World Health Organization
REGIONAL OFFICE FOR THE Eastern Mediterranean

منظمة الصحة العالمية
المكتب الإقليمي لشرق المتوسط

REGIONAL TRAINING ON CAPACITY DEVELOPMENT OF DATA ANALYTICS AND DISSEMINATION USING "R" SOFTWARE

AMMAN, JORDAN, 3 - 7 DECEMBER, 2023

Day #1

HEALTH FOR ALL BY ALL
a call for solidarity and action

World Health Organization
REGIONAL OFFICE FOR THE Eastern Mediterranean

Image source: Stats-illustrations

HEALTH FOR ALL BY ALL
a call for solidarity and action

World Health Organization
REGIONAL OFFICE FOR THE Eastern Mediterranean

Training sessions

Intro to R software (Session 1) and packages (Session 2)

Overview of R & Rstudio. Basic R programming language and best practices

Data importation (session 3) and processing (session 4)

Import data into R and start data steps on a sample dataset

Data processing (session 5) and summarization (session 6)

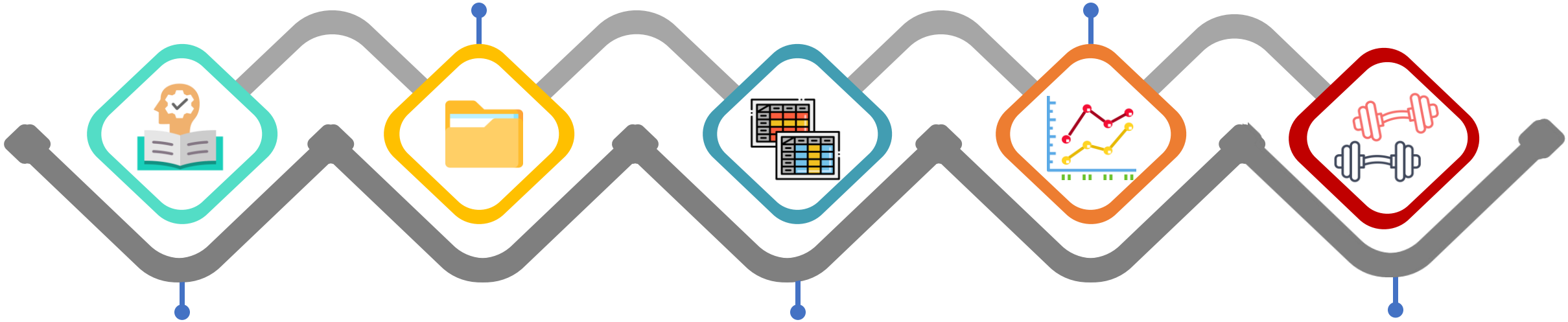
Continue data processing and start summarizing clean data into tables

Data summarization (session 7) and report generation (session 8)

Creating graphs using ggplot2 package. Automated report generation with RMarkdown

Final wrap-up exercise

Apply the acquired knowledge on a case-study dataset to generate report



HEALTH FOR ALL BY ALL
a call for solidarity and action

World Health Organization
REGIONAL OFFICE FOR THE Eastern Mediterranean

Session's structure

01

Presentation

(20 min)

02

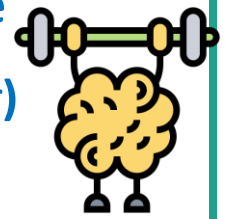
Live Demo

(30 min)

03

Exercise

(1.5 - 2 hr)



04

Quick debrief

(20 min)

HEALTH FOR ALL BY ALL
a call for solidarity and action

World Health Organization
REGIONAL OFFICE FOR THE Eastern Mediterranean

Data used during training sessions

- **Day 2-4:** we will use the cholera line list for 108 records/observations and 53 variables

  (data dictionary and case reporting form (CRF) are available in your training folder)

- **Day 5:** we will use the monkeypox line list for 427 records/observations and 76 variables

  (data dictionary and case reporting form (CRF) are available in your training folder)

Overview of R software

Session 1 agenda

- 9:00 – 9: 20 (20 min): **Introduction and Training Objectives**
- 9:20 – 10:20 (60 min): **Presentation "Overview of R software"**
- 10:20 – 11:00 (40 min): **Group photo/ coffee break**
- 11:00 – 11:30 (30 min): **Demonstration**
- 11:30 – 12:40 (70 min): **Practice/Exercise**
- 12:40 – 13:00 (20 min): **Quick debrief/ Q&A**
- 13:00 – 14:00 (60 min): **Lunch**
- 14:00 – 15:00 (60 min): Presentation "Packages and functions"
- 15:00 – 15:30 (30 min): Demonstration
- 15:30 – 15:50 (20 min): Stretching / coffee break
- 15:50 – 16:50 (60 min): Practice/Exercise
- 16:50 – 17:10 (20 min): Quick debrief/ Q&A

