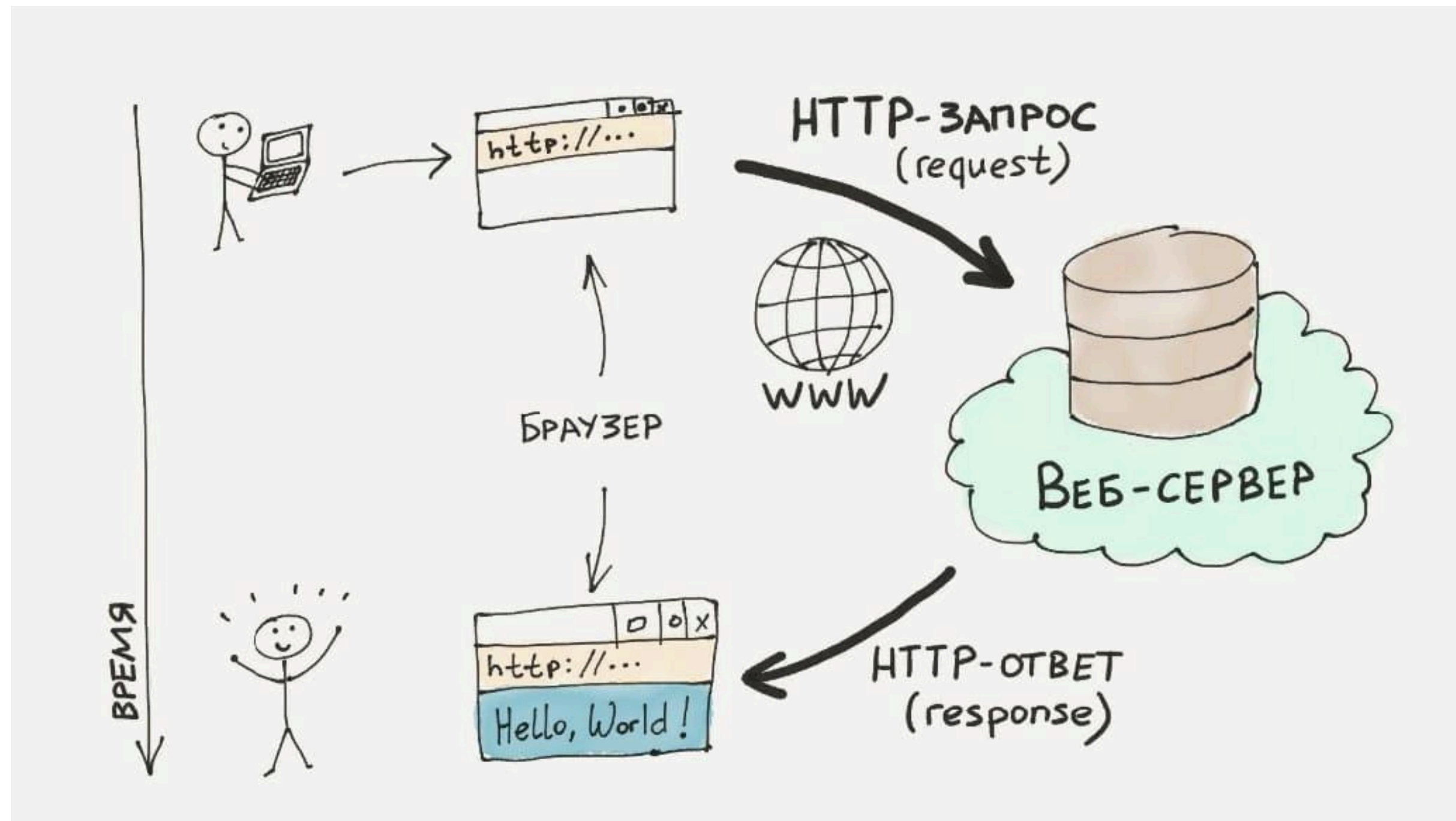




Создаём первое API: Основы веба и асинхронность в FastAPI

“ИТ-класс”. Бэкенд-разработка на Python.
Бусыгин Дмитрий, 2025 год.

