

Scripts



- Overview
- Automate
 - How to create
 - Gherkin Feature File
 - Edit Automate scripts
 - Delete Automate scripts
- Manual Scripts
 - How to create
 - Incorporating “Parameterized expected results” into your plans
 - Essential Usage Tips and things to know about the Expected Results feature

Overview

Scripts are line-by-line descriptions containing information about the system transactions that should be performed to validate the application or system under test. The test script should list each step that should be taken with the expected results.

The Xray Test Case Designer Scripts feature creates detailed, consistent execution instructions with conditionally expected results for your testing efforts.

You can quickly transform optimized test data like this

A) Airplane Ticket Reservation 1

Search... 14 scenarios and 154 2-way interactions i

# ^	Flying From	Flying to	Class	Seat Preference	Adults	Children	Amount of luggage
1	India	the United States	Coach	None	1	0	0
2	the Philippines	the Philippines	Business	Selected	More than 1	0	1
3	the United States	India	First	Selected	1	0	2 or more
4	India	India	Business	None	More than 1	1	0
5	the United States	the United States	Coach	Selected	More than 1	1	1
6	India	the Philippines	First	None	1	More than 1	0
7	the Philippines	India	Coach	None	1	More than 1	1
8	the United States	the United States	Business	None	More than 1	More than 1	2 or more
9	the Philippines	the Philippines	Coach	None	1	1	2 or more

... into customizable scripts. You can even add automatically generated Expected Results to your steps if you want to.

 Clear Auto-Scripts

Step 1 In the "Flying from" text box, type **India**.

Step 2 In the "Flying to" text box, type **the United States**.

  **when** Flying to is **India**

then Make sure the special "Incredible India" discount is applied.

Automate

Automate scripts make it easy to write behavioral-driven development (BDD) automated acceptance tests in Gherkin format as data-driven scripts. These data-driven test scripts are based on a test model, making them more maintainable than traditional BDD/Gherkin test scripts.

Automated test scripts are notoriously expensive to maintain, and BDD/Gherkin test scripts are not immune to this problem. Typically changes to the system under test or changes to how the system will be tested mean having to make changes to tens, hundreds, or (in some cases) thousands of feature files. With data-driven Automate test scripts, that's not the case. Adding or removing parameter values, requirements, or constraints to the test model updates the data driving Automate scripts, but there is no need to update the scripts themselves.

In addition, Automated scripts make it easier to write correct Gherkin through its syntax highlighting and autocomplete features.

How to create

Create or open a test model and navigate to the "Scripts" menu. Once there, select the "Automate" option in the menu.

You can create as many Gherkin/Robot Framework files as you need for your test model, one per tab. Click the "+" button in the tab bar to add another file. Click the pencil icon on a tab to rename a file and the trash can icon to delete a file.

Gherkin Feature File

The Automate screen is an intelligent Gherkin feature file editor. It is aware of the parameters, values, & generated test cases of the current model and the Gherkin syntax. It makes it easy to create a feature file with data-driven Scenario and/or Scenario Outline sections populated by the model's generated test cases.

Feature

Each Gherkin script has a Feature as its first section keyword followed by ":" and a little bit of text to describe the scope, usually in the form of a user story which is demonstrated in the placeholder text. This section does not support parameterization of any kind.

Save (Cmd+S)

2-way interactions 


- 1 **Feature:** Customize a Tesla Vehicle
- 2 As a visitor to the Tesla website
- 3 A user should be able to customize a Model S or Model X
- 4 And see how much their ideal configuration would cost

Background

A Background section is useful when a set of initial setup steps is repeated for every Scenario and/or Scenario Outline in the feature file. The steps of the Background are run before each Scenario Outline and Scenario. Starting a new line in the Background and further sections brings up a suggestion dialog with standard Gherkin keywords.

Remember that restricting a parameter in the Background (with {}, as explained below) is valid syntax and applies only the selected value to all Scenarios or Scenario Outlines for a given Feature.

