Coding Bootcamp

Cucumber

TECHNOLOGY

Introduction to Cucumber and Behavior-Driven Development (BDD)

simplilearn

Learning Objectives

By the end of this lesson, you will be able to:

- Explain Cucumber and its benefits through clear examples for Behavior-Driven Development (BDD)
- Develop feature files with clear descriptions using Gherkin syntax to ensure tests are understandable
- Explain step definitions and their role in linking Gherkin steps with executable code to understand their importance in Cucumber
- Use Gherkin syntax to write precise feature files for effective test automation
- Install the Cucumber Eclipse plugin to ensure a functional testing environment

TECHNOLOGY

Cucumber: Introduction

What Is Cucumber?

Cucumber is a behavior-driven development (BDD) tool. It has the following characteristics:

It helps bridge the communication gap between development and business teams by using a common language for defining requirements and tests.

Advantages of Cucumber

Runs on Java, .NET, and Ruby languages

Offers an end-to-end testing framework

Bridges the gap between technical language and enterprise language

Provides code reusability

Allows writing scripts without needing programming knowledge

Enables fast and easy execution and integration

Behavior-Driven Development (BDD): Introduction

What Is BDD?

It is an approach to software development that builds upon Test-Driven Development (TDD) by focusing on collaboration between developers, QA, and non-technical stakeholders. The following are the benefits of BDD:

Aligns the perspectives of engineers and clients

Is written in plain English

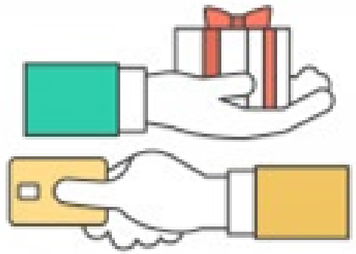
Creates communication between technical and non-technical teams

How Does BDD Work in Cucumber Automation?

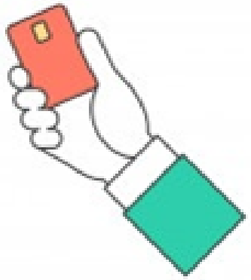
Cucumber BDD follows the Given-When-Then steps to create functions and test code effectively. To illustrate the process, consider a scenario for a payment module in a food delivery application.

Payment can be made through various methods:

Bank account



Cash on delivery or online payment



Net banking, UPI, credit, or debit card

BDD: Example

Given:

- A payment module in a food ordering application has been developed.
- I am accessing it with proper login credentials for authentication.

When:

- I make a payment with sufficient funds in my bank account.
- I make a payment using different modes of payment.
- I make a payment through UPI.
- The company's payment interface is operational.
- The transaction one-time password authentication is correct.
- I press or click the pay button.

Then:

- The amount must be transacted.
- The event will be recorded in the log file.

Characteristics of BDD

Reduces the cost of maintenance and project risk

Ensures collaborative effort among stakeholders

Enables teams to be more confident

Improves the clarity and readability of tests

Provides a common understanding of requirements

Places a high value on commercial enterprise values and desires

Feature File on Cucumber Testing

It is a file written in plain text that describes the software's features and scenarios to be tested. It is a key component of the Cucumber tool, which is:

- Essential for producing files that will pass all automation tests
- A standard file that stores the scenarios and descriptions to be tested
- The starting point for writing Cucumber tests
- A live document during the testing process

Note

It implements BDD principles, ensuring clarity and collaboration in automated testing. The feature file's extension is **.feature**.

TECHNOLOGY

Introduction to Gherkin Language

simplilearn

Gherkin Language

Gherkin is a programming language used in Cucumber that acts as documentation and executable tests, bridging the gap between specifications and implementation. It:

Describes business behavior

Provides a standard set of keywords in English text

Gherkin Language

Gherkin is a plain English text language used to interpret and execute test scripts.

Gherkin also supports other international and local languages, such as:

- French
- Finnish
- Indonesian
- Hungarian
- Hindi
- Urdu
- Gujarati

This multilingual support ensures that teams worldwide can write test scripts in their native language, enhancing comprehension and collaboration across diverse teams.

Why Use Gherkin?

Read these statements and identify the most testable ones:

Statement 1

Customers are not allowed to enter invalid credit card information.

Direct without any test data

Statement 2

If a consumer inputs a credit card number that is not precisely 16 digits long while submitting a form, an error notice with the right number of digits should appear.

Contains test data, and looks much more testable

Gherkin creates more specific requirements for a test case.

Gherkin Syntax

Gherkin is a line-oriented language, just like YAML and Python. Cucumber reads Gherkin and runs the test to ensure that the software follows Gherkin's syntax.

Test scenario structure:

Feature: Title of the scenario
Given [Preconditions or Initial Context]
When [Event or Trigger]
Then [Expected Output]

- **Keywords:** Use keywords like Feature, Given, When, and Then to structure the scenarios
- **Indentation:** Use the Tab key to provide space for better readability
- **Comments:** Begin a comment in a script with a # sign

Gherkin Terms

Gherkin Terms

The commonly used terms or keywords used in Gherkin are as follows:

Feature	Given	And
Background	When	But
Scenario	Then	Scenario outline

Gherkin Terms

Feature

The Feature keyword in Gherkin describes a specific functionality or feature of the software.

Background

The Background keyword allows users to define a set of steps that are common to all scenarios in a feature.

Scenario

A Scenario represents how a user interacts with the feature under different conditions.

Gherkin Terms

Given

Allows placing the system in a familiar state

Syntax:

Given– a test step that defines the context

Given I am on “HomePage.”

When

Defines the actions to be performed

Syntax:

A When– a test step that defines the action performed

When I perform “Registration.”

