ISEN ALL IS DIGITAL! BREST | yncréa

TP N°2.0 - Python
Syntaxe orientée objet

Thibault Napoléon
thibault.napoleon@isen-ouest.yncrea.fr

Notions de syntaxe abordées :

- ✓ Classe
- ✓ Constructeur
- ✓ Attributs privé, protégés, publics
- ✓ Méthode
- ✓ Héritage
- ✓ Import de classe
- ✓ Tableau à deux dimensions

1 Syntaxe orientée objet

Classe

En *Python* les classes sont définit simplement avec le mot clé *class*. Exemple :

```
1 class Test:
2         """Class Test."""
3
4         ...
```

Le constructeur

En *Python* le constructeur est une méthode particulière nommée *__init__()* et définit de la manière suivante :

```
1 class Test:
2         """Class Test."""
3
4         def __init__(self, value):
5                 """Initialize the value."""
6                 self.__value = value
```

Ici *__value* est un attribut de la classe. Notez bien qu'il n'est pas nécessaire, ni possible, de le déclarer au préalable. La présence de la variable *self* est obligatoire et fait référence à l'instance de la classe.

Les attributs

En *Python*, outre le fait qu'on ne déclare pas les attributs, la notions de portée n'existe pas. On peut cependant utiliser le formalisme suivante :

```python
class Test:
        """Class Test."""

        def __init__(self, value):
                """Initialize the values."""
                self.__value = 1 # Privé
                self._value = 1  # Protégé
                self.value = 1   # Public
```

Méthode

En *Python* une méthode s'écrit comme suit :

```python
class Test:
        """Class Test."""

        def add(self, v1, v2): # Public
                """Addition."""
                return v1 + v2

        def __substract(self, v1, v2): # Privé
                """Substraction."""
                return v1 - v2
```

Notez encore la présence de la variable *self* ainsi que le formalisme de portée.

Héritage

La notion d'héritage s'obtient par la syntaxe :

```python
class Test2(Test): # Test2 hérite de Test
        """Class Test2."""

        def __init__(self):
                """Call the parent constructor."""
                Test.__init__(self, 5)
```

Import de classe

Pour importer une classe qui se trouve dans un autre fichier, il est nécessaire d'utiliser la syntaxe suivante :

```python
# Import de la classe Test à partir du fichier Test.py.
from Test import Test
```

Tableau à deux dimensions

En *Python* il n'existe pas la notion de tableau à deux dimensions. On la simuler grâce aux listes avec la syntaxe suivante :

```python
tab = [[0]*nbcol for i in range(0, nbligne)]
```