Outline

- Why R and RStudio
- R vs. RStudio
- R script Vs. R console
- R terminologies
- Objects
- Operators
- Some helpful references

Why R

- Powerful tool for data management and analysis
- Free, open-source software
- Reproducible workflow
- Compatible with all operating systems
- Fast
- Flexible
- Popular
- supported by a large community

RStudio

- User-friendly interface for R
- Makes coding easier
- incorporates RMarkdown that helps create automated reports in different formats (docx, PDF, HTML,…)

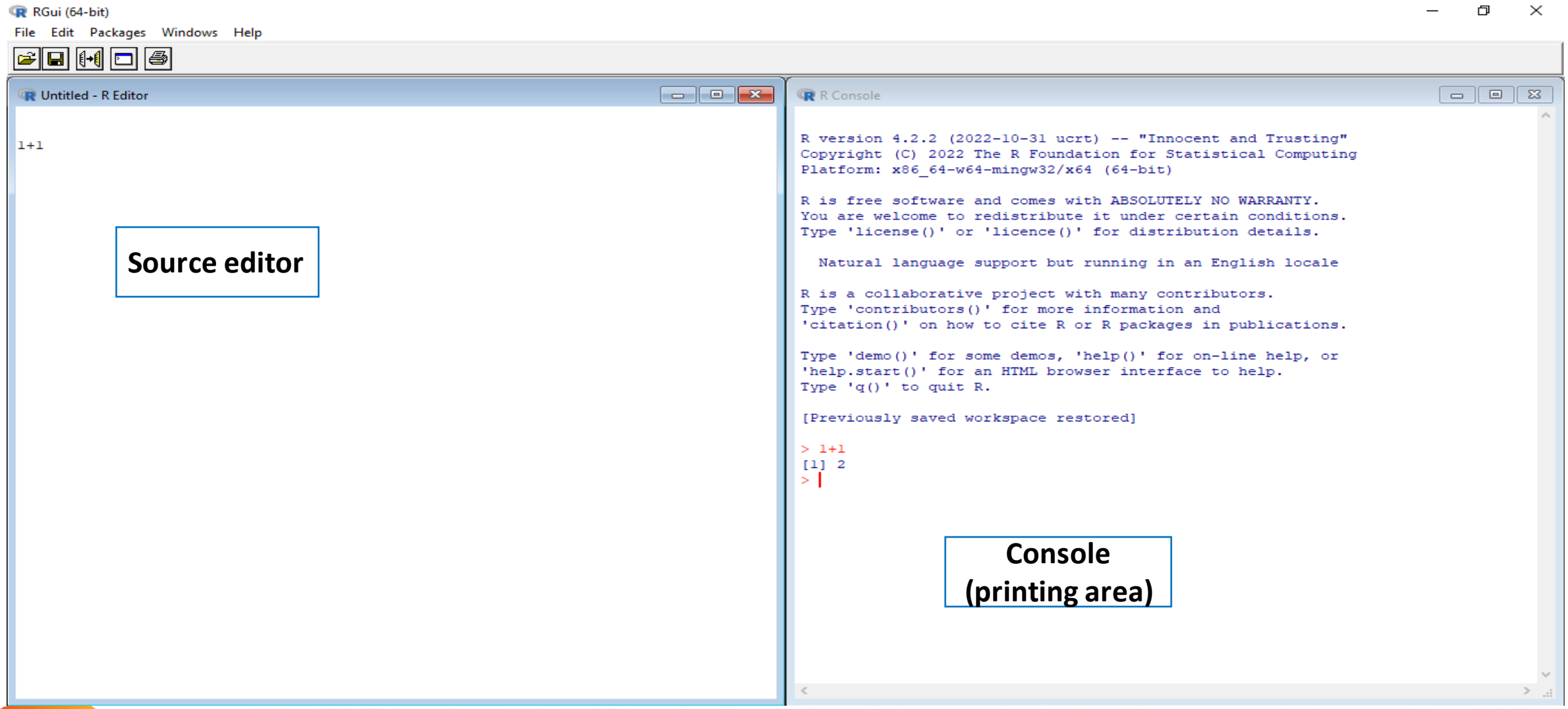
R and RStudio

