Débrief séance 4

UE12 P24 - Python

Indentation et espaces

Le principal objectif de votre code n'est pas d'être lu par Python, mais par vous. Utilisez des espaces pour le rendre lisible :

```
1 def channel_detail_screen(self, channel_id: int):
2     messages = self.server.get_channel_messages(channel_id)
3 
4     for message in messages:
5         print(f'{message.reception_date} | {message.sender_id}: {message.co
6 
7     print()
8     message = input('Write a message or press <Enter> to go back: ')
```

Indentation et espaces

```
class User:
    def __init__(self, id: int, name: str):
        self.id = id
        self.name = name

    def to_dict(self) -> dict:
        return {
            'id': self.id,
            'name': self.name
        }
```

Commentaires

Ajoutez des commentaires pour expliquer votre code :

```
def save_server(server_with_classes):
    '''Sauvegarde le serveur dans un fichier json.'''

    # `server_with_classes` est un dictionnaire qui contient des listes d'o
    # Il faut commencer par convertir chacun de ces objets en dictionnaire,
    # pour ensuite convertir le tout en json.
    server_with_dicts = {}
```

Commentaires

Vous pouvez même mettre des exemples d'utilisation de votre code :

```python
def to_dict(self) -> dict:
    '''Convertit un objet `user` de la classe `User` en `dict`.

    Exemple :
    >>> user_dict = user.to_dict()
    '''
    return {
        'id': self.id,
        'name': self.name
    }
```

Listes

Vous avez plusieurs façons de créer une liste à partir d'une autre.

Avec une boucle `for` classique :

```
1 self._users = []
2 for user_as_dict in data['users']:
3     user_as_object = User(user_as_dict['id'], user_as_dict['name'])
4     self._users.append(user_as_object)
```

Par "compréhension" (ressemble aux définitions d'ensembles en math) :

```
1 self._users = [User(user['id'], user['name']) for user in data['users']]
```

Choisissez la méthode qui vous convient le mieux.

Classes

Création d'une classe :

```python
class User:
    def __init__(self, id: int, name: str):
        self.id = id
        self.name = name

    def say_hello(self):
        print(f"Hello, I'm {self.name}!")
```

Dictionnaires et objets

Création et utilisation d'un dictionnaire :

```
1 alice_as_dict = {
2     'id': 1,
3     'name': 'alice'
4 }
5 alice_id = alice_as_dict['id']
```

Création et utilisation d'un objet :

```
1 alice_as_obj = User(1, 'alice')
2 alice_id = alice_as_obj.id
```

Appels de fonctions

Une fonction s'appelle toujours de la même façon. Si elle a été définie dans un module ou une classe, il faut préciser le nom du module, le nom de la classe, ou l'objet, pour que Python la trouve :

```
1  empy_server = {}
2  save_server(empy_server)
3
4  alice_as_obj.say_hello()
```

```
Hello, I'm alice!
```

```
1  import pandas as pd
2  pd.DataFrame.from_dict({'col_1': [True, False], 'col_2': [1, 2]})
```

	col_1	col_2
0	True	1
1	False	2