AN INTRODUCTION TO R FOR POLICY ANALYSIS

Giles Dickenson-Jones

COURSE OBJECTIVES

Designed to:

- **Provide a gentle introduction** to R Programming
- **Be practically focused**
- **Show how R can be useful** in your work
- **Help you learn how to learn R**

Not designed to:

- **Cover the theory** of statistics and policy analysis; or
- **Make you an expert** at using R

Why free?

- To test if programming really is an underappreciated skill in public policy
- To share our nerdy passion

A LITTLE ABOUT ME

- **Started career as economist in government** working on issues such as climate change policy, international taxation and tax stimulus measures during the GFC.
- **Passion for *everything*** made me an 'applied generalist' that has worked as in government, consulting, thinktanks and International NGOs on issues ranging from road engineering to the economics of peace
- **Understanding how economics works in hard places** has resulted in spending half of career working overseas (incl. Myanmar, the Philippines, Cambodia and Nepal).
- **I wanted to learn a flexible tool for data** I could use regardless of the sector, industry or problem
- **I learned R via an online Data Science Specialization** that was developed by John Hopkins University's (on coursera.org)

COURSE SCHEDULE:

Webinar each Monday @ 10am (AEST) ~ 20 July to 17 August

- Week 1 ~ **An introduction to programming**, the R language and Rstudio
- Week 2 ~ **Doing stuff with data** - Importing, Exploring and Summarizing Data
- Week 3 ~ **Making pixels pretty** (1) - Data Cleaning, Merging and Basic Visualization
- Week 4 ~ **Making pixels pretty** (2) - Data Cleaning and Visualization
- Week 5 ~ **Bringing together the R programming pipeline**

WORKSHOP 1 OUTLINE:

1. Course intro
2. The logic of programming
3. Why & When R?
4. An intro to R and R studio
5. R in practice: Swirl 'basic building blocks'
6. Elements, Objects and packages in R
7. R in practice: Titanic survivor Scenario
8. Workshop survey

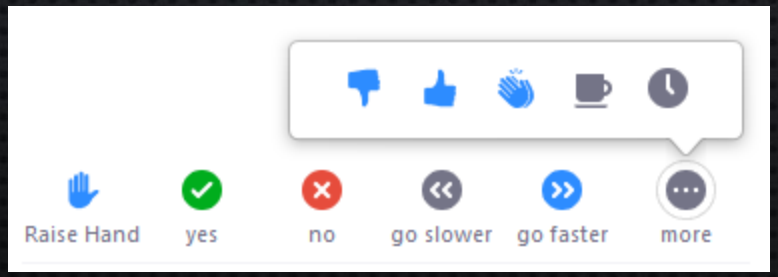
ZOOM TIPS:

First time using zoom to deliver course (so appreciate your patience!)

To give feedback as we go:

- Go slower | Go faster
- Like | Dislike
- Yes | No ~ to respond to questions from host

Raise hand ~ if you need direct assistance and we'll do our best to help



PROGRAMMING IN THE WILD

The 5 Dollar Solution to Budget Transparency in Myanmar

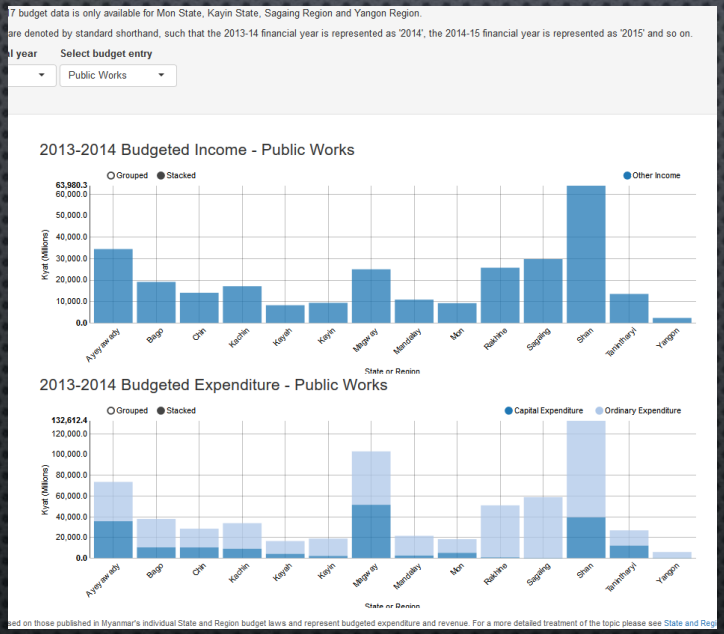

@GilesDJones

A CRITICAL LINK BETWEEN CITIZENS AND GOVT

HOW MANY PEOPLE ACCESSED BUDGET INFORMATION IN BURMA/MYANMAR:

Or

The Asia Foundation



Open Myanmar Initiative

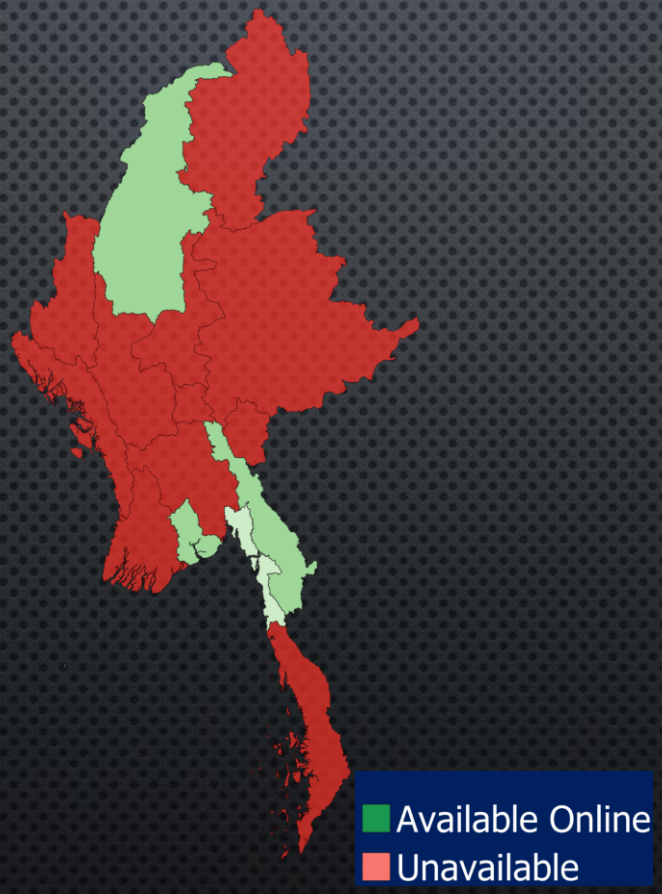
STATISTICS WITHOUT BORDERS

Statisticians in compassionate service.

