InterviewBit

Cucumber Interview Questions

To view the live version of the page, click here.

Contents

Cucumber Interview Questions for Freshers

Cucumber Interview Questions for Experienced

20. Explain briefly how Behavioral Driven Development works?
21. What is a test harness in the context of Cucumber?
22. What is the difference between RSpec and Cucumber?
23. Difference between Selenium and Cucumber.
24. Why do we need to use Cucumber with Selenium?
25. In a feature file, what is the maximum number of scenarios?
26. What do you mean by Test Driven Development (TDD)?
27. Difference between TDD and BDD.
28. What is the use of the Options tag in the Cucumber Framework?
29. How does the execution start in Cucumber?
30. What is grouping in the context of Cucumber?
31. How can you run Cucumber tests parallelly?
32. What are some of the prerequisites that you should consider while building a Selenium Cucumber automation application?
33. Difference between JBehave and Cucumber.
34. How can you run a selected test from a group of tests in Cucumber?

Let's get Started

Cucumber is a behavior-driven development (BDD) testing tool. It provides a method for writing tests that anyone, regardless of technical knowledge, can comprehend. We can write test scripts from both the developer's and the customer's perspectives with Behavior Driven Development.

The Ruby programming language was originally used in the creation of the Cucumber framework. Cucumber now supports a variety of programming languages, including Java and JavaScript, through several implementations. SpecFlow is the name of the open-source Cucumber port for .Net.

In this article, we will cover the most frequently asked interview questions on Cucumber.

Cucumber Interview Questions for Freshers

1. What is Cucumber? Explain the need of using Cucumber.

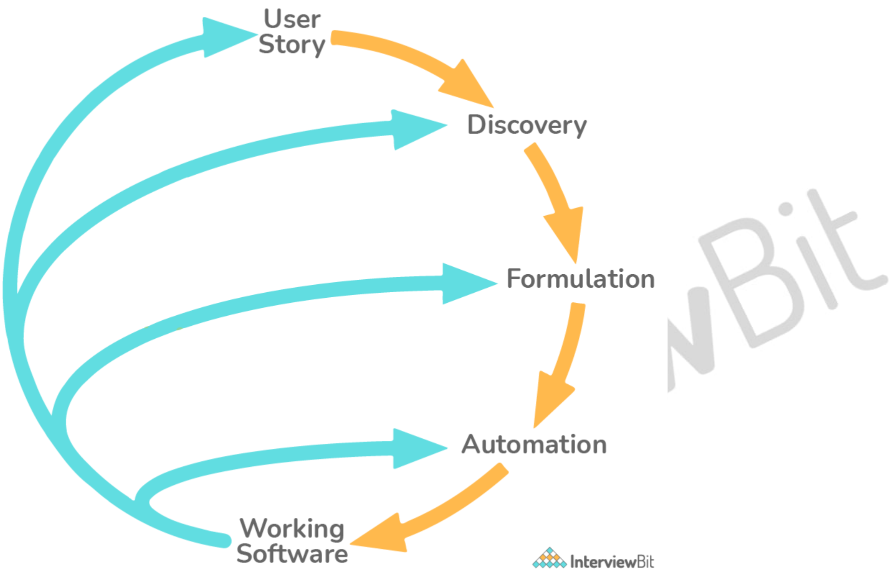
Cucumber is a behavior-driven development (BDD) testing tool. The BDD framework's major goal is to bring together a variety of project responsibilities, such as quality assurance, developers, and business analysts, to understand the application without diving too deeply into the technical components.
Testers use Cucumber to create test cases for evaluating program behaviour. It is an important tool to automate acceptance tests written in logical language that customers can understand. It's primarily used to develop acceptance tests for web apps based on their features' behaviour.

2. What is Gherkin Language?

Gherkin is a readable business language that allows you to define business activity without getting bogged down in implementation specifics. It's a domain-specific language for defining specs tests in Cucumber format. It describes use cases in plain English and helps users to remove logic elements from behaviour testing.

3. What is the principle of Behaviour Driven Development?

Behaviour Driven Development (BDD) is a synthesis and refinement of practices stemming from Test Driven Development (TDD) and Acceptance Test-Driven Development (ATDD). BDD augments TDD and ATDD by
applying the “Five Why’s” principle to each proposed user story so that its purpose is clearly related to business outcomes. Five Why's is an iterative interrogative approach for uncovering the cause-and-effect links at the root of a problem. The main purpose of this technique is to uncover the core cause of a flaw or problem by asking "Why?" repeatedly. Each response serves as the foundation for the next question.



