Reading Repetitive Nodes in JSON Files

Hands-on Guide

Tricentis

Contents

What is this document for?

This hands-on guide is designed to help you explore the process of Reading Repetitive Nodes in JSON Files and enhance your functional knowledge of the product.

In this document, you are provided with detailed instructions to walk you through the steps to achieve the desired result(s) for the listed assignment(s).

We encourage you to use this document to support your knowledge of Tosca, compare it with your alternate practical implementation method(s), and improve your overall know-how of the tool.

Disclaimer

Reading Repetitive Nodes in JSON Files

+ Objective

By the end of this exercise, you will be able to demonstrate how to:

- Create a Module by scanning a JSON file with Tosca API Scan Wizard and export it as an XML/JSON TestCase
- Modify the scanned Module to create a dynamic TestCase out of it
- Create a dynamic TestCase that can read repetitive nodes of a source JSON file
- Buffer the count of repetitive nodes in a JSON file

+ Why is this important?

JSON and XML files can be steered with Tosca by creating modules using the Tosca API Scan Wizard.

If you have a business use case to read repetitive nodes from a JSON file, it is possible to create a dynamic TestCase to handle and read the repetitive nodes irrespective of the number of the nodes.

+ Steps to perfprm

Here are the steps you need to perform to be able to read repetitive nodes from a JSON file. Please remember to follow the steps in sequence to achieve the desired results.

Section I : Scan the JSON file and create a Module out of it

Note

As mentioned in the pre-requisites, import the base Subset either at the project root level or in a new Component Folder to begin with the steps below.

1| Navigate to the Module section and create a new Module Folder, rename it to **JSON Files**

2| Right- click on the newly created Module Folder JSON Files, and from the context menu, select the option **Scan** and then click on **API**

3| In the API Scan Wizard, on the Ribbon menu **HOME**, click on **File**  to scan the JSON file

4| The file explorer window opens, navigate to C:\Tosca_Projects\ and select the JSON file **Ratings.json**

5| Click on **API Test Case** dropdown on the Ribbon menu and select **XML/JSON Test Case** to export it into Tosca Commander

Note

Make sure you click on the dropdown portion of the API Test Case icon 

6| Close the API Scan Wizard without saving it

7| You can see that the Module and the TestCase are created in a Component Folder. Cut the Module **Ratings Request** from the Component Folder and paste it to the newly created Module Folder **JSON Files**

8| Delete the Component Folder created during the API scan import. (The Folder named as **ApiScan_Import**)

9| In the Module Folder **JSON Files**, rename the Module **Ratings Request** to **Ratings JSON**

Section II: Make the TestCase dynamic

In continuation to the last section, follow the steps below.

1| In the Module **Ratings JSON** delete all the ModuleAttributes from **item#2** to **item#14**, keeping only **item#1**

Note

You can reach the level item#2 following the path: Ratings JSON-> RootObject-> Ratings -> item#2

2| Open the **Properties** pane of the ModuleAttribute **item#1** by clicking the expand arrow on the right-hand pane

3| Rename the value of the parameter **Name** from **item#1** to **item***

4| On the Properties pane add the Configuration Parameter **Explicit Name** following the steps:

a) First, make sure to select **item#1** on the working pane

b) Right click on the **{...}** icon on the top left corner of the Properties Pane

c) Select **Create Configuration Param** by clicking on the icon  on the dynamic menu

d) Rename it to **ExplicitName**

e) Set its value to **True**

Section III: Create TestCases to read repetitive nodes

In continuation to the last section, navigate to the TestCases section in the imported Subset and follow the steps below.

1| Create a new TestCase, rename it to **Reading repetitive nodes in JSON file**

2| Change the WorkState to **IN_WORK**

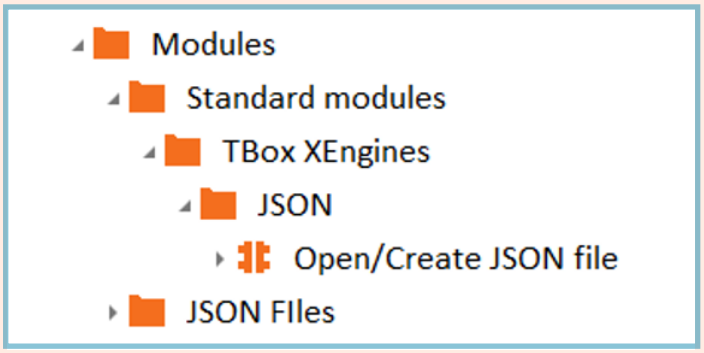
3| In the newly created TestCase, create two TestStepFolders and rename them to **Precondition** and **Process**

4| Drag and drop the Standard Module **Open/Create JSON file** to the TestCaseFolder **Precondition**.

Note

You can find it under the Standard Modules folder of the imported Subset. As you can see it in the image below:

Alternatively, use the fuzzy search by pressing Ctrl+T and select the Module - Open/Create JSON file.



