RESTful API

Monday, April 6, 2020

9:57 AM

**API:**

API stands for Application Program Interface and is an entry gate to a distributed software system where the API expose a set of resources and available functionalities of a software service to other parties next to defining a clear contract to inform clients from the proper and correct way of communication with the server. As a result, different clients know how they can interact with the server by knowing following essential information:

* 1. Available resources, functionalities and operations.
  2. Correct and expected format of sending requests to Server
  3. The format of server response next to its data transmission type like JSON , XML
  4. Table of error code in order to diagnose what was the problem in the communication

In this manner the client know how to submit their request properly to server for using its functionalities such that server understand the request and process them next to be aware of what server is going to send back as the response to the client in defined format.

API can exist almost anywhere but they are fundamental concept of Web Development

An API in web context would be called as Web Service but not all the APIs are webservices.

**REST:**

REST stands for Representational State Transfer which is an architecture style for designing network applications based on stateless client-server protocol which usually relies on HTTP as the most used protocol for transferring data. It treats the server objects as an resource.

HTTP or Hypertext Transfer Protocol is the an application layer in the OSI model for sending and receiving messages over network.

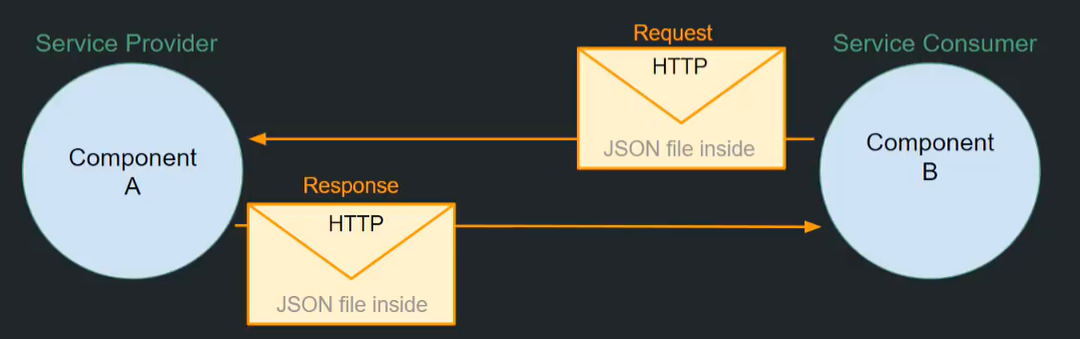
All programming languages have feature to interact with a RESTful API by creating requests in different ways next to interpret the response of server in order to satisfy their needs.

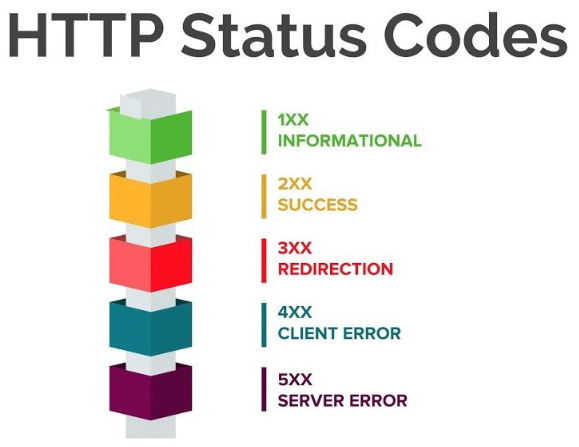
**RESTful API:** RESTful API is an API whose contract is being designed according REST principles :

Machine generated alternative text:
Stateless 
Cacheable 
Client Server 
Layered System 
API 
Uniform Interface 
Code on Demand 

* 1. Stateless refer to a concept that all the required information that server needs to understand and process the request is being send within the request via either URL query parameter, HTTP header or body.
  2. Uniform Interface refers to concept that resources are being exposed and accessed in a consistent way using resource identifiers (URI) for the sake of separation of concerns
  3. Use HTTP protocol as common way of communication for sending and receiving data
  4. Use only HTTP method/verb for any kind of operation on the resources
  5. The format of transmitting data is being independent (JSON/XML/…)
  6. Response Code would be provided upon the response of each requests in order to indicate the result of that request. There are a set of predefined status codes

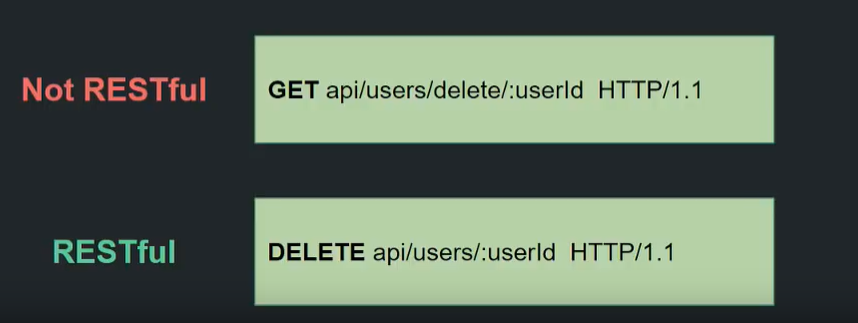
In this manner, in the response of a request, the API will return the state and data of an object instead of the object by itself which means it only gives the data and all other operations should be done in the server by making requests. So we are fetching and sending the data only.







* The data required would be transmitted within the body of requests & responses



Most public API has some documentation to inform the user upon the endpoint or actions that user can do that specific API.