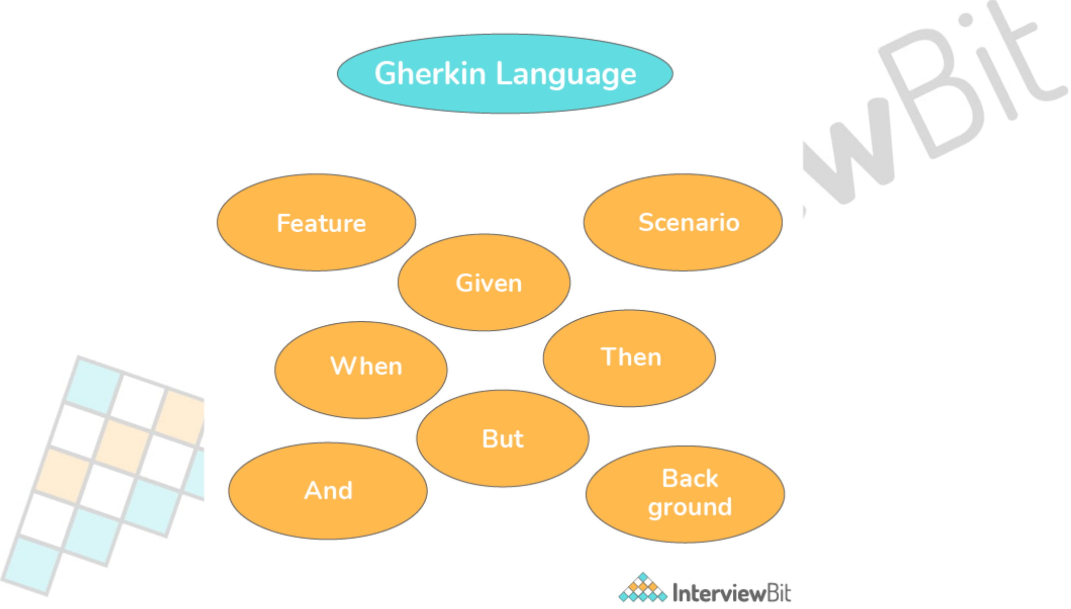
4. What are the primary keywords in Cucumber?

Following are the primary keywords in Cucumber:-

- **Feature:** The Feature keyword's aim is to collect relevant scenarios and provide a high-level description of a software feature.
- **Rule:** The Rule keyword is used to express a single business rule that should be followed. It adds to the information about a feature.
- **Example:** This is a practical illustration of a business rule. It comprises a series of steps.
- **Given:** The given steps are used to describe the system's initial context - the scenario's scene. It usually refers to an event that occurred in the past.
- **When:** When describing an occurrence or an action, When is employed. It could be a user interacting with the system or an event generated by another system.
- **Then:** Then steps are employed to indicate an anticipated outcome, or result.
- **Background:** A background helps you to give the situations that follow it some context. It can have one or more Given steps, which are executed prior to each scenario but after any Before hooks.

5. Which language is used in Cucumber?

Cucumber understands Gherkin. It's a straightforward English representation of the app's functionality. It is used for defining test cases. It is intended to be non-technical and human-readable, and it describes use cases for a software system as a whole. It's a domain-specific (DSL), business-friendly language.



6. What do you mean by scenario in Cucumber Testing?

Scenario is a fundamental Gherkin structure. Every scenario begins with the keyword "Scenario:" (or a localized version of it) and ends with a scenario title. Every feature can have one or more scenarios, each of which has one or more steps.

As an example of a scenario, consider the following:

Scenario – Verify My Orders Functionality.
Explanation: When a user clicks on the My Orders option he/ she should be taken to the My Orders page.

