REST API

* Core Java
  + Collections ( Collection API )
  + Java 1.8 Features ( Optional )
* Maven – Build Automation Tool
* TestNG - Framework
* Rest Assured – Java library, used to automate RESTFul Web Services
* Live projects : GitHub, Trello, google map, JIRA, Twitter
* POST MAN
* Rest Assured Framework

## Road-Map :

Learning Collection API 🡪 Basics of REST API 🡪

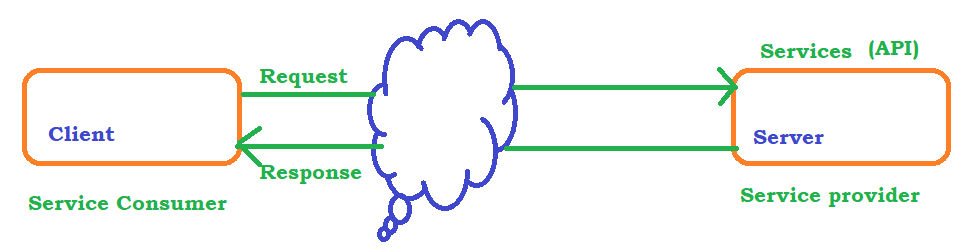
Architecture of REST 🡪 Executing REST API using POST MAN 🡪

Create project 🡪 REST Assured to automate REST APIs 🡪

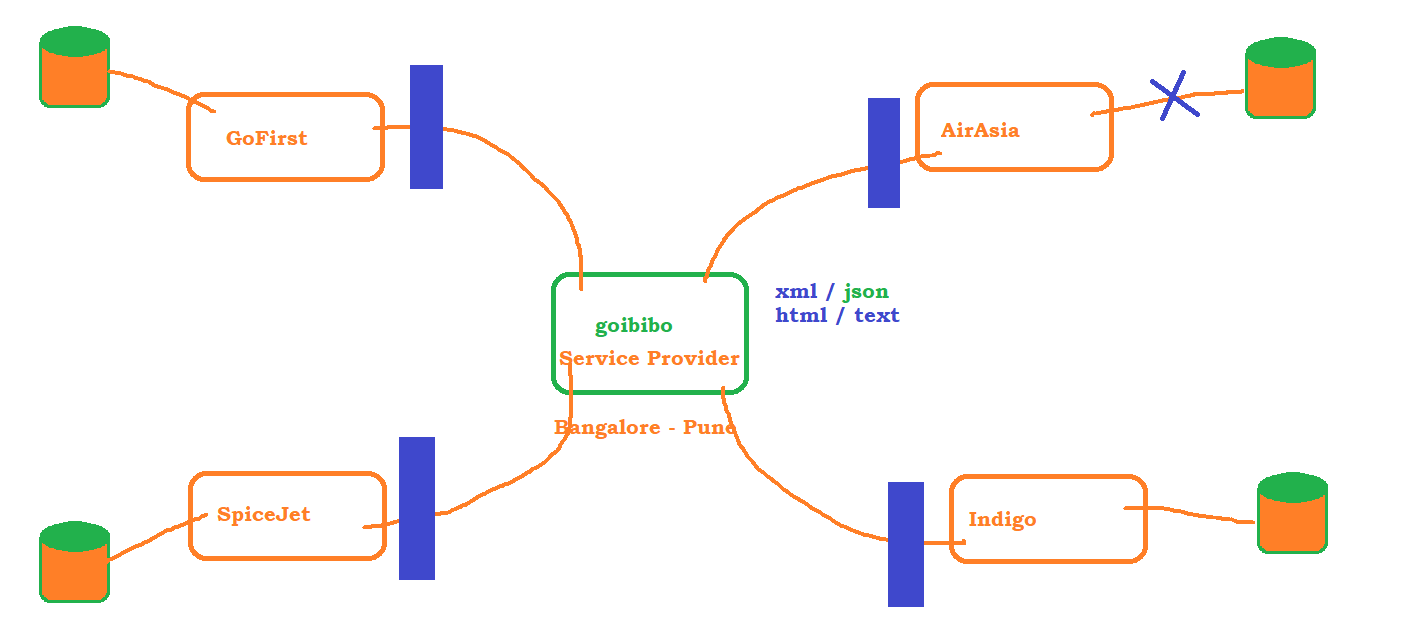
Automate 🡪 TestNG (run in batch) 🡪 Reports 🡪 CI/CD 🡪 APIs

using BDD.

