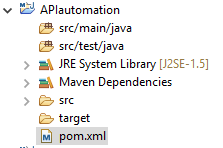
REST Api bdd approach

Requirements:

Java

Eclipse

Create maven project



Add rest assure dependencies in pom.xml:

rest -Assured

Json-scema-validator

Json- path

Xml- path

Testng

Java-hamcrest

Hamcrest -core

Hamcrest-library

1.keywords used in BDD

2.how to send GET Request(WeatherAPI)

<URL:http://restapi.demoqa.com/utilities/weather/city/Hyderabad>

Validations:

Satus code

Status line

Response Body

Header

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

Given()

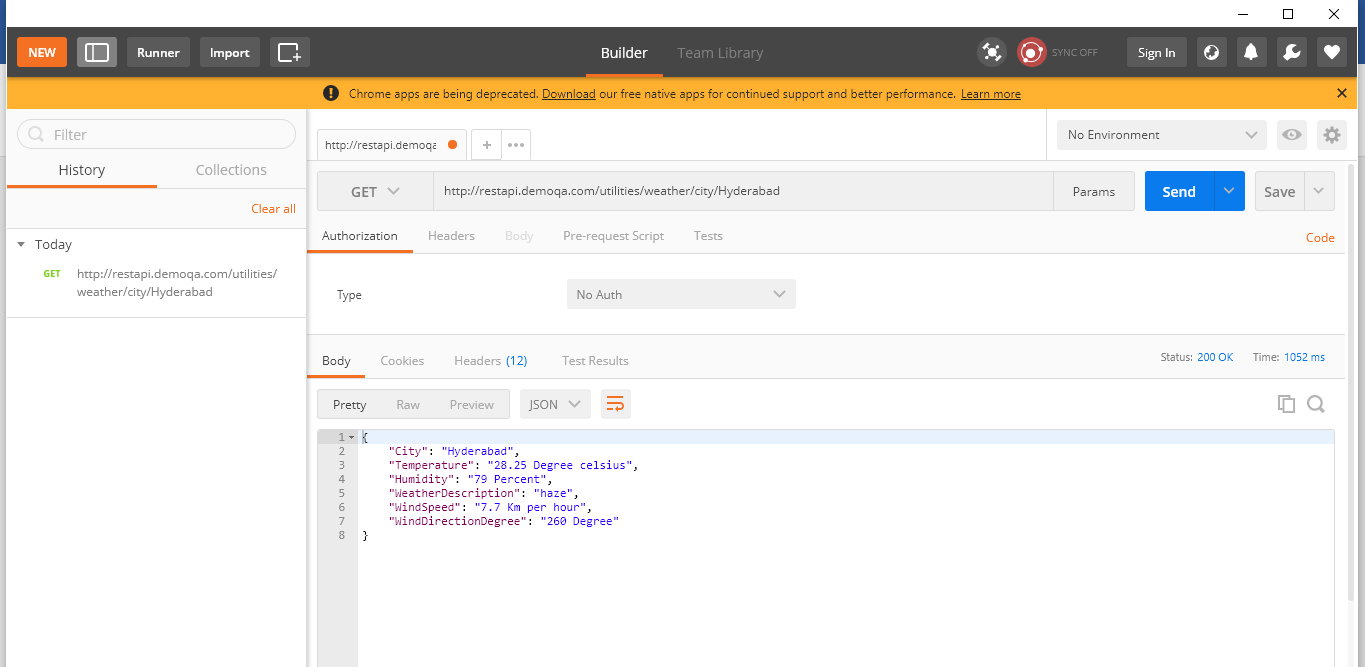
set cookies,add auth,add param, set headers info etc…

When()

get,post,put,deete…

Then ()

Validate status code,extract response,extract headers cookies and response body…..



