Let's Eat Gherkin

Handbook for Gherkin Beginners

Written by Giridhar Rajkumar

About this handbook

Are you new to Gherkin Writing?

Then this is the right handbook for you. This handbook contains all basic information you need to start with Gherkin Writing. It has tips that you might find very helpful.

By reading this handbook, you will be able to nurture the way of writing the Gherkin Scenarios for your projects.

About the author

***Giridhar Rajkumar** is a Test Automation Consultant. He is having more than 9 years of experience in Information Technology Industry and has more than 5 years of experience with Behavior Driven Development concepts. He has engaged with multiple projects in Insurance, Banking and Media domains to implement the BDD process.*

அகர முதல எழுத்தெல்லாம் ஆதி

பகவன் முதற்றே உலகு

திருவள்ளுவர்

A, as its first of letters, every speech maintains;

The "Primal Deity" is first through all the world's domains.

Thiruvalluvar

Dedicated to

my lovable wife

Vaishnavi Chakrapani

Contents

Introduction to BDD

BDD is abbreviated as Behaviour Driven Development. BDD is a software development process which is more test-centric like Test Driven Development (TDD). In general, BDD focuses on the behaviour of the application.

Behaviours are multiple chunks of the requirements written as Scenarios. These scenarios are written with several examples for the benefit of business analysts, developers and testers. The scenarios will not be in general description like a used case, rather it specifically talks the behaviour of the application when a particular action / input is taken in the UAT. To achieve these kind of scenarios BDD uses a language call Gherkin language.

What is Gherkin language?

Gherkin is the most famous languages which is used to write the BDD scenarios. It is written in the format of GWTs (Given-When-Then). When written in this format, scenarios look more specific to the behaviour of the application.

One thing we need to keep in mind is Gherkin scenarios should be written from the end user's point of view

Now, let's see the annotations we use in Gherkin –

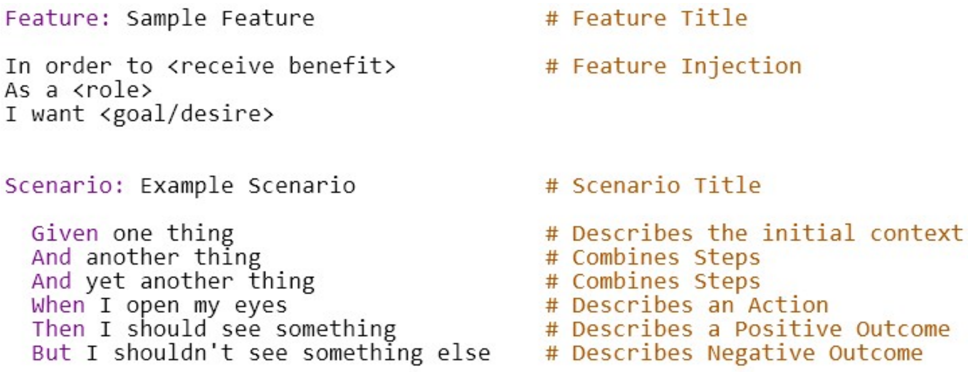
Given	Steps are used to describe the initial context of the system
When	Steps are used to describe an event, or an action
Then	Steps are used to describe an expected outcome, or result
And	Steps are used to combine more than one event or outcome
But	Steps are used to write the negative outcome (rarely used)

	Gherkin *scenarios should* ***NOT*** *be written from developers or testers point of view. It should be written only from end user's point of view* ***Bad example:*** *Given I am a tester of the application (or) Given I develop the application*

Gherkin examples

Let’s learn the culture of writing Gherkin with some examples. In this example we will use the functionality we all are familiar with, the Login page. In BDD, the scenarios are drafted in a file called “Feature” file. Basically, the feature files will contain all the scenarios used to develop a feature.

