Way2Automation - Tutorial 5 – Cucumber Tags

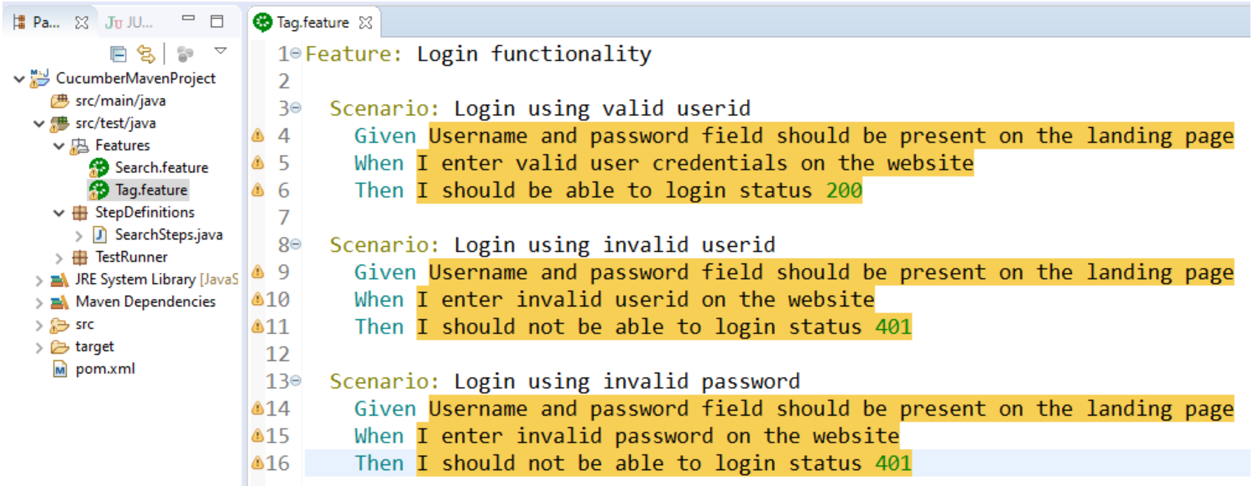
What you will Learn :

- Importance of cucumber tags
- Add 2 tags separated by ‘or’
- Add 2 tags separated by ‘and’
- Use tags to ignore a scenario
- Execute all the scenarios

Importance of cucumber tags

We can do some excellent things when we execute our test cases with cucumber. We can tag sanity based test cases, tag smoke test cases, regression test cases etc. We can also do some documentation using tags.

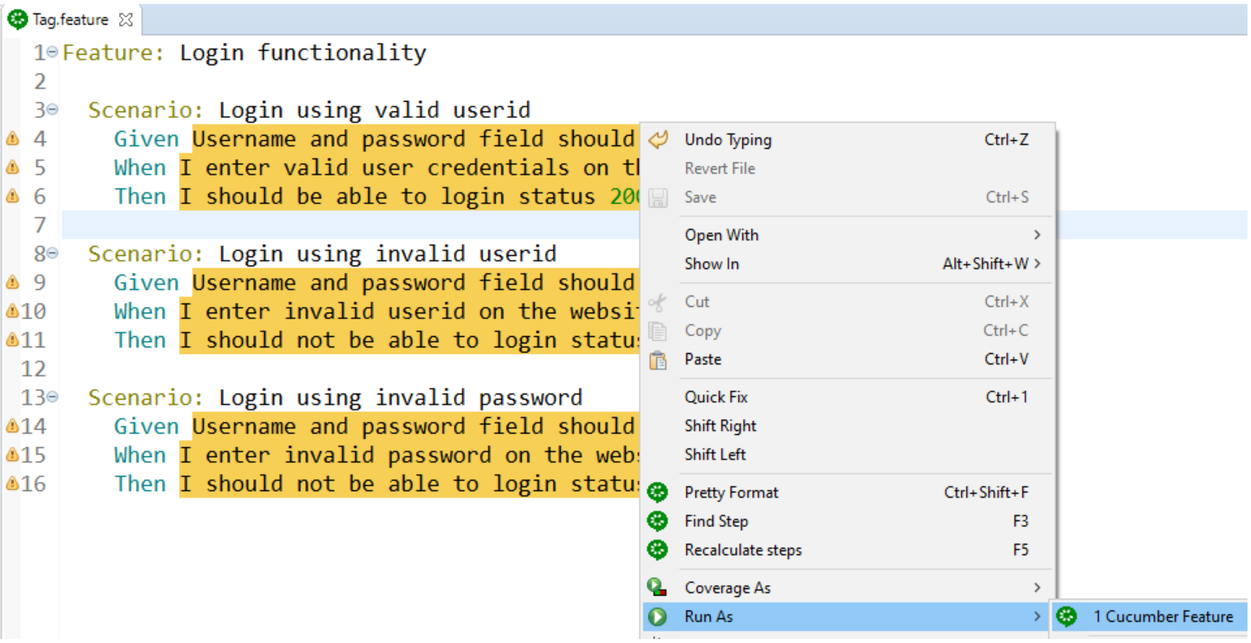
To further understand the tags, let us create a new feature file having 3 scenarios



```
Feature: Login functionality

 Scenario: Login using valid userid
   Given Username and password field should be present on the landing page
   When I enter valid user credentials on the website
   Then I should be able to login status 200

 Scenario: Login using invalid userid
   Given Username and password field should be present on the landing page
   When I enter invalid userid on the website
   Then I should not be able to login status 401

 Scenario: Login using invalid password
   Given Username and password field should be present on the landing page
   When I enter invalid password on the website
   Then I should not be able to login status 401
```