GET method:

**package** Apitestdemotest;

**import** org.testng.annotations.Test;

**import** **static** io.restassured.RestAssured.\*;

**import** **static** org.hamcrest.Matchers.\*;

**public** **class** GETrequestdemo {

@Test

**public** **void** getweather(){

*given*()

.when()

.get("http://restapi.demoqa.com/utilities/weather/city/Hyderabad")

.then()

.statusCode(200)

.statusLine("HTTP/1.1 200 OK")

.assertThat().body("City", *equalTo*("Hyderabad"))

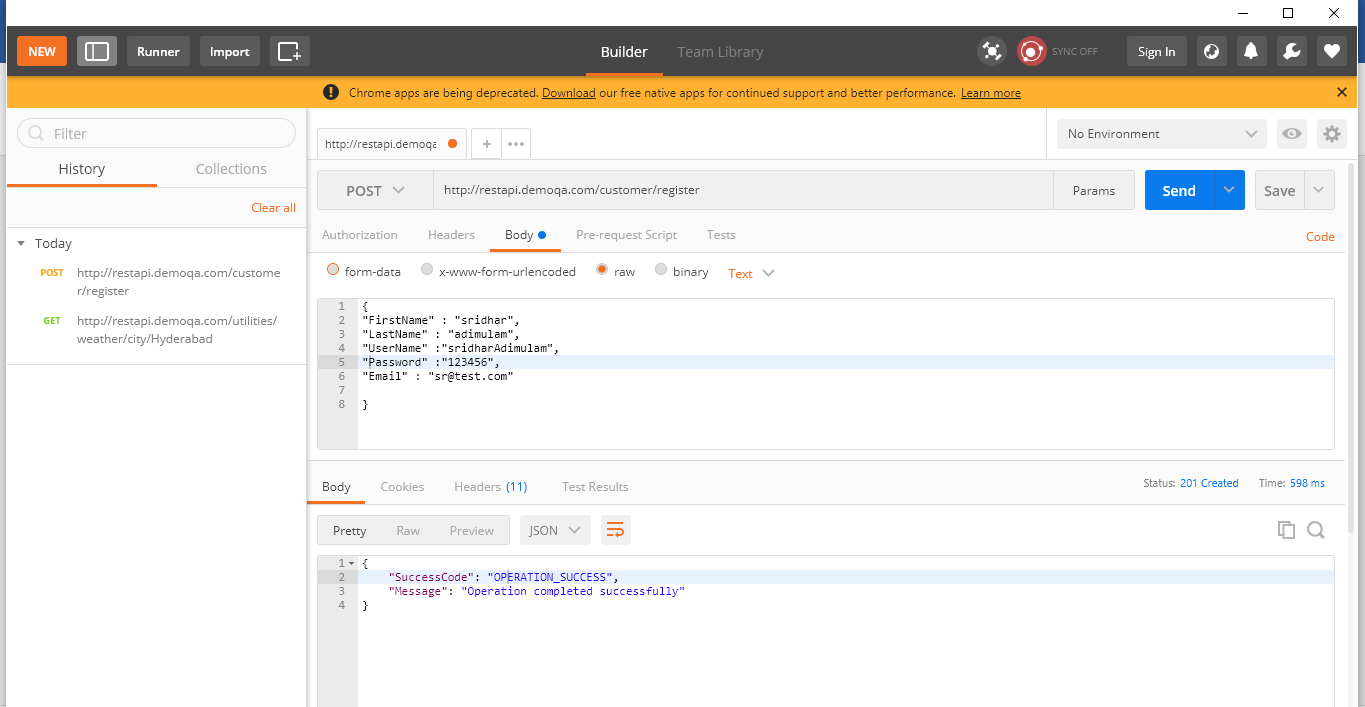
.header("Content-Type","application/json");

}

}

POST method:

URL: http://restapi.demoqa.com/customer/register



RESTutil.java:

**package** Apitestdemotest;

**import** org.apache.commons.lang3.RandomStringUtils;

**public** **class** RESTutils {

**public** **static** String getFirstName(){

String generatedString=RandomStringUtils.*randomAlphabetic*(3);

