

# Работа с библиотекой requests, http-запросы



# Проверка связи



**Отправьте, пожалуйста, смайлик в чат**

Если у вас все отлично со связью :-)

Если у вас есть какие-то проблемы со связью :-(



**Если у вас нет звука:**

- убедитесь, что на вашем устройстве и на колонках включен звук
- обновите страницу вебинара (или закройте страницу и заново присоединитесь к вебинару)
- откройте вебинар в другом браузере
- перезагрузите компьютер (ноутбук) и заново попытайтесь зайти



Поставьте “+”, если меня видно и слышно

# Николай Свиридов

## О спикере:

- Backend-разработчик
- IT-блогер



---

## 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

---

# HTTP и клиент-серверное взаимодействие

---

## Что такое HTTP?

**HTTP (Hypertext Transfer Protocol)** — протокол для передачи данных (текстов, изображений, видео, аудио и т.д.) по интернету.

**Протокол** — набор правил.

---

# Что такое HTTP?

**HTTP (Hypertext Transfer Protocol)** — протокол для передачи данных (текстов, изображений, видео, аудио и т.д.) по интернету.

**Протокол** — набор правил.

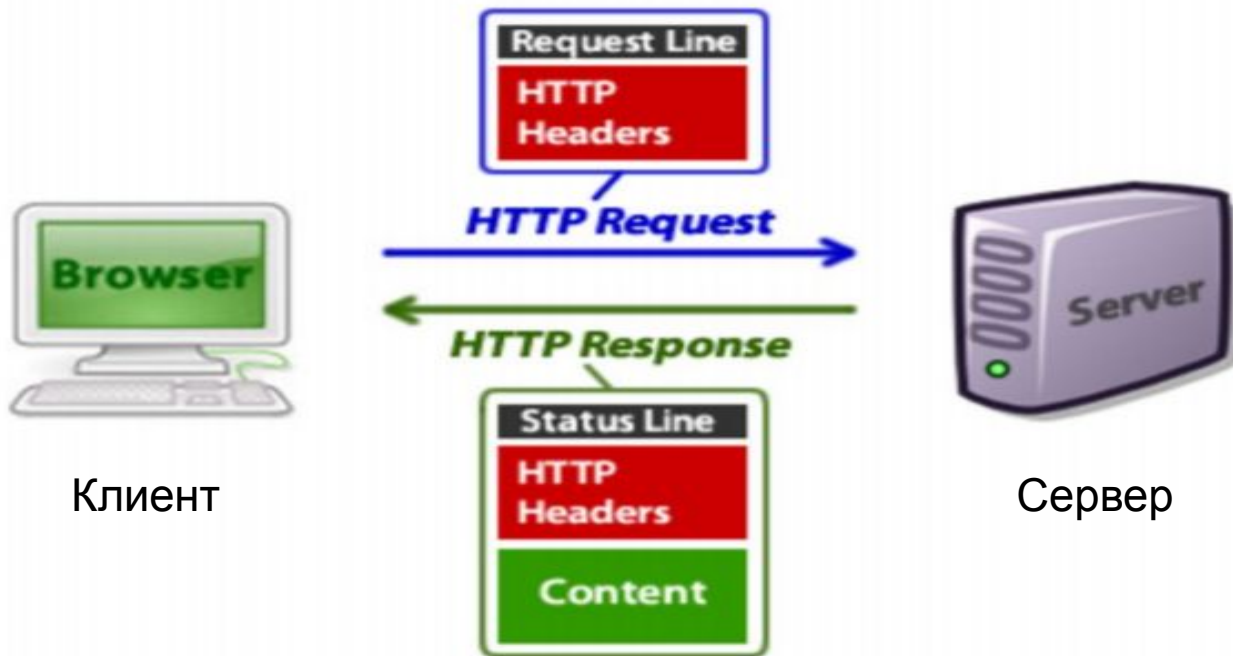
- HTTP не хранит состояние между запросами
- Каждый запрос интерпретируется независимо от других запросов

---

# Client-Server



# Client <-> Server



---

# HTTP с высоты птичьего полета

---

# НТТР с высоты птичьего полета

1. Установить соединение (TCP)