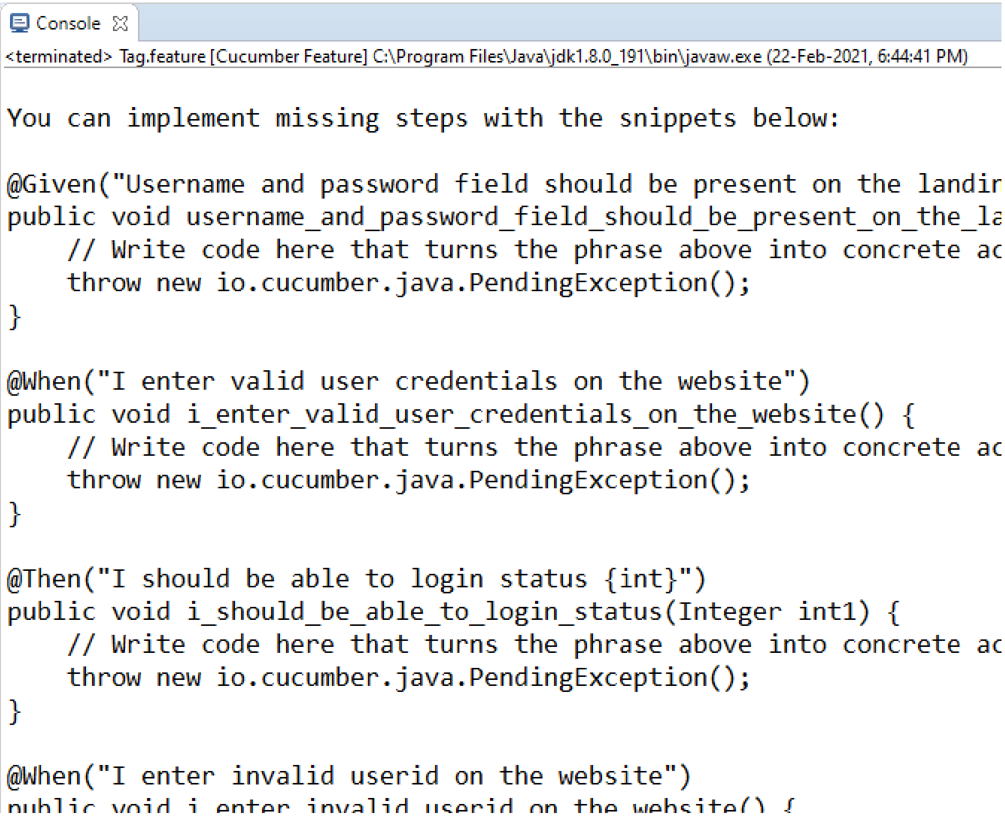
Save the file

Run this feature file



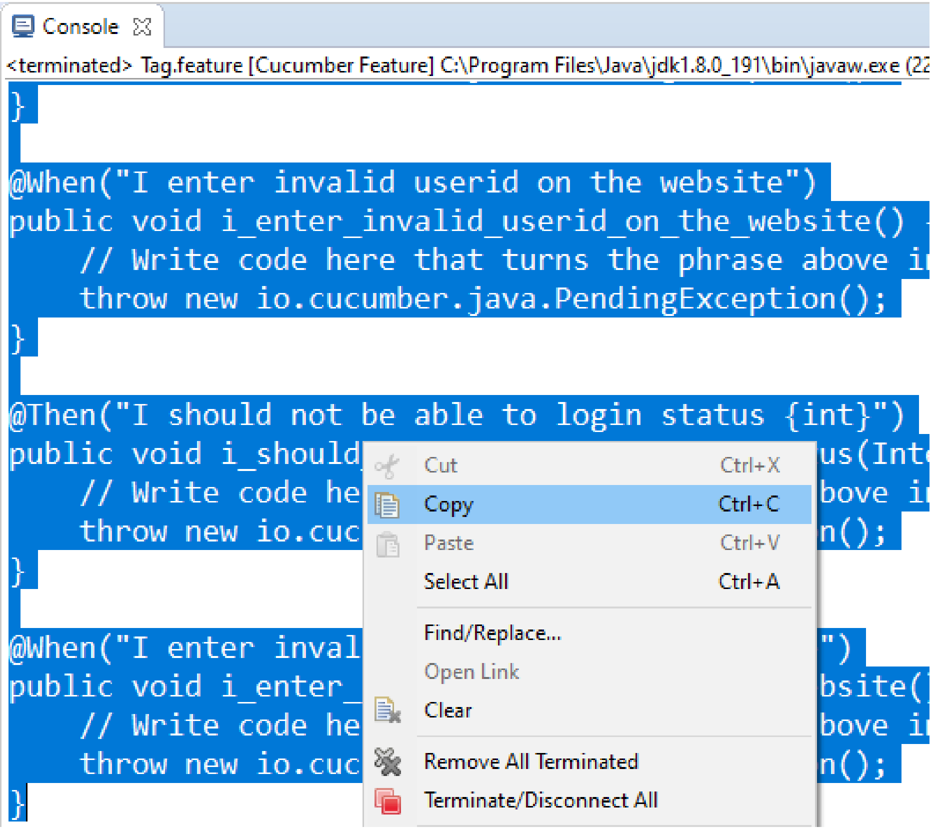
```
Feature: Login functionality

  Scenario: Login using valid userid
    Given Username and password field should
    When I enter valid user credentials on t
    Then I should be able to login status 20

  Scenario: Login using invalid userid
    Given Username and password field should
    When I enter invalid userid on the websi
    Then I should not be able to login statu

  Scenario: Login using invalid password
    Given Username and password field should
    When I enter invalid password on the web
    Then I should not be able to login statu
```

As expected, the console would show us below error



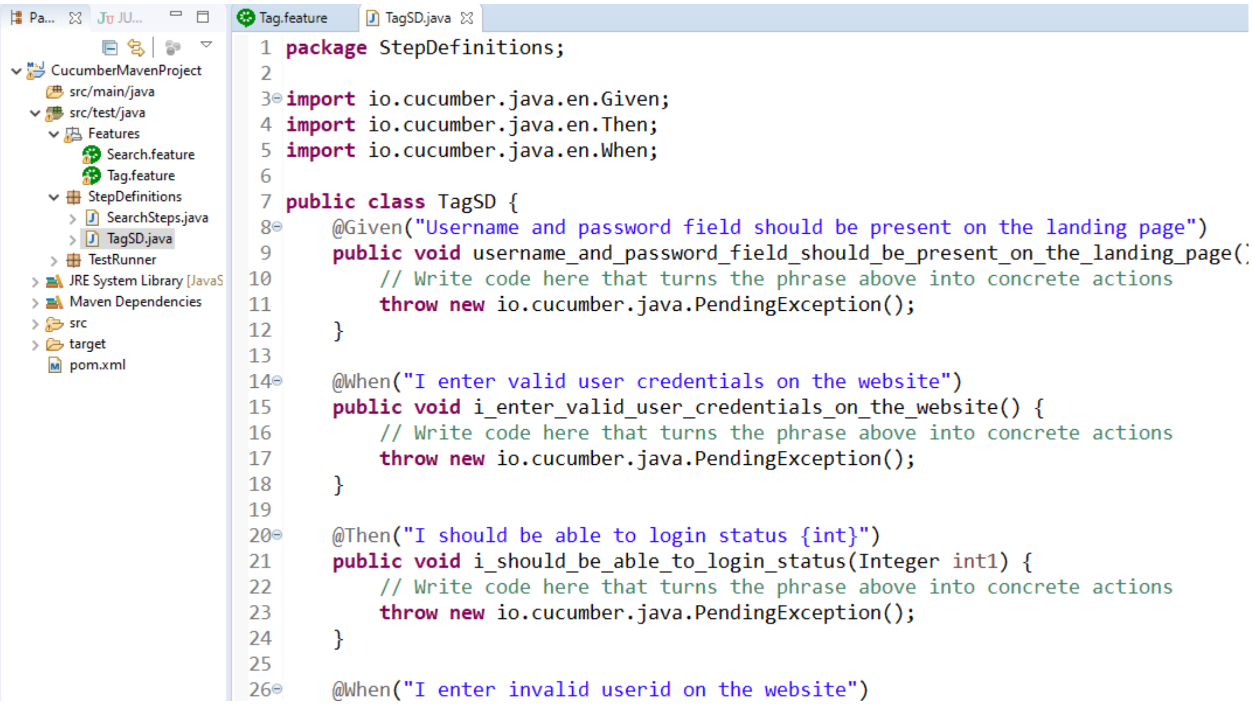
```
<terminated> Tag.feature [Cucumber Feature] C:\Program Files\Java\jdk1.8.0_191\bin\javaw.exe (22-Feb-2021, 6:44:41 PM)

You can implement missing steps with the snippets below:

@Given("Username and password field should be present on the landir
public void username_and_password_field_should_be_present_on_the_la
    // Write code here that turns the phrase above into concrete ac
    throw new io.cucumber.java.PendingException();
}

@When("I enter valid user credentials on the website")
public void i_enter_valid_user_credentials_on_the_website() {
    // Write code here that turns the phrase above into concrete ac
    throw new io.cucumber.java.PendingException();
}

@Then("I should be able to login status {int}")
public void i_should_be_able_to_login_status(Integer int1) {
    // Write code here that turns the phrase above into concrete ac
    throw new io.cucumber.java.PendingException();
}

@When("I enter invalid userid on the website")
public void i_enter_invalid_userid_on_the_website() {
```

Copy all the code snippets from the console file



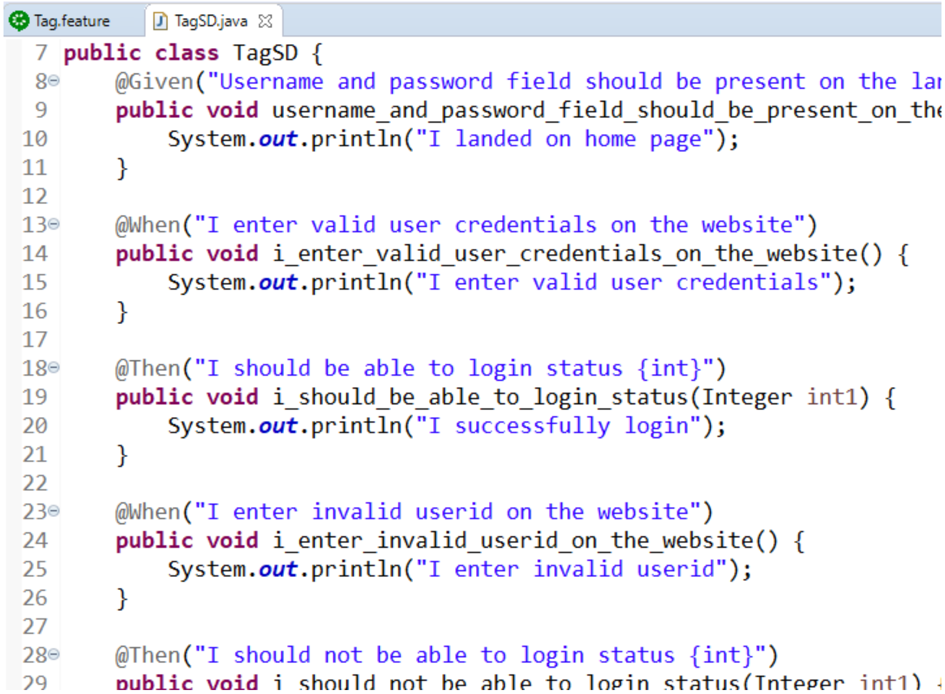
```
<terminated> Tag.feature [Cucumber Feature] C:\Program Files\Java\jdk1.8.0_191\bin\javaw.exe (22
}

@When("I enter invalid userid on the website")
public void i_enter_invalid_userid_on_the_website() 
    // Write code here that turns the phrase above i
    throw new io.cucumber.java.PendingException();
}

@Then("I should not be able to login status {int}")
public void i_should                               us(Int
    // Write code he                               bove i
    throw new io.cuc                               n();
}

@When("I enter inval                               ")
public void i_enter_                               bsite(
    // Write code he                               bove i
    throw new io.cuc                               n();
}
```