**return** ("sridhar"+generatedString);

}

**public** **static** String getLastName(){

String generatedString=RandomStringUtils.*randomAlphabetic*(3);

**return** ("Adimulam"+generatedString);

}

**public** **static** String getUserName(){

String generatedString=RandomStringUtils.*randomAlphabetic*(3);

**return** ("sridhar"+generatedString);

}

**public** **static** String getPassword(){

String generatedString=RandomStringUtils.*randomAlphabetic*(3);

**return** ("sridhar"+generatedString);

}

**public** **static** String getEmail(){

String generatedString=RandomStringUtils.*randomAlphabetic*(3);

**return** (generatedString+"gmail.com");

}

**public** **static** String empName(){

String generatedString=RandomStringUtils.*randomAlphabetic*(1);

**return** (generatedString+"gmail.com");

}

**public** **static** String empsal(){

String generatedString=RandomStringUtils.*randomAlphabetic*(5);

**return** (generatedString);

}

}

**package** Apitestdemotest;

**import** java.util.HashMap;

**import** org.testng.annotations.BeforeClass;

**import** org.testng.annotations.Test;

**import** **static** io.restassured.RestAssured.\*;

**import** **static** org.hamcrest.Matchers.\*;

**import** io.restassured.RestAssured;

**public** **class** POSTtest {

**public** **static** HashMap *map*=**new** HashMap();

@BeforeClass

**public** **void** postdata(){

*map*.put("FirstName", RESTutils.*getFirstName*());

*map*.put("LastName", RESTutils.*getLastName*());

*map*.put("UserName", RESTutils.*getFirstName*());

*map*.put("Password", RESTutils.*getFirstName*());

*map*.put("Email", RESTutils.*getFirstName*());

RestAssured.*baseURI*="http://restapi.demoqa.com/customer";

RestAssured.*basePath*="/register";

}

@Test

**public** **void** testpost(){

*given*()

.contentType("application/json")

.body(*map*)

.when()

.post()

.then()

.statusCode(201)

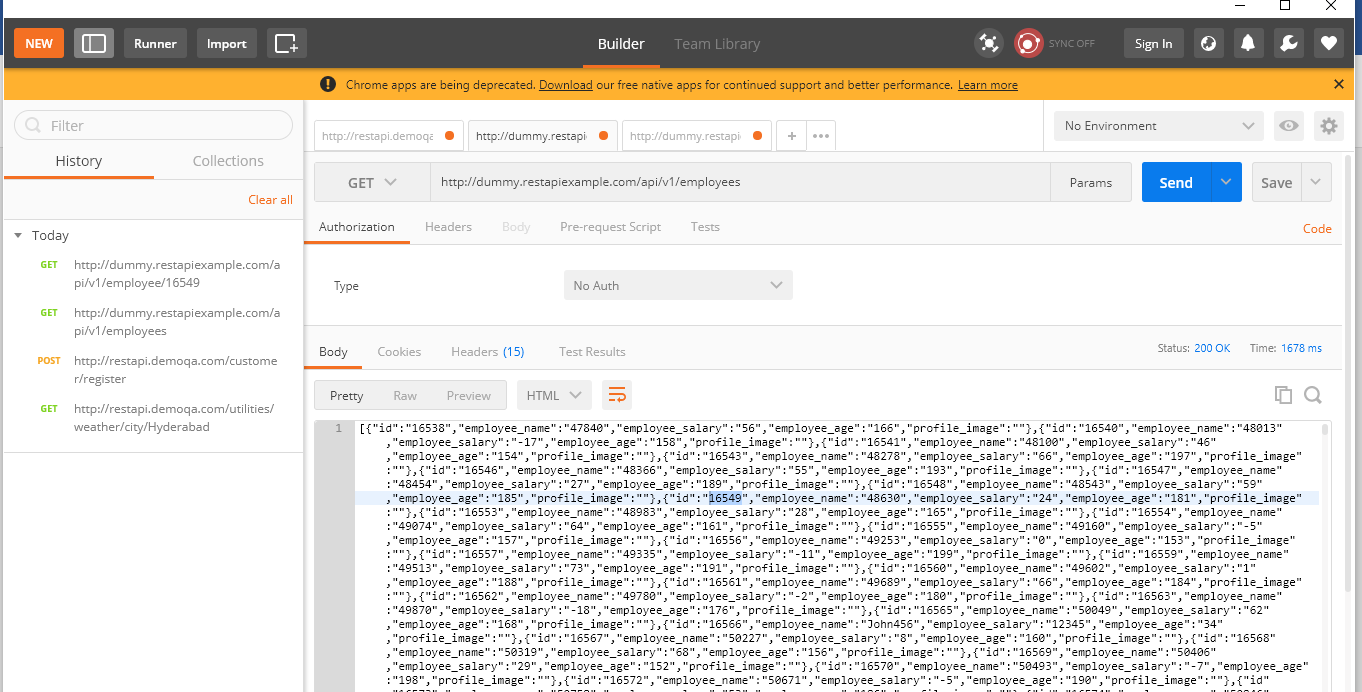
.body("SuccessCode", *equalTo*("OPERATION\_SUCCESS"))

