Practical Session 4

Foundations Spatial Data Science

Today Goals & Aims

Practical goals

- Get familiar with the **logic of building functions** in Python.
- Build a function to process data.
- Combine Lists and Dictionaries for storing and interacting with data.
- Make your own package to access and use functions.

Why are we doing this

- Objects and Classes are the building blocks to perform data analysis in Python.
- Understand the logic of functions is essential to successfully use the data analysis libraries of Python.
- Functions are at the heart of every programming language. Critical to develop your thinking as a programmer.

Today Goals & Aims

Term Calendar

	Weekly Topic		Lead	WORKSHOP Date (Monday)	PRACTICAL Date Groups 1,2,3 (Tuesday)	Groups 4,5,6 (Wednesday)
1	Getting Oriented	initiate	David, Nicolas	4 Oct	4 Oct	5 Oct
2	Foundations (Part 1)	initiate	Nicolas	11 Oct	11 Oct	12 Oct
3	Foundations (Part 2)	initiate	Nicolas	18 Oct	18 Oct	19 Oct
4	Objects & Classes	initiate	David	25 Oct	25 Oct	26 Oct
5	Numeric Data	engage	David	1 Nov	1 Nov	2 Nov
	Reading Week					
6	Spatial Data	engage	Nicolas	15 Nov	15 Nov	16 Nov
7	Textual Data	engage	Nicolas	22 Nov	22 Nov	23 Nov
8	Visualising Data	solve	David	29 Nov	29 Nov	30 Nov
9	Classifying Data	solve	David	6 Dec	6 Dec	7 Dec
10	Clustering Data	solve	Nicolas	13 Dec	13 Dec	14 Dec

