Rest(Representational State Transfer) Architecture Style

Software Design Architecture Lab # 14

Muhammad Rehan Baig

REST Architecture

- REST stands for Representational State Transfer and API stands for Application Program Interface. REST is a software architectural style that defines the set of rules to be used for creating web services. Web services which follow the REST architectural style are known as RESTful web services. It allows requesting systems to access and manipulate web resources by using a uniform and predefined set of rules. Interaction in REST based systems happen through Internet’s Hypertext Transfer Protocol (HTTP).
- A Restful system consists of a:
- client who requests for the resources.
- server who has the resources.
- It is important to create REST API according to industry standards which results in ease of development and increase client adoption.

REST Architecture

- REST stands for Representational State Transfer and API stands for Application Program Interface. REST is a software architectural style that defines the set of rules to be used for creating web services. Web services which follow the REST architectural style are known as RESTful web services. It allows requesting systems to access and manipulate web resources by using a uniform and predefined set of rules. Interaction in REST based systems happen through Internet’s Hypertext Transfer Protocol (HTTP).
- A Restful system consists of a:
- client who requests for the resources.
- server who has the resources.

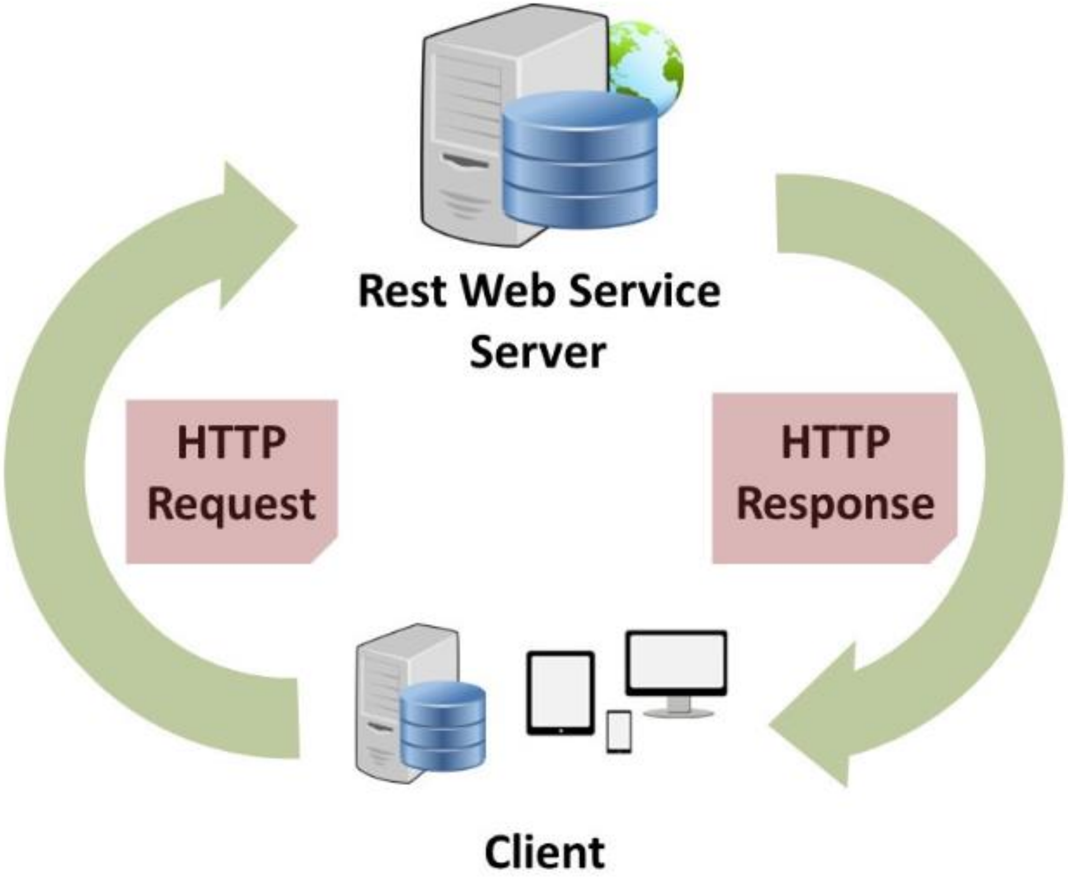
REST Architecture

- It is important to create REST API according to industry standards which results in ease of development and increase client adoption.
- The only optional constraint of REST architecture is code on demand. If a service violates any other constraint, it cannot strictly be referred to as RESTful.
- It relies on a stateless, client-server, cacheable communications protocol — and in virtually all cases, the HTTP protocol is used.
- What is HTTP?
  HTTP means ***H**yper**T**ext **T**ransfer **P**rotocol.* HTTP is the underlying protocol used by the World Wide Web and this protocol defines how messages are formatted and transmitted, and what actions Web servers and browsers should take in response to various commands

