**Where we required API in the architecture?**

Application programming interface:

Backend createReservation()(Java technology)

Front End (Java technology)

Database Store data

**Example: Hotel**

Collect the information of the hotel from the specific hotel

1. Same technology backend and frontend

Backend Mariotte hotel javajcreateReservation()

Front End – Hotels.com java

Database Store data

How to communicate?

1. Different technology backend and front end

Backend Mariotte hotel javajcreateReservation()

Front End – Hotels.com Angular

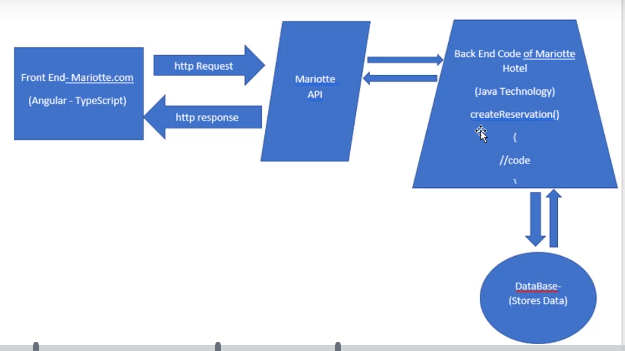
Database Store data

How to communicate?

So the problem is that Hotel doesn’t want to share their code. Second the different front end and backend technology.

**Solution: API**

API acts as interface between you client- server. It is also called middle tier.



API is independent of any language. Front end sends the data using Http protocol which is independent of any language. By using this info createReservation() to create reservation. Http request send using Json or xml which are also independent to any language.

API shares only data not code to any other guys.

API is communication-protocol/interface between Client and Server intended to simplify the building of Client-side software. It also implements abstraction so that not to expose the data to another client only expose the shareable field

Soap is tight weight and rest assured light weight.

Legacy is still using

Services or API anonyms, very short differences.

**Need of API**

**What is the need of API?**

Well, in addition to all the answers, I am just adding an example.

As others said API stands for Application Programming Interface through which softwares can interact with each other. Note, not a human interaction.

***Where it is used***

An example: You are buying an item online through your credit card. You will provide credit card details and press 'continue' button. It will tell you whether your information is correct or not. To provide these results, there are lot of things in the background.

The application will send your credit card details to a remote application which will validate your information and send the result back to your application. API is used in this scenario.

ANOTHER EXAMPLE

Weather application

Without API - Weather application must open weather.com site and read the details as a human does.

With API - Weather application will send a message to weather.com and receive the result and then display it.

Some IMP terms:

**End point/Base URL:** Address where API hosted on the server.

**HTTP methods** which are used commonly.

**CRUD**: C: Create- Post, R: Retrieve- Get, U: Update- Put, D: Delete- delete

**Get**: used to retrieve some information from the server. For example: current reservation status

* Only extract data,
* no effect on existing data,
* no payload and body required

**Post:** A post request is used to send data to the Server. Example: customer details, file information, photo

**Put**: Replace or update the existing data on server. Example: Updating reservation by modify mobile number or details

**Delete**: Delete some information from the server, Example: Cancel reservation, delete profile

**Resource**: Resource represent API/collection which can be accessed from server like google.com/maps here map is the resource and google.com is the base URL.

**Path parameters**: this is the variable part of a URL path. This denotes to point a specific resource with in a collection or resource. Like User can specified by ID.

Amazon.com/orders/112222 – means you want order of 112222

**Query parameters:** used to sort/filter resource. Identified using ?. If multiple parameters are there we used the &

Amazon.com/orders?sort\_by=11/02/2020

**Difference between path and query parameters: Simple in query you have? While in path we use /**

**Structure of API**

Base URL/resource/(query/path parameters)

Base URL/resource- mandatory

/(query/path parameters)- optional

**Header and cookies:** Header represent the the meta data associated with the API request and response. Or we can say that additional details to API to process our request.

**Postman basics:**

Google Maps Add API (POST):

This API Will add new place into Server

**Complete URL** : [https://rahulshettyacademy.com](https://rahulshettyacademy.com/)/maps/api/place/add/json?key= qaclick123

**Base URL:** [https://rahulshettyacademy.com](https://rahulshettyacademy.com/)

**Resource:** /maps/api/place/add/json

**Query Parameters:** key =qaclick123

**Http Method:** POST

**Sample Body :**

{

"location": {

"lat": -38.383494,

"lng": 33.427362

},

"accuracy": 50,

"name": "Frontline house",

"phone\_number": "(+91) 983 893 3937",

"address": "29, side layout, cohen 09",

"types": [

"shoe park",

"shop"

],

"website": "http://google.com",

"language": "French-IN"

}

**Sample Response**

{

"status": "OK",

"place\_id": "928b51f64aed18713b0d164d9be8d67f",

"scope": "APP",

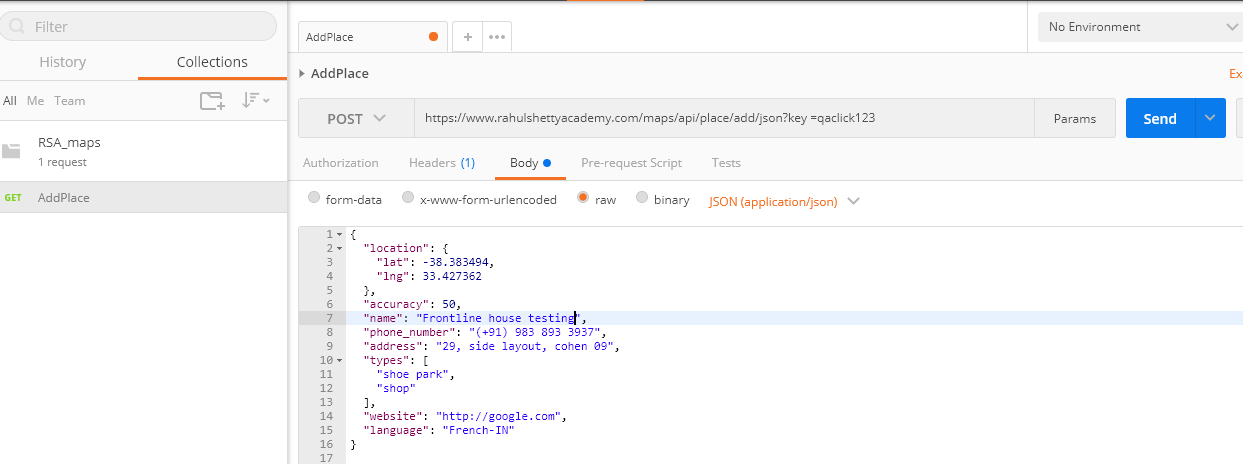
"reference": "736f3c9bec384af62a184a1936d42bb0736f3c9bec384af62a184a1936d42bb0",

"id": "736f3c9bec384af62a184a1936d42bb0"

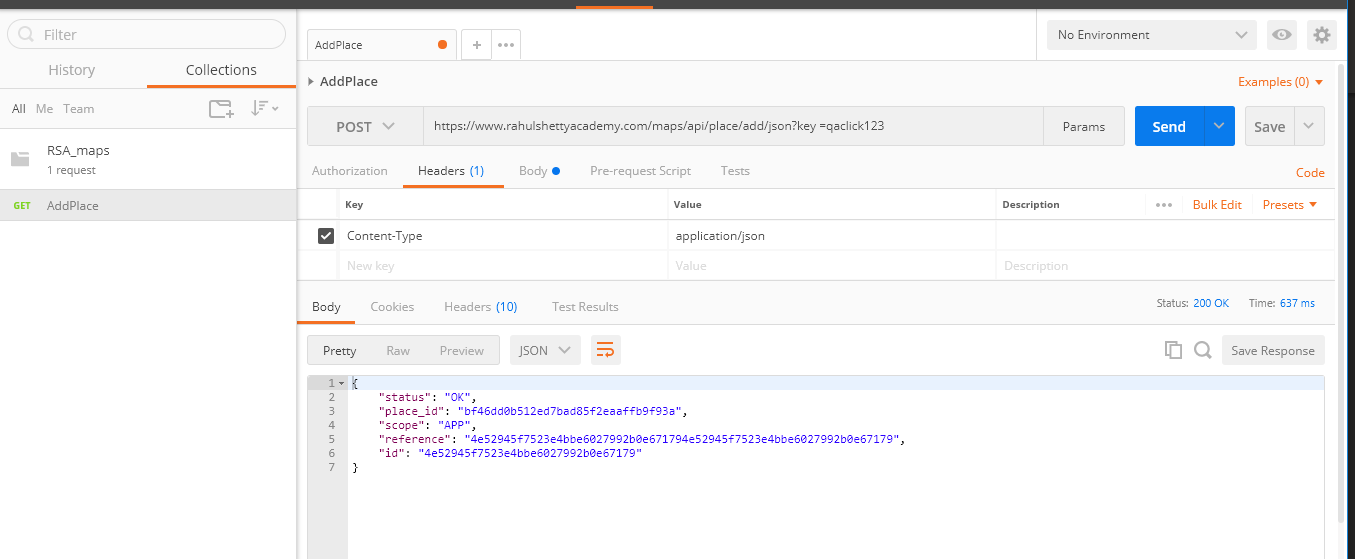
}

**How to use in the Postman:**

1. create the collection>>add request
2. put the Base URL/Resource/(query/path parameter)
3. Go to body >>select raw radio button >> text(json) – copy the body



1. Click on the send and check the response



Response :

{

"status": "OK",

"place\_id": "4aaab0bb8767def93fe01d102351951e",

"scope": "APP",

"reference": "f98eeae68d9174f350bac77530a468b6f98eeae68d9174f350bac77530a468b6",

"id": "f98eeae68d9174f350bac77530a468b6"

}

Google Maps get Place API (GET):

This API Will get existing place details from Server

Complete URL : http://rahulshettyacademy.com/maps/api/place/get/json?place\_id=xxxx&key=qaclick123

Base URL: [https://rahulshettyacademy.com](https://rahulshettyacademy.com/)

Resource: /maps/api/place/get/json

Query Parameters: key, place\_id //( place\_id value comes from Add place response)

Http request: GET

Note: Key value is hardcoded and it is always qaclick123

Sample Response for the Provided Place\_Id

{

"location":{

"lat" : -38.383494,

"lng" : 33.427362

},

"accuracy":50,

"name":"Frontline house",

"phone\_number":"(+91) 983 893 3937",

"address" : "29, side layout, cohen 09",

"types": ["shoe park","shop"],

"website" : "http://google.com",

"language" : "French-IN"

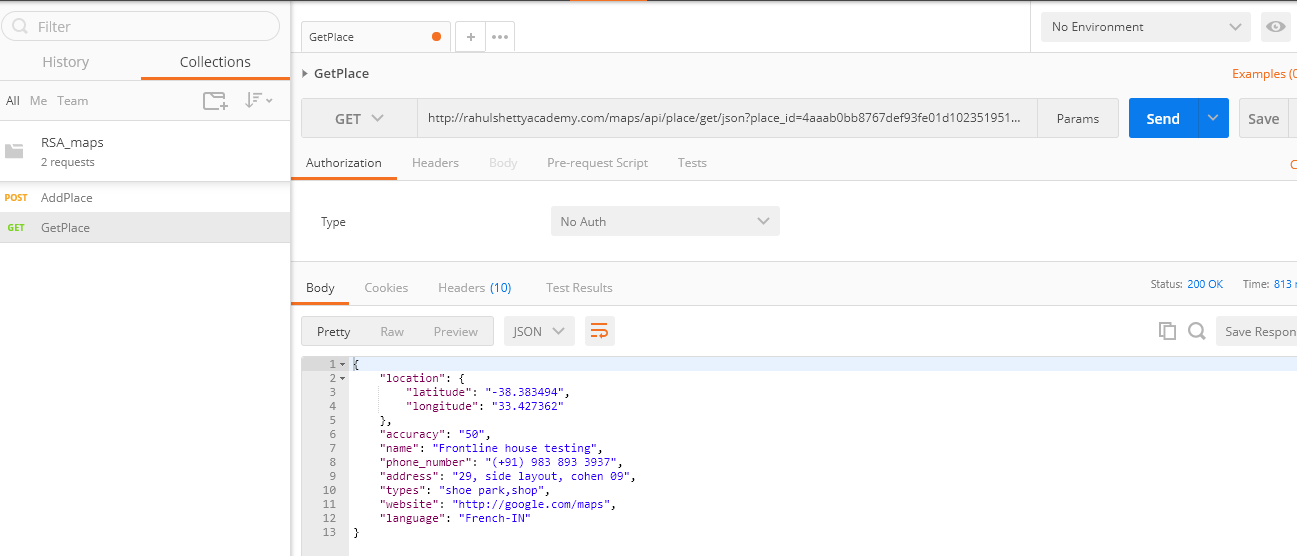
}

**How to use in the Postman:**

1. create the collection>>add request
2. put the Base URL/Resource/(query/path parameter)
3. click on send – get request url : <http://rahulshettyacademy.com/maps/api/place/get/json?place_id=4aaab0bb8767def93fe01d102351951e&key=qaclick123>