7. What do you mean by Scenario Outline?

Consider the situation when we need to run a test scenario multiple times. Assume we need to ensure that the login feature is functional for all types of subscribers. This necessitates repeating the login functionality scenario. Copying and pasting the identical instructions to just re-run the code does not appear to be a good approach. Gherkin adds another framework, the scenario outline, to help with this. The scenario outline is similar to scenario, with the exception that several inputs are provided.
Example:-

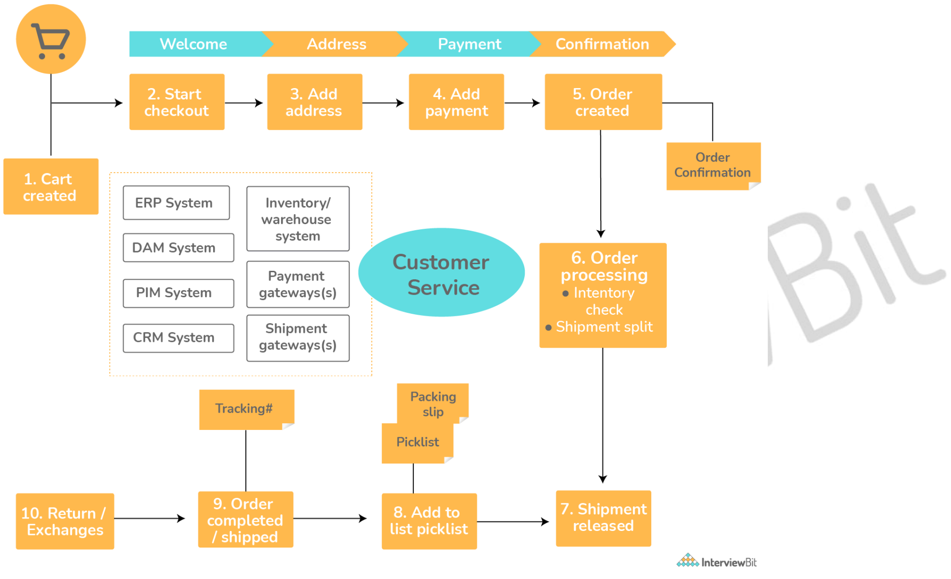
Scenario Outline - Sign In Feature for a website.
Explanation: The website can have multiple users and so we need to consider all the users while implementing the sign-in functionality.

8. What do you mean by feature in Cucumber?

A project's feature can be described as a stand-alone unit or functionality. A list of scenarios to test for a feature is frequently included with it. The Feature File is a file in which we store features, descriptions of features, and situations to be evaluated. For each feature under test, it is recommended that a separate feature file be created. The feature file must have the extension ".feature." You can make as many feature files as you want.
Here's an example:-

For an e-commerce website, we can have the following features:-

- User registers and signs up on the website.
- User tries to log in to their account using their credentials.
- Users add a product to their cart.
- User clicks on checkout now.
- User pays for their items.
- User logs out from the website.



All these are different features. The website will have many such features. All these features will have a separate Feature File.

9. What are the basic requirements to run Cucumber Web test cases?

We need the following minimum requirements to successfully run a Cucumber Web test case:-

1. The compiler and the development kit for the programming language we will be using. Example: JDK and JRE for using Java as our programming language.
2. An IDE (Integrated Development Environment) wherein we can write our code. Example: Eclipse.
3. Build tools so as to do tasks such as compiling code, packaging code to a jar, creating source code. Example: Maven, Gradle.

10. What are the advantages of using Cucumber?

Following are the advantages of using Cucumber:-

- Cucumber supports a variety of programming languages, including Java.net and Ruby.
- It serves as a link between commercial and technical language. This can be done by writing a test case in plain English text.
- It enables the test script to be developed without any prior knowledge of programming, as well as the participation of non-programmers.
- Unlike other tools, it functions as an end-to-end test framework.
- Cucumber allows for code reuse thanks to its simple test script architecture.

11. What are Step Definitions in the context of Cucumber?

Step definitions connect Gherkin steps to programming code. The mapping between each step of the scenario defined in the feature file and a code of the function to be executed is stored in the steps definition file. A step definition carries out the action that should be performed by the step. So step definitions hard-wire the specification to the implementation.

12. What are annotations in Cucumber?

An annotation is a type of text that has been pre-defined and has a specified meaning. It tells the compiler/interpreter what to do when the program runs. The annotations on Cucumber are as follows:

- **Given:** It specifies the requirements for running the test.
  Example: Given I have an account on Interviewbit.
- **When:** It establishes the starting point for any test scenario.
  Example: When I log in to Interviewbit.
- **Then:** It contains the expected result of the test which is to be executed.
  Example: Then registration should be successful.
- **And:** Between any two statements, it gives the logical AND condition. AND can be combined with the GIVEN, WHEN, and THEN statements.
  Example: When I enter my account number AND CVV.
- **But:** It denotes a logical OR relationship between two propositions. OR can be combined with the GIVEN, WHEN, and THEN statements.
  Example: Then I should be logged in BUT I must enter the OTP.

