Programmation orientée objet en python / classes

La programmation orientée objet (POO) permet de créer des entités (objets) que l'on peut manipuler . La programmation orientée objet impose des structures solides et claires. Les objets peuvent interagir entre eux, cela facilite grandement la compréhension du code et sa maintenance. On oppose souvent la programmation objet à la programmation procédurale , la première étant plus “professionnelle” que l'autre car plus fiable et plus propre. Les classes

Une classe regroupe des fonctions et des attributs qui définissent un objet. On appelle par ailleurs les fonctions d'une classe des “ méthodes ”.

Créons une classe Voiture :

class001.py

```
# coding: utf-8

class Voiture:

    def __init__(self):
        self.nom = "Ferrari"
```

Notre classe Voiture est une sorte d'usine à créer des voitures.

La méthode __init__() est appelée lors de la création d'un objet.

self.nom est une manière de stocker une information dans la classe. On parle d'attribut de classe. Dans notre cas, on stock le nom dans l'attribut nom .

Les objets

Un objet est une instance d'une classe . On peut créer autant d'objets que l'on désire avec une classe .

Créons maintenant notre voiture:

```
>>> ma_voiture = Voiture()
```

Les attributs de class

Les attributs de classe permettent de stocker des informations au niveau de la classe. Elle sont similaires aux variables.

Dans notre exemple:

```
>>> ma_voiture = Voiture()
>>> ma_voiture.nom
'Ferrari'
```

Vous pouvez à tout moment créer un attribut pour votre objet:

```
>>> ma_voiture.modele = "250"
```

Et le lire ainsi:

```
>>> ma_voiture.modele
'250'
```

Les méthodes

Les méthodes sont des fonctions définies dans une classe.

Créons une nouvelle méthode dans notre classe voiture:

class002.py

```
# coding: utf-8

class Voiture:

    def __init__(self):
        self.nom = "Ferrari"

    def donne_moi_le_modele(self):
        return "250"
```

Utilison cette méthode:

```
>>> ma_voiture=Voiture()
>>> ma_voiture.donne_moi_le_modele()
'250'
```