Place id comes from the first post request

Response:



Google Maps Delete API (POST):

This API Will delete existing place from Server

Complete URL: https://rahulshettyacademy.com/maps/api/place/delete/json?key=qaclick123

Base URL: https://rahulshettyacademy.com

Resource: /maps/api/place/delete/json

Query Parameters: key

Http request : DELETE

Sample Body :

{

"place\_id":"4aaab0bb8767def93fe01d102351951e"

}

Sample Response

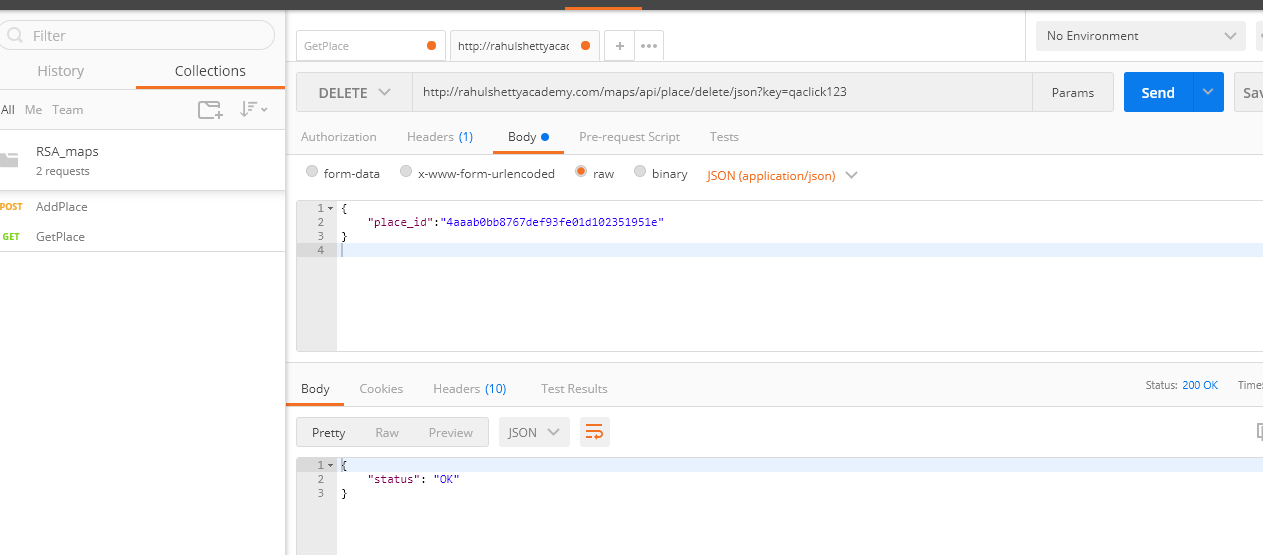
{

"status": "OK"

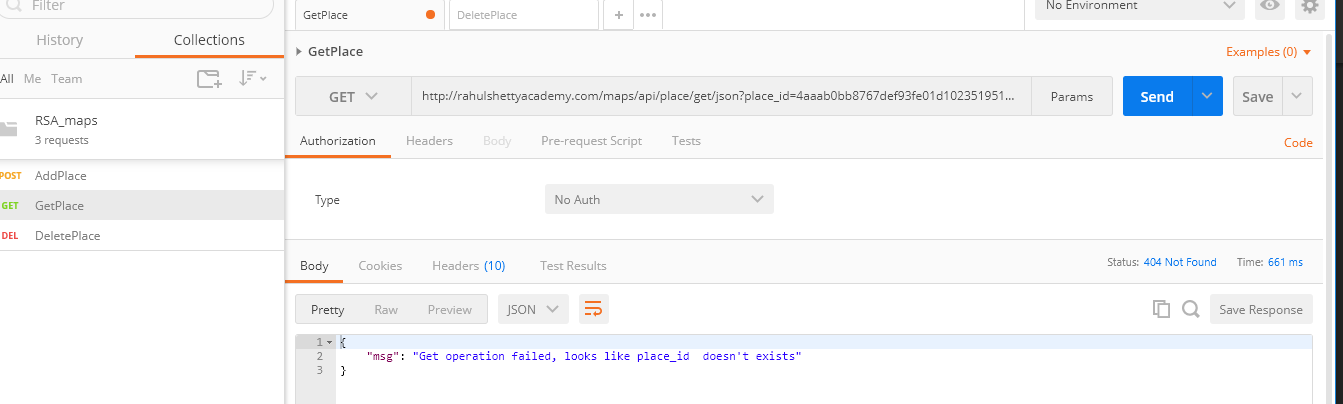
}

**How to use in the Postman:**

1. create the collection>>add request
2. put the Base URL/Resource/(query/path parameter)
3. Go to body >>select raw radio button >> text(json) – copy the body



Check it delete or not using Get



Google Maps Put Place API (PUT):

This API Will update existing place in Server with new values

Complete URL : http://rahulshettyacademy.com/maps/api/place/get/json?place\_id=xxxx&key=qaclick123

Base URL : [https://rahulshettyacademy.com](https://rahulshettyacademy.com/)

Resource: /maps/api/place/update/json

Query Parameters: key

Http Method: PUT -

Note: Key value is hardcoded and it is always qaclick123

Sample Request:

{

"place\_id":"8d2573bdf6ceec0e474c5f388fa917fb",

"address":"70 Summer walk, USA",

"key":"qaclick123"

}

Sample Response for the Provided Place\_Id

{

"location":{

"lat" : -38.383494,

"lng" : 33.427362

},

"accuracy":50,

"name":"Frontline house",

"phone\_number":"(+91) 983 893 3937",

"address" : "29, side layout, cohen 09",

"types": ["shoe park","shop"],

"website" : "http://google.com",

"language" : "French-IN"

}

**How to use in the Postman:**

1. create the collection>>add request
2. put the Base URL/Resource/(query/path parameter)
3. Go to body >>select raw radio button >> text(json) – copy the body

First create :

{

"status": "OK",

"place\_id": "7ea19025f7d60fd5435abc076f0183c8",

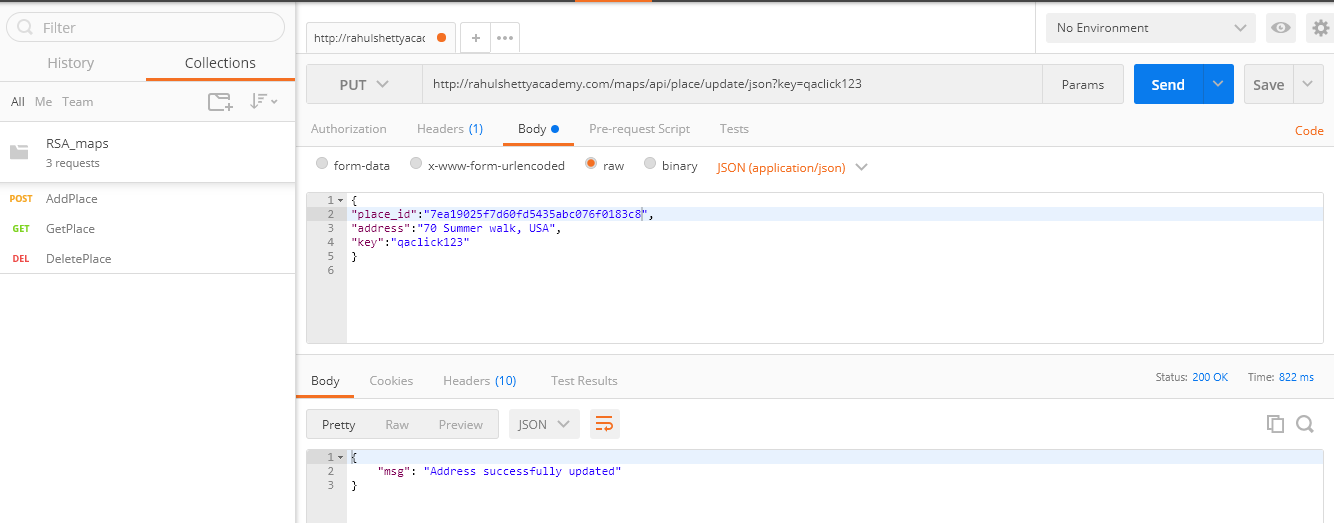
"scope": "APP",

"reference": "11dacf5781fd0afd2625144eff4382f611dacf5781fd0afd2625144eff4382f6",

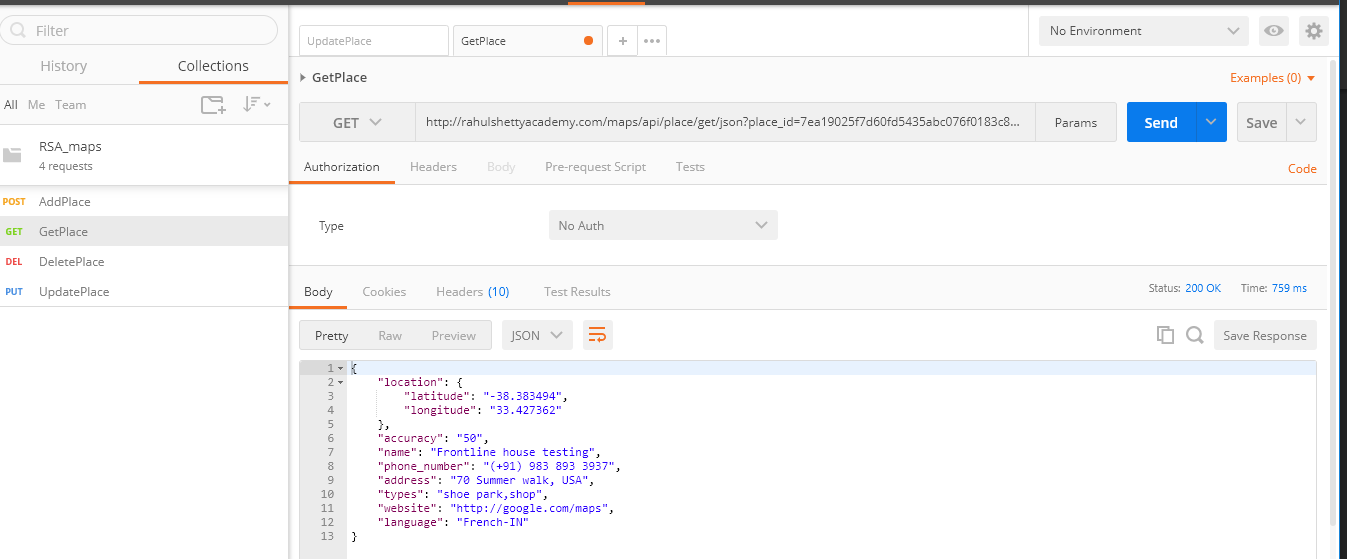
"id": "11dacf5781fd0afd2625144eff4382f6"