Let us create the corresponding step definition file for the above feature file. Paste the code snippets and import the packages



```java
package StepDefinitions;

import io.cucumber.java.en.Given;
import io.cucumber.java.en.Then;
import io.cucumber.java.en.When;

public class TagSD {
    @Given("Username and password field should be present on the landing page")
    public void username_and_password_field_should_be_present_on_the_landing_page(
        // Write code here that turns the phrase above into concrete actions
        throw new io.cucumber.java.PendingException();
    }

    @When("I enter valid user credentials on the website")
    public void i_enter_valid_user_credentials_on_the_website() {
        // Write code here that turns the phrase above into concrete actions
        throw new io.cucumber.java.PendingException();
    }

    @Then("I should be able to login status {int}")
    public void i_should_be_able_to_login_status(Integer int1) {
        // Write code here that turns the phrase above into concrete actions
        throw new io.cucumber.java.PendingException();
    }

    @When("I enter invalid userid on the website")
```

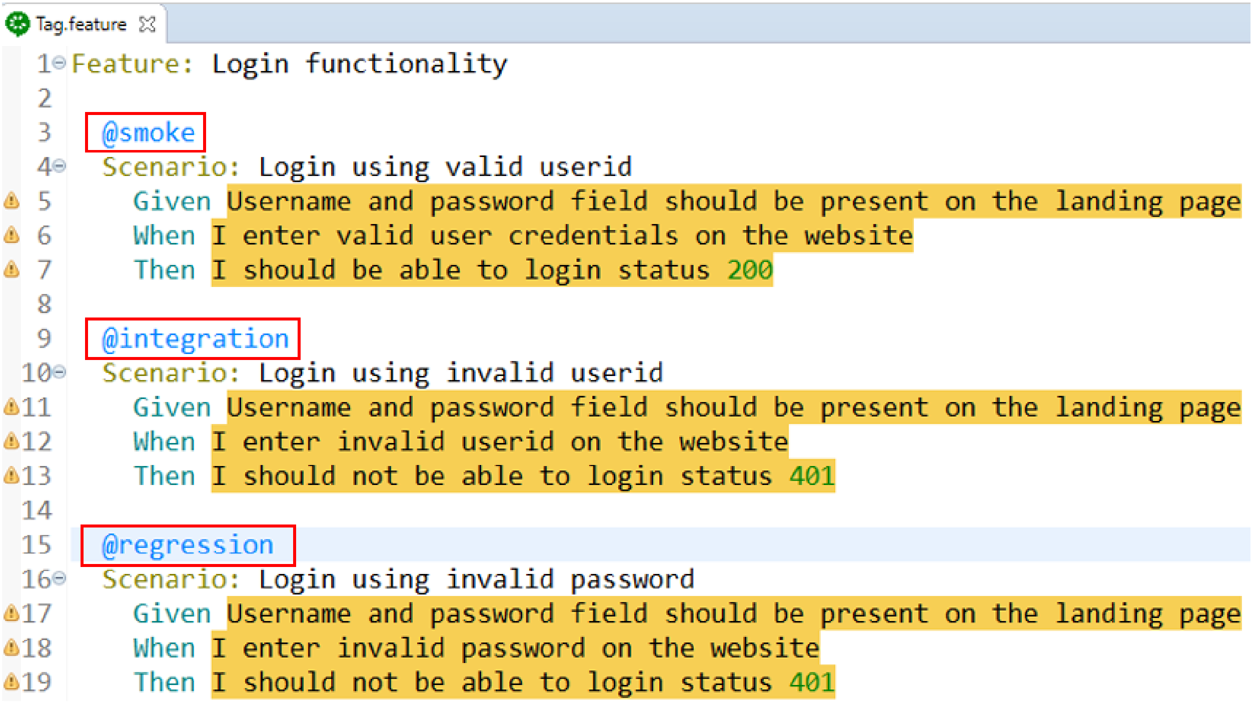
Let us remove the exceptions and simple method implementations



```
public class TagSD {
    @Given("Username and password field should be present on the lan
    public void username_and_password_field_should_be_present_on_the
        System.out.println("I landed on home page");
    }

    @When("I enter valid user credentials on the website")
    public void i_enter_valid_user_credentials_on_the_website() {
        System.out.println("I enter valid user credentials");
    }

    @Then("I should be able to login status {int}")
    public void i_should_be_able_to_login_status(Integer int1) {
        System.out.println("I successfully login");
    }

    @When("I enter invalid userid on the website")
    public void i_enter_invalid_userid_on_the_website() {
        System.out.println("I enter invalid userid");
    }

    @Then("I should not be able to login status {int}")
    public void i_should_not_be_able_to_login_status(Integer int1) {
```

Save the file.

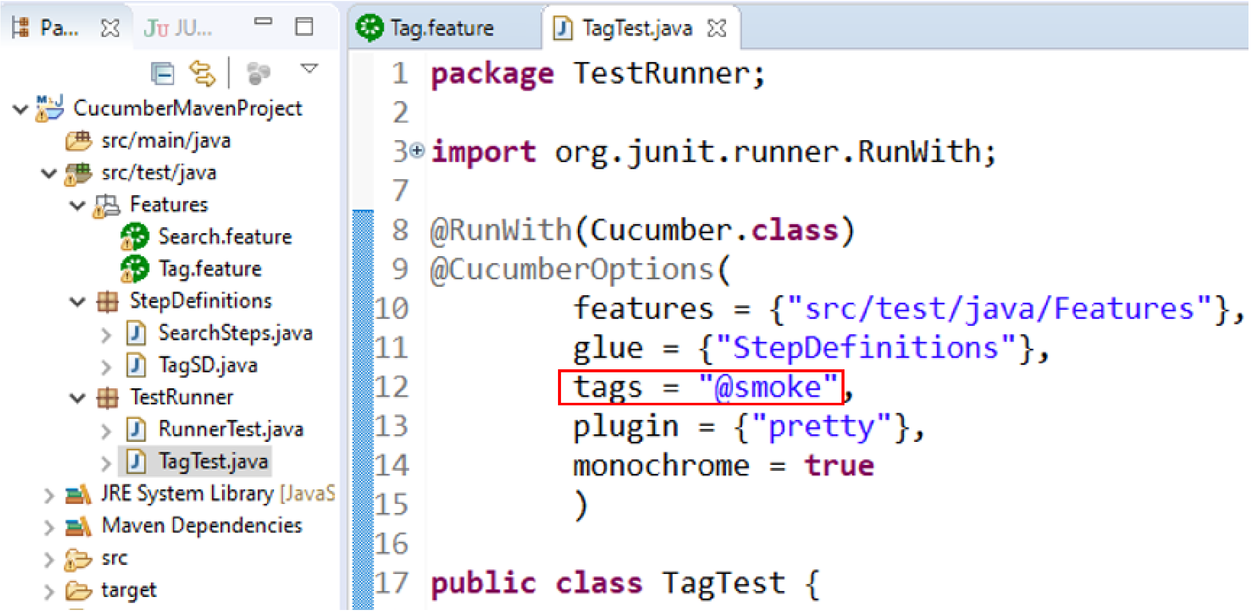
We will now create ‘tag’ in our feature file for all the 3 scenarios. So we have created 3 tags for the respective test cases or scenarios



```
Feature: Login functionality

 @smoke
 Scenario: Login using valid userid
   Given Username and password field should be present on the landing page
   When I enter valid user credentials on the website
   Then I should be able to login status 200

 @integration
 Scenario: Login using invalid userid
   Given Username and password field should be present on the landing page
   When I enter invalid userid on the website
   Then I should not be able to login status 401

 @regression
 Scenario: Login using invalid password
   Given Username and password field should be present on the landing page
   When I enter invalid password on the website
   Then I should not be able to login status 401
```

