3.4 Send XML Payload in REST Assured,  
Parse XML Response in REST Assured,   
Parse JSON Response in REST Assured,   
and Explore Native Logging of REST Assured



This section will guide you to understand:

* How to create and send XML Payload using REST Assured
* How to parse XML Response using REST Assured
* How to parse JSON Response using REST Assured
* How to explore Native Logging of REST Assured

**Development Environment:**

* Java 1.8
* Eclipse
* TestNG
* Maven

This guide has six subsections, namely:

3.4.1 Creating a Maven project in Eclipse

3.4.2 Sending XML Payload in REST Assured

3.4.3 Parsing XML Response in REST Assured

3.4.4 Parsing JSON Response in REST Assured

3.4.5 Exploring Native Logging of REST Assured

3.4.6 Pushing the code to GitHub repositories

**Step 3.4.1:** Creating a Maven project in Eclipse

* Open Eclipse.
* Click on File---> click on New--->Project.
* Select the Maven project and click on Next.
* Enter the Group id, Artifact id, and click on Finish.
* Right click on Project---> src/test/java---> Package.
* Enter the package name and click Finish.
* Right click on Package---> New---> Class.
* Enter the class name and click on Finish.
* Add dependencies to pom.xml file.

**Step 3.4.2:** Sending XML Payload in REST Assured

* Write the program for XML Payload in REST Assured and click on Save.

**package** Response;

import static io.restassured.RestAssured.given;

import org.hamcrest.Matchers;

import org.testng.Assert;

import org.testng.annotations.Test;

import io.restassured.http.ContentType;

import io.restassured.response.Response;

public **class** XMLResponse

{

@Test

public void post\_xml\_test() {

Response response = given().contentType(ContentType.XML).accept(ContentType.XML).body("<Employee>" +

"<empName>Lavanya Gowda</empName>" + "<empAddress>abc</empAddress>"

+ "<mobileNumber>1592211560</mobileNumber>" + "<department>abc</department>" + "<project>abc</project>"

+ "<teamLead>abc</teamLead>" + "<salary>10000</salary>" + "<joiningDate>11-10-19</joiningDate>"

+ "</Employee>").when().post("http://192.168.1.207:8080/api/employee/add/xml");

System.out.println("POST Response\n" + response.asString());

// tests

int statusCode = response.getStatusCode();

System.out.println("Status code is:" + statusCode);

Assert.assertEquals(200, statusCode);

}

}

* Click on run and check the output in TestNG

[RemoteTestNG] detected TestNG version 6.14.3

POST Response

78

Status code is:200

PASSED: post\_xml\_test

===============================================

Default test

Tests run: 1, Failures: 0, Skips: 0

===============================================

===============================================

Default suite

Total tests run: 1, Failures: 0, Skips: 0

===============================================

**Step 3.4.3:** Parsing XML Response in REST Assured

* Write the program for XML Response in REST Assured and click on Save.

**package** Response;

import org.testng.annotations.Test;

import org.testng.Assert;

import io.restassured.RestAssured;

import io.restassured.http.ContentType;

import io.restassured.response.Response;

import static io.restassured.RestAssured.\*;

public **class** ParseXML

{

@Test

public void parse\_xml\_test() {

Response response = given().contentType(ContentType.XML).accept(ContentType.XML).body("<Employee>" +

"<empName>Lavanya Gowda</empName>" + "<empAddress>abc</empAddress>"

+ "<mobileNumber>1591111560</mobileNumber>" + "<department>abc</department>" + "<project>abc</project>"

+ "<teamLead>abc</teamLead>" + "<salary>10000</salary>" + "<joiningDate>11-10-19</joiningDate>"

+ "</Employee>").when().post("http://192.168.1.207:8080/api/employee/add/xml");

System.out.println("POST Response\n" + response.asString());

// tests

int statusCode = response.getStatusCode();

System.out.println("Status code is:" + statusCode);

Assert.assertEquals(200, statusCode);

}

}

* Click on run and check the output in TestNG.

[RemoteTestNG] detected TestNG version 6.14.3

POST Response

85

Status code is:200

PASSED: parse\_xml\_test

===============================================

Default test

Tests run: 1, Failures: 0, Skips: 0

===============================================

===============================================

Default suite

Total tests run: 1, Failures: 0, Skips: 0

**Step 3.4.4:** Parsing JSON Response in REST Assured

Write the program for Parse JSON Response in REST Assured and click on Save.