Как устроен интернет



1. Клиент отправляет запрос (request) по адресу
2. Происходит обработка запроса сервером
3. Отправка ответа (response) обратно клиенту

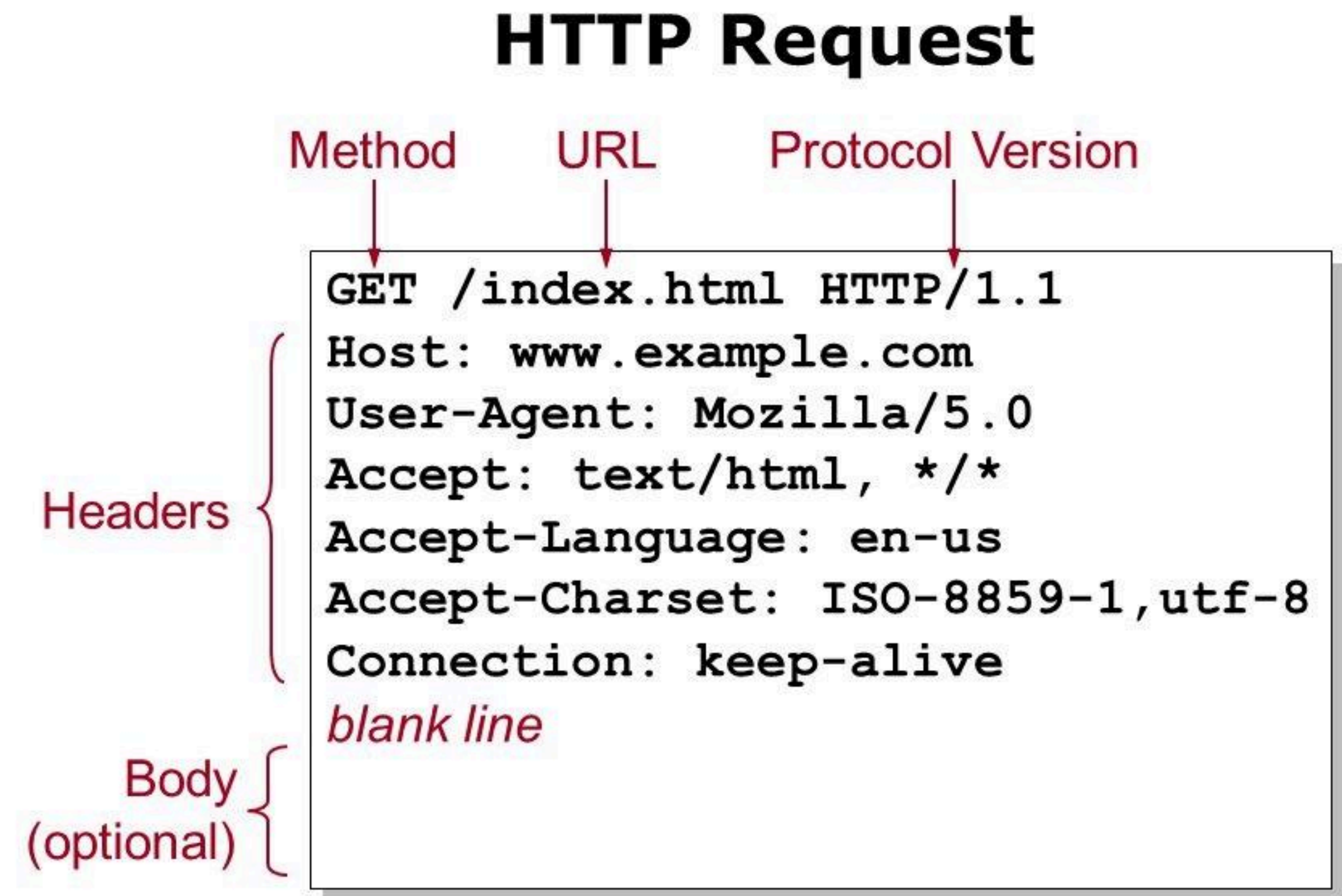
API (Application Programming Interface)

— это набор правил и инструкций, которые позволяют разным программным приложениям общаться между собой и обмениваться данными.