---

# HTTP с высоты птичьего полета

1. Установить соединение (TCP)
2. Отправить/получить запрос

---

# HTTP с высоты птичьего полета

1. Установить соединение (TCP)
2. Отправить/получить запрос
3. Получить/отправить ответ

---

## НТТР с высоты птичьего полета

1. Установить соединение (ТСР)
2. Отправить/получить запрос
3. Получить/отправить ответ
4. Закреть/оставить открытым соединение

# — HTTP запрос и ответ

---

## HTTP Request (запрос)





---

# HTTP Request (запрос)

HTTP request (**server: httpbin.org**)

# HTTP Request (запрос)

HTTP request (**server: httpbin.org**)



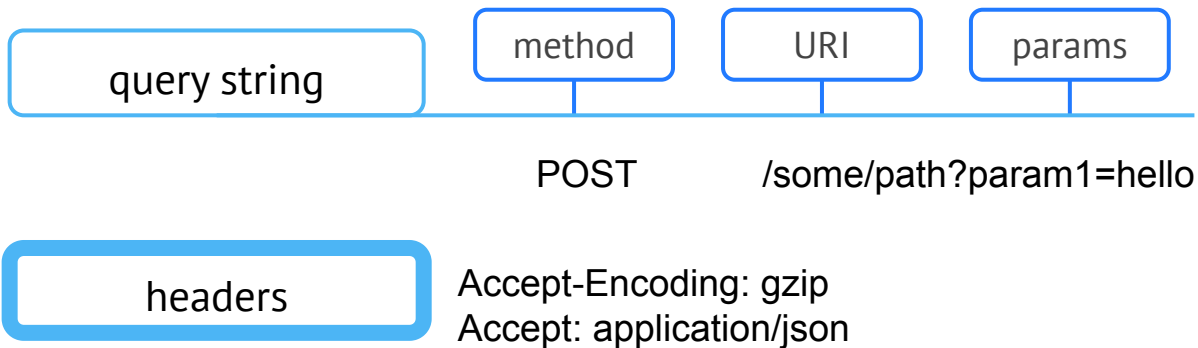
# HTTP Request (запрос)

HTTP request (**server: httpbin.org**)



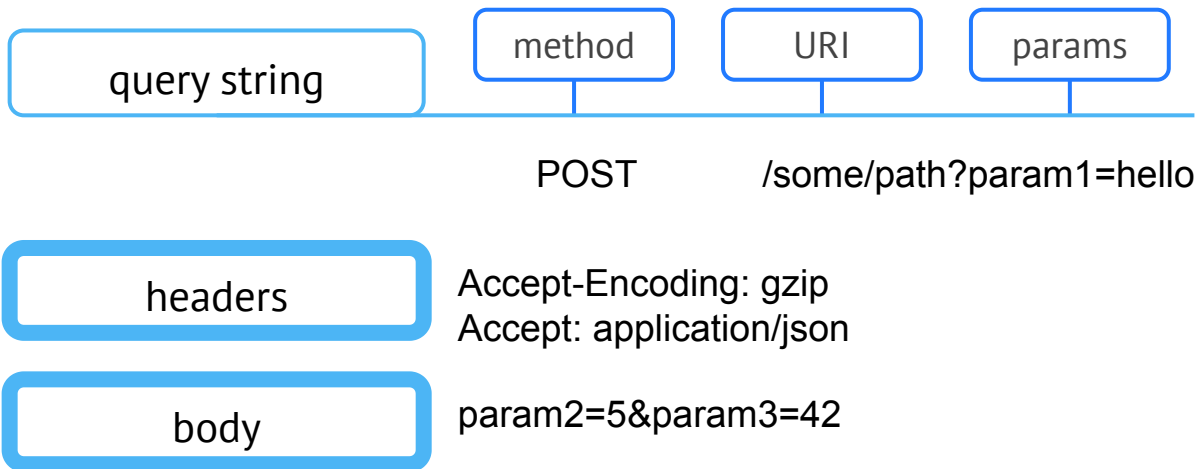
# HTTP Request (запрос)

HTTP request (**server: httpbin.org**)



