Les variables

À quoi sert une variable

Une variable permet d'associer un **objet** à un **nom**. On peut ensuite accéder à cet objet en mémoire grâce au nom de la variable.

Cela a plusieurs avantages :

- On peut ré-utiliser un objet à différents endroits de notre code plus rapidement.
- On peut modifier un objet à un seul endroit et répercuter ces changements partout dans notre script.

Des objets et des noms

Quand on crée une variable, on associe un **objet** à un **nom**.

Il est important de comprendre ce concept et de comprendre **qu'un même objet peut ainsi être associé à plusieurs noms différents**.

Cela peut parfois nous jouer des tours, notamment avec les objets **muables** (que l'on peut modifier) comme les listes :

```
>>> a = [1, 2, 3]
>>> b = a
>>> b += [4]
>>> print(a)
[1, 2, 3, 4]
```

Dans l'exemple ci-dessus, les noms `a` et `b` pointent vers la même liste en mémoire.

Ainsi, quand on modifie la liste, on modifie un seul et même objet. Les variables `a` et `b` pointant vers ce même objet en mémoire, on modifie donc `a` **et** `b`.

Affectations simples, parallèles et multiples

Il existe **trois façons principales** d'affecter une valeur à une variable :

1. L'affectation **simple**
2. L'affectation **parallèle**
3. L'affectation **multiple**

Avec l'affectation simple, on associe **une** valeur à **une** variable :

```
a = 5
```

Avec l'affectation parallèle, on associe **plusieurs** valeurs à **plusieurs** variables :

```
a, b = 5, 10
```

Avec l'affectation multiple, on associe **une** valeur à **plusieurs** variables :

```
a = b = 5
```

Singleton et 'small integer caching'

Python dispose de nombreux processus interne qui permettent d'optimiser vos scripts. Parmis ces processus, on retrouve le concept de *Singleton* et de *small integer caching*.

Le mot *Singleton* fait référence au fait qu'un objet est unique.

C'est le cas de plusieurs objets de Python comme les booléens `True` et `False` ou l'objet `None`.

Quand vous créez un booléen `True`, vous référez ainsi toujours au même objet en mémoire.

Ce même concept est valable pour certains objets comme les nombres compris entre `5` et `256` et les chaînes de caractères courtes.

Règles et conventions de nommage

Il existe plusieurs conventions très suivies dans la communauté Python que vous retrouverez sous le nom de **PEP8**.

Parmi ces conventions, on retrouve de nombreuses conventions de nommage qui indique les règles à suivre lorsque l'on crée des noms de variable.

Ainsi, il est conseillé d'utiliser uniquement des lettres en minuscule et des tirets du bas pour séparer les mots.

Voici quelques exemples de noms de variables qui respectent ces règles :

- website_url
- number_of_students
- bank_account_id
- name

Et quelques contre-exemples :

- StudentID
- Lastname
- firstName
- PhoneNUM

Votre script fonctionnera si vous ne suivez pas ces conventions. Mais elles sont très respectées dans la communauté Python et permettent d'avoir une uniformité très agréable parmi les scripts sur lesquels vous allez travailler.

Si vous respectez toutes ces conventions et que vous vous habituez à les utiliser, vous verrez qu'avec en plus la syntaxe épurée de Python, il vous sera très facile de lire des scripts d'autres développeurs.