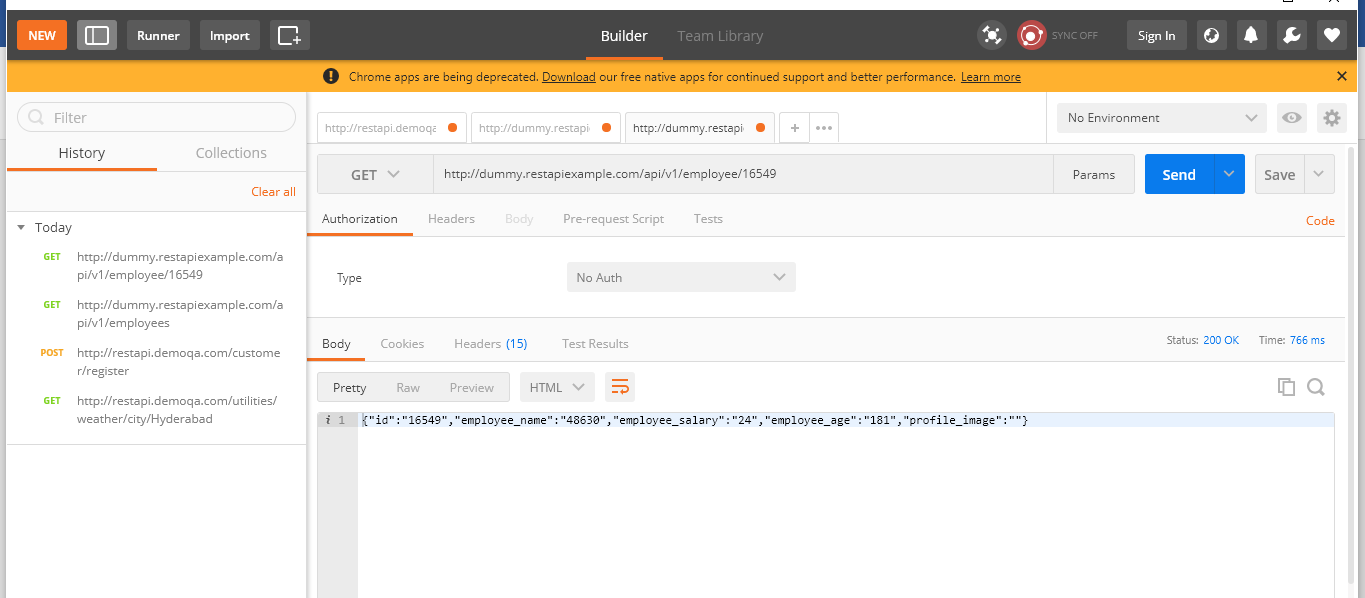
.body("Message",*equalTo*("Operation completed successfully"));

