Hands-On Lab: Writing Feature Files

Estimated time needed: 20 minutes

Welcome to the **Writing Feature Files** lab. The first step in BDD testing is writing the feature files that explain the behavior that drives the tests. In this lab, you will write the feature file that you will use in subsequent labs to drive the testing.

Learning Objectives

After completing this lab, you will be able to:

- Write the initial feature statement
- Define the initial state before running tests
- Describe a scenario for testing to ensure that the feature behaves as expected

About Theia

Theia is an open-source IDE (Integrated Development Environment) that can be run on a desktop or the cloud. You will be using the Theia IDE to complete this lab. When you log into the Theia environment, you are presented with a 'dedicated computer on the cloud' exclusively for you. This is available to you as long as you work on the labs. Once you log off, this 'dedicated computer on the cloud' is deleted, along with any files you may have created. So, it is a good idea to finish your labs in a single session. If you finish part of the lab and return to the Theia lab later, you may have to start from the beginning. Plan to work out all your Theia labs when you have the time to finish the complete lab in a single session.

Set Up the Lab Environment

You have a little preparation to do before you can start the lab.

Open a Terminal

Open a terminal window by using the menu in the editor: **Terminal > New Terminal**.

In the terminal, if you are not already in the `/home/projects` folder, change to your project folder now.

```
cd /home/project
```

Clone the Code Repo

Now, get the code that you need to test. To do this, use the `git clone` command to clone the git repository:

```
git clone https://github.com/ibm-developer-skills-network/duwjx-tdd_bdd_PracticeCode.git
```

Change into the Lab Folder

Once you have cloned the repository, change to the lab directory:

```
cd duwjx-tdd_bdd_PracticeCode/labs/09_writing_feature_files
```

You are now ready to start the lab.

Optional

If working on the terminal becomes difficult because of the length of the command prompt, you can shorten the prompt using the following command:

```
export PS1="[\[\033[01;32m\]\u\[\033[00m\]: \[\033[01;34m\]\W\[\033[00m\]]\$ "
```

Navigate to the Code

In the IDE on the right of your screen, navigate to the `duwjx-tdd_bdd_PracticeCode/labs/09_writing_feature_files` folder. This folder contains all of the source code that you will use for this lab.

You will work with the `petshop.feature` file throughout this lab.

In the following steps, you will create a feature file for “Search for pets by category” that was presented in the lecture video. This will be used in later labs for running tests.

Describe the Feature

You will start by describing the feature that you are testing. Here is the scenario:

The feature you are working on is to search for pets by category. You are a customer of the pet shop. You already know the category of pet you are looking for, so when you go to the home page of the pet shop you want, you can search for a pet by category. The value of doing this is you should be able to see the category of the pet that you are interested in buying.

Your Task

Start by opening the file `petshop.feature`.

Open **petshop.feature** in IDE

Start with the keyword `Feature:` and the title “Search for pets by category”. Then use the **As a**, **I need**, **So that** syntax to describe who the feature is for, what the feature is about, and the value that it brings based on the description above.

▼ Click here for a hint.
Here is an outline of the feature story.

```
Feature: {title here}
As a {who is it for?}
I need {what is the function?}
So that {what is the value?}
```

Solution

▼ Click here for the answer.
Make sure that your solution looks like this:

```
Feature: Search for pets by category
As a pet shop customer
I need to be able to search for a pet by category
So that I only see the category of pet I'm interested in buying
```

Define the initial state

It is always good to have some initial data in the database to run the tests against. This also enables everyone who reads the feature file to know the initial state of the system before testing. In BDD testing, we use the `Background:` keyword, along with the `Given` keyword, to define the initial state as a table of data. The scenarios will use this data to testify that the feature is working correctly.

The database table for pets contains the columns `name`, `category`, and `available`. For this feature, you will define three pets with the names 'Fido', 'Kitty', and 'Leo', who have a category of 'dog', 'cat', and 'lion' respectively. Both Fido and Kitty are available to buy (available=True), while Leo is not (available=False).

Your Task

Start by opening the file `petshop.feature`.