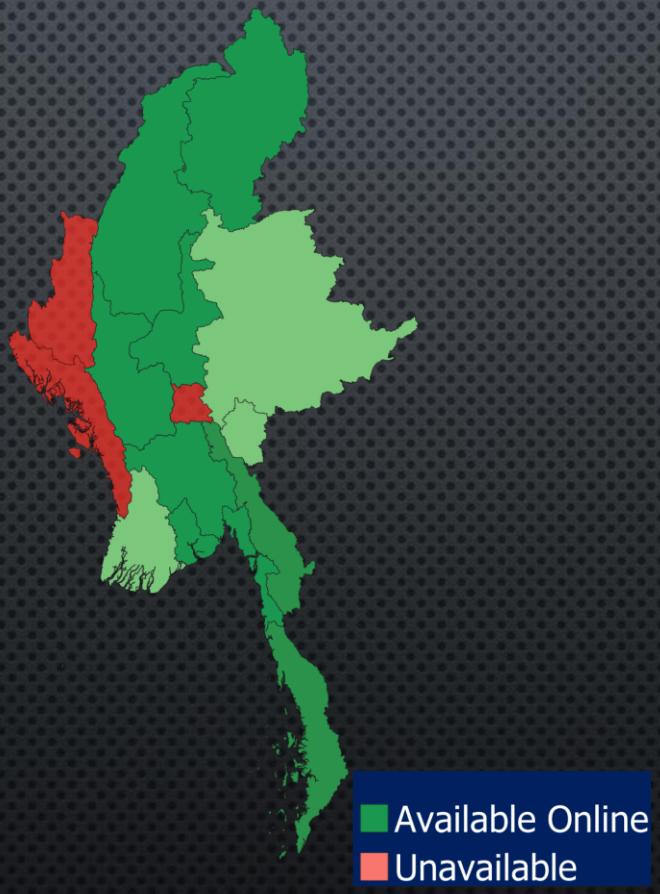
R

BUDGET DATA AVAILABLE ONLINE BEFORE LAUNCH:

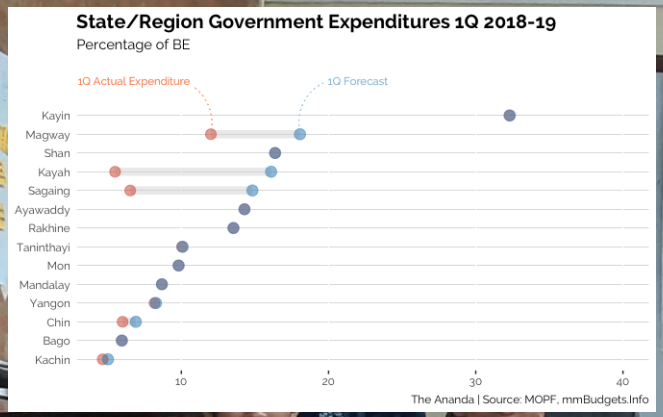
BUDGET DATA AVAILABLE ONLINE AT LAUNCH:

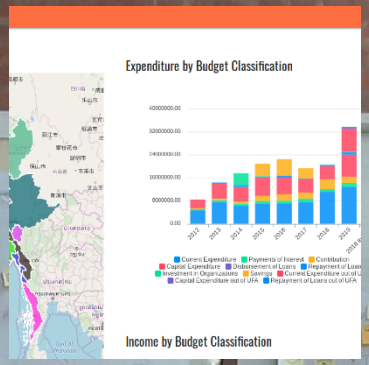


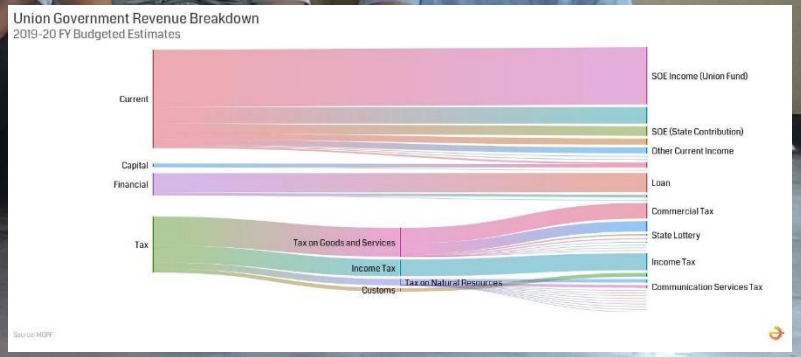
BUDGET DATA AVAILABLE ONLINE ONE YEAR AFTER LAUNCH:



NOT JUST NUMBERS ON A SCREEN...







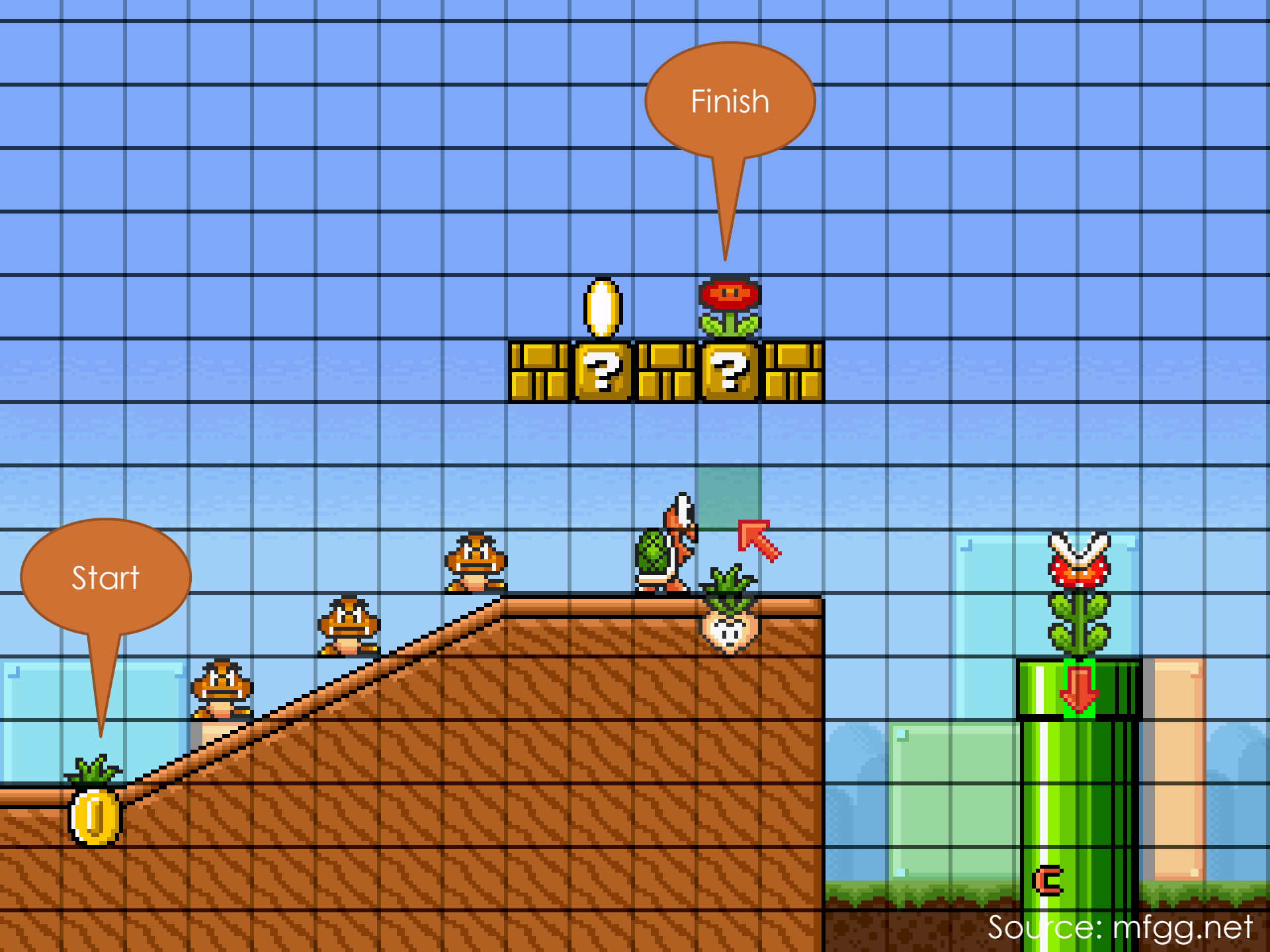
ESSENTIAL LOGIC OF PROGRAMMING

ESSENTIAL LOGIC OF PROGRAMMING

R Programming is about being able to split big problems into smaller logical steps to get to our desired result.

