Read and Write data in Excel File

Hands-on Guide

Tricentis

Contents

What is this document for?

This hands-on guide is designed to help you explore the process of reading and writing the data in an Excel file using Excel 3.0 Engine in Tricentis Tosca 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

Read and Write data in Excel File – Use Case

+ Objective

By the end of these exercise(s), you will be able to demonstrate how to:

- Buffer the table content/data from an application
- Store the data into an Excel file
- Reading data from excel file

+ Why is this important?

You might come across use cases that require creating excel files and writing data into it or updating an existing excel file or reading data from excel file , this use case would help you understand the test steps to be created in Tosca to achieve it

+ Scenario

Scenario 1:

Open the Obstacle Course webpage where you will see a table with the information of individuals like First name, Last name and Email. You need to buffer each table value and save this into an Excel file, forming a similar table like structure.

+ Steps to perform

Here are the steps you need to perform to buffer the table values and write them into a newly generated Excel file.

Section 01: Precondition – Configure file path

> **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| Go to the TestCase section of imported Component Folder and locate the **TestCase Folder Scenario 01**

2| Switch to the **Test configuration** tab of the working pane

3| Create a new Test Configuration Parameter and set its name as **FilePath** and set its value to **C:\Tosca_Projects**

4| Switch back to the **Details** tab in the working pane and navigate to the TestCase **Write data in Excel file**

Section II: Precondition - Open URL and buffer the row count

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

1| Within the TestCase **Write data in Excel file**, navigate to the TestStepFolder **Precondition**

2| Add a TestStep using the **OpenUrl** Standard Module and rename it to **Open SUT**

Note

You can find this Module under the Standard Modules Folder of the imported Subset. Alternatively, use the fuzzy search by pressing Ctrl+T and select the required Module.

3| Add the TestStepValues as mentioned below:

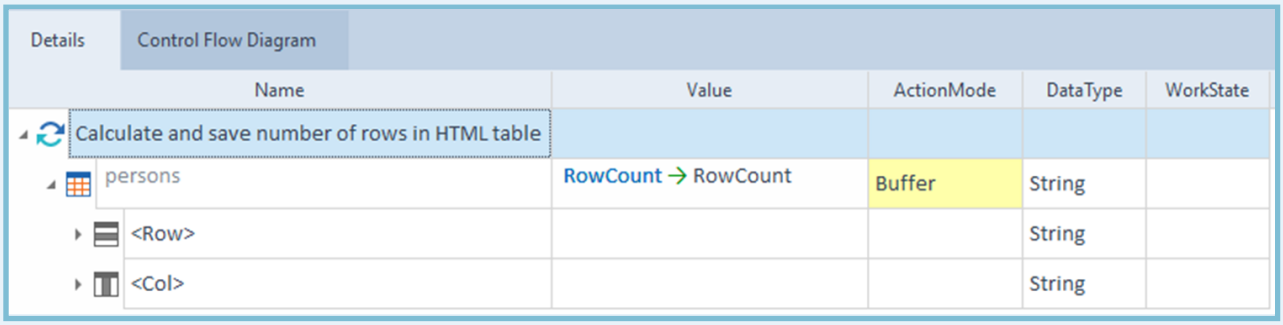
TestStep Name	Name	Value	Action Mode
Open SUT	Url	https://obstaclecours.tricentis.com/Obstacles/92248/	Input

4| Navigate to the TestStepFolder **Precondition**, add a TestStep using the Module **Test Automation Obstacle: Persons Table** from the module folder - Obstacle: **Table steering** and rename it to **Calculate and save number of rows in SUT**

5| Add the TestStepValues as mentioned below:

TestStep Name	Name	Value	Action Mode
Calculate and save number of rows in HTML table	Persons	.RowCount->RowCount	Buffer

For a visual representation of the previous step(s) in Tosca, refer to the image below:



Section III: Precondition – Buffer the time stamp

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

1| In the **Precondition** TestStepFolder, add another TestStep, using Standard Module **TBox Set Buffer** in which we want to save the current date and time

2| Rename it to **Set Time Stamp in buffer**

3| Add the TestStepValues as mentioned below:

TestStep Name	Name	Value	Action Mode
Set Time Stamp in buffer	DateTimeNow	{DATETIME}	Input

Section IV: Precondition – Open excel and define range

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

1| In the **Precondition** TestStepFolder, add another TestStep using the Standard Module **TBox Open Excel Workbook**

