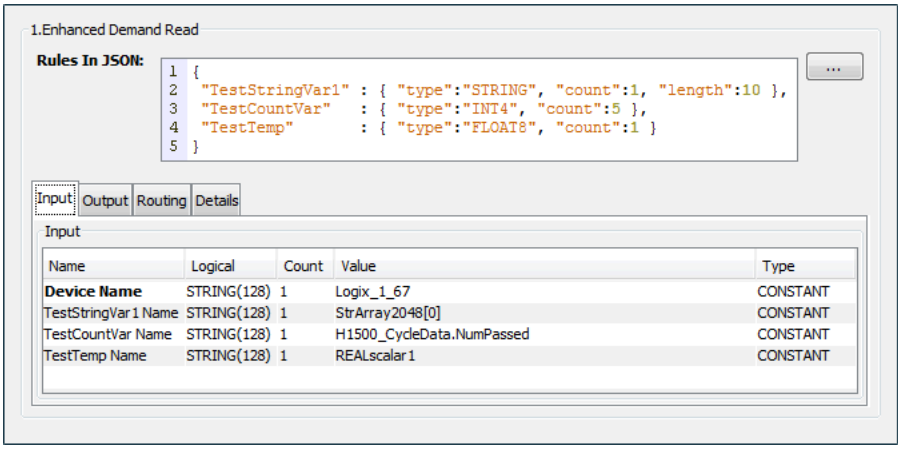
You are here: IoT Gateway User Guide > Projects and triggers > Trigger actions reference > Enhanced Demand Read

Topic updated on September 21, 2023

Enhanced Demand Read

The **Enhanced Demand Read** action reads a group of device variables at that point in a trigger's execution, bypassing the initial trigger buffer read at the start of the trigger's execution. When a trigger starts it execution, all variables referenced by the trigger's actions are read into internal buffers. The variables values are read from or written to this internal buffer when the variables are referenced by any of the trigger's actions. The internal buffer is written to the devices when the trigger completes its execution.

To bypass this internal buffering, the **Enhanced Demand Read** action forces a read of the device variables through the device manager and device driver to the device.



1.Enhanced Demand Read

Rules In JSON:

```
{
 "TestStringVar1" : { "type":"STRING", "count":1, "length":10 },
 "TestCountVar"   : { "type":"INT4", "count":5 },
 "TestTemp"       : { "type":"FLOAT8", "count":1 }
}
```

Input

Name	Logical	Count	Value	Type
Device Name	STRING(128)	1	Logix_1_67	CONSTANT
TestStringVar1 Name	STRING(128)	1	StrArray2048[0]	CONSTANT
TestCountVar Name	STRING(128)	1	H1500_CycleData.NumPassed	CONSTANT
TestTemp Name	STRING(128)	1	REALscalar1	CONSTANT

Example

```
{
    "TestStringVar1": {"type":"STRING", "count":1, "length": 10},
    "TestCountVar": {"type":"INT4", "count":5},
    "TestTemp": {"type":"FLOAT8", "count":1}
}
```

> **Tip**
>
> The Enhanced Demand Read action is part of the Advanced Features package. For more information on obtaining and installing the package, see Packages.

Parameter descriptions

Parameter	Description
Rules in JSON	The description of the device variables to read. Each variable is specified as shown in the example with: • Map variable name - The name of the map variable, which will be added as a row on the Input tab. • type - The data type of the variable to read. This type must match the data type of the variable. The supported types are: BOOL, INT1, INT2, INT4, INT8, UINT1, UINT2, UINT4, UINT8, FLOAT4, FLOAT8, STRING, TIMESTAMP, and BINARY. For the Mitsubishi MESInterface IT product: BIT, BYTE, WORD, DWORD, LWORD, UBYTE, UWORD, UDWORD, ULWORD, REAL, LREAL, STRING, TIMESTAMP and BINARY. • count - The number of variable elements. Specify a 1 for a single (scalar) element or a value greater than 1 for an array. • length - For types STRING and BINARY, the length of the variable element. The multi-line input icon ... can be used to display a larger input area for the JSON variable description.