Start by:

1. Define the goal you want to achieve;
2. Understand the steps needed to get there; and
3. The order the steps must be taken to work


Finish
Start
Source: mfgg.net

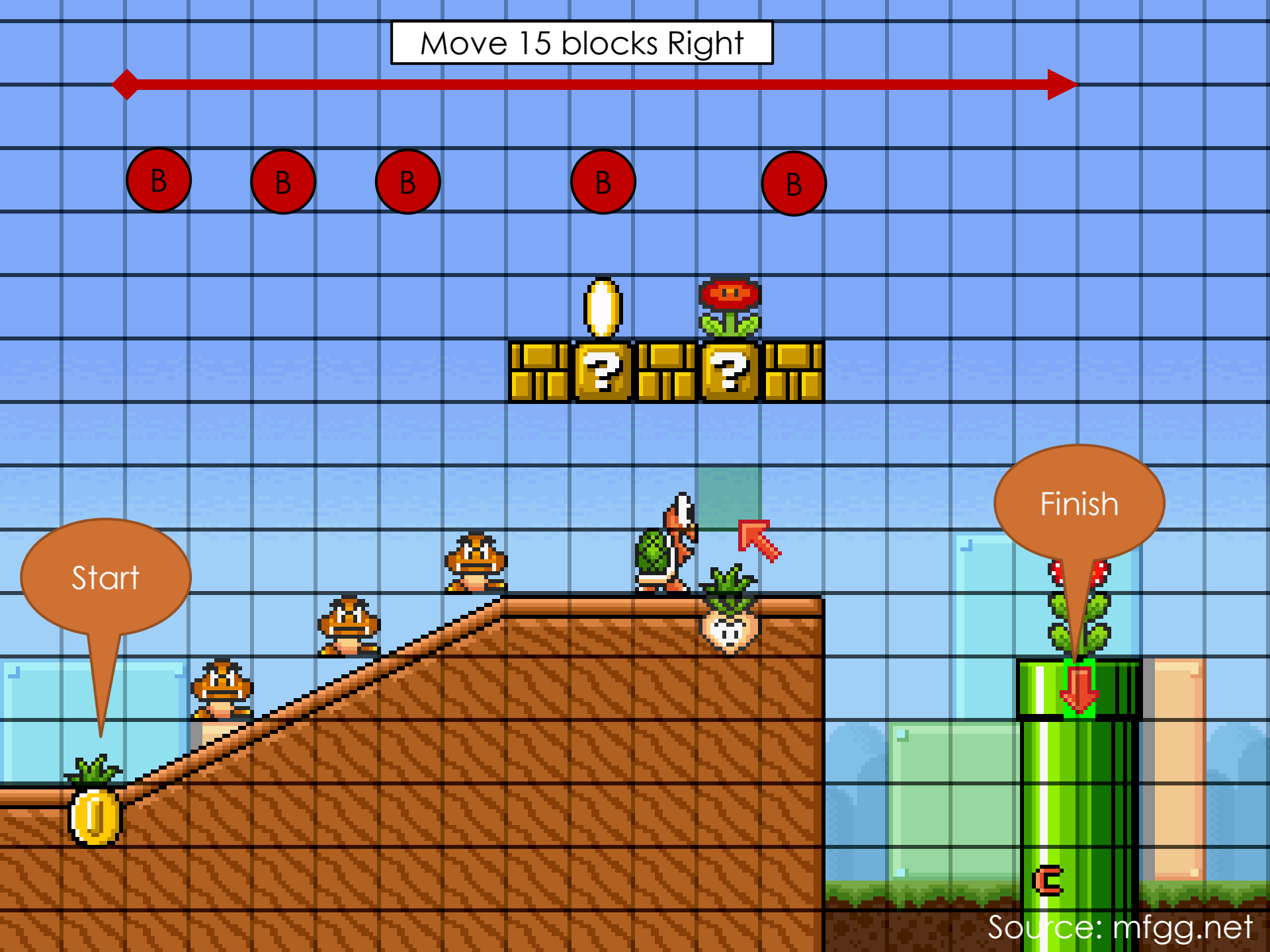
PROGRAMMING WITH MARIO

- **Move:**

  Up | Down | Left | Right

- **Action:**

  B = Jump


Move 15 blocks Right
B
B
B
B
B
Start
Finish
Source: mfgg.net

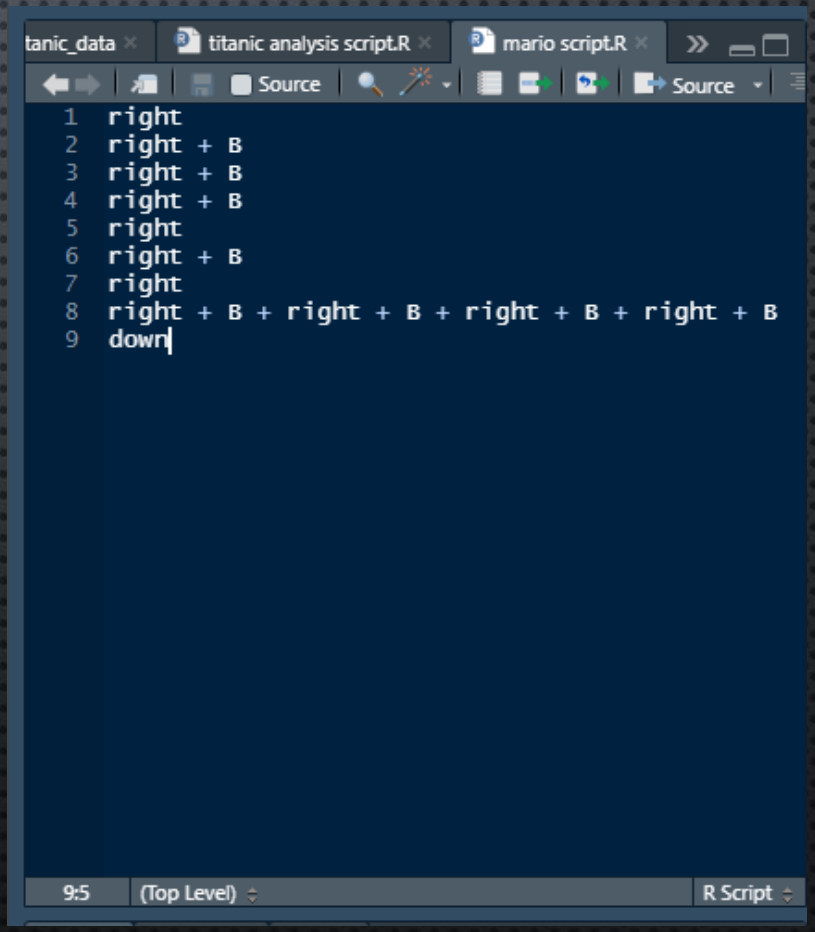
AS A PROGRAMING SCRIPT

To get where we want to be we use:

- the right action(s)
- in the right order
- at the right time

Which is why we need to:

- Start with the goal in mind
- Identify the functions we need
- Know when and how to use them
- Set these out step by step so even a computer can understand