Gherkin Terms

Then

Allows validating the outcome

Syntax:

Then– test step that defines the ‘outcome.’

Then I should see “Happy Birthday.”

But

Adds false conditions to the steps

Syntax:

A But- additional test step which defines the ‘action’ ‘outcome.’

But I should see “Happy Birthday.”

Gherkin Terms

And

Adds more conditions to the steps

Syntax:

And– additional test step which defines the ‘action’ performed

And I write “CityName.” with “NewYork”

Scenario Outline

Identifies the parameter name symbols, "<" and ">”

Given, When, Then, And, and But are often used interchangeably.

Gherkin Terms: Examples

Gherkin identifies every step written within the step definition document.

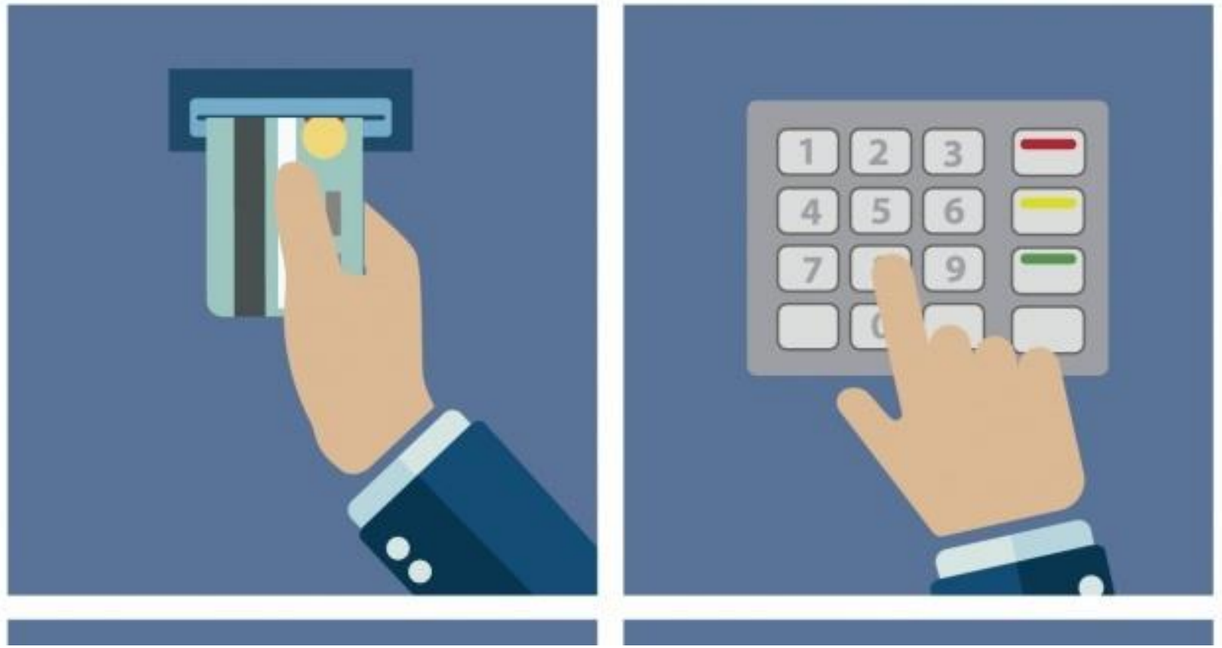
Example:

Feature: Login functionality of social network site Facebook

Given: I am a Facebook user.
When: I enter a *username* as the username.
And: I enter a *password* as the password
Then: I should be redirected to the home page of Facebook.

Gherkin Scenario

When submitting a form, if the customers enter their ATM card pin that is not four digits long, the error message should show the correct number of digits.



Gherkin Scenario

The following scenario is written in Cucumber using Gherkin terms:

Feature: Message on entering an invalid ATM pin

Background
Given that I have decided to withdraw some money
And I insert my debit card into the ATM
And I enter my debit card pin

Scenario
When I enter a PIN of fewer than four digits
And the other details are correct
And I submit the form
Then, the form must be redisplayed
And I should see a notification informing me of the correct digits

Best Practices for Using Gherkin

The best practices for using Gherkin are as follows:

- Each situation needs to be executed one at a time.
- Every characteristic should be able to be completed alongside.
- Steps statistics should be proven independently.
- The scenarios should relate to the requirements.

Best Practices for Using Gherkin

The best practices for using Gherkin are as follows:

- The whole track of scenarios should be covered in a demand report.
- Steps should be modular and smooth.
- All the common situations need to be executed.

Advantages and Disadvantages of Gherkin

Advantages of Gherkin

Enables stakeholders to understand and participate in the process

Converts requirements into executable tests easily

Ensures clarity for all team members

Aligns development with business objectives

Advantages of Gherkin

Integrates real-world feedback

Ensures all aspects meet standards

Encourages code reuse and reduces redundancy

Communicates use cases clearly

Disadvantages of Gherkin

While Gherkin offers many benefits, it also has some disadvantages that need to be considered:

It fails in
certain scenarios.

It has high
test-maintenance costs.

Step Definitions

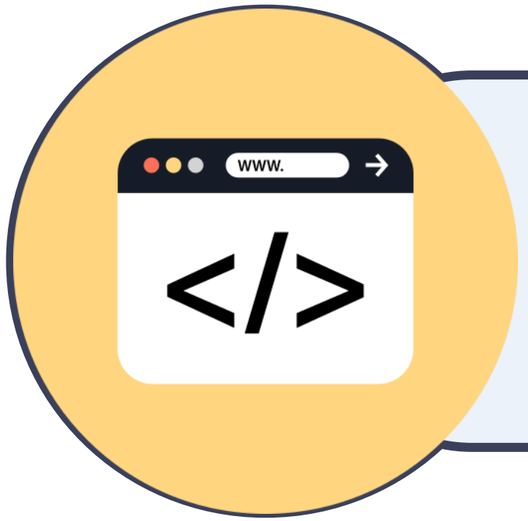
What Is Step Definition?