Before we write the scenarios in the feature file, let’s see the template we need to use to write it.



```
Feature: Sample Feature                 # Feature Title

In order to <receive benefit>           # Feature Injection
As a <role>
I want <goal/desire>

Scenario: Example Scenario              # Scenario Title

  Given one thing                       # Describes the initial context
  And another thing                     # Combines Steps
  And yet another thing                 # Combines Steps
  When I open my eyes                   # Describes an Action
  Then I should see something           # Describes a Positive Outcome
  But I shouldn't see something else    # Describes Negative Outcome
```

Golden Rules

1. *In a given **Scenario**, there should be only one **Given**, **When** and **Then** steps*

2. *Maximum usage of two **And** steps are allowed*

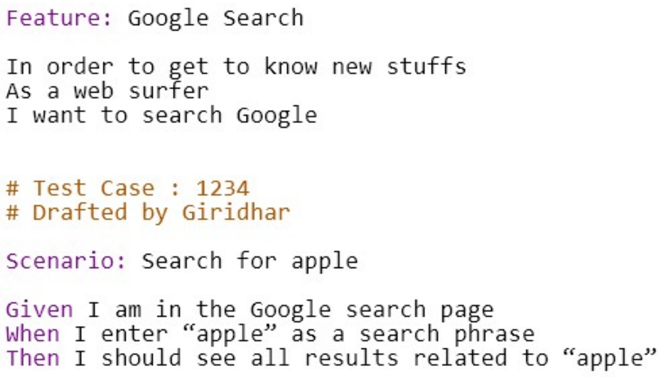
3. *Make sure your Scenario has maximum of 5 steps and not more than that. In case, if it is exceeding, try to split it into two different scenarios*

Advanced Gherkin Writing

Comments

Comments allow the author of the feature file to add additional information for the benefit of developers and testers. Comments will start with a hash tag "#".

For Example:



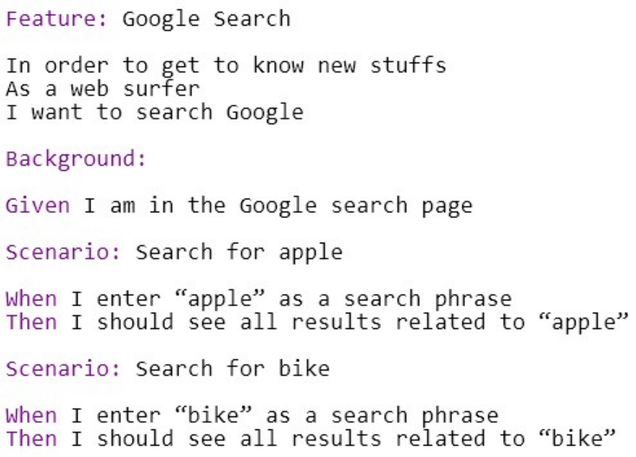
```
Feature: Google Search

In order to get to know new stuffs
As a web surfer
I want to search Google


# Test Case : 1234
# Drafted by Giridhar

Scenario: Search for apple

Given I am in the Google search page
When I enter "apple" as a search phrase
Then I should see all results related to "apple"
```

Background

There are some instances where feature file might contain or share common steps. Instead of duplicating those steps we can put it under Background section.

In this case, the steps given in the Background will be executing for every scenario.

For Example:



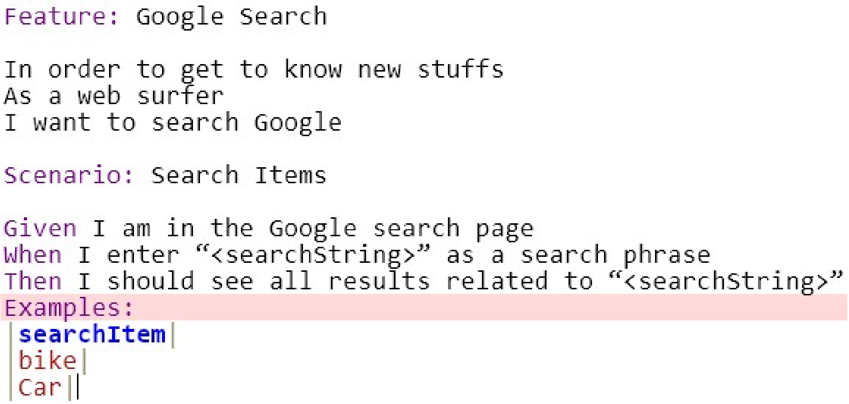
```
Feature: Google Search

In order to get to know new stuffs
As a web surfer
I want to search Google

Background:

Given I am in the Google search page

Scenario: Search for apple

When I enter “apple” as a search phrase
Then I should see all results related to “apple”

Scenario: Search for bike

When I enter “bike” as a search phrase
Then I should see all results related to “bike”
```

Scenario-Outlines

The Scenario-Outline is highly reusable in Gherkin. Let’s see how it is more user friendly and reusable.

