Maximize the efficiency of using Cucumber Scenarios in Xray

(i) What you'll learn

- The shortcomings of "typical" Cucumber Scenarios
- · How BDD Steps Library helps with reusability
- How data-driven Cucumber testing optimizes scaling & efficiency

Overview

- Parameterized Cucumber Tests
 - Data-driven precondition
- Tips
- References

Overview

Let's say we have five Cucumber tests that validate the same flow but with different data permutations. Furthermore, since several users created these tests manually, we can see slight discrepancies in the step wording.

Scenario

- 1 Given I open the store page as a Guest customer during the Regular period
- 2 When I add 1 productA items to the cart
- 3 And I apply the discount code
- 4 And I click "Purchase"
- 5 Then The transaction is successful

Scenario

- 1 Given I open the store page as a VIP customer during the Promotion period
- 2 When I add 3 productA items to the cart
- 3 And I apply the discount code
- 4 And I click "Purchase"
- 5 Then The transaction is successful

Scenario

- 1 Given I open the store page as a Guest customer during the Promotion period
- 2 When I add 2 productB items to the cart
- 3 And I do not apply the discount code
- 4 And I click "Purchase"
- 5 Then The transaction is successful

Scenario

- 1 Given I open store page as a Registered customer during the Regular period
- 2 When I add 2 productB items to cart
- 3 And I do not apply discount code
- 4 And I click "Purchase" button
- 5 Then Transaction is successful

Scenario

- 1 Given I open store page as a Guest customer during Promotion period
- 2 When I add 1 productC items to cart
- 3 And I do not apply discount code
- 4 And I click "Purchase" button
- 5 Then Transaction is successful

Why is this problematic? Having 5 test issues means 5x maintenance whenever the specification needs to change. Further, manual multi-user creation or adjustment leads to messier step definitions for automation engineers (as we can have slightly different steps, with different wording, that perform exactly the same actions).

In this tutorial, we will optimize the efficiency of this mini regression suite via data-driven automation with the help of BDD Steps Library and Parameterized Tests features of Xray.

BDD Steps Library

First, we will consolidate the steps and improve future reusability. Xray indexes Gherkin steps automatically in the library whenever users create/edit Cucumber-type Tests/Preconditions. We will open the project library from the Testing Board.

í	1	FINDEMO Software project		÷	4	Testing Board
	F	Backlog	^		Q	Getting Started
I	Ш	Active sprints			@ 1	Test Case Designer
I	~	Reports			<u>مہ</u>	Test Papasitory
	0	Goals			ര്പ	BDD Step Library
ļ	⊵	lssues			(1) 同	Test Plans
I	Ë	Components NEW			<u>ک</u>	Test Executions
C	DEVE	LOPMENT			REPOR	TS
ł	≞	Releases				Test Coverage
1	F	Project pages			، د	Traceability
I	£1 &	Pick Pagistar				Test Plans Metrics
-	<u> </u>	Structure			▦	Tests List
	2	Risks			⊞	Test Sets List
ć		ScriptRunner Enhance			⊞	Test Plans List
	4	Testing Board			⊞	Test Executions List
6	<u>گ</u>	Test Case Designer			⊞	Test Runs List

Now, we will choose any redundant steps and edit it to have the desired syntax version. We will parameterize the dynamic parts with descriptive variable names. We will also surround those parameters in quotes as an optional good practice.

Q I open Search by label *				
Deprecated steps Only unused steps Only static steps				
I open the store page as a <var1> customer during the Promotion period None</var1>	Definition I I open the store page as a " <customertype>" customer during the "<promotion>" period</promotion></customertype>			
I open the store page as a Guest customer during the Regular period None				
I open store page as a Registered customer during the Regular period None	This step is being used in 1 issue			
I open store page as a Guest customer during Promotion period None	Normal text ▼ B I ···· A × III II Ø ⊞ <> 99 -			
I open the store page as a Guest customer during the Promotion period None				

In the list of the steps on the left, we can see that Xray already extracted the notion of a parameter for one dynamic occurrence (<var1>). The rules for the automatic identification of data-driven elements are defined in the settings we visited earlier.

Step index heuristics	Quoted strings e.g. "red", "blue"
	Numbers e.g. 355, 2.3
	Enums e.g. LEFT, RIGHT, CENTER

Enums include uppercase strings (like VIP in our example). Also, docstrings and elements delimited by pipes are supported.

Once the library is updated for all step variations, we can go to one of the Cucumber tests for which we've done the edits (using the link under the definition).

Definition

1	I	open	the	store	page	as	а	" <customertype>"</customertype>	customer	during	the	" <promotion>"</promotion>	period
	_												
										Т	his st	ep is being used i	n 1 issue(s).
											1115 51	ep is being used i	ii issue(s).