REST Architecture

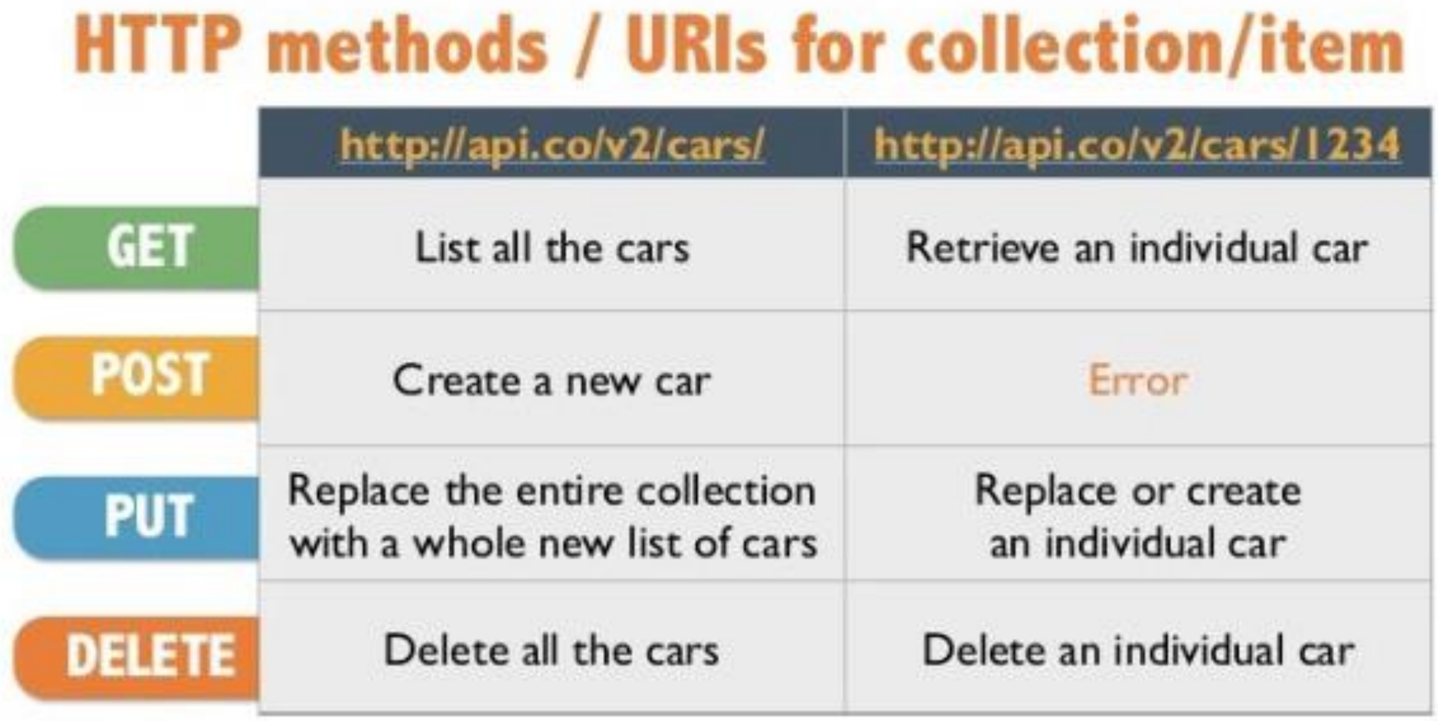
- The idea is that, rather than using complex mechanisms such as CORBA, RPC or SOAP to connect between machines, simple HTTP is used to make calls between machines.
- For Example:

REST Architecture



REST Architecture

- RESTful applications use HTTP requests to post data (create and/or update), read data (e.g., make queries), and delete data. Thus, REST uses HTTP for all four CRUD (Create/Read/Update/Delete) operations.



HTTP methods / URIs for collection/item

	http://api.co/v2/cars/	http://api.co/v2/cars/1234
GET	List all the cars	Retrieve an individual car
POST	Create a new car	Error
PUT	Replace the entire collection with a whole new list of cars	Replace or create an individual car
DELETE	Delete all the cars	Delete an individual car

REST Architecture - Benefits

Here are 5 benefits of using REST architecture:

- **RESTful as lightweight Web Services:** The RESTful architecture was a reaction to the more heavy-weight SOAP-based standards. In REST web services, the emphasis is on simple point-to-point communication over HTTP using plain XML. In addition, RESTful permits many different data formats whereas SOAP only permits XML.
- **The simplicity of RESTful:** The RESTful architecture is much simpler to develop than SOAP. One of the main reasons for REST popularity is the simplicity and ease of use, as it does an extension of native Web technologies such as HTTP.

REST Architecture - Benefits

- **RESTful architecture is closer in design to the Web:** RESTful is the architectural style of the web itself, so the developer with knowledge in web architecture will naturally develop in the RESTful architecture.
- **Scalability:** As RESTful forbids conversational state, which means we can scale very wide by adding additional server nodes behind a load balancer.
- **Expose APIs as HTTP Services:** When developers need the universal presence with minimum efforts, given the fact that RESTful APIs are exposed as HTTP Services, which is virtually present on almost all the platforms.

REST Architecture

- Here are 6 REST constraints that we can consider as design rules, that must be applied to establish the distinct characteristics of the REST architectural style. In other words, These 6 architectural constraints which make any web service — a true RESTful API:

1. Interface / Uniform Contract: *Once a developer becomes familiar with one of your API, he should be able to follow the similar approach for other APIs.*
2. Client-Server: *Servers and clients may also be replaced and developed independently, as long as the interface between them is not altered.*
3. Stateless: *No client context shall be stored on the server between requests. The client is responsible for managing the state of the application.*
4. Cache: *Well-managed caching partially or completely eliminates some client-server interactions, further improving scalability and performance.*
5. Layered System
6. Code-On-Demand (optional)

Example Step by Steps

Tasks

- **Create** webapi for restaurant named Alibaba the restaurant can be able to
  Post new item in menu,
  Get All items that restaurant offers,
  Put existing items,
  Delete items
- **Create** webapi that uses all verbs of Rest Api's for employee management **Create** new employee, **Update** Current employee, **Get** Employees, **Delete** employees with their respective Id's ->
  (Optional) For best professional practices use layered architecture for getting data from data access layer and applies logic with business logic layer then controller will get data and http will consume it.