It is a small code that has an example attached to it or a Java technique in a class with an explanation.

1. It is connected by a comment followed by an example of the coordinates with steps.
2. Cucumber executes the code when it encounters a Gherkin step.
3. Cucumber validates the step definition report with the help of the glue code.

Step Definitions

A step definition is a Java method that includes an articulation that connects it to at least one Gherkin step.



Cucumber finds the corresponding step definition and executes it in the scenario context. This coordination between Gherkin steps and step definitions enables the automation of behavior-driven development (BDD) tests.

Step Definitions: Examples

Here is an example of a step definition using Java 8 lambdas to demonstrate how Gherkin steps are connected to executable code:

Scenario: Some food items

Given: I have 40 food items in my shopping basket.

The “I have 40 food items in my shopping basket” part of the step will match the following definition:

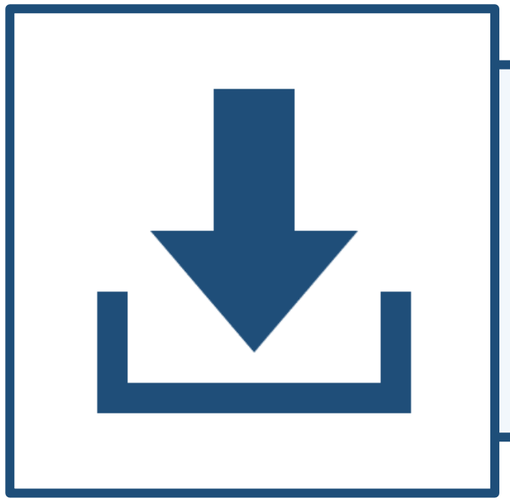
Using Java8 lambdas:

```
package com.example;
import io.cucumber.java8.En;
public class StepDefinitions implements En {
    public StepDefinitions() {
        Given("I have {int} food items in my shopping basket", (Integer
items) -> {
            System.out.format("Food Items: %n\\n", items);
        });
    }
}
```

Installing Cucumber Eclipse Plugin

Steps to Install Cucumber Eclipse Plugin

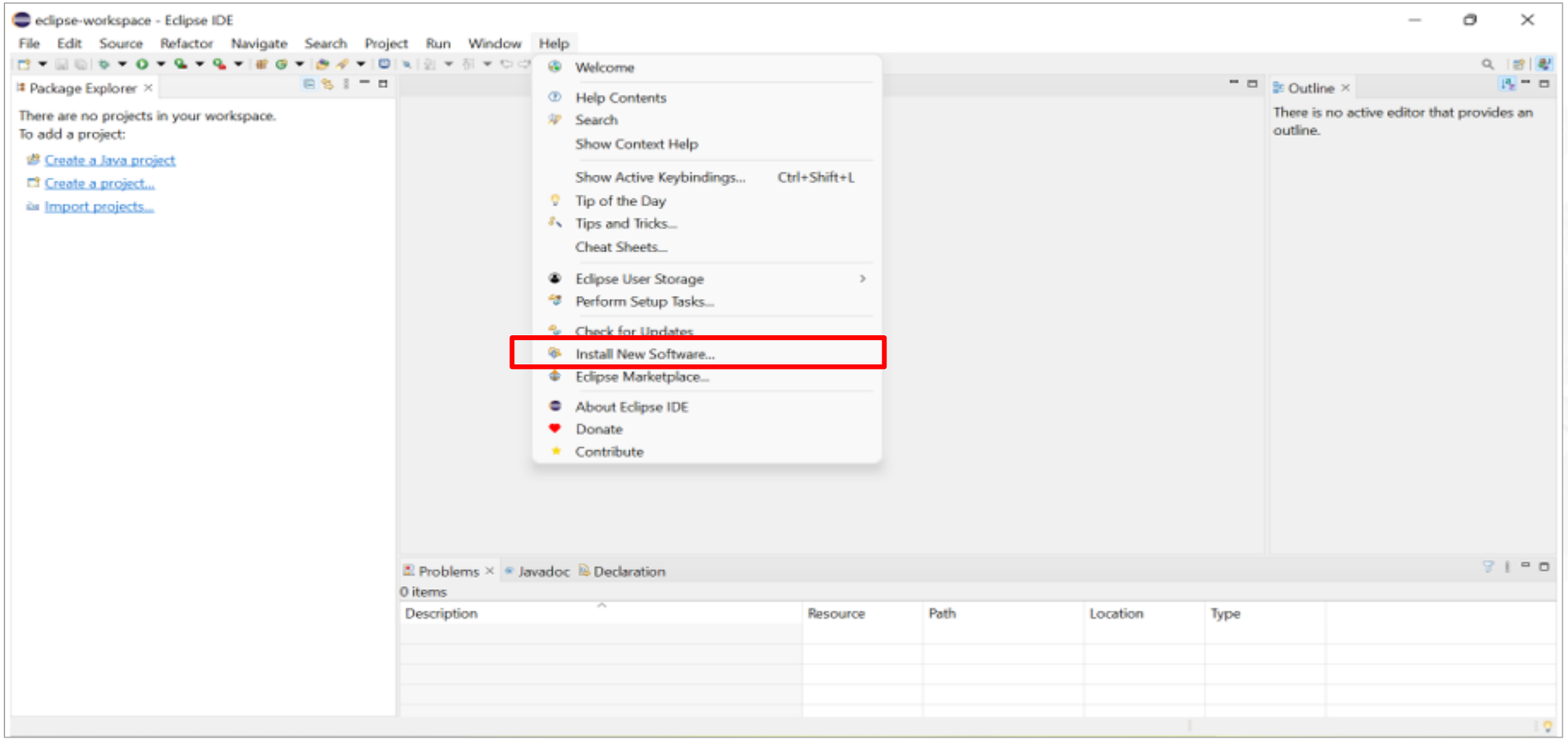
The Cucumber Eclipse plugin is available as a module for the Eclipse IDE.