R: Engine

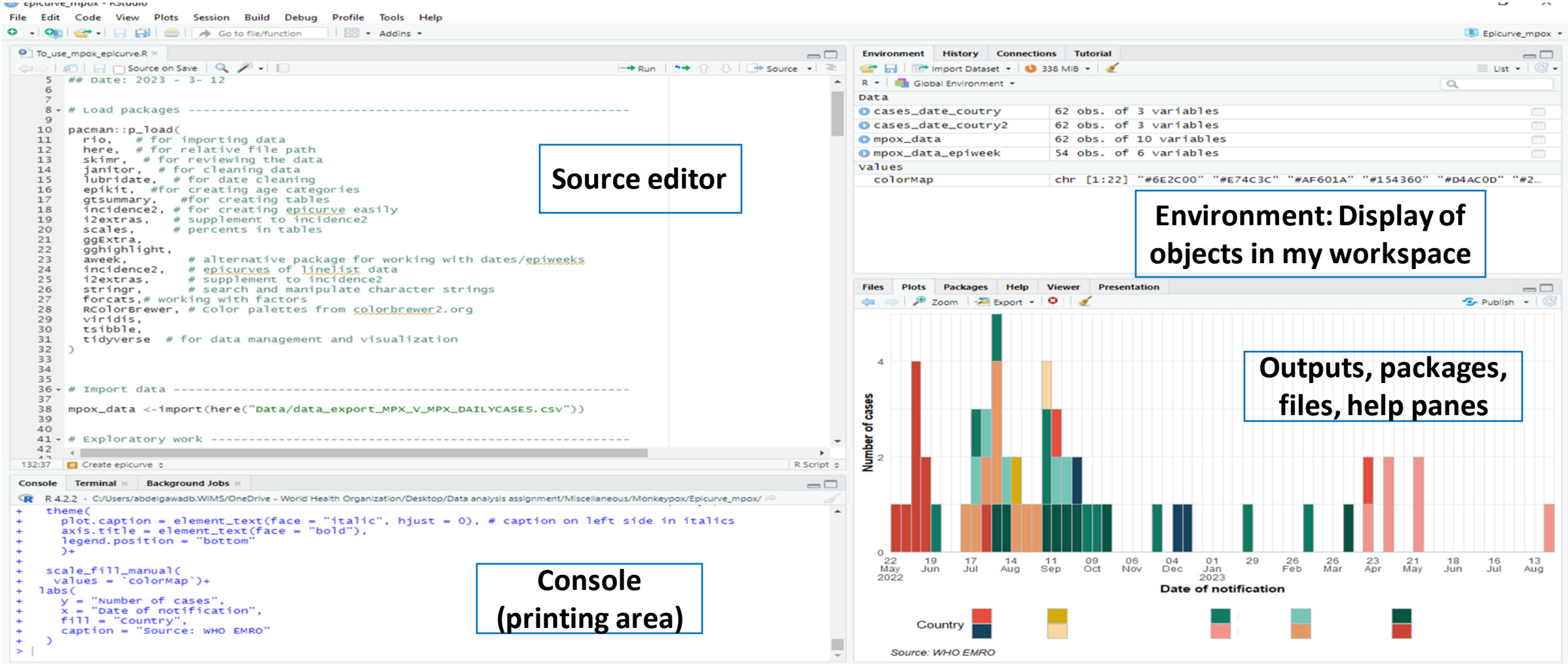
RStudio: Dashboard

Image source: https://community.rstudio.com/t/differentiating-r-from-rstudio/
Image source: https://moderndrive.com

R interface


RGui (64-bit)
File Edit Packages Windows Help
Untitled - R Editor
1+1
Source editor
R Console
R version 4.2.2 (2022-10-31 ucrt) -- "Innocent and Trusting"
Copyright (C) 2022 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (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.
[Previously saved workspace restored]
> 1+1
[1] 2
>
Console
(printing area)