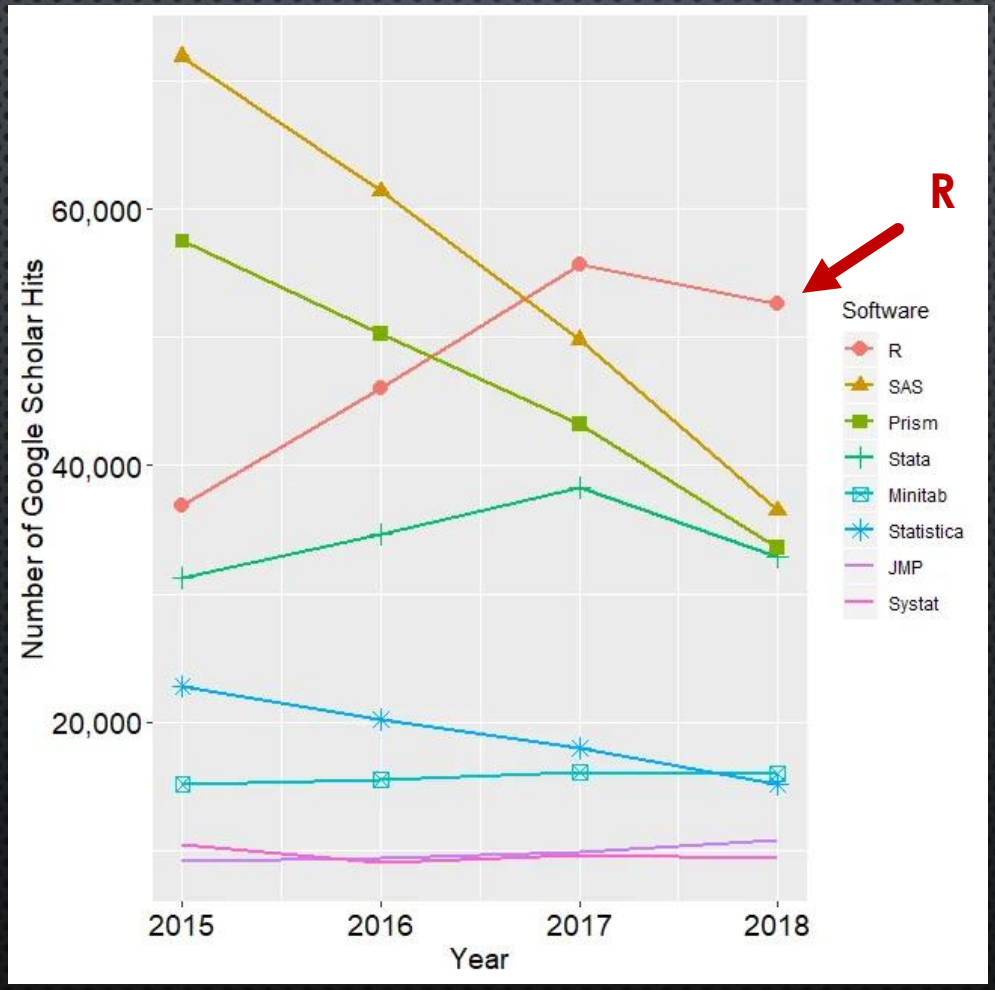


WHY R?

WHY R?

- **Powerful, flexible and fast :** It is powerful, used across sectors and can save you time.
- **Compatibility:** It plays well with other software like Excel, Stata and SPSS.
- **Portable:** available for free on mac, windows and Linux so you can take it anywhere throughout your career.
- **Popularity:** It is one of the world's most popular and in-demand statistical languages.
- **Transparency and Community:** It's free and help is always at hand with a large community of users.

GOOGLE SCHOLAR HITS BY SOFTWARE:

