WinActor®

- Supplement for JSON Function -

NTT ADVANCED TECHNOLOGY CORPORATION

Introduction

■ This document describes the details of JSON functions in user libraries.

- Related user libraries are as follows.

  98_StructureData/

  - StructureData_JSONFileArraySize.ums7
  - StructureData_JSONFileReadArray.ums7
  - StructureData_JSONSaveVariableToFile.ums7
  - StructureData_JSONVariableNewObject.ums7
  - StructureData_JSONVariableNewArray.ums7
  - StructureData_JSONVariableAppendElement.ums7
  - StructureData_JSONVariableRead.ums7
  - StructureData_JSONVariableArraySize.ums7
  - StructureData_JSONVariableAppendArrayElement.ums7
  - StructureData_JSONVariableReadArray.ums7
  - StructureData_JSONFormatWrite.ums7
  - StructureData_JSONFormatRead.ums7

Trademarks

The names described below and other names of companies and products in this document are trademarks or registered trademarks of their respective companies. The TM, ®, and © marks are omitted in this document.

- WinActor is a registered trademark of NTT ADVANCED TECHNOLOGY CORPORATION.

- Microsoft, Windows*1, Microsoft Edge, Excel, and VBScript*2 are trademarks or registered trademarks of Microsoft Corporation in the United States and other countries.

  *1 The official name of Windows is Microsoft Windows Operating System.
  *2 The official name of VBScript is Microsoft Visual Basic Scripting Edition.

- The names of other companies and products are trademarks or registered trademarks of their respective companies.

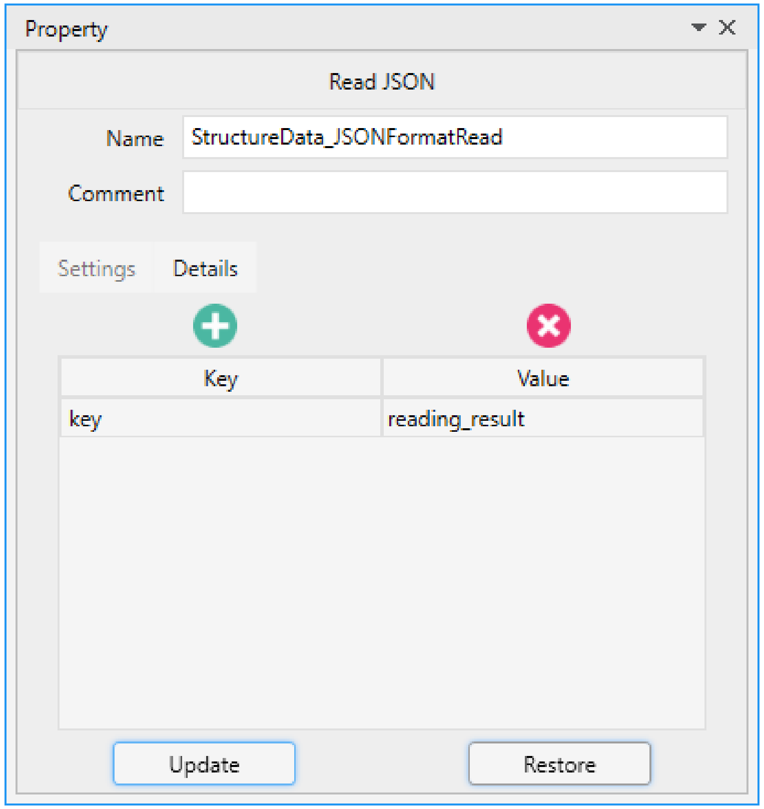
Notes on this document

- The copyright notice "Copyright © 2013-2025 NTT, Inc. & NTT ADVANCED TECHNOLOGY CORPORATION" attached to this manual and the provided software cannot be changed or deleted.
  The copyright of this manual belongs to NTT, Inc. and NTT ADVANCED TECHNOLOGY CORPORATION.
- The descriptions in this manual assume that users understand Windows operations and functions. For information that is not described in this manual, see the documents provided by Microsoft.

StructureData_JSONFormatRead (1/2)

This section describes how each JSON data type is read by "StructureData_JSONFormatRead."

In the "Read JSON" property, specify the source JSON data (file or variable) and a pair of the key string and a variable name to store the corresponding value.
The value type is automatically determined by WinActor.