13. Enlist the files needed in the Cucumber framework.

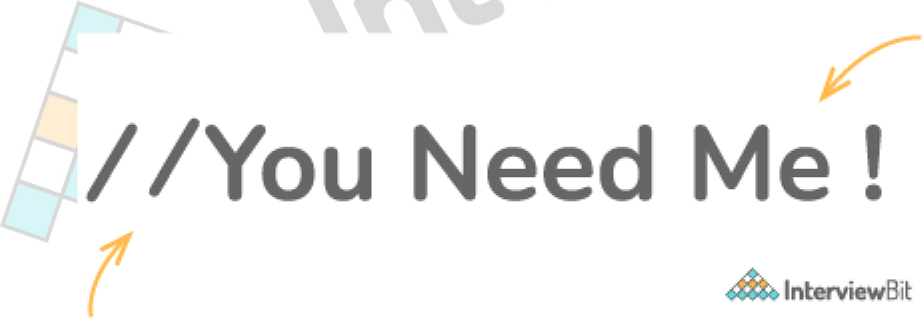
The following are the files required for a Cucumber framework:

- **Feature File**: It has plain text descriptions of single or numerous test situations. Keywords like Then, When, Background, Scenario Outline, Feature, And, But, and so on are used in the tests. As a result, it's a file that keeps track of features and their descriptions.
- **Step Definition File**: It has the extension .java. It essentially acts as a translator between the test scenario steps provided in the feature file and the automation code. Cucumber searches the step definition file and executes the relevant functions that are assigned to that step when it runs a step described in the feature file.
- **TestRunner**: .java is the file extension for this file. It connects the feature file and the step definition file. It allows the user to run one or more feature files at the same time. It contains the locations of the step definition and feature files.

14. How do you comment the code in Cucumber? What is the importance of comments?

A comment is a chunk of code that is intended for documentation rather than execution. To make it more legible and clear, whether it's a step definition file or a feature file. As a result, it's critical to use/insert comments in the right places throughout the file. This is also beneficial for troubleshooting the code. Comments can be added to Cucumber feature files at any time. To add comments, simply begin the statement with the “#” sign.
Different programming languages have different standards for commenting. Let's see how Cucumber handles the situation:

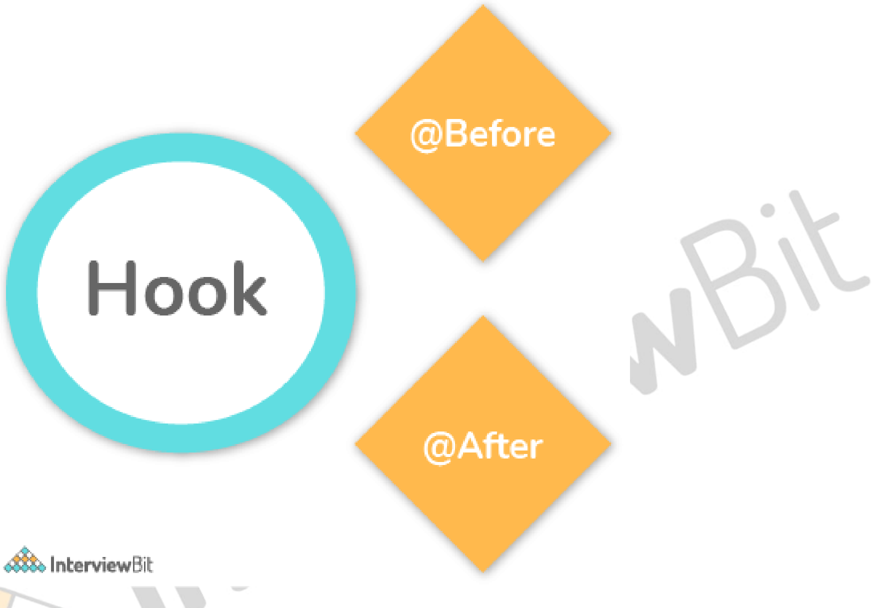
- For Step Definition File, if you're using Java as a platform, start your comments with "/."
- In the case of a feature file, we only need to type # before starting our comment.