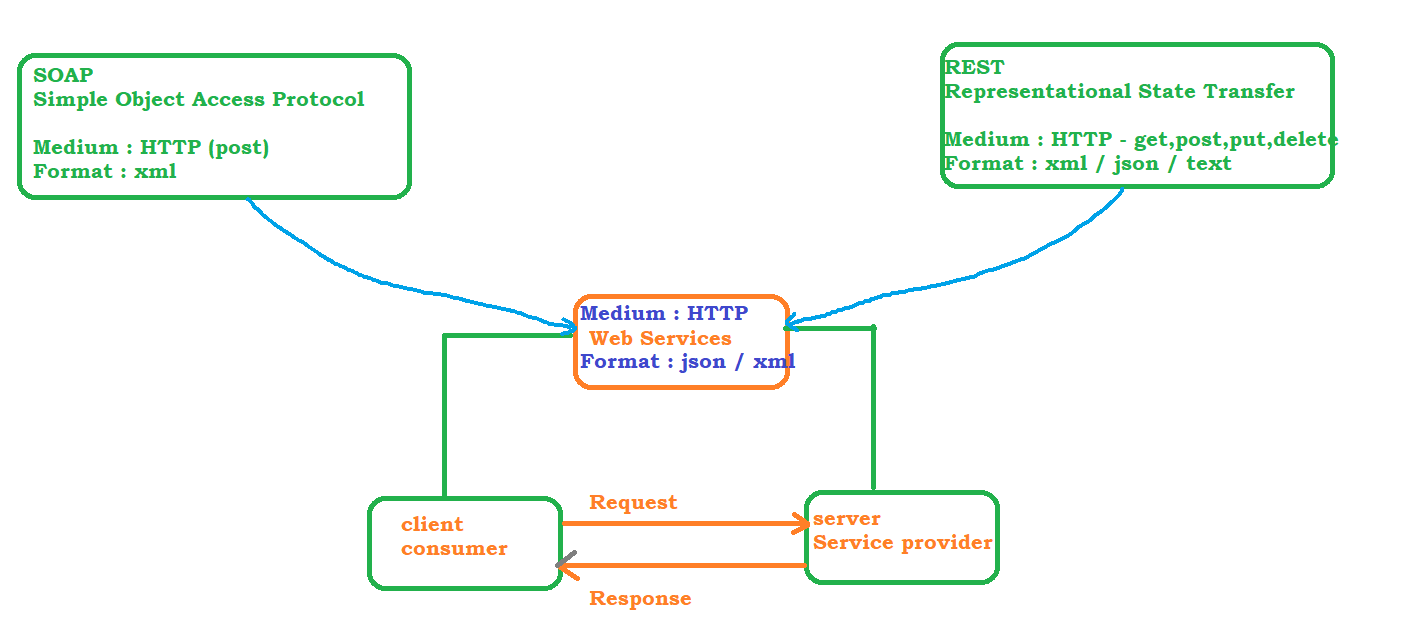
### What is API :