Save the file

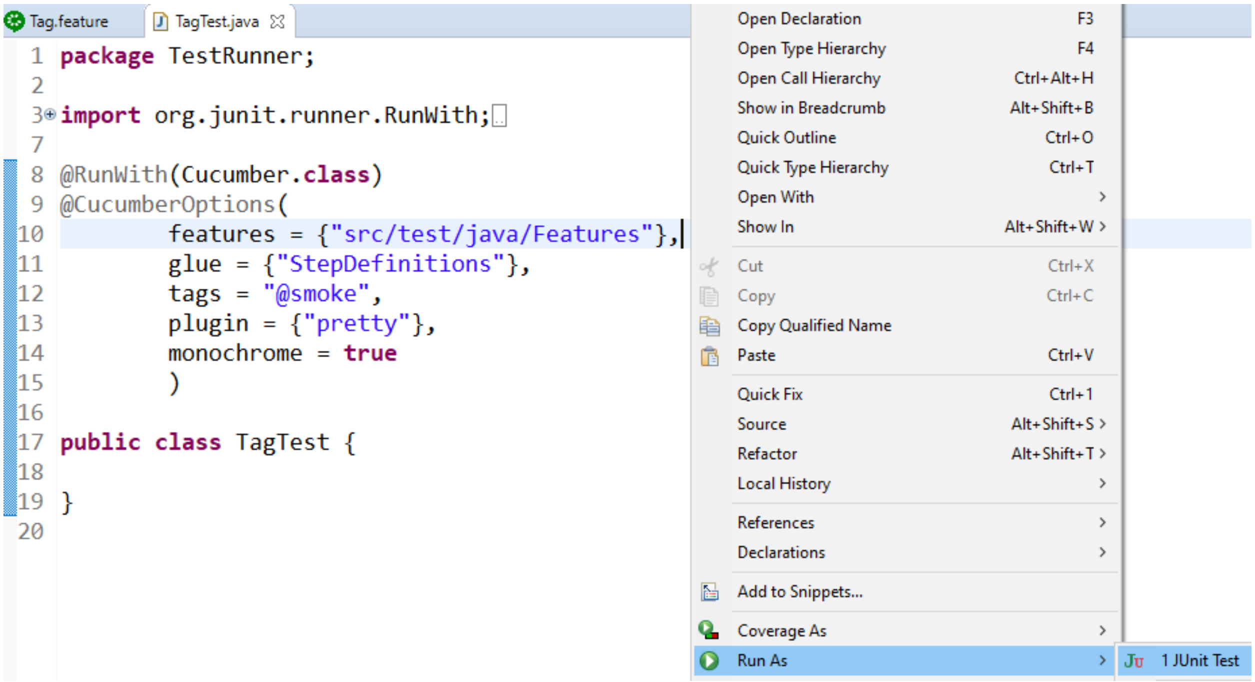
We will now create a runner file and add ‘tags’ option as seen below. So, we want to run only ‘smoke’ based test cases.



```
package TestRunner;

import org.junit.runner.RunWith;

@RunWith(Cucumber.class)
@CucumberOptions(
        features = {"src/test/java/Features"},
        glue = {"StepDefinitions"},
        tags = "@smoke",
        plugin = {"pretty"},
        monochrome = true
        )

public class TagTest {
```

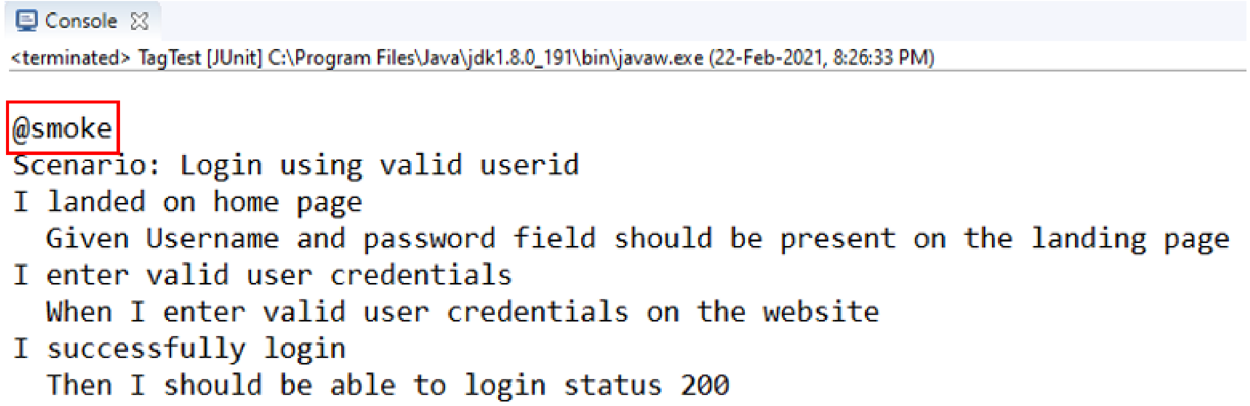
Note that, although there are 2 feature files in the ‘Features’ package, still we haven’t explicitly mentioned the ‘Tag.feature’ file in the ‘features’ option.

Execute this runner file



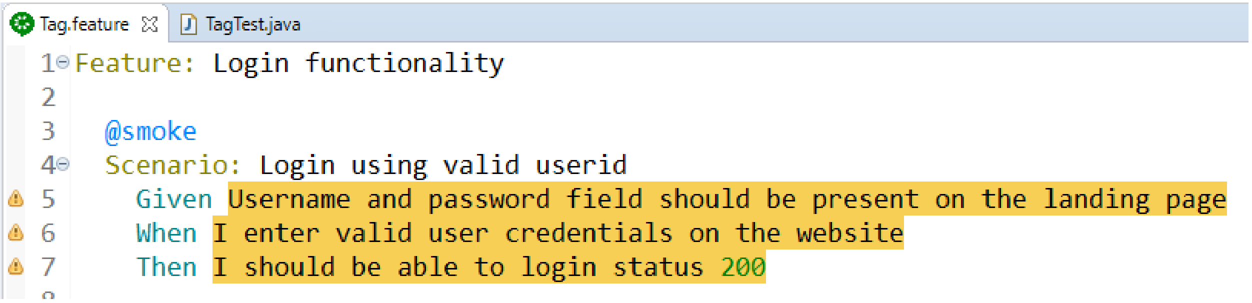
```
package TestRunner;

import org.junit.runner.RunWith;

@RunWith(Cucumber.class)
@CucumberOptions(
        features = {"src/test/java/Features"},
        glue = {"StepDefinitions"},
        tags = "@smoke",
        plugin = {"pretty"},
        monochrome = true
        )

public class TagTest {

}
```

Notice below that @smoke scenario got executed. Cucumber automatically scanned both the feature files and it found/executed @smoke tag in Tag.feature file. It did not execute other scenarios



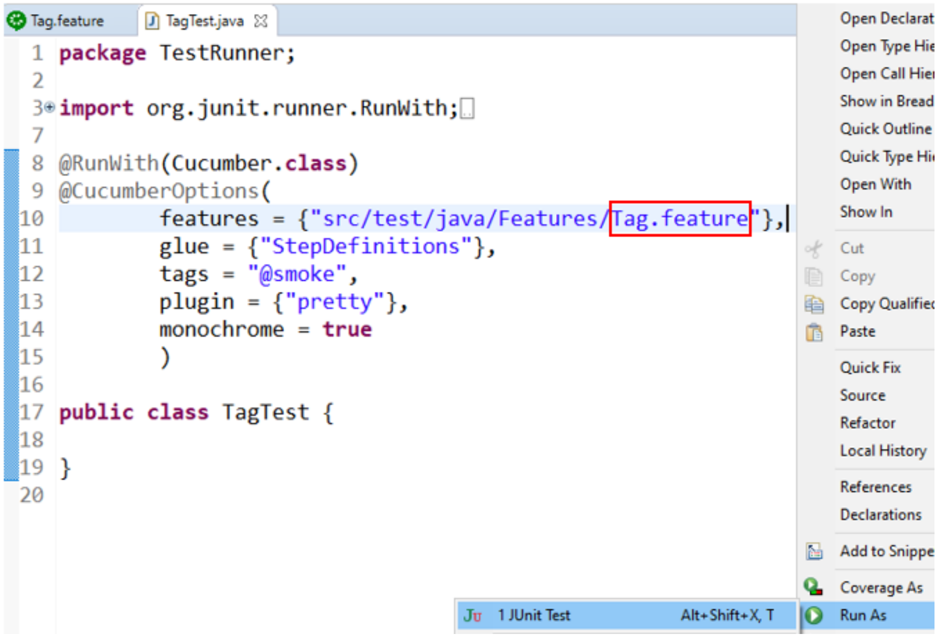
```
<terminated> TagTest [JUnit] C:\Program Files\Java\jdk1.8.0_191\bin\javaw.exe (22-Feb-2021, 8:26:33 PM)

@smoke
Scenario: Login using valid userid
I landed on home page
  Given Username and password field should be present on the landing page
I enter valid user credentials
  When I enter valid user credentials on the website
I successfully login
  Then I should be able to login status 200
```

To double check, the console o/p matches with the @smoke tag in feature file



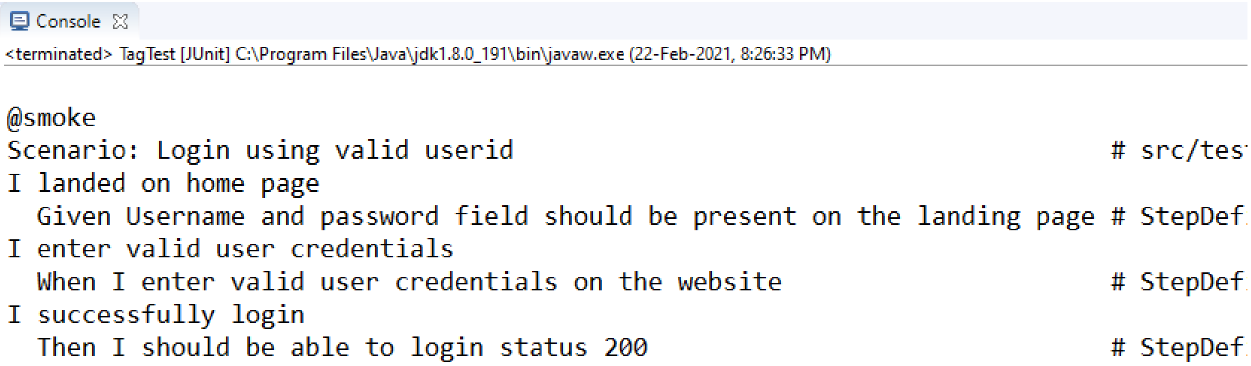
```
Feature: Login functionality

  @smoke
  Scenario: Login using valid userid
    Given Username and password field should be present on the landing page
    When I enter valid user credentials on the website
    Then I should be able to login status 200
```