Бэкенд и фронтенд, например, тоже взаимодействуют через API.

Из чего состоит запрос

1. Стартовая строка ("request line")
 - a. HTTP-метод (GET, POST, ...)
 - b. Адрес ресурса (URL)
 - c. Версия протокола
2. Заголовки ("headers") — метаданные, уточняющие запрос
3. Тело запроса ("body") — входные данные от самого клиента



Основные HTTP-методы

GET — получить данные

- Длина запроса < 2048 символов
- Видны аргументы, которые отправляются (не отправляйте персональные данные)

`http://localhost:8000/endpoint?arg=value&arg2=value&...`

POST — отправить данные

- Размер данных неограничен
- Аргументы передаются отдельно в теле запроса — нельзя прочесть, какие данные были отправлены.

`http://localhost:8000/endpoint`
`body: {"arg1": "value1", "email": "value1"}`

Дополнительные HTTP-методы

PUT — обновить данные

PATCH — обновить данные частично

DELETE — удалить данные

OPTIONS — проверка возможностей сервера

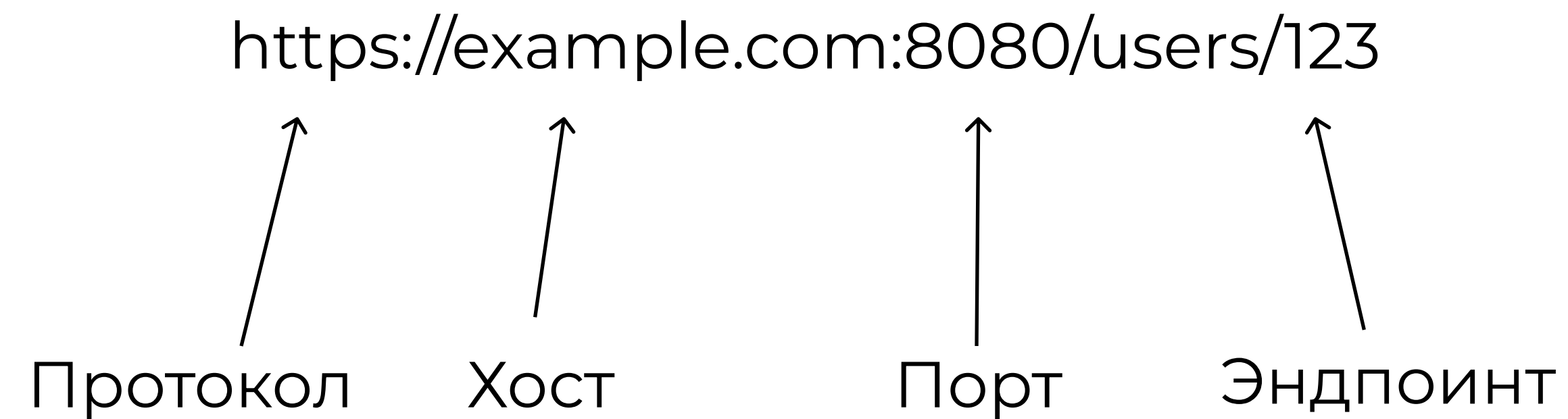
Но никто не мешает пользоваться
везде только GET и POST :)

GET	/pet/{petId}	Find pet by ID
PUT	/pet	Update an existing pet
DELETE	/pet/{petId}	Deletes a pet
POST	/pet/{petId}/uploadImage	uploads an image

Адрес ресурса

URL (Uniform Resource Locator) — это стандартизированный адрес в интернете, указывающий местоположение веб-страницы, файла или другого ресурса.

https://example.com:8080/users/123



Протокол Хост Порт Эндпоинт

The diagram illustrates the structure of the URL 'https://example.com:8080/users/123'. Four labels are positioned below the URL, each with an arrow pointing to a specific part of the address: 'Протокол' (Protocol) points to 'https://', 'Хост' (Host) points to 'example.com', 'Порт' (Port) points to ':8080', and 'Эндпоинт' (Endpoint) points to '/users/123'.