Python Lists vs. Dictionaries

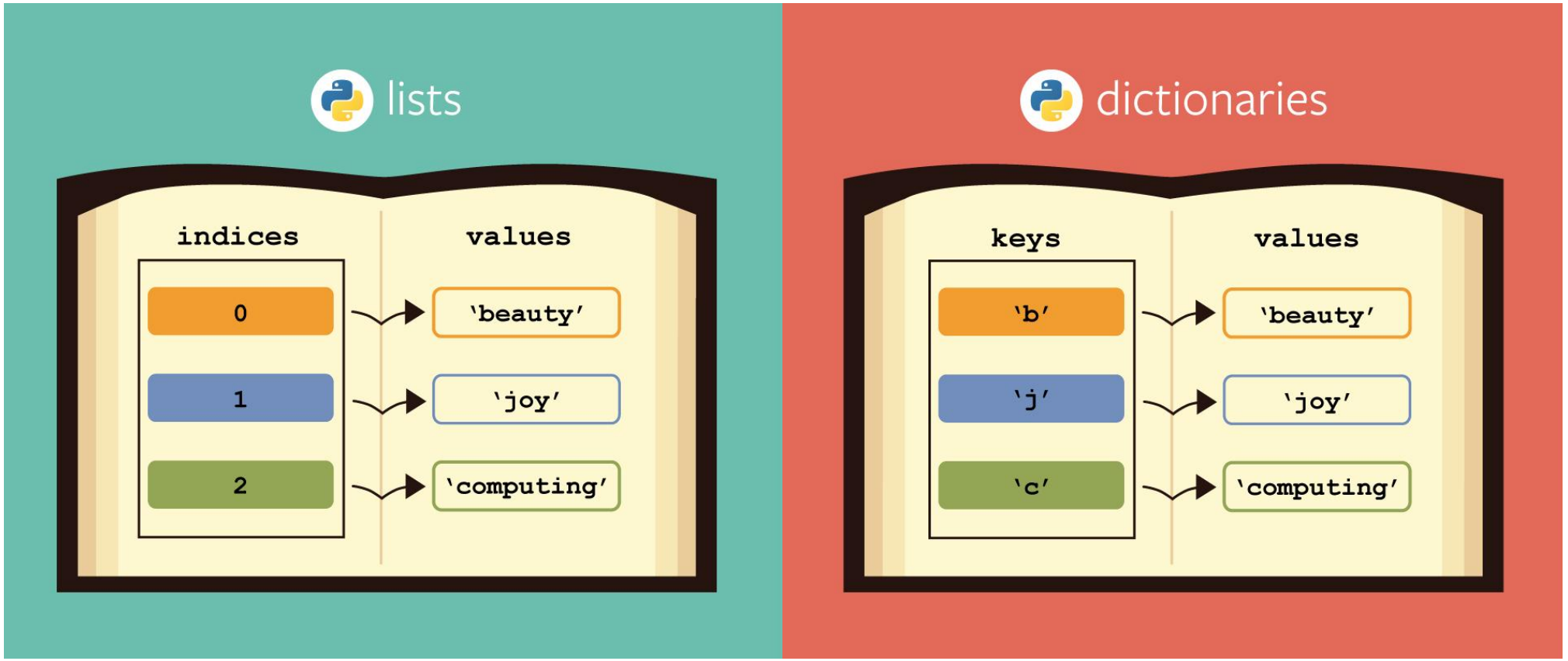


Image Source: https://www.analyticsvidhya.com/blog/2021/06/working-with-lists-dictionaries-in-python/

Python Lists vs. Dictionaries

Dictionaries:

1. No need to loop over the entire data set to find one data point!
    a. Use the index() function → `myData['Key'].index('Value')`

2. Every values associated with a key will be of the same type → `type(myData['Key'][0])`

Accessing Dictionary ?

How could we visualise a **dictionary**?

```
data = {
        'id'            :    [0,1,2]
        'Name'          :    ['Apple', 'Samsung', 'Xiaomi']
        'Product'       :    ['iPhone', 'Galaxy', 'Redmi']
        'Market_Share'  :    [14.2, 18.9, 16.9]