Let us now mention Tag.feature in the ‘feature’ option and run the file



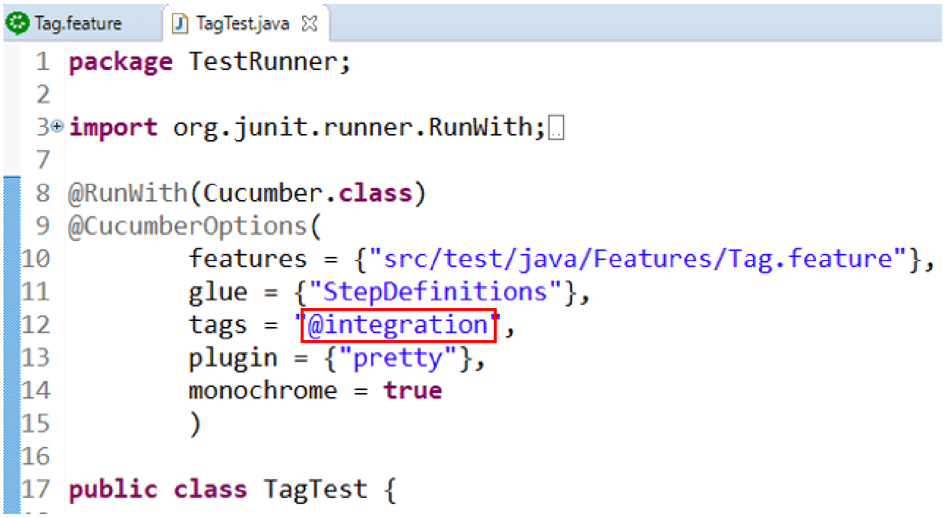
```
package TestRunner;

import org.junit.runner.RunWith;

@RunWith(Cucumber.class)
@CucumberOptions(
        features = {"src/test/java/Features/Tag.feature"},
        glue = {"StepDefinitions"},
        tags = "@smoke",
        plugin = {"pretty"},
        monochrome = true
        )

public class TagTest {

}
```

We again get the same o/p



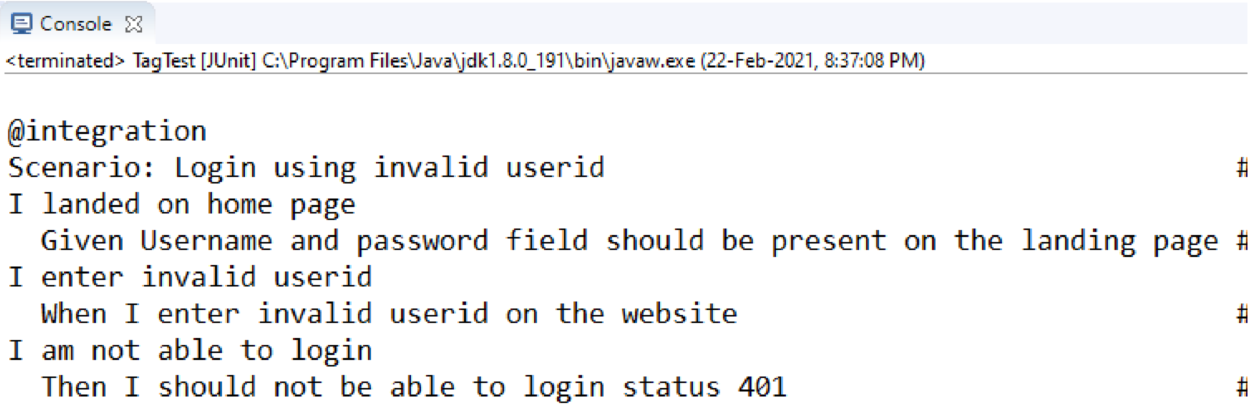
```
<terminated> TagTest [JUnit] C:\Program Files\Java\jdk1.8.0_191\bin\javaw.exe (22-Feb-2021, 8:26:33 PM)

@smoke
Scenario: Login using valid userid                                              # src/tes
I landed on home page
  Given Username and password field should be present on the landing page # StepDef
I enter valid user credentials
  When I enter valid user credentials on the website                          # StepDef
I successfully login
  Then I should be able to login status 200                                   # StepDef
```

Let us now change the tag to @integration



```
package TestRunner;

import org.junit.runner.RunWith;

@RunWith(Cucumber.class)
@CucumberOptions(
        features = {"src/test/java/Features/Tag.feature"},
        glue = {"StepDefinitions"},
        tags = "@integration",
        plugin = {"pretty"},
        monochrome = true
        )

public class TagTest {
```

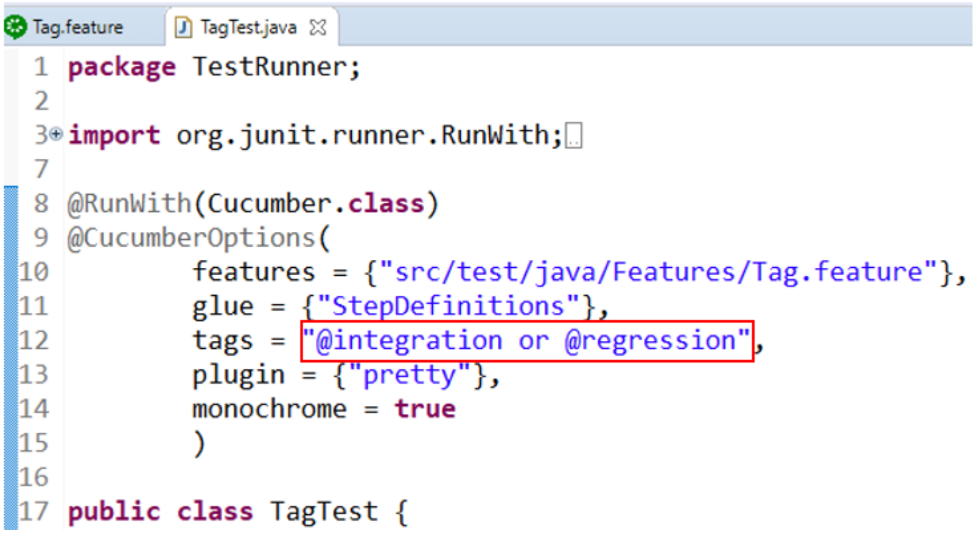
Save and run, notice below that this time only @integration scenario got executed



```
<terminated> TagTest [JUnit] C:\Program Files\Java\jdk1.8.0_191\bin\javaw.exe (22-Feb-2021, 8:37:08 PM)

@integration
Scenario: Login using invalid userid                                          #
I landed on home page
  Given Username and password field should be present on the landing page #
I enter invalid userid
  When I enter invalid userid on the website                                #
I am not able to login
  Then I should not be able to login status 401                             #
```

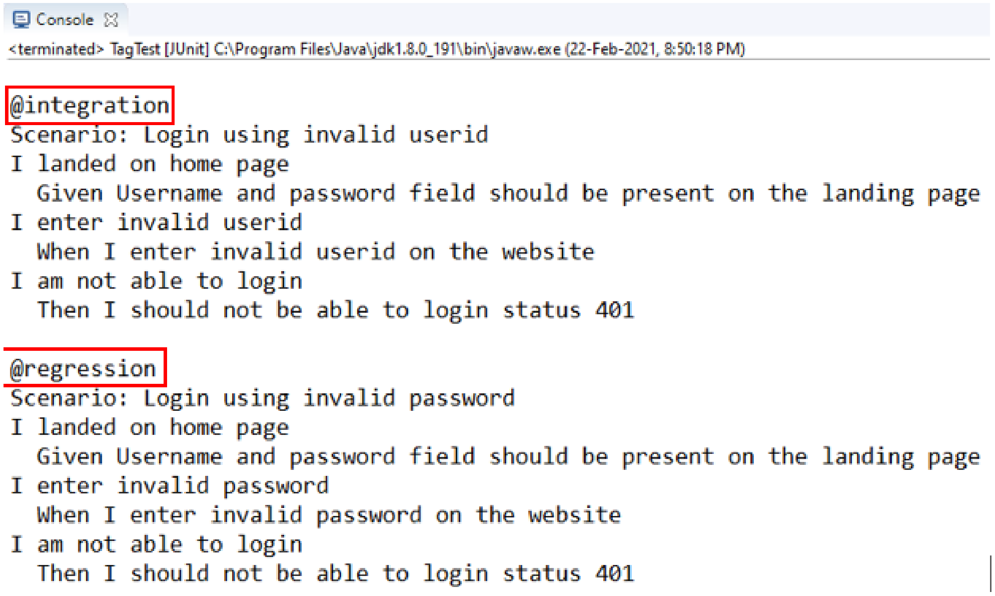
<u>Add 2 tags seperated by ‘or’</u>

You can execute 2 tags as well, see below. By using ‘or’ we mean, execute those test cases which are tagged with @integration or @regression