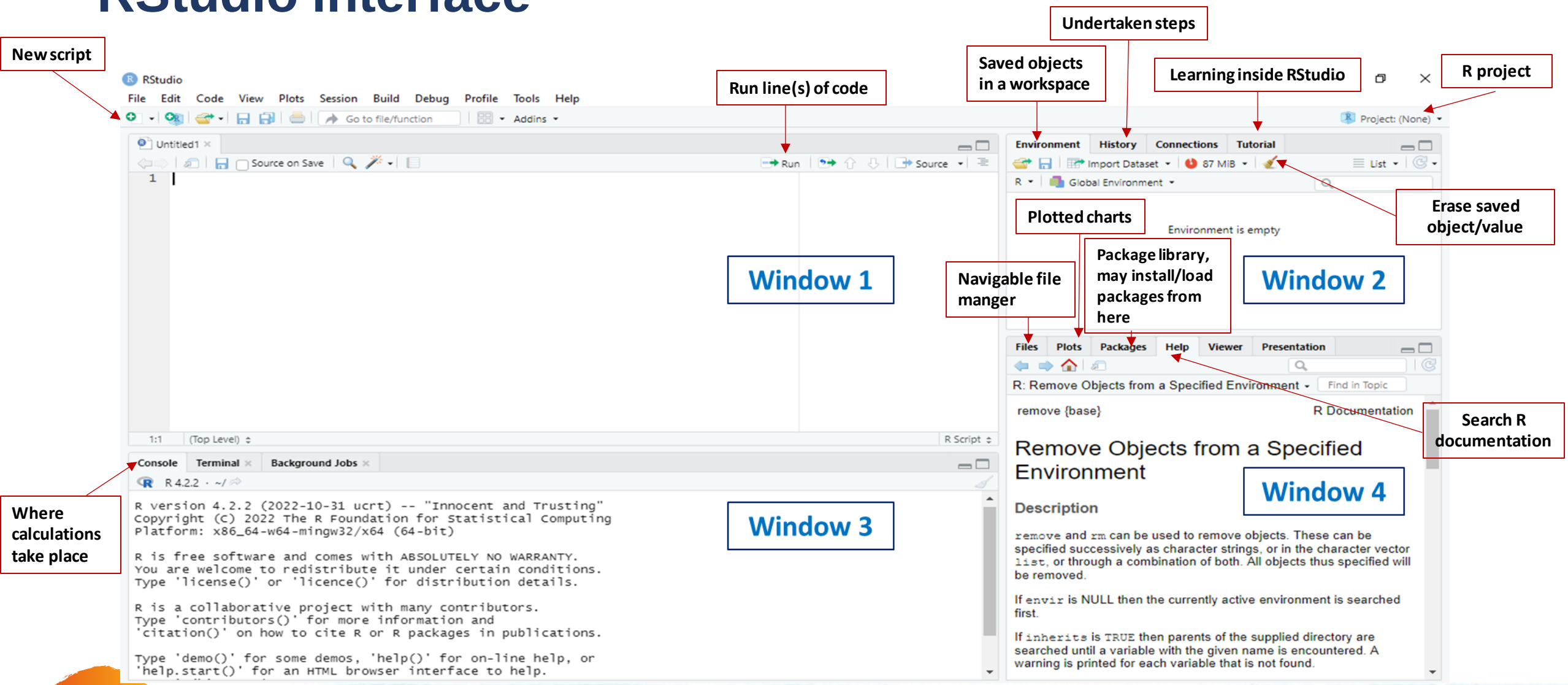
Rstudio interface



HEALTH FOR ALL BY ALL
a call for solidarity and action

World Health Organization
REGIONAL OFFICE FOR THE Eastern Mediterranean

RStudio interface



RStudio terminologies

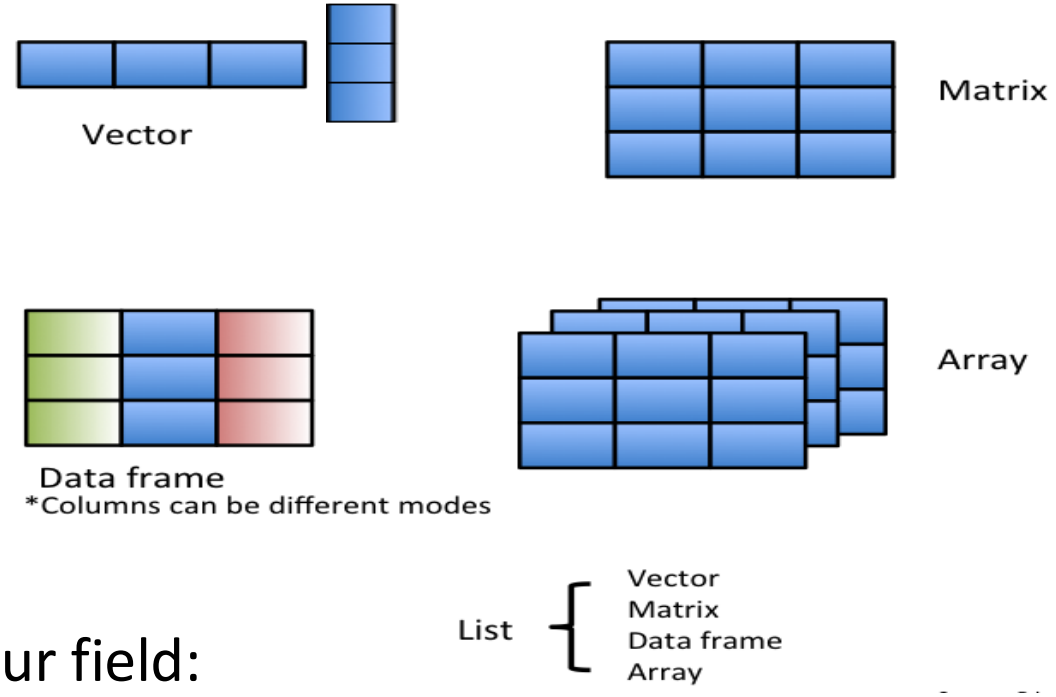
Term	Meaning
Syntax	Code (specific combination of words and symbols)
Script	Most used (where we type and save our commands for reproducible analysis and sharing with others)
Console	where calculations take place (can't save commands, make corrections)
Function	Helps do a definite task
Package	Pack of functions
Assignment operator (= or <-)	Tell R to assign certain value(s) to an object
Pipe (%>%)	And then (pass one piece of code to the other)
Data frame	Dataset
Vector	Set of values of the same class (numeric, date, string,…)
Object	The saved data type (value, data frame, list, matrix)
File path	Your file address in your computer
Working directory	Where your work in R will be saved (it would be where your R is installed, if not set otherwise)