}

}

Get all data :

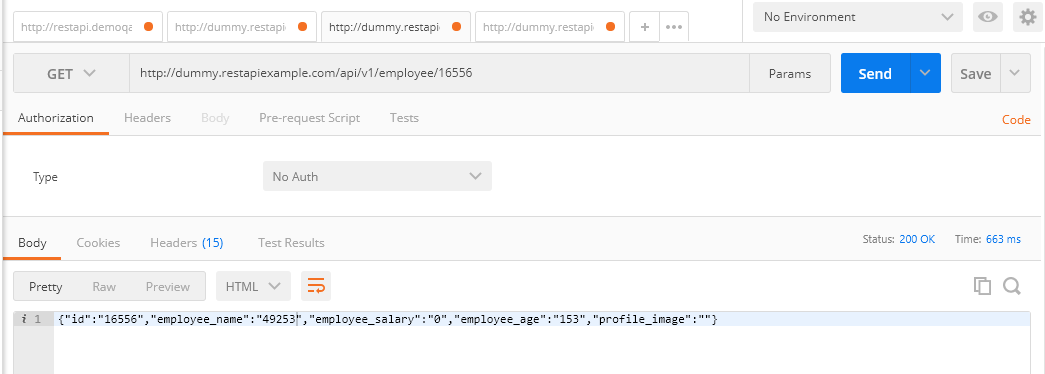


Get the data for one 

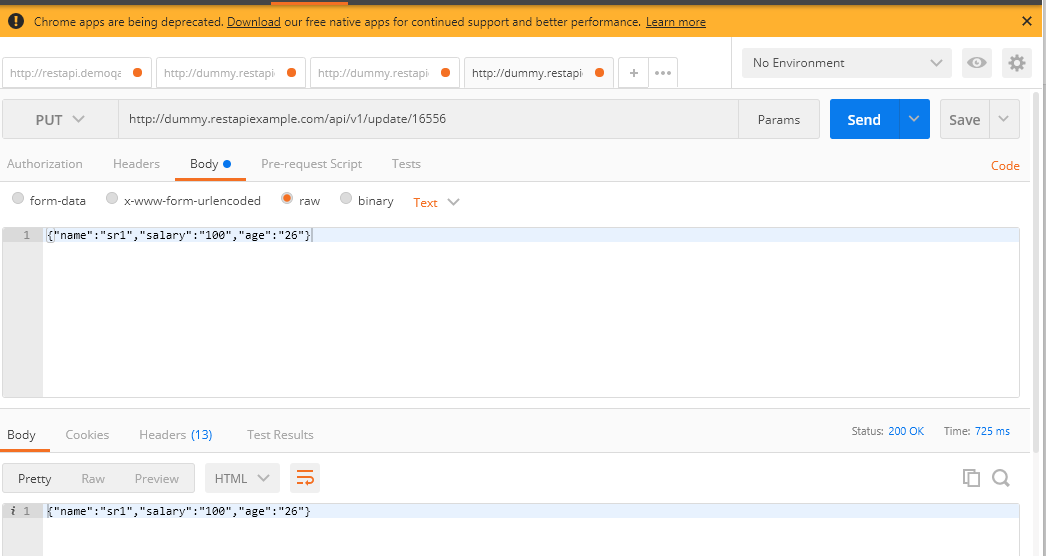
Change the details for a perticular item