}

Then update using new body



Check using get api again



**Rest Assured:** Rest assured is java domain specific language for simplifying the REST bases services built on top of the HTTP builder. It supports GET, POST, DELETE, PATCH, PUT and HEAD Request and can be used to validate and verify the response of the requests

Setup requirement:

* Java
* Set up environment path – set system variable JAVA\_Home and path variable
* Download eclipse/intellij
* Download REST assured

Every rest assured working on 3 keywords

**Given**: take all the input what you want to

**When**: Submit the API- resource (name of the API), http method (get ,post etc) apart from these 2 all in Given

**Then**: validate the response

**Remember**: We need to add static package for the static methods using key word static like this

For given() method

import static io.restassured.RestAssured.\*;

For equalTo() method

import static org.hamcrest.Matchers.\*; //used for the equalTo() method

log().all() use for showing the input logs and output logs in the console

Example:

import io.restassured.RestAssured;

import static io.restassured.RestAssured.\*; //used for the given() method

import static org.hamcrest.Matchers.\*; //used for the equalTo() method

public class Basics {

public static void main(String[] args) {

//add place API

RestAssured.baseURI="https://www.rahulshettyacademy.com"; //set the base url

given().log().all()

.queryParam("key ", "qaclick123")

.header("Content-Type","application/json")

.body("{\n" +

" \"location\": {\n" +

" \"lat\": -38.383494,\n" +

" \"lng\": 33.427362\n" +

" },\n" +

" \"accuracy\": 50,\n" +

" \"name\": \"Frontline house testing\",\n" +

" \"phone\_number\": \"(+91) 983 893 3937\",\n" +

" \"address\": \"29, side layout, cohen 09\",\n" +

" \"types\": [\n" +

" \"shoe park\",\n" +

" \"shop\"\n" +

" ],\n" +

" \"website\": \"http://google.com/maps\",\n" +

" \"language\": \"French-IN\"\n" +

"}\n")

.when().post("maps/api/place/add/json") //resource name

.then().log().all().assertThat() //check the assertion

.statusCode(200)

.body("scope",equalTo("APP")) //checking the value in the response body

.header("server", "Apache/2.4.18 (Ubuntu)"); //header response [how to see in postman: response>>header]

}

}

Test case: Add a place, update the Address and check whether it is updated or not?

package codes;

import files.Payload;

import io.restassured.RestAssured;

import io.restassured.path.json.JsonPath;

import static io.restassured.RestAssured.given;

import static org.hamcrest.Matchers.equalTo;

public class AddUpdateGetExampleTestCase {

public static void main(String[] args) {

//add place API - update the Place in API - get the API

//Stored response into String

RestAssured.baseURI="https://www.rahulshettyacademy.com"; //set the base url

String response = given().log().all()

.queryParam("key ", "qaclick123")

.header("Content-Type","application/json")

.body(Payload.addPlace())

.when().post("maps/api/place/add/json") //resource name

.then().assertThat() //check the assertion

.statusCode(200)

.body("scope",equalTo("APP")) //checking the value in the response body

.header("server", "Apache/2.4.18 (Ubuntu)") //header response [how to see in postman: response>>header]

.extract().response() //used to extract body

.asString(); //used to convert into string

//print the response

System.out.println("Response: "+response);

//JsonPath converts string into the Json or parsing Json, take String as an argument

JsonPath jsonPath = new JsonPath(response);

//get the place Id value

String placeID = jsonPath.getString("place\_id"); //remember place id is on parent position so directly passed

System.out.println("placeID: "+placeID);

String addressNew = "Winter Mall, Sydney";

//Update API

given().log().all()

.queryParam("key", "qaclick123")

.header("Content-Type","application/json")

.body("{\n" +

"\"place\_id\":\""+placeID+"\",\n" +

"\"address\":\""+addressNew+"\",\n" +

"\"key\":\"qaclick123\"\n" +

"}\n")

.when().put("maps/api/place/update/json")

.then().assertThat().statusCode(200)

.body("msg",equalTo("Address successfully updated"));

//get API

String updatedResponse = given().log().all()

.queryParam("key", "qaclick123")

.queryParam("place\_id", placeID)

.when().get("maps/api/place/get/json")

.then()

.assertThat()

.log().all().statusCode(200)

.extract().response().asString();

//JsonPath converts string into the Json or parsing Json, take String as an argument

JsonPath jsonPath1 = new JsonPath(updatedResponse);

//get the place Id value

String actualAddress = jsonPath1.getString("address"); //remember address is on parent position so directly passed

System.out.println("actualAddress: "+actualAddress);

}

}

**Nested Json and parsing**

Json example:

{

"dashboard": {

"purchaseAmount": 910,

"website": "rahulshettyacademy.com"

},

"courses": [

{

"title": "Selenium Python",

"price": 50,

"copies": 6

},

{

"title": "Cypress",

"price": 40,

"copies": 4

},

{

"title": "RPA",

"price": 45,

"copies": 10

}

]

}

1.Print No of courses returned by API

2.Print Purchase Amount

3. Print Title of the first course

4. Print All course titles and their respective Prices

5. Print no of copies sold by RPA Course

6. Verify if Sum of all Course prices matches with Purchase Amount

* Whenever you see the [] square bracket then it means it is an array
* Use any json editor online for better json request/ response. Ex : <https://jsoneditoronline.org/#left=local.cinemi&right=local.yiceke>
* If you don’t have the real API we need to create mock API, i.e. create json body

**Solution:**

package codes;

import files.Payload;

import io.restassured.path.json.JsonPath;

import org.testng.Assert;

import org.testng.annotations.Test;

public class ComplexJsonParse {

public static void main(String[] args) {

JsonPath jsonPath = new JsonPath(Payload.complexJsonBody());

//1.Print No of courses returned by API

//Size() method returns the size of array

int courseSize = jsonPath.getInt("courses.size()");

System.out.println("No of courses: "+courseSize);

//2.Print Purchase Amount

int totalAmount = jsonPath.getInt("dashboard.purchaseAmount");

System.out.println("Purchase Amount: "+totalAmount);

//3. Print Title of the first course

// courses is an array and we want to first title which is on index 0

//we can use get or getString

String firstCourse = jsonPath.get("courses[0].title");

System.out.println("first Course: "+firstCourse);

//4. Print All course titles and their respective Prices

//getInt for integer

for(int i=0; i<courseSize;i++)

{

String title = jsonPath.get("courses["+i+"].title");

int price = jsonPath.getInt("courses["+i+"].price");

System.out.println("Course title : "+title+ " and price is : "+price);

}

//5. Print no of copies sold by RPA Course

int RPACopies = jsonPath.getInt("courses[2].copies");

System.out.println("RPA Copies: "+RPACopies);

//6. Verify if Sum of all Course prices matches with Purchase Amount

for(int i=0; i<courseSize;i++)

{

String title = jsonPath.get("courses["+i+"].title");

int price = jsonPath.getInt("courses["+i+"].price");

int copies = jsonPath.getInt("courses["+i+"].copies");

System.out.println("Course title : "+title+ " and price is : "+price);

}

}

//6. Verify if Sum of all Course prices matches with Purchase Amount

@Test

public void sumOfAllCourses()

{

int sum = 0;

JsonPath jsonPath = new JsonPath(Payload.complexJsonBody());

int courseSize = jsonPath.getInt("courses.size()");

for(int i=0; i<courseSize;i++)

{

String title = jsonPath.get("courses["+i+"].title");

int price = jsonPath.getInt("courses["+i+"].price");

int copies = jsonPath.getInt("courses["+i+"].copies");

int total = price\*copies;

sum = sum+total;

}

int expectedTotal = jsonPath.getInt("dashboard.purchaseAmount");

System.out.println("actual sum: "+sum);

System.out.println("expectedTotal sum: "+expectedTotal);

Assert.assertEquals(sum,expectedTotal,"not matched");

}

}

**Why Dynamic Json payloads are important to understand**

Advance payload strategies - we are using ***Library API*** for that

Example

**Library API :**

BaseURI : http://216.10.245.166

1. **Resource** : Library/Addbook.php       **Method** : POST

**Input Payload : Json**:

{

"name":"Learn Appium Automation with Java",

"isbn":"bcd",

"aisle":"227",

"author":"John foe"

}

**Output Json**

{

"Msg": "successfully added",

"ID": "bcd227"

}

1. **Resource** : /Library/GetBook.php?AuthorName=somename **Method** : GET

**Output Json** :

Output the array of Json object books with all below  details

{

Name : “bookname”   ( String)

Isbn :  “A2fdsf”   (String)

Aisle : 32 (Integer)

}

1. **Resource** : Library/GetBook.php?ID=3389      - **Method** : GET

**Output Json :**

{

"book\_name": "Selenium automation using Java",

"isbn": "a23hd738",

"aisle": "1223"

}

1. **Resource** :/Library/DeleteBook.php      **Method** : POST

**Input Payload : Json:**

{

"ID" : "a23h345122332"

}

**Output Response** :

{  
  
msg : book is successfully deleted”

}

Example:

**Body**

public static String addBoobBody(String isbn, String aisle)

{

String payload = "{\n" +

"\n" +

"\"name\":\"Learn Appium Automation with Java\",\n" +

"\"isbn\":\""+isbn+"\",\n" +

"\"aisle\":\""+aisle+"\",\n" +

"\"author\":\"John foe\"\n" +

"}\n";

return payload;

}

**Code**

package codes;

import static io.restassured.RestAssured.\*;

import files.Payload;

import files.RawUsableToJson;

import io.restassured.path.json.JsonPath;

import org.testng.annotations.Test;

import io.restassured.RestAssured;

public class DynamicJsonLibraryAPI {

@Test

public void addBook()

{

RestAssured.baseURI = "http://216.10.245.166";

String response = given()

.header("Content-Type","application/json")

.body(Payload.addBoobBody("abcd","32232"))

.when()

.post("Library/Addbook.php")

.then().log().all().assertThat().statusCode(200)

.extract().response().asString();

JsonPath jsonPath = RawUsableToJson.rawToJson(response);

String bookID= jsonPath.get("ID");

System.out.println("Book id = "+bookID);

}

}

**Pass Data using Data Provider**

package codes;

import static io.restassured.RestAssured.\*;

import files.Payload;

import files.RawUsableToJson;

import io.restassured.path.json.JsonPath;

import org.testng.annotations.DataProvider;

import org.testng.annotations.Test;

import io.restassured.RestAssured;

public class DynamicJsonLibraryAPI {

@Test(dataProvider = "getBookData")

//need to declare arguments in the addBook test case as per Data provider getBookData

//test case and the data provider have same arguments

public void addBook(String isbn,String aisle)

{

RestAssured.baseURI = "http://216.10.245.166";

String response = given()

.header("Content-Type","application/json")

.body(Payload.addBoobBody(isbn,aisle))

.when()

.post("Library/Addbook.php")

.then().log().all().assertThat().statusCode(200)

.extract().response().asString();

JsonPath jsonPath = RawUsableToJson.rawToJson(response);

String bookID= jsonPath.get("ID");

System.out.println("Book id = "+bookID);

}

@DataProvider(name = "getBookData")

public Object[][] getData()

{

//Array - collection of elements

//Multi Dimensions array: collection of arrays

return new Object[][] {{"abda","323232"}, {"fsfs","323232"}, {"dsds","323232"}};

}

}

**Passing the static json in the Json file**

Open the Notepad >> Paste the json body without special character like \n\t\r>>save with .json format

Json file - AddBook.json

{

"name":"Learn Appium Automation with Java",

"isbn":"bcd",

"aisle":"228333",

"author":"John foe"

}

package codes;

import files.Payload;

import files.RawUsableToJson;

import io.restassured.RestAssured;

import io.restassured.path.json.JsonPath;

import org.testng.annotations.DataProvider;

import org.testng.annotations.Test;

import java.io.IOException;

import java.nio.file.Files;

import java.nio.file.Paths;

import static io.restassured.RestAssured.given;

public class StaticJson {

@Test

//Pass static json using .json file

public void addBook() throws IOException {

RestAssured.baseURI = "http://216.10.245.166";

String response = given()

.header("Content-Type","application/json")

.body(StaticJson.GenerateStringFromSource

("C:\\Sankalp Gupta Work\\Knowledge and Learning\\javaRest\\AddBook.json"))

.when()

.post("Library/Addbook.php")

.then().log().all().assertThat().statusCode(200)

.extract().response().asString();

JsonPath jsonPath = RawUsableToJson.rawToJson(response);

String bookID= jsonPath.get("ID");

System.out.println("Book id = "+bookID);

}

//method that generate body payload from file statically

public static String GenerateStringFromSource(String path) throws IOException {

return new String(Files.readAllBytes(Paths.get( path)));

}

}

Restart Localhost: C:\Users\cook\Atlassian\Jira\bin\start-jira run it

sankalp.test1 /sis1234!

