Занятие №5

I T E A

IT Education Academy

WWW.ITEA.UA

I T E A

www.itea.ua

Поляков Антон

Фото инструктора

- Python backend developer

КОНТАКТНЫЕ ДАННЫЕ

Telegram: @polyakov1

Потоки

Классически — подпрограмма, которая берёт на себя некую функциональность, которая требует достаточно времени на выполнение. Потоки следует использовать для операций ввода/вывода (работа с сетью, файлами, ОС). Потоки не следует использовать для операций, которые используют большую вычислительную мощность (разного рода математические операции, потребляющие ресурсы CPU).

Создание потоков

```
from threading import Thread


Def file_w(file_name, number):
        with open(filename, «w») as f:
                for i in range(number):
                        f.write(random.randint(0, 1000))


T = Thread(target=file_w, args=(«file.txt», 10), kwargs=())
T.start()
```

Потоки на классах

```
import threading

Class OwnThread(threading.Thread):
    def __init__(self, num):
        super().__init__(self, num=num)
        self.num = num

    def run(self):
        print(f«Tread number {self.num} started»

    thread1 = OwnThread(1)
    thread2 = OwnThread(2)
    thread1.start()
    thread2.start()
```

Методы потоков

t.join() - Данный метод блокирует, поток из которого был создан текущий поток, до окончания выполнения.

is_alive() - Жив ли поток? :)

GetName() - получаем имя потока

SetName() - устанавливаем имя потока

Контекстные менеджеры

```
f_obj = open(path, 'w')
f_obj.write(some_data)
f_obj.close()

==

with open(path, 'w') as f_obj:
    f_obj.write(some_data)
```

Контекстные менеджеры

```
class File():

    def __init__(self, a, b):
        self.a = a
        self.b = b
    def __enter__(self):
        print(f"Entered args are {self.a, self.b}")
        return "Hello"

    def __exit__(self, *args):

        print("The end of context manager")
        print(args)



with File(1, 2) as f:
    print(f)
```

Модуль shelve

Он сохраняет объекты в файл с определенным ключом. Затем по этому ключу может извлечь ранее сохраненный объект из файла. Процесс работы с данными через модуль shelve напоминает работу со словарями, которые также используют ключи для сохранения и извлечения объектов.

```
import shelve

FILENAME = «file»
with shelve.open(FILENAME) as states:
    states[«name»] = «New Name»
```

Модуль shelve

Также мы можем использовать метод get(). Первый параметр метода - ключ, по которому следует получить значение, а второй - значение по умолчанию, которое возвращается, если ключ не найден.

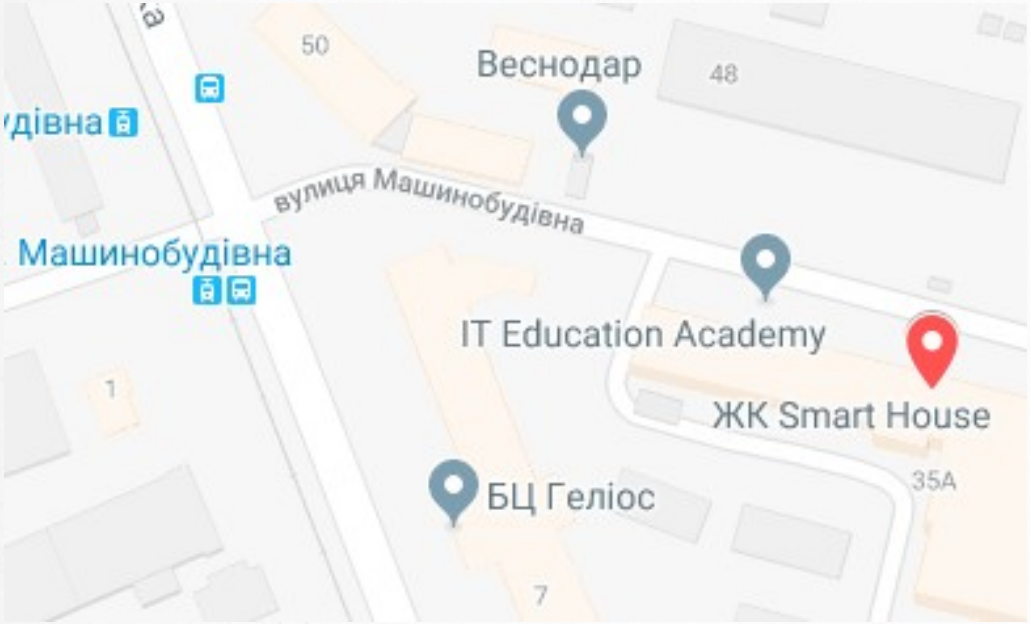
```
with shelve.open(FILENAME) as states:
    state = states.get("Brussels", "Undefined")
    print(state)
```

Используя цикл for, можно перебрать все значения из файла.
Метод keys() возвращает все ключи из файла, а метод values() - все значения.

Еще один метод items() возвращает набор кортежей. Каждый кортеж содержит ключ и значение.

Задачи

1) Создать декоратор, который будет запускать функцию в отдельном потоке. Декоратор должен принимать следующие аргументы: название потока, является ли поток демоном.
2) Создать функцию, которая будет скачивать файл из интернета по ссылке, повесить на неё созданный декоратор. Создать список из 10 ссылок, по которым будет происходить скачивание. Создать список потоков, отдельный поток, на каждую из ссылок. Каждый поток должен сигнализировать, о том, что, он начал работу и по какой ссылке он работает, так же должен сообщать когда скачивание закончится.
3) Написать свой контекстный менеджер для работы с файлами.
4) Дополнение к предыдущей работе с соц. Сетью. Все хранение данных пользователей реализовать на основе модуля shelve.



КОНТАКТНЫЕ ДАННЫЕ

ITEA

ул. Машиностроительная, 41, ЖК «Smart House», Киев

ул. Срибнокильская, 1, офис 269, Киев

пр. Академика Глушкова, 1, корп.17, Киев

+38 (044) 599-01-79
facebook.com/Itea
info@itea.ua
itea.ua