# Integration of Selenium WebDriver with Cucumber

In last module we discussed Cucumber tool, its usage and different features.

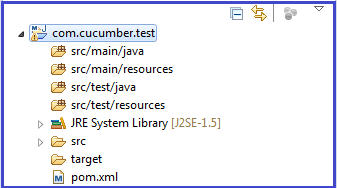
Moving ahead in, we will discuss how to set up a cucumber project and will discuss about integration of selenium WebDriver with Cucumber.

We will set up Cucumber project with Maven

**Cucumber Project Setup:**

**Step #1:** Create New Maven Project:  
*Right Click -> New -> Others -> Maven -> Maven Project -> Next*

**Step #2:** Now the project will look like this:

[](http://cdn.softwaretestinghelp.com/wp-content/qa/uploads/2014/11/Cucumber-Selenium-1.jpg)

**Step #3**: Add below dependencies in pom.xml

|  |  |  |
| --- | --- | --- |
| 1 | <dependencies> | |
| 2 | <dependency> |

|  |  |
| --- | --- |
| 3 | <groupId>info.cukes</groupId> |
| 4 | <artifactId>cucumber-java</artifactId> | |

|  |  |  |
| --- | --- | --- |
| 5 | <version>1.0.2</version> | |
| 6 | <scope>test</scope> |

|  |  |  |
| --- | --- | --- |
| 7 | </dependency> | |
| 8 | <dependency> |

|  |  |  |
| --- | --- | --- |
| 9 | <groupId>info.cukes</groupId> | |
| 10 | | <artifactId>cucumber-junit</artifactId> | |

|  |  |  |
| --- | --- | --- |
| 11 | <version>1.0.2</version> | |
| 12 | <scope>test</scope> |

|  |  |  |
| --- | --- | --- |
| 13 | </dependency> | |
| 14 | <dependency> |

|  |  |
| --- | --- |
| 15 | <groupId>junit</groupId> |
| 16 | <artifactId>junit</artifactId> | |

|  |  |  |
| --- | --- | --- |
| 17 | <version>4.10</version> | |
| 18 | <scope>test</scope> |

|  |  |
| --- | --- |
| 19 | </dependency> |
| 20 | </dependencies> | |

**Step #4**: Create a sample.feature file under src/test/resources.

@smokeTest  
**Feature**: To test my cucumber test is running  
I want to run a sample feature file.

**Scenario**: cucumber setup

**Given** sample feature file is ready  
**When** I run the feature file  
**Then** run should be successful

**Step #5**: Create a class under src/test/java which will implement all the steps.

|  |  |
| --- | --- |
| 1 | public class stepDefinition { |
| 2 | @Given("^sample feature file is ready$") | |

|  |  |
| --- | --- |
| 3 | public void givenStatment(){ |
| 4 | System.out.println("Given statement executed successfully"); | |

|  |  |
| --- | --- |
| 5 | } |
| 6 | @When("^I run the feature file$") | |

|  |  |
| --- | --- |
| 7 | public void whenStatement(){ |
| 8 | System.out.println("When statement execueted successfully"); | |

|  |  |  |
| --- | --- | --- |
| 9 | } | |
| 10 | | @Then("^run should be successful$") | |

|  |  |
| --- | --- |
| 11 | public void thenStatment(){ |
| 12 | System.out.println("Then statement executed successfully"); | |

|  |  |  |
| --- | --- | --- |
| 13 | } | |
| 14 | } |

**Step #6**: Create a Junit runner to run the test.

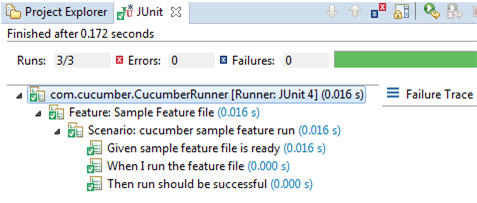
|  |  |
| --- | --- |
| 1 | @RunWith(Cucumber.class) |
| 2 | @Cucumber.Options(format={"pretty","html:reports/test-report"},tags= "@smokeTest") | |

|  |  |  |
| --- | --- | --- |
| 3 | public class CucumberRunner { | |
| 4 | } |

Provide the path of the report as given here. The reports will store in ‘test-report’ folder under project folder and “pretty” format specifies the type of report.