**Session Authentication in JIRA**

<http://localhost:8080/rest/auth/1/session>

-H 'content-type: application/json' \

-d '{ "username": " Sankalp.test1 ", "password": " sis1234!" }'

**Response**

{

"session": {

"name": "JSESSIONID",

"value": "081086F3DE0774E9347746A5D8938B1C"

},

"loginInfo": {

"failedLoginCount": 1,

"loginCount": 4,

"lastFailedLoginTime": "2020-07-04T15:36:08.005+0530",

"previousLoginTime": "2020-07-04T15:52:40.295+0530"

}

}

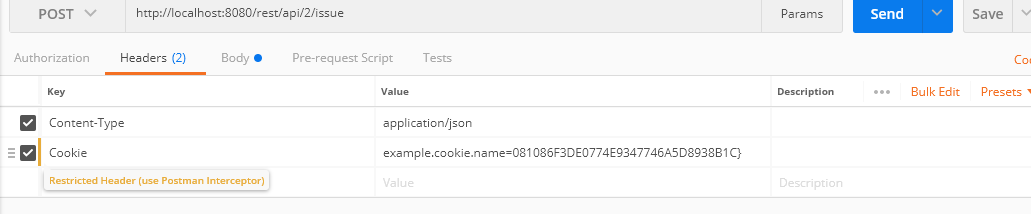
**Now creating issue**

headers: {cookie: example.cookie.name=081086F3DE0774E9347746A5D8938B1C}

**//pass ur own session id**

**For every project there is a key – RestAssured key : RES**

**Header**



**Body**

{

"fields": {

"project": {

"key": "RES"

},

"summary": "REST Project - Sankalp",

"description": "Creating my second bug",

"issuetype": {

"name": "Bug"

}

}

}

**Response**

{

"id": "10023",

"key": "RES-24",

"self": "http://localhost:8080/rest/api/2/issue/10023"

}

{

"id": "10024",

"key": "RES-25",

"self": "http://localhost:8080/rest/api/2/issue/10024"

}

**Delete issue:**

Delete

<http://localhost:8080//rest/api/2/issue/10023>

**Add Comment**

http://localhost:8080/rest/api/2/issue/10100/comment

{

"body": "Test Comment",

"visibility": {

"type": "role",

"value": "Administrators"

}

}

Response:

{

"self": "http://localhost:8080/rest/api/2/issue/10100/comment/10100",

"id": "10100",

"author": {

"self": "http://localhost:8080/rest/api/2/user?username=sankalp.test1",

"name": "sankalp.test1",

"key": "JIRAUSER10000",

"emailAddress": "sankalp.test1@gmail.com",

"avatarUrls": {

"48x48": "https://www.gravatar.com/avatar/75e0d5cf87534c92277e61d09ce36c2b?d=mm&s=48",

"24x24": "https://www.gravatar.com/avatar/75e0d5cf87534c92277e61d09ce36c2b?d=mm&s=24",

"16x16": "https://www.gravatar.com/avatar/75e0d5cf87534c92277e61d09ce36c2b?d=mm&s=16",

"32x32": "https://www.gravatar.com/avatar/75e0d5cf87534c92277e61d09ce36c2b?d=mm&s=32"

},

"displayName": "sankalp.test1@gmail.com",

"active": true,

"timeZone": "Asia/Calcutta"

},

"body": "Test Comment",

"updateAuthor": {

"self": "http://localhost:8080/rest/api/2/user?username=sankalp.test1",

"name": "sankalp.test1",

"key": "JIRAUSER10000",

"emailAddress": "sankalp.test1@gmail.com",

"avatarUrls": {

"48x48": "https://www.gravatar.com/avatar/75e0d5cf87534c92277e61d09ce36c2b?d=mm&s=48",

"24x24": "https://www.gravatar.com/avatar/75e0d5cf87534c92277e61d09ce36c2b?d=mm&s=24",

"16x16": "https://www.gravatar.com/avatar/75e0d5cf87534c92277e61d09ce36c2b?d=mm&s=16",

"32x32": "https://www.gravatar.com/avatar/75e0d5cf87534c92277e61d09ce36c2b?d=mm&s=32"

},

"displayName": "sankalp.test1@gmail.com",

"active": true,

"timeZone": "Asia/Calcutta"

},

"created": "2020-07-06T14:45:01.011+0530",

"updated": "2020-07-06T14:45:01.011+0530",

"visibility": {

"type": "role",

"value": "Administrators"

}

}

**Update comment:**

**Using JIRA API**

**SessionFilter :** class is used to listen and store the response of the session.

Or we can use the json path for auth session

How to read Curl body

curl -D- -u admin:admin -X POST -H "X-Atlassian-Token: no-check" -F "file=@myfile.txt" http://myhost/rest/api/2/issue/TEST-123/attachments

-D- Different paramaters

-u : Credentials

-X: Type of method Like post, delete etc

-H : Header

-F: File name : file is the key

Path parameter: extract data from the data

Query parameter: extract data from the results

package codes;

import io.restassured.RestAssured;

import io.restassured.filter.session.SessionFilter;

import io.restassured.path.json.JsonPath;

import org.testng.Assert;

import java.io.File;

import static io.restassured.RestAssured.\*; //used for the given() method

public class JiraTest {

public static void main(String[] args) {

RestAssured.baseURI="http://localhost:8080"; //set the base url

//Session auth

SessionFilter sessionFilter = new SessionFilter();

String response = given().header("Content-Type","application/json")

.body("{ \"username\": \"sankalp.test1\", \"password\": \"sis1234!\" }")

.log().all()

.filter(sessionFilter) //store session use before when

.when()

.post("/rest/auth/1/session")

.then().log().all().

extract().response().asString();

//add comment

String expectedMessage = "HI - Test Comment";

String addCommentResponse = given().pathParam("id","10100")

.log().all()

.header("Content-Type","application/json")

.body("{\n" +

" \"body\": \""+expectedMessage+"\",\n" +

" \"visibility\": {\n" +

" \"type\": \"role\",\n" +

" \"value\": \"Administrators\"\n" +

" }\n" +

"}")

.filter(sessionFilter) //store session use before when

.when()

.post("/rest/api/2/issue/{id}/comment") //same path parameter which is declare above, you can use any

.then().assertThat().statusCode(201).extract().response().asString();

JsonPath jsonPath = new JsonPath(addCommentResponse);

String commentID = jsonPath.getString("id");

System.out.println("commentID: "+commentID);

//add attachment - use header multipart form data

given().header("X-Atlassian-Token","no-check")

.filter(sessionFilter)

.pathParam("id","10100")

.header("Content-Type","multipart/form-data")

.multiPart("file",new File("One.txt"))

.when()

.post("/rest/api/2/issue/{id}/attachments")

.then()

.log().all()

.assertThat()

.statusCode(200);

//get issue and check

String issueDetails= given().filter(sessionFilter)

.pathParam("id","10100")

//pass query parameter to restrict only selected value in response

.queryParam("fields","comment")

.log().all()

.when().get("/rest/api/2/issue/{id}")

.then()

.log().all()

.extract().response().asString();

System.out.println("issueDetails: "+issueDetails);

JsonPath jsonPath1 = new JsonPath(issueDetails);

int commentSize = jsonPath1.getInt("fields.comment.comments.size()");

for (int i=0;i<commentSize;i++)

{

String commentIdInList = jsonPath1.get("fields.comment.comments["+i+"].id").toString();

System.out.println(commentIdInList);

if (commentIdInList.equalsIgnoreCase(commentID))

{

String message = jsonPath1.get("fields.comment.comments["+i+"].body").toString();

System.out.println(message);

Assert.assertEquals(message,expectedMessage,"Not matched");

}

}

}

}

**OAuth2:**

**What is OAuth 2.0?**

It is the Industry based Authorization protocol.

It can used third party website for integrating the Web services like FB, Google etc.

**Authorization Grant Type**

**Need:**

* no data breach headque for application,
* need not maintain user profile data,
* Allow richer website talk to each other like FB and bool my show share password and details which are needed

**Terms:**

**Client**: Application where it is expecting to the serve. For ex: Book my show to register in Google for OAuth2.0 with google.

**Client ID:** Which can uniquely identify the Book my show in Google

**Client Secret ID:** Which can uniquely identify the Book my show in Google

**Resource Owner:** Owner/resources – profile information

**Resource/Authorization server**: Server for the profile provider like Google

**Step1:**

1. User sign into google by google authorization and get code
2. Application will receive code to hit google resource server and get [access token, Name, Image]

Get code and exchange code needed for Auth

|  |  |
| --- | --- |
| **Grant Type** | **Authorization code** |
| redirect URL/Callback URL | https://rahulshettyacademy.com/getCourse.php |
| Authorization server url | https://accounts.google.com/o/oauth2/v2/auth |
| Access token url | https://www.googleapis.com/oauth2/v4/token |
| Client ID | 692183103107-p0m7ent2hk7suguv4vq22hjcfhcr43pj.apps.googleusercontent.com |
| Client Secret | erZOWM9g3UtwNRj340YYaK\_W |
| Scope | https://www.googleapis.com/auth/userinfo.email |
| State | Any random string |
| How to pass oauth in request | Headers |

Mandatory fields for GetAuthorization Code Request ;

**End Point** : Authorization server url  
**Query Params**:Scope, Auth\_url, client\_id, response\_type, redirect\_uri

This operation performed in the browser  
output : Code

Mandatory fields for GetAccessToken Request :  
**End point** : Access token url

**Query Params** :Code, client\_id, client\_secret, redirect\_uri, grant\_type

Output : Access token

**Oauth using Rahul Shetty example:**

**scope:https://www.googleapis.com/auth/userinfo.email**

**auth\_url:https://accounts.google.com/o/oauth2/v2/auth**

**client\_id:692183103107-p0m7ent2hk7suguv4vq22hjcfhcr43pj.apps.googleusercontent.com**

**response\_type:code**

**redirect\_uri:https://rahulshettyacademy.com/getCourse.php**

**Client:** Rahulshetty.com

**Scope:** What you want from google

**auth\_url:** From which authorization like google account, FB account

**client\_id**: when first time your website registered in google, provided by the google

**response\_type:** like code

**redirect\_uri:** From where google send you once login success

**state:** Optional parameter [for security purpose]

**Step1: Getcode**:

Hit this URL

[https://rahulshettyacademy.com/getCourse.php?code=4%2F3QGrDKS4c4xMZcv6oFWsKApUoyxz-vPddKXONAK4dNx-nsCc0j\_Cwmsx9ByC1MQZelxgmIt6TouZotmqEFx4ovk&scope=email+https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fuserinfo.email+openid&authuser=0&prompt=consent#](https://rahulshettyacademy.com/getCourse.php?code=4%2F3QGrDKS4c4xMZcv6oFWsKApUoyxz-vPddKXONAK4dNx-nsCc0j_Cwmsx9ByC1MQZelxgmIt6TouZotmqEFx4ovk&scope=email+https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fuserinfo.email+openid&authuser=0&prompt=consent)

Login with your account

Copy URL in postman

Get Code: 4%2F3QGrDKS4c4xMZcv6oFWsKApUoyxz-vPddKXONAK4dNx-nsCc0j\_Cwmsx9ByC1MQZelxgmIt6TouZotmqEFx4ovk

Details in postman: Get

Copy this

<https://accounts.google.com/o/oauth2/v2/auth?scope=https://www.googleapis.com/auth/userinfo.email&auth_url=https://accounts.google.com/o/oauth2/v2/auth&client_id=692183103107-p0m7ent2hk7suguv4vq22hjcfhcr43pj.apps.googleusercontent.com&response_type=code&redirect_uri=https://rahulshettyacademy.com/getCourse.php>

**Param pass**

scope:https://www.googleapis.com/auth/userinfo.email

auth\_url:https://accounts.google.com/o/oauth2/v2/auth

client\_id:692183103107-p0m7ent2hk7suguv4vq22hjcfhcr43pj.apps.googleusercontent.com

response\_type:code

redirect\_uri:https://rahulshettyacademy.com/getCourse.php

**response: copy code**

[https://rahulshettyacademy.com/getCourse.php?code=4%2F3QFtHWSAQL8G8brmZ2nFyQIlZWMuZPOtEPLCMzR-Ht0GrxJc0hmr2c0rgnj3yjsiOJVZEjgyP8PJoGLChcocspg&scope=email+https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fuserinfo.email+openid&authuser=0&prompt=consent#](https://rahulshettyacademy.com/getCourse.php?code=4%2F3QFtHWSAQL8G8brmZ2nFyQIlZWMuZPOtEPLCMzR-Ht0GrxJc0hmr2c0rgnj3yjsiOJVZEjgyP8PJoGLChcocspg&scope=email+https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fuserinfo.email+openid&authuser=0&prompt=consent)

**Step2: Exchange code**:

Client id and client secret id should be generated when Login first time with google, FB. Should asked to Developer for the same.