15. What are hooks in Cucumber?

Hooks are code blocks that execute before or after each Cucumber scenario in the execution cycle. This enables us to better control the development workflow and decrease code redundancy. Setting up the web driver and terminating the web driver session resembles a test setup. When dealing with different scenarios, it's best to do the setup and clean up only once. Hooks are used to bringing optimization.

Certain preconditions, such as executing the program, creating a database connection, preparing the test data, and so on, may be required in some cases. There are also several postconditions to be fulfilled, such as ending the database connection, closing the browser, refreshing test data, and logging out of the program. Cucumber handles all of these situations with the use of hooks.



The methods @Before and @After can be used to define hooks anywhere in the project or step definition layers. Before hook is executed before any other test situations, and after the hook is executed after all test scenarios have been completed.

16. What are tags in Cucumber and why are they important?

When we only have one, two, or maybe five situations in a feature file, it appears to be simple. In reality, however, this does not occur. In a single feature file, we may have 10, 20, or even more scenarios for each feature under test. They could reflect various purposes (smoke test/regression test), perspectives (developer/QA/BA), and statuses (ready for execution/work in progress).
Tags in cucumber provide a way to run scenarios in a specific sequence from a runner file. Each situation can be labeled with a useful tag. Later, in the runner file, we may specify which tag (and hence which scenario(s)) Cucumber should run. “@” is the first character in a tag. Any relevant content after "@" can be used to define your tag. Example - ‘@InitialTest’

17. What is Cucumber Dry Run?

The purpose of the Cucumber dry run is to verify compilation faults and compile the Step Definition and Feature files. Dry run's value might be either true or false. Dry run has the value false by default and it is present in the Test Runner Class file.

If the dry run value is true, Cucumber will check all steps in the Feature file. Within the Step Definition file, it will also check the implementation code of steps in the Feature file.

If any of the steps in the Feature file is missing its implementation in the Step Definition file, a message is thrown. The @CucumberOptions has a dry run parameter that is used to configure the test parameters.

18. What do you mean by profile in Cucumber?

When testing a feature, cucumber profiles make it simple to define groupings of tests in a feature file so that we can choose to execute only a subset of them rather than all of them. It was created to help people save time. In a cucumber.yml file, the user can reuse commonly used cucumber flags.

19. What programming languages are used by Cucumber?

Cucumber supports a variety of programming languages, including Java,.NET, Ruby, and others. It can also be used with other tools like Capybara and Selenium.

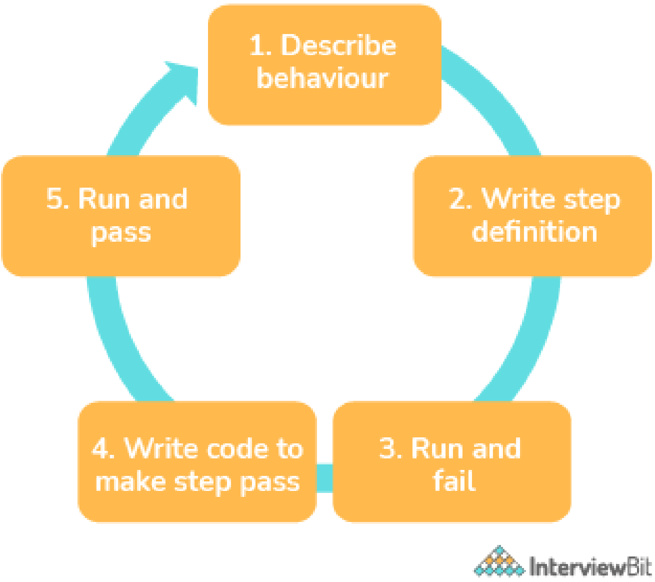
The Gherkin text serves as a skeleton for your automated tests and serves as documentation. Gherkin is based on TreeTop Grammar, which is used in more than 37 languages. As a result, you can write your gherkin in more than 37 different spoken languages.

Cucumber Interview Questions for Experienced

20. Explain briefly how Behavioral Driven Development works?

There are majorly three steps in the working of BDD. They are as follows:-

- **Behaviour Description**: We list down the features of our application first in the feature file.
- **Making the Step Definition file**: The mapping between each step of the scenario defined in the feature file and a code of the function to be executed is stored in the steps definition file.
- **Testing and running**: We run the test cases to check if we pass. In general, a lot of failures are observed before achieving the final code.