**package** Response;

import static org.testng.Assert.assertEquals;

import java.util.List;

import org.json.simple.JSONObject;

import org.testng.annotations.Test;

import io.restassured.RestAssured;

import io.restassured.http.Method;

import io.restassured.path.json.JsonPath;

import io.restassured.response.Response;

import io.restassured.specification.RequestSpecification;

import junit.framework.Assert;

public **class** JsonParse {

    @Test

    public void testJsonParsing() {

        //specify base URI

RestAssured.baseURI="http://192.168.1.207:8080/api/employee/";

//Request object

RequestSpecification httpRequest=RestAssured.given();

//contains the information in the json format

//Request payload sending along with post request

JSONObject requestParams=new JSONObject();

requestParams.put("empName","Lavanya");

requestParams.put("empAddress","Bomanahalli");

requestParams.put("mobileNumber","9109320002");

requestParams.put("department","testing");

requestParams.put("teamLead","Aruna");

requestParams.put("salary","10000");

requestParams.put("joiningDate","14-05-19");

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

httpRequest.body(requestParams.toJSONString());//attach above data to the request

//Response object

Response response=httpRequest.request(Method.POST,"/add");

//print response in console window

String responseBody=response.getBody().asString();

System.out.println("Response Body is:" +responseBody);

//status code validation

int statusCode=response.getStatusCode();

System.out.println("Status code is:" +statusCode);

Assert.assertEquals(200,statusCode);

}

}

* Click on run and check the output in TestNG.

[RemoteTestNG] detected TestNG version 6.14.3

Response Body is:81

Status code is:200

PASSED: testJsonParsing

===============================================

Default test

Tests run: 1, Failures: 0, Skips: 0

===============================================

===============================================

Default suite

Total tests run: 1, Failures: 0, Skips: 0

===============================================

**Step 3.4.5:** Exploring Native Logging of REST Assured

* Write the program for Native Logging of REST Assured and click on Save.

**package** Response;

import org.testng.annotations.Test;

import static io.restassured.RestAssured.given;

import static io.restassured.RestAssured.when;

public **class** Logging

{

@Test

public void testLogging1()

{

given().get("http://192.168.1.207:8080/employee/api/search/8970922880").

then()

//.log().headers();

//.log().body();

//.log().cookies();

.log().all();

}

//Logs only in case of errors

@Test

public void testLogging2()

{

given().

get("http://192.168.1.207:8080/employee/api").

then().

log().ifError();

}

//conditional logging

@Test

public void testLogging3()

{

given().

get("http://192.168.1.207:8080/employee/api/search/8970922880").

then().

log().ifStatusCodeIsEqualTo(200);

}

}

* Click on run and check the output in TestNG.

[RemoteTestNG] detected TestNG version 6.14.3

HTTP/1.1 404

Content-Type: application/hal+json;charset=UTF-8

Transfer-Encoding: chunked

Date: Fri, 11 Oct 2019 07:01:31 GMT

{

"timestamp": "2019-10-11T07:01:31.720+0000",

"status": 404,

"error": "Not Found",

"message": "No message available",

"path": "/employee/api/search/8970922880"

}

HTTP/1.1 404

Content-Type: application/hal+json;charset=UTF-8

Transfer-Encoding: chunked

Date: Fri, 11 Oct 2019 07:01:31 GMT

{

"timestamp": "2019-10-11T07:01:31.961+0000",

"status": 404,

"error": "Not Found",

"message": "No message available",

"path": "/employee/api"

}

PASSED: testLogging1

PASSED: testLogging2

PASSED: testLogging3

===============================================

Default test

Tests run: 3, Failures: 0, Skips: 0

===============================================

===============================================

Default suite

Total tests run: 3, Failures: 0, Skips: 0

===============================================

**Step 3.4.6**: Pushing the code to GitHub repositories

Open your command prompt and navigate to the folder where you have created your files.

cd <folder path>

Initialize your repository using the following command:

git init

Add all the files to your git repository using the following command:

git add . 

Commit the changes using the following command:

git commit . -m “Changes have been committed.”

Push the files to the folder you initially created using the following command:

git push -u origin master