Grant type: Authorization code

OAuth 2.0 Contract Details:

|  |  |
| --- | --- |
| GrantType | Authorization code |
| redirect URL/Callback URL | https://rahulshettyacademy.com/getCourse.php |
| Authorization server url | https://accounts.google.com/o/oauth2/v2/auth |
| Access token url | https://www.googleapis.com/oauth2/v4/token |
| Client ID | 692183103107-p0m7ent2hk7suguv4vq22hjcfhcr43pj.apps.googleusercontent.com |
| Client Secret | erZOWM9g3UtwNRj340YYaK\_W |
| Scope | https://www.googleapis.com/auth/userinfo.email |
| State | Any random string |
| How to pass oauth in request | Headers |

Mandatory fields for GetAuthorization Code Request ;

**End Point** : Authorization server url  
**Query Params**:Scope, Auth\_url, client\_id, response\_type, redirect\_uri  
  
output : Code

Mandatory fields for GetAccessToken Request :  
**End point** : Access token url

**Query Params** :Code, client\_id, client\_secret, redirect\_uri, grant\_type

Output : Access token

Post:

<https://www.googleapis.com/oauth2/v4/token?code=4%2F3QGrDKS4c4xMZcv6oFWsKApUoyxz-vPddKXONAK4dNx-nsCc0j_Cwmsx9ByC1MQZelxgmIt6TouZotmqEFx4ovk&client_id=692183103107-p0m7ent2hk7suguv4vq22hjcfhcr43pj.apps.googleusercontent.com&client_secret=erZOWM9g3UtwNRj340YYaK_W&redirect_uri=https://rahulshettyacademy.com/getCourse.php&grant_type=authorization_code>

**Params**

code:4%2F3QGrDKS4c4xMZcv6oFWsKApUoyxz-vPddKXONAK4dNx-nsCc0j\_Cwmsx9ByC1MQZelxgmIt6TouZotmqEFx4ovk

client\_id:692183103107-p0m7ent2hk7suguv4vq22hjcfhcr43pj.apps.googleusercontent.com

client\_secret:erZOWM9g3UtwNRj340YYaK\_W

redirect\_uri:https://rahulshettyacademy.com/getCourse.php

grant\_type:authorization\_code

**Response: copy access token**

{

"access\_token": "ya29.a0AfH6SMBKVFIle\_oy2hc72L9YN7Z4nAaC9oePryfF77Jyu02kZdrq76fLdn6HiueS1DD7WNJWMvD5V-Kxqc-j\_9ALsODw31q7th3eYFXu\_tZq4RkM0KyP8ukXNkJG6UePtuLF5LdhXrEES3nYw3ohZbYruLzIldWCQfA",

"expires\_in": 3599,

"scope": "openid https://www.googleapis.com/auth/userinfo.email",

"token\_type": "Bearer",

"id\_token": "eyJhbGciOiJSUzI1NiIsImtpZCI6IjBhN2RjMTI2NjQ1OTBjOTU3ZmZhZWJmN2I2NzE4Mjk3Yjg2NGJhOTEiLCJ0eXAiOiJKV1QifQ.eyJpc3MiOiJodHRwczovL2FjY291bnRzLmdvb2dsZS5jb20iLCJhenAiOiI2OTIxODMxMDMxMDctcDBtN2VudDJoazdzdWd1djR2cTIyaGpjZmhjcjQzcGouYXBwcy5nb29nbGV1c2VyY29udGVudC5jb20iLCJhdWQiOiI2OTIxODMxMDMxMDctcDBtN2VudDJoazdzdWd1djR2cTIyaGpjZmhjcjQzcGouYXBwcy5nb29nbGV1c2VyY29udGVudC5jb20iLCJzdWIiOiIxMDkzMzc2Nzg1OTA2MzI1MzE5OTYiLCJlbWFpbCI6InNhbmthbHAudGVzdDFAZ21haWwuY29tIiwiZW1haWxfdmVyaWZpZWQiOnRydWUsImF0X2hhc2giOiJUSWNtY2RBOVE3dDhNZWFUZzJVaGFBIiwiaWF0IjoxNTk4MzgyODk1LCJleHAiOjE1OTgzODY0OTV9.N8bnXbHmhovuBTqNVileuR6WMl4aPTObYx8K9kNmfXfw-uZVCCiOMhISlQD01fzJaPUGnvxV5SnJYQ4352HMaLE86If96t\_\_cpnZ4ewunl76YTBstLrSAYfOldfS7NUnWFfnQdSKaHxHznCrvE\_yM42sDRg5ANGCEx4nRVclNDZEu\_YrOA1JlzZd7JbZqlsxtOqgBL3vRpKNIWxBPwQ8eTDCKQ9EsQ\_VzySLqxn1jzSDNijJmkvZHdPhwWcMtSZm9PE1nknMqpvn2D42WiWWNiL6sMUj8ygs5bHp5Y3wkNmbF-7G\_4ZQlbqlAszyY95Bu88SRRO3539WCYKzZrxCzg"

}

**Step3: ActualRequest**:

Get: <https://rahulshettyacademy.com/getCourse.php>

Copy params: access\_token:ya29.a0AfH6SMBKVFIle\_oy2hc72L9YN7Z4nAaC9oePryfF77Jyu02kZdrq76fLdn6HiueS1DD7WNJWMvD5V-Kxqc-j\_9ALsODw31q7th3eYFXu\_tZq4RkM0KyP8ukXNkJG6UePtuLF5LdhXrEES3nYw3ohZbYruLzIldWCQfA

Response:

{ "instructor": "RahulShetty", "url": "rahulshettycademy.com", "services": "projectSupport", "expertise": "Automation", "courses": { "webAutomation": [ { "courseTitle": "Selenium Webdriver Java", "price": "50" }, { "courseTitle": "Cypress",

"price": "40"

},

{

"courseTitle": "Protractor",

"price": "40"

}

],

"api": [

{

"courseTitle": "Rest Assured Automation using Java",

"price": "50"

},

{

"courseTitle": "SoapUI Webservices testing",

"price": "40"

}

],

"mobile": [

{

"courseTitle": "Appium-Mobile Automation using Java",

"price": "50"

}

]

},

"linkedIn": "https://www.linkedin.com/in/rahul-shetty-trainer/"

}

**Generate the Access token using Postman – need to check**

**1. OAuth 2.0 – Authorization code grant type**

**Authorization code grant type** is needed when a user want to access any application uses third party integration for user interaction.

Ex: Bookmyshow.com

**Oauth code**

package codes;

import com.fasterxml.jackson.annotation.JsonAlias;

import io.restassured.path.json.JsonPath;

import org.openqa.selenium.By;

import org.openqa.selenium.Keys;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeDriver;

import static io.restassured.RestAssured.given;

public class OAuthTest {

public static void main(String[] args) throws InterruptedException {

//Google now won't allow to automate account login

// System.setProperty("webdriver.chrome.driver","C:\\Sankalp Gupta Work\\Knowledge and Learning\\Udemi DOc and JArs");

// WebDriver driver = new ChromeDriver();

//use to get the code

// driver.get("https://accounts.google.com/o/oauth2/v2/auth?scope=https://www.googleapis.com/auth/userinfo.email&auth\_url=https://accounts.google.com/o/oauth2/v2/auth&client\_id=692183103107-p0m7ent2hk7suguv4vq22hjcfhcr43pj.apps.googleusercontent.com&response\_type=code&redirect\_uri=https://rahulshettyacademy.com/getCourse.php");

// driver.findElement(By.id("identifierId")).sendKeys("sankalp.test1@gmail.com");

// driver.findElement(By.id("identifierId")).sendKeys(Keys.ENTER);

// Thread.sleep(5);

// driver.findElement(By.name("password")).sendKeys("sis1234!");

// driver.findElement(By.id("password")).sendKeys("sankalp.test1@gmail.com");

// Thread.sleep(5);

// String url = driver.getCurrentUrl();

String url = "https://rahulshettyacademy.com/getCourse.php?code=4%2F4AHjzpaocwsoiE1sCfufPQOSMoD2ydtrTobO7oyJ3Ntcv9DbPbKweX33391QaLr4Lw9UbCSgjztLs05M6T66DC0&scope=email+https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fuserinfo.email+openid&authuser=0&prompt=none#";

String partialCode= url.split("code=")[1];

String code = partialCode.split("&")[0];

System.out.println("code= "+code);

//Get access code

String accessTokenResponse = given().urlEncodingEnabled(false) //not convert any value in numeric and other of the code

.queryParam("code",code)

.queryParam("client\_id","692183103107-p0m7ent2hk7suguv4vq22hjcfhcr43pj.apps.googleusercontent.com")

.queryParam("client\_secret","erZOWM9g3UtwNRj340YYaK\_W")

.queryParam("redirect\_uri","https://rahulshettyacademy.com/getCourse.php")

.queryParam("grant\_type","authorization\_code")

.when()

.log().all()

.post("https://www.googleapis.com/oauth2/v4/token")

.asString();

JsonPath jsonPath = new JsonPath(accessTokenResponse);

String accessToken = jsonPath.getString("access\_token");

//get Response after access token

String response = given().queryParam("access\_token",accessToken)

.when()

.log().all()

.get("https://rahulshettyacademy.com/getCourse.php").asString();

System.out.println("access token: "+response);

}

}

**If you want third party authorization, then you must need authorization code grant type**

**2. OAuth 2.0 - Client Credential grant type**

**Client Credential grant type** is needed when a website needs own data from the other website. For example : I have website named xyz.com and people tagging me on twitter , so I want to get my tweet on my xyz.com website so I can use client credential grant type

It does not require authorization code, it comes into picture when human intervention required.

In this you did not require, as you need to your data..

In above example we don’t need authorization, we can direct start with access code

//Get access code

String accessTokenResponse = given().urlEncodingEnabled(false) //not convert any value in numeric and other of the code

.queryParam("code",code)

.queryParam("client\_id","692183103107-p0m7ent2hk7suguv4vq22hjcfhcr43pj.apps.googleusercontent.com")

.queryParam("client\_secret","erZOWM9g3UtwNRj340YYaK\_W")

.queryParam("redirect\_uri","https://rahulshettyacademy.com/getCourse.php")

.queryParam("grant\_type","authorization\_code")

.when()

.log().all()

.post("https://www.googleapis.com/oauth2/v4/token")

.asString();

JsonPath jsonPath = new JsonPath(accessTokenResponse);

String accessToken = jsonPath.getString("access\_token");

//get Response after access token

String response = given().queryParam("access\_token",accessToken)

.when()

.log().all()

.get("https://rahulshettyacademy.com/getCourse.php").asString();

System.out.println("access token: "+response);

**Serialization and Deserialization of request/response with POJO classes**

**Serialization** is the process of converting a Java object into Request body (payload) or it covert java object into the Json payload

Java **POJO** refers to the Plain old Java object

Rest-assured converts POJO classes into json payload.