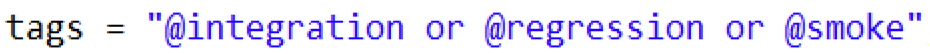
```
package TestRunner;

import org.junit.runner.RunWith;

@RunWith(Cucumber.class)
@CucumberOptions(
         features = {"src/test/java/Features/Tag.feature"},
         glue = {"StepDefinitions"},
         tags = "@integration or @regression",
         plugin = {"pretty"},
         monochrome = true
         )

public class TagTest {
```

Save and run the file, see below, 2 scenarios got executed



```
<terminated> TagTest [JUnit] C:\Program Files\Java\jdk1.8.0_191\bin\javaw.exe (22-Feb-2021, 8:50:18 PM)

@integration
Scenario: Login using invalid userid
I landed on home page
  Given Username and password field should be present on the landing page
I enter invalid userid
  When I enter invalid userid on the website
I am not able to login
  Then I should not be able to login status 401

@regression
Scenario: Login using invalid password
I landed on home page
  Given Username and password field should be present on the landing page
I enter invalid password
  When I enter invalid password on the website
I am not able to login
  Then I should not be able to login status 401
```

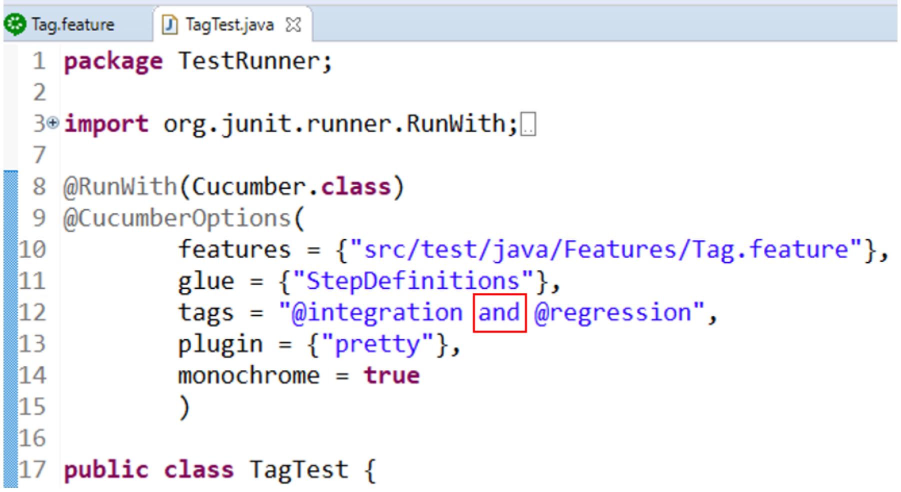
Similarly you can add more tags and execute them



```
tags = "@integration or @regression or @smoke"
```

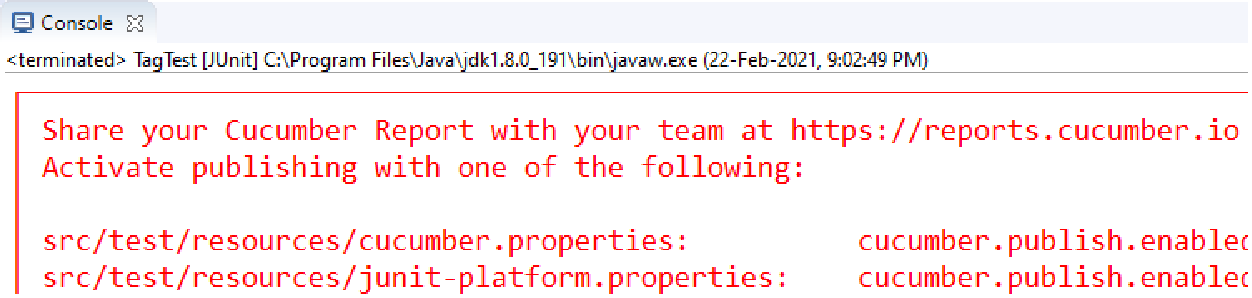
Add 2 tags seperated by ‘and’

Let us separate the tags using ‘and’, see below



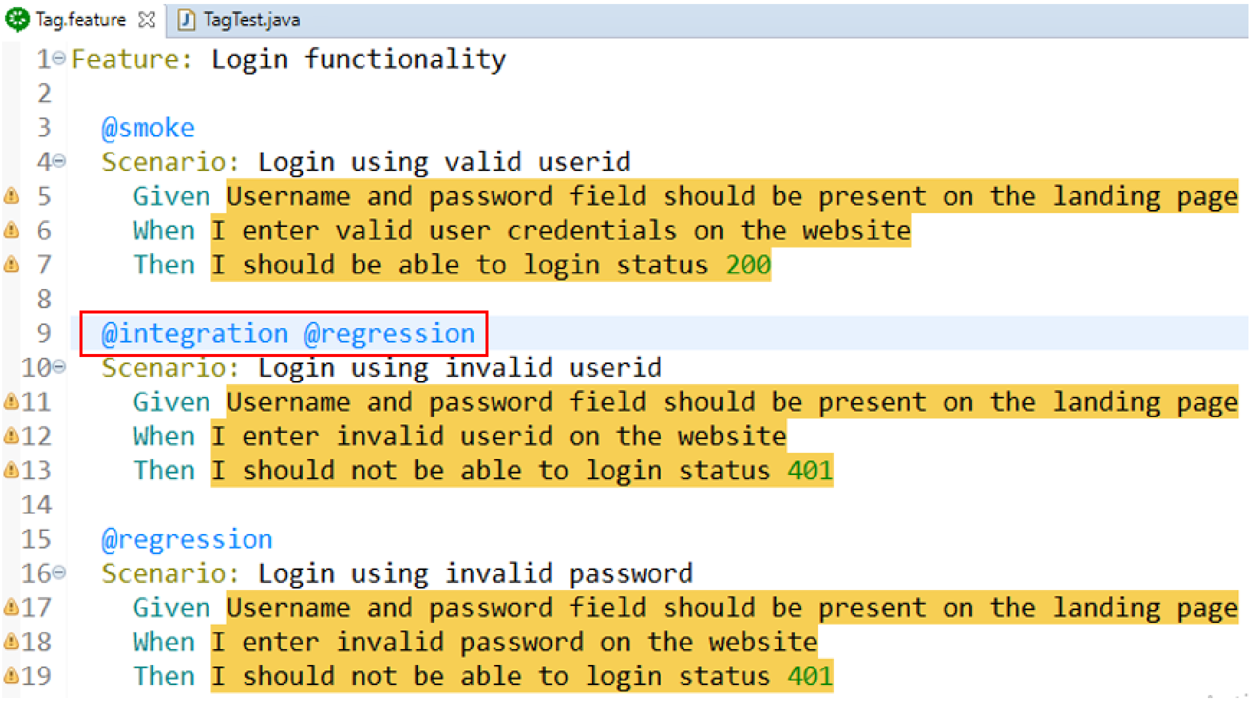
```java
package TestRunner;

import org.junit.runner.RunWith;

@RunWith(Cucumber.class)
@CucumberOptions(
        features = {"src/test/java/Features/Tag.feature"},
        glue = {"StepDefinitions"},
        tags = "@integration and @regression",
        plugin = {"pretty"},
        monochrome = true
        )

public class TagTest {
```

Save and run. Nothing gets executed. The reason being there is no scenario represented by above 2 tags



```
<terminated> TagTest [JUnit] C:\Program Files\Java\jdk1.8.0_191\bin\javaw.exe (22-Feb-2021, 9:02:49 PM)

Share your Cucumber Report with your team at https://reports.cucumber.io
Activate publishing with one of the following:

src/test/resources/cucumber.properties:          cucumber.publish.enablec
src/test/resources/junit-platform.properties:    cucumber.publish.enablec
```