Consider the above example in Background section. Here, we see that both the scenarios are doing the exact same action perhaps with different parameters. Scenario-Outlines are generally parameterised using Examples. Rows in the table represents the value it has to use as a parameter. The values are replaced with angular brackets “< >” in the scenario.

They can be combined together as follows -

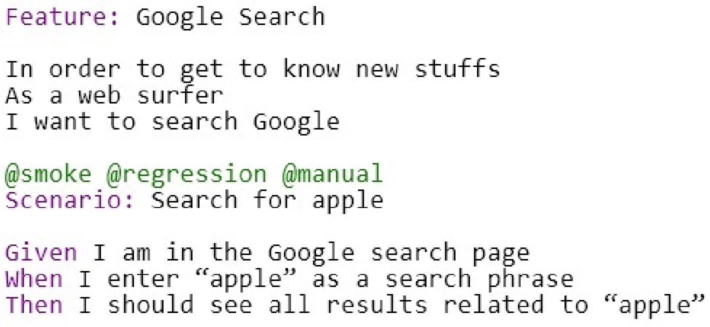


```
Feature: Google Search

In order to get to know new stuffs
As a web surfer
I want to search Google

Scenario: Search Items

Given I am in the Google search page
When I enter “<searchString>” as a search phrase
Then I should see all results related to “<searchString>”
Examples:
|searchItem|
|bike|
|Car||
```

Tags

Tags are the best way to categorize the scenarios. Tags can be used to run specific set of tests when they are tagged appropriately. They are 100% customisable, meaning you can any word as tags. Tags will always start with @ character before the scenario as given below.

For Example:



```
Feature: Google Search

In order to get to know new stuffs
As a web surfer
I want to search Google

@smoke @regression @manual
Scenario: Search for apple

Given I am in the Google search page
When I enter “apple” as a search phrase
Then I should see all results related to “apple”
```

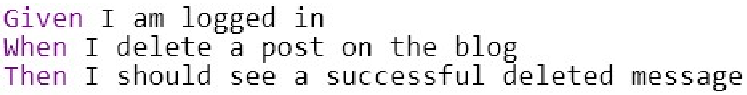
Tags can be used to categorise the environments any types of tests as well like *@test, @uat, @regression, @smoke*

Three common mistakes while writing Gherkin Scenarios

Writing scripts in first person instead of third person

When starting with Gherkin, people quickly feel like they should describe the actions they are performing when going through a functionality.

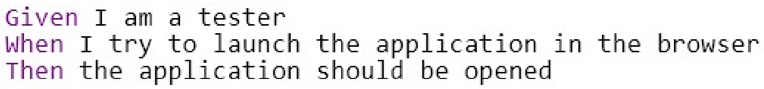
✗ ***Wrong***



```
Given I am logged in
When I delete a post on the blog
Then I should see a successful deleted message
```

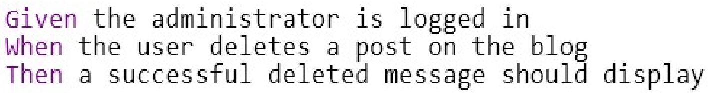
The problem with the case above is that it's not clear who this "I" person is. This is quite important information when reading Gherkin, so you can understand it quicker. This post goes more in-depth about why third person is better.