**CODE CODE TO ASSIGN Rest Assured JSON**

Message m = new Message();

m.setMessage(“Hello”); m.setGreet(“Hi”);

{

“Message”:”Hello”

“greet”:”Hi”

}

Given().body(m)

.when().post(“/message”)

Public Class Messge

{

Private String message;

Private String greet;

Public String getMessage()

{ return message }

Public String setMessage(String message)

{ this.message = message }

Public String getGreet()

{ return greet }

Public String setGreet (String greet)

{ this. greet = greet }

}

**Serialization –** Setting the values so we use setters method

**Deserialization –** Getting the values so we use getters method

Adv: Easy to parse and extract responses (json/xml) if they are wrapped into java object

Library needed: Jackson, gson [

## [Jackson Databind](https://mvnrepository.com/artifact/com.fasterxml.jackson.core/jackson-databind) » [2.10.0](https://mvnrepository.com/artifact/com.fasterxml.jackson.core/jackson-databind/2.10.0)

## [Jackson Annotations](https://mvnrepository.com/artifact/com.fasterxml.jackson.core/jackson-annotations) » [2.10.0](https://mvnrepository.com/artifact/com.fasterxml.jackson.core/jackson-annotations/2.10.0)

## [Jackson Core](https://mvnrepository.com/artifact/com.fasterxml.jackson.core/jackson-core) » [2.10.0](https://mvnrepository.com/artifact/com.fasterxml.jackson.core/jackson-core/2.10.0)

For deserialization

Message m = new Message();

m.setMessage(“Hello”);

m.setGreet(“Hi”);

me.getMessage();

**How to write POJO for complex json**

{

"instructor": "RahulShetty",

"url": "rahulshettycademy.com",

"services": "projectSupport",

"expertise": "Automation",

"courses": {

"webAutomation": [

{

"courseTitle": "Selenium Webdriver Java",

"price": "50"

},

{

"courseTitle": "Cypress",

"price": "40"

},

{

"courseTitle": "Protractor",

"price": "40"

}

],

"api": [

{

"courseTitle": "Rest Assured Automation using Java",

"price": "50"

},

{

"courseTitle": "SoapUI Webservices testing",

"price": "40"

}

],

"mobile": [

{

"courseTitle": "Appium-Mobile Automation using Java",

"price": "50"

}

]

},

"linkedIn": "https://www.linkedin.com/in/rahul-shetty-trainer/"

}

Steps:

1. First write getters and setters for the parameters which are not array or return string

**Parent class**

public class GetCourse {  
  
 //for getter setter - select variables and press Alt+insert  
 private String url;  
 private String instructor;  
 private String services;  
 private String expertise;  
 private Courses courses; //return type is sub json  
 private String linkedIn;  
  
 public String getUrl() {  
 return url;  
 }  
  
 public void setUrl(String url) {  
 this.url = url;  
 }  
  
 public String getInstructor() {  
 return instructor;  
 }  
  
 public void setInstructor(String instructor) {  
 this.instructor = instructor;  
 }  
  
 public String getServices() {  
 return services;  
 }  
  
 public void setServices(String services) {  
 this.services = services;  
 }  
  
 public String getExpertize() {  
 return expertize;  
 }  
  
 public void setExpertise(String expertise) {  
 this.expertise = expertise;  
 }  
  
 public Courses getCourses() {  
 return courses;  
 }  
  
 public void setCourses(Courses courses) {  
 this.courses = courses;  
 }  
  
 public String getLinkedIn() {  
 return linkedIn;  
 }  
  
 public void setLinkedIn(String linkedIn) {  
 this.linkedIn = linkedIn;  
 }  
  
  
  
  
}

1. Now the Courses having array [] having Webautomation, API and Mobile sub-json
2. So we need to write child POJO classes for Courses

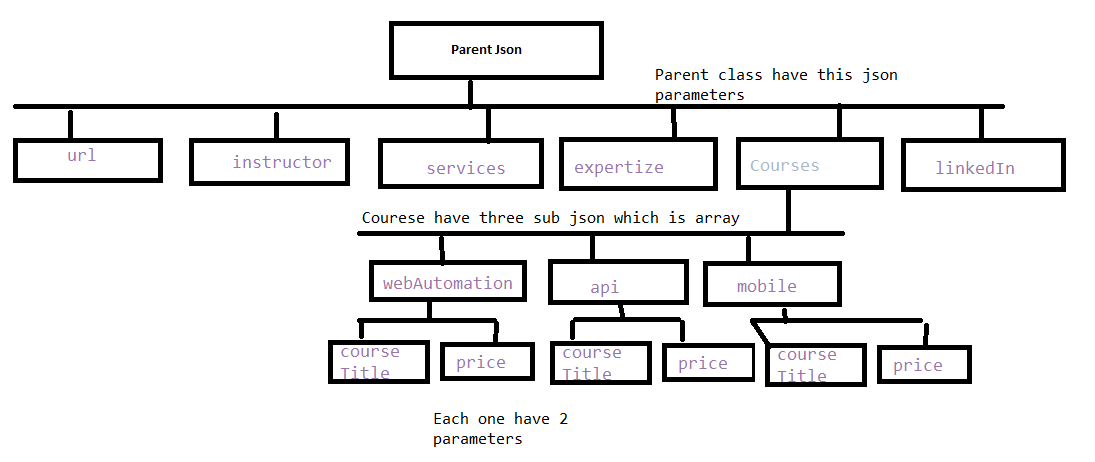
package pojo;  
  
import java.util.List;  
  
public class Courses {  
  
 private List<WebAutomation> webAutomation;  
 private List<API> api;  
 private List<Mobile> mobile;  
  
  
  
 public void setWebAutomation(List<WebAutomation> webAutomation) {  
 this.webAutomation = webAutomation;  
 }  
  
 public List<WebAutomation> getWebAutomation() {  
 return webAutomation;  
 }  
  
 public List<API> getApi() {  
 return api;  
 }  
  
 public void setApi(List<API> api) {  
 this.api = api;  
 }  
  
 public List<Mobile> getMobile() {  
 return mobile;  
 }  
  
 public void setMobile(List<Mobile> mobile) {  
 this.mobile = mobile;  
 }  
  
  
  
  
  
}

1. Now again courses have 3 sun array which are Webautomation, API and Mobile sub-json
2. Again create POJO classes for each
3. package pojo;  
     
   public class WebAutomation {  
     
    private String courseTitle;  
    private String price;  
     
    public String getCourseTitle() {  
    return courseTitle;  
    }  
     
    public void setCourseTitle(String courseTitle) {  
    this.courseTitle = courseTitle;  
    }  
     
    public String getPrice() {  
    return price;  
    }  
     
    public void setPrice(String price) {  
    this.price = price;  
    }  
   }

package pojo;  
  
public class Mobile {  
  
 private String courseTitle;  
 private String price;  
  
 public String getCourseTitle() {  
 return courseTitle;  
 }  
  
 public void setCourseTitle(String courseTitle) {  
 this.courseTitle = courseTitle;  
 }  
  
 public String getPrice() {  
 return price;  
 }  
  
 public void setPrice(String price) {  
 this.price = price;  
 }  
  
}

package pojo;  
  
public class API {  
  
 private String courseTitle;  
 private String price;  
  
 public String getCourseTitle() {  
 return courseTitle;  
 }  
  
 public void setCourseTitle(String courseTitle) {  
 this.courseTitle = courseTitle;  
 }  
  
 public String getPrice() {  
 return price;  
 }  
  
 public void setPrice(String price) {  
 this.price = price;  
 }  
   
}

in summary



Issue : Webdriver is not working

 Closed

[java.lang.NoSuchMethodError: com.google.common.base.Preconditions.checkState(ZLjava/lang/String;Ljava/lang/Object;Ljava/lang/Object;Ljava/lang/Object;)V](https://github.com/SeleniumHQ/selenium/issues/3880)

To resolve this issue Remove if you have added any jar files under referenced Library as it creates conflicts.  
Remove the jar it should work fine.

Example:

package codes;  
  
import io.restassured.parsing.Parser;  
import io.restassured.path.json.JsonPath;  
import org.openqa.selenium.By;  
import org.openqa.selenium.Keys;  
import org.openqa.selenium.WebDriver;  
import org.openqa.selenium.chrome.ChromeDriver;  
import pojo.GetCourse;  
  
import static io.restassured.RestAssured.*given*;  
  
  
//OAuthTest - AuthorizationType  
public class EndToEndPojo {  
  
 public static void main(String[] args) {  
// System.setProperty("webdriver.chrome.driver","C:/driver/chromedriver.exe");  
// WebDriver driver = new ChromeDriver();  
//  
// //use to get the code  
// driver.get("https://accounts.google.com/o/oauth2/v2/auth?scope=https://www.googleapis.com/auth/userinfo.email&auth\_url=https://accounts.google.com/o/oauth2/v2/auth&client\_id=692183103107-p0m7ent2hk7suguv4vq22hjcfhcr43pj.apps.googleusercontent.com&response\_type=code&redirect\_uri=https://rahulshettyacademy.com/getCourse.php");  
// driver.findElement(By.id("identifierId")).sendKeys("sankalp.test1@gmail.com");  
// driver.findElement(By.id("identifierId")).sendKeys(Keys.ENTER);  
// try {  
// Thread.sleep(5);  
// } catch (InterruptedException e) {  
// e.printStackTrace();  
// }  
// driver.findElement(By.name("password")).sendKeys("sis1234!");  
// driver.findElement(By.id("password")).sendKeys(Keys.ENTER);  
// try {  
// Thread.sleep(5);  
// } catch (InterruptedException e) {  
// e.printStackTrace();  
// }  
// String url = driver.getCurrentUrl();  
  
 String url = "https://rahulshettyacademy.com/getCourse.php?code=4%2F5AEU1e7nFoH0JLnPABk01OGQgyilOMcqtyOmDWJWpeUs0NPRiGBMFs29ovX\_ULRTrWdLSvZ4Pxu0KitbgYseoRc&scope=email+https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fuserinfo.email+openid&authuser=0&prompt=none#";  
 String partialCode= url.split("code=")[1];  
 String code = partialCode.split("&")[0];  
  
 System.*out*.println("code= "+code);  
  
 //Get access code  
 String accessTokenResponse = *given*().urlEncodingEnabled(false) //not convert any value in numeric and other of the code  
 .queryParam("code",code)  
 .queryParam("client\_id","692183103107-p0m7ent2hk7suguv4vq22hjcfhcr43pj.apps.googleusercontent.com")  
 .queryParam("client\_secret","erZOWM9g3UtwNRj340YYaK\_W")  
 .queryParam("redirect\_uri","https://rahulshettyacademy.com/getCourse.php")  
 .queryParam("grant\_type","authorization\_code")  
 .when()  
 .log().all()  
 .post("https://www.googleapis.com/oauth2/v4/token")  
 .asString();  
  
 JsonPath jsonPath = new JsonPath(accessTokenResponse);  
 String accessToken = jsonPath.getString("access\_token");  
  
 //get Response after access token  
 GetCourse getCourse = *given*().queryParam("access\_token",accessToken)  
 .expect()  
 .defaultParser(Parser.*JSON*)  
 .when()  
 .get("https://rahulshettyacademy.com/getCourse.php").as(GetCourse.class);  
  
 System.*out*.println("Linked in "+getCourse.getLinkedIn());  
 System.*out*.println("getInstructor "+getCourse.getInstructor());  
 System.*out*.println("getUrl "+getCourse.getUrl());  
 System.*out*.println("getServices "+getCourse.getServices());  
 System.*out*.println("getCourses "+getCourse.getCourses());  
  