Ensure the internet connection is fully operational and Eclipse IDE is installed on the PC.

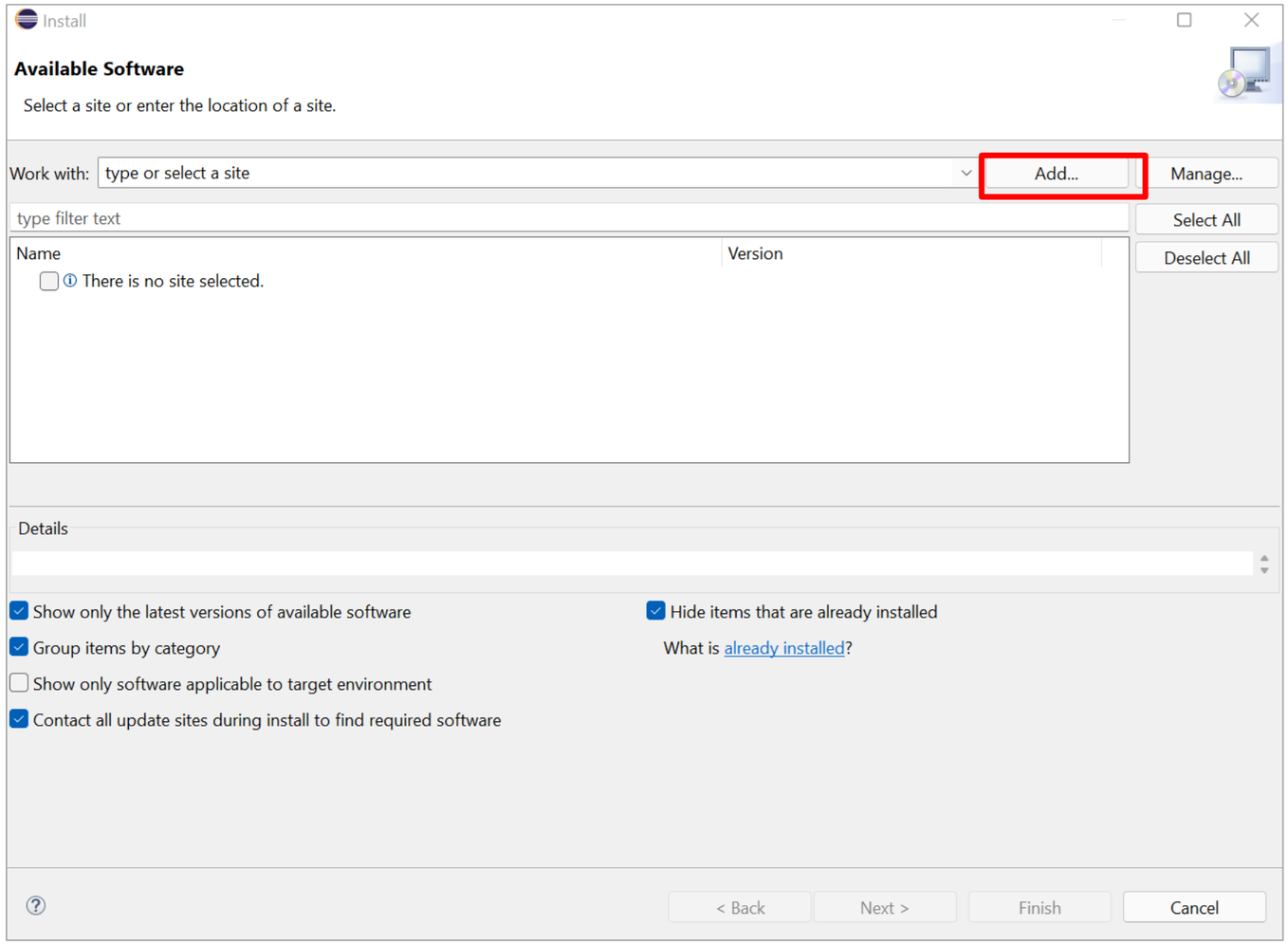
Steps to Install Cucumber Eclipse Plugin

Step 1: Open the Eclipse IDE and click **Install New Software** from the **Help** menu.



