ADALTAS

1. Introduction to

Petra KAFERLE DEVISSCHERE

Before we start...

- Data Scientist @ Adaltas

- petra@adaltas.com

Before we start...

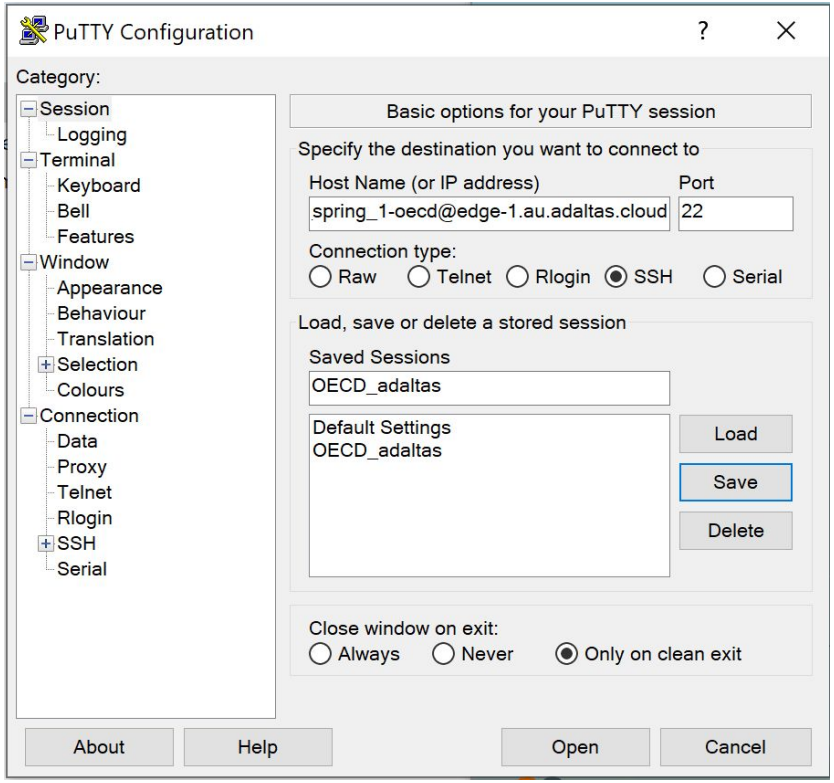
ADALTAS

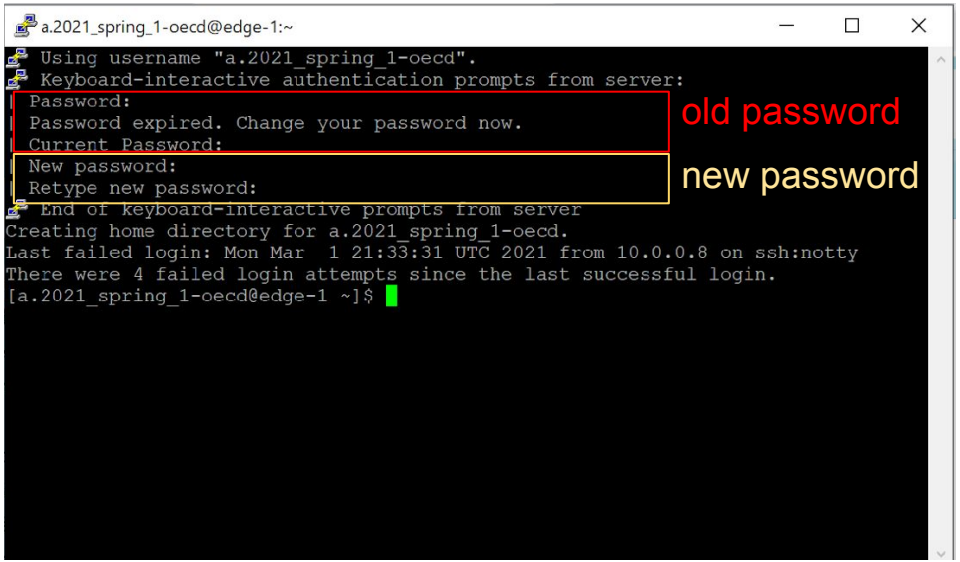
- Sessions are interactive
- 4 modules of 2x3h:
  - 9 & 11/03 - Introduction to Python
  - 16 & 18/03 - Introduction to Spark (PySpark)
  - 30/03 & 01/04 - Support session Python
  - 04 & 08/04 - Support session Spark

