Cucumber - Tags

In **Cucumber**, **tags** are used to group **scenarios** or **features** so you can control which ones to execute. Tags start with `@` and can be placed above:

- **Feature**: to apply the tag to all scenarios in the feature.
- **Scenario** / **Scenario Outline**: to apply the tag to individual test cases.

✅ Syntax Example

```
gherkin

@SmokeTest
Feature: Login functionality

  @Regression @Login
  Scenario: Valid login
    Given User is on login page
    When User enters valid credentials
    Then User should be redirected to the dashboard

  @Negative
  Scenario: Invalid login
    Given User is on login page
    When User enters invalid credentials
    Then User should see an error message
```

✅ Running Specific Tags (Command Line)

You can run tests by tags using the **`--tags`** option:

Run scenarios with specific tag:

```
bash

cucumber --tags @SmokeTest
```

Run scenarios with multiple tags (AND):

```
bash

cucumber --tags "@SmokeTest and @Login"
```

Run scenarios with either tag (OR):

```
bash

cucumber --tags "@SmokeTest,@Regression"
```

Exclude a tag:

```
bash

cucumber --tags "not @Negative"
```

✅ Tag Scopes

Where Placed	Affects
Above **`Feature`**	All scenarios in that feature
Above **`Scenario`**	Only that specific scenario

✅ Use Cases

- **`@SmokeTest`** – for quick checks
- **`@Regression`** – for full regression suite
- **`@WIP`** – for work in progress scenarios (you can skip them temporarily)
- **`@Negative`** – for negative test cases

✅ Best Practices

- Use consistent naming (e.g., camelCase or snake_case).
- Group related scenarios using meaningful tags.
- Avoid over-tagging. Use tags for **execution control**, not for documentation.
- Combine tags logically using AND, OR, and NOT in the command line.

Would you like an example using **Java + Cucumber + TestNG or JUnit** with tag filtering?

To **run specific tagged tests** in **Cucumber**, you'll use the `--tags` option (for CLI) or configure tags in your test runner file (for Java with JUnit/TestNG). Here's how it works depending on your setup:

🖥️ 1. Using Command Line (Generic Cucumber CLI)

✅ Run tests with a single tag:

```
bash

cucumber --tags @SmokeTest
```

✅ Run tests with multiple tags (AND):

```
bash

cucumber --tags "@SmokeTest and @Login"
```

✅ Run tests with either tag (OR):

```
bash

cucumber --tags "@SmokeTest,@Regression"
```

✅ Exclude a tag:

```
bash

cucumber --tags "not @WIP"
```

☕ 2. Java + Cucumber + JUnit Example

✨ Sample Runner File (JUnit 5):

```
java

import org.junit.platform.suite.api.Cucumber;
import org.junit.platform.suite.api.CucumberOptions;
import org.junit.platform.suite.api.SelectClasspathResource;

@Cucumber
@SelectClasspathResource("features")
@CucumberOptions(
    plugin = {"pretty", "html:target/cucumber-report.html"},
    tags = "@SmokeTest"  // <--- run only @SmokeTest tagged scenarios
)
public class RunCucumberTest {
}
```

✅ Run via Maven:

```
bash

mvn test
```

🧪 3. Java + Cucumber + TestNG Example

✨ Sample Runner Class:

```
java

import io.cucumber.testng.AbstractTestNGCucumberTests;
import io.cucumber.testng.CucumberOptions;

@CucumberOptions(
    features = "src/test/resources/features",
    glue = "stepdefs",
    tags = "@Regression",
    plugin = {"pretty", "html:target/cucumber-html-report.html"}
)
public class TestRunner extends AbstractTestNGCucumberTests {
}
```

🏷️ Multiple Tags in Test Runner

- AND: **`@Tag1 and @Tag2`**
- OR: **`@Tag1 or @Tag2`** (JUnit 5)
- NOT: **`not @Tag1`**

Example:

```
java

tags = "@Login and not @WIP"
```

✅ Summary

Goal	Syntax
Run specific tag	**`--tags @SmokeTest`**
AND condition	**`--tags "@Tag1 and @Tag2"`**
OR condition	**`--tags "@Tag1,@Tag2"`**
Exclude a tag	**`--tags "not @TagName"`**

Would you like a complete **sample project structure with tags** or **step-by-step runnable example**?