Save (Cmd+S)

2-way interactions 

- 1 **Feature:** Customize a Tesla Vehicle
- 2 As a visitor to the Tesla website
- 3 A user should be able to customize a Model S or Model X
- 4 And see how much their ideal configuration would cost
- 5
- 6 **Background:**
- 7 **Given** a user without an account
- 8 **And** the user is on the Tesla homepage

Scenario

Next in the feature file is the enumeration of different testing scenarios. One way to describe a test case is with a Scenario block. It contains sequential steps, expressed as plain English Given/When/Then statements, describing the necessary actions to execute the automated test script. If the test model generates 28 test cases, then 28 Scenario sections would need to be written out to achieve the combinatorial coverage.

However, instead of writing out all 28 test cases individually, with Automate, a single data-driven Scenario template that includes the <Parameter Name> syntax can generate all 28 needed Scenario sections in the exported feature file. You can check the number of associated test cases in the Preview panel.

<pre> 1 Feature: Customize a Tesla Vehicle 2 As a visitor to the Tesla website 3 A user should be able to customize a Model S or Model X 4 And see how much their ideal configuration would cost 5 6 Background: 7 Given a user without an account 8 And the user is on the Tesla homepage 9 10 Scenario: A user should be able to specify which upgrades, paint col 11 Given a user is customizing a <Model> 12 When they select the <Configuration> configuration 13 And Full Self-Driving Capability is <Full Self-Driving Capability> 14 And a Carbon Fiber Spoiler is <Carbon Fiber Spoiler> 15 And Rear Facing Seats are <Rear Facing Seats> 16 And they select the <Roof> option 17 Then the user should see an accurate final price for the <Payment> </pre>	<pre> 1 Feature: Customize a Tesla Vehicle 2 As a visitor to the Tesla website 3 A user should be able to customize a Model S or Model X 4 And see how much their ideal configuration would cost 5 6 Background: 7 Given a user without an account 8 And the user is on the Tesla homepage 9 10 # Scenario for test case 1 11 Scenario: A user should be able to specify which upgrade 12 Given a user is customizing a Model X 13 When they select the P100D configuration 14 And Full Self-Driving Capability is Equipped 15 And a Carbon Fiber Spoiler is N/A 16 And Rear Facing Seats are N/A 17 And they select the N/A option </pre>
--	--

For example, writing the test step:

And they select <Color> for the car's color.

This means that for each generated Scenario section (one for each test case), the <Color> portion of the test step will be replaced with the value of the Color parameter in the generated test case:

And they select Deep Blue Metallic for the car's color.

Simply typing "<" will populate a list of autocomplete suggestions related to all of the parameters in the model, also adding:

- Test Number (sequential number within the Scenario),
- Test Case (constant number of the row in the complete table on the Scenarios screen)
- Expected Outcome (from the Forced Interactions screen, if applicable)

Wherever the <Parameter Name> appears in the test steps, it will be replaced by the specified parameter's value from the associated row on the Scenarios screen. Automate feature files can contain as many data-driven Scenario sections as needed to automate all the testing ideas contained in the model.



it is common to put dynamic elements into quotes (e.g., they select "<Color>" for the car's color) to make them more visually distinguishable).

Scenario Outline

A Scenario Outline behaves almost exactly like a data-driven Scenario. When using a Scenario Outline, instead of having 28 Scenario sections in the export (one for each test case), the export has one data-driven Scenario Outline script block with a Gherkin Examples data table attached to it that contains 28 rows (one for each test case).

The Preview panel shows the first 10 rows of the Examples data table for each Scenario Outline. The order of parameters in the Examples is dictated by the script, not by the Parameters screen.