We adjust its whole Scenario definition using the auto-complete functionality (for the steps that have not been updated automatically from the library). Remember to prompt the suggestions using the shortcut specified in the tooltip.

🔋 📃 Test deta	I do not apply the discount code step A	
	I onen store nage as a Guest cu sten	Test Repository
Test Type	I open the store page as " <customertype>" customer during "<promotion>" period</promotion></customertype>	
Cummhan	I open the store page as " <cust step<="" td=""><td></td></cust>	
Cucumber	I open the store page as a Gues… step	Dataset
	I prepare <browser> on <platfor step<="" td=""><td></td></platfor></browser>	
	I select <option> with <amount> step</amount></option>	
Scenario	Precondition is completed step -	
1 Given	I	
2 When I	add 1 productA items to the cart	
3 And I	apply the discount code	
4 And I	click "Purchase"	
5 Then T	he transaction is successful	
	Press Ctrl + Space to get step suggestions.	~ X

The final form of our data-driven Scenario looks like this:



- 4 And I click "Purchase" button
- 5 Then Transaction is successful

Once the step optimization is completed in this updated "master" test issue, we can delete or archive the other four Cucumber tests and remove the outdated iterations of steps from the library (keep in mind, steps not marked as "static" are automatically deleted when they are no longer being used by any issue).

If tests require the same actions, you can select the option from the suggested list instead of typing. For new actions, consider creating them in the BDD Steps Library first, rather than in the test issues - that way you can avoid redundancy and improve consistency upfront.

Next, we will complete the upgrade to data-driven automation using parameterized Cucumber tests.

Parameterized Cucumber Tests

With the variables syntax (<>) already in place, we only need the dataset to provide the permutations for the execution. We can add it from the same screen by specifying the data manually or importing it from CSV.

📒 Test details	Preconditions	Test Sets	🗾 Test Plans	D Test Runs	
					Test Repository
Test Type					
Cucumber	~				🞛 Dataset
Scenario					
1 Given I	open the store page	as " <customert< td=""><td>ype>" customer d</td><td>uring "<promotion>" per</promotion></td><td>iod</td></customert<>	ype>" customer d	uring " <promotion>" per</promotion>	iod
2 When I a	dd " <quantity>" "<p< td=""><td>roductType>" it</td><td>ems to cart</td><td></td><td></td></p<></quantity>	roductType>" it	ems to cart		
3 And 1 "<	Discount>" discount	code			
4 And I cl	ick "Purchase" butt	on			

5 Then Transaction is successful

There are a total of 5 iterations to execute.

A

(i)

In this case, we recreated the same permutations that were present in the five original tests. Just keep in mind that one of the benefits of this data-driven approach is that you can easily scale the dataset up or down without worrying about adjusting multiple test issues one by one.

#	CustomerType	··· Promotion	 Quantity	 ProductType	 Discount
1	guest	Regular	1	ProductA	apply
2	VIP	Promotion	3	ProductA	apply
3	guest	Promotion	2	ProductB	do not apply
4	registered	Regular	2	ProductB	do not apply
5	guest	Promotion	1	ProductC	do not apply

Additionally, you could consider leveraging the Test Case Designer feature of Xray Enterprise to not only generate more complex datasets faster but also access data coverage visualizations.

When we get to the execution, here is what you see in the Test Run details - several iterations of the same Scenario, with the data values replacing parameter syntax, and the execution status per data permutation.

Test details CUCUMBER SCENARIO OUTLINE		
* Examples S		$\overline{\tau}$
Example 1 - guest Regular 1 ProductA apply	то ро	45
✓ Steps		
1 Given I open the store page as "guest" customer during "Regulan" period 2 When I add "l" "ProductA" items to cart 3 And I "apply" discount code 4 And I click "Purchase" button 5 Then Transaction is successful		
Example 2 - VIP Promotion 3 ProductA apply	TO DO	56
> Example 3 - guest Promotion 2 ProductB do not apply	то ро	%
> Example 4 - registered Regular 2 Product8 do not apply	то ро	%
> Example 5 - guest Promotion 1 ProductC do not apply	то ро	

Data-driven precondition

As an optional exercise, let's say we notice that many different tests will use the Given step as the precondition. We then decide to separate it out.