  
 }  
}

expect() = is used to what output response you expected   
defaultParser(Parser.*JSON*) – response type \

more example:

package codes;  
  
import io.restassured.parsing.Parser;  
import io.restassured.path.json.JsonPath;  
import org.openqa.selenium.By;  
import org.openqa.selenium.Keys;  
import org.openqa.selenium.WebDriver;  
import org.openqa.selenium.chrome.ChromeDriver;  
import pojo.API;  
import pojo.Courses;  
import pojo.GetCourse;  
import pojo.WebAutomation;  
  
import java.util.List;  
  
import static io.restassured.RestAssured.*given*;  
  
  
//OAuthTest - AuthorizationType  
public class EndToEndPojo {  
  
 public static void main(String[] args) {  
  
 String url = "https://rahulshettyacademy.com/getCourse.php?code=4%2F5QG1IGSbvxwBLCmgKBs8TvBRL5oOewrzCTb-CuORrhHCYGU\_ryS2xwzJFrEWl-v3ANoTftITWGT6uaErosrrBkQ&scope=email+https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fuserinfo.email+openid&authuser=0&prompt=none#";  
 String partialCode= url.split("code=")[1];  
 String code = partialCode.split("&")[0];  
  
 System.*out*.println("code= "+code);  
  
 //Get access code  
 String accessTokenResponse = *given*().urlEncodingEnabled(false) //not convert any value in numeric and other of the code  
 .queryParam("code",code)  
 .queryParam("client\_id","692183103107-p0m7ent2hk7suguv4vq22hjcfhcr43pj.apps.googleusercontent.com")  
 .queryParam("client\_secret","erZOWM9g3UtwNRj340YYaK\_W")  
 .queryParam("redirect\_uri","https://rahulshettyacademy.com/getCourse.php")  
 .queryParam("grant\_type","authorization\_code")  
 .when()  
 .log().all()  
 .post("https://www.googleapis.com/oauth2/v4/token")  
 .asString();  
  
 JsonPath jsonPath = new JsonPath(accessTokenResponse);  
 String accessToken = jsonPath.getString("access\_token");  
  
 //get Response after access token  
 GetCourse getCourse = *given*().queryParam("access\_token",accessToken)  
 .expect()  
 .defaultParser(Parser.*JSON*)  
 .when()  
 .get("https://rahulshettyacademy.com/getCourse.php").as(GetCourse.class);  
  
 System.*out*.println("Linked in "+getCourse.getLinkedIn());  
 System.*out*.println("getInstructor "+getCourse.getInstructor());  
 System.*out*.println("getUrl "+getCourse.getUrl());  
  
  
 //print price of the soapUI   
 List<API> apiList = getCourse.getCourses().getApi();  
 for(int i=0;i<apiList.size();i++)  
 {  
 if(apiList.get(i).getCourseTitle().equalsIgnoreCase("SoapUI Webservices testing"))  
 {  
 System.*out*.println(apiList.get(i).getPrice());  
 }  
 }  
 //print price of the list WebAutomation  
 List<WebAutomation> webAutomationList = getCourse.getCourses().getWebAutomation();  
 for(int i=0;i<webAutomationList.size();i++)  
 {  
 System.*out*.println("title: "+webAutomationList.get(i).getCourseTitle());  
 }  
  
  
  
 }  
}

**Google Add place API – Add input in Post parameters using POJO classes**

Input:

{

"accuracy": 50,

"name": "Indore",

"phone\_number": "11-22-3333",

"address": "Sudama Nagar",

"types": [

"ShoePark",

"Garden"

],

"website": null,

"language": "Eng",

"location": {

"lat": 33.3322,

"lng": 44.4224

}

}

POJO classes:

package pojo;

import java.util.List;

public class AddPlace {

private int accuracy;

private String name;

private String phone\_number;

private String address;

private List<String> types;

private String website;

private String language;

private Location location;

public int getAccuracy() {

return accuracy;

}

public void setAccuracy(int accuracy) {

this.accuracy = accuracy;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getPhone\_number() {

return phone\_number;

}

public void setPhone\_number(String phone\_number) {

this.phone\_number = phone\_number;

}

public String getAddress() {

return address;

}

public void setAddress(String address) {

this.address = address;

}

public List<String> getTypes() {

return types;

}

public void setTypes(List<String> types) {

this.types = types;

}

public String getWebsite() {

return website;

}

public void setWebsite(String website) {

this.website = website;

}

public String getLanguage() {

return language;

}

public void setLanguage(String language) {

this.language = language;

}

public Location getLocation() {

return location;

}

public void setLocation(Location location) {

this.location = location;

}

}

Sub-json pojo class

package pojo;

public class Location {

private double lat;

private double lng;

public double getLat() {

return lat;

}

public void setLat(double lat) {

this.lat = lat;

}

public double getLng() {

return lng;

}

public void setLng(double lng) {

this.lng = lng;

}

}

**Code**

package codes;

import io.restassured.RestAssured;

import pojo.AddPlace;

import pojo.Location;

import java.util.ArrayList;

import java.util.List;

import static io.restassured.RestAssured.\*; //used for the given() method

public class SerializeTest {

public static void main(String[] args) {

//Add list of array for set types

List<String> listTypes = new ArrayList<>();

listTypes.add("ShoePark");

listTypes.add("Garden");

//Add location subjson

Location location = new Location();

location.setLat(33.3322);

location.setLng(44.4224);

//set all the json parameters

AddPlace addPlace = new AddPlace();

addPlace.setAccuracy(50);

addPlace.setAddress("Sudama Nagar");

addPlace.setLanguage("Eng");

addPlace.setName("Indore");

addPlace.setPhone\_number("11-22-3333");

addPlace.setTypes(listTypes);

addPlace.setLocation(location);

RestAssured.baseURI = "https://rahulshettyacademy.com";

String response = given()

.log().all()

.queryParam("key","qaclick123")

.body(addPlace)

.when()

.post("maps/api/place/add/json")

.then()

.assertThat().statusCode(200)

.extract()

.response().asString();

System.out.println("Response : "+response);

}

}

**Significance of Spec Builders in Rest**

Request and Response Spec Builders :

ADD PLACE :

RestAssured.*baseURI*="XXXX";

Response res=*given*().queryParam("key", "qaclick123").header("Content-Type","application/json")

.body(add\_place\_json)

.when().post("/maps/api/place/add/json").

then().assertThat().statusCode(200). contentType("application/json")

extract().response();

GET\_PLACE

RestAssured.*baseURI*="XXXX";

Response res=*given*().queryParam("key", "qaclick123").header("Content-Type","application/json")

when (). get("/maps/api/place/get/json").

then().assertThat().statusCode(200).contentType("application/json").extract().response();

DELETE\_PLACE

RestAssured.*baseURI*="XXXX";

Response res=*given*().queryParam("key", "qaclick123").header("Content-Type","application/json")

.body(“delete\_Place\_json”)

.when().post("/maps/api/place/delete/json").

then().assertThat().statusCode(200). contentType("application/json").extract().response();

Build -Request Spec Builder-

req= **new** RequestSpecBuilder().setContentType(ContentType.***JSON***)

.setBaseUri("XXXX")

.addQueryParam("key","qaclick123")

.build();

*given*().spec (req ).body(add\_place\_json) .post(“/maps/api/place/add/json).

Build Response Spec Builder:

**res = new** ResponseSpecBuilder().expectStatusCode(200).expectContentType(ContentType.***JSON***). build();

then().spec(re).extract().response();

Rewriting Test with Request and Response Spec Builder :

*given*().spec(req).body(add\_place\_json).post(“/maps/api/place/add/json).

then().spec(res).extract().response();

**Spec(): is used to provide all the details of the api method which is used in all the methods of APIs like common query parameter, baseURI etc.**

Req = new RequestSpecBuilder().serContentType(ContentType.JSON)

.setBaseUri(“”)

.addQueryParam(“”,””)

.build();

*given*().spec(Req).body(Payload.*addPlace*())  
.when().post("maps/api/place/add/json")

.then().spec(Req).extract().response

Example:

package codes;

import files.Payload;

import io.restassured.builder.RequestSpecBuilder;

import io.restassured.builder.ResponseSpecBuilder;

import io.restassured.http.ContentType;

import io.restassured.response.Response;

import io.restassured.specification.RequestSpecification;

import io.restassured.specification.ResponseSpecification;

import static io.restassured.RestAssured.given;

public class SpecBuilderCase {

public static void main(String[] args) {

//create the request specification for request

RequestSpecification requestSpecification = new RequestSpecBuilder().setBaseUri("https://www.rahulshettyacademy.com") //uri

.addQueryParam("key ", "qaclick123") //query param

.setContentType(ContentType.JSON) //header

.build();

//create the Response specification for response

ResponseSpecification responseSpecification = new ResponseSpecBuilder().expectStatusCode(200)

.expectContentType(ContentType.JSON).build(); //build to whole response

RequestSpecification addRequestSpecification = given().spec(requestSpecification)

.body(Payload.addPlace());

Response response =addRequestSpecification.when().post("maps/api/place/add/json") //resource name

.then().spec(responseSpecification).extract().response();//used to extract body

String responseAsString = response.asString();

System.out.println("responseAsString: "+responseAsString);

}

}

**Maven**

**Apache Maven** is the build management and project management tool or software for the Java framework.

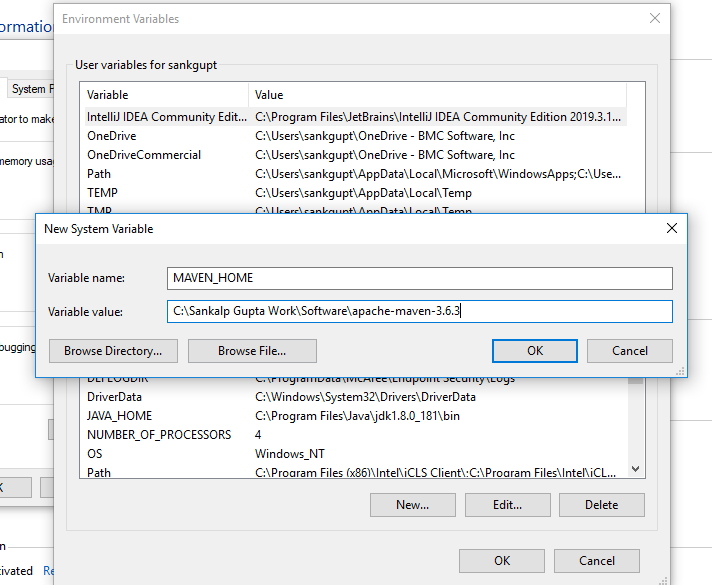
**1.Why Maven?**

* Center repository to get dependencies – you get all the jars in maven repositories
* Maintaining common structure across organization – maintain and provide templates like src>>Test or src>>java
* Flexibility in integration with CI tool – Jenkins integration to run multiple test cases
* Plugins for Test framework execution – support plugins

**2. Install Maven** – from [https://maven.apache.org/download.cgi - apache-maven-3.6.3-bin.zip](https://maven.apache.org/download.cgi%20-%20apache-maven-3.6.3-bin.zip)

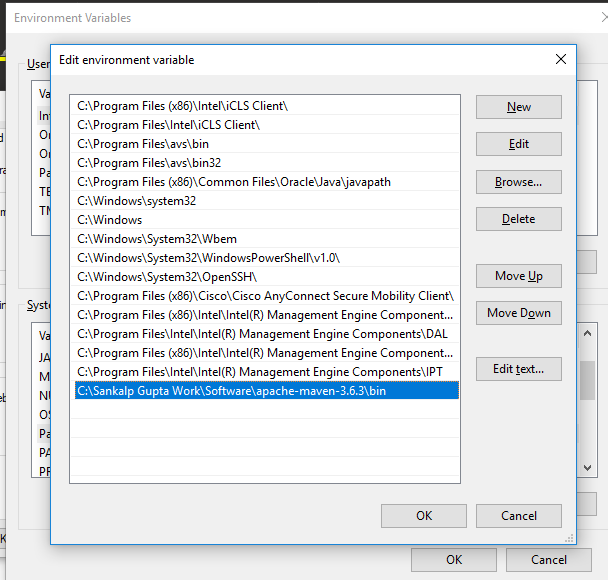
**3. System variable setting:**

Copy the path as per image and add it on env variable>>system variable -- copy C:\Sankalp Gupta Work\Software\apache-maven-3.6.3



Also add path variable -- env variable>>system variable>>click on path variable – copy till bin folder

C:\Sankalp Gupta Work\Software\apache-maven-3.6.3\bin



Run on cmd

C:\Users\sankgupt>mvn --version

Apache Maven 3.6.3 (cecedd343002696d0abb50b32b541b8a6ba2883f)

Maven home: C:\Sankalp Gupta Work\Software\apache-maven-3.6.3\bin\..