Objects

- Objects are anything you store in R (data frame, value, table, graph,..) and can be used for reference
- Object_name <- certain value
- Name of object:
  - Begin with a letter (start with a number, will return an error)
  - Case-sensitive
  - Easy understand
  - Contain no spaces, use "_" or "."
  - Can be over-written, just re-run your code
  - Avoid naming with function names (reserved keyword)

```
> x <- 5^2
> x
[1] 25
> EMR <- "Eastern Mediterrenean Region"
> EMR
[1] "Eastern Mediterrenean Region"
> EMR <- "Estern_Mediterrenean_Region"
> EMR
[1] "Estern_Mediterrenean_Region"
>
```

Structure of objects



- Vector
- List
- Data frame
- Matrix
- Array

We will focus here on what is commonly used in our field:

- Vector: like a variable column in a dataset, a sequence of values/ data elements of the same type/ class (one-sided object).
- Date frame: like dataset, Rows are observations and columns are different variables (vectors) of different data types. Each variable (vector) includes values of the same type. (2- dimensions)

Image source: http://r.qcbs.ca/workshop01/book-en/manipulating-objects-in-r.html

Classes of objects

- Numeric (with decimals e.g. 2.345)
- Integer (no decimals e.g. 2, 4,..)
- Character (string, wrapped in “ ” or ‘ ‘)
- Factors (ordinal, has specific order)
- Date
- Logical (TRUE/FALSE)
- Data frame

Operators in R

Operator	Description
+ , - , * , /	Arithmetic operators
** or ^	exponent
==	Equal to (returns a logic)
=	Assignment operator
!	Not
! =	Not equal (returns a logic)
> , >=	Greater than or equal (returns a logic)
<, < =	Less than or equal (returns a logic)
&	And
|	Or
$	Indexing vector(column) or value

Errors & warnings

Before command execution

Your command will not be executed

(alerts you before running the command, mostly a forgotten parenthesis or extra comma)

```
35
36 # Import data ------------------------------------------------------------
37
38 mpox_data <-import(here("Data/data_export_MPX_V_MPX_DAILYCASES.csv")
39
40
41 # Exploratory work ----------------------------------------------------------
42
43 class(mpox_data$DATEREP)
44
45 sum(mpox_data$NEW_CONFCASES)
46
```

After command execution

Your command will not be executed

(occur in the console. read it carefully, it will give you a clue about the solution or where is it)

```
> mpox_data <-import(here("Data/data_export_MPX_V_MPX_DAILYCASES.csv"))
Error in import(here("Data/data_export_MPX_V_MPX_DAILYCASES.csv")) :
  could not find function "import"
```