21. What is a test harness in the context of Cucumber?

The test harness in Cucumber helps in separating the task of establishing the context and interacting with the browser from cleaning up the step definition files. It gathers the stubs, drivers, and other tools needed to enable test execution automation in testing.

The following is the purpose of the test harness:-

- To run a set of tests either within the framework or with the help of the test harness
- To enter data into the program being tested.
- Debugging becomes more flexible and supported.
- To record the outputs of the software under test
- To keep track of the test results (pass/fail) for each test.
- Aids developers in determining code coverage at the code level.

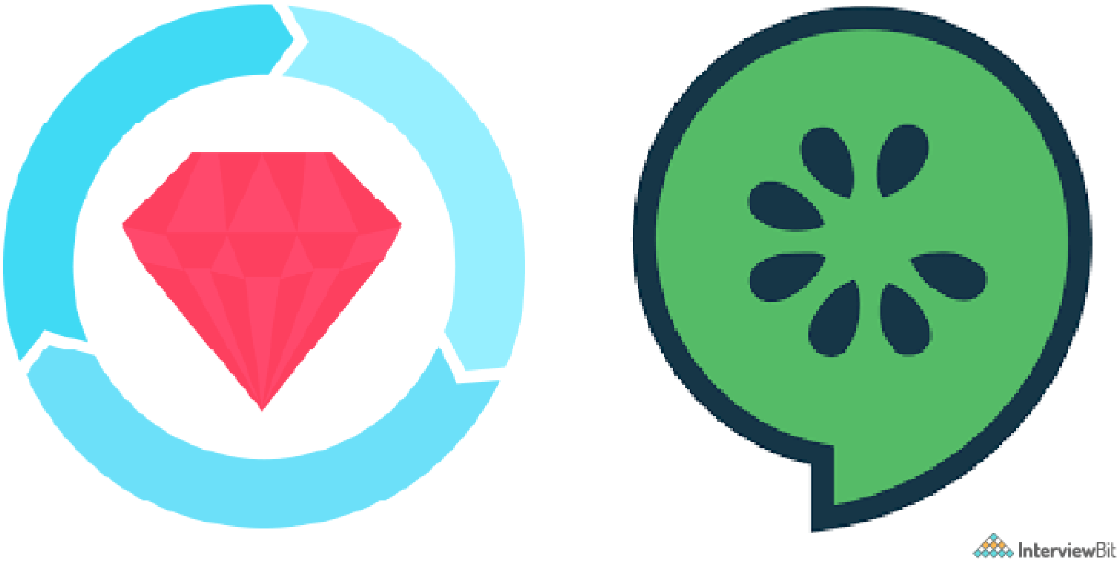
The advantages of the test harness are as follows:-

- As a result of automation, productivity increases.
- Improved software quality as a result of automation allows us to be more productive.
- Tests can be scheduled.
- Can handle complex conditions that testers have a hard time simulating.

22. What is the difference between RSpec and Cucumber?

RSpec and Cucumber are two examples of testing frameworks. Traditional Unit Testing is used by RSpec. It refers to the practice of testing a section of an application separately from the remaining part of the application. As a result, your model performs what it's expected to do, the controller does what it's expected to do, and so on. Both RSpec and Cucumber are used for Acceptance Testing, also known as ATDD, BDD, and other terms.
The following are the major differences between RSpec and Cucumber:-

- The fundamental distinction between RSpec and Cucumber is the element of business readability.
- Unit testing is the primary purpose of RSpec. Cucumber, on the other hand, is primarily utilized in behavior-driven development. It can be used for System and Integration Testing as well.
- Cucumber separates the specs or features from the test code, allowing product owners to provide or review the specification without having to walk through the code.
- RSpec includes a similar method, but instead of elaborating a step with a Describe, it executes the statement using the business specification. This method is easier for developers to use, but a little more difficult for non-technical people.



23. Difference between Selenium and Cucumber.

Open-source testing tools, Selenium and Cucumber are both used for functional testing. However, there are some distinctions between them.