Response Code:

**HTTP Methods (Verbs):**

Machine generated alternative text:
METHODS OF REST API 
Create 
Read 
Update 
Delete 
POST 
GET 
PUT 
DELETE 

We call any kind of data as resources either be text, image, sound or objects. In Restful API every resource should be identified by unique id within its own resource group.

* 1. **GET:** for **retrieving** the resources (data) out from a specified server.

We use it any time that we want to receive any kind of data from a server, like viewing a website or accessing the data or downloading or any other means of data consuming.

In the area of designing a RESTful API, get method should give back the same object every time and also it should not make any changes to the database at all.

* 1. **POST:** for submitting (**creating**) the data to be processed by the specified server

We usually use this method for example when we are sending our data of a filled form to be processed which also can be done with get but since get expose the data, it is not secure so we use post instead. We need to send data along this.

* 1. **PUT:** for **updating** an existing resource on the server

We usually use this when we want to perform some modifications to an existing asset where we need to send our requests to an end-point where we defined that id of that asset to let the server know which we want to update. We need to send data along this.

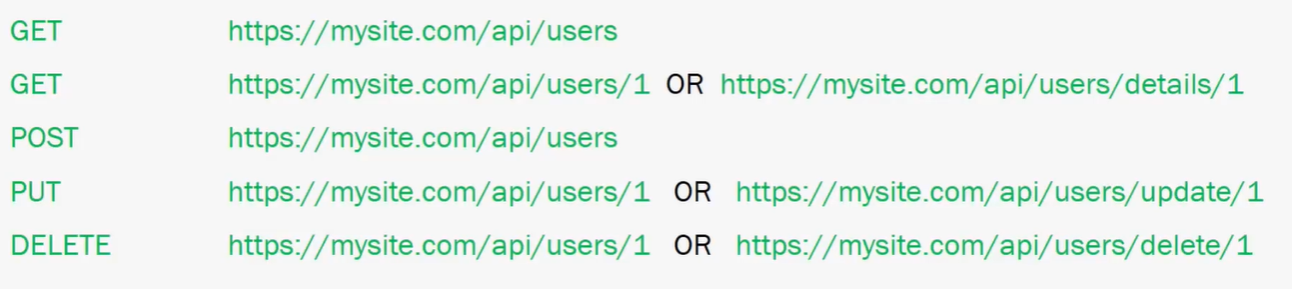
* 1. **Delete:** for **removing** an existing resource on the server

* 1. **Head:** similar to get but does not return body in response and it return head info.

* 1. **OPTIONS:** to retrieve the supported method of the server in order to know which HTTP operations we can perform in our requests.

**Endpoints:** The address of the server in (URI/URL) format which we can use that for accessing server/API for sending our request in our client application. In RESTful APIs the address can start either by HTTP or HTTPS for transferring data on a secure channel.

In the endpoint address, after the protocol, the domain of the company comes and it is a convention to use API and at the end resources would be come.



**Authentication:** API use authentication in order to let clients to use their method as a means of security to not allow unauthorized user to access or perform operations and also as means of being able to sell the specific features to the user groups like number of calls & etc.

Good Tutorial for creating an API design:

1. <https://www.youtube.com/watch?v=HeXQ98sogs8&list=PLWPirh4EWFpGRdVZcQCzeTXFBNSTDAdQX&t=28s>
2. <https://www.youtube.com/watch?v=TERMdBN4FxA&list=PLsyeobzWxl7ps4Z1C4VMtvZEx5-PgyoYI&t=1s>
3. <https://www.youtube.com/watch?v=GZvSYJDk-us&t=3s>

|  |  |
| --- | --- |
| Video Name | Video |
| [What are RESTful Services (RESTful APIs)? | Mosh](https://www.youtube.com/watch?v=SLwpqD8n3d0&t=58s)  Programming with Mosh | Video web content titled: What are RESTful Services (RESTful APIs)? | Mosh |
| [What Is A RESTful API? Explanation of REST & HTTP](https://www.youtube.com/watch?v=Q-BpqyOT3a8)  Traversy Media |  |
| [REST API concepts and examples](https://www.youtube.com/watch?v=7YcW25PHnAA&t=0s)  WebConcepts | Video web content titled: REST API concepts and examples |
| [What is REST API? | Web Service](https://www.youtube.com/watch?v=qVTAB8Z2VmA&t=0s)  Telusko | Video web content titled: What is REST API? | Web Service |
| [REST API & RESTful Web Services Explained | Web Services Tutorial](https://www.youtube.com/watch?v=LooL6_chvN4&t=0s)  Clever Techie | Video web content titled: REST API & RESTful Web Services Explained | Web Services Tutorial |
| [APIs | REST | REST APIs Demystified](https://www.youtube.com/watch?v=FOZtRzY5x8E&t=236s)  The TechCave | Video web content titled: APIs | REST | REST APIs Demystified |
| [What is REST API? | REST API Tutorial | REST API Concepts and Examples | Edureka](https://www.youtube.com/watch?v=rtWH70_MMHM&t=751s)  edureka! | Video web content titled: What is REST API? | REST API Tutorial | REST API  Concepts and Examples | Edureka |
| [What is an API?](https://www.youtube.com/watch?v=s7wmiS2mSXY&t=0s)  MuleSoft Videos | Video web content titled: What is an API? |