2| Rename it to **Open Excel Workbook - Personal details**

3| Add the TestStepValues as mentioned below:

TestStep Name	Name	Value	Action Mode
Open Excel Workbook - Personal details	Workbook Name	Personal details	Input
	Path	{CP[FilePath]}\Personal details_{B[DateTimeNow]}.xlsx	Input
	Create New	True	Input

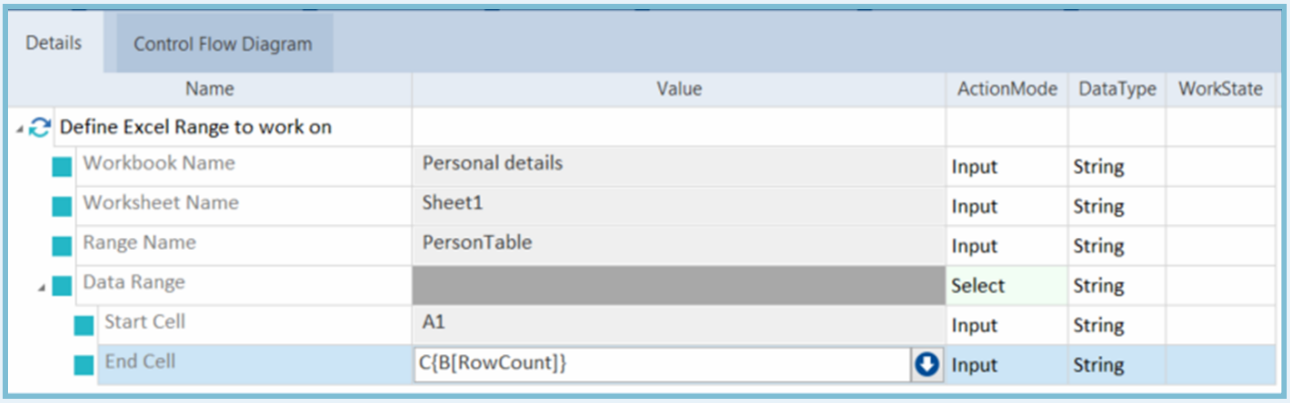
4| Again in the **Precondition** TestStepFolder Add a TestStep using Standard Module **TBox Define Excel Range** to define the Excel range in which the table data will be added

5| Rename it to **Define Excel Range to work on**

6| Add the TestStepValues as mentioned below:

TestStep Name	Name	Value	Action Mode
Define Excel Range to work on	Workbook Name	Personal details	Input
	Worksheet Name	Sheet1	Input
	Range Name	PersonTable	Input
	Data Range		Select
	Start Cell	A1	Input
	End Cell	C{B[RowCount]}	Input

For a visual representation of the previous step(s) in Tosca, refer to the image below:



Section V: Define Header Rows

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

1| Navigate to the TestStepFolder **Process** and add a TestStep using the Standard Module **TBox Excel Range Manipulation**

