

W2-S1-C3-introduction-types

December 15, 2014

1 Un peu de calcul sur les types

1.1 Complément - niveau basique

1.1.1 La fonction type

Nous avons vu dans la vidéo que chaque objet possède un type. On peut très simplement accéder au type d'un objet en appelant une fonction “built-in”, c'est-à-dire prédéfinie dans python, qui s'appelle, eh bien oui, `type`.

On l'utilise tout simplement comme ceci

```
In []: type(1)
```

```
In []: type('spam')
```

Cette fonction est assez peu utilisée par les programmeurs expérimentés, mais va nous être utile à bien comprendre le langage, notamment pour manipuler les valeurs numériques.

1.2 Complément - niveau avancé

1.2.1 La fonction isinstance

Une autre fonction prédéfinie, voisine de `type` mais plus utile dans la pratique, est la fonction `isinstance` qui permet de savoir si un objet est d'un type donné; par exemple

```
In []: isinstance(23, int)
```

À la vue de ce seul exemple on pourrait penser que `isinstance` est presque identique à `type`; en réalité elle est un peu plus élaborée, notamment pour la programmation objet et l'héritage, nous aurons l'occasion d'y revenir.

On remarque ici en passant que la variable `int` est connue de python alors que nous ne l'avons pas définie. Il s'agit d'une variable prédéfinie, qui désigne le type des entiers, que nous étudierons très bientôt.

Pour conclure sur `isinstance`: cette fonction est utile en pratique précisément parce que python est à typage dynamique. Aussi il est souvent utile de s'assurer qu'une variable passée à une fonction est du ou des types attendus, puisque contrairement à un langage typé statiquement comme C++, on n'a aucune garantie de ce genre à l'exécution. À nouveau nous aurons l'occasion de revenir sur ce point.