(/)

Selenium Easy
(/)

Complete Automation Testing Tutorials

Database testing using Java, Selenium, TestNG example

As we know every application has to maintain a database like My SQL, Oracle or any other databases to store all its data. And where as Selenium Webdriver is used for testing web applications and we perform many operations like submitting information and some times retrieving information and validate them.

In selenium scripts, when there is a need of getting the Data from the database we may have to use APIs which helps to interact with database like JDBC.

Java Database Connectivity(JDBC) (https://en.wikipedia.org/wiki/Java_Database_Connectivity) is a Java API which is used to connect and interact with Database.

Why do we use JDBC instead ODBC?

ODBC is developed and written in C, which is platform Independent. where as JDBC API is a specification provides set of interfaces which are written and developed in Java programming language.

How to Connect to database using JDBC?

The below are the Steps to Connect to database, before proceeding, you need to have MySQL Connector. You can download from here Download MySQL Connector Jar (https://dev.mysql.com/downloads/connector/j/5.0.html) and add it the build path as we add selenium webdriver jar.

1. Load and Registering the Driver
2. Establishing Connection.
3. Creating Statement Object
4. Execute the Statement
5. Closing the connection.

1. Load and Register the Driver:

For registering the Driver we Load the Driver class using forName() method.

forName() is the static factory method which is present in predefined class called "Class". This method loads the class which is mentioned as parameter.

```
Class.forName("com.mysql.jdbc.Driver");// class.forName load the Driver class
```

Internally this Driver class will register the driver by using static method called registerDriver().

2. Establishing Connection:

For establishing connection with database we call static method called getConnection(...) present in DriverManager Class. This method contains three arguments of string type. i.e., url, username and password

```
DriverManager.getConnection("jdbc:mysql://localhost:3306/Employee","root","root");
```

URL contains "jdbc(main protocol):mysql(sub protocol for mySql)://localhost:3306(sub name for mysql (host:prot))/Employee(database)" and this method return type is Connection Object ie.,

```
Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/Employee","root","root");
```

3. Creating Statement Object:

For creating statement object we need to call a method called createStatement() which is present in Connection Interface.

```
con.createStatement();
```

And this method returns Statement object and it is no argument method.

```
Statement st= con.createStatement();
```

4.Executing Queries:

For executing queries there are different methods present in Statement Interface for retrieving records and for updating records.

Retrieving records:
for executing select queries(for fetching records) we call a method called executeQuery(String qry) by taking string as parameter.

```
st.executeQuery("Select * from Employee");
```

This method returns ResultSet object.

```
Resultset rs= st.executeQuery("Select * from Employee");// once executeQuery() executes the query and stores the records in to ResultSet object.
```

Now we need to get the records from ResultSet object. To access the resultset object it uses a method called next() which presents in ResultSet Interface.

By default Resultset reference 'rs' points to before first row. it moves rs to next row and returns true. When it returns true we retrieve the data in first row. next() returns false when rs points to after the last row. this next() will repeats the execution using while loop till it returns false.

To get the data from rows we use getXxx(..) taking string or integer as parameters. Here integer means column position and string means column name of the record. xxx indicates primitive datatypes or string object.

```
while(rs.next()) {
int EmpId= rs.getInt("EmpId");
String EmpName= rs.getString("EmpName");
String EmpAddress=rs.getString(3);
Double EmpSal= rs.getDouble(4);
String EmpDept=rs.getString("EmpDept");
System.out.println(EmpId+"\t"+EmpName+"\t"+EmpAddress+"\t"+EmpSal+"\t"+EmpDept);
}
```

Updating records:

To update records in a table we use a method called executeUpdate(String str)

```
st.executeUpdate("update Employee set EmpName='Robert' where EmpId=2");
```


ny records have been updated.

```
ployee set EmpName='Robert' where EmpId=2");
dated is" +result);
```

completed we need to close all the connections by using method called close() present in Connection interface

```
con.close();
```

