3.9 Write Log File Using Log4j

This section will guide you to:

* Generate logs in log file using Log4j

**Development Environment:**

* Eclipse IDE
* Java 1.8

This guide has five subsections, namely:

3.9.1 Creating a Maven project

3.9.2 Updating the pom.xml file with the required dependencies

3.9.3 Creating a log4j.properties file for RollingFileAppender

3.9.4 Testing the REST API using REST Assured and log4j

3.9.5 Pushing the code to GitHub repositories

**Step 3.9.1:** Creating a Maven project

* Open Eclipse.
* Click on File---> New--->Project.
* Select the Maven project and click on Next.
* Enter the Group id and Artifact id and click on Finish.

**Step 3.9.2:** Updating the pom.xml file with the required dependencies

* Open the pom.xml file.
* Add the dependencies given below to the pom.xml file:

<dependency>

<groupId>io.rest-assured</groupId>

<artifactId>rest-assured</artifactId>

<version>3.3.0</version>

<scope>test</scope

></dependency>

<dependency>

<groupId>org.testng</groupId>

<artifactId>testng</artifactId>

<version>6.14.3</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>log4j</groupId>

<artifactId>log4j</artifactId>

<version>1.2.17</version>

</dependency>

**Step 3.9.3:** Creating alog4j.properties file for RollingFileAppender

* Right click on Project --> New --> File.
* Name the file as **log4j.properties** and click on Finish.
* Open log4j.properties.
* Write the code given below:

# Root Logger option

log4j.rootLogger=INFO, file, stdout

# Direct log messages to a log file

log4j.appender.**file**=org.apache.log4j.RollingFileAppender

log4j.appender.**file**.**File**=${user.dir}/logs/restAPI.log

log4j.appender.**file**.MaxFileSize=10MB

log4j.appender.**file**.MaxBackupIndex=10

log4j.appender.**file**.layout=org.apache.log4j.PatternLayout

log4j.appender.**file**.layout.ConversionPattern=

%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L %m%n

log4j.appender.**file**.Append=true

**Step 3.9.4:** Testing the REST API using REST Assured and log4j

* Create a package **com.employeeapi.base** inside the src/test/java directory.
* Create a class **TestBase.java** inside the package **com.employeeapi.base.**
* Write the code given below:

1. EmployeesRestAPI is the name given to logger.
2. Log4j.properties is the name of the file we created.

**package** com.employeeapi.base;

**import** org.apache.log4j.Level;

**import** org.apache.log4j.Logger;

**import** org.apache.log4j.PropertyConfigurator;

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

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

**import** io.restassured.specification.RequestSpecification;

**public** **class** TestBase {

**public** **static** RequestSpecification httpRequest;

**public** **static** **Response** response;

**public** **String** empId="55123";

**public** **Logger** logger;

**@BeforeClass**

**public** void setup()

{

logger=**Logger**.getLogger("EmployeesRestAPI");

PropertyConfigurator.configure("Log4j.properties");

logger.setLevel(**Level**.DEBUG);

}

}

* Create a package **com.employeeapi.testcases** inside the src/test/java directory.
* Create a class **GetAllEmployees.java** inside the package **com.employeeapi.testcases.**
* Write the code given below:

**package** com.employeeapi.testcases;

**import** org.testng.Assert;

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

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

**import** com.employeeapi.base.TestBase;

**import** io.restassured.RestAssured;

**import** io.restassured.http.Method;

**import** io.restassured.response.ResponseBody;

**public** **class** GetAllEmployees **extends** TestBase{

**@BeforeClass**

void getAllEmployees() **throws** **InterruptedException** {

logger.info

("\*\*\*\*\*\*\*\*statrt of GetAllEmployees class\*\*\*\*\*\*\*\*\*\*");

RestAssured.baseURI=

"http://192.168.1.207:8080/api/employee/search";

httpRequest = RestAssured.given();

response =

httpRequest.request(**Method**.GET,"/8095393564");

**Thread**.sleep(5000);

}

**@Test**

void checkResponseBody() {

logger.info

("\*\*\*\*\*\*\*Inside checkResponseBody\*\*\*\*\*\*\*");

