



REST SERVICES METHODS/APIS USING HTTP

Bhupendra Pratap Singh

CDAC, ACTS, Pune


INTRODUCTION

- REST stands for Representational State Transfer
- HTTP – Hyper Text Transfer Protocol
- RESOURCES – GET, PUT, POST and DELETE
- It is not a protocol, not a standard it is just a way of doing things
- Way of doing things - Communication

CONT.

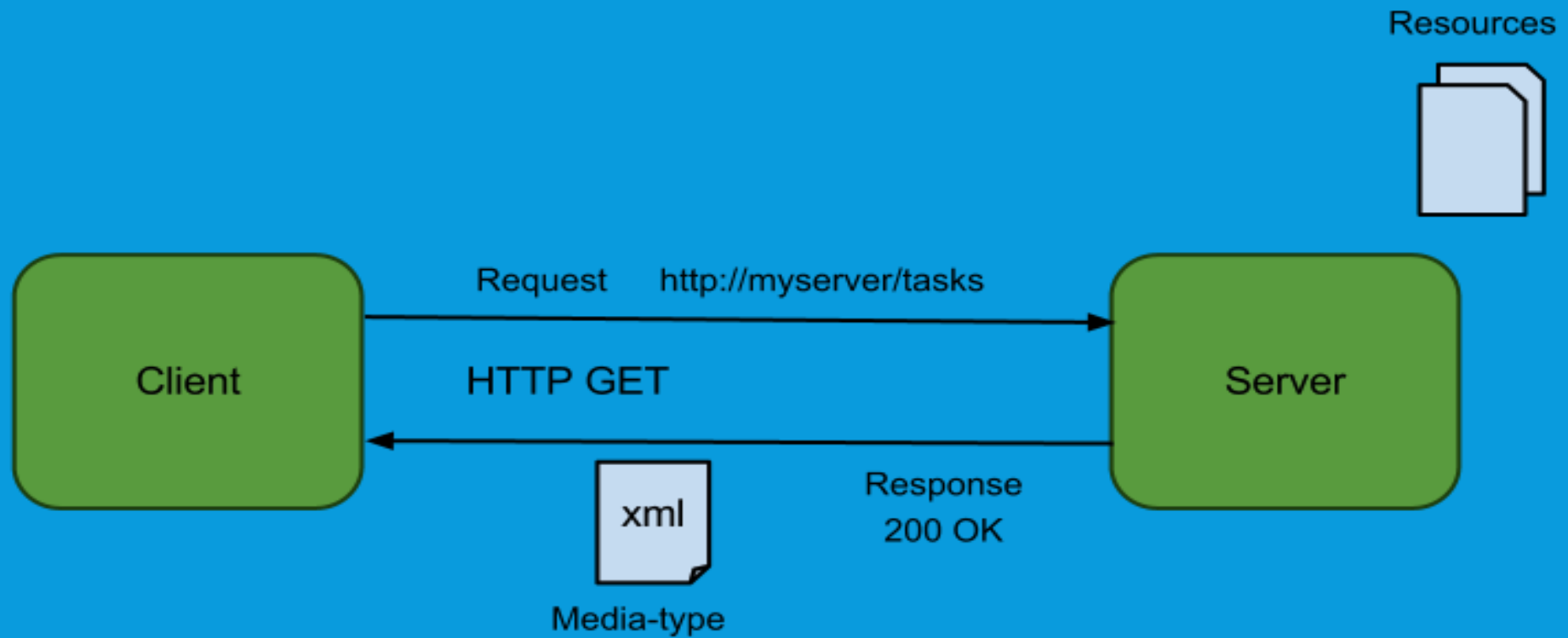
- Data Exchange formats – JSON, XML, pdf, Ppt etc.
- Mostly in Practice REST APIs using HTTP (1.1)
- Request and Response model
- Request is made from client to Server.
- Response is made from server to client.

Through
URLs



How to make requests and
How to understand
response?

CLIENT – SERVER PARADIGM



URLS – UNIFORM RESOURCE LOCATOR

- What is URL?
- URL is an acronym for *Uniform Resource Locator* and is a reference (an address) to a resource on the Internet.
- Identify the hostname, port name, resource
- Examples-
- `http://www.example.com:8080/v1/api/humidity?result=10`
- `http`: Protocol, `8080` : Port, `result = 10` : Querystring
- `www.example.com` – hostname (can be IP address (numeric))
- `V1/api/humidity` : Resource (path)

SOME ADVANTAGES

- Stateless
- Simple CRUD (Create, Read, Update, Delete) mappings
- Caching, authentication, authorization
- Web oriented
- Idempotent and safe methods

WHAT IS STATELESSNESS?