Let’s connect to Adaltas cluster

ADALTAS

1. Install OpenVPN
2. Open it and import .ovpn file
3. Open PuTTY (or another ssh tool)
4. When the connection is opened:
    - Type (copy) twice the password you received by mail
    - Type and confirm your new password
5. Open the browser and connect to the Zeppelin with your username and password


PuTTY Configuration
Category:
Session
Logging
Terminal
Keyboard
Bell
Features
Window
Appearance
Behaviour
Translation
Selection
Colours
Connection
Data
Proxy
Telnet
Rlogin
SSH
Serial
Basic options for your PuTTY session
Specify the destination you want to connect to
Host Name (or IP address)
Port
spring_1-oecd@edge-1.au.adaltas.cloud
22
Connection type:
Raw
Telnet
Rlogin
SSH
Serial
Load, save or delete a stored session
Saved Sessions
OECD_adaltas
Default Settings
OECD_adaltas
Load
Save
Delete
Close window on exit:
Always
Never
Only on clean exit
About
Help
Open
Cancel


a.2021_spring_1-oecd@edge-1:~
Using username "a.2021_spring_1-oecd".
Keyboard-interactive authentication prompts from server:
Password:
Password expired. Change your password now.
Current Password:
New password:
Retype new password:
End of keyboard-interactive prompts from server
Creating home directory for a.2021_spring_1-oecd.
Last failed login: Mon Mar 1 21:33:31 UTC 2021 from 10.0.0.8 on ssh:notty
There were 4 failed login attempts since the last successful login.
[a.2021_spring_1-oecd@edge-1 ~]$
old password
new password

What is Python?

ADALTAS

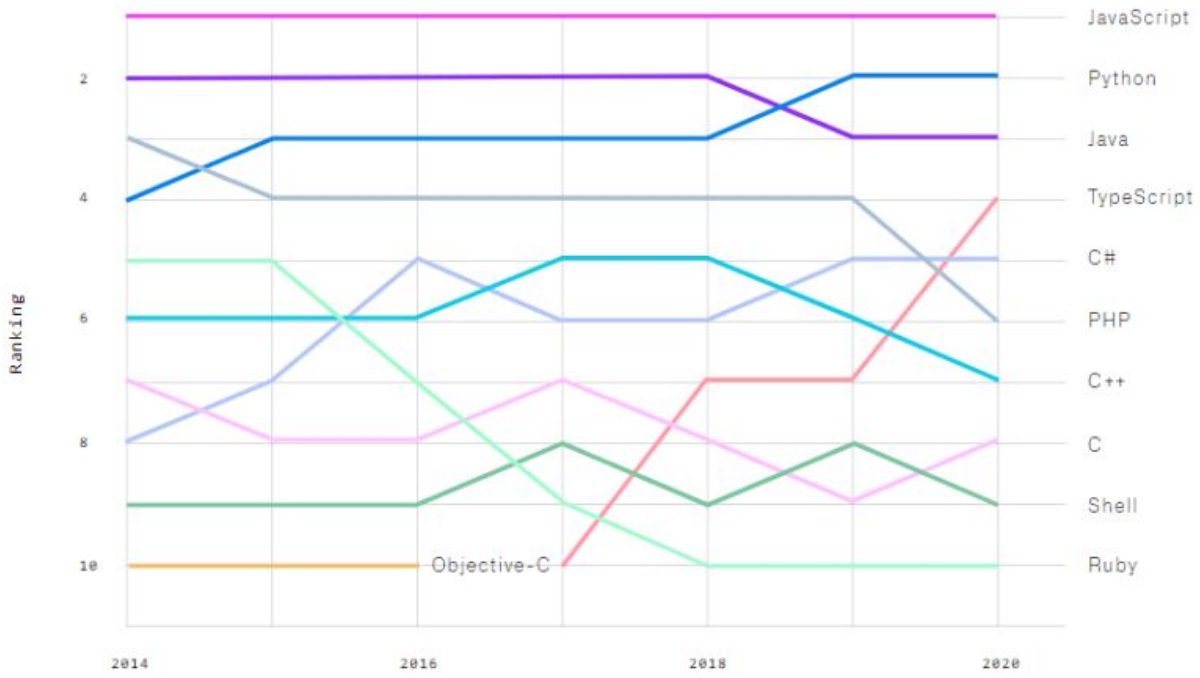
- Created in 1991 (and named after Monty Python show)
- General-purpose programming language
- Interpreted (scripting) language

Why everybody is using it?

