# Example Jupyter Notebook

**The R4DS Online Learning Community**

# Table of contents

## Welcome

This is a companion for the book Practical Python Programming by David Beazley.

This website is being developed by the R4DS Online Learning Community. Follow along and join the community to participate.

This companion follows the R4DS Online Learning Community Code of Conduct.

## Book club meetings

- Each week, a volunteer will present a chapter from the book.
  - This is the best way to learn the material.
- Presentations will usually consist of a review of the material, a discussion, and/or a demonstration of the principles presented in that chapter.
- More information about how to present is available in the GitHub repo.
- Presentations will be recorded and will be available on the R4DS Online Learning Community YouTube Channel.

# Part I

# 1. Introduction to Python

## Learning Objectives

- Learn Python basics from the ground up.
- Learn Strings,Lists, Files, Functions

# 1 Example Jupyter Notebook

# 2 Introduction to Python

His webiste : https://dabeaz.com/advprog.html

> The goal of this first section is to introduce some Python basics from the ground up. Starting with nothing, you'll learn how to edit, run, and debug small programs. Ultimately, you'll write a short script that reads a CSV data file and performs a simple calculation.

- 1.2 Intro
- 1.2 First Program
- 1.3 Numbers
- 1.4 Strings
- 1.5 Lists
- 1.6 Files
- 1.7 Functions

## 2.1 More stuff from the author

> The course is taught by David Beazley, author of the

- Python Essential Reference, 4th Edition (Addison Wesley)
- Python Cookbook, 3rd Edition (O'Reilly Media).
- Distilled Python

## 2.2 1.1 Introducing Python

- Python is an interpreted high level programming language. It is often classified as a "scripting language" and is considered similar to languages such as Perl, Tcl, or Ruby. The syntax of Python is loosely inspired by elements of C programming.

### 2.2.1 How to Check Your Python Version

#### 2.2.1.1 Check Python Version: Command Line

- python –version
- python -V
- python -VV # Starting from Python 3.6 for more detailed information

##### 2.2.1.1.1 Check Python Version: Script

The sys module provides functions and variables used to manipulate different parts of the Python runtime environment.

```
import sys
print (sys.version)
```

```
3.9.9 | packaged by conda-forge | (main, Dec 20 2021, 02:38:53)
[Clang 11.1.0 ]
```

```
# print (sys.version_info)
import sys
print (sys.version_info[0])
```

```
3
```

> Python 2 is no longer under development and HAS BEEN DISCONTINUED STARTING FROM JANUARY 1, 2020.
>
> Because Python 2 and Python 3 might be installed on the same machine you might need to type python2 or python3 to pick a version.

## 2.3 1.2 A First Program

- Creation of your first program
- running the interpreter

### 2.3.1 Running Python

- Python programs always run inside an interpreter.
- The interpreter is a "console-based" application that normally runs from a command shell.

#### 2.3.1.1 Interactive Mode

- When you start Python, you get an interactive mode where you can experiment.
- If you start typing statements, they will run immediately.

#### 2.3.1.2 Creating programs

Python Programs are put in .py files.

- Run program in Terminal

```
%%bash
python greeting.py
```

```
Welcome to R4DS
```

- Run program in Jupyter

```
%run greeting.py
```

```
Welcome to R4DS
```

```
!python greeting.py
```

```
Welcome to R4DS
```

#### 2.3.1.3 Statements

A python program is a sequence of statements:

```
a = 3 + 4
b = a * 2

a+b
```

```
21
```

underscore ( _ ) holds the result og last operation

```
_ + 3
```

```
24
```

```
a = 3 + 4 ; b = a * 2
print(b)
print(a)
```

```
14
7
```

```
a , b = 3 + 4 , a * 2
print(b)
```

```
14
```

#### 2.3.1.4 Variables

```
height = 442 # valid
_height = 442 # valid
height2 = 442 # valid
#2height = 442 # invalid
```

#### 2.3.1.5 Types

Python is dynamically typed.

```
height = 442           # An integer
height = 442.0         # Floating point
height = 'Really tall' # A string
```

#### 2.3.1.6 Case Sensitivity

Python is case sensitive

```
name = 'Jake'
Name = 'JAKE'

name == Name
```

```
False
```

Language statements are always lower-case

```
#while x < 0:   # OK
#WHILE x < 0:   # ERROR
```

#### 2.3.1.7 Conditionals

IF condition

```
if a > b:
    print('Computer says no')
else:
    print('Computer says yes')
```

```
Computer says yes
```

elif for multiple condition