JSON provider

HTTP / HTTPS-compatible

Version 1.0.0

User’s guide

December 14, 2017

Remarks:

[Revision history]

Version	Date	Content
1.0.0	2016-7-29	First edition.
	2017-12-14	Typo correction

Content

1. Introduction.. 4
2. Outline of this provider 5
2.1. Installation...5
2.2. Outpine 5
2.3. Method and Properties...6
2.3.1. CaoWorkspace::AddController method 6
2.3.2. CaoController::AddFile method 7
3. Command reference... 8
3.1. File class..8
3.1.1. CaoFile::AddVariable method..8
3.1.2. CaoFile::AddFile method..9
3.1.3. CaoFile::Execute method 11
3.1.3.1. CaoFile::Execute("SetReqeustHeader") command...............12
3.1.3.2. CaoFile::Execute("SetTimeout") command.........................12
3.1.3.3. CaoFile::Execute("CreateAndPost") command....................12
3.1.3.4. CaoFile::Execute("CreateJSON") command.........................13
3.1.3.5. CaoFile::Execute("Post") command....................................13
4. CAO structure example 14
Appendix A. 2-clause BSD license For picojson 15

1. Introduction

JSON provider transmits data to a server in JSON format.

This provider enables to transmit JSON data to any servers.

Note that the web server's API needs to be created based on the REST principle.

This document describes the overview of JSON provider and implemented CAO interface (function specifications)

2. Outline of this provider

2.1. Installation

JSON provider module consists of the following DLL. You do not need to install it manually if it is installed by ORiN2 SDK. To install manually, use information of Table 2-1.

Table 2-1 JSON provider

File name	CaoProvJSON.dll
ProgID	CaoProv.JSON
Registration	regsvr32 CaoProvJSON.dll
Deregistration	regsvr32 /u CaoProvJSON.dll

2.2. Outpine

JSON provider converts registered data to JSON format, and then transmits the data to a web server using HTTP/HTTPS protocol.

2.3. Method and Properties

2.3.1. CaoWorkspace::AddController method

Syntax AddController (<bstrCtrlName:BSTR>, <bstrProvName:BSTR>, <bstrPcName:BSTR>, [<bstrOption:BSTR>])

< bstrCtrlName >	:	[in] Controller name
< bstrProvName >	:	[in] Provider name. Fixed to ="CaoProv.JSON"
<bstrPcName>	:	[in] Computer name where provider runs (not used).
<bstrOption>	:	[in] Option strings

- Web server’s base URI (BaseURI)
- Whether to allow invalid SSL certificates (AllowInvalidCert)

The following shows details of option strings (bstrOption).

Table 2-2 Option strings of CaoWorkspace::AddController method

Item	Description	Required	Remarks
BaseURI	URI strings of web server	✓	
ArrowInvalidCert	Invalid SSL certificate authorization "YES" : Allow "NO" : Not allow	-	Available only for Https. If there is no entry, it is regarded as "NO". Select "YES" only when the connection destination is reliable such as in-house web server.

Example As an example of use, "123.168.1.3:443" shows a code that permits and connects with an invalid SSL certificate.

```
Private caoEng As CaoEngine          ' Engine Object
Private caoWs As CaoWorkspace        ' WorkSpace Object
Private caoCtrl As CaoController     ' Controller Object

Set caoEng = New CaoEngine
Set caoWS = caoEng.CaoWorkspaces.Item(0)
Set caoCtrl = CaoWS.AddController("JSON", "CaoProv.JSON", "",
"BaseURI=https://192.168.1.3:443,AllowInvalidCert=YES")
```

2.3.2. CaoController::AddFile method

Syntax AddFile (<bstrName:BSTR>, <bstrOption:BSTR>)

You can set the endpoint URI of the destination Web server resource in bstrOption.

< bstrName >	:	[in] File name
<bstrOption>	:	[in] Option strings

- End point URI of web server resource

Table 2-3 Option strings of CaoController::AddFile method

Item	Description	Required	Remarks
EndpointURI	Web server resource endpoint URI string	-	If no endpoint is specified, POST is sent to the base URI specified by the CaoController :: AddFile method.