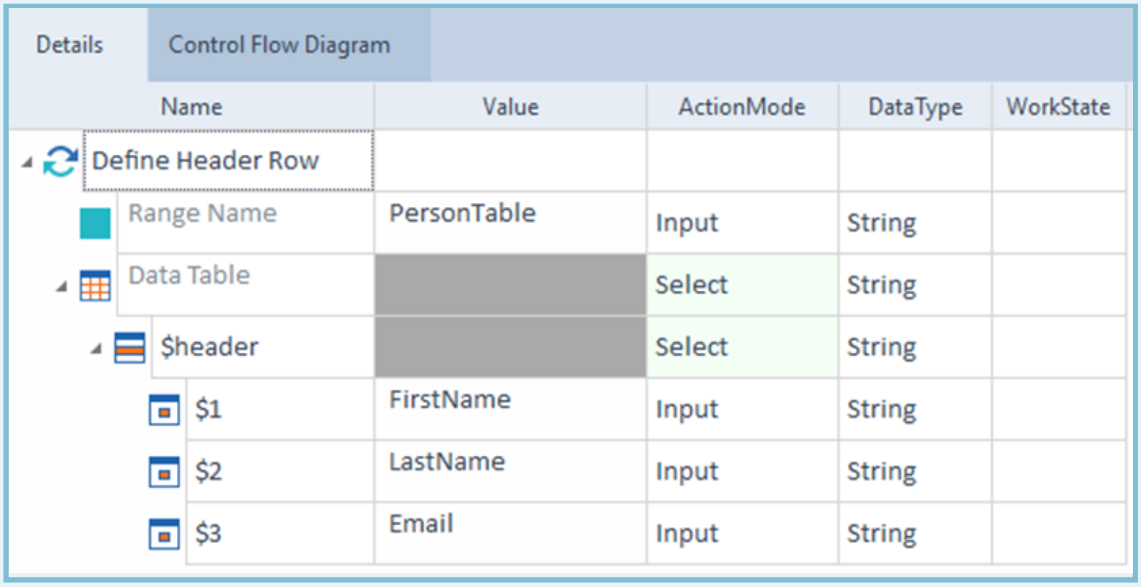
2| Rename it to **Define Header Row**

3| Add the TestStepValues as mentioned below:

Read and Write data in Excel File – Use Case

TestStep Name	Name	Value	Action Mode
Define Header Row	Range Name	PersonTable	Input
	Data Table		Select
	$header		Select
	$1	FirstName	Input
	$2	LastName	Input
	$3	Email	Input

For a visual representation of the previous step(s) in Tosca, refer to the image below:



Section VI: Read and Buffer the Table Data from the SUT and write it to the Excel file

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

1| In the TestStepFolder **Process** add another TestStepFolder **Store data from SUT to Excel File**

2| In the newly created Folder add TestStep using the Module **Test Automation Obstacle: Persons Table**

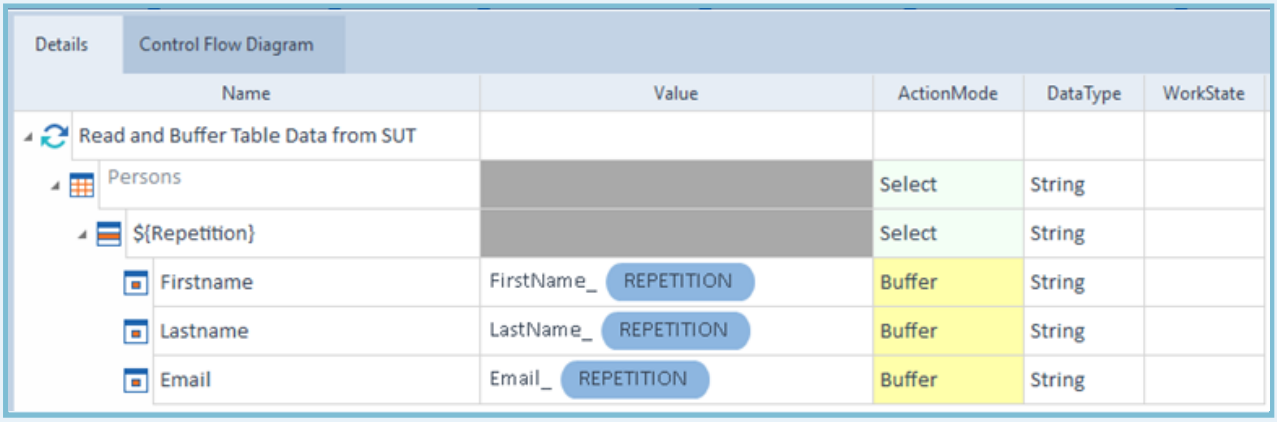
3| Rename it to **Read and Buffer Table Data from SUT**

4| Add the TestStepValues as mentioned below:

Read and Write data in Excel File – Use Case

TestStep Name	Name	Value	Action Mode
Read and Buffer Table Data from SUT	Persons		Select
	${Repetition}		Select
	Firstname	FirstName_{Repetition}	Buffer
	Lastname	LastName_{Repetition}	Buffer
	Email	Email_{Repetition}	Buffer

For a visual representation of the previous step(s) in Tosca, refer to the image below:



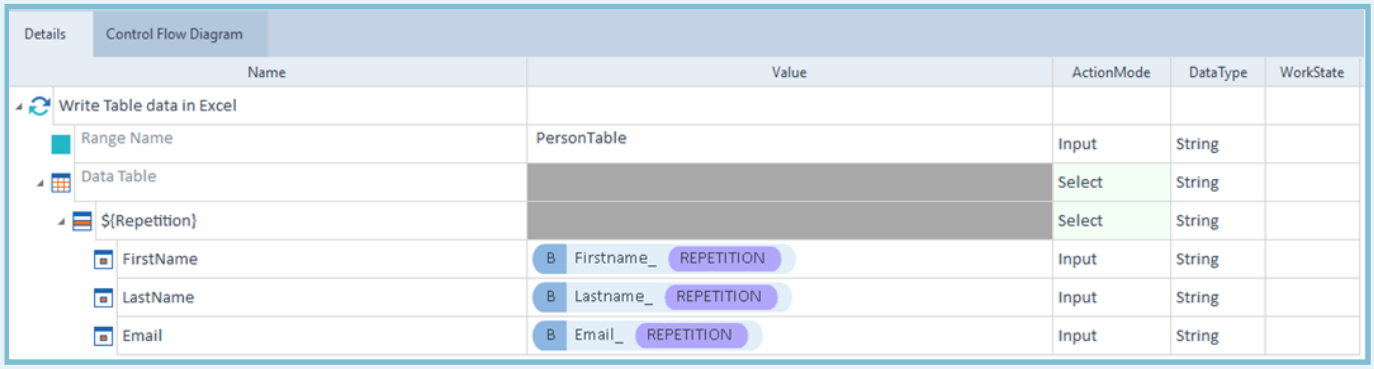
5| In the TestStepFolder **Store data from SUT to Excel File** add a TestStep using Standard Module **TBox Excel Range Manipulation**

6| Rename it to **Write Table data in Excel**

7| Add the TestStepValues as mentioned below:

TestStep Name	Name	Value	Action Mode
Write Table data in Excel	Range Name	PersonTable	Input
	Data Table		Select
	${Repetition}		Select
	FirstName	{B[Firstname_{Repetition}]}	Input
	LastName	{B[Lastname_{Repetition}]}	Input
	Email	{B[Email_{Repetition}]}	Input

For a visual representation of the previous step(s) in Tosca, refer to the image below:



8| Select the TestStepFolder **Store data from SUT to Excel File** and open the **Properties** pane by clicking the arrow icon on the right hand

9| Set the Repetition Property value to **{MATH[{B[RowCount]}-1]}**

Section VII: Save and close the Excel workbook

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

1| Navigate to TestStepFolder **Postcondition**. Add a TestStep using Standard Module **TBox Close Excel Workbook** to close and save the Excel file

2| Rename it to **Save and Close Excel Workbook - Personal details**

3| Add the TestStepValues as mentioned below:

TestStep Name	Name	Value	Action Mode
Save and Close Excel Workbook - Personal details	Workbook Name	Personal details	Input
	Path	{CP[FilePath]}	Input
	Save	True	Input

4| Mark the TestCase **Completed** Execute the TestCase **Write data in Excel File** in ScratchBook, and observe the changes

Expected result: You will now notice that the TestCase has passed and the HTML Table data has been saved in an Excel file.

Important note before starting the Scenario 02:

Please make sure you do not close the Obstacle challenge HTML application that you will use for the next Scenario.

If you closed the SUT, please Execute the TestCase **Write Data in Excel File from Scenario 1** and Do Not close the **Obstacle Course** webpage in your browser before you execute the TestCase for Scenario 02.

Scenario 02

Now consider a scenario exactly opposite to Scenario 1 where you are supposed to verify the data present in an excel file with SUT data. To achieve this we will read the data from the Excel file and verify against the SUT.

We will leverage the Excel file we created in Scenario 1 for this use case.

Here are the steps to perform for the verification of data in Excel file.

Section I: Precondition – Set file path and define range

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

1| Go to the TestCase section and locate the TestCase Folder **Scenario 02** and Switch to Test configuration tab of the working pane

2| Create a new Test Configuration Parameter, name it as **FilePath** and set its value to **C:\Tosca_Projects**

3| Switch back to the **Details** tab in the working pane and navigate to the TestCase **Verify data in Excel file**

4| Within the TestCase, navigate to the TestStepFolder **Precondition** and add a TestStep using the Standard Module **TBox Open Excel Workbook**

5| Rename it to **Open Excel Workbook- Personal details**

6| Add the TestStepValues as mentioned below:

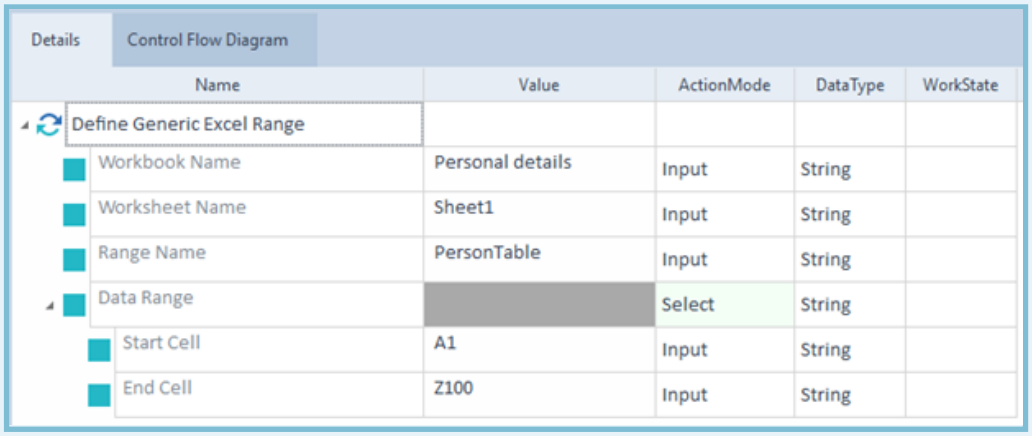
TestStep Name	Name	Value	Action Mode
Open Excel Workbook- Personal details	Workbook Name	Personal details	Input
	Path	{CP[FilePath]}\Personal details_ {B[DateTimeNow]}.xlsx	Input

7| Add another TestStep in the **Precondition** TestStepFolder using the Standard Module **TBox Define Excel Range** and rename it to **Define Generic Excel Range**

8| Add the TestStepValues as mentioned below:

TestStep Name	Name	Value	Action Mode
Define Generic Excel Range	Workbook Name	Personal details	Input
	Worksheet Name	Sheet1	Input
	Range Name	PersonTable	Input
	Data Range		Select
	Start Cell	A1	Input
	End Cell	Z100	Input

For a visual representation of the previous step(s) in Tosca, refer to the image below :



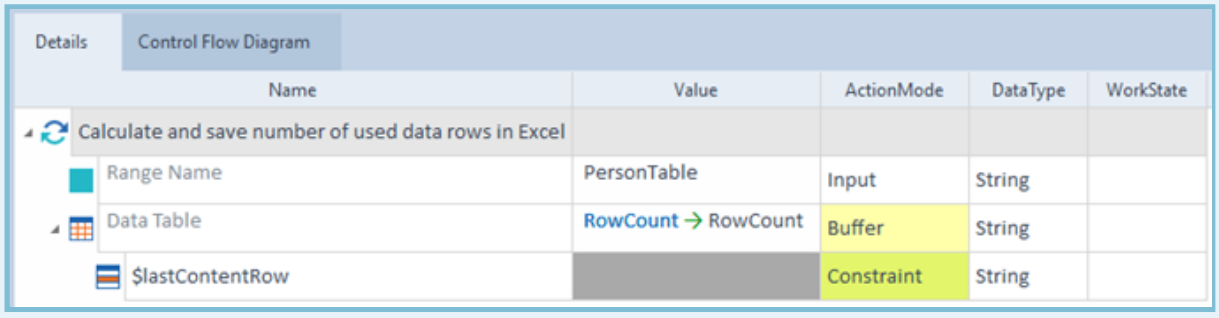
Section II: Precondition – Buffer number of rows

1| Add another TestStep in the **Precondition** TestStepFolder using Standard Module **TBox Excel Range Manipulation** and rename it to **Calculate and save number of used data rows in Excel**

2| Add the TestStepValues as mentioned below:

TestStep Name	Name	Value	Action Mode
Calculate and save number of used data rows in Excel	Range Name	PersonTable	Input
	Data Table	.RowCount->RowCount	Buffer
	$lastContentRow		Constraint

For a visual representation of the previous step(s) in Tosca, refer to the image below :