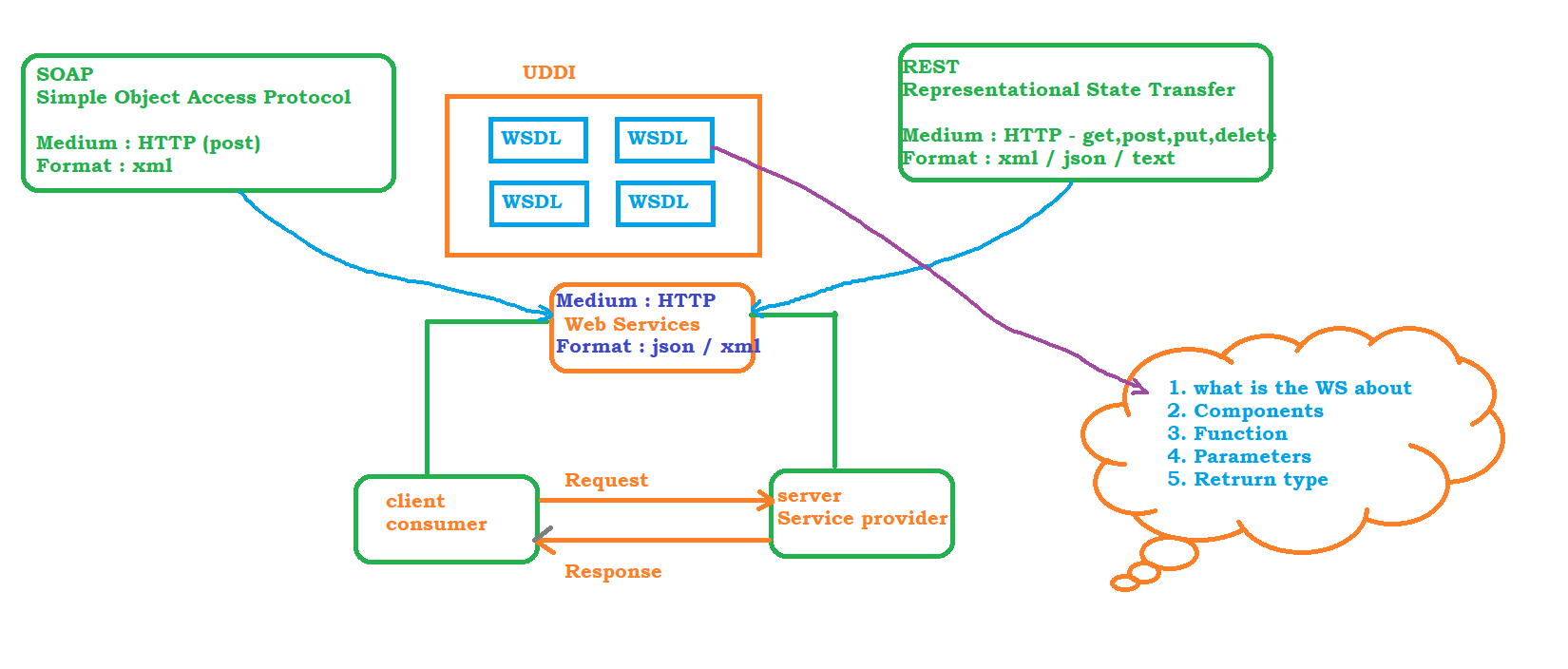
### Web Services Introduction



### High level Architecture



Example of SOAP : <https://documenter.getpostman.com/view/8854915/Szf26WHn#96a53688-6305-45be-ab8b-ca1d1c88f830>



REST :

Webservice that communicates or exchange information between 2 applications using REST Architecture / REST principles are called RESTFull WS

* REST is nothing but REpresentational State Transfer and its a Architectural Style
* REST is not a protocol , ie there is no Standard Rules or Central body to control the rules
* REST is just a design Principle

1. Uniform Interface

* Resource
* URI
* HTTP – get, post, put, delete, patch etc..

2. Stateless

* Every Request which is sent from client to server must contain all the required information OR data. Server will not store any prior request information

3. Cachable

Happens at client side, along with the response server will send meta data information which tells the client weather to store the response or not

4. Layered System

* proxy
* gateway
* caching

5. Code on demand

Its Optional, and its an ability to download and execute the code at the client side

PostMan

<https://www.postman.com/downloads/>

1. WorkSpace – Where all the projects are stored
2. Collection – represent the project, each collection holds one project and each project hold multiple APIs
3. HTTP Request –
   1. Request Line : GET, POST, PUT
   2. Request Header : Optional
   3. Request Body : Optional
4. HTTP Response –
   1. Request Line : GET, POST, PUT
   2. Response Header :
   3. Response Body :
   4. Status Code : Validation purpose
5. Global Variables
6. Environment Variables
7. Pre-Request Script
8. Write Test in POST-MAN

### REST API Elements :

* Resource :
* Resource Identifier :
* Representation :
* Representation Metadata :
* URI

SCHEME://AUTHORITY / PATH OR QUERY / [#]

SCHEME – HTTP, HTTPS, FTP, HTTP2

AUTHORITY – Domain

PATH Parameter – access collection of resource or single resource

QUERY Parameter – sort or filter the resources

* HTTP Methods
  1. GET : Retrieve the data from the server, 404 if no resource found
  2. POST : to insert the data or create new resource, body is needed to send the data to be created
  3. DELETE – Delete the record
  4. PUT – update the record, single record or resource
  5. PATCH - update the record, with partial information