Name: StructureData_JSONFormatRead
Comment:

Key	Value
key	reading_result

StructureData_JSONFormatRead (2/2)

Table: Stored value examples

No	JSON (*1)	Type (*2)	Stored value (*3)
1	{ key : 123 }	Integer	123
2	{ key : 12.3 }	Decimal	12.3
3	{ key : "123" }	String	"123"
4	{ key : "null" }	String	"null"
5	{ key : "" }	String	""
6	{ key : { sub : 123 } }	Object	{ sub : 123 }
7	{ key : [1, 2, 3] }	Array	[1, 2, 3]
8	{ key : true }	Boolean	true
9	{ key : null }	Null	null

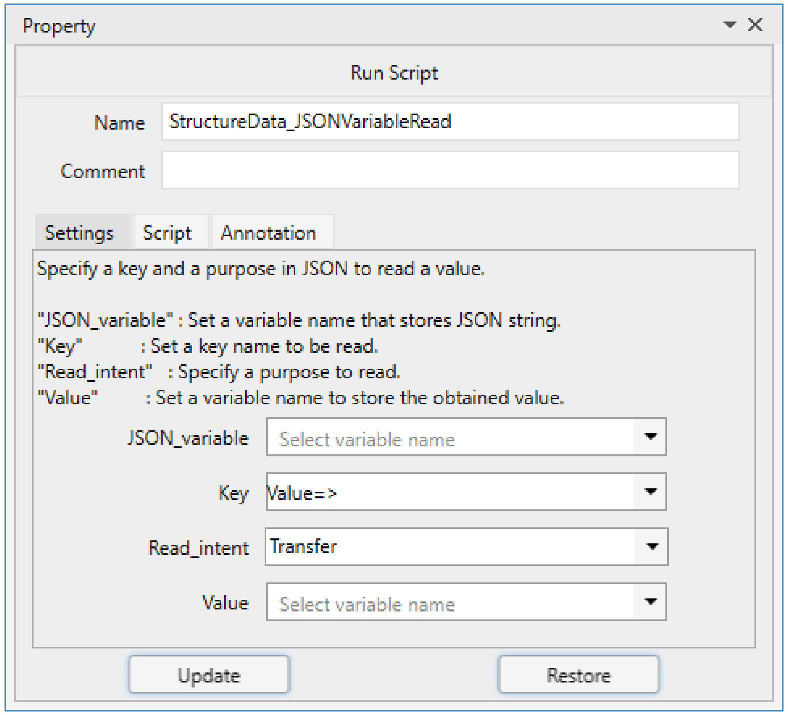
*1 Source JSON to be read.
*2 Value type which WinActor automatically determines. The value type is not stored in a variable.
*3 Value to be stored in a specified variable.

Notes:
The values of No.3 and 4 are quoted with "".
The null value of No.9 is read as null (unquoted four-character string).

StructureData_JSONVariableRead (1/4)

While StructureData_JSONFormatRead is to read a value in JSON, StructureData_JSONVariableRead can be used to transfer a value as-is or read a value type.

In the "StructureData_JSONVariableRead" property window, specify a source JSON for "JSON_variable," a key to read for "Key," what to read for "Read_intent," and a variable to store the result for "Value."



Name: StructureData_JSONVariableRead

Comment:

Specify a key and a purpose in JSON to read a value.

"JSON_variable" : Set a variable name that stores JSON string.
"Key" : Set a key name to be read.
"Read_intent" : Specify a purpose to read.
"Value" : Set a variable name to store the obtained value.

JSON_variable: Select variable name

Key: Value=>

Read_intent: Transfer

Value: Select variable name

StructureData_JSONVariableRead (2/4)

Table: Stored value examples (Read_intent = "Refer_to_value")

No	JSON (*1)	Type (*2)	Stored value (*3)
1	{ key : 123 }	Integer	123
2	{ key : 12.3 }	Decimal	12.3
3	{ key : "123" }	String	"123"
4	{ key : "null" }	String	"null"
5	{ key : "" }	String	""
6	{ key : { sub : 123 } }	Object	{ sub : 123 }
7	{ key : [1, 2, 3] }	Array	[1, 2, 3]
8	{ key : true }	Boolean	true
9	{ key : null }	Null	null

*1 Source JSON to be read.
*2 Value type which WinActor automatically determines. The value type is not stored in a variable.
*3 Value to be stored in a specified variable.

Notes:
The values of No.3 and 4 are quoted with "".
The null value of No.9 is read as null (unquoted four-character string).

In case of Read_intent="Refer_to_value," the stored value is the same as that of "StructureData_JSONFormatRead."