Here are some key distinctions between Selenium and Cucumber:

- Cucumber is a behavior-driven development automation tool that may be used with Selenium. Selenium is a web browser automation tool for web projects (or Appium).
- Cucumber is used for acceptance testing, while Selenium is used for automated UI testing.
- Technical teams (SDETs/programmers) favour Selenium, while non-technical teams often choose Cucumber (business stakeholders and testers).
- Cucumber isn't required for Selenium to work. Cucumber's step-definition implementation is based on Selenium or Appium.
- The script creation with Selenium is complicated, whereas Cucumber is straightforward.

24. Why do we need to use Cucumber with Selenium?

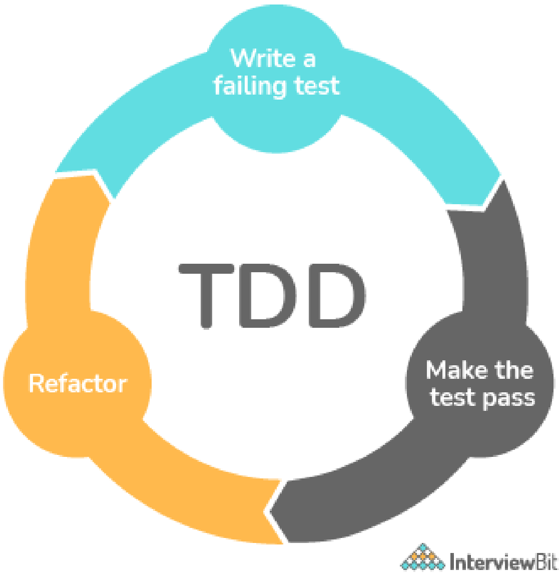
Cucumber and Selenium are two widely used testing frameworks and technologies. Selenium is widely used for functional testing in many organizations. These companies use Cucumber in conjunction with Selenium because Cucumber makes the application flow easier to read and comprehend. The most important advantage of combining Cucumber and Selenium is that it allows developers to build test cases in simple feature files that managers, non-technical stakeholders, and business analysts can understand. It allows you to develop tests in Gherkin, a human-readable programming language. Java,.NET, PHP, Python, Perl, and other programming languages are supported by the Selenium-Cucumber framework.

25. In a feature file, what is the maximum number of scenarios?

A feature file in Cucumber can include a maximum of 10 scenarios. This quantity can differ from one project to the next and from one organization to the next. It's advisable to keep the number of scenarios in the feature file to a minimum.

26. What do you mean by Test Driven Development (TDD)?

TDD is an abbreviation that stands for Test-Driven Development. This is a development practice in which the test cases are created first, followed by the code that underpins the test cases. TDD may also be used to construct automation testing. TDD takes longer to develop due to the fact that it finds fewer flaws. The TDD development practice has increased the quality of code, which is more reusable and flexible as a result. TDD also aids developers in achieving high test coverage, ranging from 90% to 100%. The sole disadvantage of TDD for developers is that they must build test cases before producing code.



The following is a list of the TDD methodology's basic six-step process:

- First, all the test cases are written. Based on your requirements, you must create an automated test case.
- Carry out all of the tests: Carry out these automated test cases on the code that has been developed so far.
- Modify the code for that test case: You must develop the code to make that test casework as intended if it fails throughout this step.
- Rerun the test cases: Now you must rerun the test cases to ensure that all of the previously developed test cases have been implemented.
- Modularize your code as follows: This is a step that can be skipped. However, refactoring your code to make it more readable and reusable is recommended. That is why it is necessary.
- For new test scenarios, repeat steps 1–5: This is the final phase in the process. You must now repeat the process for the remaining test cases till all of them have been implemented.

27. Difference between TDD and BDD.

TDD	BDD
Test-Driven Development (TDD) is a method of developing software that is driven by tests. This means that the developers must first write the test cases before writing the code.	BDD is an acronym for behavior-driven development. It's a behavior-based development approach.
TDD tests are developed in a variety of programming languages, including Java,.NET, Python, Ruby, and others.	Given-When-Then steps are used to write BDD tests in a human-readable fashion. Non-technical people may read and comprehend these tests as well.
The scope is the key distinction between TDD and BDD. TDD is a development methodology.	BDD, on the other hand, is a collaborative methodology.
When a test fails because the specified function does not exist, TDD recommends writing the simplest code possible to pass the test, then reworking to remove duplication, and so on.	Creating an executable specification that fails because the feature isn't available, then writing the simplest code possible to make the spec pass in BDD. This process is repeated until a release candidate is ready to be delivered.
The test cases are written by the developers in TDD.	Users or testers write automated specifications in BDD, which are then wired to the code under test by developers.

