Cucumber - Gherkin basics

Gherkin is a **plain-text language** used to write **test cases** in a **human-readable format**. It's most commonly used in **Behavior-Driven Development (BDD)** tools like **Cucumber**.

🔰 What is Gherkin?

- Gherkin is a **business-readable domain-specific language** (DSL).
- It lets you describe **software behaviors** without needing to understand the code.
- Gherkin is written in simple English using keywords like **`Feature`**, **`Scenario`**, **`Given`**, **`When`**, **`Then`**, etc.

🧠 Why use Gherkin?

- It bridges the gap between **developers**, **testers**, and **non-technical stakeholders**.
- Everyone can understand the requirements or test cases.
- Encourages collaboration between business and technical teams.

🧱 Gherkin Syntax and Keywords

Keyword	Purpose
`Feature:`	Describes the feature under test
`Scenario:`	Defines a particular test case
`Given`	Describes the initial context (pre-condition)
`When`	Describes the action performed by the user
`Then`	Describes the expected outcome
`And`	Used to combine multiple **`Given`**, **`When`**, or **`Then`** steps
`Background`	Defines steps common to all scenarios in the feature
`Scenario Outline`	Used for data-driven (parameterized) testing
`Examples:`	Used with **`Scenario Outline`** to provide different sets of inputs

✅ Simple Gherkin Example

```
gherkin

Feature: Login functionality

  Scenario: Successful login with valid credentials
    Given User is on the login page
    When User enters valid username and password
    And User clicks the login button
    Then User should be redirected to the homepage
```

✅ Explanation:

- **`Feature`**: Describes what the feature is about.
- **`Scenario`**: This is one test case.
- **`Given`**: User starts on the login page.
- **`When`**: User takes an action (inputs credentials).
- **`Then`**: Expected result (user lands on homepage).

🔄 Scenario Outline Example (Data-driven test)

```
gherkin

Feature: Search functionality

  Scenario Outline: Search for items
    Given User is on the search page
    When User searches for "<item>"
    Then Search results for "<item>" should be displayed

    Examples:
      | item     |
      | laptop   |
      | mobile   |
      | headset  |
```

✅ Explanation:

- **`Scenario Outline`**: Same scenario tested with different data.
- **`Examples:`**: Table provides the values for **`<item>`**.

🛠️ How is Gherkin used in practice?

Gherkin files are saved with a **`.feature`** extension. For example:

```
plaintext

login.feature
search.feature
```

These are then linked with **step definitions** written in a programming language like Java, Python, or JavaScript using tools like **Cucumber**.

🧪 Step Definitions Example (Java + Cucumber)

```
java

@Given("User is on the login page")
public void user_on_login_page() {
    // Code to navigate to login page
}
```

```
@When("User enters valid username and password")
public void enter_valid_credentials() {
    // Code to enter credentials
}

@Then("User should be redirected to the homepage")
public void verify_homepage() {
    // Code to verify user is on homepage
}
```

📦 Summary

- **Gherkin** is used to write **test scenarios** in plain English.
- Helps teams **collaborate** easily.
- Works with **BDD tools** like **Cucumber**.
- Tests are connected to code via **step definitions**.

Would you like a mini project using Gherkin with Java + Cucumber step definitions?

Absolutely! Let’s walk through **writing feature files in Gherkin from scratch**, step by step, in a way that is beginner-friendly.

✅ Step 1: What is a `.feature` File?

A `.feature` file is a **plain text file** written using **Gherkin syntax** that describes your application’s behavior in a structured way.
It should have the `.feature` extension, like `login.feature`, `search.feature`, etc.

✅ Step 2: Structure of a Gherkin Feature File

Here’s the **basic structure** of a `.feature` file:

```
gherkin

Feature: Title of your feature
  Description of the feature (optional)

  Scenario: Title of your scenario
    Given some precondition
    When some action is taken
    Then expect some outcome
```

✅ Step 3: Create Your First Feature File

Let’s write a basic example: **Login feature**.

📄 File name: `login.feature`

```
gherkin

Feature: Login functionality

  As a registered user,
  I want to login to the website
  So that I can access my dashboard

  Scenario: Successful login with valid credentials
    Given User is on the login page
    When User enters valid username and password
    And User clicks the login button
    Then User should be redirected to the dashboard
```

🔍 Explanation:

- `Feature`: Defines what feature you're testing.
- Description: Optional but useful for context.
- `Scenario`: A specific use case or behavior.
- `Given`: Precondition (where the user starts).
- `When`: Action performed.

- **`Then`**: Expected outcome.
- **`And`**: Adds additional steps of the same type.

