REST API

Профессия Java- разработчик на Hexlet

Преподаватель : Яковлев Егор

Вопросы к лекции :

Что такое API?
Что такое REST API?
Как проектировать REST API?

План

1. API
2. REST API
3. Ограничения REST API
4. REST архитектура
5. Best practice: REST API
6. REST API: Javalin

API

API (программный интерфейс приложения) — описание способов (набор классов, процедур, функций, структур или констант), которыми одна компьютерная программа может взаимодействовать с другой программой.

REST API

A REST API (also known as RESTful API) is an application programming interface (API or web API) that conforms to the constraints of REST architectural style and allows for interaction with RESTful web services.

REST

REST - набор архитектурных ограничений - НЕ протокол и НЕ стандарт
Когда клиент делает запрос через REST API - передается состояние представления ресурса на сервер
данные могут предоваться по протоколу HTTP
в формате: JSON, XML, HTML...

Ограничения REST

- единый интерфейс;
- разграничение клиента и сервера;
- нет сохранения состояния;
- кэширование всегда разрешено;
- многоуровневая система

Единый интерфейс

Ресурсы должны быть однозначно идентифицированы посредством одного URL-адреса и только с помощью базовых методов сетевого протокола (DELETE, PUT, GET, HTTP).

Разграничение клиента и сервера

пользовательский интерфейс и вопросы сбора запросов — на стороне клиента
доступ к данным, управление рабочей нагрузкой и безопасность — на стороне сервера

Нет сохранения состояния

Все клиент-серверные операции должны быть без сохранения состояния. Любое необходимое управление состоянием должно осуществляться на клиенте, а не на сервере.

Кэширование всегда разрешено

Все ресурсы должны разрешать кэширование, если явно не указано, что оно невозможно.

Многоуровневая система

REST API допускает архитектуру, которая состоит из нескольких уровней серверов.

REST архитектура

URL-адрес конечной точки;
метод HTTP;
заголовки HTTP;
данные тела (body)

URL

Приложение с REST API будет определять один или несколько URL-адресов конечных точек с доменом, порт, путь, строки запроса.

Best practice: URL

существительные, не глаголы (accounts, users)
множественное число (/users, /accounts)
версионирование (/v1.0, /v1.2)

Методы HTTP

GET - безопасный и идемпотентный;
POST - небезопасный и неидемпотетный;
PUT - небезопасный, но идемпотентный;
DELETE - небезопасный, но идемпотентый

Заголовки

Такая информация, как токены аутентификации или файлы cookie, может содержаться в заголовке HTTP-запроса.

Body

Данные обычно передаются в теле HTTP идентично HTML-представлению или путем отправки одной строки данных в кодировке JSON.

Best practice: REST

1. Конечные точки в URL – имя существительное, не глагол
2. Множественное число
3. Документация
4. Версия вашего приложения
5. Пагинация
6. Использование SSL
7. HTTP методы
8. Эффективное использование кодов ответов HTTP

REST API: Javalin

Демо

Почему REST API?

разделение клиента и сервера
открытость - понятные правила взаимодейтсвия
дружественность к кэшу
поддержка нескольких форматов

Минусы REST API

нет единой структуры
большие полезные нагрузки - стало поводом для создания GraphQL
проблемы чрезмерных или недостаточных данных

Домашнее задание

hexlet program download java rest-api
hexlet program submit java

Вопросы?