Java version: 1.8.0\_181, vendor: Oracle Corporation, runtime: C:\Program Files\Java\jdk1.8.0\_181\jre

Default locale: en\_US, platform encoding: Cp1252

OS name: "windows 10", version: "10.0", arch: "amd64", family: "windows"

**4.Under the maven terminology**

**Group id:** every jar and feature have a unique id that is called group id

<groupId>io.qameta.allure</groupId>

**Artifact id:** it is the sub details like selenium and I want java jars. So It is basically details of sub projects

<artifactId>allure-testng</artifactId>

Version: version needed

Architype.genertate : Create a new project in maven, provide simple skelton

my-app

|-- pom.xml

`-- src

|-- main

| `-- java

| `-- com

| `-- mycompany

| `-- app

| `-- App.java

`-- test

`-- java

`-- com

`-- mycompany

`-- app

`-- AppTest.java

5.Create Maven project

**For IntelliJ**

C:\Users\sankgupt>mvn archetype:generate -DgroupId=qaClickAcademy -DartifactId=MavenJava -DarchetypeartifactId=maven-archetype-quickstart -DinterectiveMode=false

-D for the parameter

-DarchetypeartifactId=maven-archetype-quickstart : for the test template of maven. Quickstart is the template which is used highly

-DgroupId=qaClickAcademy : Main project name

-DartifactId=MavenJava: Name from which project saved or workspace

Artifactid is the project name

After completion is output like:

Confirm properties configuration:

groupId: qaClickAcademy

artifactId: MavenJava

version: 1

package: qaClickAcademy

Y: : y

[INFO] ----------------------------------------------------------------------------

[INFO] Using following parameters for creating project from Archetype: maven-archetype-quickstart:1.4

[INFO] ----------------------------------------------------------------------------

[INFO] Parameter: groupId, Value: qaClickAcademy

[INFO] Parameter: artifactId, Value: MavenJava

[INFO] Parameter: version, Value: 1

[INFO] Parameter: package, Value: qaClickAcademy

[INFO] Parameter: packageInPathFormat, Value: qaClickAcademy

[INFO] Parameter: version, Value: 1

[INFO] Parameter: package, Value: qaClickAcademy

[INFO] Parameter: groupId, Value: qaClickAcademy

[INFO] Parameter: artifactId, Value: MavenJava

[INFO] Project created from Archetype in dir: C:\Users\sankgupt\MavenJava

[INFO] ------------------------------------------------------------------------

[INFO] BUILD SUCCESS

[INFO] ------------------------------------------------------------------------

[INFO] Total time: 01:50 min

[INFO] Finished at: 2020-10-28T14:54:34+05:30

[INFO] ------------------------------------------------------------------------

**6. Integrate maven with Eclipse**

C:\Users\sankgupt>mvn archetype:generate -DgroupId=qaClickAcademy -DartifactId=MavenJava -DarchetypeartifactId=maven-archetype-quickstart -DinterectiveMode=false

**But for eclipse we need .class file for project combability**

1. Goto project where you POM.xml lies - C:\Users\sankgupt\MavenJava
2. Go the tat path in cmd : C:\Users\sankgupt\MavenJava
3. Write command mvn ***eclipse:eclipse –*** *on doing this .classpath and .project file are shown there*

**CUCUMBER**

**What is framework?**

A framework is execution system where we run and developed out tests or testcases.

**Cucumber**: is the BDD framework for running automation tests. Cucumber does not automate testcases.

**When my test is already automated and run, then what does cucumber does?**

Data driven, parameterization, execution control, hooks[group], reports, automation utility and more...

**When you say automated test, which type?**

Any test (Web, mobile, Unit, API) which is return in java/ruby by cucumber.

**How cucumber is best in market in compare with other framework?**

Because test cases are defined with BDD methodology.

No coding is required to implement functionality

**Json and Json Array – Nested**

Json used in this Section with Queries to solve

{

"dashboard": {

"purchaseAmount": 910,

"website": "rahulshettyacademy.com"

},

"courses": [

{

"title": "Selenium Python",

"price": 50,

"copies": 6

},

{

"title": "Cypress",

"price": 40,

"copies": 4

},

{

"title": "RPA",

"price": 45,

"copies": 10

}

]

}

1. **Print No of courses returned by API**

//Size - Print No of courses returned by API - () is optional

//int noOfCourses = jsonPath.getInt("courses.size");

int noOfCourses = jsonPath.getInt("courses.size()");

System.out.println("No of courses: "+noOfCourses);

**2.Print Purchase Amount**

//purchaseAmount parent is Dashboard so dashboard.purchaseAmount

int purchaseAmount = jsonPath.getInt("dashboard.purchaseAmount");

System.out.println("purchase Amount: "+purchaseAmount);

**3. Print Title of the first course**

//3. Print Title of the first course, as course is array so we need to

//use courses.title[0] as it is on zeroth position

// String firstTitle = jsonPath.get("courses[0].title"); //both are correct

String firstTitle = jsonPath.get("courses.title[0]");

System.out.println("Title of the first course: "+firstTitle);

1. **Print All course titles and their respective Prices**

//4. Print All course titles and their respective Prices

for (int i=0;i<noOfCourses;i++)

{

//String title = jsonPath.get("courses["+i+"].title"); //both are correct

String title = jsonPath.get("courses.title["+i+"]");

System.out.println("Title of the ["+i+"] course: "+title);

}

1. **Print no of copies sold by RPA Course**

//5. Print no of copies sold by RPA Course

int priceRPA = jsonPath.get("courses[2].price");

System.out.println("Print no of copies sold by RPA Course: "+priceRPA);

//if you don't know index

for (int i=0;i<noOfCourses;i++)

{

String title = jsonPath.get("courses.title["+i+"]");

if(title.equals("RPA")) {

int priceRPA1 = jsonPath.get("courses["+i+"].price");

System.out.println("Print no of copies sold by RPA Course: " + priceRPA1);

}

}

1. **Verify if Sum of all Course prices matches with Purchase Amount**

//6. Verify if Sum of all Course prices matches with Purchase Amount

int sumPrice = 0;

for (int i=0;i<noOfCourses;i++)

{

int price = jsonPath.get("courses["+i+"].price");

sumPrice = sumPrice+price;

}

System.out.println(sumPrice);

Assert.assertEquals(sumPrice,purchaseAmount,"sumPrice and purchaseAmount are not equal");

}

Go to Editor and if you get [] – then it is an array

"courses": [ //ARRAY

{ //index 0

"title": "Selenium Python",

"price": 50,

"copies": 6

},

{ //index 1

"title": "Cypress",

"price": 40,

"copies": 4

},

{ //index 2

"title": "RPA",

"price": 45,

"copies": 10

}

]

}

**More practice**

[

{

"id": "0001",

"type": "donut",

"name": "Cake",

"ppu": 0.55,

"batters":

{

"batter":

[

{ "id": "1001", "type": "Regular" },

{ "id": "1002", "type": "Chocolate" },

{ "id": "1003", "type": "Blueberry" },

{ "id": "1004", "type": "Devil's Food" }

]

},

"topping":

[

{ "id": "5001", "type": "None" },

{ "id": "5002", "type": "Glazed" },

{ "id": "5005", "type": "Sugar" },

{ "id": "5007", "type": "Powdered Sugar" },

{ "id": "5006", "type": "Chocolate with Sprinkles" },

{ "id": "5003", "type": "Chocolate" },

{ "id": "5004", "type": "Maple" }

]

},

{

"id": "0002",

"type": "donut",

"name": "Raised",

"ppu": 0.55,

"batters":

{

"batter":

[

{ "id": "1001", "type": "Regular" }

]

},

"topping":

[

{ "id": "5001", "type": "None" },

{ "id": "5002", "type": "Glazed" },

{ "id": "5005", "type": "Sugar" },

{ "id": "5003", "type": "Chocolate" },

{ "id": "5004", "type": "Maple" }

]

},

{

"id": "0003",

"type": "donut",

"name": "Old Fashioned",

"ppu": 0.55,

"batters":

{

"batter":

[

{ "id": "1001", "type": "Regular" },

{ "id": "1002", "type": "Chocolate" }

]

},

"topping":

[

{ "id": "5001", "type": "None" },

{ "id": "5002", "type": "Glazed" },

{ "id": "5003", "type": "Chocolate" },

{ "id": "5004", "type": "Maple" }

]

}

]

Questions

1. **Select all the batter of the food whose name is Old Fashioned**

//You can use any name like data.size for size or directly pass size

//if there is no parent element

int noMainArrayElements = jsonpath.getInt("data.size");

//int noMainArrayElements = jsonpath.getInt("size");

System.out.println("parent element: " +noMainArrayElements);

//Select all the batter of the food whose name is Old Fashioned

for (int i=0;i<noMainArrayElements;i++)

{

String nameOfFood = jsonpath.get("["+i+"].name");

System.out.println("name Of Food: "+nameOfFood);

if(nameOfFood.equals("Old Fashioned"))

{

int noBatterElements = jsonpath.getInt("["+i+"].batters.batter.size()"); //"["+i+"] for the array of parent

System.out.println("no Batter Elements: "+noBatterElements);

for ( int j=0;j<noBatterElements;j++) {

//["+i+"] = parent array and batters.batter[" + j + "].type" type of selected array named cake

String nameOfBatterFood = jsonpath.get("["+i+"].batters.batter[" + j + "].type");

System.out.println("name Of Batter Food: "+nameOfBatterFood);

}

break; // if value matched out of the loop

}

}

**2. Select all the Toppings ID and type of the food whose name is Cake**

//2. Select all the Toppings ID and type of the food whose name is Cake

for (int i=0;i<noMainArrayElements;i++)

{

String nameOfFood = jsonpath.get("["+i+"].name");

System.out.println("name Of Food: "+nameOfFood);

if(nameOfFood.equals("Cake"))

{

int noTopingElements = jsonpath.getInt("["+i+"].topping.size()"); //"["+i+"] for the array of parent

System.out.println("no Batter Elements: "+noTopingElements);

for ( int j=0;j<noTopingElements;j++) {

//["+i+"] = parent array and batters.batter[" + j + "].type" type of selected array named cake

String nameOfBatterFood = jsonpath.get("["+i+"].topping[" + j + "].type");

String id = jsonpath.get("["+i+"].topping[" + j + "].id");

HashMap<String,String> hashMap = new HashMap<>();

hashMap.put(id,nameOfBatterFood);

//System.out.println(hashMap); //Printing hashmap first way

for (Map.Entry entry: hashMap.entrySet()) //second way using map.entry

{

System.out.println("id: "+entry.getKey()+" - Type: "+entry.getValue());

}

}

break; // if value matched out of the loop

}

}

Takeaway:

1. **Size or Size() is used for the to get the size of json array**

int noMainArrayElements = jsonpath.getInt("data.size");

//int noMainArrayElements = jsonpath.getInt("size");

System.out.println("parent element: " +noMainArrayElements);

1. **If there is no parent element name then you can use any name or directly use the size**

int noMainArrayElements = jsonpath.getInt("data.size");

//int noMainArrayElements = jsonpath.getInt("size");

System.out.println("parent element: " +noMainArrayElements);