# ООП

Классы и объекты

# ООП на python

Хотя мы не говорили о классах и объектной ориентации в предыдущих главах, мы все время работали с классами. По сути, в Python все является классом. Гвидо ван Россум разработал язык по принципу «первоклассно все». Он писал: «Одной из моих целей в Python было сделать так, чтобы все объекты были «первоклассными». классы, модули, методы и т. д.), чтобы они имели одинаковый статус. То есть их можно присваивать переменным, помещать в списки, хранить в словарях, передавать в качестве аргументов и т. д.». (Блог, The History of Python, 27 февраля 2009 г.) Другими словами, «все» рассматривается одинаково, все является классом: функции и методы являются значениями, такими же, как списки, целые числа или числа с плавающей запятой. Каждый из них является экземпляром своего соответствующего класса.

# Ну а что мы

Мы поймем паттерны программирования вместе и каждый углубиться в чем хочет изучая нюансы. Долго - не значит хорошо, нужно начать создавать проекты, чтобы быстрее понять, что вы хотите в этой жизни. Сначала мы тестируем что-то, и если нам это понравится мы углубимся в детали.

Возможно вы захотите остаться в backend, возможно перейдете в full-stack, может станете мобильным разработчиком или вообще пойдете в разработки игр, etc. Чтобы узнать нравиться ли это вам backend нужно попробовать себя в нем, создать хотя бы один проект, поэтому мы освоим необходимый минимум и перейдем к созданию проектов.

После базовых знаний ООП мы перейдем к Object Relational Mapper, и улучшим понимание backend технологий реализуя проект - наш собственный интернет магазин.

# Атрибуты

```
class Human:
    'Человек разумный'
    counter = 0
    nucleus = 'eukaryotic'
    chromosomes = 46
    def __init__(self,name,age):
        self.name = name
        self.age = age
        self.counter = self.counter + 1
```

# Elon Musk

```
ElonMask = Human('Elon Musk', 51)

print(ElonMask)

print(ElonMask.chromosomes)
```

# Костя

```
KostynSRayona = Human('Костян с Района',19)

KostynSRayona.chromosomes = 46 + 1

print(KostynSRayona.chromosomes)

print(ElonMask.chromosomes)

print(ElonMask.__dict__)

print(KostynSRayona.__dict__)
```

# Вылечим Костью

```
del KostynSRayona.chromosomes

print(KostynSRayona.chromosomes)

print(KostynSRayona.__doc__)
```

# Начнем создать переменные объектов

```
class Dog():
    def set_parametrs(self, n,w):
        self.n = n
        self.w = w
    def get_parametrs(self):
        return (self.n , self.w)
bobik= Dog()
bobik.set_parametrs('Bobik', 50)
print(bobik.get_parametrs())
```

# __init__

```
class Planet:

    def __init__(self,name,population=None):

        self.name = name

        self.population = population or []
```

# __str__

```
class Planet:
    def __init__(self,name,population=None):
        self.name = name
        self.population = population or []
    def __str__(self):
        return self.name
```

# Private, protected, public

```
class A():
    def __init__(self):
        self.__priv = "I am private"
        self._prot = "I am protected"
        self.pub = "I am public"
    def get_paremets(self):
         return (self.pub, self._prot, self.__priv)
```