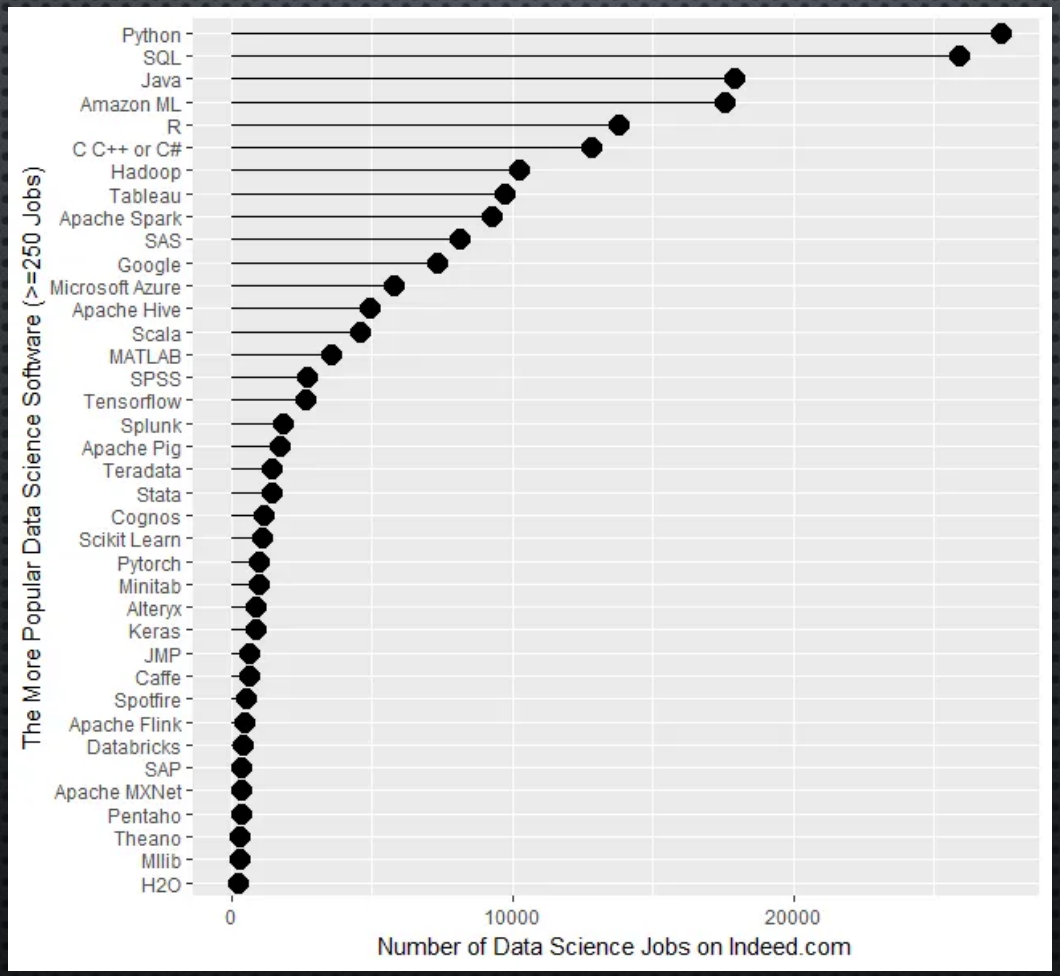


http://r4stats.com/articles/popularity/

2019 DATA SCIENCE JOBS BY SOFTWARE:

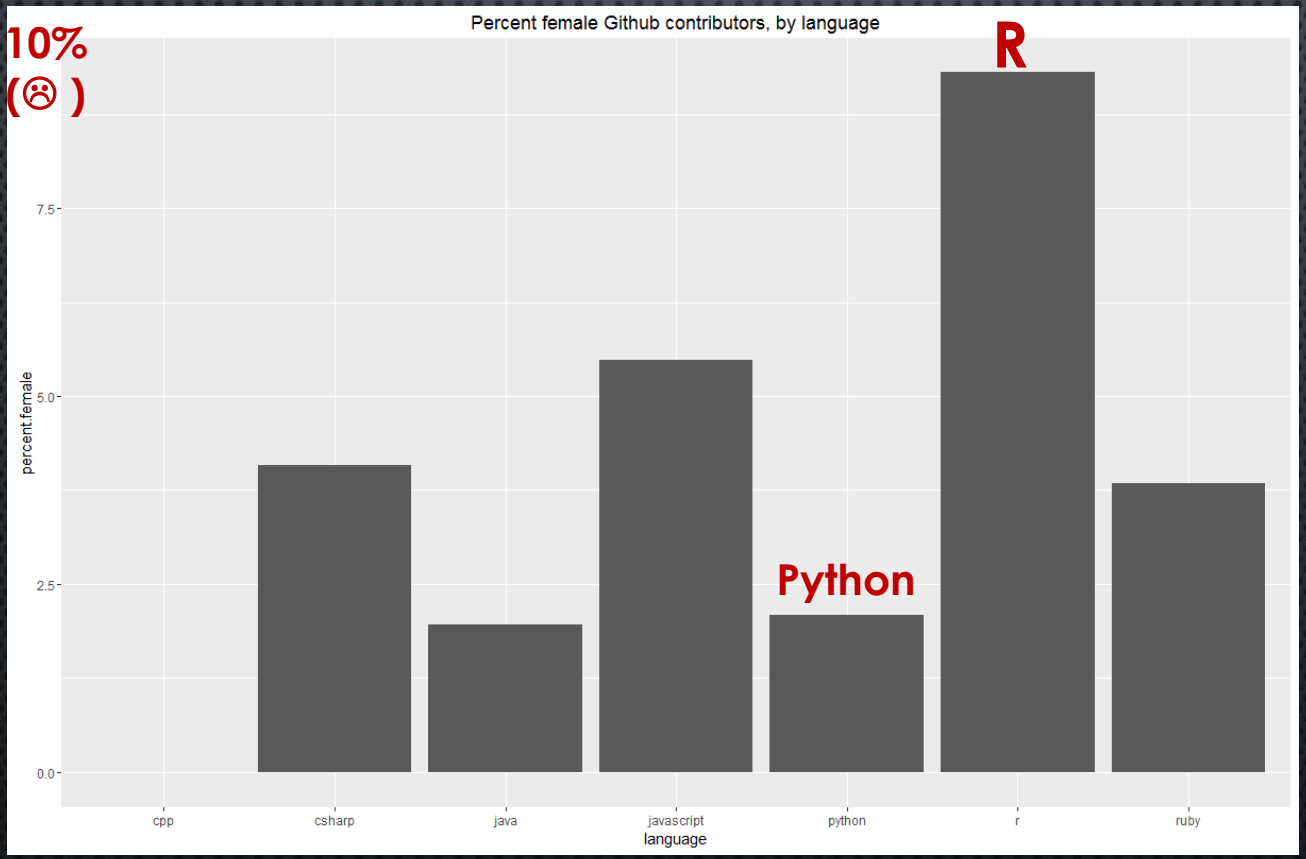
Python →

R →



http://r4stats.com/articles/popularity/

% OF FEMALE GITHUB CONTRIBUTORS BY LANGUAGE (2016)

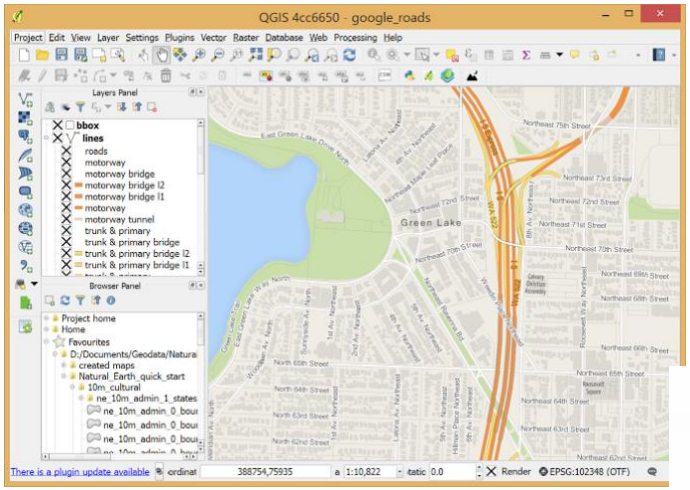


https://reshamas.github.io/why-women-are-flourishing-in-r-community-but-lagging-in-python/

WHEN R?

- R can more efficiently deal with larger datasets than traditional business tools.
- Being free and widely used, it can perform most data-related tasks.
- Being a programming language allows the automation of repetitive tasks.
- R scripts create a record of what has been done making it easier to spot errors, share analysis and reproduce results.

As R loads everything in memory you need to be strategic when the data becomes big (> 6gb).



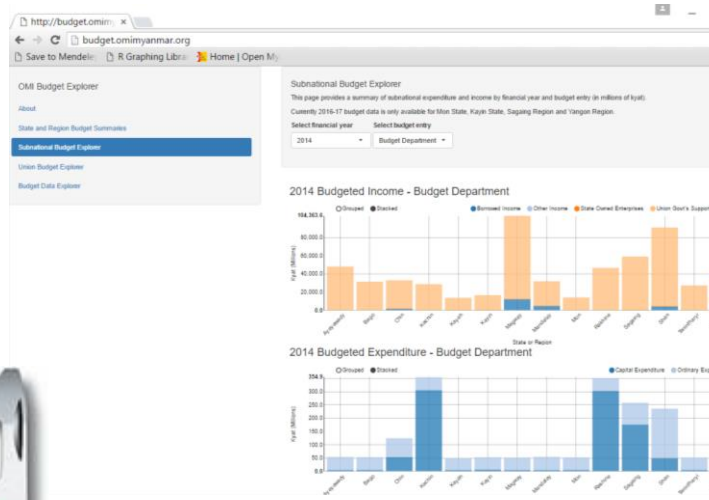
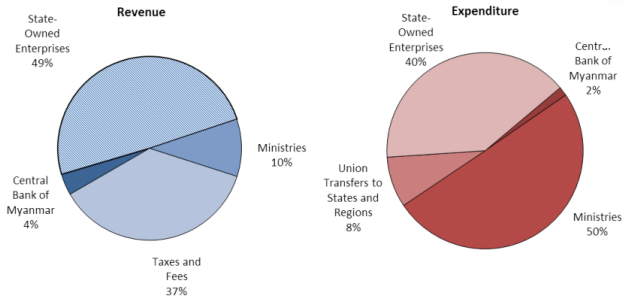
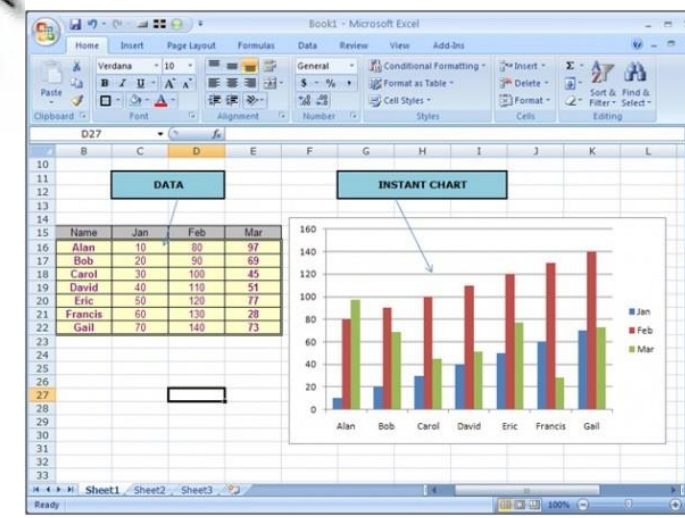