You're referring to a **tag expression** like:

```
less

@Smoke and @Regression and not @Sanity
```

This tag expression means:

> **Run only those scenarios that are tagged with both `@Smoke` AND `@Regression`, but exclude any that are also tagged with `@Sanity`.**

✅ Example in Gherkin

```
gherkin

@Smoke @Regression
Scenario: Valid login
  Given user is on login page
  When user enters valid credentials
  Then user should see dashboard

@Smoke @Regression @Sanity
Scenario: Logout
  Given user is logged in
  When user clicks logout
  Then user is redirected to login page

@Smoke
Scenario: Forgot password
  Given user is on login page
  When user clicks forgot password
  Then user receives a reset link
```

🪛 Result of using: `@Smoke and @Regression and not @Sanity`

✅ Executed:

- **`Valid login`** → has **`@Smoke`** and **`@Regression`**, not **`@Sanity`**

❌ Skipped:

- **`Logout`** → has **`@Smoke`**, **`@Regression`**, and **`@Sanity`** → excluded
- **`Forgot password`** → only **`@Smoke`** → excluded (missing **`@Regression`**)

🛠️ How to Use This Expression

🖥️ In CLI (Cucumber command line):

```
bash

cucumber --tags "@Smoke and @Regression and not @Sanity"
```

☕ In Java JUnit/TestNG Runner File

✅ **JUnit:**

```
java

@CucumberOptions(
    features = "src/test/resources/features",
    glue = "stepdefs",
    tags = "@Smoke and @Regression and not @Sanity"
)
```

✅ **TestNG:**

```
java

@CucumberOptions(
    features = "src/test/resources/features",
    glue = "stepdefs",
    tags = "@Smoke and @Regression and not @Sanity"
)
```

Would you like me to generate a **full working sample feature file and runner class** based on this tag expression?

Here's a **complete working sample project** demonstrating:

✅ A **feature file** with various tagged scenarios
✅ A **Cucumber test runner** using `@Smoke and @Regression and not @Sanity`
✅ Sample **step definitions** (Java)

🗂️ Project Structure (Maven-style)

```
pgsql

project-root/
├── src/
│   └── test/
│       ├── java/
│       │   └── stepdefs/
│       │       └── StepDefinitions.java
│       └── resources/
│           └── features/
│               └── login.feature
├── pom.xml
└── TestRunner.java
```

📝 1. `login.feature`

📍 **`src/test/resources/features/login.feature`**

```
gherkin

@Smoke @Regression
Feature: Login functionality

  @Smoke @Regression
  Scenario: Valid login
    Given user is on login page
    When user enters valid credentials
    Then user should see dashboard

  @Smoke @Regression @Sanity
  Scenario: Logout
    Given user is logged in
    When user clicks logout
    Then user is redirected to login page

  @Smoke
  Scenario: Forgot password
    Given user is on login page
    When user clicks forgot password
    Then user receives a reset link
```

💡 2. `TestRunner.java`

📍 **`src/test/java/TestRunner.java`**

```
java

import org.junit.platform.suite.api.Cucumber;
import org.junit.platform.suite.api.CucumberOptions;
```

```
import org.junit.platform.suite.api.SelectClasspathResource;

@Cucumber
@SelectClasspathResource("features")
@CucumberOptions(
    plugin = {"pretty", "html:target/cucumber-report.html"},
    glue = "stepdefs",
    tags = "@Smoke and @Regression and not @Sanity"
)
public class TestRunner {
}
```

This runs only scenarios with both **`@Smoke`** **and** **`@Regression`**, **excluding** **`@Sanity`**.

✅ 3. `StepDefinitions.java`

📍 **`src/test/java/stepdefs/StepDefinitions.java`**

```
java

package stepdefs;

import io.cucumber.java.en.*;

public class StepDefinitions {

    @Given("user is on login page")
    public void user_on_login_page() {
        System.out.println("User is on login page");
    }

    @When("user enters valid credentials")
    public void user_enters_valid_credentials() {
        System.out.println("User enters valid credentials");
    }

    @Then("user should see dashboard")
    public void user_sees_dashboard() {
        System.out.println("User sees dashboard");
    }

    @Given("user is logged in")
    public void user_is_logged_in() {
        System.out.println("User is logged in");
    }

    @When("user clicks logout")
    public void user_clicks_logout() {
        System.out.println("User clicks logout");
    }

    @Then("user is redirected to login page")
    public void user_redirected_to_login_page() {
        System.out.println("User redirected to login page");
    }

    @When("user clicks forgot password")
    public void user_clicks_forgot_password() {
        System.out.println("User clicks forgot password");
    }

    @Then("user receives a reset link")
    public void user_receives_reset_link() {
        System.out.println("User receives a reset link");
    }
}
```

📦 4. `pom.xml` (Cucumber + JUnit 5)

Make sure you include these dependencies:

```xml
<dependencies>
    <dependency>
        <groupId>io.cucumber</groupId>
        <artifactId>cucumber-java</artifactId>
        <version>7.16.1</version>
    </dependency>
    <dependency>
        <groupId>io.cucumber</groupId>
        <artifactId>cucumber-junit-platform-engine</artifactId>
        <version>7.16.1</version>
    </dependency>
    <dependency>
        <groupId>org.junit.platform</groupId>
        <artifactId>junit-platform-suite</artifactId>
        <version>1.10.2</version>
    </dependency>
</dependencies>
```

🚀 Run the Test

Run using your IDE or Maven:

```bash
mvn test
```

✅ Only this scenario will run:

- **`Valid login`** — has **`@Smoke`** and **`@Regression`** and not **`@Sanity`**

Would you like this zipped up as a downloadable Maven project?