

# Expressions: Transform Your Data

Exercise to Accompany  
Variables

Lesson 5 Exercise

<b>Introduction</b>	<b>2</b>
Exercise Environment	2
Save Often	2
Additional Resources	2
<b>1 – Configure and Test a Rule Input</b>	<b>3</b>
<b>2 – Answer Key</b>	<b>4</b>
<b>3 – Configure and Test a Local Variable</b>	<b>4</b>
<b>4 – Answer Key</b>	<b>5</b>
<b>5 – Challenge Scenario</b>	<b>5</b>
<b>6 – Answer Key</b>	<b>6</b>

## Notice of Rights

This document was created by Appian Corporation, 7950 Jones Branch Dr, Tysons, Virginia 22102. Copyright 2021 by Appian Corporation. All rights reserved. Information in this document is subject to change. No part of this document may be reproduced or transmitted in any form by any means, without prior written permission of Appian Corporation. For more information on obtaining permission for reprints or excerpts, contact Appian Training at [academyonline@appian.com](mailto:academyonline@appian.com).

## Introduction

This exercise offers you the opportunity to practice the skills you just learned in the previous lesson.

### Exercise Environment

Complete the exercises in this course using an Appian Community Edition environment. You should have obtained one earlier to complete exercises in the previous courses in this learning path. If you do not have an Appian Community Edition environment, get started by visiting: <https://community.appian.com/p/my-learning-journey>.

In this exercise, you will practice writing expression rules, and save them in a practice folder created in the AX application. These are practice exercises that are separate from the remainder of your AX application. If you are working on these exercises independently and do not have the AX application, create an application first before proceeding. Review the Create an Application: First Steps course for instructions on creating an application.

### Save Often

Appian does not automatically save updates, so save your objects frequently.

### Additional Resources

Appian provides a plethora of training resources for novice Appian developers. The following resources are particularly popular with our learners:

- [Academy Online](#) - Appian's online courses provide useful survey courses, step-by-step tutorials, and some additional practice exercises. Explore these resources at your own pace. Survey courses will help you with developing a better grasp of the range of topics you need to learn about. Video and print tutorials will help you with getting hands-on experience with Appian.
- [Appian Documentation](#) - Appian's product documentation will provide you with the overview of key Appian features, newest release information, additional tutorials, and helpful patterns and recipes to implement in your app.
- [Community Discussions for New Users](#) - Join our community of experts to ask questions and find answers from past discussions.

## 1 – Configure and Test a Rule Input

In this exercise, you'll write an expression that uses a rule input and test it to verify the output.

1. Log into your Appian Community Edition environment, and select the **Acme Exercise** application.
2. Create an expression rule object named **AX\_L5Ex1**. Add the description, "Practice for Lesson 5," and store it in the **AX Expressions Practice** folder.
3. Read the following scenario:

Acme Auto HQ loans out vehicles to its branches. When each vehicle is returned, the mileage needs to be compared to the starting mileage.

- If the difference exceeds 50,000 miles, then the message "Run full diagnostics" should be displayed.
  - Otherwise, the message "Return to inventory" should be displayed.
  - Assume that every loaned vehicle was used, so the incoming mileage should be greater than the outgoing mileage.
4. Write an **If()** statement that will perform what is needed in the scenario.

**Hint:** Create rule inputs for the two mileage values.

Refer to the [Appian Docs All Functions](#) page for more information.

5. Enter values, and click **Test Rule** to verify your test output meets the scenario requirements.
6. Save changes.

## 2 – Answer Key

The correct expression is shown in the following screenshot. Your rule input names may be different.

The screenshot shows the Appian Rule Editor interface for a rule named **AX\_L5Ex1**. The rule expression is as follows:

```

1 if(
2   (r1returnedMileage - r1startingMileage) > 50000,
3   "Run full diagnostics",
4   "Return to inventory"
5 )
  
```

The **Test Inputs** section shows two inputs:

Rule Input Name	Expression	Value
startingMileage (Number (Integer))	1	1000
returnedMileage (Number (Integer))	1	60000

The **Test Output** section shows the following output:

Time	Value	Type
1 ms (View Performance)	"Run full diagnostics"	Text

## 3 – Configure and Test a Local Variable

Next, you'll create and configure a local variable.