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

logger.info("Response Body ==> "+responseBody);

Assert.assertTrue(responseBody!=**null**);

}

**@Test**

void checkStatusCode() {

logger.info("\*\*\*\*Inside checkStatusCode\*\*\*\*\*\*\*");

int statusCode=response.getStatusCode();

logger.info("StatusCode ==>"+statusCode);

Assert.assertEquals(statusCode, 200);

}

**@Test**

void checkStatusLine() {

logger.info

("\*\*\*\*\*\*\*\*Inside checkStatusLine\*\*\*\*\*\*\*\*\*");

**String** statusLine=response.getStatusLine();

logger.info("StatusLine ==>"+statusLine);

Assert.assertEquals(statusLine, "HTTP/1.1 200 ");

}

**@Test**

void checkContentType() {

logger.info

("\*\*\*\*\*\*\*\*Inside checkContentType\*\*\*\*\*\*\*\*\*");

**String** contentType=response.header("Content-Type");

logger.info("Content type is ==>"+contentType);

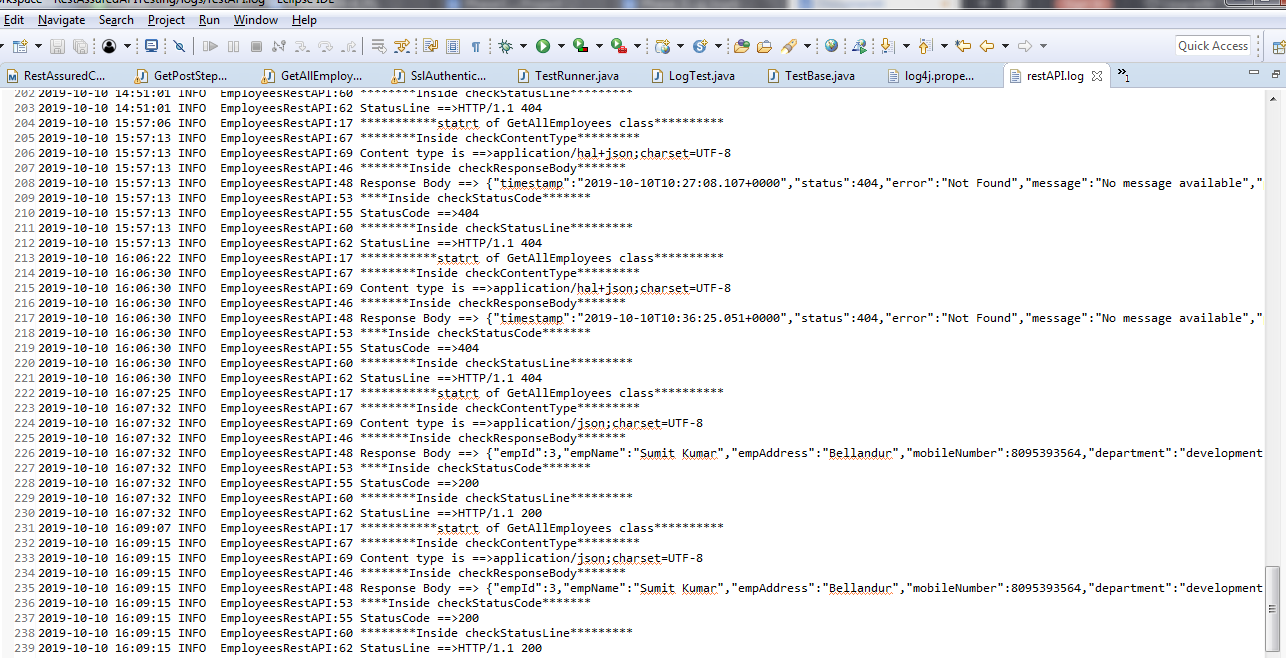
Assert.assertEquals

(contentType, "application/json; charset=UTF-8");

}

}

* Right click on GetAllEmployees class --> Run As --> TestNG Test.
* Open restAPI.log file from logs folder and verify the output:



**Step 3.9.5:** 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