Input tab

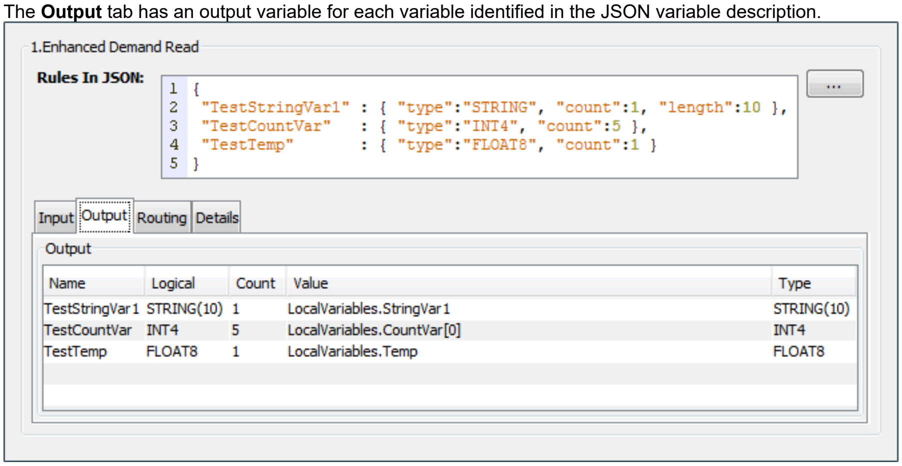
The **Input** tab has the device name parameter and an input map variable for each variable identified in the JSON variable description.

Parameter	Description
Device Name	The name of the device containing the variables whose values you want to read.

Parameter	Description
Input map variables	The map variables identified in the JSON variable description. Each variable in the JSON variable description will be added as a map variable row in the Input tab. In the **Value** column, specify the device variable name. The name can be a constant (entered directly in the parameter), or the name can be dynamically specified in a variable. In the example input tab above, the variables names are specified as constants.

Output tab

The **Output** tab has an output variable for each variable identified in the JSON variable description.



1.Enhanced Demand Read

Rules In JSON:

```
{
 "TestStringVar1" : { "type":"STRING", "count":1, "length":10 },
 "TestCountVar"   : { "type":"INT4", "count":5 },
 "TestTemp"       : { "type":"FLOAT8", "count":1 }
}
```

Output

Name	Logical	Count	Value	Type
TestStringVar1	STRING(10)	1	LocalVariables.StringVar1	STRING(10)
TestCountVar	INT4	5	LocalVariables.CountVar[0]	INT4
TestTemp	FLOAT8	1	LocalVariables.Temp	FLOAT8

Example

```
{
    "TestStringVar1": {"type":"STRING", "count":1, "length": 10},
    "TestCountVar": {"type":"INT4", "count":5},
    "TestTemp": {"type":"FLOAT8", "count":1}
}
```

Parameter	Description
Output variables	Each variable in the JSON variable description will be added as a row in the Output tab. In the **Value** column, specify the variable where the value of the read input variable will be placed. The output variables can be trigger variables (local or static) or device variables. In the example output tab above, the variables are read into trigger local variables.

Enhanced Demand Read considerations

Device variables are normally "read" when they are the source variable in an action. For example, a **Set** action has a source variable and a destination variable.

When a trigger starts it execution, all variables referenced by the trigger's actions are read into internal buffers. The variables' values are read from or written to this internal buffer when the variables are referenced by any of the trigger's actions. The internal buffer is written to the devices when the trigger completes its execution. This process maximizes performance of the trigger execution.

In cases where the device variables need to be read directly from the device at a certain point in the trigger's execution and not from the previously read internal trigger buffers, the **Enhanced Demand Read** action is used. These cases need to take into account the overall application logic in the triggers and in the devices to understand the dynamic interaction of all parts of the application. The **Enhanced Demand Read** action does take additional resources in terms of processing time (compared to for example a **Set** action), since the trigger engine and device driver must communicate down to the physical device and get a response. This includes the communication across the network to the device.

The data type specified for each variable must match the data type of the device variable. If it does not match, that variable will not be successfully read.

The action will fail if there is a problem with the device specified (not defined or not started). The action will succeed if there are individual problems with the variables specified. The variables that had individual problems (variable not defined, data type mismatch, data overflow writing to the output variable, data cast problem writing to the output variable) will not update the corresponding output variable. The variables that did not have a problem will update the corresponding output variables.

Related Topics

About Telit | Contact Us | Legal Notices | Terms of Service | Privacy Policy