Authentication and Authorization in REST WebServices

Authentication and Authorization in REST WebServices

Authentication and Authorization in REST WebServices are two very important concepts in the context of ***REST API***. Majority of the time you will be hitting REST API's which are secured. By secure we mean that the API's which require you to provide ***identification***. Identification can be provided in the form of

- *Username and a Password*
- *Authentication tokens*
- *Secret keys*
- *Bio-metrics and many other ways*

In the context of REST API, we will be more interested in the first three options. The Authentication and Authorization models that we will discuss are spread across multiple tutorials, starting from this tutorial.

What is Authentication? and How does Authorization works in REST WebServices?

Authentication is a process to prove that you are the person who you intend to be.

*For e.g. while logging into your email account, you prove that **you are you** by providing a **Username** and a **Password**. If you have the **Username** and the **Password** you are who you profess to be. This is what Authentication means.*

In the context of REST API authentication happens using the HTTP Request.

Note: Not just REST API, authentication on any application working via HTTP Protocol happens using the HTTP Request.

Basic Authentication Flow

Taking the example of email login, we know that in order to Authenticate our self we have to provide a Username and a Password. In a very basic Authentication flow using Username and Password, we will do the same thing in REST API call as well. but how do we send the Username and Password in the REST request ?

A REST request can have a special header called ***Authorization Header,*** this header can contain the credentials (username and password) in some form. Once a request with Authorization Header is received, server can validate the credentials and can let you access the private resources.

Note: I hope from previous tutorials you are able to understand the meaning of a Resource. If not, please go through this tutorial: Rest architectural elements. A private resource is one which is not accessible to everyone. You need to Authenticate yourself to access the private resource. For e.g. the email inbox, you have to login to see the emails.

Let us see it with an example, we have created an API which needs a valid Username and Password to access the Resource.

Endpoint: ***http://restapi.demoqa.com/authentication/CheckForAuthentication***

In the code below we will try to hit the URL and see what is the Response that we get.

```
@Test
public void AuthenticationBasics()
{
 RestAssured.baseURI = "http://restapi.demoqa.com/authentication/CheckForAuthentication";
 RequestSpecification request = RestAssured.given();

 Response response = request.get();
 System.out.println("Status code: " + response.getStatusCode());
 System.out.println("Status message " + response.body().asString());
}
```

In the code above we are simply making a ***HTTP GET*** request to the endpoint. In this code we have not added any **Authorization** header. So the expected behaviour is that we will get Authorization error. If you run this test, you will get following output.

```
Status code: 401
Status message:
{
    "StatusID": "FAULT_USER_INVALID_USER_PASSWORD",
    "Status": "Invalid or expired Authentication key provided"
}
```

The output clearly says that we have ***"Invalid or expired Authentication key provided"*** error. Which means that either there was no Authentication information or the information supplied was invalid. Eventually, server denies our request and returns an Error response.

Note: *Pay special attention to the* ***Status code*** *returned. In case of* ***Authentication*** *failures Server should respond with a status code of* ***401 Unauthorized.***

Try to hit that URL using a browser. You should get a Username and Password prompt. Below image shows what you should be getting when you hit this URL from browser.

In this tutorial we will not discuss about how to pass Authentication information in the Request header. Here we will only focus on the definitions of **Authentication** and **Authorization**. In the next set of tutorials we will see different ***Authentication models***, which will solve the above problem.

What is Authorization? and How does Authorization works in REST WebServices?

Authorization is the process of giving access to someone. If you are Authorized then you have access to that resource. Now to Authorize you need to present credentials and as we discussed earlier that process is called Authentication. Hence Authorization and Authentication are closely related terms and often used interchangeably.

Before ending the tutorial let us see the contents of the private resource in the URL mentioned above. To do that enter the following credentials

Username: ToolsQA

Password: TestPassword

Server will be able to **Authenticate** and then **Authorize** you to access the private resource content. Below image shows the content after successful Authentication.

With this basic understanding of Authentication and Authorization, read the coming tutorials where we will discuss the specif types of **Authentication** models in **REST API**.

*For more updates on **Rest API Testing using Rest Assured**, please **Subscribe** to our Newsletter.*

*Please ask any questions on **ForumsQA**, in case of any issues or doubts.*

Category: Rest Assured • By Virender Singh • December 27, 2017

Tags: Rest-Assured RestAPI Testing

Share this post

Author: Virender Singh

I am Virender Singh and I am a software Engineer.I have been in the Software profession for more than 12 years now. I worked on large spectrum of projects, from being a QA engineer to being a Development Engineer. At present I work as a Software Engineer for Microsoft India Development centre. I love to learn new technologies, specially in the field of Image Processing and Digital Signal processing. You can find me at following locations @LinkedIn @FaceBook @ToolsQA @Github

Related posts

Deserialize JSON Array to an Array

March 21, 2018

Deserialize Json Response

December 25, 2017

POST Request using Rest Assured

December 20, 2017

Serialization and Deserialization in Java

December 2, 2017

Rest Architectural Elements

October 18, 2017

Deserialize JSON Array to List

October 16, 2017

SUBSCRIBE TO NEWSLETTER

Enter your email address:

Subscribe

GOT SELENIUM PROBLEMS ?

RECENT POST

Different types of Asserts in Postman

10 Simple Tips to Approach API Testing for Beginners

How to Handle Scroll to Element in Mobile Automation with Katalon Studio

Cookies in Postman

What is a Cookie?

Sessions In Postman

OAuth 2.0 Authorization with Postman

OAuth 2.0 Authorization

A Guide on Integrating Katalon Studio with TestRail

Using Katalon Studio to Approach Web Element Locators

Mock Server in Postman

Automate Shadow DOM Elements with Katalon Studio

Postman Cheat Sheet

Postman Interview Questions

Got a Question?

.

.Subscribe in a reader

.

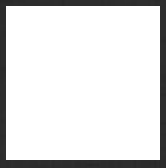
Site Links

Selenium Training
Corporate Training
Video Tutorials
About Us
Guest Blogs
Testimonials
Contact Us
SITEMAP

Tutorials

- **Software Testing**
- **Selenium - Java**
- **Selenium - C#**
- **Cucumber**
- **SpecFlow**
- **Appium**
- **TestNg**
- **JUnit**
- **Maven**
- **Java**
- **Postman**
- **Katalon**

Author



I'M LAKSHAY SHARMA AND I'M A TEST AUTOMATION ENGINEER.

Have passed 11 years playing with automation in mammoth projects like O2 (UK), Sprint (US), TD Bank (CA), Canadian Tire (CA), NHS (UK) & ASOS(UK).
Currently I am working with **BLOOMREACH** as SDET.

I am passionate about designing Automation Frameworks that are effective and easy to maintain. For automating websites my weapons are **QTP** and **Selenium (Webdriver)**. I live in Amsterdam(NL), with my wife and a lovely daughter.

Please connect with me at **LinkedIn** or follow me on **Instagram**.