Trello API

Trello is a Project management application which can be used to create Dashboard to monitor many project activities

* Go to <https://trello.com/> and register yourself and login to the application
* Go to developer API and understand the API calls <https://developer.atlassian.com/cloud/trello/rest/api-group-actions/#api-group-actions>
* To Generate key and token <https://developer.atlassian.com/cloud/trello/guides/rest-api/api-introduction/>
* GET
* POST
* PUT
* DELETE

GIT-HUB

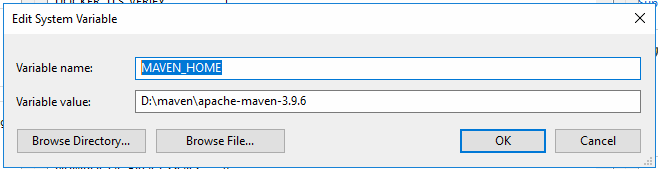
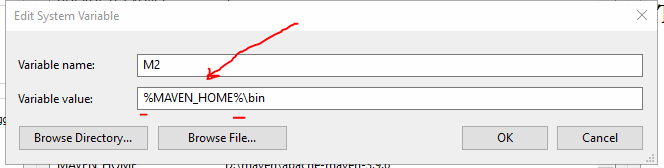
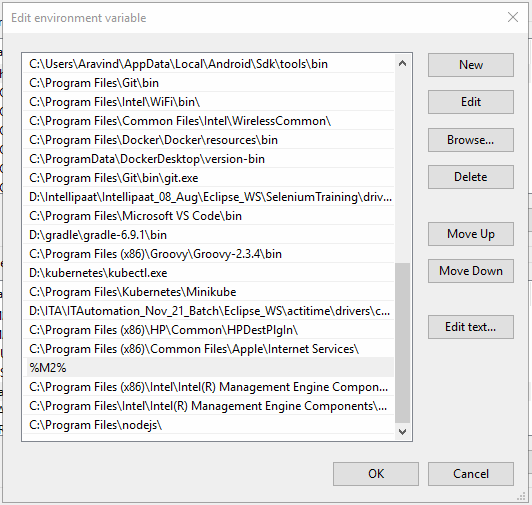
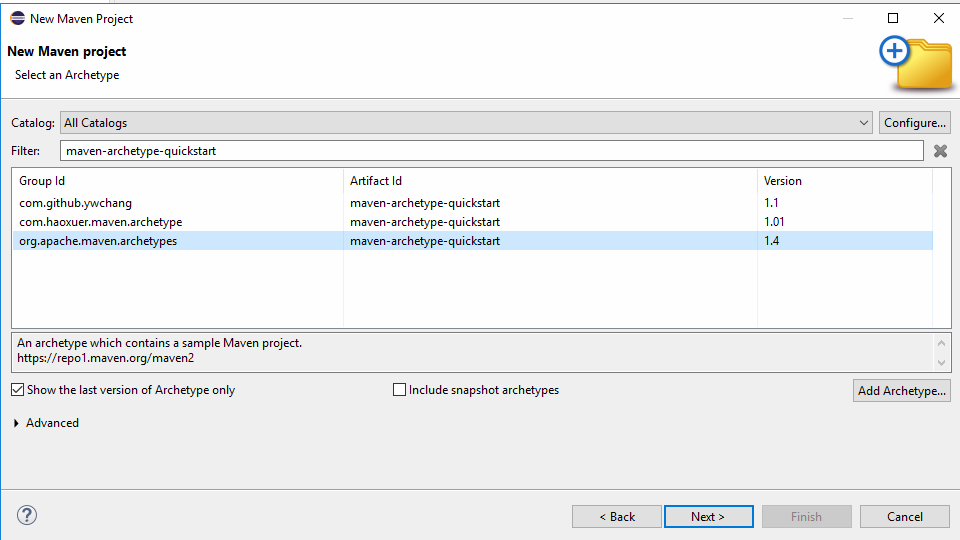
* Create a Git Hub Account
* open google page and search for github developer api
* <https://docs.github.com/en/rest>
* Get the Bearer Token from github.com
  + click on the user icon on the top right
  + go to settings
  + go to developer settings
  + click on personal access token
  + click on Generate New token
  + Make sure to decide selecting admin selection (select only if it is needed )
  + Once the token is generated make sure to copy and keep it in safe place