Database which we use for example has the following records in Table 'Employee'

EmpId	EmpName	EmpAddress	EmpDept	EmpSal
1	Jack	New Jersey	IT	20000
2	Rose	California	IT	50000
3	Rachel	Colorado	HR	40000
4	James	New Jersey	Manager	70000
5	Smith	California	Admin	30000

Now its time to look into below simple example program and later we will jump into framework example (http://seleniumeasy.com/selenium-tutorials/how-to-do-database-testing-using-selenium-webdriver-framework-example) : - In next tutorial, We will try to take an example which will take the parameters (like database credentials , environment details etc) from properties files and proceed the test based on parameters specified.

```
/**
 * Created by VISISHTA on 12/17/2015.
 */

import org.testng.annotations.AfterClass;
import org.testng.annotations.BeforeClass;
import org.testng.annotations.Test;

import java.sql.*;

public class SeleniumDataBaseTesting {

   private Connection connection;
   private static Statement statement;
   private static ResultSet rs;

    @BeforeClass
    public void setUp() {
            String databaseURL = "jdbc:mysql://localhost:3306/easy";
            String user = "root";
            String password = "root";
            connection = null;
            try {
                Class.forName("com.mysql.jdbc.Driver");
                System.out.println("Connecting to Database...");
                connection = DriverManager.getConnection(databaseURL, user, password);
                if (connection != null) {
                    System.out.println("Connected to the Database...");
                }
            } catch (SQLException ex) {
               ex.printStackTrace();
            }
            catch (ClassNotFoundException ex) {
               ex.printStackTrace();
            }
    }

    @Test
    public void getEmployeesFromDataBase() {
        try {
            String query = "select * from employee";
            statement = connection.createStatement();
            rs = statement.executeQuery(query);

            while(rs.next()){
                int EmpId= rs.getInt("EmpId");
                String EmpName= rs.getString("EmpName");
                String EmpAddress=rs.getString(3);
                String EmpDept=rs.getString("EmpDept");
                Double EmpSal= rs.getDouble(5);
                System.out.println(EmpId+"\t"+EmpName+"\t"+EmpAddress+"\t"+EmpSal+"\t"+EmpDept);
            }
        } catch (SQLException ex) {
           ex.printStackTrace();
        }
    }
```

```
    @AfterClass
    public void tearDown() {
      if (connection != null) {
                try {
                    System.out.println("Closing Database Connection...");
                    connection.close();
                } catch (SQLException ex) {
                    ex.printStackTrace();
                }
            }
      }
}
```

In the above example, we have, setUp() will create a database connection, and in @Test method we will execute and iterate the table to print all the values in the database. And last tearDown() has close connection which will close the database connection if it is opened.

Output of the above program should look like below:

```
"C:\Program ...
[TestNG] Running:
  C:\Users\VISISHTA\.IdeaIC13\system\temp-testng-customsuite.xml

Connecting to Database...
Connected to the Database...
1   Jack    New Jersey  20000.0 IT
2   Rose    California  50000.0 IT
3   Rachel  Colorado    40000.0 HR
4   James   New Jersey  70000.0 Manager
5   Smith   California  30000.0 Admin
Closing Database Connection...

===============================================
Custom suite
Total tests run: 1, Failures: 0, Skips: 0
===============================================
```

Till now we have seen example on how to access database,now let us see how we actually perform database testing using Selenium.

There are mainly two things that we might want to test to verify database with Selenium.

First, after making any action in the web application (Front-end), we may want to check the database if all the details we submitted are stored correctly. Now for example, when creating an account, we will input some data email Id, username etc. To make sure an account is created, we can check record created in database with all the details user provided.

Other case can be like, after performing any action, we may want to check the data displayed in the UI is correct by comparing the Database.

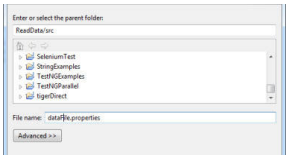
Hope this article helps you. Do let us know if have come across any difficulties when working with the above program and feedback is also welcome.

