

Работа с библиотекой requests, http запросы



Роман
Гордиенко



Роман Гордиенко

Backend Developer, sdvor.com

План занятия

1. [Что такое HTTP](#)
2. [Client-Server](#)
3. [HTTP запрос](#)
4. [HTTP методы](#)
5. [Параметры в адресе](#)
6. [Заголовки](#)
7. [Коды ответов](#)
8. [Библиотека Requests](#)



Что такое HTTP

Что такое HTTP

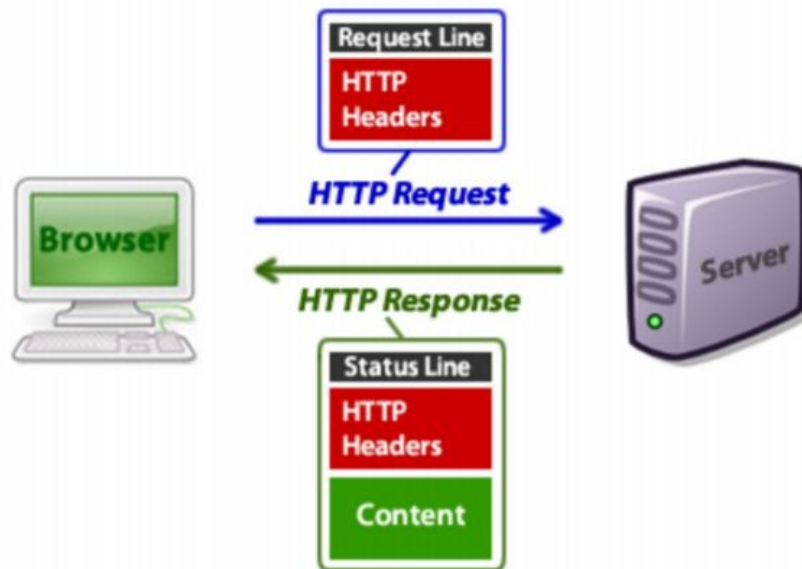
- HTTP (Hypertext Transfer Protocol) – это протокол для передачи данных (текстов, изображений, видео, аудио и т.д.) по интернету.
- HTTP не хранит состояние между запросами. Каждый запрос интерпретируется независимо от других запросов. Если нужно сохранять и передавать состояние – это должны делать реализации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



Client-Server

Client <-> Server

Как выглядит клиент-серверное взаимодействие:



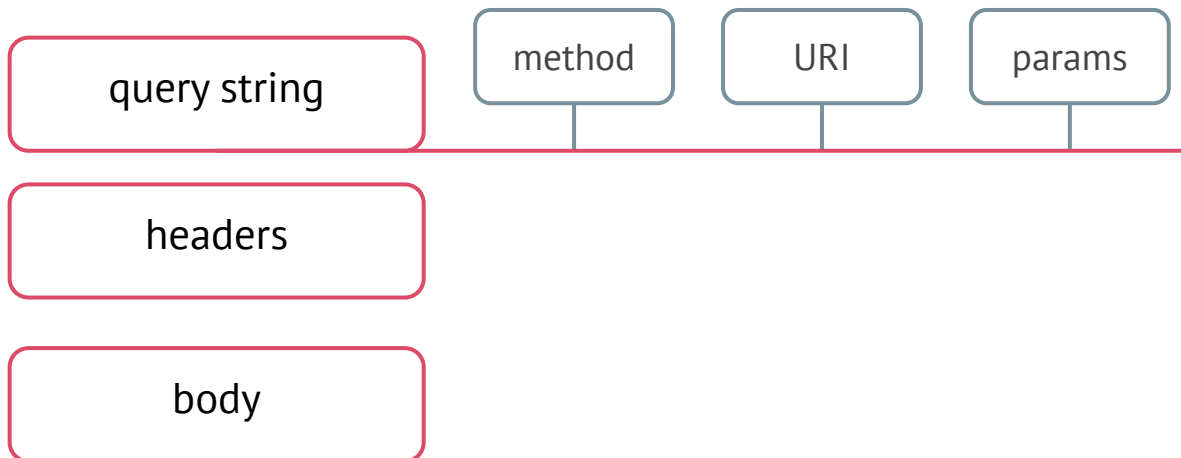
В роли клиента может выступать не только браузер, но и любое устройство и даже другая программа. Когда говорят «клиент-сервер», подразумевают, что есть сторона, которая запрашивает данные, а другая сторона ей отвечает.