- Statelessness means communication must be **stateless** in nature as in the client stateless server style, i.e. **Each request from client to server must contain all of the information necessary to understand the request**, and cannot take advantage of any stored context on the server. Session state is therefore kept entirely on the client.

WHAT IS CATCHABLE/CACHE?

- In order to improve network efficiency, cache constraints are added to the REST style.
- Cache constraints require that the data within a response to a request be implicitly or explicitly labeled as cacheable or non-cacheable. If a response is cacheable, then a client cache is given the right to reuse that response data for later, equivalent requests.

SOME LIGHT ON PROTOCOL (HTTP)

- Works on TCP transport layer -- secure (SSL/TLS)
- HTTP 0.9, initial release in 1991
- HTTP 1.0 in 1996
- HTTP 1.1, IETF standards
- RFC 7230, 7231, 7232, 7233, 7234, 7235, 7236
- HTTP 2.0 in 2015, described by RFC 7540

STATUS CODES

- **2.x.x – denotes success**

- 200 Ok
- 201 Created
- 202 Accepted

- **4.x.x – Error due to client requests**

- 400 Bad Request
- 401 Unauthorized Error
- 402 Payment not done
- 403 forbidden
- 404 not found (resource not found/ path incorrect)
- 405 Not allowed

TODO: Refer <https://www.w3.org/Protocols/HTTP/HTRESP.html>

CONT.

- 5.x.x. – Server Internal Error
- 3.x.x – Redirection status/errors
- 1.x.x – information to connection upgrades

MIME TYPES & CONTENT TYPES

- Content Types:- (www.iana.org/assignments/media-types)
- MIME types :- Multipurpose Internet Mail extensions
- MIME a way of identifying files on the Internet according to their nature and format
- eg:- text/plain, text/html, text/xml
- application/json, application/pdf
- image/jpg, audio/mp3, video/h264
- Content-Type is request, response headers
- Accept element part of request

REQUESTS METHODS

- GET -- retrieve the resource state
- POST -- create a new resource(or) sub resource
- PUT -- update state of existing resource
- DELETE -- delete the resource (or) state

SAFE AND IDEMPOTENT METHODS

- safe methods -- which doesn't change resource state, eg:- GET
- idempotent -- multiple requests should have same effect, eg:- PUT
- (with same parameters)
- eg:- PUT method for updating channel settings
- POST -- neither safe nor idempotent

CONT..

- GET request -- body is empty, query string applicable for filters (horizontal, vertical), sorting requirements, response carries body with requested information
- POST/PUT request typically carries body/payload, response may or may not be empty
- Note : Horizontal Scanning/Filtering – Group of IP and Single Port and Vertical Scanning – Group of Port and Single IP

REQUEST HEADER ELEMENTS

- Content-Type
- Accept
- Accept-Language
- Accept-Encoding
- User-Agent
- Connection
- Authorization
- Request may contain payload(eg:- POST/PUT)

```
1 GET /home.html HTTP/1.1
2 Host: developer.mozilla.org
3 User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.9; rv:50.0) Gecko/20100101 Firefox/50.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Referer: https://developer.mozilla.org/testpage.html
8 Connection: keep-alive
9 Upgrade-Insecure-Requests: 1
10 If-Modified-Since: Mon, 18 Jul 2016 02:36:04 GMT
11 If-None-Match: "c561c68d0ba92bbeb8b0fff2a9199f722e3a621a"
12 Cache-Control: max-age=0
```


RESPONSE HEADER ELEMENTS

- status code
- server
- date
- Connection
- max age (
- Response may contain payload(body), eg:- GET

```
HTTP/1.1 200 OK
```

```
Connection: Keep-Alive
```

```
Content-Encoding: gzip
```

```
Content-Type: text/html; charset=utf-8
```

```
Date: Thu, 11 Aug 2016 15:23:13 GMT
```

```
Keep-Alive: timeout=5, max=1000
```

```
Last-Modified: Mon, 25 Jul 2016 04:32:39 GMT
```

```
Server: Apache
```

WHATN DOES KEEP-ALIVE

- timeout: indicating the minimum amount of time an idle connection has to be kept opened (in seconds). Note that timeouts longer than the TCP timeout may be ignored if no keep-alive TCP message is set at the transport level.
- max: indicating the maximum number of requests that can be sent on this connection before closing it. Unless 0, this value is ignored for non-pipelined connections as another request will be sent in the next response. An HTTP pipeline can use it to limit the pipelining.

CURL

- curl is used in command lines or scripts to transfer data. It is also used in cars, television sets, routers, printers, audio equipment, mobile phones, tablets, settop boxes, media players and is the internet transfer backbone for thousands of software applications affecting *billions of humans* daily.

REFERENCES

- <https://reqres.in/>
- <http://restcookbook.com/>



THANK YOU !!