        'Country'       :    ['USA', 'South Korea', 'China']
}
```

id	Name	Product	Market_Share	Country
0	Apple	iPhone	14.2	USA
1	Samsung	Galaxy	18.9	South Korea
2	Xiaomi	Redmi	16.9	China

Questions:

1. data['Name'] = …
2. data['Name].index('Samsung') = …
3. data['Product'][1] = …

Accessing Dictionary ?

```
data = {
        'id'            :    [0,1,2]
        'Name'          :    ['Apple', 'Samsung', 'Xiaomi']
        'Product'       :    ['iPhone', 'Galaxy', 'Redmi']
        'Market_Share'  :    [14.2, 18.9, 16.9]
        'Country'       :    ['USA', 'South Korea', 'China']
}
```

Values for key: "Name"

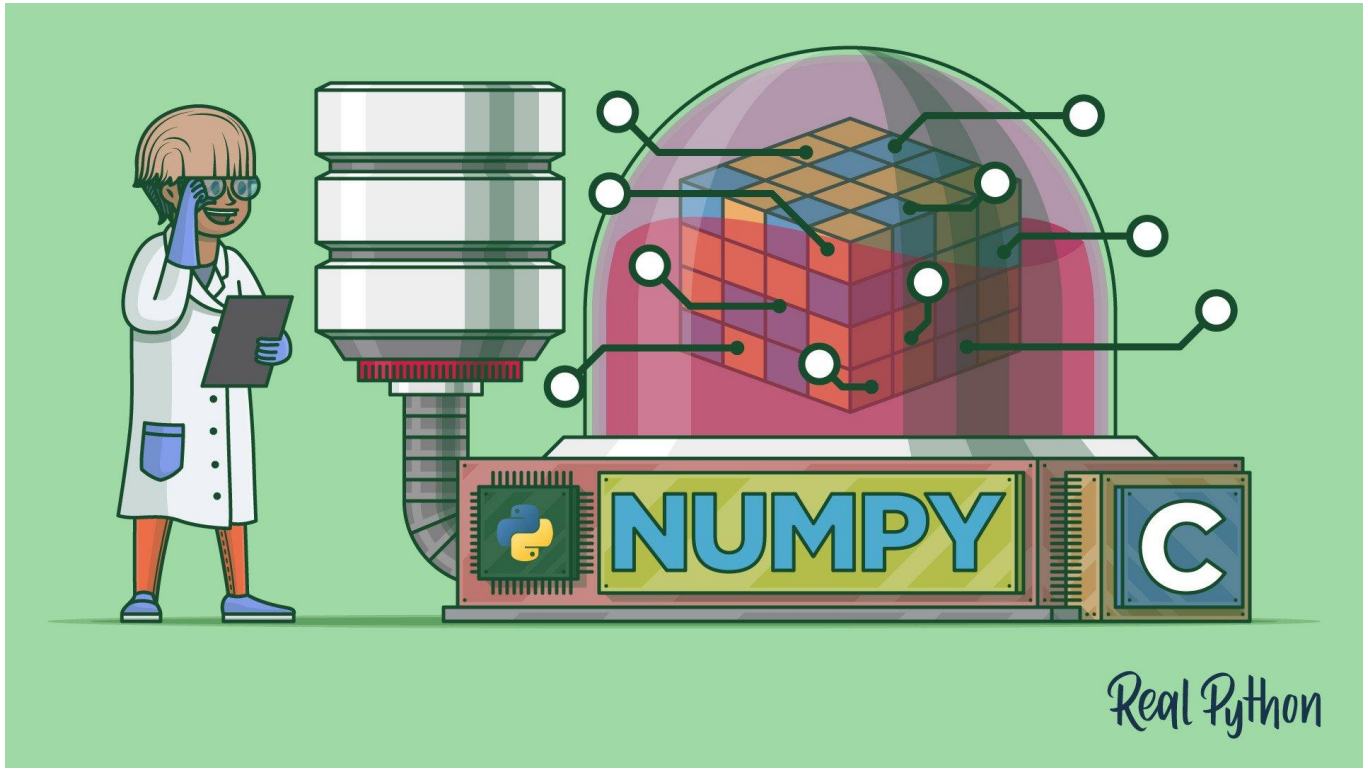
id	Name	Product	Market_Share	Country
0	Apple	iPhone	14.2	USA
1	Samsung	Galaxy	18.9	South Korea
2	Xiaomi	Redmi	16.9	China