‹ Handling Security Certificates in Chrome and IE browser using WebDriver (/selenium-tutorials/handling-security-certificates-chrome-and-ie-browser-using-webdriver) How to do Database testing using Selenium Webdriver Framework Example › (/selenium-tutorials/how-to-do-database-testing-using-selenium-webdriver-framework-example)

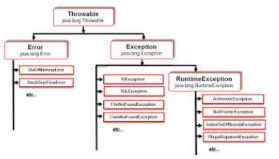
Selenium Tutorials:
Selenium Tutorials (/selenium-tutorials)

EmpName	EmpAddress	EmpDept	EmpSa
Jack	New Jersey	IT	20000
Rose	California	IT	50000
Rachel	Colorado	HR	40000
James	New Jersey	Manager	70000
Smith	California	Admin	30000

[How to do Database testing using Selenium Webdriver Framework...]

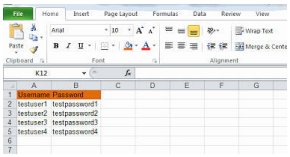


[Read data from Properties file using Java Selenium]



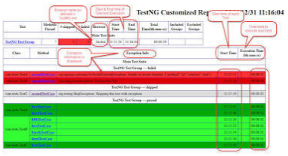
[Exception Handling in Java selenium webdriver]

[Selenium Automation Framework Example]

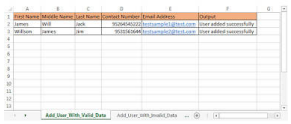


[How to Read Excel file using Java]

[How to verify tooltip text with selenium webdriver using java]



[Customize TestNG html Report]



[Test Data in automation framework]

Tags:

Database Testing (/tags/database-testing)

Add new comment (/selenium-tutorials/database-testing-example-with-selenium-using-java#comment-form)

Comments

@beforeclass (/comment/4349#comment-4349)

Permalink (/comment/4349#comment-4349) Submitted by Gaurav on Wed, 04/27/2016 - 06:18

Hi Team,

Hope you to find in a great spirit.i want to know actual behavior of @beforeclass ???

what does actually means of this @beforeclass

reply (/comment/reply/182/4349)

Very Good information and (/comment/4622#comment-4622)

Permalink (/comment/4622#comment-4622) Submitted by Viewer on Tue, 05/10/2016 - 08:44

Very Good information and detailed explanation with example. Appreciated

reply (/comment/reply/182/4622)

Add new comment

Your name

Subject

Comment *

More information about text formats (/filter/tips)

I'm not a robot reCAPTCHA Privacy - Terms

Save Preview

Recent Post

- FirefoxOptions for running Webdriver Tests (/selenium-tutorials/firefoxoptions-running-webdriver-tests)
- If, If Else and Nested If examples with Selenium (/java-tutorials/how-to-use-if-else-nested-if-conditions-in-webdriver)
- Chrome Options for running WebDriver Tests (/selenium-tutorials/using-chrome-options-for-webdriver-tests)
- Double Click on element using Webdriver (/selenium-tutorials/how-to-double-click-on-element-using-webdriver)
- Appium grid configuration Example (/appium-tutorials/connecting-appium-to-selenium-grid-example)

Selenium Popular Posts

- Page Factory Framework (http://www.seleniumeasy.com/selenium-tutorials/page-factory-pattern-in-selenium-webdriver)
- POM example (http://www.seleniumeasy.com/selenium-tutorials/simple-page-object-model-framework-example)
- Working with Selenium Grid (http://www.seleniumeasy.com/selenium-tutorials/how-to-configure-selenium-grid)
- Click element with JavaScriptExecutor (http://www.seleniumeasy.com/selenium-tutorials/click-element-using-javascriptexecutor)
- Css selectors examples (http://www.seleniumeasy.com/selenium-tutorials/css-selectors-tutorial-for-selenium-with-examples)

TestNG Popular Posts