Steps to Install Cucumber Eclipse Plugin

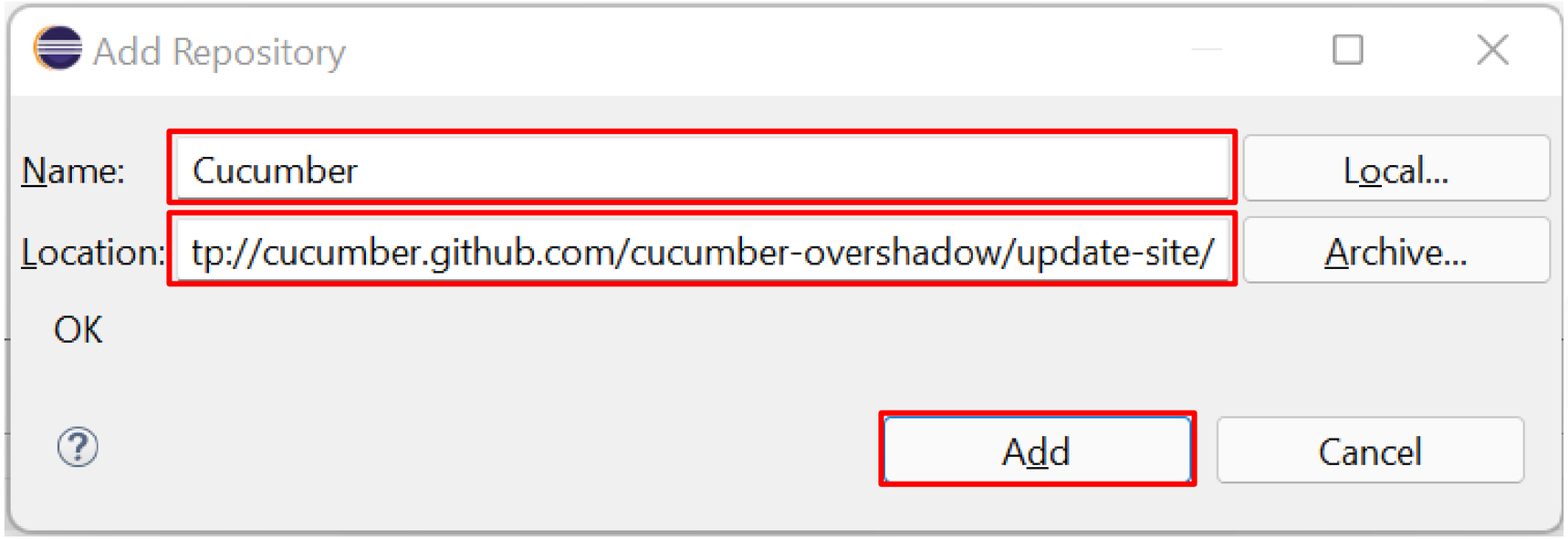
Step 2: A dialog window will open; click the **Add** button.



Steps to Install Cucumber Eclipse Plugin

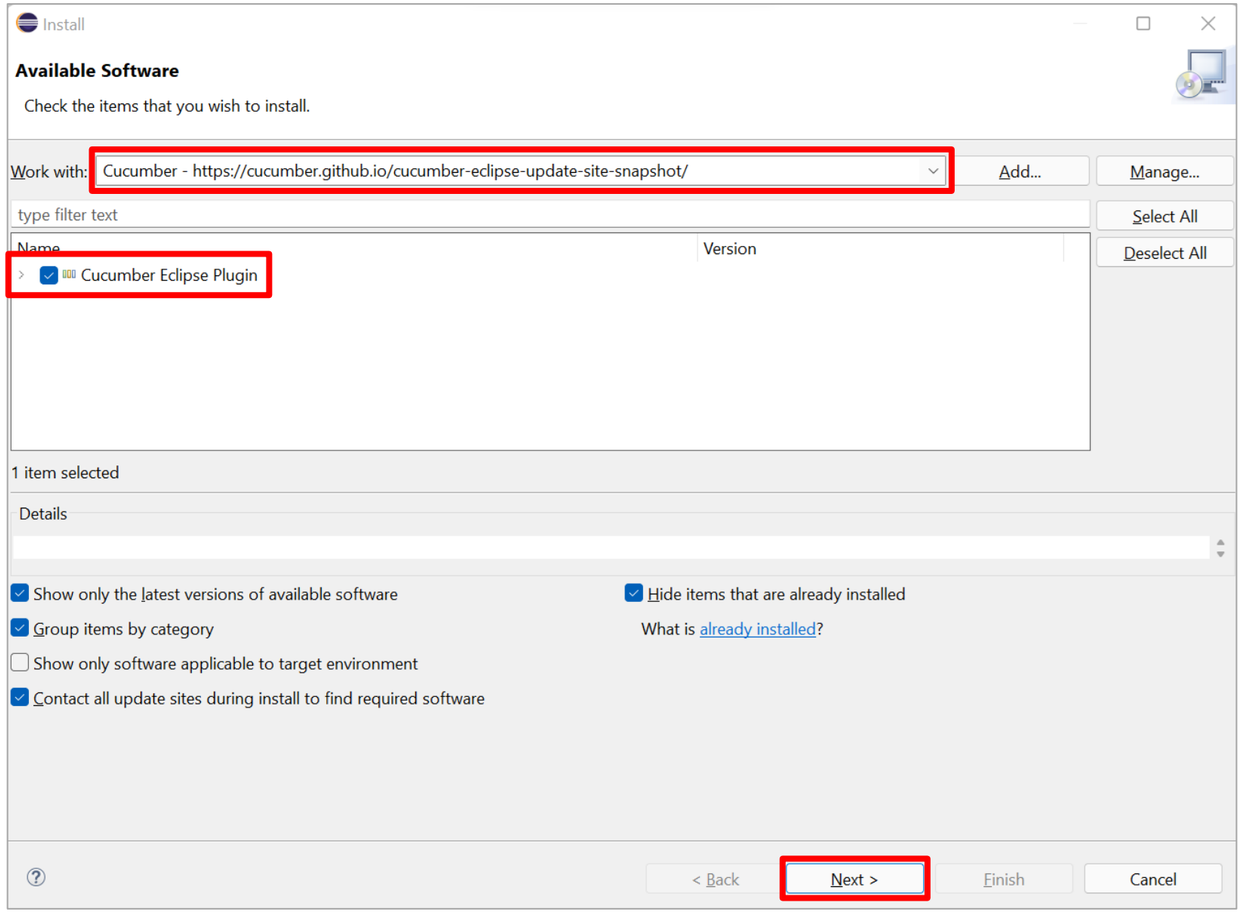
Step 3: Give any name and type **http://cucumber.github.com/cucumber-overshadow/update-site** in the text box.