# HTTP Request (запрос)

HTTP request (**server: httpbin.org**)



---

# 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

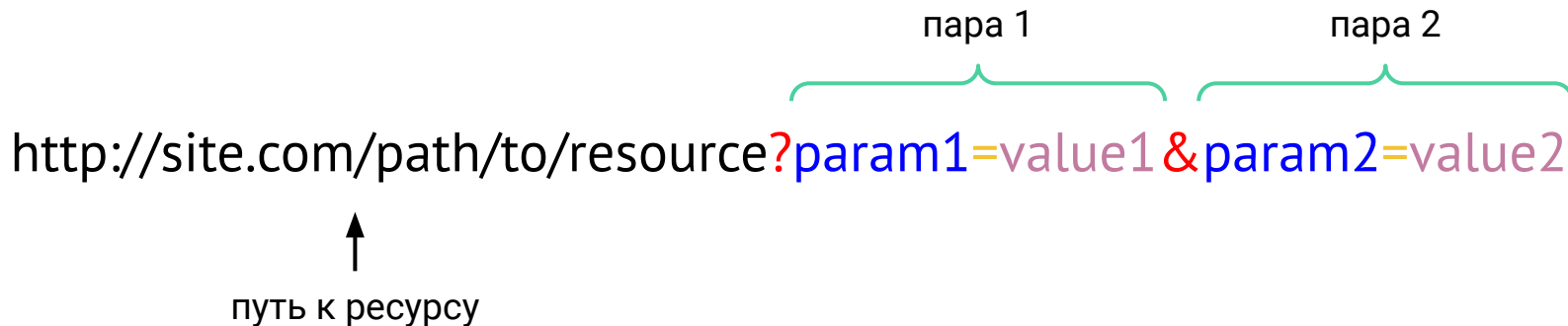


# Структура URL

http://site.com/path/to/resource?param1=value1&param2=value2

↑  
путь к ресурсу

пара 1      пара 2



The diagram illustrates the structure of a URL. The URL is 'http://site.com/path/to/resource?param1=value1&param2=value2'. An upward-pointing arrow is positioned below the path '/path/to/resource', with the text 'путь к ресурсу' (path to resource) underneath it. Two green curly braces are positioned above the query string. The first brace spans 'param1=value1' and is labeled 'пара 1' (pair 1). The second brace spans 'param2=value2' and is labeled 'пара 2' (pair 2). The query string is color-coded: the question mark is red, 'param1' is blue, 'value1' is purple, the ampersand is red, 'param2' is blue, and 'value2' is purple.