Open **petshop.feature** in IDE

Start with the keyword `Background:` on one line. On the next line, use the `Given` keyword followed by a statement on the same line to indicate that the following pets should be in the database. Create a table starting on the next line that describes the pets "Fido", "Kitty", and "Leo" from the description above.

▼ Click here for a hint.
Here is the start of the table

```
Background:
    Given the following pets
        | {column} | {column} | {column} |
        | {data}   | {data}   | {data}   |
```

Solution

▼ Click here for the answer.
Make sure that your solution looks like this:

```
Background:
    Given the following pets
        | name    | category | available |
        | Fido    | dog      | True      |
        | Kitty   | cat      | True      |
        | Leo     | lion     | False     |
```

Describe the scenario

You are now ready to describe the scenario for “Search for dogs”. This is the series of steps that a customer would use to search for the dog that they want to buy. It is important to visualize the user interface that the customer will be manipulating and be specific about where they start, what actions they perform, and what results they observe.

Here is the scenario:

The customer will start on the “Home Page” of the pet shop. They will enter the word “dog” into the “Category” field, and then click the “Search” button. They will wait for the message “Success” to appear before looking at the results. They should see “Fido” in the results because Fido is a dog. They should not see “Kitty” or “Leo” in the results because Kitty is a cat, and Leo is a lion.

Your Task

Start by opening the file `petshop.feature`.

Open **petshop.feature** in IDE

Document the steps that the customer needs to take:

1. Use the `Scenario:` keyword with the title “Search for dogs”
2. Use the `Given` keyword to specify to start on the “Home Page”
3. Use the `When` keyword to describe the action of setting the category to “dog”
4. Use the `And` keyword to describe the action of clicking the “Search” button
5. Use the `Then` keyword to describe that they should see the message “Success”
6. Use the `And` keyword to state that they should also see “Fido” in the results
7. Use the `But` keyword to state that they should not see “Kitty” in the results
8. Use the `And` keyword to state that they should also not see “Leo” in the results

▼ Click here for a hint.
Here is the outline for the scenario

```
Scenario: Search for dogs
    Given {where to start}
    When {what to do}
    And {more to do}
    Then {what do you see}
    And {more things to see}
    But {some things not to see}
    And {more things not to see}
```

Please see an example below on how to write the sentence and follow the same for the rest:

```
Given I am on the "Home Page"
```

Solution

▼ Click here for the answer.
Make sure that your solution looks like this:

```
Scenario: Search for dogs
    Given I am on the "Home Page"
    When I set the "Category" to "dog"
    And I click the "Search" button
    Then I should see the message "Success"
    And I should see "Fido" in the results
    But I should not see "Kitty" in the results
    And I should not see "Leo" in the results
```

Final Solution

The final feature now describes the feature and the user story that it addresses. It also defines the initial state of the system before the tests are run, and it describes the scenario for searching for dogs.

Solution

Your `petshop.feature` file should look like this:

```
Feature: Search for pets by category
As a pet shop customer
I need to be able to search for a pet by category
So that I only see the category of the pet I am interested in buying
Background:
    Given the following pets
        | name    | category | available |
        | Fido    | dog      | True      |
        | Kitty   | cat      | True      |
        | Leo     | lion     | False     |
Scenario: Search for dogs
    Given I am on the "Home Page"
    When I set the "Category" to "dog"
    And I click the "Search" button
    Then I should see the message "Success"
    And I should see "Fido" in the results
    But I should not see "Kitty" in the results
    And I should not see "Leo" in the results
```

Conclusion

Congratulations! You just completed the **Writing Feature Files** lab.

Hopefully, you now understand how to create a feature file for testing the behavior of a system. You described the feature using the User Story method of: **As a** user, **I need** some capability, **So that** I gain some business value. You also defined the initial state of the system using the `Background:` keyword. Finally, you describe the scenario in step by step detail to ensure that the feature is behaving as expected.

In a future lab, you will write steps for this feature file and use the `behave` tool to run the scenarios to test that the system is working properly.

Author(s)

John J. Rofrano