JSON

JSON (JavaScript Object Notation) — текстовый формат обмена данными, основанный на JavaScript. Но при этом формат независим от JS и может использоваться в любом языке программирования.

Именно в таком формате передаются данные от клиента к серверу и обратно.

Это словарь, который потом конвертируется в строку, которую мы отправляем, а на бэкенд-части из строки обратно делается словарь.

Пример JSON

---->

```
{  
  "name": "John Doe"  
  "age": 31  
  "hobbies": ["soccer", "chess", "math", "art"]  
}
```

Ответы и их коды состояния

Самые распространённые

- 200 — **OK**
- 400 — Некорректный запрос (я тебя не понимаю)
- 401 — Клиент не авторизован (ты кто?)
- 403 — Запрещено (тебе нельзя)
- 404 — Не найдено (я не знаю)
- 422 — Необрабатываемый экземпляр
(я тебя вроде понимаю, но запрос странный)
- 500 — Ошибка сервера
- 501 — Сервис нереализован
- 503 — Сервис недоступен

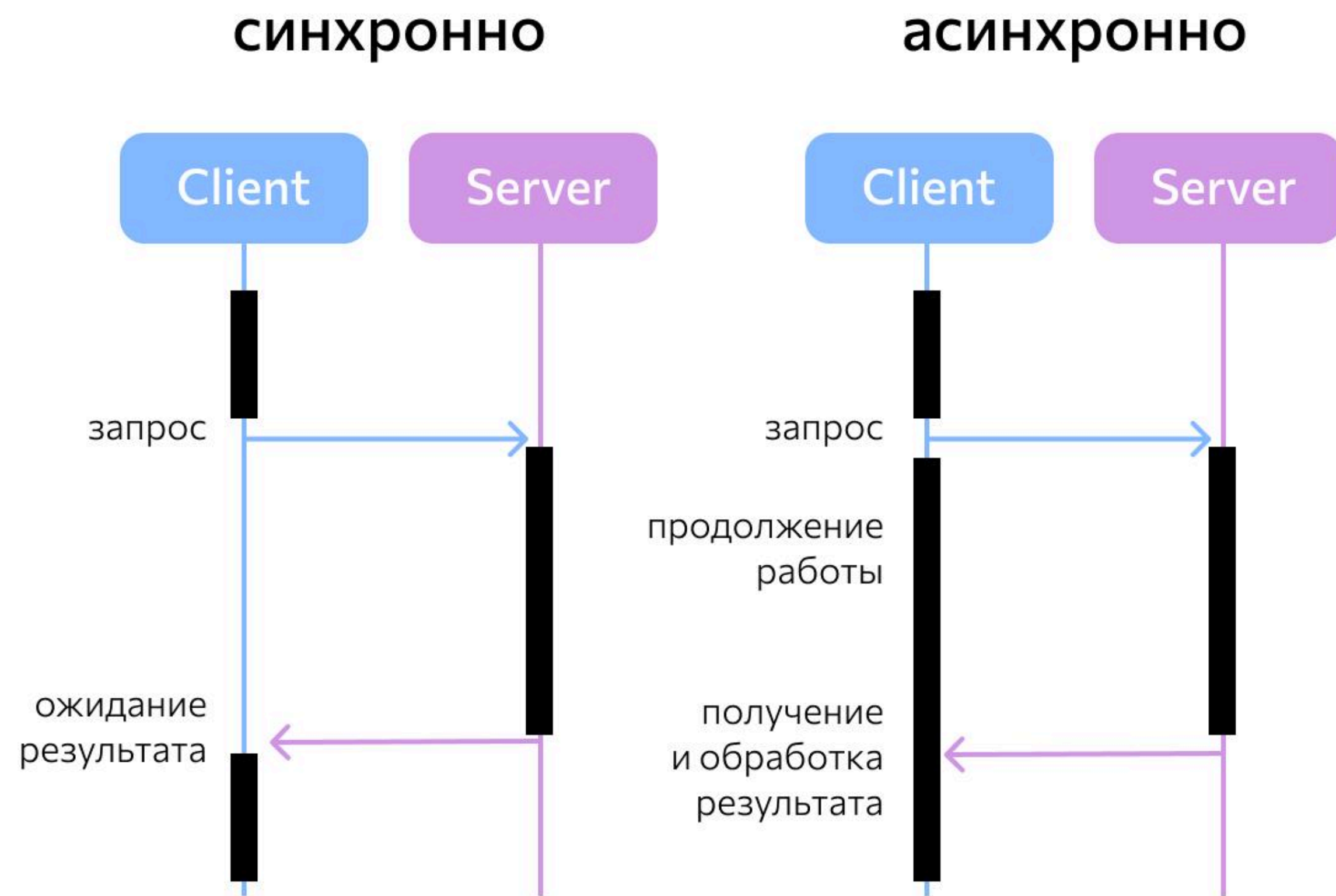


AWWW...DON'T CRY.

It's just a 404 Error!

**Но как сервера принимают
тысячи запросов в секунду?**

Асинхронность



Асинхронность — это концепция выполнения задач без блокировки основного потока программы, позволяющая выполнять другие действия, пока одна из операций ожидает завершения.