# Структура URL

http://site.com/path/to/resource?param1=value1&param2=value2

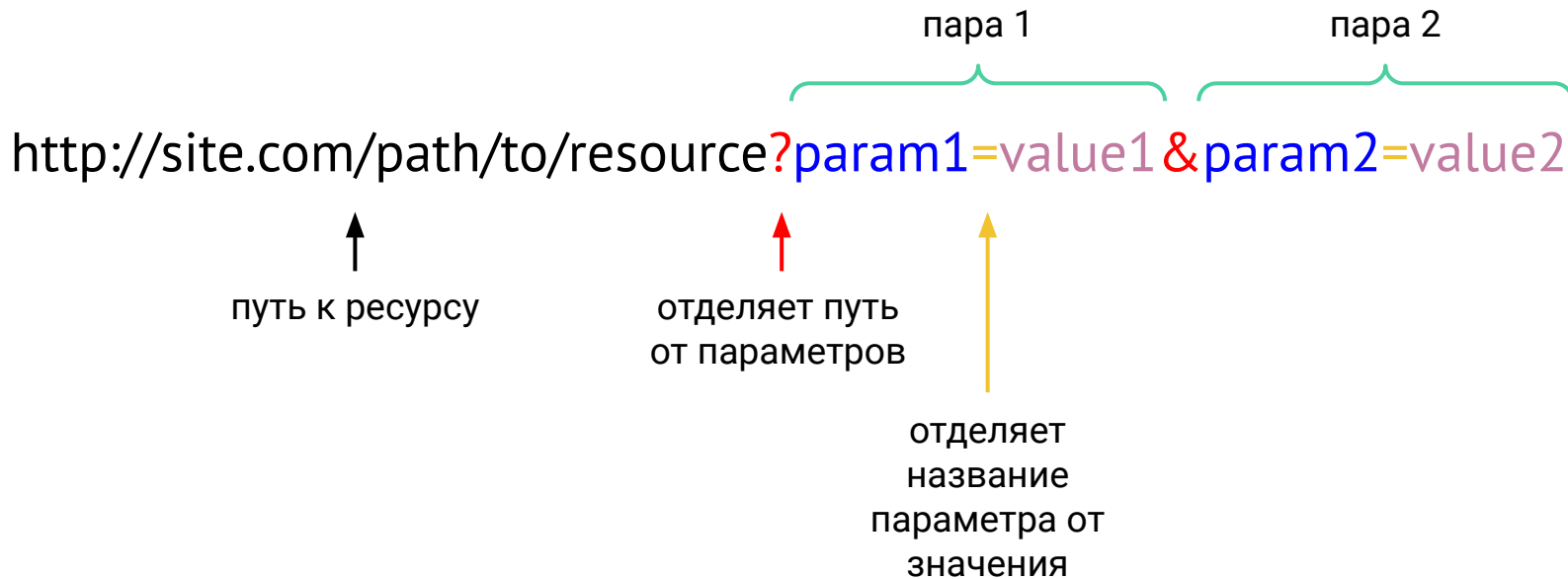
↑  
путь к ресурсу

↑  
отделяет путь  
от параметров

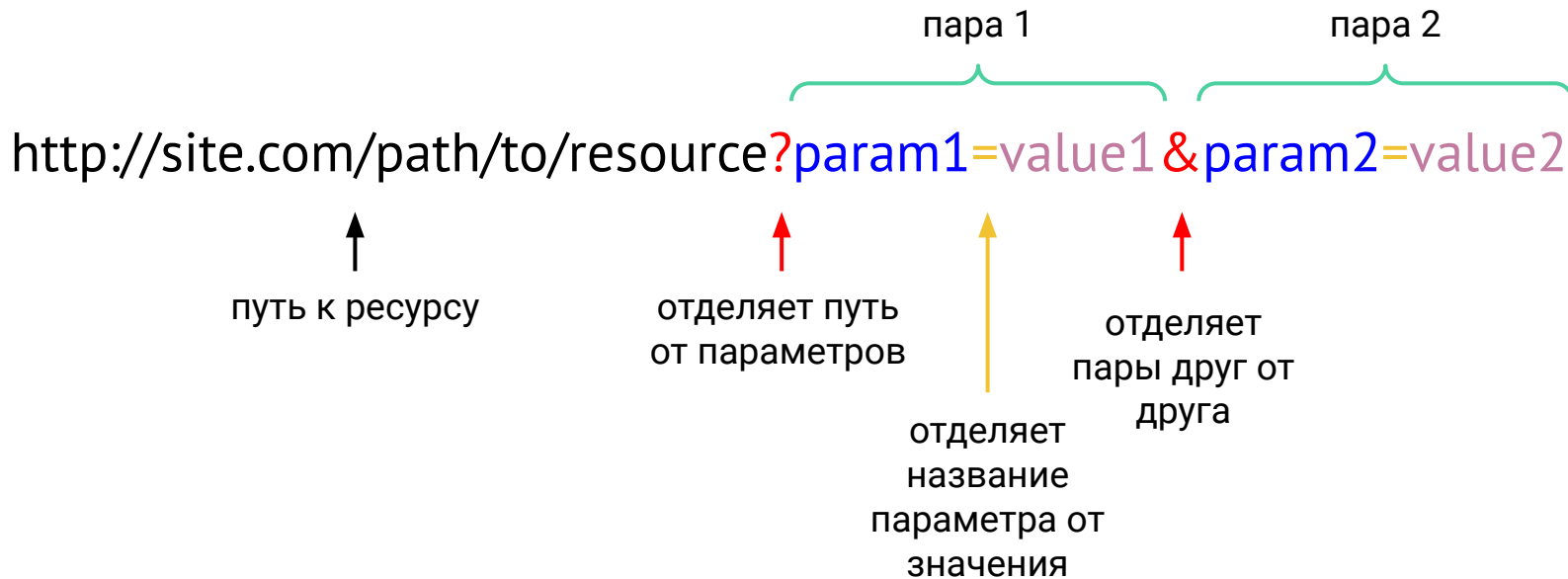
пара 1      пара 2

The diagram illustrates the components of a URL. The URL 'http://site.com/path/to/resource?param1=value1&param2=value2' is shown. A black arrow points to 'path/to/resource' with the label 'путь к ресурсу'. A red arrow points to the '?' character with the label 'отделяет путь от параметров'. Two green curly braces above the query string group 'param1=value1' as 'пара 1' and 'param2=value2' as 'пара 2'.

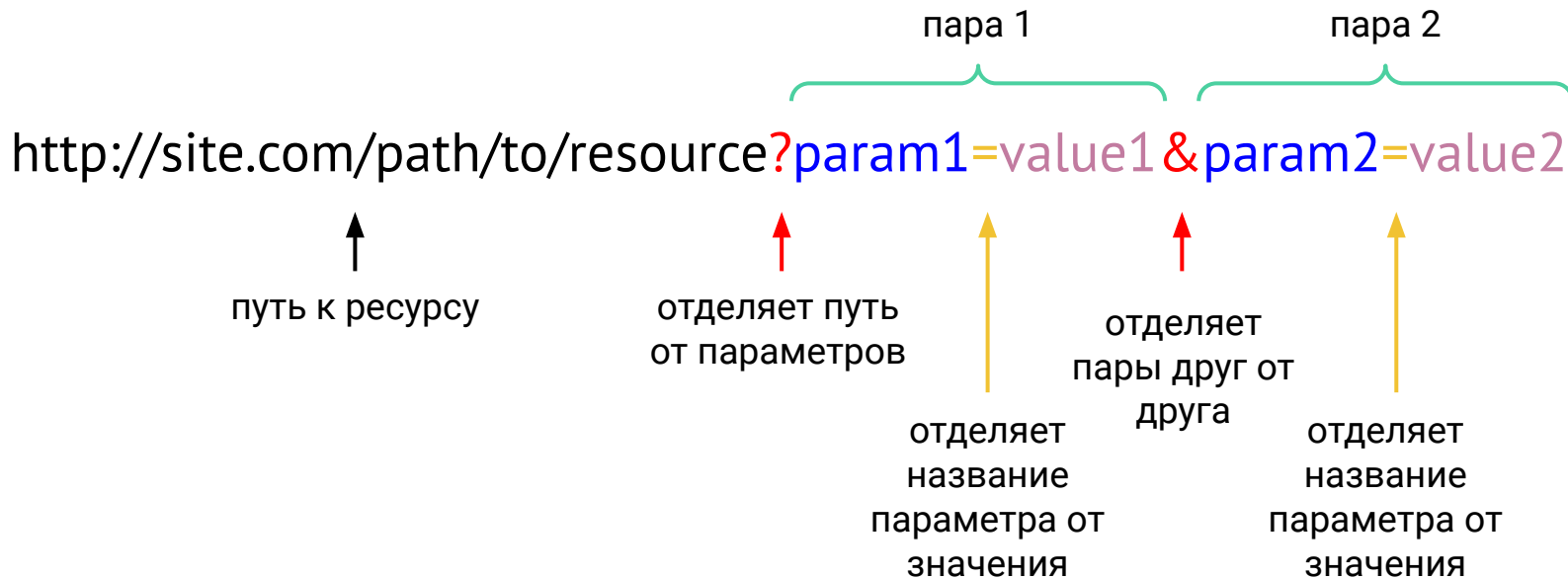
# Структура URL



# Структура URL



# Структура URL



---

## Заголовки и параметры

- В заголовках передают информацию о запросе.  
Например, авторизационные данные пользователя, версию браузера, выставленные cookie, поддерживаемые форматы сжатия данных.