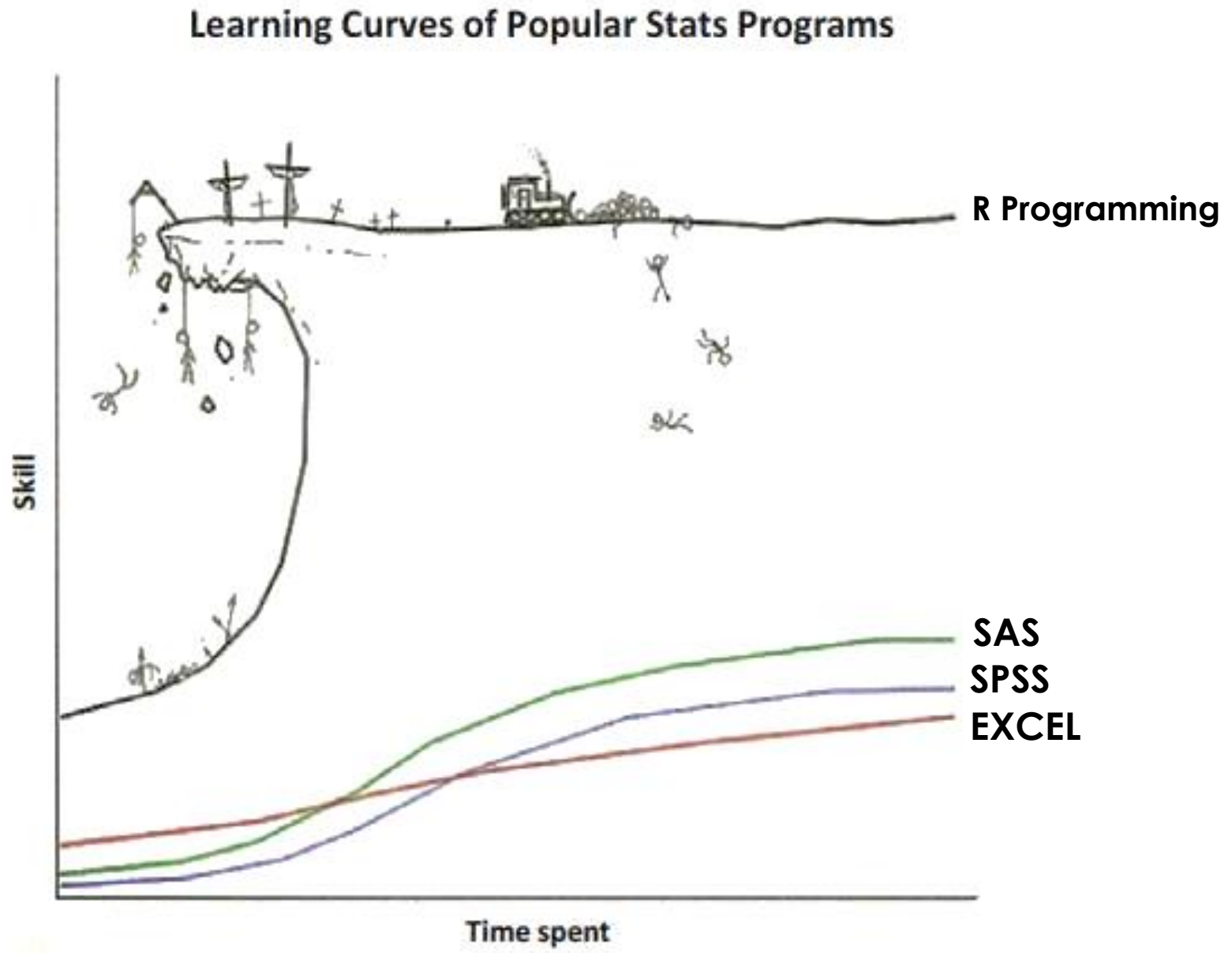
Figure 5: Major Union Revenue and Expenditure Sources 2016-17[11]



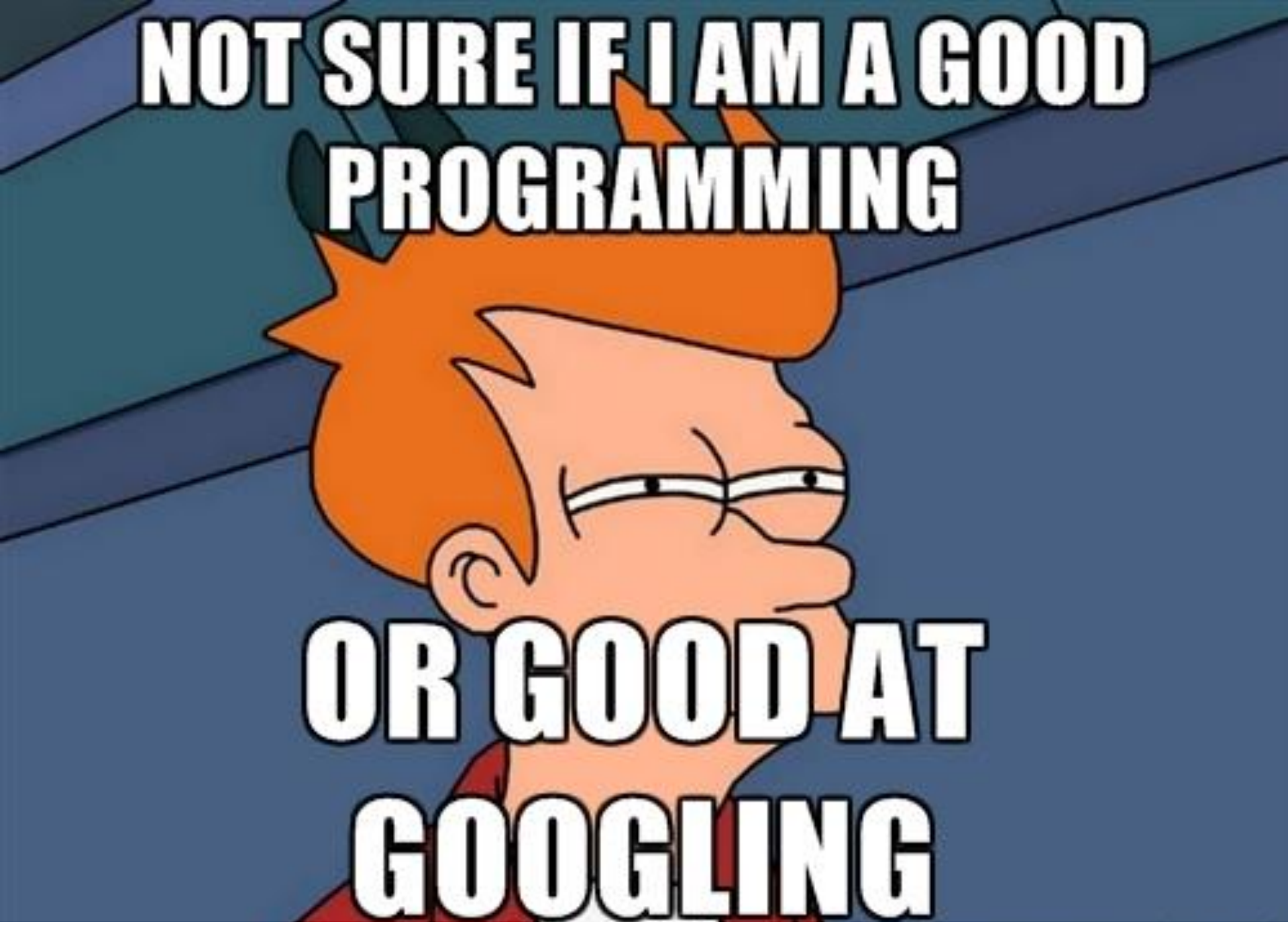
Source: Myanmar Budget Laws 2016-17

In terms of tax administration, MOF with its Internal Revenue and Customs Department collects the majority of Union revenue. For instance, the Internal Revenue Department, collects income tax, commercial tax, stamp duty and lottery taxes, while the Customs Department is predominantly responsible for the administration of taxes on imports and exports. [12]