Пока поток №2 занят, поток №1 может работать дальше и принять ответ от №2 позже.

Не путать с многопоточностью

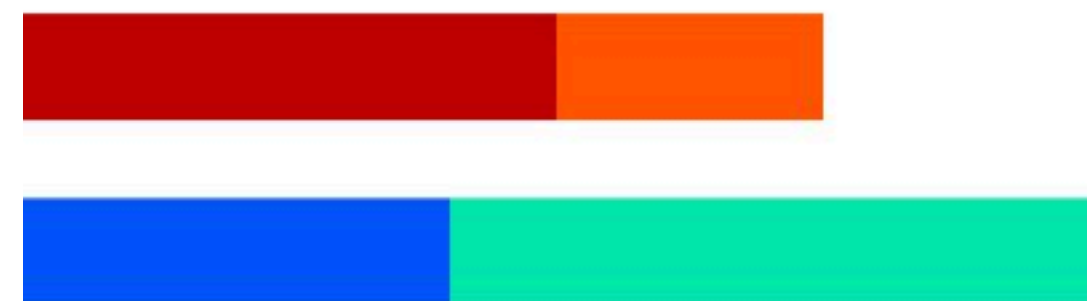
Synchronous



Asynchronous?



Multithreaded



Time



Многопоточность — каждый поток берет на себя всю задачу.

Асинхронность — каждая подзадача грамотно делегируется соответствующему потоку.

Пример: сервер ждет ответ от базы данных 2 секунды. Синхронно он бы просто ждал, а асинхронно в это время обрабатывает другие запросы

Практика на `asyncio` в Google Colab

Асинхронная обработка web-запросов



FastAPI — веб-фреймворк для создания API со встроенной асинхронностью. Один из самых быстрых и популярных веб-фреймворков.



Pydantic — это библиотека для Python, для валидации данных. Она помогает разработчикам гарантировать, что входные данные соответствуют установленным правилам и форматам.

Как тестировать наш API



Postman — это программа для тестирования API и REST. Через него тестировщики отправляют различные http-запросы к приложениям, получают и проверяют ответы.

- Не нужно писать код
- Сразу видно запросы и ответы

Доступна веб-версия!