ADALTAS

- Designed to be easy to learn -> teaching
  - readable code
- Open-source and free
- Easy to interact with
- Early adopters were Google, YouTube, NASA…
- Big community -> many libraries

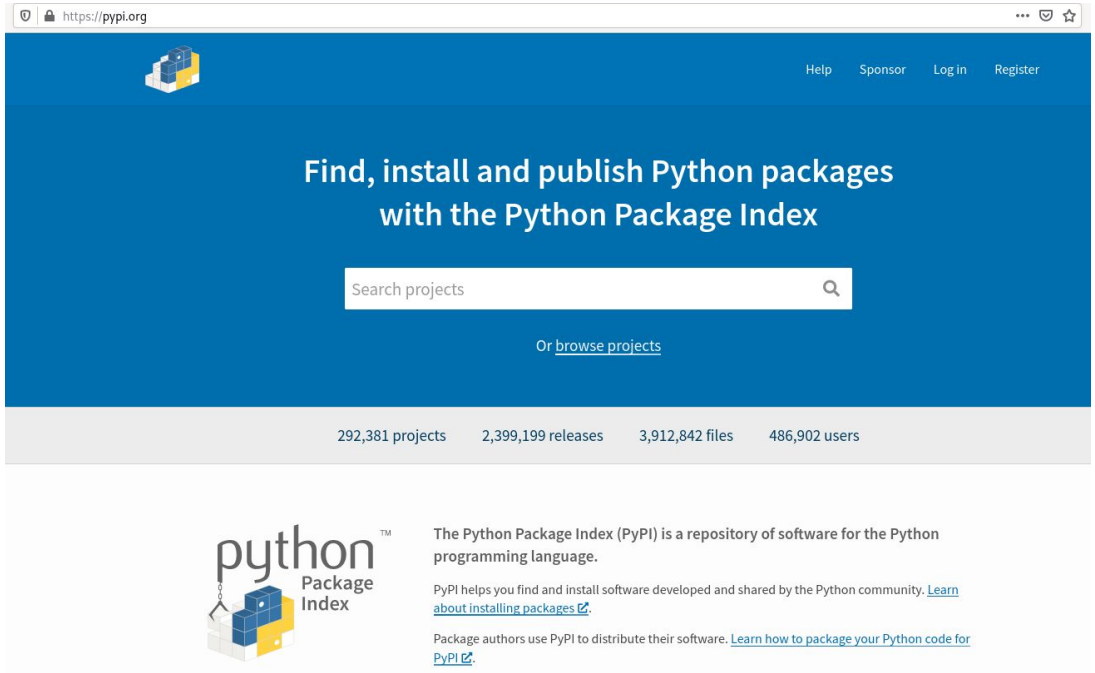
Why everybody is using it?



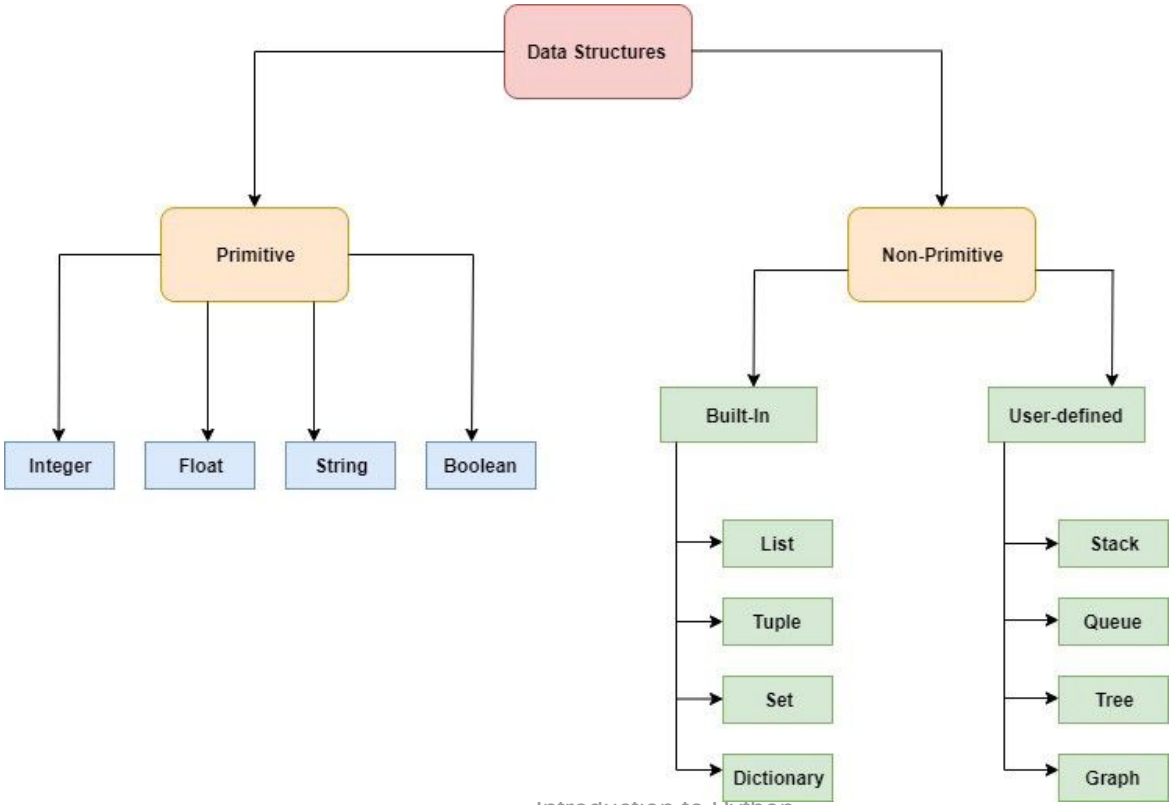
Packages (libraries)