Automation with REST Assured

Rest Assured is a java library built on topof HTTP Builder, All HTTP Methods are available inside Rest Assured, (GET, POST, PUT, DELETE)

Rest Assured library we can easily automate REST APIs

STEPS :

* Java 1.8
* Eclipse - [eclipse for java developers](https://www.eclipse.org/downloads/packages/)
* Maven – [download and install maven](https://maven.apache.org/download.cgi)
  + 
  + 
  + 
* Maven Project
  + 
  + Make sure to check java compiler and set it to the same version which is installed on your machine
  + Make sure to check java Runtime environment and use the JRE from jdk location
  + Update pom.xml with REST Assured dependencies <https://rest-assured.io/>
* Terminologies in REST Assured library
  + BASE URI – end point or the exact resource
  + given() – is the first method to be called in REST Assured, it defines the pre-condition to the REST Call
    - given holds many information to the REST CALL Like,
    - path parameter
    - query parameter
    - body parameter
    - request header information
    - cookies information

### when() – Action performed by the REST, it holds what type of request that we are making

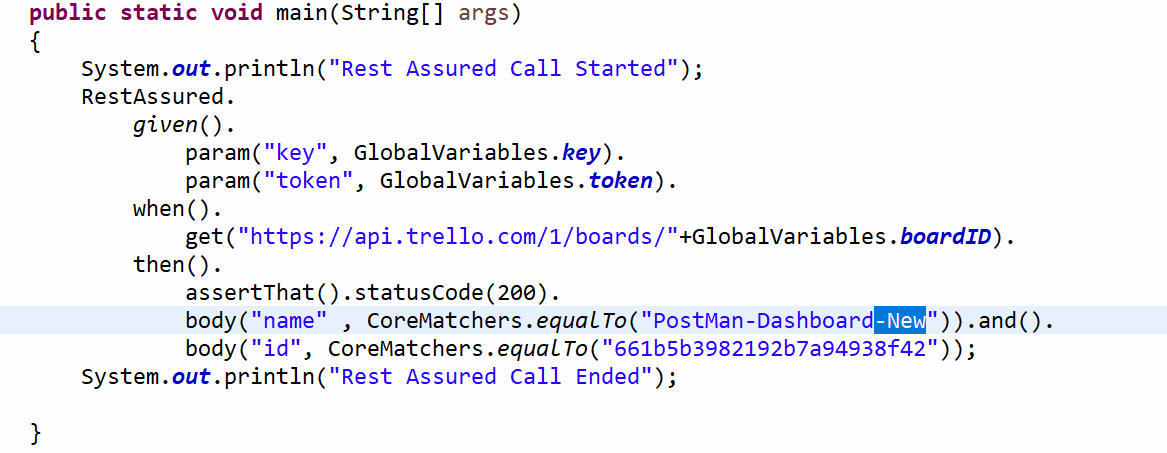
* + - get
    - post
    - put
    - patch
    - delete

### then() – Actual Result, it holds the validation part of REST

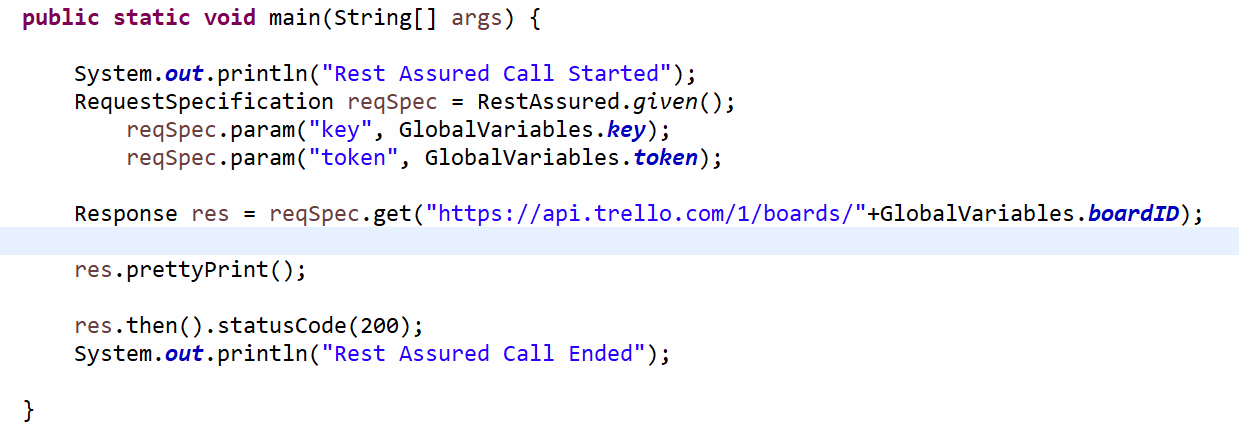
### extract() – whenever we want to extract the response and play with json response

