

Testing using Cucumber in Ruby/JRuby

Overview

In this tutorial, we will create some tests in Cucumber for Ruby (or JRuby).

The test (specification) is initially created in Jira as a Cucumber Test and afterwards, it is exported using the UI or the REST API.

Requirements

- Install Ruby or JRuby
- Install the "cucumber" gem

Description

After creating a Cucumber Test, of Cucumber Type "Scenario Outline", in Jira, you can export the specification of the test to a Cucumber .feature file via the REST API or the **Export to Cucumber** UI action from within the Test Execution issue.

The created file will be similar to the following:

1_CALC-889.feature

@REQ_CALC-889

Feature: As a user, I can calculate the sum of 2 numbers

@TEST_CALC-908 @UI @core

Scenario Outline: Cucumber Test As a user, I can calculate the sum of 2 numbers

Given I have entered <input_1> into the calculator

And I have entered <input_2> into the calculator

When I press <button>

Then the result should be <output> on the screen

Examples:

| input_1 | input_2 | button | output |
|---------|---------|--------|--------|
| 20 | 30 | add | 50 |
| 2 | 5 | add | 7 |
| 0 | 40 | add | 40 |
| 4 | 50 | add | 54 |

After running the tests and generating the Cucumber JSON report (e.g., [data.json](#)), it can be imported to Xray via the REST API or the **Import Execution Results** action within the Test Execution.

```
cucumber -x -f json -o data.json
```



Please note

As the report format in Cucumber JSON is being deprecated in favour of [Cucumber Messages](#), a protocol buffer based implementation, the previous command needs to be adapted slightly.

The report starts by being generated in Cucumber Messages, using "-f message" argument, and then converted to the legacy Cucumber JSON report using the tool [cucumber-json-formatter](#).

```
cucumber -x -f message -o data.ndjson
cat data.ndjson | cucumber-json-formatter --format ndjson > data.json
```

The execution screen details will not only provide information on the test run result, but also of each of the examples provided in the Scenario Outline.



The Cucumber Scenarios Example/Result details (i.e., **Hooks**, **Backgrounds** and **Steps**) are only available for executions done in Xray v2.2.0 and above.

Test Details

Test Type: Cucumber

Scenario Type: Scenario Outline

Scenario:

```
1 Given I have entered <input_1> into the calculator
2 And I have entered <input_2> into the calculator
3 when I press <button>
4 Then the result should be <output> on the screen
5
6 Examples:
7   | input_1 | input_2 | button | output |
8   | 20      | 30      | add    | 50      |
9   | 2       | 5       | add    | 7       |
10  | 0       | 40      | add    | 40      |
11  | 4       | 50      | add    | 54      |
```

Examples

| <input_1> | <input_2> | <button> | <output> | Duration | Status |
|--|-----------|----------|----------|---------------|--------|
| 20 | 30 | add | 50 | 128 millicsec | PASS |
| Hooks | | | | | |
| Before features/step_definitions/calculator_steps.rb:7 | | | | 0 millicsec | PASS |
| After features/step_definitions/calculator_steps.rb:11 | | | | 0 millicsec | PASS |
| Background | | | | | |
| Given a calculator I just turned on | | | | 128 millicsec | PASS |
| Steps | | | | | |
| Given I have entered 20 into the calculator | | | | 0 millicsec | PASS |
| And I have entered 30 into the calculator | | | | 0 millicsec | PASS |
| When I press add | | | | 0 millicsec | PASS |
| Then the result should be 50 on the screen | | | | 1 millicsec | PASS |
| 2 | 5 | add | 7 | 0 millicsec | PASS |
| 0 | 40 | add | 40 | 0 millicsec | PASS |
| 4 | 50 | add | 54 | 1 millicsec | PASS |



The icon represents the evidences ("embeddings") for each **Hook**, **Background** and **Steps**, but is only available for executions done in Xray v2.3.0 and above.



Learn more

Please see [Testing in BDD with Gherkin based frameworks \(e.g. Cucumber\)](#) for an overview on how to use Cucumber Tests with Xray.

References

- <https://cucumber.io/docs/reference/ruby>
- [Automated Tests \(Import/Export\)](#)
- [Exporting Cucumber Tests - REST](#)