5| Rename the created TestStep to **Open JSON file and create a resource**

6| Within the TestStep **Open JSON file and create a resource**, enter the TestStepValues as mentioned below:

TestStep Name	Name (TestStepValue)	Value	Action Mode
Open JSON file and create a resource	Resource	Ratings	Select
	Filepath	"C:\Tosca_Projects\Ratings.json"	Select

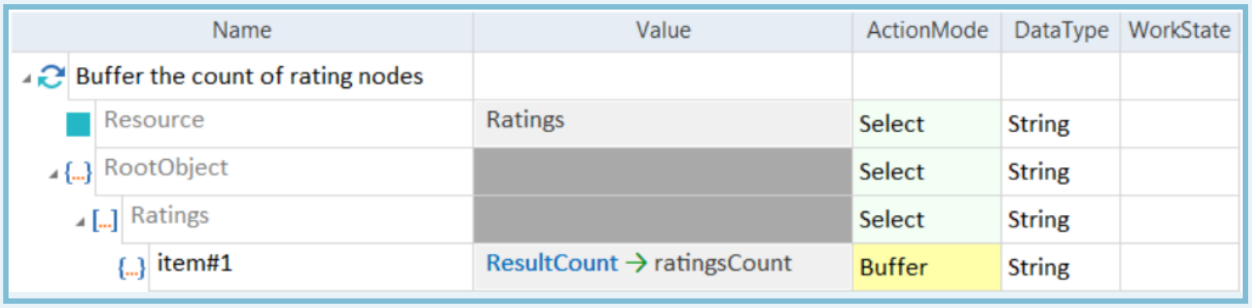
7| Drag and drop the Module **Ratings JSON** to the **Precondition** TestStepFolder

8| Rename the TestStep **Ratings JSON** to **Buffer the count of rating nodes**

9| Within the TestStep **Buffer the count of rating nodes**, enter the TestStepValues as mentioned in the below table:

TestStep Name	Name (TestStepValue)	Value	Action Mode
Buffer the count of rating nodes	Resource	Ratings	Select
	item#1	.ResultCount == ratingsCount	Buffer

For a visual representation of the previous step in Tosca, refer the image below :



Section IV: Buffer the count of rating nodes

1| Create a new TestStepFolder within the TestStepFolder **Process**. Rename it to **Iterate and store the ratings details**

2| Click on the newly created TestStepFolder **Iterate and store the ratings details** and open the Properties pane by clicking on the left arrow button in the working pane

3| Update the Property **Repetition** by entering **{B[ratingsCount]}** in its value. Observe that the icon of the TestStepFolder has changed, showing the repetition icon 

4| Drag and drop the Module **Ratings JSON** to the TestStepFolder **Iterate and store the rating details**

5| Rename the TestStep to **Read and buffer the ratings details**

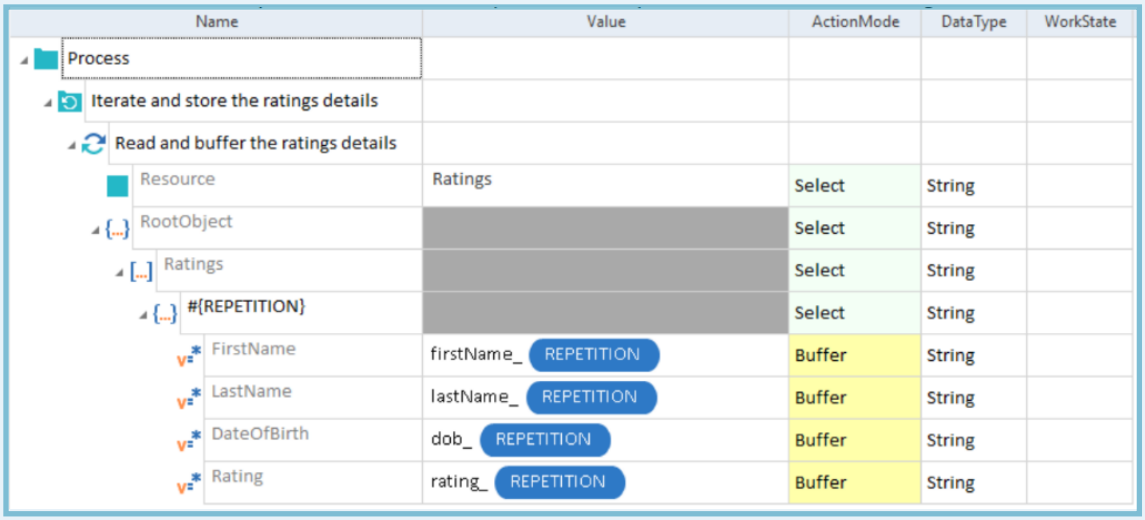
6| Set the TestStepValue Resource to **Ratings**

7| Update the Name of TestStepValue item#1 to **#{REPETITION}**

8| Within the TestStepFolder **Iterate and store the rating details**, enter the TestStep values as mentioned in the below table:

TestStep Name	Name (TestStepValue)	Value	Action Mode
Read and buffer the rating details	FirstName	firstName_{REPETITION}	Buffer
	LastName	lastName_{REPETITION}	Buffer
	DateOfBirth	dob_{REPETITION}	Buffer
	Rating	rating_{REPETITION}	Buffer

For a visual representation of the previous step in Tosca, refer the image below :



9| Mark the WorkState of the TestCase to **COMPLETED**

10| Execute the TestCase in ScratchBook

Expected result – The TestStepFolder repeated itself for the repetition set, and the details of the nodes were buffered in accordance with the data in the JSON file.

+ Observation

You will observe that the TestCase we created to read repetitive nodes in a JSON file:

1| Takes the count of repetitive nodes.

2| Repeated the test steps of reading the file content that many times and in each iteration, it reads and buffers the content present in each node.

3| This TestCase is dynamic and robust and would work successfully even if the number of nodes in this JSON file changes.