Example As a usage example, in the case of 2.3.1 configuration example, the code to POST to https://192.168.1.3:443/users/ is shown

```
Private m_caoFile as caoFile          ' File Object
set m_caoFile = caoCtrl.AddFile("point", " EndpointURI=users/")
```

3. Command reference

3.1. File class

3.1.1. CaoFile::AddVariable method

Executing CaoFile::AddVariable method adds a property to a file. An argument (bstrName) of this method is used as an object key at the timing of JSON output.

Syntax AddVariable(<bstrName:BSTR>, <bstrOption:BSTR>)

< bstrName >	:	[in] Variable name
< bstrOption >	:	[in] Option strings (not used)

To set a value in a JSON object, set a desired value in the variable added.
The following table shows a list of available data types.

Table 3-1 Data types available for variables

Type	JSON type	Data type to set
Null value	null	VT_NULL
Array	Boolean	VT_BOOL
Integer	Number	VT_I4
Floating point	Number	VT_R8
String	String	VT_BSTR
Array (Boolean)	Boolean[]	VT_ARRAY | VT_BOOL
Array (Integer)	Number[]	VT_ARRAY | VT_I4
Array (Floating point)	Number[]	VT_ARRAY | VT_R8
Array (String)	String[]	VT_ARRAY | VT_BSTR

Example To set values to CaoVariable

```
caoVar = m_caoFile.AddVarialbe("name", "")
caoVar.Value = "File1"
```

3.1.2. CaoFile::AddFile method

Add a File class (child File class) to CaoFile class.

As an option string, you need to specify a key (BSTR type) that connects a child File class and a parent File class where the child File class is added.

Also, specify how this child File class is maintained in the parent File class.

The setting method is same as CaoController::AddFile method.

Syntax AddFile(<bstrName:BSTR>, <bstrOption:BSTR>)

< bstrName >	:	[in] File name
< bstrOption >	:	[in] Option strings

- A key that connects this child File to a parent File (Key).
- How this child file is maintained in a parent File (Type).

The following shows details of option strings (bstrOption).

Table 3-2 Option strings of CaoFile::AddFile method

Items	Description	Required	Remarks
Key	Strings of the key	✓	
Type	"Array" : maintain as an array "Object" : maintain as an object	-	If there is no entry, it is regarded as "Object".

Example

The following sample program shows how to specify option strings and JSON that is output by this program

Table 3-3 Example of AddFile option specification and JSON to be output

Command example	JSON to be output
caoFile1 = caoContoller.addFile("Parent", "EndpointURI=xxxx")	{
parentName = caoFile1.addVariable("name", "")	"name": "Taro",
parentName.Value = "Taro"	"age": 34,
parentAge = caoFile1.addVariable("age", "")	"child": {
parentage.Value = 34	"name: "Hanako"
	}
caoFile2 = caoFile1.AddFile("child", Key=child,Type=Object")	}
childName = caoFile2.addVariable("name": "")	
childName.Value = "Hanako"	
caoFile1 = caoContoller.addFile("Parent", "EndpointURI=xxxx")	{
parentName = caoFile1.addVariable("name", "")	"name": "Jiro",
parentName.Value = "Jiro"	"children": [{
	"name: "Yoshiko"
caoChildFile1 = caoFile1.AddFile("child1", "Key=children,Type=Array")	},{
childName1 = caoChildFile1.addVariable("name", "")	"name: "Masako"
childName1.Value = "Yoshiko"	}]
	}
caoChildFile2 = caoFile.AddFile("child2", Key=children,Type=Array")	
childName2 = caoChildFile2.addVariable("name": "")	
childName2.Value = "Masako"	

3.1.3. CaoFile::Execute method

Execute a provider-original expansion command that belongs to CaoFile class.

This method is available only for the File class that is added by CaoController::AddFile method.

This method fails if it is executed in the File class that is added by CaoFile::AddFile method.