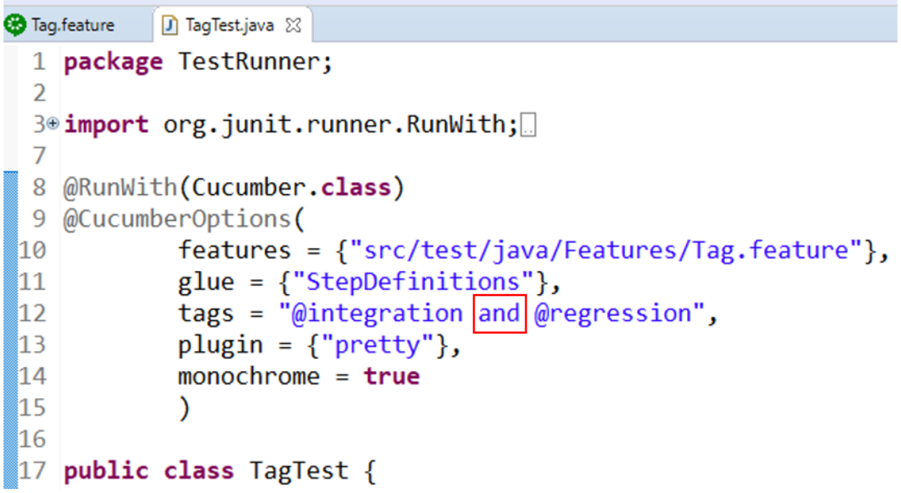
Now, go to the feature file and let us represent second scenario with 2 tags, see below



```
Feature: Login functionality

  @smoke
  Scenario: Login using valid userid
    Given Username and password field should be present on the landing page
    When I enter valid user credentials on the website
    Then I should be able to login status 200

  @integration @regression
  Scenario: Login using invalid userid
    Given Username and password field should be present on the landing page
    When I enter invalid userid on the website
    Then I should not be able to login status 401

  @regression
  Scenario: Login using invalid password
    Given Username and password field should be present on the landing page
    When I enter invalid password on the website
    Then I should not be able to login status 401
```

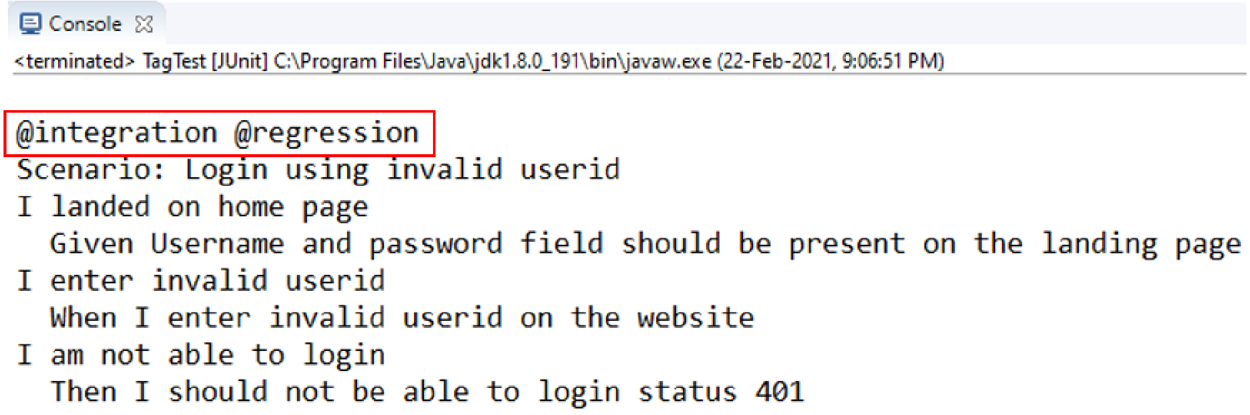
Save the file

Now let us again run the below file



```
package TestRunner;

import org.junit.runner.RunWith;□

@RunWith(Cucumber.class)
@CucumberOptions(
        features = {"src/test/java/Features/Tag.feature"},
        glue = {"StepDefinitions"},
        tags = "@integration and @regression",
        plugin = {"pretty"},
        monochrome = true
        )

public class TagTest {
```

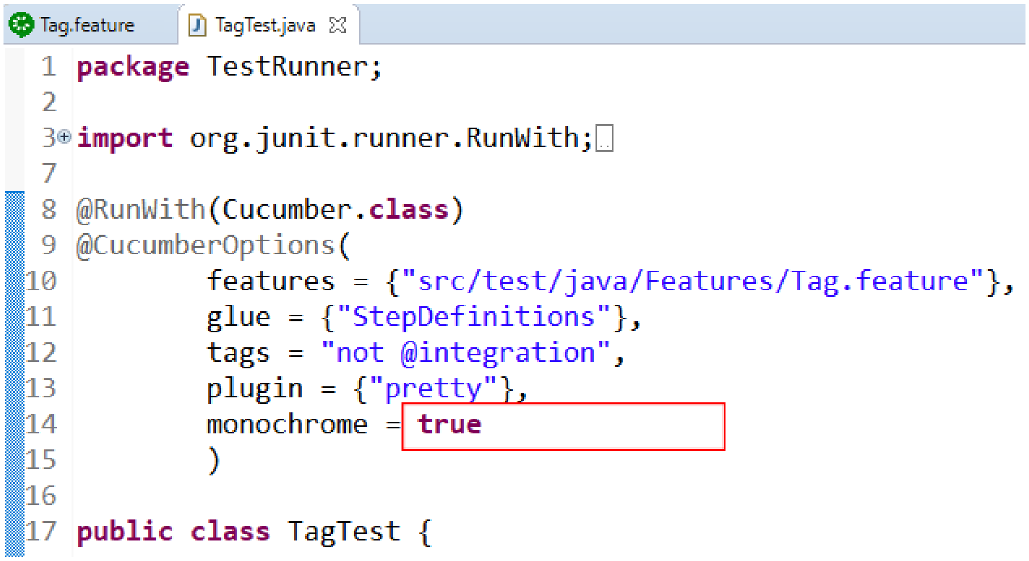
Notice below that this time the scenario got executed



```
<terminated> TagTest [JUnit] C:\Program Files\Java\jdk1.8.0_191\bin\javaw.exe (22-Feb-2021, 9:06:51 PM)

@integration @regression
Scenario: Login using invalid userid
I landed on home page
  Given Username and password field should be present on the landing page
I enter invalid userid
  When I enter invalid userid on the website
I am not able to login
  Then I should not be able to login status 401
```

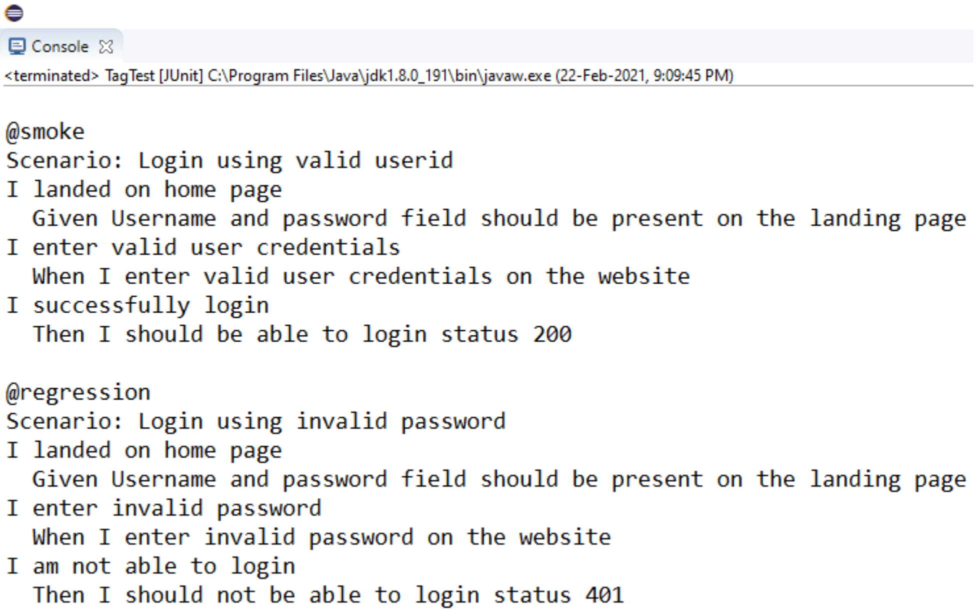
Use tags to ignore a scenario

Let us say we want to execute all the scenarios except @integration. We can do that using 'not' keyword, see below



```java
package TestRunner;

import org.junit.runner.RunWith;

@RunWith(Cucumber.class)
@CucumberOptions(
        features = {"src/test/java/Features/Tag.feature"},
        glue = {"StepDefinitions"},
        tags = "not @integration",
        plugin = {"pretty"},
        monochrome = true
        )

public class TagTest {
```