✅ Step 4: Add Another Scenario

```
gherkin

Scenario: Login fails with invalid password
  Given User is on the login page
  When User enters valid username and invalid password
  And User clicks the login button
  Then An error message should be displayed
```

✅ Step 5: Use Scenario Outline for Data-Driven Testing

If you want to test the same scenario with multiple inputs, use **`Scenario Outline`** and **`Examples`**.

```
gherkin

Scenario Outline: Login attempts with multiple credentials
  Given User is on the login page
  When User enters "<username>" and "<password>"
  And User clicks the login button
  Then "<message>" should be displayed

  Examples:
    | username  | password     | message                 |
    | user1     | correct123   | Dashboard page appears   |
    | user2     | wrongpass    | Invalid credentials error|
```

✅ Step 6: Naming Best Practices

- Use **meaningful names** for your **`Feature`** and **`Scenario`**.
- Stick to the **Given-When-Then** structure.
- Use **simple, clear language** anyone can understand.
- Keep scenarios **focused and short**.

🗂️ Where to Save Feature Files?

- Inside your project: **`src/test/resources/features`**
- File name: **`login.feature`**, **`search.feature`**, etc.

🧠 Summary

Concept	Purpose
`.feature` file	Contains Gherkin syntax for test cases

Concept	Purpose
`Feature:`	Describes the feature under test
`Scenario:`	Describes a specific test case
`Given`, `When`, `Then`, `And`	Describe steps of the scenario
`Scenario Outline`	Used for testing with multiple inputs
`Examples:`	Table of data for the outline

Would you like a complete beginner mini-project folder structure for writing and executing feature files using Cucumber + Java?

Here's a complete and beginner-friendly list of **Gherkin keywords**, along with their **purpose and examples**. These keywords form the core of writing `.feature` files in **Gherkin** syntax.

🏷️ Gherkin Keywords List

Keyword	Purpose
`Feature`	Describes the feature you are testing
`Scenario`	Describes a specific test case or user story
`Scenario Outline`	Used for data-driven testing with multiple sets of values
`Examples`	Defines the data table for a `Scenario Outline`
`Given`	Sets up the initial context or preconditions
`When`	Describes an action or event
`Then`	Describes the expected outcome/result
`And`	Adds additional steps of the same type (`Given`, `When`, or `Then`)
`But`	Adds a negative step to a condition (`Given`, `When`, or `Then`)
`Background`	Defines common steps for all scenarios in a feature
`*` (asterisk)	Alias for `Given`, `When`, `Then`, `And`, or `But`
`@tags`	Used to tag scenarios for filtering or grouping
`#` (comment)	Used for adding comments

🔍 Detailed Explanation with Examples

✅ `Feature`

Defines the feature under test.

```
gherkin

Feature: User Login
```

✅ `Scenario`

Defines a single example or test case.

```
gherkin

Scenario: Successful login with valid credentials
```

✅ `Given`

Describes the **initial state or context** (precondition).

```gherkin
Given User is on the login page
```

✅ When

Describes the **action or event** performed by the user.

```gherkin
When User enters valid username and password
```

✅ Then

Describes the **expected outcome or result**.

```gherkin
Then User should be redirected to the homepage
```

✅ And

Used to **combine multiple steps** of the same type.

```gherkin
And User clicks the login button
```

✅ But

Used to **add exceptions or negatives**.

```gherkin
But User should not see the admin dashboard
```

✅ Background

Defines **common preconditions** for all scenarios.

```gherkin
Background:
  Given User is on the login page
```

```gherkin
Scenario: Login with correct credentials
  When User enters valid credentials
  Then Dashboard is displayed

Scenario: Login with wrong credentials
  When User enters invalid credentials
  Then Error message is shown
```

✅ `Scenario Outline` and `Examples`

Used for **data-driven testing** (one scenario with multiple data inputs).

```gherkin
Scenario Outline: Login attempts
  Given User is on the login page
  When User enters "<username>" and "<password>"
  Then "<message>" should be displayed

  Examples:
    | username | password   | message                 |
    | user1    | pass123    | Welcome user1           |
    | user2    | wrongpass  | Invalid credentials     |
```

✅ `@Tags`

Used to mark or group scenarios.

```gherkin
@smoke @login
Scenario: Login with valid credentials
  Given ...
```

You can run only **`@smoke`** tests with Cucumber CLI or test runners.

### ✅ `#` (Comments)

```gherkin
# This is a comment and will be ignored by the parser
Feature: Search functionality
```

🎯 Summary Table

Keyword	Usage Description
`Feature`	What feature is under test
`Scenario`	Individual test case
`Scenario Outline`	Parametrized test case

Keyword	Usage Description
`Examples`	Table of test data for outline
`Background`	Common setup for all scenarios
`Given`	Precondition
`When`	User action
`Then`	Expected result
`And`, **`But`**	Additional steps or conditions
`@tag`	Group/mark scenarios
`#`	Comment line

Would you like a **cheat sheet PDF** for these Gherkin keywords or a **practice assignment** using them?