---

## Заголовки и параметры

- В заголовках передают информацию о запросе.  
Например, авторизационные данные пользователя, версию браузера, выставленные cookie, поддерживаемые форматы сжатия данных.
- Параметры запроса содержат информацию о том, что именно запрашивает пользователь.

---

## Заголовки и параметры

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



---

# HTTP ответ

# 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
```

← код ответа

```
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>
```

# Пример HTTP ответа

HTTP/1.x 200 OK

код ответа

Transfer-Encoding: chunked

Date: Sat, 28 Nov 2009 04:36:25 GMT

Server: LiteSpeed

Connection: close

X-Powered-By: W3 Total Cache/0.8

заголовки ответа

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>

# Пример HTTP ответа

HTTP/1.x 200 OK

код ответа

Transfer-Encoding: chunked

Date: Sat, 28 Nov 2009 04:36:25 GMT

Server: LiteSpeed

Connection: close

X-Powered-By: W3 Total Cache/0.8

заголовки ответа

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

тело ответа

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.

---

# HTTP коды

Первая цифра обозначает семантику кода.

- **1XX** — Информационные

---

# HTTP коды

Первая цифра обозначает семантику кода.

- **1XX** — Информационные
- **2XX** — Успешный вызов

---

# HTTP коды

Первая цифра обозначает семантику кода.

- **1XX** — Информационные
- **2XX** — Успешный вызов **200** — **ок**



---

# HTTP коды

Первая цифра обозначает семантику кода.

- **1XX** — Информационные
- **2XX** — Успешный вызов **201** — создано

---

# HTTP коды

Первая цифра обозначает семантику кода.

- **1XX** — Информационные
- **2XX** — Успешный вызов
- **3XX** — Перенаправление

---

# HTTP коды

Первая цифра обозначает семантику кода.

- **1XX** — Информационные
- **2XX** — Успешный вызов
- **3XX** — Перенаправление
- **4XX** — Ошибка на стороне клиента

---

# HTTP коды

Первая цифра обозначает семантику кода.

- **1XX** — Информационные
- **2XX** — Успешный вызов
- **3XX** — Перенаправление
- **4XX** — Ошибка на стороне клиента **403 — недостаточно прав**

---

# HTTP коды

Первая цифра обозначает семантику кода.

- **1XX** — Информационные
- **2XX** — Успешный вызов
- **3XX** — Перенаправление
- **4XX** — Ошибка на стороне клиента **404 — не найдено**

---

# HTTP коды

Первая цифра обозначает семантику кода.

- **1XX** — Информационные
- **2XX** — Успешный вызов
- **3XX** — Перенаправление
- **4XX** — Ошибка на стороне клиента **405 — метод не поддерживается**

---

# HTTP коды

Первая цифра обозначает семантику кода.

- **1XX** — Информационные
- **2XX** — Успешный вызов
- **3XX** — Перенаправление
- **4XX** — Ошибка на стороне клиента
- **5XX** — Ошибка на стороне сервера

---

# HTTP коды

Первая цифра обозначает семантику кода.

- **1XX** — Информационные
- **2XX** — Успешный вызов
- **3XX** — Перенаправление
- **4XX** — Ошибка на стороне клиента
- **5XX** — Ошибка на стороне сервера **500 — внутренняя ошибка сервера**



---

## Обобщаем

- **Метод** — это действие, которое клиент просит сделать на сервере.
- Клиент и сервер общаются методом запросов и ответов.
- HTTP коды позволяют отслеживать статус взаимодействия

---

# Библиотека requests

# Атрибуты и методы объекта response

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

---

## Полезные источники

- Документация requests
- HTTP-клиент для VS Code
- Сервис, на котором удобно тестировать свой HTTP-клиент

# — Домашнее задание

---

## Задание 1

Кто самый умный супергерой?

Есть **API по информации о супергероях** с информацией по всем супергероям.

Нужно определить кто самый умный(intelligence) из трех супергероев — Hulk, Captain America, Thanos.

## Задание 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".

HOST: <https://cloud-api.yandex.net:443>

---

## Задание 3 (необязательное)

Самый важный сайт для программистов это [stackoverflow](#).

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.





**Ваши вопросы?**