**Step #7**: Junit Result and Test Report:

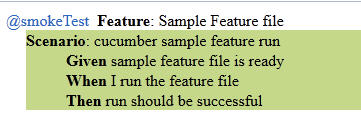
Below is the report when the cucumber test is successful. Green bar in Junit describes the test is passed. Similarly red bar describes that test has failed.

[](http://cdn2.softwaretestinghelp.com/wp-content/qa/uploads/2014/11/Cucumber-Selenium-2.jpg)

------------

If we want to use default reporting then navigate the path mentioned in Junit Runner. In this case we have given path as *reports->test-reports->index.html.*

Open this report in Internet Explorer or in Firefox to verify the result. Below is the sample of the report:

[](http://cdn.softwaretestinghelp.com/wp-content/qa/uploads/2014/11/Cucumber-Selenium-3.jpg)

**Cucumber and Selenium WebDriver:**

Cucumber framework can be used to test the web based applications along with Selenium WebDriver. The test cases are written in simple feature files which are easily understood by managers, non-technical stake holders and business analysts. And those feature file steps are implemented in step definition file. If you are using maven then you have to add dependencies for Cucumber and WebDriver.

So here is the sample test case we have implemented using Cucumber and WebDriver. As given below, the scenario in feature file is self-explanatory.

**Feature: Login Feature File**  
@selenium  
**Scenario**: Login scenario test for Gmail

**Given** navigate to gmail page  
**When** user logged in using username as “userA” and password as “password”  
**Then** home page should be displayed

**WebDriver Implementation in Cucumber stepDefinitions:**

|  |  |  |
| --- | --- | --- |
| 1 | public class stepDefinition { | |
| 2 | WebDriver dr; |

|  |  |  |
| --- | --- | --- |
| 3 | @Given("^navigate to gmail page$") | |
| 4 | public void navigate(){ |

|  |  |
| --- | --- |
| 5 | dr=new FirefoxDriver(); |
| 6 | dr.get("http://www.gmail.com"); | |

|  |  |
| --- | --- |
| 7 | } |
| 8 | @When ("^user logged in using username as \"(.\*)\" and password as \"(.\*)\"$") | |

|  |  |  |
| --- | --- | --- |
| 9 | public void login(String username,String password){ | |
| 10 | | dr.findElement(By.xpath("//\*[@id='Email']")).sendKeys(username); | |

|  |  |  |
| --- | --- | --- |
| 11 | dr.findElement(By.xpath("//\*[@id='Passwd']")).sendKeys(password); | |
| 12 | dr.findElement(By.xpath("//\*[@id='signIn']")).click(); |

|  |  |  |
| --- | --- | --- |
| 13 | dr.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS); | |
| 14 | } |

|  |  |  |
| --- | --- | --- |
| 15 | @Then("^home page should be displayed$") | |
| 16 | public void verifySuccessful(){ |

|  |  |
| --- | --- |
| 17 | String expectedText="Gmail"; |
| 18 | String actualText=         dr.findElement(By.xpath("//\*[@id='gbq1']/div/a/span")).getText(); | |

|  |  |  |
| --- | --- | --- |
| 19 | Assert.assertTrue("Login not successful",expectedText.equals(actualText)); | |
| 20 | } |

|  |  |
| --- | --- |
| 21 | } |

In this test we have used the Firefox as the browser to test the Gmail login functionality.  
Clearly WebDriver object is a class variable and used across the class.

**Given** statement initializes the browser and navigates to the page.  
**When** statement logs into the application using username as “userA” and password as “password”. Both the values ‘username’ and ‘password’ are passed from feature file and both the values to be used in the same order.  
**Then** Statement only validates the conditions after logging into the application.

This is a sample test describing the usage of Cucumber and Selenium. You can create multilayer architecture depending upon your project requirement.

**Conclusion:**

In this Module we have covered most of the Cucumber concepts which includes Cucumber features and its usage along with WebDriver.

This reduces the complexity of code which is written to design the traditional frameworks like Keyword Driven and Hybrid Framework. Cucumber is used in most of the project where people follow agile methodology as Behavior Driven Development is an Agile Software practice.