REST Assured Test Format

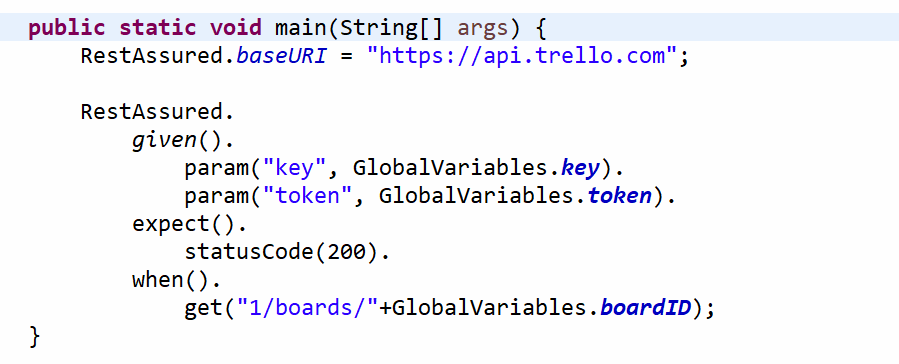
# Given When Then Format (BDD Approach)



# Rest Assured Class Format



# Given Expect When Format



#### Post Request in REST Assured

while sending data we can send in 3 ways

* path parameter
* query parameter
* body parameter -> can be passed as a string or as java object (payload)
  1. Java POJO Object contains only getters and setters method

#### Faker API :

* Search for faker api for java
* <https://github.com/DiUS/java-faker>
* Update your project dependency – pom.xml

#### Understanding Complex Json

json is a key/value pair file, which is used to represent the complex data/ Objects.

if you see any data with in [ ] then its a List in json

if you see any data with in { } then its a Map in json

we can have complex data inside

* List of List
* List of Map
* Map
* List
* Map of Map
* Map of List of Map ..
* List of Map of Map ..

There are lot of plug-in available which will convert the json to any format like

* HTML
* XML
* PDF etc

### Enabling Logs in REST

<https://github.com/rest-assured/rest-assured/wiki/Usage#logging>

Logs in REST enables to print many meta informations like,

* what is the parameter passed
* what is the request headers passed
* what are the cookies passed
* what is the response
* what is the response headers

# Request and Response Specification

These are the classes provided by REST to simplify the Request and Response code.

using Request and Response specification we can make sure that we are not having duplicate code in our test.

Steps :

1. Identify the common arguments OR parameters passed as a request parameter

2. Identify the common Validations OR Status code validated in each response

3. create a RequestSpecBuilder and ResponseSpecBuilder Object to perform the same

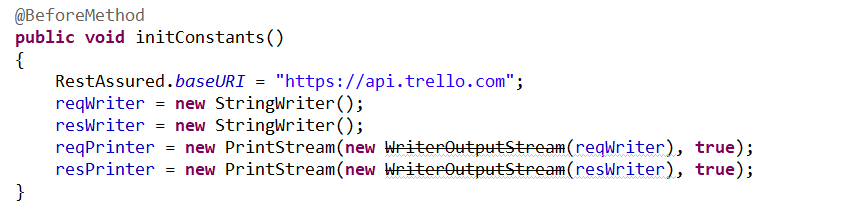


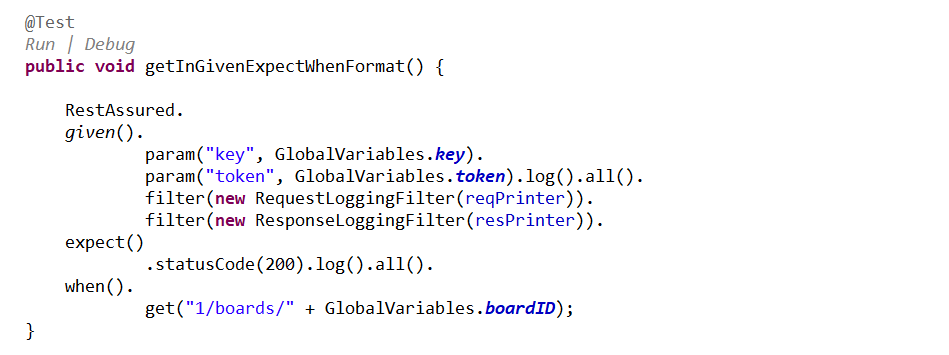
### Filters in REST API:

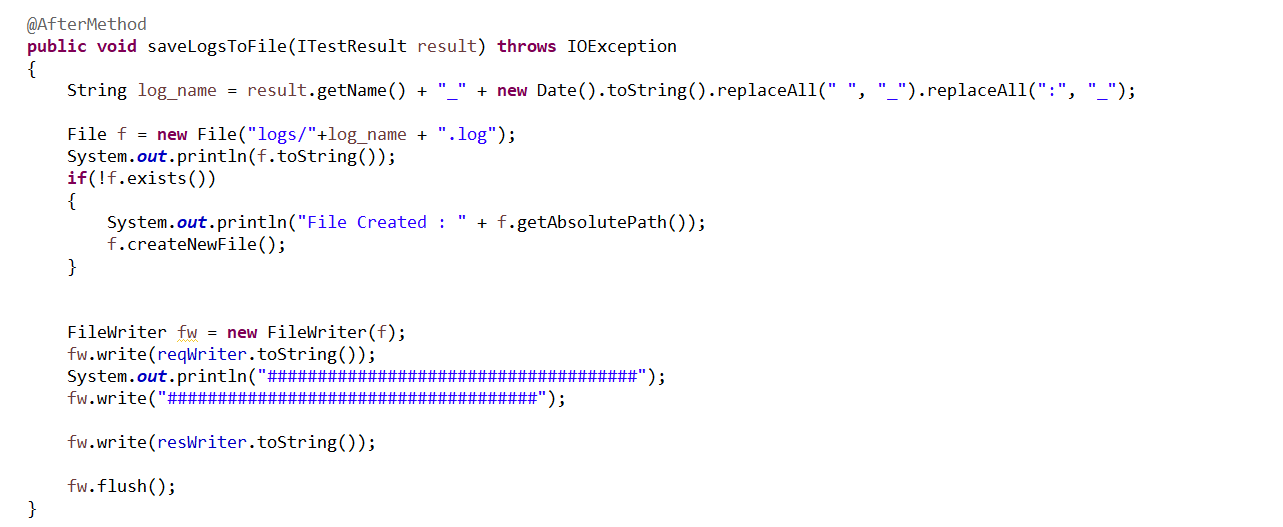
are mainly used to store

* request header
* response header
* cookies
* parameters (path params, query params, body params)
* response body etc

Download commons-io library : <https://mvnrepository.com/artifact/commons-io/commons-io/2.16.1>



****

****

Validations in REST Assured

* Jayway Jsonpath – Java
  + Update the pom.xml <https://github.com/json-path/JsonPath>
  + Steps to validate
    1. Extract the Response
    2. convert the response object to String
    3. call read method present inside Jayway Jsonpath library and pass the expression
* JsonPath (builtin) – Groovy Gpath
  + <https://docs.groovy-lang.org/next/html/documentation/#_gpath>
  + Expressions can be tested here <https://www.tutorialspoint.com/execute_groovy_online.php>
  + Steps to validate
    1. Extract the Response
    2. Create a object to JsonPath using the above response in string format
    3. call get method and pass the valid Groovy Gpath

NOTE : we can use any valid Groovy code, like max(),min() etc in the groovy expression

* ValidatableResponse - Grooy Gpath
  + Steps to validate
    1. Extract the ValidatableResponse
    2. call extract method
    3. call path method and pass the valid Groovy Gpath

# Static imports :

Using static import we can import all the static methods of a class so that we no need to specify the classname while calling a function

# Root Path :

to simplify the search of an element whenever we are in deep search for json we can set the root path to make simplify the search

# CoreMatches : <https://hamcrest.org/JavaHamcrest/javadoc/1.3/org/hamcrest/CoreMatchers.html>

* + equalTo
  + hasItem
  + hasItems

# ~~BDD-Cucumber~~

# Jenkins