😑 Test details	Preconditions	💽 Test Sets 🛛 🗾 Te	st Plans 💽 Test Runs					
				Test Repos	sitory			
Test Type								
Cucumber	~			🖽 Data	aset			
Scenario 1 Giver Pr 2 When I a 3 And I "< 4 And I cl 5 Then Tra	econdition is complete dd " <quantity>" "<prod Discount>" discount co ick "Purchase" button nsaction is successful</prod </quantity>	d act ¹ ype>" items to d de	art					
StoreSet	tupPrecondition	on						
🖉 Attach	Create subtask	C Link issue	 Precondition details 	Risk assessment	•••			
Description								
Add a descrip	tion							
Precondition	details				7			
📃 Precon	dition details 0 Te	ests						
Cucumber	~							
Background								
1 Given I open the store page as " <customertype>" customer during "<promotion>" period</promotion></customertype>								

Here is the tricky part - the dataset, including parameters from both the test and the precondition, is always in the **test issue**. Whenever you have a datadriven precondition, you must ensure the dataset for **each** associated test has the parameters and values specified in that precondition.

So, in our example, we can execute the updated test issue without any changes to the dataset to see this breakdown in the Test Run details. Notice that the value replacement highlighted in blue is exactly the same as before we had a precondition.

Test details CUCUMBER SCENARIO OUTLINE		
 Examples (\$) 		$\overline{\mathbf{x}}$
Y Example 1 - guest Regular 1 ProductA apply	TO DO	95
Preconditions 1		
✓ FIN-112 - StoreSetupPrecondition		
1 Given I open the store page as "guest" customer during "Regular" period		
✓ Steps		
1 Given Precondition is completed 2 When I add "1" "ProductA" items to cart 3 And I "apply" discount code 4 And I Click "Purchase" button 5 Then Transaction is successful		
Example 2 - VIP Promotion 3 ProductA apply	TO DO	55 🔜 🔜 🔜
Example 3 - guest Promotion 2 ProductB do not apply	TO DO	95
Example 4 - registered Regular 2 Product8 do not apply	TO DO	95

Tips

• If the BDD Steps Library feature is out of sync for any reason, you can launch Reindex from the project settings to make sure all the steps you need are in the library.

FINDEMO Saftware project Back to project Details Summary	Xray Settings Summary Macellaneous Remote Jobs Trigger Test Coverage	BDD Step Library DRAW Whenever EDD steps are created or modified the following fields represent heuristics ap users from seeing the same step (with diff	Hed. Yoy will indee the steps to be displayed in a 8000 library, beloning will process all atraps from a given t splicit by Xoy whenever indexing 1000 regs. Xoy can recognize multiple tags with different values as all event values) deplicated within the 800 library. Check the documentation here.	ent (or precondition) and store them in a BOO Use a different literary gile stop with variables when these heuristics a Reindex
People Permissions Notifications Automation Features Toolchain Viorkflows	Defect Mapping Test Types Test Environments Document Generator Test Step Fields Test Run Custom Fields Parameter value Itels	A results operation of the ELD a Step index heuristics Permissions	erge mot er und segrete an a subsegreve aux memo segi Aldel Robello del Calagdo. Consequente del single aj "ref", "bia" Numbers aj 355, 23 Entras aj UTI: BORT, CONTRE Select. Conse span in tea an etti si tea y, sg. Attrapatetto prenig seg, or configureg seg, tre deception nel 7 a no 3 approximate al can contell leng deg.	v
Issues Components NEW Apps Risk register Stack integration Xnay Settings	BDD Step Levary Default Column Layouts Re-Indexing	Save		

You can track the status from the same triple-dot menu.

li	Use a different library
а	Reindex in progress

- Ensure the parameter name syntax is the same between the script and the dataset.
- Consider adding the project-level lists in Xray settings for parameter values that you frequently reuse.
- If necessary, you can override the test-level dataset with the one in the test plan or the test execution.
- You can achieve the same results using the Scenario Outline format with the Examples table in the definition (like having embedded data) and no dataset defined at the Test level; Xray supports this implicit data in case you want to embed it directly in the Gherkin specification. It depends on the dataset size and if you wish to mix it with the script for review clarity purposes.

Cucumb	ber 🗸	 Dataset
cenario		
1	Given I open the store page as " <customertype>" customer during "<promotion>" period</promotion></customertype>	
2	When I add " <quantity>" "<producttype>" items to cart</producttype></quantity>	
3	And I " <discount>" discount code</discount>	
4	And I click "Purchase" button	
5	Then Transaction is successful	
6		
7	Examples:	
8	CustomerType Promotion Quantity ProductType Discount	
9	guest Regular 1 ProductA apply	
10	VIP Promotion 3 ProductA apply	
11	guest Promotion 2 ProductB do not apply	
12	registered Regular 2 ProductB do not apply	
13	guest Promotion 1 ProductC do not apply	

References

- BDD Steps Library
- Parameterized Tests