Then, click **OK.**



Steps to Install Cucumber Eclipse Plugin

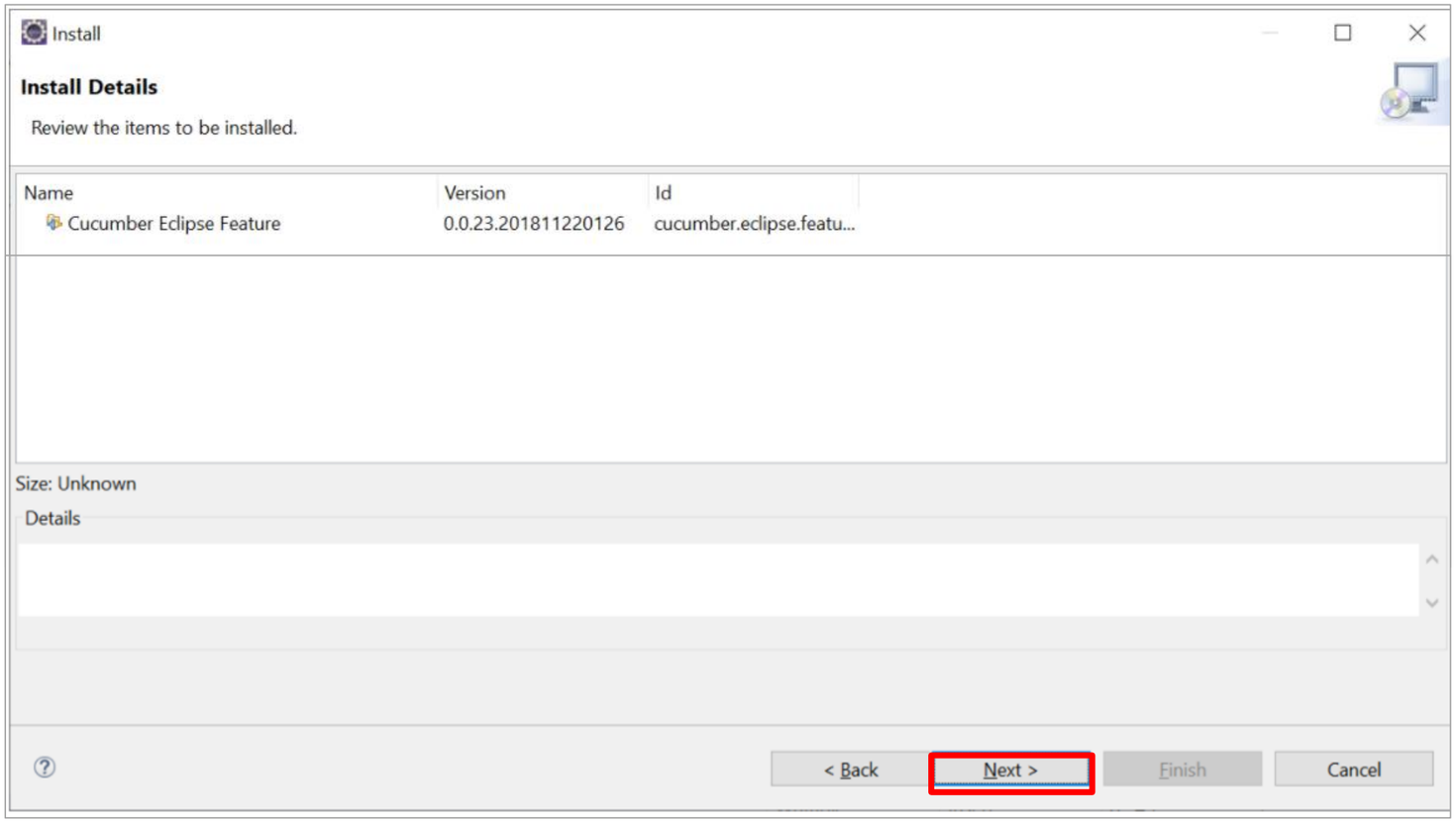
Step 4: Click the **Cucumber Eclipse Plugin** option in the available software list.



Check the box and then click on the **Next** button.