Select put method

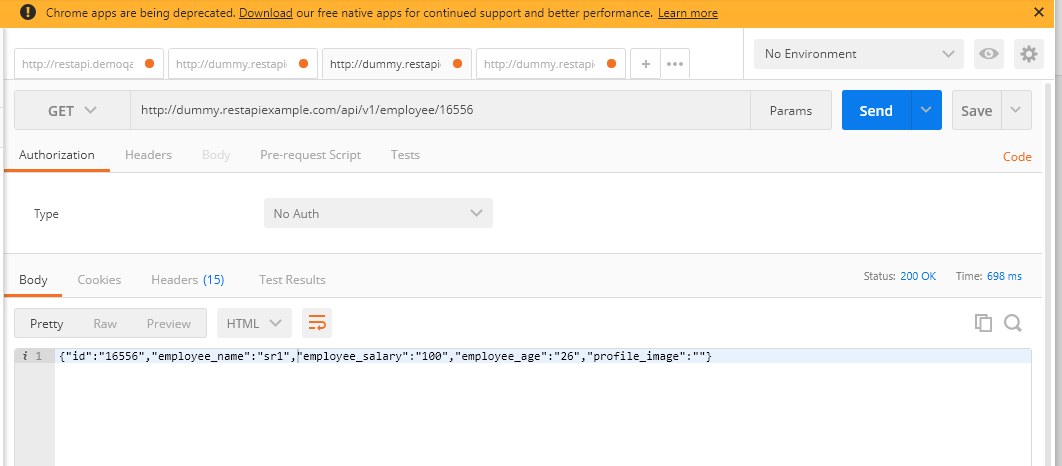
Before put:



Put the values



After sending the put method.the details is updated for the particular item



**package** Apitestdemotest;

**import** java.util.HashMap;

**import** org.testng.annotations.BeforeClass;

**import** org.testng.annotations.Test;

**import** io.restassured.RestAssured;

**import** **static** io.restassured.RestAssured.\*;

**import** **static** org.hamcrest.Matchers.\*;

**public** **class** PUTrequest {

**public** **static** HashMap *map*=**new** HashMap();

String empName=RESTutils.*empName*();

String empAge=RESTutils.*empAge*();

String empSalary=RESTutils.*empsal*();

**int** emp\_id=16561;

@BeforeClass

**public** **void** putdata(){

*map*.put("name",empName);

*map*.put("salary",empSalary);

*map*.put("age",empAge);

RestAssured.*baseURI*="http://dummy.restapiexample.com/api/v1";

RestAssured.*basePath*="/update/"+emp\_id;

}

@Test

**public** **void** testputdata(){

*given*()

.contentType("application/json")

.body(*map*)

.when()

.put()

.then()

.statusCode(200)

.log().all();

}

}

Delete item:

**package** Apitestdemotest;

**import** org.testng.Assert;

**import** org.testng.annotations.Test;

**import** io.restassured.RestAssured;

**import** io.restassured.response.Response;

**import** **static** io.restassured.RestAssured.\*;

**import** **static** org.hamcrest.Matchers.\*;

**public** **class** DeleteitemPUT {

@Test

**public** **void** del(){

RestAssured.*baseURI*="http://dummy.restapiexample.com/api/v1";

RestAssured.*basePath*="/delete/16547";

Response response=

*given*()

.when()

.delete()

.then()

.statusCode(200)

.statusLine("Http/1.1 200 OK")

.log()

.all().extract().response();

String jsonAsString=response.asString();

Assert.*assertEquals*(jsonAsString.contains("sucessfully! deleted Records"),**true**);

}

}

1.Test Status code

2.Log Response

3. verifying single content in response

4.verifying multiple contents in response

5.setting parameters and headers.

