Dictionnaires : manipulation et méthodes

Algorithmique et structuration de données

Léna Gaubert
lena.gaubert@proton.me

L3 TIM/TAL @ INALCO

Dictionnaires

Définition et manipulation

Définition

Un **dictionnaire** (`dict`) est une structure de données qui contient une collection d'objets Python auxquels on accède à l'aide d'une **clé** (*key*) dite de *correspondance* (*mapping*). Ces objets ne sont *pas* séquentiels, il s'agit d'objets dits de correspondance (*mapping*) ou tableaux associatifs.

Définition

Un **dictionnaire** (`dict`) est une structure de données qui contient une collection d'objets Python auxquels on accède à l'aide d'une **clé** (*key*) dite de *correspondance* (*mapping*). Ces objets ne sont *pas* séquentiels, il s'agit d'objets dits de correspondance (*mapping*) ou tableaux associatifs.

Plus simplement : un dictionnaire stocke des paires `clé:valeur` (ou `key:value`). Un dictionnaire est *ordonné* (depuis Python 3.7), *modifiable*, et n'autorise pas les *doublons*.

Définition

Un dictionnaire est défini avec des *accolades* ({}). Les paires `key:value` sont séparées par une *virgule*.

> *La clé d'un dictionnaire peut être un nombre (entier ou flottant) ou une chaîne de caractère.*

Créer un dictionnaire

Initialiser et stocker des éléments dans un dictionnaire :

```
lang = {}  # init
lang["en"] = "english"
lang["it"] = "italian"
print(lang)
```

Ligne 2 : on associe la valeur `"english"` à la clé `"en"`.

Accéder à une valeur

Accéder à une valeur :

On accède à une **valeur** du dictionnaire à l'aide de sa **clé** correspondante.
La syntaxe est la suivante : `dict[key]`

```
course = {
    "class": "Python",
    "level": 1,
    "semester": "Fall",
    "teacher": "Gaubert"
}
```

Accéder à une valeur

Accéder à une valeur :

On accède à une **valeur** du dictionnaire à l'aide de sa **clé** correspondante.
La syntaxe est la suivante : `dict[key]`

```
course["level"]  # 1
cours["teacher"]  # Gaubert
```

Accéder à une valeur

On peut aussi utiliser la méthode `.get(key)`.

```
course.get("level")  # 1
```

> Pour vérifier la présence d'une clé dans un dictionnaire, utiliser la méthode `.has_key(key)`.

Supprimer un élément

Pour supprimer un élément d'un dictionnaire (paire clé:valeur), utiliser la syntaxe suivante :

```
del course["level"]
print(course)
```

Parcourir un dictionnaire

Parcourir les *clés* du dictionnaire

On utilise la méthode `.keys()`. Appliquée à un dictionnaire, elle renvoie les clés du dictionnaire. (type : `dict_keys`)

```
date = {
    "month": "december",
    "day": 1
    }

for key in date.keys():
    print(key)
    print(date[key])
```

Parcourir un dictionnaire

Parcourir les *valeurs* du dictionnaire

Si on veut uniquement parcourir les *valeurs* du dictionnaire, on peut utiliser la méthode `.values()`. Renvoie les *valeurs* stockées dans le dictionnaire. (type : `dict_values`)

```
for value in date.values():
    print(value)
```

Parcourir un dictionnaire

Parcourir les *clés* et les *valeurs* du dictionnaire

Utiliser la méthode `.items()`. Renvoie les paires de clés, valeurs stockées dans le dictionnaire (type : `dict_items`, les éléments sont des *tuples*).

```
for key, value in date.items():
    print("Key:", key)
    print("Value:", value)
```

Fusionner des dictionnaires

Pour fusionner deux dictionnaires, on utilise la méthode `.update()`.

```
a = {"surname": "Gaubert"}
b = {"name": "Léna"}
a.update(b)
print(a)
```

Time to practice! (´｡• ᵕ •｡`) ♡