Questions:

1. data['Name'] = ['Apple', 'Samsung', 'Xiaomi']
2. data['Name].index('Samsung') = 1
3. data['Name'][1] = 'Samsung'

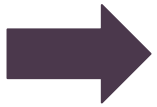
NumPy

Explore NumPy Python library using max(), min(), mean(), abs(), std()



F-Strings

```
print( str(x) + “some text”)
```



```
print( f “ {x} some text here>”)
```

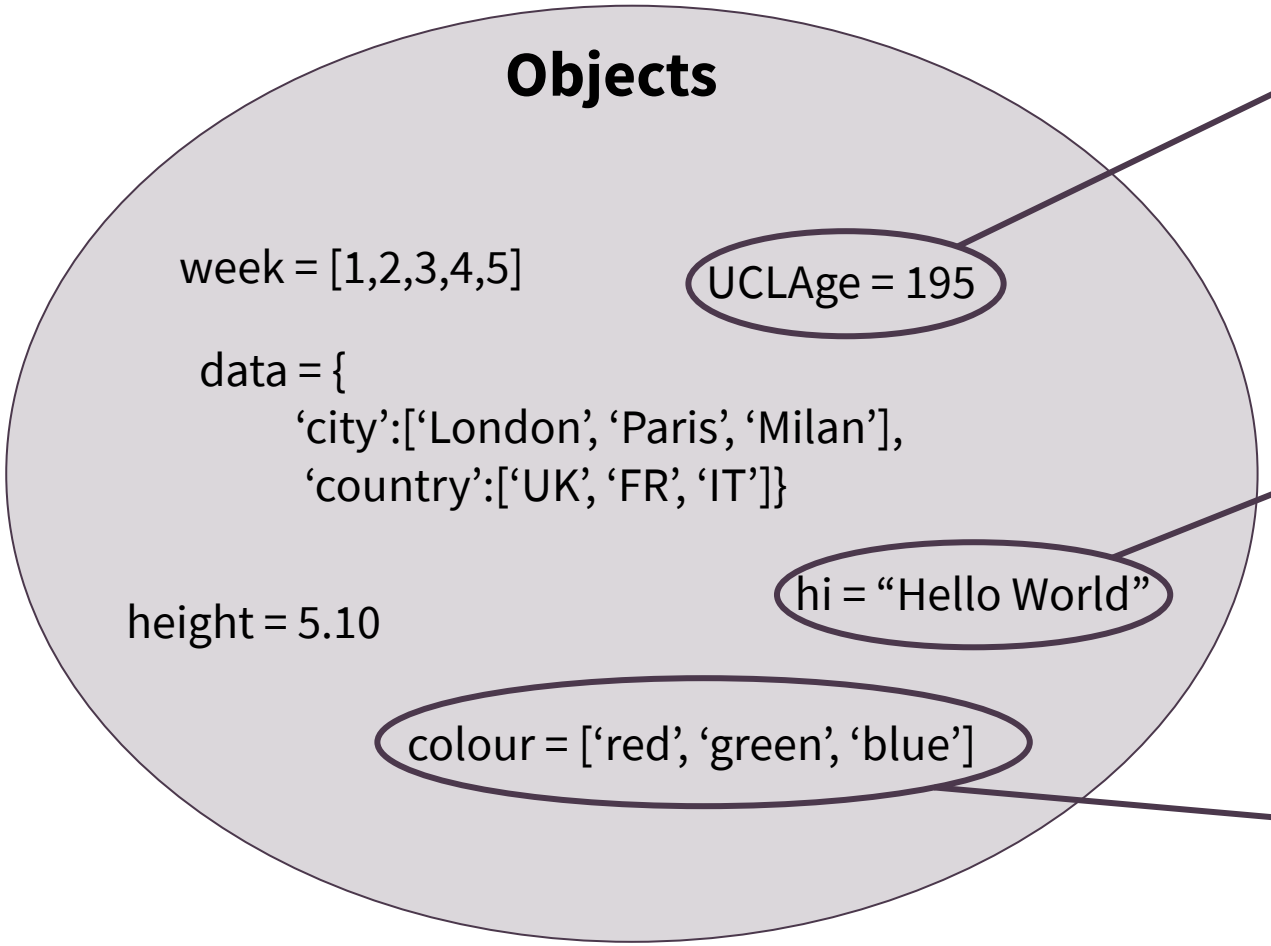
Python Objects

Objects: Python's abstraction for data. All data in python are represented by objects. Objects have **an identity, a type and a value.**

Object's Type: Determine the operation that the object supports. (len(),...)

Object's Value: Mutable vs. Immutable. Possibility of the value of some objects to change. **Numbers, strings = immutable** BUT **lists, dicts = mutable**

Objects, Types, Values



Object = UCLAge
Object's **Type** = integer
Object's **Value** = 195

Object = hi
Object's **Type** = string
Object's **Value** = "Hello World"

Object = colour
Object's **Type** = list
Object's **Value** = ['red', 'green', 'blue']

Python Functions

Function: Executable statement. **Self-contained block of code** that encapsulate a specific task or related group of tasks.

Example: mylist = [‘London’, ‘Manchester’, ‘Amsterdam’, ‘Cork’]

len(mylist) → 4

What do we do when we use len()? We **call** it and **pass** it arguments.

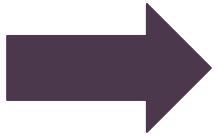
Arguments - give objects to the function len()

Values - function len() return a value.

Define Functions

Functions allow complex processes to be broken into smaller steps!

```
# Code to read file in
    url = "https://git.com/data.csv"
    # Empty list to store the data
    urlData = []
    response = urlopen(url)
    …..
# Code to process file
    ……
```



```
def read_file():
    # Code to read file in
    ….

def process_file():
    # Code to process file
    ….
```