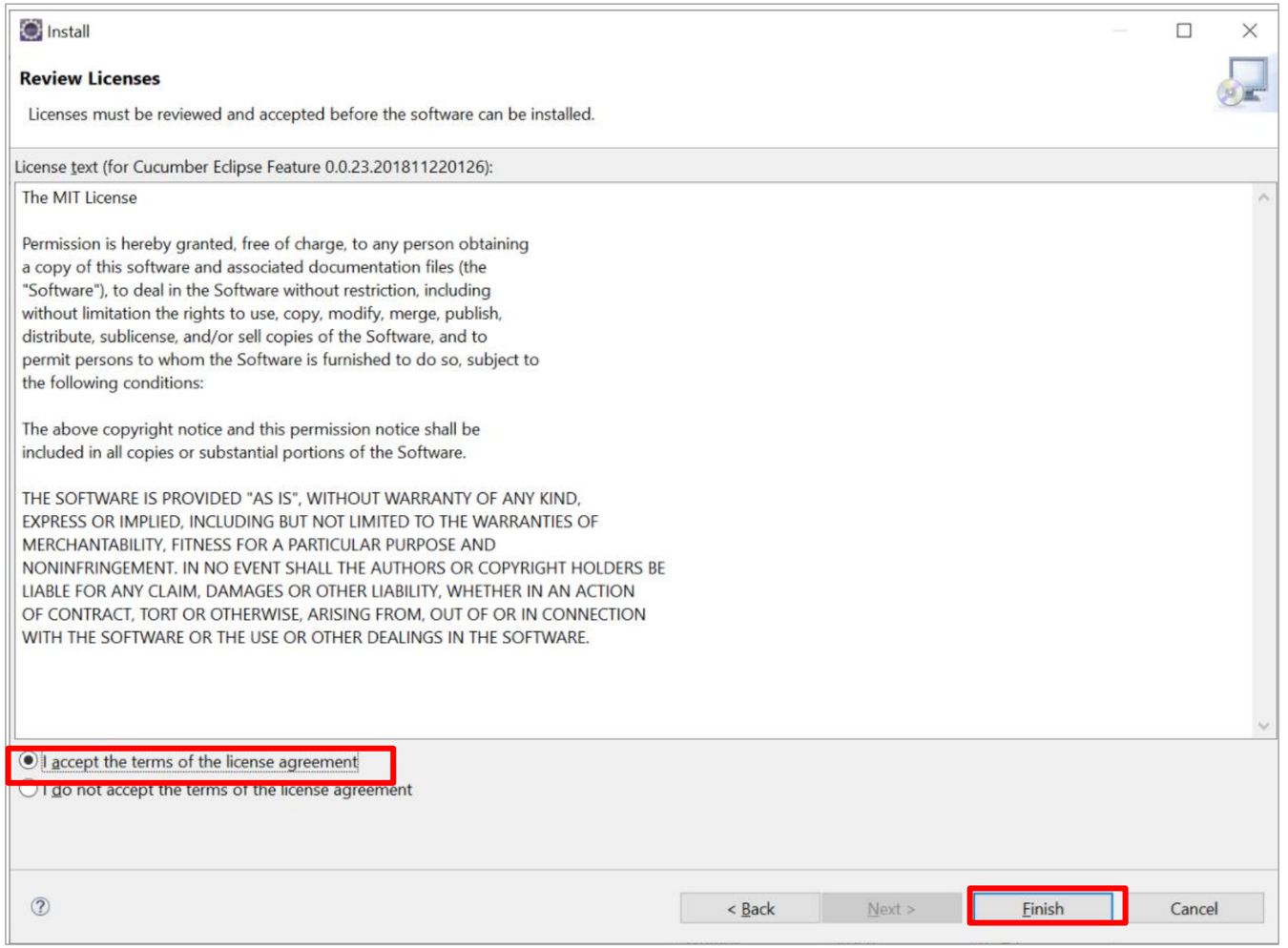
Steps to Install Cucumber Eclipse Plugin

Step 5: Click **Next.**



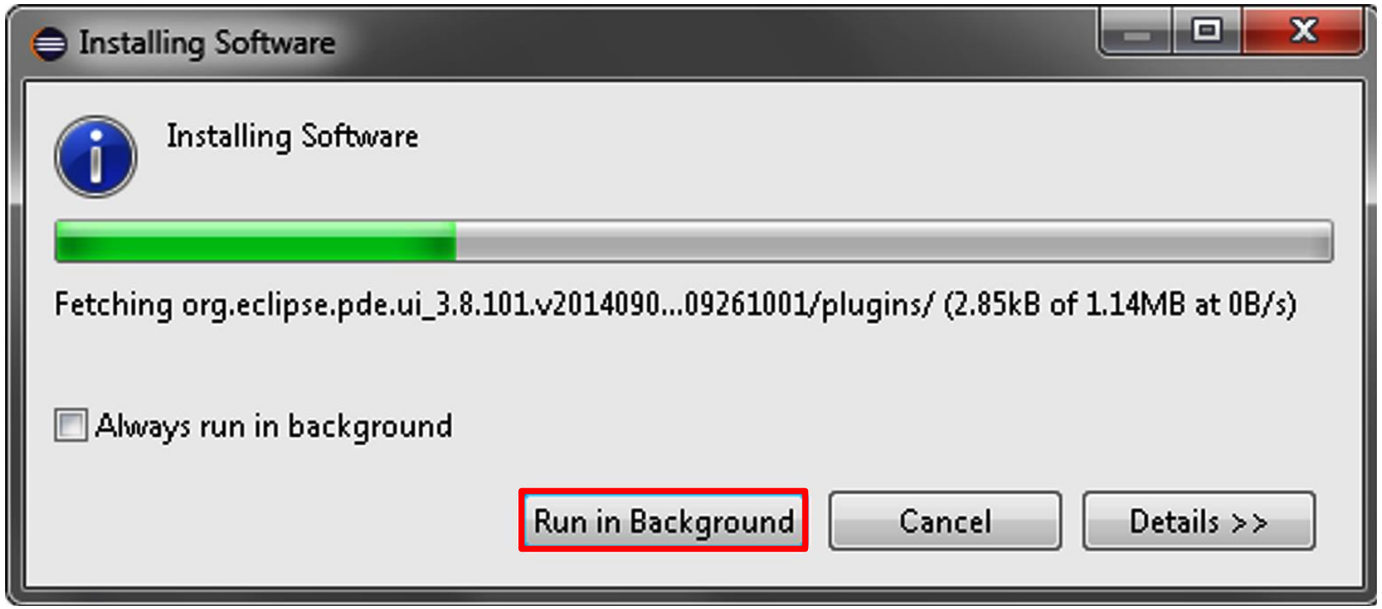
Steps to Install Cucumber Eclipse Plugin