Based off: https://twitter.com/chrisbeeley/status/727945556128059396



https://ardalis.com/working-through-roadblocks-a-guide-for-new-programmers

R AND R STUDIO

R BASICS

R is an interactive, **object-orientated statistical programming language**

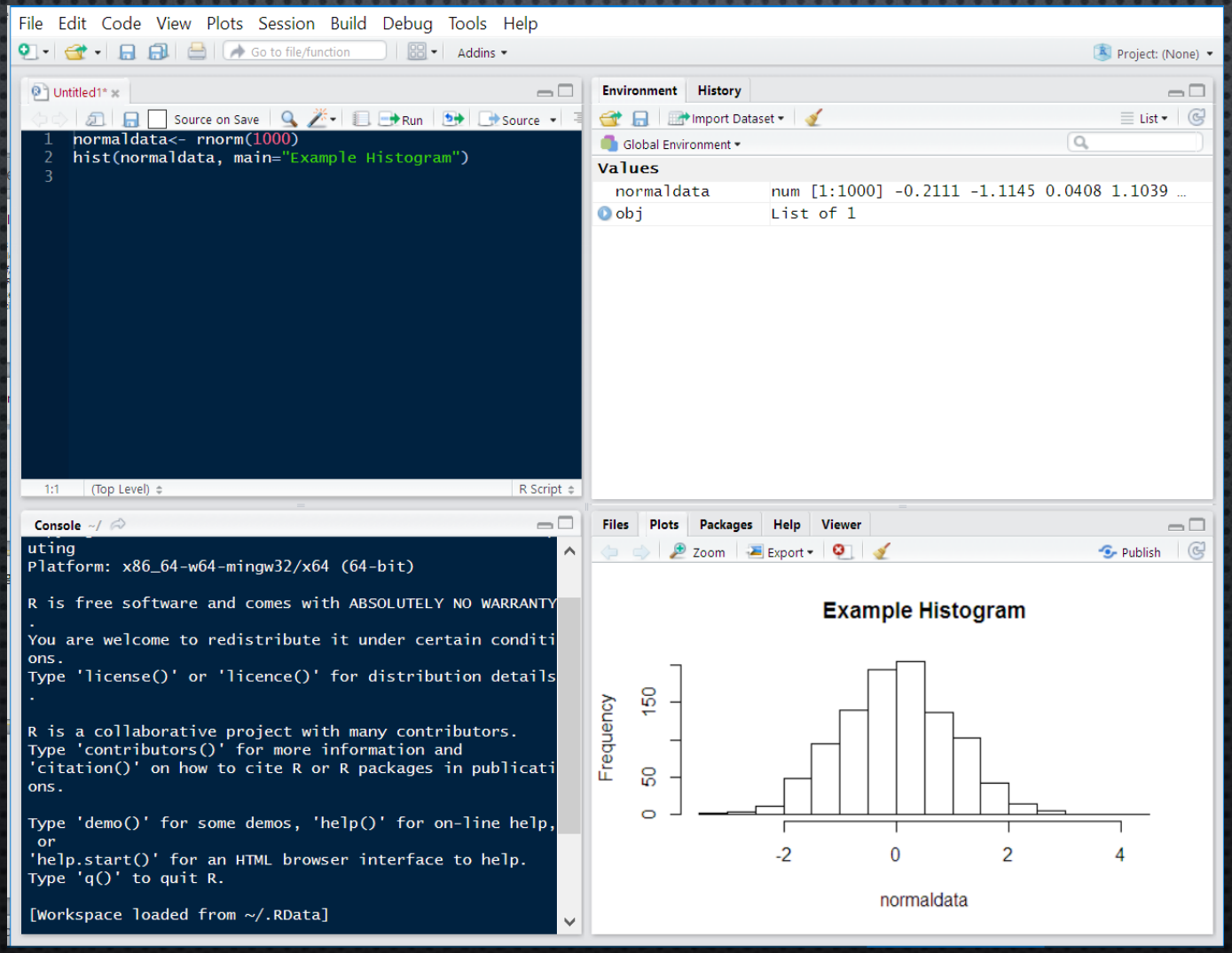
- The 'Objects' are things R work with (eg data)
- It is a language as it's a collection of words that tell a computer what to do
- It is 'interactive' as when you tell it to do something, it responds

RStudio is a 'wrapper' to R that makes working with R easier

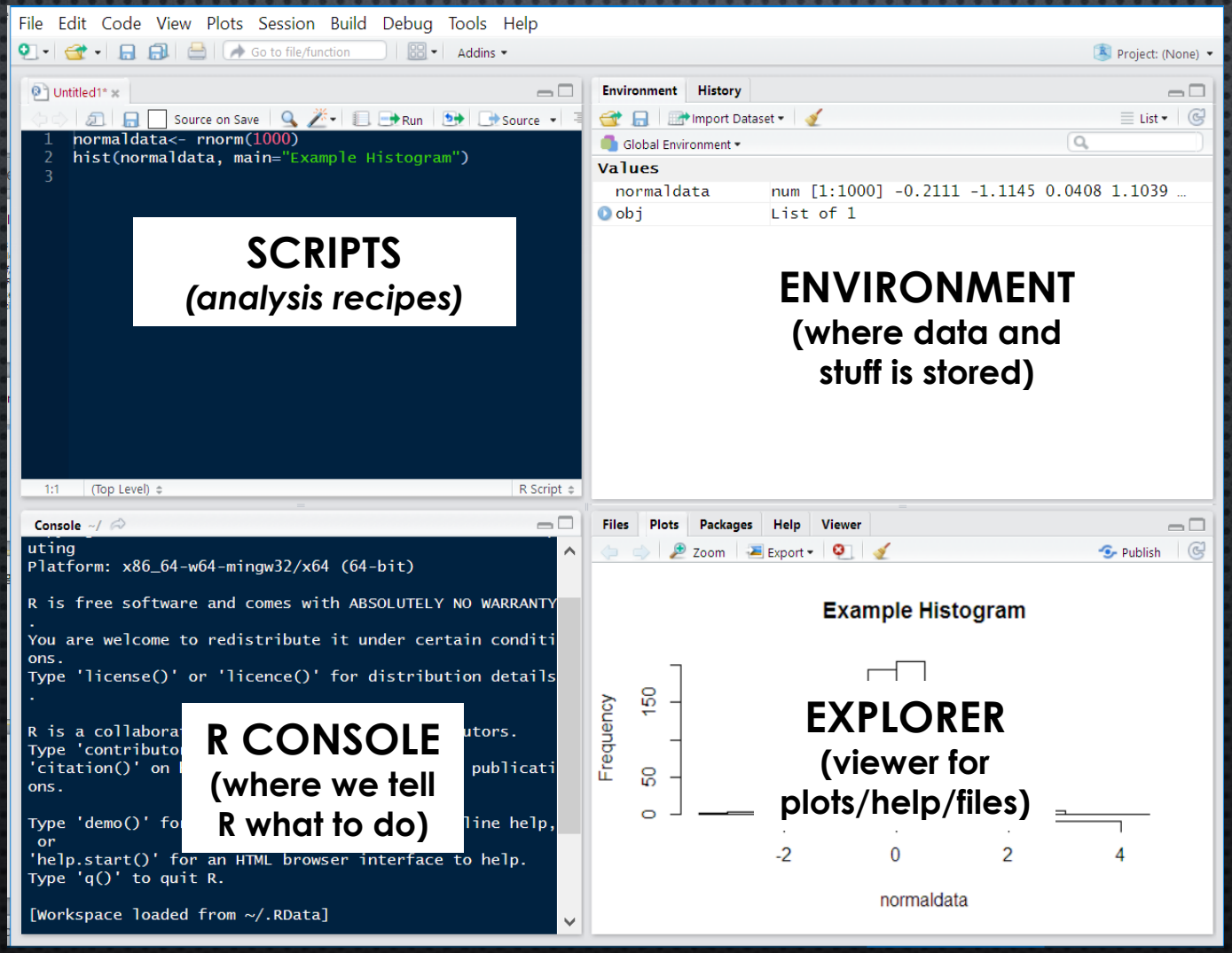
THE BASICS:

- **R console** - where R does stuff
- **R studio –** a program to help us use R
- **Scripts –** Text file with our 'analysis recipe'
- **Packages** – Groups of commands that do stuff
- **Environment** – Where R stores stuff it works with
- **Working directory** – where your project is
- **Base R** – R without loading packages

RSTUDIO HELPS US USE R:



RSTUDIO HELPS US USE R:



SWIRL EXERCISE: BASIC BUILDING BLOCKS

LEARNING R WITH SWIRL

The 'swirl' package provides a set of exercises to help you learn R within R. We'll be completing the 'basic building blocks' exercise now:

- install.packages('swirl') ~ (if you haven't installed already)
- library(swirl)
- swirl()
- type your name
- Select '1: R Programming' and '1 Basic Building Blocks'

SWIRL EXERCISE: BASIC BUILDING BLOCKS

Raise your hand if you have any questions and tell me when you're done using the chat

DATA, ELEMENT TYPES AND OBJECTS

ELEMENTS (WE CARE ABOUT)

Objects are collections of elements

- eg: stuff <- c(31,42,13)
- eg: 42 is the 2nd element in the object 'stuff'

Element types:

- **Numeric** - 1.2, 45.0, 21.6
- **Character** - "Ben" ,"Sheryl", "Bazza"
- **Dates** – '23/12/85'
- **Logical** - TRUE, FALSE
- **Factors –** numbers and text (used for labelling)

OBJECT TYPES

Objects are collections of elements

- eg: stuff <- c(31,42,13)

- **Vectors ~** one column one type of element
- **Matrices ~** multiple columns of one element type
- **Data Frames (or Tibbles) ~** multiples columns of any element type
- **Lists ~** Collection of objects (eg a bunch of data frames)

We will mainly use data frames/tibbles and vectors

COMMANDS | FUNCTIONS | PACKAGES

- **Commands:** actions we tell the computer to do
  - 1 + 1
  - print("Get off my lawn!")
- **Functions:** bunch of commands stuck together to perform a task
  - Average(1:1000)
- **Packages:** collection of functions

FUNCTIONS AND PACKAGES

- **Commands: actions we tell the computer to do**

  ```
  1 + 1
  print("Get off my lawn!")
  ```

- **Functions: a set of commands stuck together to something useful**

  such as mean(), median(), max(), min() etc

- **Packages: functions packaged together that expand what R can do**
- **Base R** = commands/functions that come with R by default

- **Tidyverse** = A collection of Core Packages that work well together

FUNCTIONS / COMMANDS

Both functions and commands require that we feed it the right ingredients to work eg:

1+1: ✔

1 + "Mary": ✘

Similarly, imagine the function:

run(where,speed,distance)

To work, run() requires that we provide it **information in the right format** on **where** to run, the **speed** and **distance**.

FUNCTIONS / COMMANDS

Importing data:

read_excel("file_name.xls") | Read_csv("file_name.csv")

- **Summary statistics:**
  - Summary(Object_name)
- **Freq-table**
  - **table(data_frame[, "variable_name_1"])**
  - **table(data_frame$variable_name_1)**
- **2 way freq-table:**
  - Table(data_frame$variable_name_1, data_frame$variable_name_2)

Saving data:

write_csv(object_to_save, "file name.csv")

GETTING DATA IN AND OUT OF R WITH TIDYVERSE:

Via 'readr':

- **Load**: read_csv; and
- **Save**: write_csv

Via the 'readxl' package:

- **Load Excel File**: read_excel

Via the 'haven' package:

- **SAS**: read_sas
- **SPSS**: read_sav
- **Stata**: read_dta
- etc

THE BASICS: SUBSETTING DATA

Vectors (use x[row])

- vector_object<-(1:100)
- 33rd item: vector_object[33]

Data frames x[row,columns]

- data_object<-data.frame(1:100, 100:1,1:100*5)
- 33rd item, in the 2nd column: Data_object[33,2]
- A range of rows and exclude a column: data_object[33:34, -3]

OR Select columns directly with $ via data frames

Eg data_object$column_name and data_object$column_name[33]

Programming in the Wild: Astrophysics and Praveen

SCENARIO: TITANIC PASSENGER SURVIVAL

Context:

On April 15, 1912, the "unsinkable" RMS Titanic collided with an iceberg and sunk, resulting in the deaths of approximately two thirds of the passengers onboard.

The Transport Minister just saw the 1997 movie 'Titanic' and wants to know how accurate James Cameron's portrayal was and what lessons the movie might have for modern maritime safety.

SCENARIO: TITANIC PASSENGER SURVIVAL

How could we answer:

- Was there a passenger named "Rose DeWitt Bukater"? What about "Jack Dawson"?
- Who was the oldest passenger on the ship? Did they survive?

(hint we can use Rstudio)

SCENARIO: TITANIC PASSENGER SURVIVAL

The Minister has Asked:

- ~~Was there a passenger named "Rose DeWitt Bukater"? What about "Jack Dawson"?~~
- ~~Who was the oldest passenger on the ship? Did they survive?~~
- **How many of each class of passenger were on board?**
- **Were higher class passengers more likely to survive?**
- **Did this change according to their gender?**

SCENARIO: TITANIC PASSENGER SURVIVAL

How many of each class of passenger were on board?

- **No. of passengers by: Class**

Passenger Class	No of Passengers
1st	?
2nd	?
3rd	?

*(**One-way** frequency table)*

SCENARIO: TITANIC PASSENGER SURVIVAL

Did this change according to their class?

- **No. of Passengers By: Class and Survival**

Passenger Class	Survived	Didn't Survive
1st	?	?
2nd	?	?
3rd	?	?

*(**Two-way** frequency table)*

SCENARIO: TITANIC PASSENGER SURVIVAL

Were higher class passengers more likely to survive?

- **No. of Passengers By: Class and Survival and Gender**

1st Class Passengers:

Gender	Survived	Didn't Survive
Male	?	?
Female	?	?

2nd Class Passengers:

Gender	Survived	Didn't Survive
Male	?	?
Female	?	?

3rd Class Passengers:

Gender	Survived	Didn't Survive
Male	?	?
Female	?	?

*(**Three-way** frequency table)*

LOAD THE TITANIC ANALYSIS SCRIPT TEMPLATE

See: "week 1" → "Policy Scenario" → "Titanic"

The script is divided into 5 sections:

- **#Set-up -** the working directory ----
- **#Load** the necessary packages ----
- **#Import, Explore + Clean** the data ----
- **#Explain** - analyse, understand and visualize ----
- **#Save -** the results ----

All the commands we need are in the script as are errors :/

THINGS TO REMEMBER:

- **set the Working directory in Rstudio using**:
  - Session → Set Working Directory
    - Hint: save your R script where your data is and select 'source file location'
- **load installed packages using *'library()'*:**
  - library('tidyverse')
- **Columns in Data Frames can be accessed using the '$'**
  - eg *titanic_data***$***age*
- **Data can be assigned to objects using '<-'**
  - *titanic_data$age_times_2* **<-** *titanic_data$age * 2*

FUNCTIONS WE NEED:

- **#Set-up ----**
  - setwd() - *tell R where to work from*
- **#Load ----**
  - library() - *load the packages we need*
- **#Import, Explore + Clean the data ---**
  - read_excel() *load the titanic data into R*
  - summary() – *see summary statistics for a data object*
  - as.logical() – *convert passenger survival column to logical*
- **#Explain - analyse, understand and visualize ----**
  - table() – *create one way, two way and three way tables*
  - plot() – *plot our results*
- **#Save ----**
  - write.csv() *– output our results to the working directory*

FUNCTIONS WE NEED:

- **#Import**
  - read_excel("file name .xls") *load the titanic data into R*
- **#Explore + Clean the data ---**
  - summary(object_name) – *see summary statistics for a data object*
  - as.logical(object_name$column_name) – *convert passenger survival column to logical*
- **#Explain - analyse, understand and visualize ----**
  - table(vector_x,vector_y…,vector_z) –
  - plot(object_we_want_to_plot) – *plot our results*
- **#Save ----**
  - write.csv(object_name, "file name.csv")

COMMON R ERRORS

R's errors can be confusing when you first start. Some common reasons for errors include:

- Misplaced brackets, commas or symbols
- When something you are referring to doesn't exist (like a file, object or function)
- When you haven't loaded the package needed for a function
- When you didn't tell the function what it needs to know or in the right order

SCENARIO: TITANIC PASSENGER SURVIVAL

The Minister has Asked:

- Was there a passenger named "Rose DeWitt Bukater"? What about "Jack Dawson"?
- Who was the oldest passenger on the ship? Did they survive?
- How many of each class of passenger were on board?
- Were higher class passengers more likely to survive?
- Did this change according to their gender?

SWIRL EXERCISES FOR THIS WEEK

Please complete before the next webinar:

- Sequences of Numbers
- Missing Values

(these are part of the 'R Programming: The basics of programming in R' course included in the swirl package)

EVALUATION FORM

Please complete the evaluation before leaving

(link shared in chat)

AN INTRODUCTION TO R FOR POLICY ANALYSIS

Giles Dickenson-Jones