HTTP запрос

HTTP Request (запрос)

HTTP request





HTTP методы

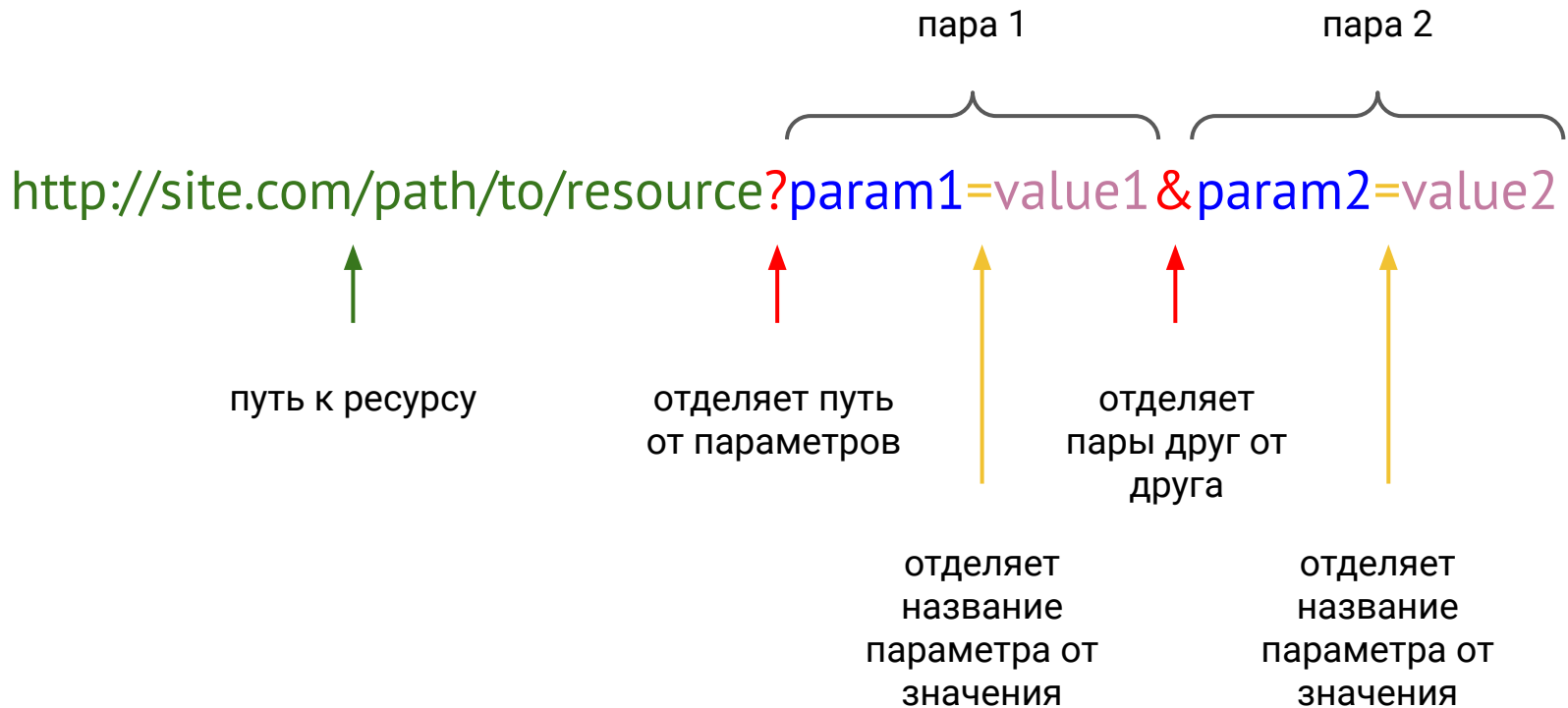
HTTP методы

SAFE METHODS NO ACTION ON SERVER	{	GET	HTTP/1.1 MUST IMPLEMENT THIS METHOD
		HEAD	INSPECT RESOURCE HEADERS
MESSAGE WITH BODY SEND DATA TO SERVER	{	PUT	DEPOSIT DATA ON SERVER — INVERSE OF GET
		POST	SEND INPUT DATA FOR PROCESSING
		PATCH	PARTIALLY MODIFY A RESOURCE
		TRACE	ECHO BACK RECEIVED MESSAGE
		OPTIONS	SERVER CAPABILITIES
		DELETE	DELETE A RESOURCE — NOT GUARANTEED



Параметры в адресе

Структура URL





Заголовки

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



Как делать запросы: POSTMAN



Библиотека requests

Выполнение простого запроса

Выполнение простого GET-запроса:

```
import requests

url = "https://httpbin.org/get"
resp = requests.get(url)

print(resp)
> '<Response [200]>'

print(resp.status_code)
> 200
```

Чтобы сделать POST-запрос, используйте функцию `post`:

```
requests.post(url)
```

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:

```
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})
```



Демонстрация

Работа с Яндекс.Диск:

- получить список файлов
- загрузить файл с компьютера в удаленное хранилище

Демонстрация 2

Скачаем самые популярные видео из раздела /gifs на Реддите.

- Документация: <https://www.reddit.com/dev/api>
- Адрес, по которому можно получить самые популярные видео: <https://reddit.com/r/gifs/top.json?t=day>
- В заголовках необходимо передавать любой нестандартный user-agent. Например, `netology-code`
- Вопрос: Как скачать файл? Как его сохранить в директорию?



Полезные источники

- [Документация requests](#)
- [HTTP-клиент для VS Code](#)
- [Сервис, на котором удобно тестировать свой HTTP-клиент](#)



Домашнее задание

Давайте посмотрим ваше [домашнее задание](#).

- Вопросы по домашней работе задаём в чате!
- Задачи можно сдавать по частям.
- Зачёт по домашней работе проставляется после того, как приняты **все обязательные задачи.**

⌘ нетология

**Задавайте вопросы и
пишите отзыв о лекции!**

Роман Гордиенко