✗ ***Wrong***



```
Given I am a tester
When I try to launch the application in the browser
Then the application should be opened
```

Right

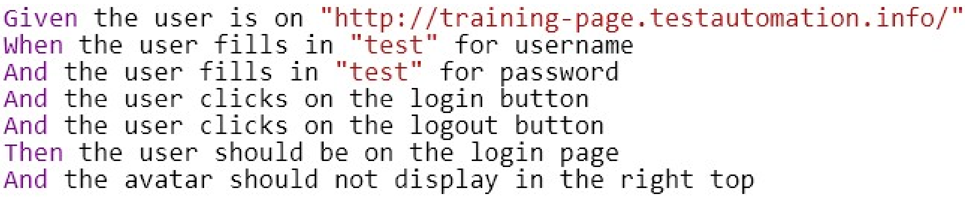


```
Given the administrator is logged in
When the user deletes a post on the blog
Then a successful deleted message should display
```

Describing every action instead of a functionality

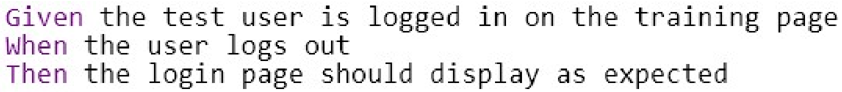
When writing Gherkin, it’s often seen as if you have to describe every action to get to the next step: Every click, every text input, web-page. But Gherkin is meant to describe a specific flow you’re going through.

❌ ***Wrong***



```
Given the user is on "http://training-page.testautomation.info/"
When the user fills in "test" for username
And the user fills in "test" for password
And the user clicks on the login button
And the user clicks on the logout button
Then the user should be on the login page
And the avatar should not display in the right top
```

Right



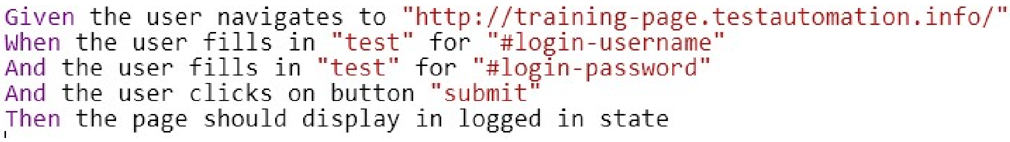
```
Given the test user is logged in on the training page
When the user logs out
Then the login page should display as expected
```

Using absolute values instead of configurable values

When writing Gherkin scripts, it’s always quick and easy to simply use the absolute values that you want to use. For example, a username and password, or the values used in a registration form.

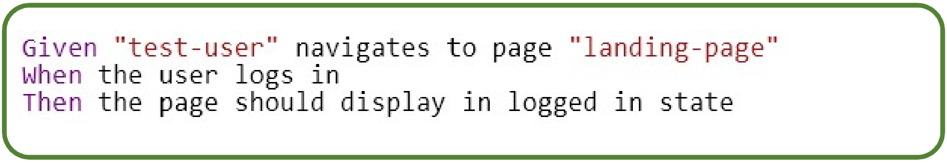
It’s best practise to move your absolute values to a configuration file, so you can set user information in a central place.

✗ ***Wrong***



```
Given the user navigates to "http://training-page.testautomation.info/"
When the user fills in "test" for "#login-username"
And the user fills in "test" for "#login-password"
And the user clicks on button "submit"
Then the page should display in logged in state
```

Right



```
Given "test-user" navigates to page "landing-page"
When the user logs in
Then the page should display in logged in state
```

So, in this case the URL is moved into a configuration value called “landing-page” and the user credentials in “test-user”. If at any point in the future one of these values change, you’ll only have to change one configuration value with it.

Tools for Gherkin

There are couple of editors available for writing the Gherkins. Eclipse has plugins like Cucumber and Visual Studio has SpecFlow package that can be installed from NuGet Gallery.

There are other open source editors that are readily available for non-technical people like Business Analysts, Product Owners or any other Business-related people. One such tool is **Tidy Gherkin**, which is readily available as Google Chrome Application. Other tools like Atom and Notepad++ offers Gherkin packages.

Conclusion

With these basics, you can definitely write effective Gherkin Features. In the next handbook we will deep dive more into the best practices of BDD from process side…

References

1. *https://www.spritecloud.com/2018/03/the-3-most-common-mistakes-writing-gherkin-features/*

2. *https://docs.cucumber.io/gherkin/reference/*