Лабораторная работа № 10

Тема: Работа с библиотекой requests, http-запросы

Цель работы: научиться использовать библиотеку requests, API.

Библиотека **requests** в Python предоставляет простой и удобный интерфейс для выполнения HTTP-запросов. Она является стандартной библиотекой для работы с сетевыми запросами и облегчает взаимодействие с веб-ресурсами.

Для установки библиотеки **requests**, можно воспользоваться **pip**:

> pip install requests

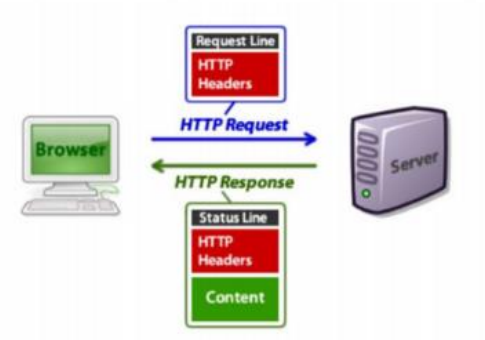
requests предоставляет простой и интуитивно понятный API для отправки HTTP-запросов и обработки ответов.

Что такое HTTP

- HTTP (Hypertext Transfer Protocol) – это протокол для передачи данных (текстов, изображений, видео, аудио и т.д.) по интернету.
- HTTP не хранит состояние между запросами. Каждый запрос интерпретируется независимо от других запросов. Если нужно сохранять и передавать состояние – это должны делать реализации клиента и сервера.

Client <-> Server

Как выглядит клиент-серверное взаимодействие:

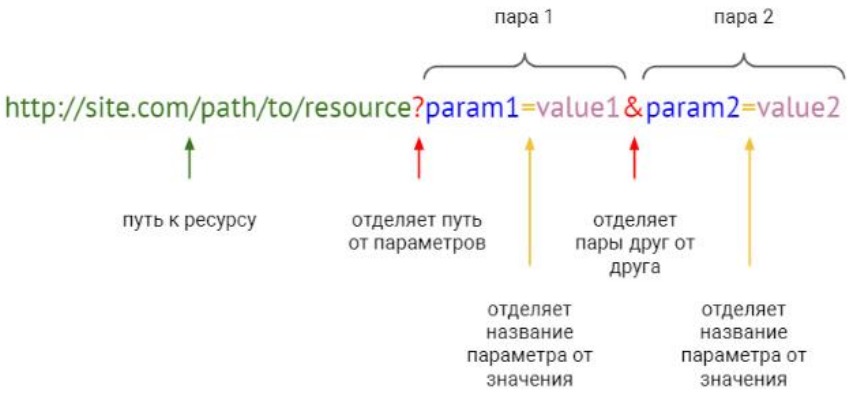


В роли клиента может выступать не только браузер, но и любое устройство и даже другая программа. Когда говорят «клиент-сервер», подразумевают, что есть сторона, которая запрашивает данные, а другая сторона ей отвечает.

HTTP Request (запрос)

HTTP методы

Структура URL



Для чего нужны заголовки, а для чего параметры?

- Как правило, в заголовках передают **информацию** о запросе. Например, авторизационные данные пользователя, версию браузера, выставленные cookie, поддерживаемые форматы сжатия данных и т.д.
- Параметры запроса содержат информацию о том, что именно запрашивает пользователь.
- Также в параметрах запроса можно передавать произвольную информацию при размещении ссылки. Например, источник, где ссылка размещена (это используется в интернет-рекламе).

Пример HTTP ответа

```
HTTP/1.x 200 OK
Transfer-Encoding: chunked
Date: Sat, 28 Nov 2009 04:36:25 GMT
Server: LiteSpeed
Connection: close
X-Powered-By: W3 Total Cache/0.8
Pragma: public
Expires: Sat, 28 Nov 2009 05:36:25 GMT
Etag: "pub1259380237.gz"
Cache-Control: max-age=3600, public
Content-Type: text/html; charset=UTF-8
Last-Modified: Sat, 28 Nov 2009 03:50:37 GMT
X-Pingback: http://net.tutsplus.com/xmlrpc.php
Content-Encoding: gzip

Vary: Accept-Encoding, Cookie, User-Agent


<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">

<head>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />

<title>Top 20+ MySQL Best Practices - Nettuts+</title>
```

HTTP коды

Первая цифра обозначает семантику кода.

