



Litchi Pi

Python

Programmation orientée objet

3 Novembre 2025

Classes

Pour créer un *type* qui soit adapté à notre utilisation, Python permet la création de *classes*

```
class Chien:  
    def __init__(self, name):  
        self.name = name  
        self.age = 0  
        self.nb_ouaf = 0  
  
    def ouaf(self):  
        print("OUAF")  
        self.nb_ouaf += 1
```

Une fonction associée à une classe est appelée *méthode*

Une variable associée à une classe est appelée *attribut*



Classes

```
fido = Chien()
ducon = Chien()

fido.ouaf()
ducon.ouaf()
ducon.ouaf()

print(fido.nb_ouaf, ducon.nb_ouaf) # Affiche "1 2"
```



Héritage

On peut définir une classe comme la “continuité” d'une autre

On dit qu'elle *hérite* d'une autre classe

Toutes les *méthodes* et les *attributs* de la classe parente seront transmises.

```
class Animal:  
    def __init__(self):  
        self.age = 0  
        self.en_vie = True  
  
    def vieillir(self):  
        self.age += 1  
        if self.age > 30:  
            print("Couic")  
            self.en_vie = False
```

```
class Chien(Animal):  
    def __init__(self, name):  
        Animal.__init__(self)  
        self.name = name  
  
    def ouaf(self):  
        print("Ouaf")  
  
fido = Chien()  
fido.ouaf()  
fido.vieillir()
```



TP

Formes géométriques

ou

Game of Life