Functions

Defining a Function:

```
def function_name(parameters):
    statement(s)
```

Calling a Function:

```
function_name(param_values)
```

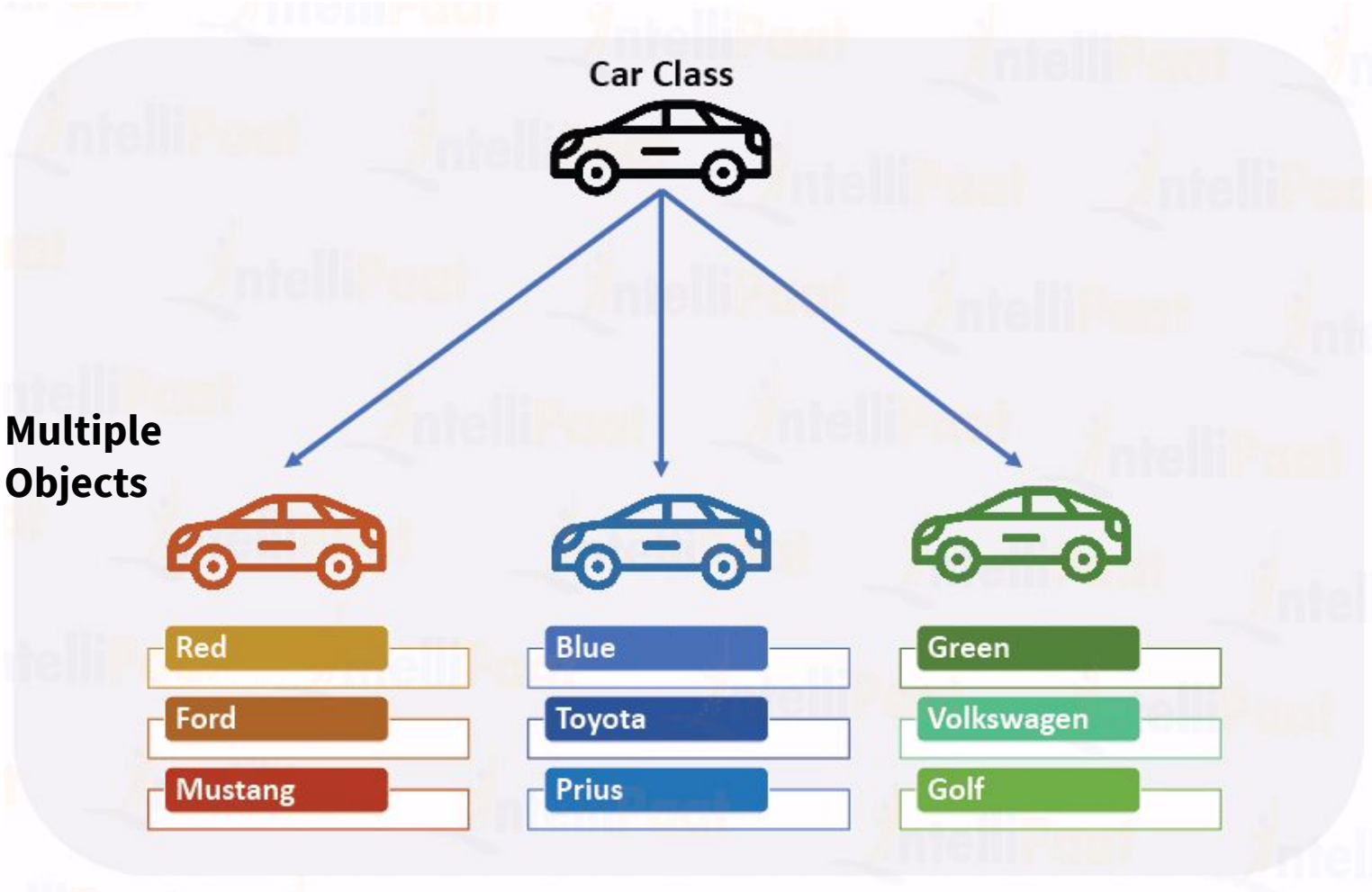
Functions - Examples

```
def TellMe_YourName(first_name, last_name):
    print(“Hello ” + first_name + “ ” + last_name + “!”)

def Food_Items(qty, item, price):
    print(f ‘The price of {qty} {item} cost £{price}.’)
```

TellMe_YourName(‘Alice’, ‘Bow’) → Hello Alice Bow !
Food_Items(4, ‘Apples’, 1.84) → The price of 4 Apples cost £1.84.