StructureData_JSONVariableRead (3/4)

Table: Stored value examples (Read_intent = "Transfer")

No	JSON (*1)	Type (*2)	Stored value (*3)
1	{ key : 123 }	Integer	123
2	{ key : 12.3 }	Decimal	12.3
3	{ key : "123" }	String	123
4	{ key : "null" }	String	null
5	{ key : "" }	String	
6	{ key : { sub : 123 } }	Object	{ sub : 123 }
7	{ key : [1, 2, 3] }	Array	[1, 2, 3]
8	{ key : true }	Boolean	true
9	{ key : null }	Null	

*1 Source JSON to be read.
*2 Value type which WinActor automatically determines. The value type is not stored in a variable.
*3 Value to be stored in a specified variable.

Notes:
The values of No.3, 4, and 5 are unquoted.
The null value of No.9 is read as an empty string.

In case of Read_intent="Transfer," the stored value is modified so that it will be the same expression in the transferred JSON.

StructureData_JSONVariableRead (4/4)

Table: Stored value examples (Read_intent = "Determine_type")

No	JSON (*1)	Type (*2)	Stored value (*3)
1	{ key : 123 }	Integer	INTEGER
2	{ key : 12.3 }	Decimal	FLOAT
3	{ key : "123" }	String	STRING
4	{ key : "null" }	String	STRING
5	{ key : "" }	String	STRING
6	{ key : { sub : 123 } }	Object	OBJECT
7	{ key : [1, 2, 3] }	Array	ARRAY
8	{ key : true }	Boolean	BOOLEAN
9	{ key : null }	Null	NULL

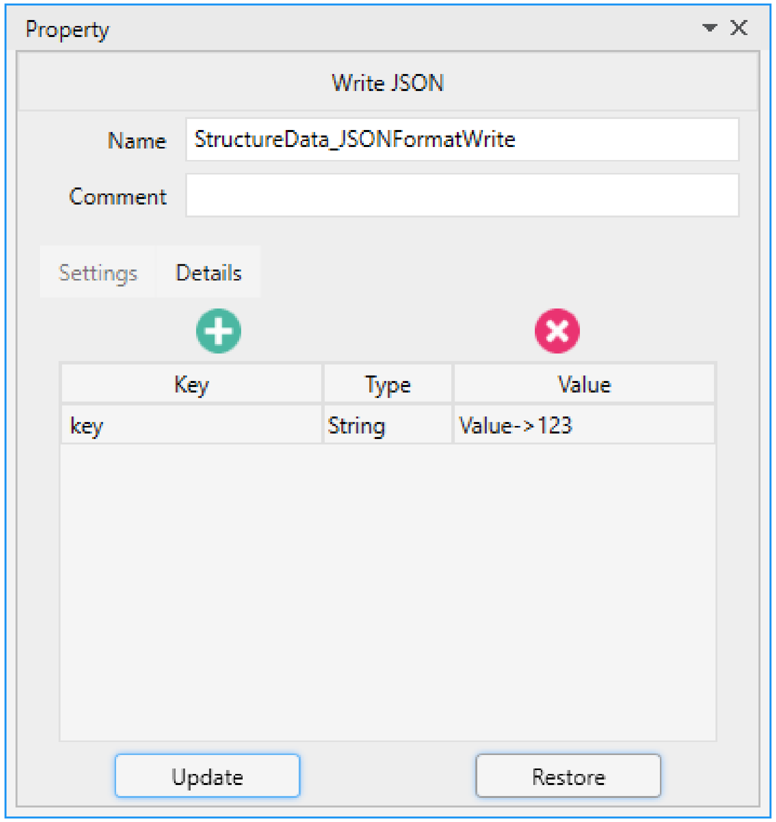
*1 Source JSON to be read.
*2 Value type which WinActor automatically determines.
*3 Value to be stored in a specified variable.

In case of Read_intent="Determine_type," the value type which WinActor automatically determined is stored.

StructureData_JSONFormatWrite (1/7)

This section describes how JSON data is written by "StructureData_JSONFormatWrite."

In the "Write JSON" property, specify JSON data to write for "Key," "Type," and "Value."



Name: StructureData_JSONFormatWrite

Comment:

Key	Type	Value
key	String	Value->123

StructureData_JSONFormatWrite (2/7)

Table: Written JSON data examples (1/6)

No	Type (*1)	Value (*2)	Written JSON data (*3)
1	Integer	123	{ "key" : 123 }
2	Integer	12.3	Error
3	Integer	null	Error
4	Integer		{ "key" : null }
5	Decimal	123	{ "key" : 123.0 }
6	Decimal	12.3	{ "key" : 12.3 }
7	Decimal	null	Error
8	Decimal		{ "key" : null }

*1 Type in the Property window.
*2 Value in the Property window.
*3 JSON data to be written to a specified variable or a file.