Syntax Execute(<bstrCmd:BSTR>, <vntParam:VARIANT>)

< bstrCmd >	:	[in] Command name
< vntParam >	:	[in] Parameter

The following table lists the available commands.

Table 3-4 CaoFile::Execute command list

Command	Function	Page
SetRequestHeader	Add an HTTP request header.	12
CreateAndPost	Post data to a server in JSON format.	12
CreateJSON	Output data as JSON string.	13
Post	Post JSON string to a web server.	13

3.1.3.1. CaoFile::Execute("SetReqeustHeader") command

Add a request header that is used to send a request to a web server.

Argument must be specified in array, such as {request header key},{request header value}.

If the same request key is specified twice or more, the older request header value will be overwritten.

You do not need to set Content-Type and Content-Length because the provider set them automatically.

Setting example : X-MyCompanyToken,0123456789ABCD

Syntax SetReqeustHeader (<Data>)

< Data >	:	[in] Request header key and the value to be added (VT_BSTR | VT_ARRAY)

Example

```
result = m_caoFile.Execute("SetRequestHeader",
                           "X-MyCompanyToken,0123456789ABCD")
```

3.1.3.2. CaoFile::Execute("SetTimeout") command

Set the timeout when a request is sent to a web server. Unit is millisecond. Enter a value larger than 0.

If this entry is omitted, 5000 (5 seconds) is set automatically.

Syntax SetRequestHeader (<Data>)

< Data >	:	[in] Timeout time of http request (millisecond) (VT_I4)

Example

```
result = m_caoFile.Execute("SetTimeout", 1000)
```

3.1.3.3. CaoFile::Execute("CreateAndPost") command

Send data that is kept in a File to a web server in JSON format by POST method.

Return value is the body part of the response from the server where the File data is posted.

Syntax CreateAndPost ()

Example

```
result = m_caoFile.Execute("CreateAndPost")
```

3.1.3.4. CaoFile::Execute("CreateJSON") command

Return data that is kept in a File in JSON String format.

Syntax CreateJSON ()

Example

```
result = m_caoFile.Execute("CreateAndPost")
```

3.1.3.5. CaoFile::Execute("Post") command

Post any strings to a web server.

If the string is not JSON format, Content-Type of Request-Header can be overwritten with any values based on 3.1.3.1.

Return value is the body part of the response from the server where the File data is posted.

Syntax Post (<Data>)

< Data >	:	[in] String to be sent (VT_BSTR)

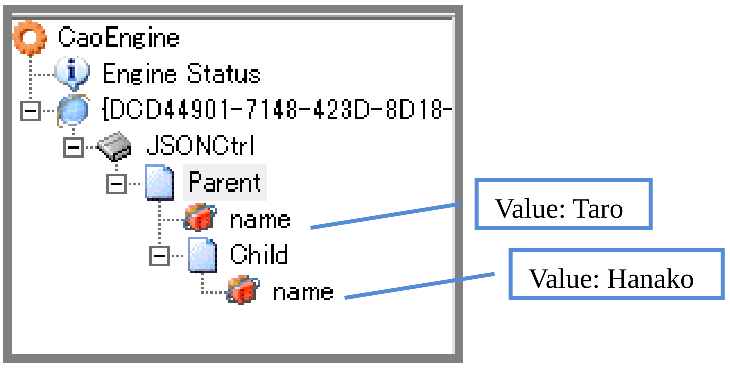
Example

```
result = m_caoFile.Execute("Post", "{""name"":""Taro""}")
```

4. CAO structure example

The following shows the CAO structure when JSON objects are expressed with this provider.

JSON objects	CAO structure

`{`
`  "name": "Taro",`
`  "child": {`
`     "name: "Hanako"`
`  }`
`}`
	 CaoEngine
	Engine Status
	{DCD44901-7148-423D-8D18-
	JSONCtrl
	Parent
	name
	Child
	name
	Value: Taro
	Value: Hanako

Appendix A. 2-clause BSD license For picojson

【picojson】

https://github.com/kazuho/picojson