1	Feature: Customize a Tesla Vehicle	10	Scenario Outline: A user should be able to specify which upgrades
2	As a visitor to the Tesla website	11	Given a user is customizing a <Model>
3	A user should be able to customize a Model S or Model X	12	When they select the <Configuration> configuration
4	And see how much their ideal configuration would cost	13	And Full Self-Driving Capability is <Full Self-Driving Capability>
5		14	And a Carbon Fiber Spoiler is <Carbon Fiber Spoiler>
6		15	And Rear Facing Seats are <Rear Facing Seats>
7	Background:	16	And they select the <Roof> option
8	Given a user without an account	17	Then the user should see an accurate final price for the <Payment>
9	And the user is on the Tesla homepage	18	Begin expansion (10 examples shown of the 25 unique & applicable test cases)
10	Scenario Outline: A user should be able to specify which upgrades, p	19	Examples:
11	Given a user is customizing a <Model>	20	
12	When they select the <Configuration> configuration	21	Model Configuration Full Self-Driving Capability Ca
13	And Full Self-Driving Capability is <Full Self-Driving Capability>	22	Model X P100D Equipped N/
14	And a Carbon Fiber Spoiler is <Carbon Fiber Spoiler>	23	Model S 75D Unequipped Ec
15	And Rear Facing Seats are <Rear Facing Seats>	24	Model X 75D Unequipped N/
16	And they select the <Roof> option	25	Model S 100D Equipped Ur
17	Then the user should see an accurate final price for the <Payment>	26	Model X 100D Equipped N/
18		27	Model S P100D Equipped Ec
19			Model X P100D Unequipped N/

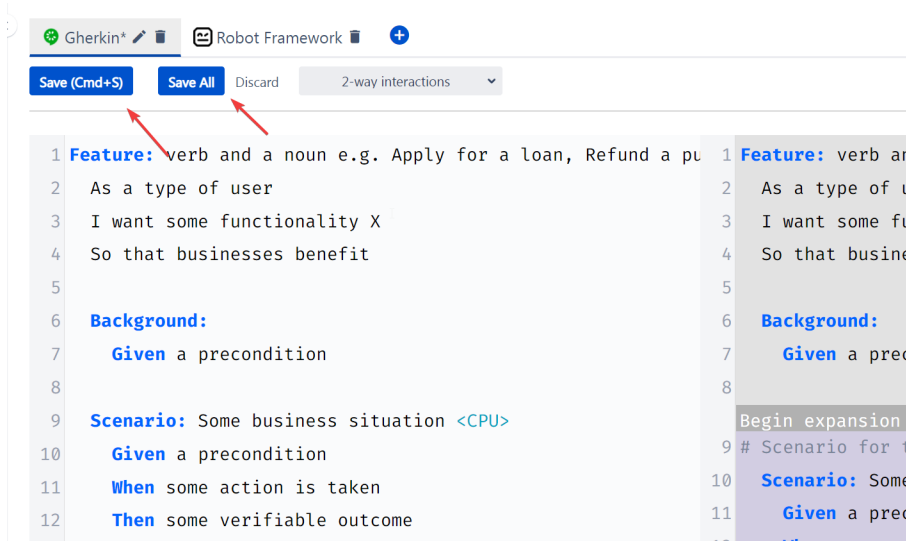
An Automate feature file can contain as many Scenario Outline sections as needed to automate all the testing ideas contained in the model.

You can include both the Scenario and Scenario Outline sections in the same feature file. Where possible, using Scenario Outline is preferable to Scenario as it minimizes the number of items in test management and automation frameworks, which simplifies planning, execution, reporting, and maintenance.

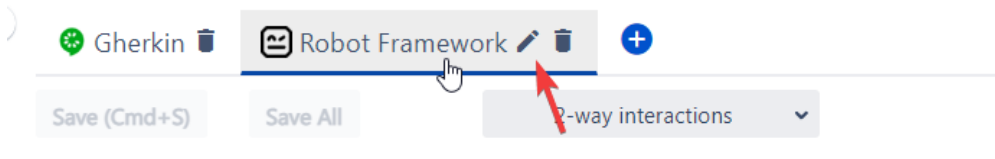
Edit Automate scripts

To edit a script you need to make the necessary changes within the editor and save the changes by using the buttons for that effect or using the keyboard shortcuts:

Keyboard shortcuts	
Save the script in the active tab	Ctrl+S
Save all the scripts in the test model	Ctrl+Shift+S
Open the auto-complete dropdown	Ctrl+Space
Undo / Redo	Ctrl+Z / Ctrl+Y

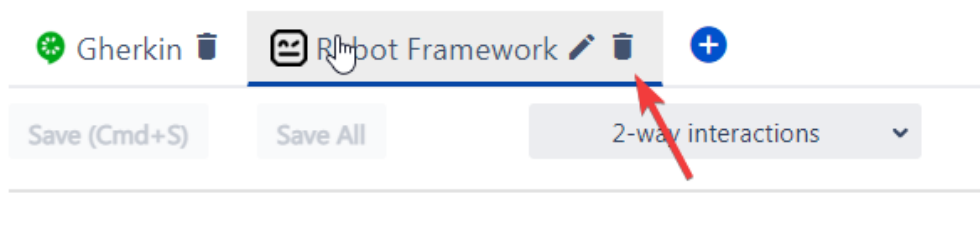


To edit the script name, click the option right to the name on the tab:



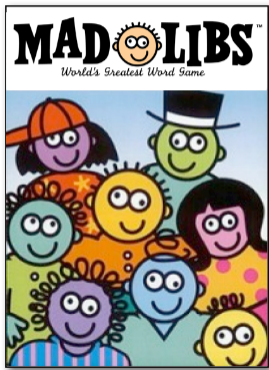
Delete Automate scripts

To delete an automated script, select the tab and click the bin icon right after the name:



Manual Scripts

Remember Mad Libs?



Letter Received by the Father of a Marriageable Daughter

Dear _____,
NAME OF MAN IN ROOM

I am in love with your _____
ADJECTIVE

daughter, _____ and would like
NAME OF GIRL

to ask for her _____ in marriage.
NOUN

She is my idea of a perfect _____.
NOUN

She is the only _____ I have ever loved.
NOUN

Creating manual scripts in Xray Test Case Designer is similar to that. Instead of adding adjectives and nouns into pre-formed sentences, you'll be more like the author of the Mad Libs sentences themselves. You need to:

1. Create sentences containing execution instructions that will be common to most of the test scripts and
2. Identify "spaces" to indicate where Xray Test Case Designer should "fill in the blanks" you've left with test data appropriate for each scenario.

How to create

1

First, navigate to the Scripts -> Manual screen and (optionally) add instructions to be completed before execution for **all** these scenarios begins. (i.e., the details in the "Start" field must be common across the board).

2

Next, click on the "pencil" icon to enter your first test step instructions. Alternatively, you may click the text already present for the step. Enter detailed instructions for a tester for each step. For now, type Mad Libs-like sentences, as shown below, with blank lines to indicate where Values are to be inserted.

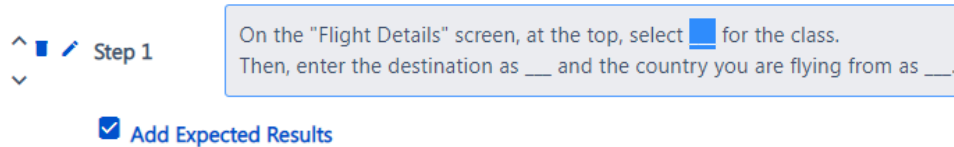
As shown above, for example, you will want to type the words that will remain the same from test to test and leave 3 blanks (one for each place that Values will change from test to test):

- One blank for the class,
- One blank for the destination country, and
- One blank for the origin country.

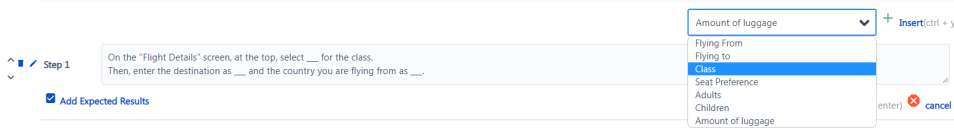
3

Next, replace those blank lines with the appropriate Parameter names.

