?
- ▶ What happens when you type `contributors()`?
- ▶ What happens when you type `citation()`?
- ▶ What happens when you type `demo()`?
- ▶ What happens when you type `help()`?



# R's Starting Message

```
# GNU GPL2 license  
license()
```

```
# humans behind R  
contributors()
```

```
# citing R  
citation()
```

```
# some demos  
demo()
```

```
# on-line help  
help()
```

# Entering Input

At the R prompt, **>** , we type *expressions*.

```
> 5 + 3
```

```
>
```

```
> "some text"
```

```
>
```

```
> 3^2
```

# CalcuatoR

You can use R as a calculator

```
2 + 3
```

```
4 - 1
```

```
3 * 4
```

```
10 / 2
```

```
3^3
```

# Using functions

```
sqrt(9)
```

```
log(5)
```

```
exp(1)
```

```
(1.3 - 5)^2 + (log(5) / 3.14)
```

# Assignments

You can assign values to objects using the assignment operator `<-` or the equal sign `=` :

```
# assignment with 'arrow'
```

```
a <- 2 + 3
```

```
# assignment with 'equal'
```

```
b = 2 * 3
```

# Comments

The hash symbol **#** (or number sign) indicates a comment. Anything to the right of **#** is ignored.

```
# this is a comment  
txt <- 'this is some text'  
  
sqrt(9)  # example of square root  
  
# -----  
# more comments  
# -----
```

# R basics

R is case sensitive!

```
# Z different from z
```

```
Z <- 1
```

```
z <- 2
```

```
Z + z
```

```
## [1] 3
```

# R basics

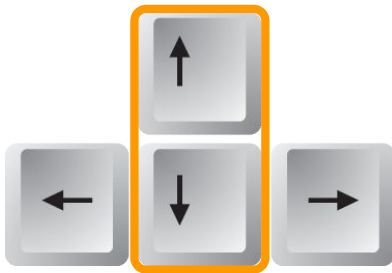
Case sensitive: this means that "hello" is not the same as "Hello" or "HELLO"

```
hello <- "hello"  
Hello <- "Hello"  
  
# are they equal?  
hello == Hello  
  
## [1] FALSE
```



## R basics

Use the up and down arrows to navigate through previous commands or instructions:



# R basics

Many languages use semicolons after each line. But in R there's (almost) no need to use semicolons

```
# no need for semicolons
```

```
2 + 4
```

```
2 + 4;
```

```
# except in this case (which I don't recommend)
```

```
# to include various statements in the same line
```

```
2 + 4; A <- 2 * 5; B <- 'abc'
```

# Terminate an R session

To quit a session simply type `quit()` or `q()`

```
# saves your workspace  
quit(save = "yes")
```

```
# doesn't save your workspace  
quit(save = "no")
```

# Terminate an R session

- ▶ If you use `quit("yes")` or `q("yes")` R will save your workspace (the created objects and variables).
- ▶ The workspace is saved in an `.RData` file.
- ▶ Next time you open R, the saved workspace should be available.

# Saved Workspace

If you previously typed `q("yes")`, open a new R session and inspect what objects do you have:

```
# list objects in your workspace  
ls()
```

# Recording your work

- ▶ In addition to `quit(save = "yes")`, there's also the function `savehistory()`
- ▶ You can use `savehistory()` to save everything you did
- ▶ It may be useful to call `savehistory()` at the end of a session
- ▶ By default, the commands-history will be saved in a file called `.Rhistory` (you can use other extension)
- ▶ You can open this file in any text editor

# Recording your work

Type some expressions, save your commands-history, and then quit R (without saving workspace)

```
2 * 2
2^10

# first comment
course <- "stat133"

# converting units
height_ft <- 5.9
height_in <- height_ft * 12
height_m <- height_ft * 0.3048

savehistory(file = 'test-session.R')
quit(save = "no")
```

Open the file "test-session.R" and see what's in it

# R Console

- ▶ Minimal GUI
- ▶ The console is OK for short expressions
- ▶ The console is good as a calculator
- ▶ But very limited for longer expressions
- ▶ It's better to alternate with **source scripts**



# Learning R

While learning R (or any programming language), keep in mind:

- ▶ You'll get frustrated
- ▶ It takes time to become fluent
- ▶ Lots of trials and errors
- ▶ Be patient
- ▶ Practice, practice, practice

## Donation

*If you find any value and usefulness in this set of slides, please consider making a one-time donation in any amount (via paypal). Your support really matters.*

Donate

[https://www.paypal.com/donate?business=ZF6U7K5MW25W2&currency\\_code=USD](https://www.paypal.com/donate?business=ZF6U7K5MW25W2&currency_code=USD)