- 1XX — Информационные
- 2XX — Успешный вызов (пример: 200 – ок, 201 – создано)
- 3XX — Перенаправление
- 4XX — Ошибка на стороне клиента (пример: 404 – не найдено, 403 – недостаточно прав)
- 5XX — Ошибка на стороне сервера

Выполнение простого GET-запроса:

```python
# Выполнение простого GET-запроса:
import requests
url = "https://httpbin.org/get"
resp = requests.get(url)
print(resp)
print(resp.status_code)

```

Оболочка ×

```
>>> %Run test_.py
  <Response [200]>
  200

>>>
```

Атрибуты и методы объекта Response

- `status_code` – HTTP-статус ответа
- `headers` – заголовки ответа
- `content` – содержимое ответа в байтах
- `text` – содержимое ответа в текстовом представлении (utf8, так как все строки в Python – юникодные)
- `json()` – представление ответа в виде словаря. Работает только в том случае, если сервер вернул валидный JSON. Используется при работе с API.

Параметры запроса и заголовки

Чтобы передать параметры и заголовки в запросе, используйте именованные аргументы:

```python
import requests

url = "https://httpbin.org/get
params = {"foo": "bar", "message": "hello"}
headers = {"Authorization": "secret-token-123"}

resp = requests.get(url, params=params, headers=headers)
```

Важно: конкретные значения параметров и заголовков зависят от сервера, к которому происходит обращение. Для того чтобы узнать точные значения параметров, нужно читать документацию или проверять опытным путем.

Загрузка файлов

Чтобы передавать файл, его нужно открыть в байтовом режиме и передать объект файла в параметр files:

```
import requests

with open('gifs/your_file.mp4', 'rb') as f:
    resp = requests.post('http://httpbin.org/post', files={"file": f})
```

Задания на лабораторную работу

Задача №1

Кто самый умный супергерой?
Есть [API по информации о супергероях](#) с информацией по всем супергероям. Нужно определить:

a. Кто самый умный (intelligence) из трех супергероев- Hulk, Captain America, Thanos.
b. Кто самый сильный ('power) из трех героев – Yoda, Venom, T-800.
c. Топ 10 самых быстрых ('speed') героев.

Примеры использования API:

```
import requests
from pprint import pprint

# Получить всех супергероев
url = 'https://akabab.github.io/superhero-api/api/all.json'
response = requests.get(url)
# вывод в консоль 1-го в списке:
pprint(response.json()[0])
```

Оболочка ×

```
>>> %Run superheroapi_test.py
 {'appearance': {'eyeColor': 'Yellow',
                 'gender': 'Male',
                 'hairColor': 'No Hair',
                 'height': ["6'8", '203 cm'],
                 'race': 'Human',
                 'weight': ['980 lb', '441 kg']},
  'biography': {'aliases': ['Rick Jones'],
                'alignment': 'good',
                'alterEgos': 'No alter egos found.',
                'firstAppearance': 'Hulk Vol 2 #2 (April, 2008) (as A-Bomb)',
                'fullName': 'Richard Milhouse Jones',
                'placeOfBirth': 'Scarsdale, Arizona',
                'publisher': 'Marvel Comics'},
  'connections': {'groupAffiliation': 'Hulk Family; Excelsior (sponsor), '
```

```python
import requests
from pprint import pprint

# Получить всех супергероев
url = 'https://akabab.github.io/superhero-api/api/all.json'
response = requests.get(url)
# все имена супергероев:
for i in response.json():
    print(i['name'])
```

Оболочка ×

```
>>> %Run superheroapi_test.py
A-Bomb
Abe Sapien
Abin Sur
Abomination
Abraxas
Absorbing Man
Adam Monroe
Adam Strange
Agent Bob
Agent Zero
Air-Walker
```

Задача 2.

Задача №2

У Яндекс.Диска есть очень удобное и простое API. Для описания всех его методов существует Полигон. Нужно написать программу, которая принимает на вход путь до файла на компьютере и сохраняет на Яндекс.Диск с таким же именем.

1. Все ответы приходят в формате json;
2. Загрузка файла по ссылке происходит с помощью метода put и передачи туда данных;
3. Токен можно получить, кликнув на полигоне на кнопку "Получить OAuth-токен".

Важно: Токен публиковать в github не нужно, переменную для токена нужно оставить пустой!

Шаблон для программы

```python
class YaUploader:
    def __init__(self, token: str):
        self.token = token

    def upload(self, file_path: str):
        """Метод загружает файлы по списку file_list на яндекс диск"""
        # Тут ваша логика
        # Функция может ничего не возвращать


if __name__ == '__main__':
    # Получить путь к загружаемому файлу и токен от пользователя
    path_to_file = ...
    token = ...
    uploader = YaUploader(token)
    result = uploader.upload(path_to_file)
```

***Задача №3**

Самый важный сайт для программистов это stackoverflow. И у него тоже есть API Нужно написать программу, которая выводит все вопросы за последние два дня и содержит тэг 'Python'. Для этого задания токен не требуется.