Python Classes



Multiple Objects

Attributes

Colour

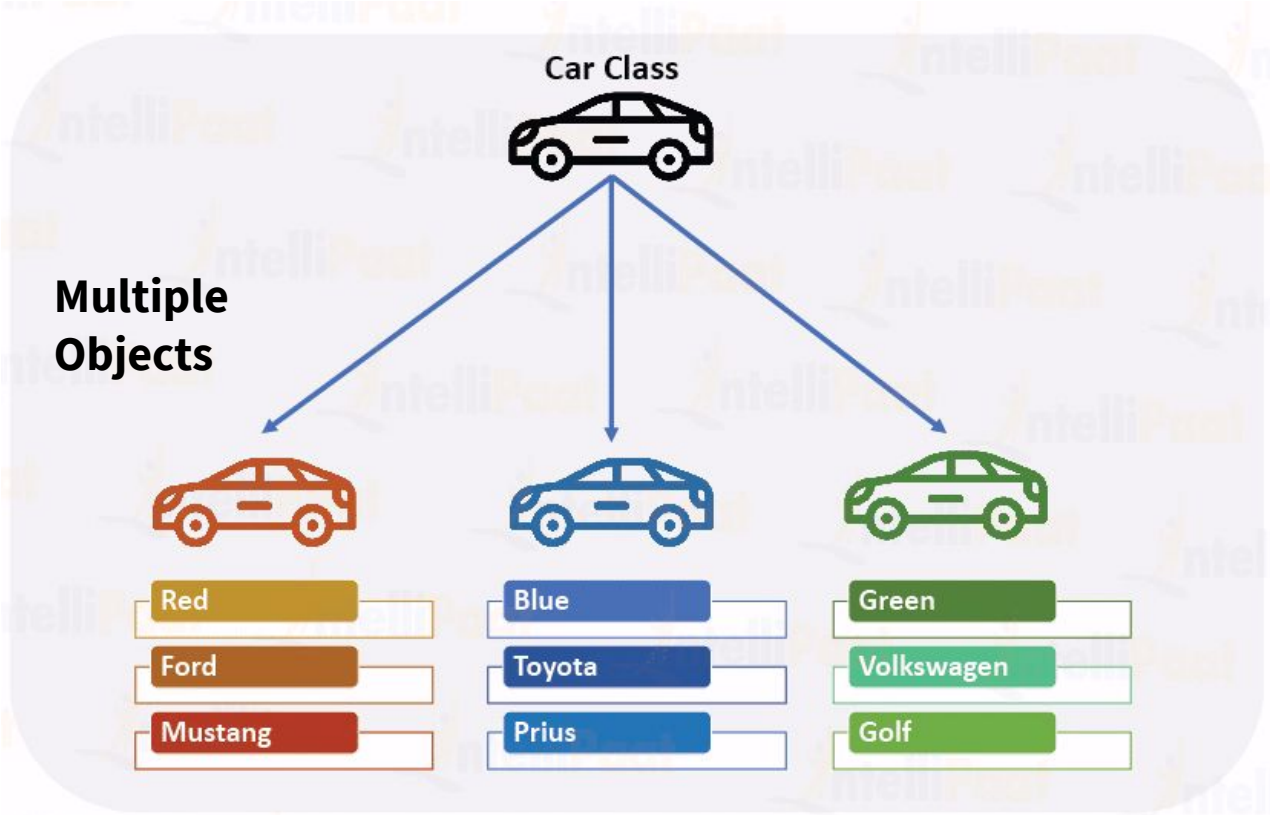
Company

Model

Python Classes

```
class Vehicle:
    vehicle_type  = ‘Cars’
    def __init__(self, name, colour):
        self.name = name
        self.colour = colour

# create 1st object
vehicle_1 = Vehicle(‘Ford’, ‘Red’)
vehicle_2 = Vehicle(‘Toyota’, ‘Blue’)
```