- Collections of functionalities
- Cover a certain topic / domain
- Everybody can share a library
  - Many domains covered
  - Not verified and not always correct
- ~ 300,000 packages

Python packages



How does Python understand data?



Hands-on: First steps to Python

ADALTAS

Collections

- Lists -- mutable, ordered
  - my_list = [1, ‘test’, 5.8]
- Tuples -- immutable, ordered
  - my_tuple = (1, ‘test’, 5.8)
- Dictionaries -- key-value pairs, no order
  - my_dict = {‘petra’ : ’petra@adaltas.com’}
- Sets -- mutable, unordered, no repeats
  - my_set = {1, 3, 6, 9}

Hands-on: Collections

ADALTAS

If you want to learn more:

https://github.com/sowmya20/DataStructures_Intro

How can we manipulate data?

ADALTAS

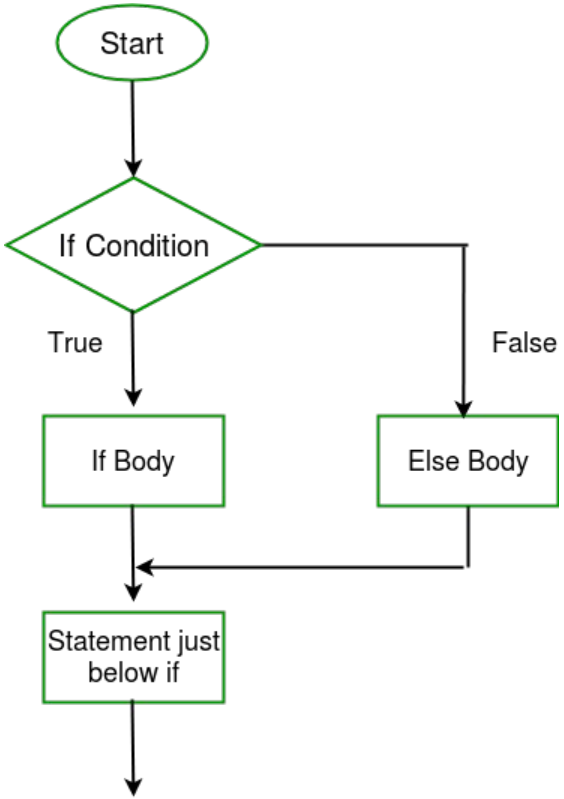
- Functions
- Code that solves a specific task
- Types:
  - Built-in: type(), print()
  - Imported from libraries: `from <module> import *`
  - Custom

Conditional statement

- Evaluates a condition and depending on the result, it executes different code

```
a = 5, b = 3
if a > b:
    b = b * 2
else:
    a = a * 2
```

- If … else
- If … elif … else



Loops

- `For`: repeat the same action n-times

```
for i in range(1, 10):
    print(i)
```

- `While`: repeat as long as condition is true

```
i = 0
while i < 10:
    print(i)
    i = i + 1
```

Boolean expressions

ADALTAS

- expressions that return a Boolean value as a result (`True, False`)
  - comparisons (>, <, =)
  - inclusions (is in)
- chaining conditions with Boolean operators: `AND, OR, NOT`

User-defined functions

ADALTAS

- When a function we need doesn't exist
- It always starts with **def**
- It can take none, one or more **arguments**
- It can **return** a value

```
def print_name(name):
    print(name)
```

```
def return_name(name):
    return(name)
```