Section III: Precondition - Define dynamic range

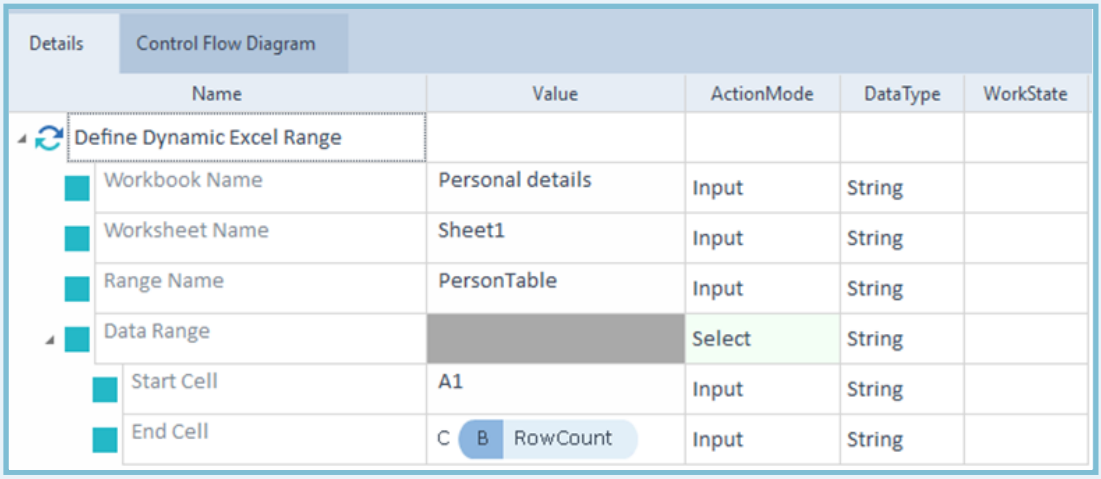
1| Within the TestCase **Verify Data in Excel File**, navigate to the TestStepFolder **Precondition**

2| Add a TestStep using the Standard Module **TBox Define Excel Range**, and rename it to **Define Dynamic Excel Range**

3| Add the TestStepValues as mentioned below:

TestStep Name	Name	Value	Action Mode
Define Dynamic Excel Range	Workbook Name	Personal details	Input
	Worksheet Name	Sheet1	Input
	Range Name	PersonTable	Input
	Data Range		Select
	Start Cell	A1	Input
	End Cell	C{B[RowCount]}	Input

For a visual representation of the previous step(s) in Tosca, refer to the image below:



Section IV: Verify the Excel data against the SUT

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

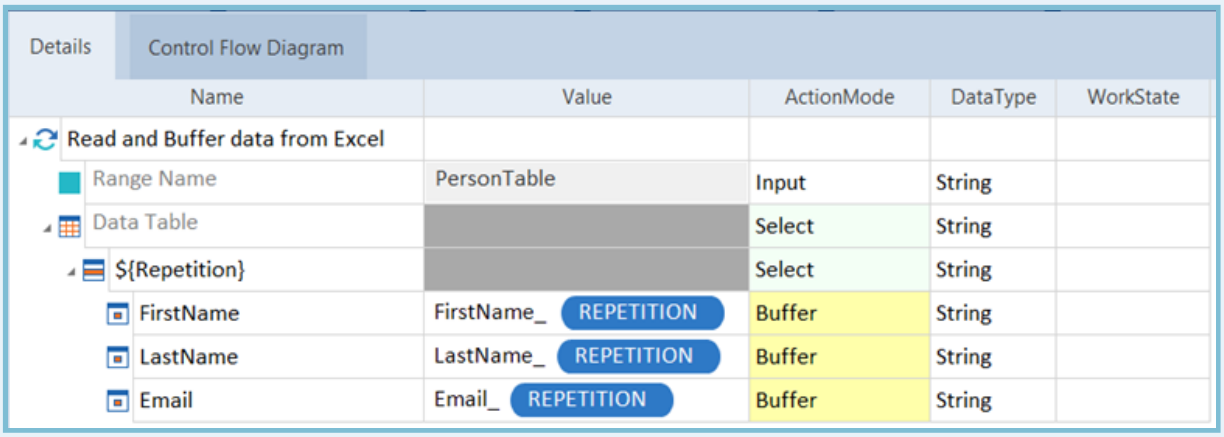
1| In the TestStepFolder **Process** add another TestStepFolder **Verify Excel data against SUT**

2| Within this new TestStepFolder, add a TestStep using the Standard Module **TBox Excel Range Manipulation** and rename it to **Read and Buffer data from Excel**

3| Add the TestStepValues as mentioned below:

TestStep Name	Name	Value	Action Mode
Read and Buffer data from Excel	Range Name	PersonTable	Input
	Data Table		Select
	${Repetition}		Select
	FirstName	FirstName_{Repetition}	Buffer
	LastName	LastName_{Repetition}	Buffer
	Email	Email_{Repetition}	Buffer

For a visual representation of the previous step(s) in Tosca, refer to the image below:



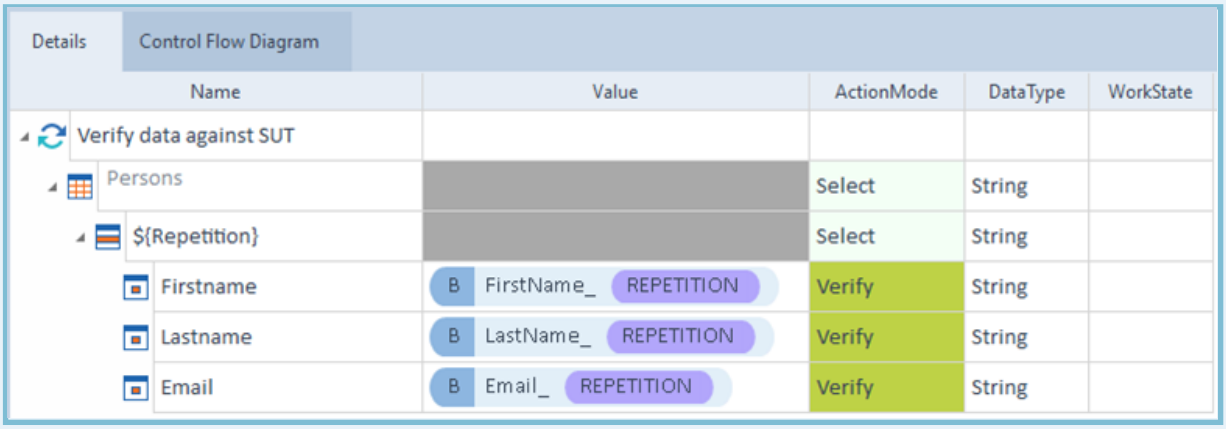
4| Add a TestStep in **Verify Excel data against SUT** TestStepFolder using the Module **Test Automation Obstacle: Persons Table** from the Module Folder **Obstacle: Table Steering** and rename it to **Verify data against SUT**

5| Add the TestStepValues as mentioned below:

5| Add the TestStepValues as mentioned below:

TestStep Name	Name	Value	Action Mode
Verify data against SUT	Persons		Select
	${Repetition}		Select
	Firstname	{B[FirstName_{Repetition}]}	Verify
	Lastname	{B[LastName_{Repetition}]}	Verify
	Email	{B[Email_{Repetition}]}	Verify

For a visual representation of the previous step(s) in Tosca, refer to the image below:



6| Select the TestStepFolder **Verify Excel data against SUT** and open **Properties** pane.

7| Set the **Repetition** Property value to **{MATH[{B[RowCount]}-1]}**

Section V: Save and close the Excel workbook

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

1| Navigate to the TestStepFolder **Postcondition** and add a TestStep using the Standard Module **TBox Close Excel Workbook**

2| Rename it to **Save and Close Excel Workbook - Personal details**

3| Add the TestStepValues as mentioned below:

TestStep Name	Name	Value	Action Mode
Save and Close Excel Workbook - Personal details	Workbook Name	Personal details	Input
	Path	C:\Tosca_Projects\	Input
	Save	True	Input

4| Within the TestStepFolder **Postcondition** add a TestStep using the Standard Module **CloseBrowser**, and rename it to **Close SUT**

5| Add the TestStepValues as mentioned below:

TestStep Name	Name	Value	Action Mode
Close SUT	Title	Tricentis Obstacle Course*	Input

6| Execute the TestCase Verify data in Excel file in ScratchBook and go through/validate the results

Expected result: You will now notice that the TestCase has executed successfully, and the Excel Table data is correct as in the SUT.

+ Observation

You will observe that:

1| Creating Configuration Parameters will allow the reuse of same parameter within multiple TestSteps

2| Adding Repetition Property to the TestStepFolder will execute the TestSteps inside withing loop which can be defined dynamically.

3| We can read and manipulate the data in an excel sheet using **TBox Excel Range Manipulation** Standard Module

4| RowCount property fetches number of rows available in the Table