Notes:
For No.2, an error occurs as a decimal value is specified for "Integer" type.
For No.5, a decimal value is written when an integer value is specified for "Decimal" type.
For No.3, 4, 7, and 8, use an empty string to write a null value.

StructureData_JSONFormatWrite (3/7)

Table: Written JSON data examples (2/6)

No	Type (*1)	Value (*2)	Written JSON data (*3)
9	String	123	{ "key" : "123" }
10	String	12.3	{ "key" : "12.3" }
11	String	"123"	{ "key" : "¥"123¥"" }
12	String	null	{ "key" : "null" }
13	String		{ "key" : "" }

*1 Type in the Property window.
*2 Value in the Property window.
*3 JSON data to be written to a specified variable or a file.

Notes:
For No.11, the outer "" is added by WinActor. When the original value contains "", the result will be enclosed in double "".
For No.12 and 13, a null value cannot be written as "String" type. When "null" is specified for Value, it will be treated as a string "null." When an empty string is specified, it will be treated as an empty string.
Use "Null" type to write a null value.

StructureData_JSONFormatWrite (4/7)

Table: Written JSON data examples (3/6)

No	Type (*1)	Value (*2)	Written JSON data (*3)

14	Object	{ }	{
			  "key" : { }
			}

15	Object	{ sub : 123 }	{
			  "key" : {
			    "sub" : 123
			  }
			}

16	Object	123	Error

17	Object	null	{
			  "key" : null
			}

18	Object		{
			  "key" : null
			}

*1 Type in the Property window.
*2 Value in the Property window.
*3 JSON data to be written to a specified variable or a file.

Notes:
For No.16, use { } to write an object.
For No.17 and 18, use an empty string or "null" to write a null value. An empty string is recommended in accordance with the way used for other types.

StructureData_JSONFormatWrite (5/7)

Table: Written JSON data examples (4/6)

No	Type (*1)	Value (*2)	Written JSON data (*3)

19	Array	[ ]	{
			  "key" : [ ]
			}

20	Array	[1,2,3]	{
			  "key" : [1, 2, 3]
			}

21	Array	123	Error

22	Array	null	{
			  "key" : null
			}

23	Array		{
			  "key" : null
			}

*1 Type in the Property window.
*2 Value in the Property window.
*3 JSON data to be written to a specified variable or a file.

Notes:
For No.21, use [ ] to write an array.
For No.22 and 23, use an empty string or "null" to write a null value. An empty string is recommended in accordance with the way used for other types.

StructureData_JSONFormatWrite (6/7)

Table: Written JSON data examples (5/6)

No	Type (*1)	Value (*2)	Written JSON data (*3)
24	Boolean	true	{ "key" : true }
25	Boolean	false	{ "key" : false }
26	Boolean	123	{ "key" : false }
27	Boolean	0	{ "key" : false }
28	Boolean	-1	{ "key" : false }
29	Boolean	abc	{ "key" : false }
30	Boolean	null	{ "key" : false }
31	Boolean		{ "key" : null }

*1 Type in the Property window.
*2 Value in the Property window.
*3 JSON data to be written to a specified variable or a file.

Notes:
For No.26, 27, 28, and 29, no error occurs and "false" is written. To avoid unintentional output, specify either "true" or "false" to write a Boolean value.
For No.30 and 31, use an empty string to write a null value.

StructureData_JSONFormatWrite (7/7)

Table: Written JSON data examples (6/6)

No	Type (*1)	Value (*2)	Written JSON data (*3)
32	Null	123	{ "key" : "123" }
33	Null	12.3	{ "key" : "12.3" }
34	Null	"123"	{ "key" : "¥"123¥"" }
35	Null	null	{ "key" : "null" }
36	Null		{ "key" : null }

*1 Type in the Property window.
*2 Value in the Property window.
*3 JSON data to be written to a specified variable or a file.

Notes:
For No.32, 33, and 34, "Null" type basically has the same output as "String" type.
For No.35 and 36, the output for an empty string is different from that of "String" type.
Use an empty string to write a null value. When "null" is specified, it will be treated as a string "null."

WinActor®

Supplement for JSON Function

NTT ADVANCED TECHNOLOGY CORPORATION

WA7-C-20250603