Create Python Packages

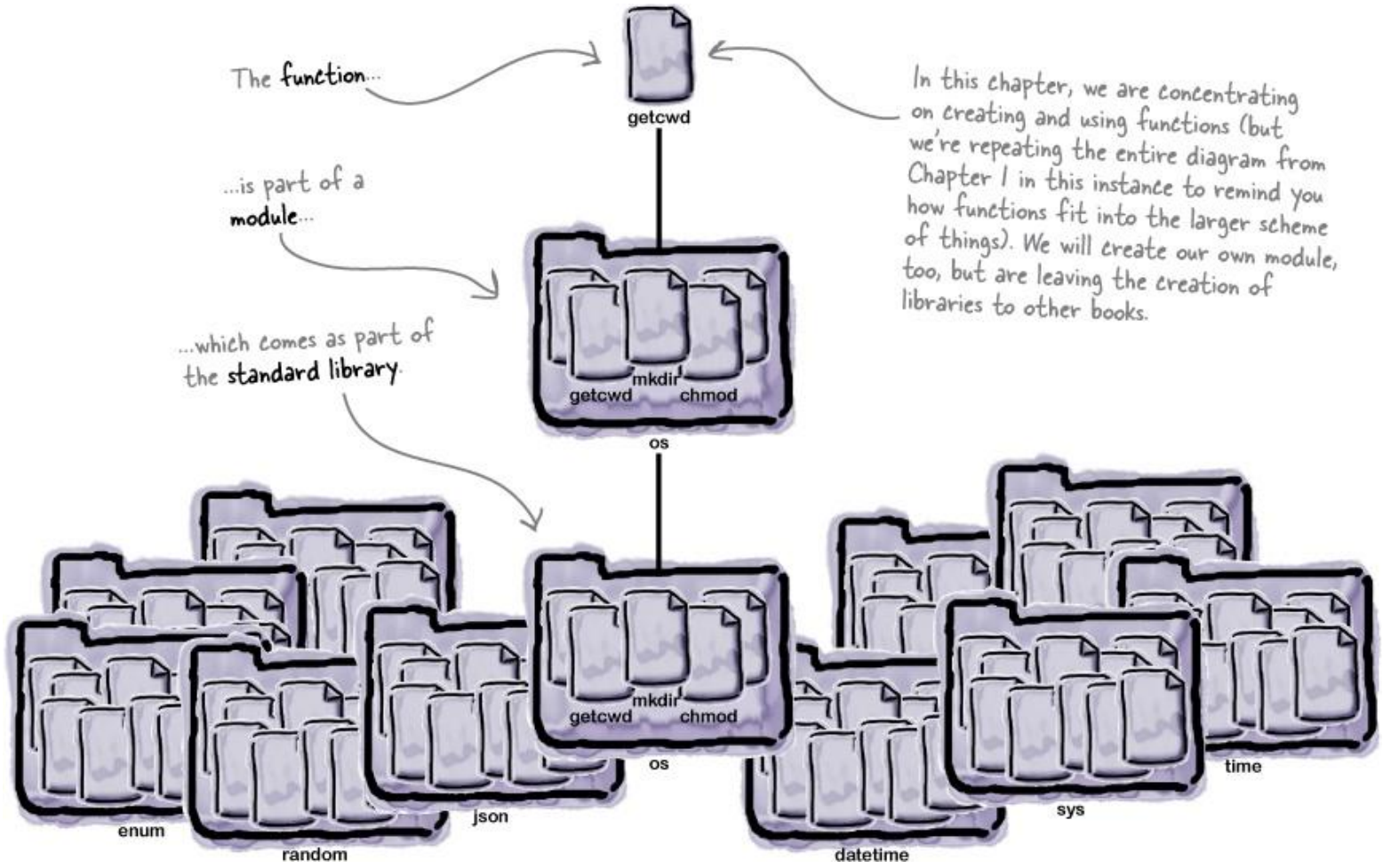


Image Source: https://learning.oreilly.com/library/view/head-first-python/9781491919521/ch04.html#introducing_functions

Packages Files - Essential

A few article to understand how to write python packages:

- “An Introduction to python Modules and Packages”: https://realpython.com/python-modules-packages/#python-packages
- “Packaging python projects”: https://packaging.python.org/tutorials/packaging-projects/
- “How to publish an open-source python package to PyPi ”https://realpython.com/pypi-publish-python-package/
- “How to package your python code”: https://towardsdatascience.com/how-to-package-your-python-code-df5a7739ab2e

Time to practice !

Notes

1. Taks 1.1 - URL may give you an error. Change url to “https://raw.githubusercontent.com/jreades/fsds/master/data/2020-08-24-sample-listings-simple.csv”

Next Week

Start to work with the Airbnb Data and pandas python library. You will discover how pandas open source library make data analysis flexible, powerful and easy!

Week 5 READINGS - Critically thinking about data

- D'Ignazio, Catherine, and Lauren F. Klein. 2020. *Data Feminism*. MIT Press. **Chapter 4** https://bookbook.pubpub.org/data-feminism.
- Airbnb and the rent gap: Gentrification through the sharing economy https://journals.sagepub.com/doi/10.1177/0308518X18778038
- Profiteers make a killing on Airbnb – and erode communities https://www.theguardian.com/commentisfree/2018/feb/12/profiteers-killing-airbnb-erode-communities