- Highlight the first blank line.



- Confirm that the Parameter Name is to be inserted in the Parameter Name drop-down list (adjust if necessary)



- Press “CTL-Y” on the keyboard (or the “Insert” link if you prefer)

When you’re entering and editing your Manual Scripts, you’ll notice that sentences probably look strange, with the words inside the curvy brackets { } not “reading” well within the structure. The trick is to think about what your steps will look like when the names of the actual Values are inserted into each sentence. As soon as you save each step, sanity prevails - you will see the “normal” sentences you’ve constructed (with Coach as the example below).

- Press “CTL-Enter” on the keyboard (or the “Add/Save” link)

Step 1

On the "Flight Details" screen, at the top, select **Coach** for the class.
Then, enter the destination as [blank] and the country you are flying from as [blank].

The words that are the same between tests are in standard text. Words that change from test to test (the Values you entered on the Parameters screen) are shown in bold.

- Rinse and repeat for the other blank lines.

Don't forget to save each step before you add your next one! Thankfully, Xray Test Case Designer notifies you that there are unsaved edits under the last edited step. Click on different test cases at the bottom half of your screen (preview section that mirrors the Scenarios screen) to see how your script steps will change. Finally, in the “Finish” section, you may want to add some instructions that will appear only once at the end of all of the scenario scripts.

Incorporating “Parameterized expected results” into your plans

In the tests shown above, for example, we might want to include this Expected Result every time the necessary values appear together in a test case:

When a customer flies to India, make sure the special "Incredible India" discount is applied.

1

In the Scripts -> Manual screen, find the specific test step you want to add your Expected Result to and hover over it

2

Click on "Add Expected Results."

Step 1

On the "Flight Details" screen, at the top, select **Coach** for the class.
Then, enter the destination as **the United States** and the country you are flying from as **India**.

when Flying to is India and is

then the special "Incredible India" discount is applied

Cancel Add (ctrl + e)

Here you're setting up simple "when / then" rules. Note that you're not restricted to rules with just 1 positive condition - you could also create a rule that reads "IS NOT" (click on "is" between dropdowns to switch the rule type).

If WHEN selection is blank, the expected result from the THEN field will apply to that step in all test cases, regardless of the test data.

Lastly, you can put parameter names with { } syntax inside the THEN statement – this is typically valuable in the validation use cases where the step would say "Enter X as {X}" and the expected result would say "Validate the X is shown as {X}." Parameter names do not have to match, in case you have already included actual expected results on the Parameters screen. In such cases WHEN conditions are often left blank.

❗ Essential Usage Tips and things to know about the Expected Results feature

1. This feature is a partial solution for straightforward Expected Results. It primarily exists so that you won't have to type many simple expected results manually. It is not designed to handle especially complex rules that you might have.

2. Be sure you understand the similarities & differences between Xray Test Case Designer Expected Results in the Manual Scripts screen and Expected Outcomes in the "Forced Interactions" one. There is a big, yet subtle, difference:

- Expected Results take the scenario data table as a "read-only" precondition and generate the "Then" content **ONLY IF** the conditions are satisfied (i.e., "reactive approach").
- Expected Outcome **guarantees** that the test conditions to satisfy it will be included in the Scenarios table at least once (i.e., "proactive approach," which may cause an increase in the number of test cases).

Suppose you want to define an Expected Result that requires 3 or more specific Values to appear in a single test script, and you're creating pairwise sets of tests. In that case, you should use the "Forced Interactions" feature or higher algorithm strength to guarantee the scenario is included in your suite. Then use the Manual Scripts feature to document the Expected Result for export.

Xray Test Case Designer Automate can directly leverage that last column from Forced Interactions as an internal variable.

If you want to define an Expected Result that requires **2 or fewer** specific Values to appear in a single test script (and you're creating pairwise sets of tests), use the Manual Scripts feature without additional prep work.