After command execution

Your command will be executed

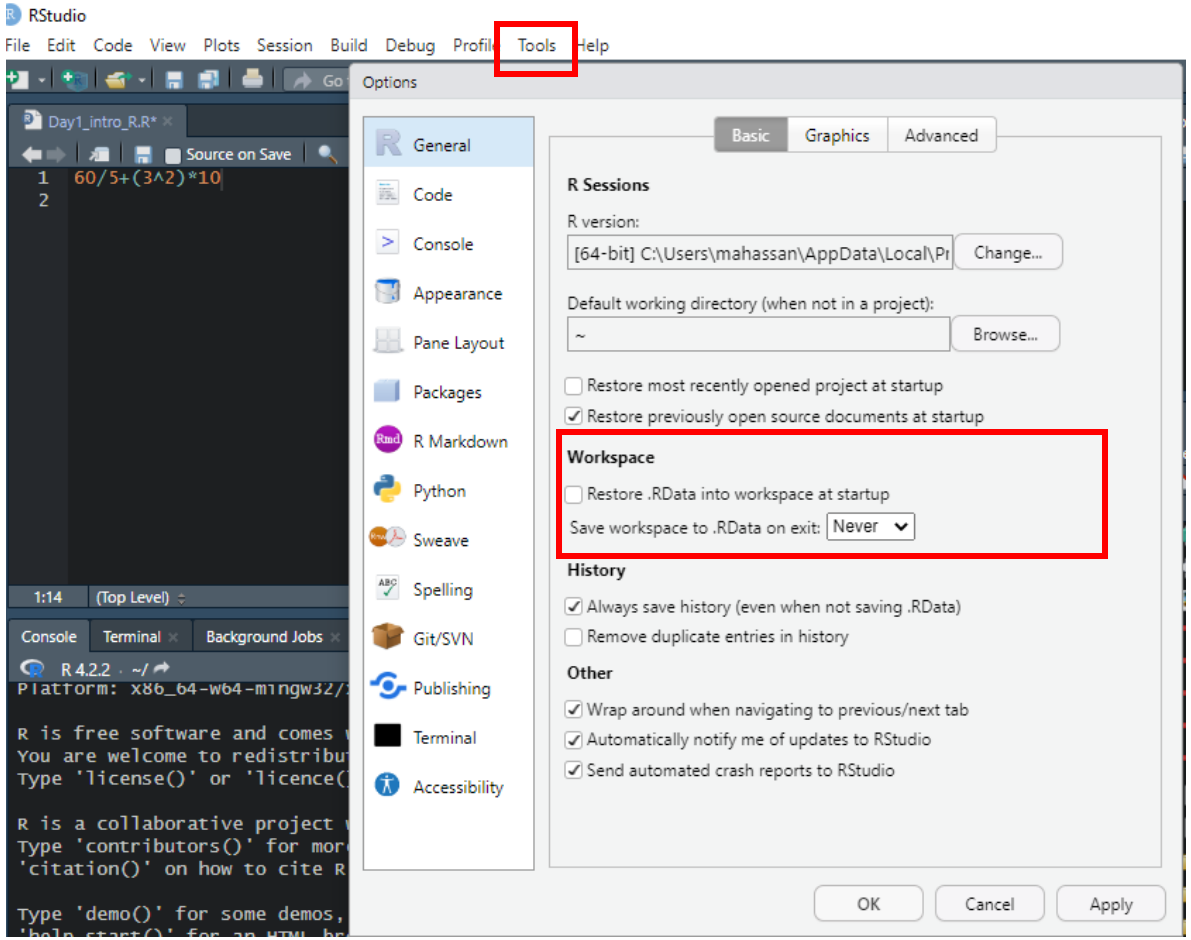
(occur in the console. read it carefully, it will give you a hint about the problem)

```
> library("dplyr")

Attaching package: 'dplyr'

The following objects are masked from 'package:stats':

    filter, lag

The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union

Warning message:
package 'dplyr' was built under R version 4.2.3
```

Best practice: Onetime Rstudio configuration

- **Objective:**
- Ensure a clean working environment for each new session
- Prevent confusion from loading outdated data or objects automatically

- Click Tools >> Global options >> Workspace;
  - un-check the box “Restore .Rdata into workspace at startup”,
  - make “save workspace to .Rdata on exit” as Never
- >> click Apply



Learning R Vs. R Knowledge Rollercoaster…

Image source: Stats-illustrations

HEALTH FOR ALL BY ALL
a call for solidarity and action

World Health Organization
REGIONAL OFFICE FOR THE Eastern Mediterranean

How to made it …

+

+

HEALTH FOR ALL BY ALL
a call for solidarity and action

World Health Organization
REGIONAL OFFICE FOR THE Eastern Mediterranean

Some helpful references

- Learn inside RStudio (tutorial tab, ilearn package), in R (swirl package)
- Posit cheat sheets. https://posit.co/resources/cheatsheets/
- The epidemiologist R handbook. https://epirhandbook.com/en/index.html
- The R4epis training materials. https://r4epis.netlify.app/training/
- Help pane in Rstudio (may type ?function_name in console)
- Package documentation. https://cran.r-project.org/web/packages/
- Popular R communities e.g. stackoverflow
- Google search engine!

