Rest Assured

It is java framework. It is use to test validate http methods.it is built on top of htttp builder

Focus on testing with BDD specification

Groovy implementation – no need groovy configuration its inbuilt standard dependcies

RestAssured : Rest api automation fw which internally uses httpclient builder but

It has more easy to use methods than HttpClient.

It has BDD style methods which makes code more readable and easy to understand.

It has easy json parsing methods which can be used with help of

Hamcrest matchers.

httpClient needs external conversion:

When req data is complex- they try to create java POJO entity class for reqdata.

Using Jackson jars we can convert a JSON object to JAVA pojo - Marshalling

JavaPojo to JSON Object – Unmarshalling

RestAssured internally it uses Jackson when we pass java pojos

Version jars----- hamcrest matches 1.3, previously it was jayway.restassured

inbuilt json parsing

httpClient needs external conversion:

Get(“/gomathy”);

Post(“/gomathy”);

Put((“/gomathy”);

Delete((“/gomathy”);

Get Method -----view resource

String uri=”http.www.gmail.com”;

///RestAssured.when().get(uri); ---interface class

Response response =given().when().get(uri).then().statuscode(200);

System.out.println(response.asString());

Different ways we can assert given below

Given().when().get(uri).then().body("login",equalTo("whatever"));

Given().when().get(uri).then().body(containsString("avatar\_url"));

Given().when().get(uri).then().body("site\_admin",equalTo(false));

Assert.assertEquals(response, "Oslo");

**Given() Function** –header information will goes here like contenttype, param,header.

Given function goes-*----given*()

.header("Author","Gomathy")

.contentType(ContentType.*TEXT*)

.param("location", "New York, New York")

.get(uri)

.then()

.assertThat()

.header("Content-Type", "application/json; charset=utf-8");

//Fetching response in JSON array [ ]

JSONArray jsonResponse = **new** JSONArray(response.asString());

System.*out*.println(jsonResponse);

//Fetching value jsonobject{ }

String capital =jsonResponse.getJSONObject(0).getString("capital");

Assert.assertEquals(capital, "Oslo");

Post method ---- create resource

Response response =given().when().body(“{\”messages\’:\”helloworld\”}”).post(uri);

Response response =given().when().body(“{\”messages\’:\”helloworld\”}”).post(uri).then().statuscode(201);

Assert.assertEquals(response.getstatuscode(),201);

System.out.pritnln(“status cose sre”+response.getstatuscode());

System.out.pritnln(“status code sre”+response.asString());

(or)

String uri = "http://jsonplaceholder.typicode.com/posts/1/comments";

String Body =“{\”messages\”:\”helloworld\”}”; //key-value pairs

// Building request using requestSpecBuilder

RequestSpecBuilder builder = new RequestSpecBuilder();

//Setting API's body

builder.setBody(Body);

builder.setContentType("application/json");

RequestSpecification requestSpec = builder.build();

Response response = given().when().body(Body).post(uri);

System.out.println(response);

//Setting API's body another type using map set enittiy

Map<string,string>body= new HashMap<string,string>();

Body.put(“messages”,”helloworld”);

Body.put(“place”,”Chennai”);

Response response = given().when().body(Body).post(uri).then().statuscode(201);

System.out.println(response);

Request Body

Given().body(“{\”messages\’:\”helloworld\”}”).when().post(uri); /// string

Given().body(message,objectmapper.GSON).when().post(URI); // java object-pojo

Put method ----update resource

Response response =given().when().body(“{\”messages\’:\”helloworld\”}”).put(“/posts/1”);

Assert.assertEquals(response.getstatuscode(),200);

Delete Method –delete resource

Response response =given().when().body(“{\”messages\’:\”helloworld\”}”).delete(“/posts/1”);

Assert.assertEquals(response.getstatuscode(),200);

Get methods------ view the data

{

“Id”:1,

“name”:”hello”,

“age”:39,

“place”:”Chennai”

}

Post method

Post method ---- create data

{ (sending the data to server)

“name”;”gomathy,

“age”:29,

“place”:”Chennai”

}

Send it

Will get post response like this below (Hint--- what server will do it will create id for request)

{

“id”:2,

“name”;”gomathy,

“age”:29,

“place”:”Chennai”

}

Once got response again double check by

Sending get request will get response like this

{

“Id”:1,

“name”:”hello”,

“age”:39,

“place”:”Chennai”

}

{

“id”:2,

“name”;”gomathy,

“age”:29,

“place”:”Chennai”

}

**Security Testing or Authentication**

REST assured also supports some authentication schemes, for example basic authentication:

given().auth().basic("username", "password").when().get("/secured/hello").then().statusCode(200);

<https://console.developers.google.com/apis/credentials?project=instant-voyager-164920>

https://maps.googleapis.com/maps/api/geocode/json?address=6500 Dublin Blvd,Dublin,CA&key=AIzaSyB7nX8gW\_l9hVW78Y8BwV7ZOXyhfqJYxW0