WHAT IS A REST API?

Paul Sembereka

What is An API?

- Application Program Interface
- APIs are everywhere
- Contract provided from on piece of software to another
- Structured request and response

Look at this

https://www.youtube.com/watch?v=s7wmiS2mSXY

What is REST

- Representational State Transfer
- Architecture style for designing networked applications
- Relies on stateless, client-server protocol, almost alway HTTP
- Treats server objects as resources that can be created or destroyed
- Can be used by virtually any programming language

REST quick tips

1. Use HTTP Verbs to Mean Something
   - greatly enhance the clarity of what a given request does
2. Sensible Resource Names
   - Improves the clarity of what a given request does
3. XML and JSON
   - Favor JSON support as the default, but unless the costs of offering both JSON and XML are staggering, offer them both.
4. Create Fine-Grained Resources
   - Make it easy on yourself and start with small, easily defined resources, providing CRUD functionality on those
5. Consider Connectedness
   - self reference informs clients how the data was or can be retrieved

REST resources

- A resource is an object with a type, associate data, relation to other resources and a set of methods that operate on it.
- Similar to object instances in object-oriented programming.
- Each resource has its own address or URI—every interesting piece of information the server can provide is exposed as a resource.
- In deciding what resources are within your system, name them as nouns.
- RESTful URI should refer to a resource that is a thing instead of referring to an action.
- Nouns have properties as verbs do not

REST resources

Some example resources are:

- Users of the system.
- Courses in which a student is enrolled.
- A user's timeline of posts.
- The users that follow another user.
- An article about horseback cooking

HTTP Verbs

The HTTP verbs comprise a major portion of our “uniform interface” constraint and provide us the action counterpart to the noun-based resource

HTTP Verbs (CRUD)

- GET: Retrieve data from specified resource
- POST: Submit data to be processed to a specified resource
- PUT: Update a specified resource
- DELETE: delete a specified resource

- HEAD: same as GET but does not return body
- OPTIONS: Returns the supported HTTP methods
- PATCH: Update partial resources

GET verb

The HTTP GET method is used to retrieve (or read) a representation of a resource. Returns a representation in XML or JSON.

Examples:

1. GET http://www.example.com/customers/12345
2. GET http://www.example.com/customers/12345/orders
3. GET http://www.example.com/buckets/sample

According to the design of the HTTP specification, GET (along with HEAD) requests are used only to read data and not change it. Therefore, when used this way, they are considered safe.

PUT verb

Most-often utilized for update capabilities.

Examples:

1. PUT http://www.example.com/customers/12345
2. PUT http://www.example.com/customers/12345/orders/98765
3. PUT http://www.example.com/buckets/sample

PUT is not a safe operation, in that it modifies (or creates) state on the server,

POST verb

Most-often utilized for update capabilities.

Examples:

1. POST http://www.example.com/customers
2. POST http://www.example.com/customers/12345/orders

POST is neither safe or idempotent.

DELETE verb

It is used to delete a resource identified by a URI.

Examples:

1. DELETE http://www.example.com/customers/12345
2. DELETE http://www.example.com/customers/12345/orders
3. DELETE http://www.example.com/buckets/sample

DELETE is NOT safe.

Endpoints

The URI/URL where api/service can be accessed by a client application

Authentication

Some API's require authentication to use their services

Examples

https://localhost:3300/users/test?token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiI1YzQ5ODJiYjY2NzRlMTE3YzQxNDUzN2MiLCJpYXQiOjE1NDg3NjA0Njd9.JOqfAjMHWnr76nsgg5uROfNckj_PB-mZghdr_kR3jWc

Tools

Postman

JSON editor

Browser