Demonstration

Exercise: Basic Configuration and Introductory Commands

- Open Rstudio and do the following configuration step to Rstudio:

Click Tools >> Global options >> Workspace; **un-check** the box "Restore .Rdata into workspace at startup", make "save workspace to .Rdata on exit" as **Never** >> click Apply

- open a new script, name it "Day1_intro_R" by saving it
- Write the following command in the console and run it [***Hint:*** click Enter] : **60 / 5 + (3^2) * 10**
- Write the same command in the source and run it [***Hint:*** Run button, Ctrl + Enter]
- Try to run part of the command in the console and script!! [***Hint:*** by highlighting the needed part]
- Assume we have an outbreak in three districts (district1, district2, district3), each reported 1000, 500, and 9dis1999 confirmed cases, respectively
  - Assign the values of the confirmed cases to three objects (**dis1**, **dis2**, **dis3**)
  - Create a new object "**total_confirmed_cases**" which is the sum of cases in the three districts
  - Create a new object "**total_population**" of a value of 1,000,000
  - Calculate the attack rate per 100,000 by dividing "**total_confirmed_cases**" / "total_population" * 100,000

📌 **Do Not Forget to make use of # to add relevant comments as needed!**

Packages and functions

Session 1 agenda

- 9:00 – 9: 30 (30 min): Welcome
- 9:30 – 10:30 (60 min): Presentation **“Overview of R software”**
- 10:30 – 11:00 (30 min): **Demonstration**
- 11:00 – 11:20 (20 min): **Stretching / coffee break**
- 11:20 – 12:40 (80 min): **Practice/Exercise**
- 12:40 – 13:00 (20 min): **quick debrief/ Q&A**
- 13:00 – 14:00 (60 min): **Lunch**
- 14:00 – 15:00 (60 min): **Presentation “Packages and functions”**
- 15:00 – 15:30 (30 min): **Demonstration**
- 15:30 – 15:50 (20 min): **Stretching / coffee break**
- 15:50 – 16:50 (60 min): **Practice/Exercise**
- 16:50 – 17:10 (20 min): **quick debrief/ Q&A**

Outline

- Packages
- Functions
- Install and load of packages
- Best practices

Packages in R

- Packages are the hidden power of R
- Each package includes a list of functions
- Each package has a publication date and a compatible R version
- As R is a community-driven software, many useful packages are created and made available to the public every day (currently ~ 20,000 packages in CRAN)
- You can even make your own package and share it with R community!
- You can download any number of packages and load only those relevant to your analysis at any time (*we will learn more about installing and loading packages later this session!!*)

Functions in R

- it takes a certain input (dataset or variable) and processes it to a desired output

Functions

{base}R

- Simple functions
- Built-in in R (no need to install any package)

Installed packages

- Need to install/load the relevant package

Function syntax

Function_name (argument1, argument2, ….)

- Mostly function name is relevant to what it is supposed to do
- Each function has its parentheses and encompassing set of arguments separated by “,”:
  - 1st argument is the data (data frame/vector) you want to process
  - Arguments have names (like data=), you may use or not (it is your coding style)
  - Some arguments are mandatory, and others are optional (no need to write) … *similar to other software!!*
  - The good news is that even optional arguments (if not supplemented) have a default option

Simple {base}R functions

Function	Description
min(x, na.rm= TRUE)	Minimum value
max(x, na.rm= TRUE)	Maximum value
sum(x, na.rm= TRUE)	summation
summary()	Statistical summary of numeric vector
str()	Description of data types of vector(s) in data frame
class()	Class of an object
mean(x, na.rm= TRUE)	Arithmetic mean of set of values
median(x, na.rm= TRUE)	Median of values
range(x, na.rm= TRUE)	Range of given values
round(x, digits=)	Round value to specific decimal points
as.character, as.numeric, as.Date	Specify the type/class of the variable

Simple {base}R functions

```
> patient_age <- c(20, 60, 55, 4, 11, 39)
```

Function	Description	Example
min(x, na.rm= TRUE)	Minimum value	`> min(patient_age)` `[1] 4`
max(x, na.rm= TRUE)	Maximum value	`> max(patient_age)` `[1] 60`
sum(x, na.rm= TRUE)	summation	
summary()	Statistical summary of numeric vector	
str()	Description of data types in data frame	
class()	Class of an object	`> class(patient_age)` `[1] "numeric"`
mean(x, na.rm= TRUE)	Arithmetic mean of set of values	`> mean(patient_age)` `[1] 31.5`
median(x, na.rm= TRUE)	Median of values	`> median(patient_age)` `[1] 29.5`
range(x, na.rm= TRUE)	Range of given values	`> range(patient_age)` `[1] 4 60`
round(x, digits=)	Round value to specific decimal points	
as.character, as.numeric, as.Date	Specify the type/class of the variable	

