**REQUESTS**

my\_req = requests.get("https://python-scripts.com/requests?ysclid=ltyqz6xk13252124311")

print(my\_req)

это простейший запрос к сайту, функцией get мы получаем информацию о сайте, ссылку можно как думаю вставить любую, на выходе получаем Response[200] – это говорит о том что мы успешно обратились к сайту, если Response[404] - то что-то не так.

print(my\_req.status\_code)

это вывод какого-то кода, тут то же самое что с response 200 ок 404 плохо

Ответ на **get** запрос содержит весьма ценную информацию, она находиться в теле запроса и называется **payload.**

Эту инфу можно получить методом **content**, но контент выводит инфу в битах, иногда надо в кодировке utf-8.

print(my\_req.content)

в тексте метод **.text**

print(my\_req.headers)

так – же метод headers выводит како-то http запрос который тоже может помочь.

Можем ввести так:

print(my\_req.headers['Content-Type'])

вот че на выходе: text/html; charset=utf-8

Для просто паркинга всего, нужно докучать две библиотеки:

from bs4 import BeautifulSoup

lxml

после чего создаем объект BS

soup = BeautifulSoup(responce.content, 'lxml')

первый параметр переменная get/post запроса, второй lxml, это парсер который мы будем использовать.

Чтобы спарсить данные мы должны, обратиться у методу find или find\_all, первый параметр наш тэг на странице, второй обращение к информации и в конце заполняем текст.

date = soup.find("span", class\_ = "hdr\_\_inner").text

Схватить user-agent для его подмены чтобы потом сайт не допедрил что мы парсимся.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Значок на компьютере

Автоматически созданное описание

Чтобы не париться с user-agent и каждый раз делать подмену можно использовать модуль fakeuseragent

**pip install fake-useragent.**

Теперь код выглядит так:

import fake\_useragent

user = fake\_useragent.UserAgent().random

header = {'user-agent': user}

responce = requests.get(f"https://pogoda.mail.ru/prognoz/moskva/extended/", headers = header)

иногда он может выкидывать ошибку, поставим обработчик.

В headers вводиться не только user-agent

header = {'user-agent' : "asbnggwe23werevrv432", "Accept" : "application/json"}

бывают и другие заголовки которые можно тянуть.

Разберемся с регистрацией на каком-то форуме.

Заходим на сайт и вводим не существующие данные в лог и пароль далее в важные данные запроса, цепляем оттуда ссылку post запроса.

После чего создаем словарь, куда будем кидать параметры входа, их копируем с полезных данных post запроса.

data = {

    "username": "123",

    "password": "123"

}

Вот код для авторизации на сайте, но есть прекл, каждый раз при запуске этого запроса авторизация будет проходить заново, нужно юзать сессию щя добавим.

import requests

from bs4 import BeautifulSoup

import fake\_useragent

header = {

    'user-agent' : "asbnggwe23werevrv432",

    "Accept" : "application/json"

    }

data = {

    "username": "nike",

    "password": "Atgiryta1321"

}

link = r"https://com-forum.ru/index.php?option=com\_content&view=featured&Itemid=348"

responce = requests.post(link, data=data, headers = header).text

print(responce)

Вот че мы поменяли, мы создали сессию и запрос отправили в нее это позволило сохранить в нее куки файлы, для повторных запросов без регистрации.

session = requests.Session()

responce = session.post(link, data=data, headers = header).text

Далее мы указываем еще одну ссылку уже для get запроса, передаем его через session, указываем только заголовки, и потом стандартный парсинг, который я уже описывал выше.

link\_my\_profil = r"https://com-forum.ru/index.php?option=com\_users&view=profile&Itemid=348"

profil\_info\_req = session.get(link\_my\_profil, headers = header)

soup = BeautifulSoup(profil\_info\_req.content, "lxml")

info\_data = soup.find('h2', class\_ = "module\_title").text

print(info\_data)

Так-же чтобы по 100 раз не заходить на наш сайт, куки можно записать в словарь и подгружать их при входе