PHASE 2
WEEK 2

DAY 1

ELBRUS/BOOTCAMP

План

1. Термины (API, REST)
2. REST принципы и REST API (RESTful API)
3. Инструменты работы с API
4. Альтернативы подходу REST

REST API

Термины

API

application programming interface

программный интерфейс приложения

Термины / API

API

interface

интерфейс

Термины / API

Интерфейс

Общая граница между компьютерными системами и правила её “перехода”.

Термины / API

Интерфейсы между…

- супермаркетом и покупателем?
- рестораном и гостем?
- ноутбуком и пользователем?
- смартфоном и пользователем?

Термины / API

API

programming interface

программный интерфейс

Термины / API

Программный интерфейс

Набор способов программного взаимодействия.

Термины / API

“**Программное** взаимодействие”

Что это значит?

Взаимодействие, которое можно описать в виде программы.

Термины / API

API

application

приложение (прикладная программа)

Термины / API

Приложение (прикладная программа)

- программа для поручения компьютеру прикладных задач;
- выполняет практические задачи пользователя.

Термины / API

API

application programming interface

программный интерфейс приложения

Термины

REST

representational state transfer

передача репрезентативного состояния

Термины / REST

REST

representational state

репрезентативное состояние

Термины / REST

“**Репрезентативное** состояние”

Что это значит?

Набор свойств, которые **достаточно полно** описывают состояние в какой-то момент.

Термины / REST

REST

state transfer

передача состояния

Термины / REST

Передача состояния

Передача набора свойств, описывающих состояние программы.

Термины / REST

REST — архитектурный стиль.

Используется для разработки API, работающих на основе протокола HTTP, где передача данных часто в JSON.

REST API

API, построенные по принципам REST

Режим зануды:

Такие API точнее называть RESTful API

REST принципы и REST API

Принципы REST архитектуры

1. Клиент и сервер — разные части системы
2. Состояние клиента не хранится на сервере между запросами
3. Возможность кэширования ответов сервера
4. Единообразный интерфейс
5. Послойная система (клиенту неважно, общается ли с сервером напрямую)
6. Код по требованию (необязательно; расширяет возможности клиента)

REST принципы и REST API

REST !== HTTP методы

HTTP-методы (aka “HTTP-глаголы”) используются на практике для **реализации** принципов единообразного интерфейса и возможности кэширования.

REST принципы и REST API

Сущность (ресурс)

- Множество чего-либо ИЛИ
  единственно возможный экземпляр
- Может содержать другие сущности / подмножества
- Определяется логикой вашей системы

REST принципы и REST API

Именование сущностей

Используйте существительные в нужном числе. Пример:

- users *(множество пользователей)*
- articles *(множество статей)*
- profile *(профиль пользователя; если это единственный доступный профиль для пользователя)*
- status *(статус сервиса; если это единственный статус для этой системы)*

REST принципы и REST API

HTTP-глаголы для обозначения действий

Например:

- GET — получить сущность или множество
- POST — создать ресурс
- PUT — изменить существующий ресурс
- DELETE — удалить ресурс

REST принципы и REST API

Пример №1

- GET /users — *получить список пользователей*
- POST /users — *создать нового пользователя*
- PUT /users/8 — *заменить данные пользователя с ID 8*
- DELETE /users/8 — *удалить данные пользователя с ID 8*

REST принципы и REST API

Идемпотентность (idempotence)

Свойство операции выдавать один и тот же результат при её повторе.

Примеры: умножение на единицу, нажатие на кнопку пешеходного перехода.

REST принципы и REST API

Идемпотентные HTTP-запросы

… если реализованы корректно.

GET — если ресурс не был изменён между запросами.

PUT — если ресурс существует.

DELETE — может быть реализован как идемпотентный.

REST принципы и REST API

POST-запрос не идемпотентный

… если реализован корректно.

Обработчик POST-запроса должен создавать новый экземпляр сущности при **каждом** запросе.

То есть, результат повторного выполнения разный.

REST принципы и REST API

Пример №2

- GET /articles — *получить список статей*
- GET /articles/latest — *получить одну последнюю статью*
- GET /articles/3/comments — *получить комментарии к статье с #3*
- POST /articles/3/comments — *создать новый комментарий к статье #3*
- PUT /articles/3/comments/5 — *заменить комментарий #5 в статье #3*
- DELETE /articles/3/comments/5 — *удалить комментарий #5 в статье #3*

REST принципы и REST API

Пример №3

- GET /status
- GET /movies
- POST /movies
- PUT /movies/:id
- DELETE /movies/:id
- POST /movies/:id/reviews
- GET /actors
- POST /actors
- PUT /actors/:id
- DELETE /actors/:id
- PUT /movies/:movieId/actors/:actorId
- DELETE /movies/:movieId/actors/:actorId

REST принципы и REST API

Параметр (parameter)

GET /movies/:id

Что в этом запросе — параметр?

Запросы с параметрами называются **параметризованными** (parameterized)

REST API

Любые API, построенные по принципам REST

Чаще всего такие API **реализованы** на основе HTTP

Инструменты

Упрощают разработку, тестирование, поддержку API

… сотни и тысячи их.

Инструменты работы с API

Postman — для тестирования, разработки и автоматического тестирования API.

https://www.postman.com/downloads/

Инструменты работы с API

Insomnia — для тестирования, разработки и автоматического тестирования API.

https://insomnia.rest/download

Инструменты работы с API

cURL — консольный инструмент.

https://curl.se/docs/

Инструменты работы с API

IDE — большая часть сред разработки имеет встроенный HTTP-клиент, либо пакеты расширений для работы с HTTP.

Инструменты работы с API

Swagger — инструмент для API-first разработки.

Методология API-first предполагает разработку документации API **до** работ по реализации API.

https://swagger.io/

Альтернативы

… не REST-ом единым

Альтернативы подходу REST

GraphQL — язык запросов по графам — множествам и их парным связям.

Запрашиваем только необходимое в формате, который хотим. В том числе из нескольких источников или сущностей.

Альтернативы подходу REST

RPC — вызов функций (процедур) на удалённом компьютере.

gRPC — протокол, подход к организации API. Не может быть реализован в HTTP 1.1, привязан к HTTP 2.

Альтернативы подходу REST

SOAP — широко использовался до создания REST.

Сейчас в основном используется в легаси-проектах.