Install vs. load packages

Images sourced from https://www.wikihow.com/Change-a-Light-Bulb

Installing and loading packages

R built-in options

- install.packages() is base R function for package installation
- library() is base R function for package loading

```r
# install package outbreaks(done once)
install.packages("outbreaks")
# load the installed package (every session)
library(outbreaks)
```

Installed package

- Install package to control the install/load process in R
- Here we introduce the pacman pacakage and its p_load function
- pacman::p_load(): saves a lot of code line

```r
# it will install &/or load already installed packages
pacman::p_load(outbreaks)
```

Best practices

- Save your work in R project (.Rproj)
- Arrange your R project into folders (Data, Scripts, Figures, Outputs,…)
- Write and save your code in R script (.R)
- Arrange your script into sections (load packages, import data, data processing,..)
- Script to include only codes for analysis (no unnecessary codes)
- Do not forget to use comments (starts with # symbol) to write notes or to highlight something
- Use short meaningful object names
- R is case-sensitive, so take care while naming your objects

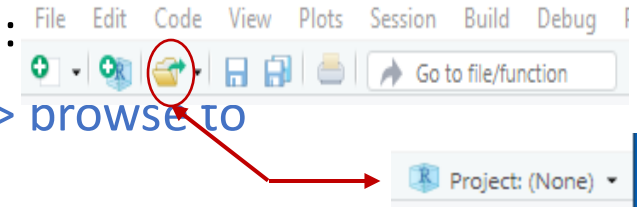
Shortcuts in RStudio

Operator	Shortcut
Assignment operator (<-)	Alt + -
Pipe (%>%)	Ctrl + shift + m
New section in script	Ctrl + shift + r
New script	Ctrl + shift + n
Run line(s) of code	Ctrl + enter
New chunk in RMarkdown	Ctrl + Alt + i
Comment (#)	Ctrl + shift + c

Demonstration

Exercise: Project Creation and Basic Data Exploration

- Open Rstudio and start by creating an R project for your training folder as follows: create it by clicking on one of the two options (side photo) >> existing directory >> browse to training folder >> create project "Regional_R_training.Rproj"



- Create a new script and save it in your **script folder** to the name "base_functions"
- We will use one of the packages' datasets in R, **outbreaks::ebola_sierraleone_2014**, and *do the following:*
- Install package "outbreaks" using install.packages() function
- *Load package "outbreaks" using library() function*
  - assign ebola dataset (ebola_sierraleone_2014) to object "**cases**"
  - Start exploring the loaded dataset:

```
> str(cases)
> summary(cases)
> class(cases)
> class(cases$date_of_onset)
```

💡 ***Cases <- oubreaks::ebola_sierraleone_2014***

Hint: ***R is case-sensitive!***

Exercise cont.

- Calculate the mean, median, and range of ebola cases' age
  - mean(cases$age)
  - median(cases$age)
  - range(cases$age)
  - You may use summary() instead

Bonus!!

- Get the earliest date of onset
- Get the date of the last collected sample

Hint: ***R is case-sensitive!***

Install and load packages used in the training

Please copy and paste the following lines of code to a new section “load packages”:

```
pacman::p_load(
      data.table,
      rio,
      here,
      dplyr,
       epikit,
      janitor,
       lubridate,
        ggplot2,
      crosstable,
        stringr,
        gtsummary,
      flextable,
      Hmisc,
      scales,
      incidence,
      tidyverse )
```

Looks like

```
# Load packages ----------------------------------------------------
pacman::p_load(
  data.table,          # fast to load large datasets
  rio,                 # to import/export different types of data
  here,                # set relative path to project root

  dplyr,               # data processing and manipulation
  lubridate,           # date manipulation
  Hmisc,               # label variables
  janitor,             # data cleaning + quick tables
  epikit,              # age categories
  stringr,             # working with string variables

  crosstable,          # generate tables
  gtsummary,           # generate tables
  flextable,           # enhance table visualization

  ggplot2,             # data visualization
  scales,
  incidence,           # epidemiological curve

  tidyverse            # data management
)
```

HEALTH FOR ALL BY ALL
a call for solidarity and action

World Health Organization
REGIONAL OFFICE FOR THE Eastern Mediterranean