**package** Apitestdemotest;

**import** org.testng.annotations.Test;

**import** **static** org.hamcrest.Matchers.\*;

**import** **static** io.restassured.RestAssured.\*;

/\*test stats code

log response

verifying single contents in response body

verify multiple contents in response body

setting parameters and headers.\*/

**public** **class** Basicvalidations\_jsonresponse {

//1 test status code

@Test(priority =1)

**public** **void** teststatuscode(){

*given*()

.when()

.get("http://jsonplaceholder.typicode.com/posts/5")

.then().statusCode(200);

//.log().all();

}

//2.logresponse

@Test(priority=2)

**public** **void** testlogresponse(){

*given*()

.when()

.get("http://jsonplaceholder.typicode.com/posts/5")

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

}

//3.veify single content in response body

@Test(priority=3)

**public** **void** testsinglecontent(){

*given*()

.when()

.get("http://jsonplaceholder.typicode.com/posts/5")

.then()

.statusCode(200).body("title", *equalTo*("nesciunt quas odio"));

//.log().all();

}

//4.veify multiple content in response body

@Test(priority=4)

**public** **void** testmultiplecontent (){

*given*()

.when()

//.get("http://jsonplaceholder.typicode.com/posts/5")

.get("http://services.groupkt.com/country/get/all")

.then()

.statusCode(200)

.body("RestResponse.result.name", *hasItems*("India","Australia","unitedstatesofamerica"));

//.log().all();

}

//5.setting parameters and headers

@Test(priority=5)

**public** **void** testsettingparametersandheaders (){

*given*()

.param("myname","sridhar")

.header("Myheader","Indian")

.when()

//.get("http://jsonplaceholder.typicode.com/posts/5")

.get("http://services.groupkt.com/country/get/all")

.then()

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

}}

1.verifying single content in xml response

2.verifying Multiple contents in xml response

3.verifying all the content in xml response In one go

4.find values using xml path in xml reponse

5.xpath with text() function

**package** Apitestdemotest;

**import** **static** org.hamcrest.Matchers.\*;

**import** org.testng.annotations.Test;

**import** **static** io.restassured.RestAssured.\*;

**public** **class** Basicvalidationsforxml {

/\*

\* 1.verifying single content in xml response

2.verifying Multiple contents in xml response

3.verifying all the content in xml response In one go

4.find values using xml path in xml reponse

5.xpath with text() function

\*/

@Test(priority=1)

**public** **void** testsinglecontent(){

//1.verifying single content in xml response

*given*()

.when()

.get("http://thomas-bayer.com/sqlrest/CUSTOMER/15/")

.then()

.body("CUSTOMER.ID", *equalTo*("15"));

//.log().all();

}

@Test(priority=2)

**public** **void** testmultiplecontent(){

//2.verifying Multiple contents in xml response

*given*()

.when()

.get("http://thomas-bayer.com/sqlrest/CUSTOMER/15/")

.then()

.body("CUSTOMER.ID", *equalTo*("15"))

.body("CUSTOMER.FIRSTNAME", *equalTo*("Bill"))

.body("CUSTOMER.LASTNAME", *equalTo*("Clancy"))

.body("CUSTOMER.STREET", *equalTo*("319 Upland Pl."))

.body("CUSTOMER.CITY", *equalTo*("Seattle"));

//.log().all();

}

@Test(priority=3)

**public** **void** testmultiplecontentsinonego(){

//3.verifying all the content in xml response In one go

*given*()

.when()

.get("http://thomas-bayer.com/sqlrest/CUSTOMER/15/")

.then()

.body("CUSTOMER.text()", *equalTo*("15BillClancy319 Upland Pl.Seattle"));

//.log().all();;

}

@Test(priority=4)

**public** **void** testusingxpath(){

//4.find values using xml path in xml reponse

*given*()

.when()

.get("http://thomas-bayer.com/sqlrest/CUSTOMER/15/")

.then()

.body(*hasXPath*("/CUSTOMER/FIRSTNAME", *containsString*("Bill")));

//.log().all();;

}

@Test(priority=4)

**public** **void** testusingxpath1(){

//4.find values using xml path in xml reponse

*given*()

.when()

.get("http://thomas-bayer.com/sqlrest/INVOICE")

.then()

.body(*hasXPath*("/INVOICEList/INVOICE[text()='30'] "))

.log().all();

}

}

Serialization and deserialization:

Serialization is the process of converting object in to the file and store in to the server or database.