28. What is the use of the Options tag in the Cucumber Framework?

The Options tag is a part of the TestRunner file in the Cucumber framework, and it takes the form of an annotation named @CucumberOptions.

It has two parameters: glue and feature:

1. **Feature:** The path to the feature file is specified by the feature option.
2. **Glue:** The glue argument is used to provide the step definition file's location.

Example:

```
import org.junit.runner.RunWith;
import cucumber.api.CucumberOptions;
import cucumber.api.junit.Cucumber;
@RunWith (Cucumber.class)
@CucumberOptions (
features = "src/test/Sample/features ",
glue = {"StepDefinitionFile"}
)
public class SampleTestRunner {
}
```

29. How does the execution start in Cucumber?

Cucumber execution will begin at the support level. In support, it will first load the env.rb file, then hooks.rb, and last start executing feature file scenario steps.

30. What is grouping in the context of Cucumber?

Cucumber is unconcerned about the names of your step definition files or the order in which you place them. Instead of maintaining all steps in a single file, we can create steps.rb file for each major action/feature. This is referred to as grouping.

31. How can you run Cucumber tests parallelly?

The Cucumber JVM Parallel Plugin, which may be used with Serenity BDD, can be used to conduct parallel tests in Cucumber. The plugin will look in the src/test/resources directory for feature files. After that, it will create runners for each file.

32. What are some of the prerequisites that you should consider while building a Selenium Cucumber automation application?

We consider the following before building a Selenium Cucumber automation application:-

- Determine the type of application you'll be testing. Is it a Web app, a mobile app, or a desktop application?
- Is there a need for backend testing? Databases or SDKs, for example.
- Is it necessary to run the app through an internationalization test?
- It must include a report that allows you to track down a problem with minimal effort.
- It must be able to generate parametrization tests automatically.
- Any setup-related settings or global attributes should be defined in a config file.
- To segregate the functionality, use abstraction at every level.

33. Difference between JBehave and Cucumber.

Despite the fact that Cucumber and JBehave have the same goal in mind, acceptance tests are two quite distinct frameworks:

- Cucumber is built on Ruby, while JBehave is a pure Java Framework.
- Cucumber is built on features, whereas JBehave is based on stories.

34. How can you run a selected test from a group of tests in Cucumber?

We may execute a single test from a set of tests in the Cucumber framework using the tags idea. This is found in the TestRunner file's @CucumberOptions section. With the use of the @t<agname> keyword, we may tag a scenario in the feature file. A scenario can have one or more tags within the feature file. We can separate test scenarios with the assistance of tagging. We must pass the <tagname> value within the tags argument to execute a selected test in Cucumber, and we must pass the <~tagname> value within the tags parameter to exclude a test from running.

Conclusion:

Cucumber is frequently used in BDD because it is an open-source technology. It's also quite simple to grasp, has a lot of room for additional features, and it's rather easy to connect Cucumber with Selenium or other third-party tools/jars, etc.
It is quite easy for anyone who has just started learning Cucumber or for those who have intermediate experience in Cucumber/BDD because it has active help groups/members.
Cucumber also allows for connectivity with an excel sheet and Jenkins.

References:
Cucumber Documentation

Recommended Resources:

Selenium Interview
Automation Testing

Links to More Interview Questions

- C Interview Questions
- Web Api Interview Questions
- Cpp Interview Questions
- Machine Learning Interview Questions
- Css Interview Questions
- Django Interview Questions
- Operating System Interview Questions
- Git Interview Questions
- Dbms Interview Questions
- Pl Sql Interview Questions
- Ansible Interview Questions
- Php Interview Questions
- Hibernate Interview Questions
- Oops Interview Questions
- Docker Interview Questions
- Laravel Interview Questions
- Dot Net Interview Questions
- React Native Interview Questions
- Java 8 Interview Questions
- Spring Boot Interview Questions
- Tableau Interview Questions
- Java Interview Questions
- C Sharp Interview Questions
- Node Js Interview Questions
- Devops Interview Questions
- Mysql Interview Questions
- Asp Net Interview Questions
- Kubernetes Interview Questions
- Aws Interview Questions
- Mongodb Interview Questions
- Power Bi Interview Questions
- Linux Interview Questions
- Jenkins Interview Questions