Step 6: Accept the terms and click **Finish.**



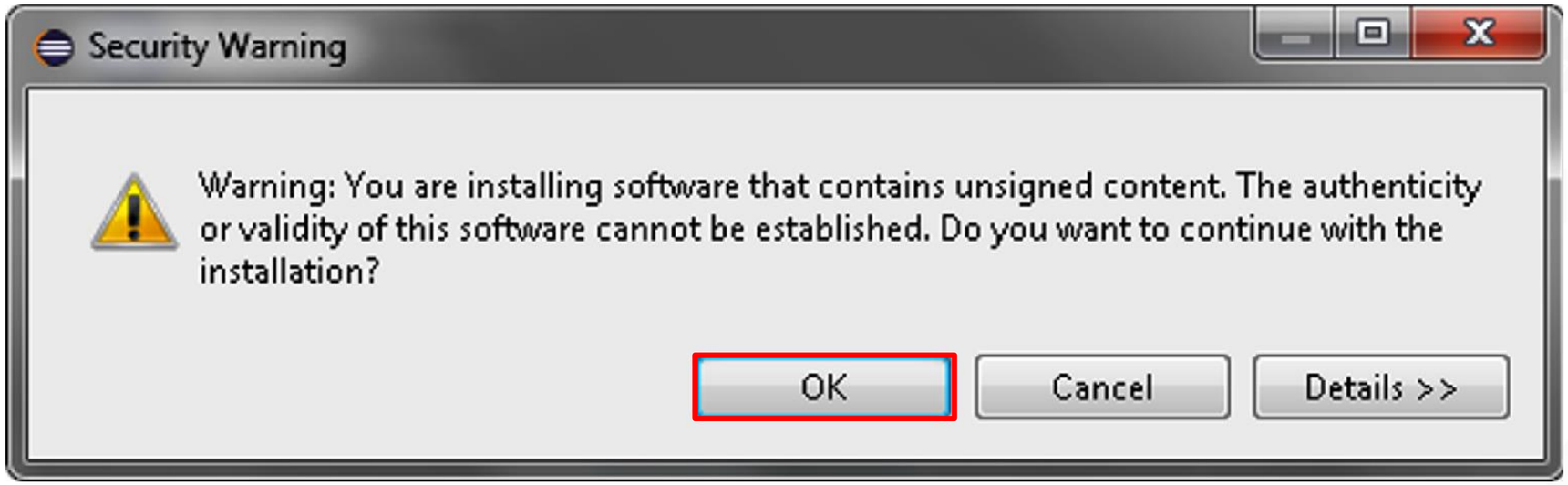
Steps to Install Cucumber Eclipse Plugin

Step 7: Click **Run in Background** and let it install.



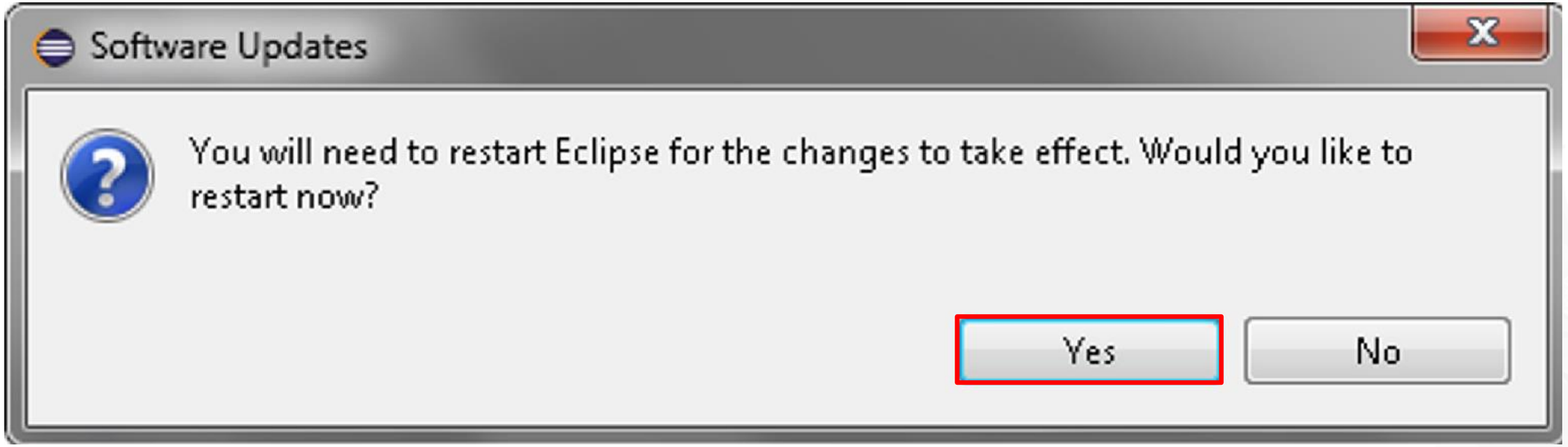
Steps to Install Cucumber Eclipse Plugin

Step 8: If a security pop-up is displayed, click **OK.**



Steps to Install Cucumber Eclipse Plugin

Step 9: It is all set; click **Yes.**



Key Takeaways

- Cucumber enables the writing of test scenarios in plain language, making them accessible to both technical and non-technical stakeholders.
- BDD extends TDD by emphasizing collaboration among developers, QA, and non-technical stakeholders. It focuses on the application's behavior from the user's perspective.
- Gherkin uses a line-oriented syntax like YAML and Python to write clear and precise test scenarios in plain English.
- Feature files are crucial in Cucumber. They provide a common language for defining application behavior and bridge stakeholder communication gaps.
- Step definitions link Gherkin's steps to executable code, ensuring that the behavior specified in scenarios is correctly validated during test execution.

TECHNOLOGY

Thank You

simplilearn