1. Create a rule expression object named **AX\_L5Ex2**, and add the description, "Practice for Lesson 5." Store it in the **AX Expressions Practice** folder.
2. Create two number (integer) rule inputs named **first** and **second**.
3. Create a local variable named **sum** to store the sum of **first** and **second**.
4. Complete the expression so it returns the string **"The new total is "** and the value of the local variable.
5. Save your changes.

## 4 – Answer Key

The correct expression is shown in the following screenshot.

The screenshot displays the Appian AX\_L5Ex2 interface. On the left, the rule editor shows the following code:

```

1. allocateVariables()
2. localSum: r1first + r1second
3. 'The new total is ' & localSum
4.

```

The right pane shows the 'Test Inputs' section with two inputs:

Name	Type	Value
first (Number (Integer))	Number (Integer)	2
second (Number (Integer))	Number (Integer)	7

Below the test inputs is the 'Local Variables' section:

Name	Type	Value
sum	Number (Integer)	9

The 'Test Output' section shows the result: 'The new total is 9' (Text).

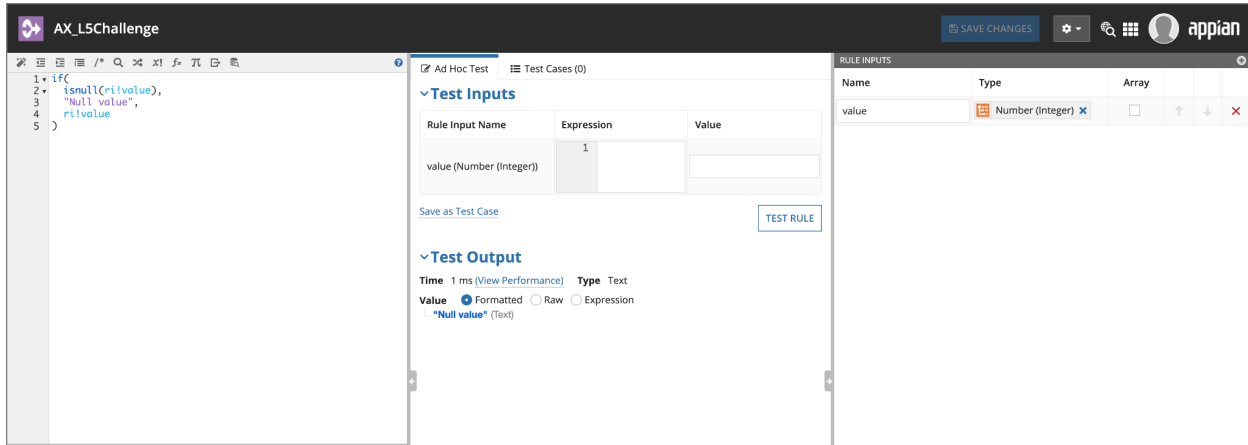
## 5 – Challenge Scenario

Ready to try writing an expression using the `isNull()` function?

1. Create an expression rule object named **AX\_L5Challenge**, and add a description, "Challenge Practice for Lesson 5." Save it in the **AX Expressions Practice** folder.
2. Write an expression that tests a rule input value. If it is null, then the message "Null value" should be displayed. Otherwise, the rule input value should be displayed.
3. Test the expression, first with a null value and then with an actual value.
4. Save changes.

## 6 – Answer Key

The correct expression is shown in the following screenshot:



The screenshot displays the Appian Rule Editor for a rule named "AX\_L5Challenge". The interface is divided into three main sections:

- Left Panel (Code Editor):** Contains a JavaScript-like expression:
 

```
1. if(
2.   isnull(r1.value),
3.   "Null value",
4.   r1.value
5. )
```
- Middle Panel (Test Inputs):**
  - Test Inputs:** A table with columns "Rule Input Name", "Expression", and "Value". It contains one row: "value (Number (Integer))" with the expression "1" and an empty value field.
  - Test Output:** Shows the execution time as "1 ms" and the output value as "Null value" (Text).
- Right Panel (Rule Inputs):** A table with columns "Name", "Type", and "Array". It contains one row: "value" with the type "Number (Integer)".

Buttons for "SAVE CHANGES", "TEST RULE", and "Save as Test Case" are visible at the top right of the editor.