Python Course

Class 4 : Not enough types to solve your problems ? create your own ! (part 1)

Presented by : Sofiane MEDJKOUNE
Email : sofiane.medjkoune@protonmail.com

What means : “Your own type” ?

- Imagine yourself solving the previous exercise where a person has an id, a first_name, a weight, a height and then a BMI comment. We can imagine that as a set of multiple variables, and not a single type that contains all these informations in one place (not talking about lists here).
- Python does not provide (natively) a special type like this, why ? Infinite number of special types obviously.
- what we mean by your own type is an ***object***, that contains variables inside of it so we don’t need to keep dozens of variables in our code, for example we can have a type person that has as informations : id, first_name, height, weight, bmi_comment inside of the type, this is called ***attributes***.
- Our own types can have attributes but also functions (that we’ll be calling ***methods*** from now on). So now our Person can modify its own attributes by itself.
- This concepts are part of the ***Oriented Object Programming*** (OOP) logic.
- Written like this these notions may seem abstract(and it’s exactly the case), coding that and illustrating with example will help us understand it better.

Your type means class

- In oriented object programming a class is the structure that contains the attributes and methods of your object, creating it is easy :

```
class Class_name:
    #we created a class, see ? easy, now we need to put attributes and methods.
    def __init__(self):
        #We can define our attributes inside the __init__(self) method aka the constructor
        self.id = None
        self.first_name = None
        self.weight = None
        self.height = None
        self.bmi_comment = None
```

- We have created our class and some attributes. ***The constructor method is automatically called when you create an instance of your class*** (like when write string_variable = “hello world ! “, this instance is called an ***object***.
- The constructor method takes “***self***” as a parameter, and we can see that every attributes is written : self.attribute_name, why and what is self ?

Self

- Self is a naming convention used for OOP in python to show that this attribute and this function is part of our class.
- When you declare a variable (oops i meant attribute, remember variables are now attributes), this attribute is declared inside the __init__ method, and its syntax is : self.attribute_name = some value.
- When you create a function (that is now now called method, don't forget), ***the method takes self as a first parameter*** like this : ***def*** method_name(***self***, other)
- The self convention tells python : careful, this is an object attribute or object method, just like this :

```
class Person:
def __init__(self):
        self.id = None
        self.name = None
        self.weight = None
        self.height = None
        self.bmi_comment = None

def make_bmi_comment(self):
        self.bmi_comment = classify_bmi(body_mass_index(self.weight,self.height)
```

- why am i saying OBJECT attribute ? is there none object attributes ? we'll see that in the second part of this course.

How to create the object ?

- What we have done until now is create the class, not the object.
- There's a difference between a class and an object, a class is literally our new type, its definition while the object is the instantiation of the object, as we said before, it's just like creating a normal integer variable.
- to create that object all you do is :

```
my_first_object = class_name()
#So now we can have
lyes = Person()
#Hallelujah, let's add some informations to Lyes
lyes.id = 201700006426
lyes.first_name = "Lyes"
lyes.weight = 55
lyes.height = 1.76
```

- Something's missing right ? we didn't give a bmi_comment to lyes, here's the beauty, to do that you just need to call the method that assigns the bmi_comment.

```
lyes.make_bmi_comment
print(lyes.bmi_comment)
```

Practice !

For this practice we'll take the exact previous exercise but to make it better, easier to read, understand and modify, we'll use classes and each line in our file (person's information) will be a distinct object.

***Hint** : Remember that lists can take any type of object, yes even your own object ;)*