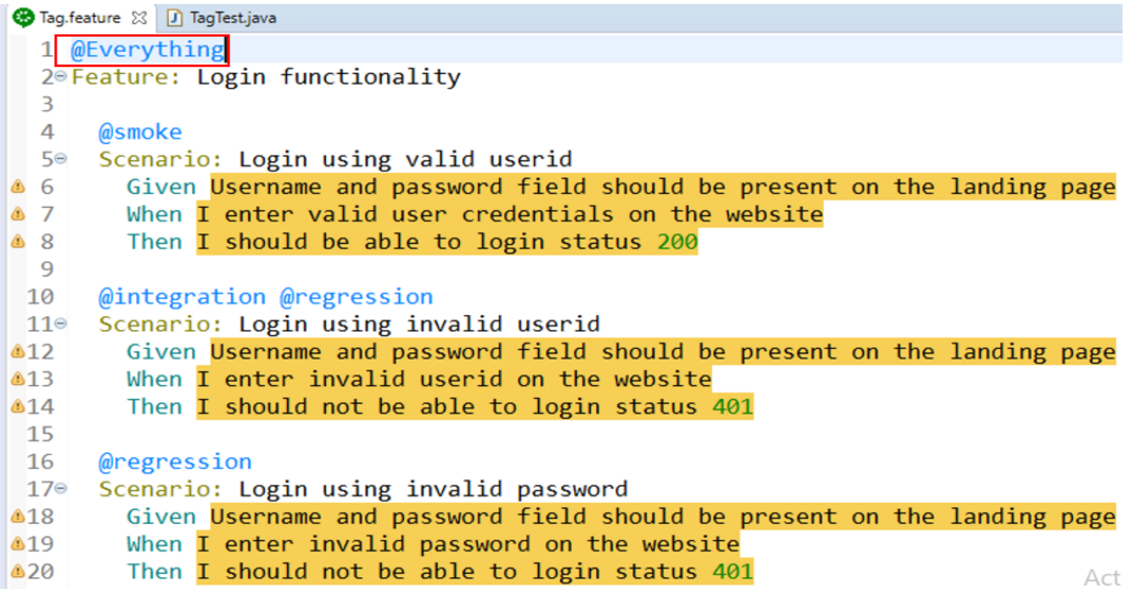
Save and run, see below. All scenarios except @integration got executed



```
<terminated> TagTest [JUnit] C:\Program Files\Java\jdk1.8.0_191\bin\javaw.exe (22-Feb-2021, 9:09:45 PM)

@smoke
Scenario: Login using valid userid
I landed on home page
  Given Username and password field should be present on the landing page
I enter valid user credentials
  When I enter valid user credentials on the website
I successfully login
  Then I should be able to login status 200

@regression
Scenario: Login using invalid password
I landed on home page
  Given Username and password field should be present on the landing page
I enter invalid password
  When I enter invalid password on the website
I am not able to login
  Then I should not be able to login status 401
```

Execute all the scenarios

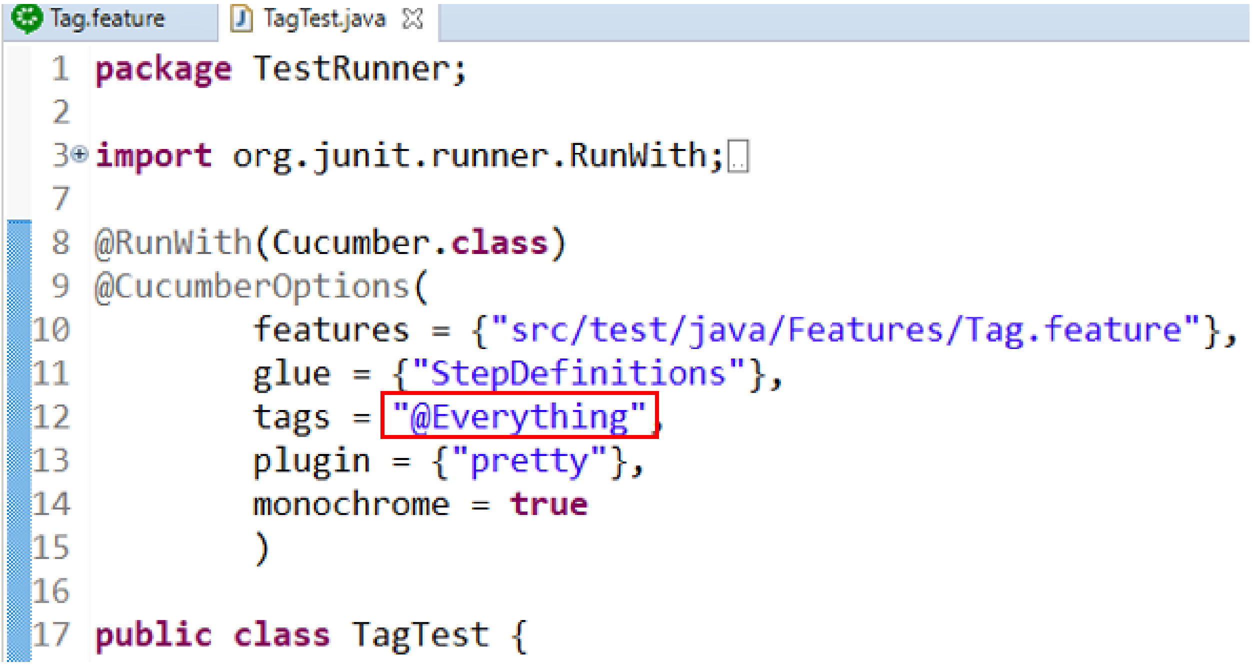
You can also tag at the ‘Feature’ level, see below. All the scenarios will inherit this tag



```
@Everything
Feature: Login functionality

  @smoke
  Scenario: Login using valid userid
    Given Username and password field should be present on the landing page
    When I enter valid user credentials on the website
    Then I should be able to login status 200

  @integration @regression
  Scenario: Login using invalid userid
    Given Username and password field should be present on the landing page
    When I enter invalid userid on the website
    Then I should not be able to login status 401

  @regression
  Scenario: Login using invalid password
    Given Username and password field should be present on the landing page
    When I enter invalid password on the website
    Then I should not be able to login status 401
```

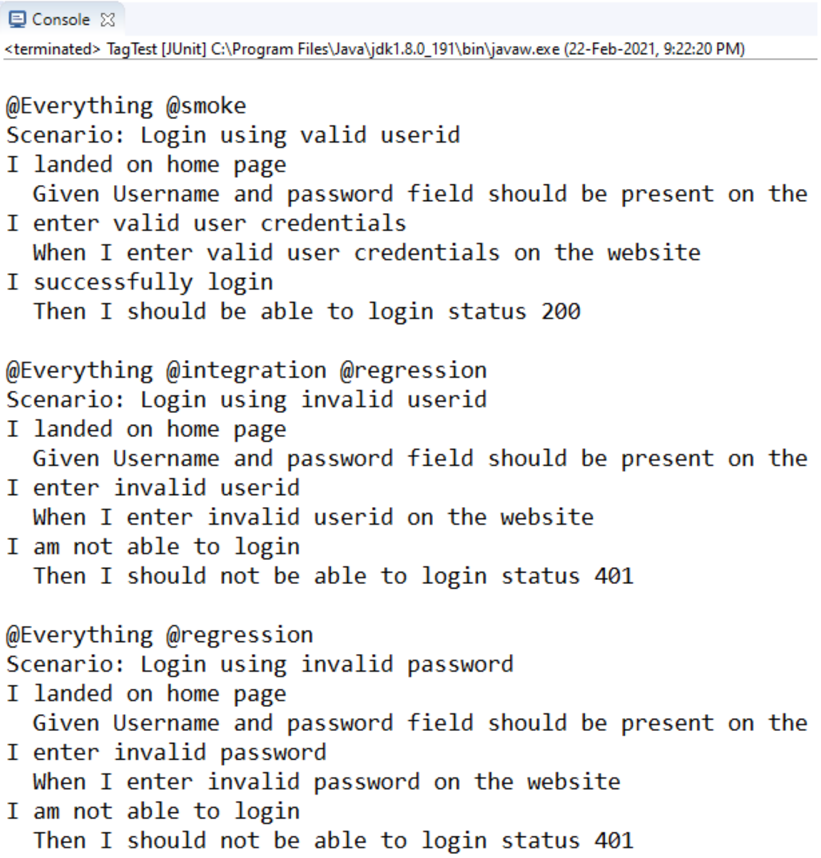
Save the file

Let ususe this tag in runner file, see below



```java
package TestRunner;

import org.junit.runner.RunWith;

@RunWith(Cucumber.class)
@CucumberOptions(
		features = {"src/test/java/Features/Tag.feature"},
		glue = {"StepDefinitions"},
		tags = "@Everything",
		plugin = {"pretty"},
		monochrome = true
		)

public class TagTest {
```

Save an run. Notice below, all scenarios are executed



```
<terminated> TagTest [JUnit] C:\Program Files\Java\jdk1.8.0_191\bin\javaw.exe (22-Feb-2021, 9:22:20 PM)

@Everything @smoke
Scenario: Login using valid userid
I landed on home page
  Given Username and password field should be present on the
I enter valid user credentials
  When I enter valid user credentials on the website
I successfully login
  Then I should be able to login status 200

@Everything @integration @regression
Scenario: Login using invalid userid
I landed on home page
  Given Username and password field should be present on the
I enter invalid userid
  When I enter invalid userid on the website
I am not able to login
  Then I should not be able to login status 401

@Everything @regression
Scenario: Login using invalid password
I landed on home page
  Given Username and password field should be present on the
I enter invalid password
  When I enter invalid password on the website
I am not able to login
  Then I should not be able to login status 401
```

So this is how we can use tags in our feature files.

Thank you for reading!