CaoScript

User's guide

Version 1.5.8

May 13, 2010

[Remarks]

[Revision history]

Date	Versions	Description
2006-03-09	1.0	First edition.
2006-09-02	1.1	Added OnMessage Event reception function of CaoController class.
2006-12-04	1.1.1	Added process priority change function.
2007-01-12	1.2.0	Added CaoWorkspace sharing object ('gCao').
2007-03-30	1.2.1	Improved usability. Expanded RunScript functions. Added Pnl and Mgr objects.
2007-04-17	1.2.2	Added OnErrorGoto method into App object.
2007-06-13	1.3.0	Added Break Point by Vars, cycle start, and Error object.
2007-08-20	1.3.1	Added RCW of CaoScript. Improved Manager usability.
2007-11-15	1.3.2	Added Mgr.Count and App.Filename properties. Improved Schedule function.
2007-12-25	1.3.3	Added Ext and Dat objects. Moved App object functions to new objects.
2008-01-08	1.4.0	Opened Mgr object as Automation interface.
2008-01-19	1.4.1	Added App.Path property. Changed 'Gcao' to 'Cao'.
2008-03-11	1.4.2	Added Task tray registration function of CaoScriptMan.
2008-03-28	1.4.3	Added App.ActivateWindow and App.SendKeyCode methods.
2008-04-18	1.4.4	Added Var.DateTime, Var.Changed, and Var.Bin8/16/32 properties.
2008-04-21	1.4.5	Added local memory area to Vars area. Added Vars.ItemValue property.
2008-04-25	1.4.6	Added Csv object and macro definition function.
2008-06-23	1.4.7	Changed specifications of App.Priority property.
2008-07-02	1.4.8	Added Vars.Help save function.
2008-07-03	1.4.9	Added Execute("GetScriptStarted").
2008-07-11	1.5.0	Enabled On Error statement.
2008-08-27	1.5.1	Added Appendix "VBScript reference".
2008-11-12	1.5.2	Added Csv.LoadFile/.SaveFile methods. Added Vars thumbnail display.
2009-05-19	1.5.3	Added local variable monitor function. Added user sharing object function.
2009-06-12	1.5.4	Added Vars event function.
2009-07-20	1.5.5	Added BeepEx statement.
2009-11-02	1.5.6	Added method of creating user common object
2009-11-06	1.5.7	Added Vars.Macro function.
2010-05-13	1.5.8	Speed-up of Vars event function (Specification change)
2010-10-22	1.5.9	Added Step-forward/Step-stop without displaying StartScript/StopScript

Contents

1. Outline

The CaoScript tool is one of the simplest CAO program development environments. With this CaoScript tool, users can create a simple CAO application program without installing integrated development environments such as Microsoft Visual Basic or Visual C++.

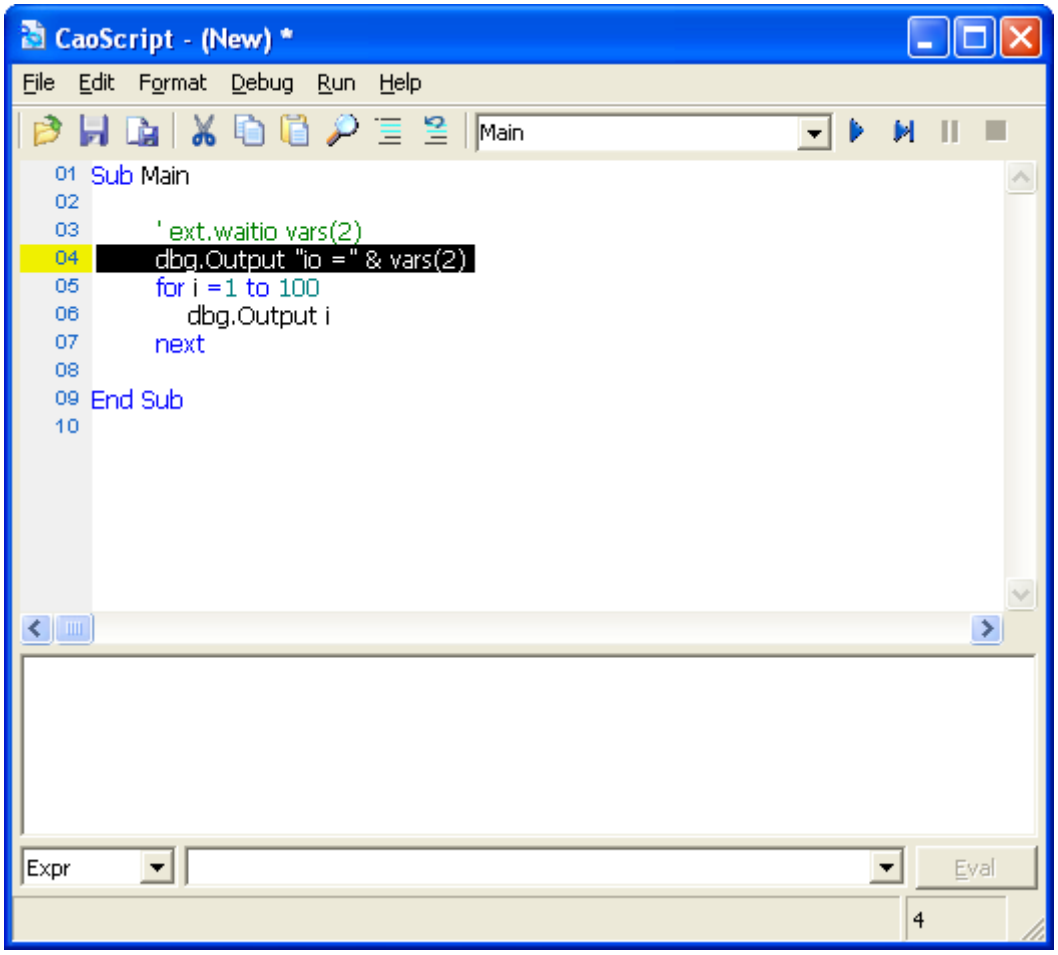


Figure1-1 CaoScript window

2. CaoScript operation

2.1. CaoScript window layout

CaoScript window consists of the script edit window, log window and debug window.

Users write programs in the Script edit window with CaoScript language and execute it. Log window displays messages when any error occurs while program is running or when a program runs a log-output command.

Debug window is used to evaluate global variable's expressions and functions, and to monitor local variables.

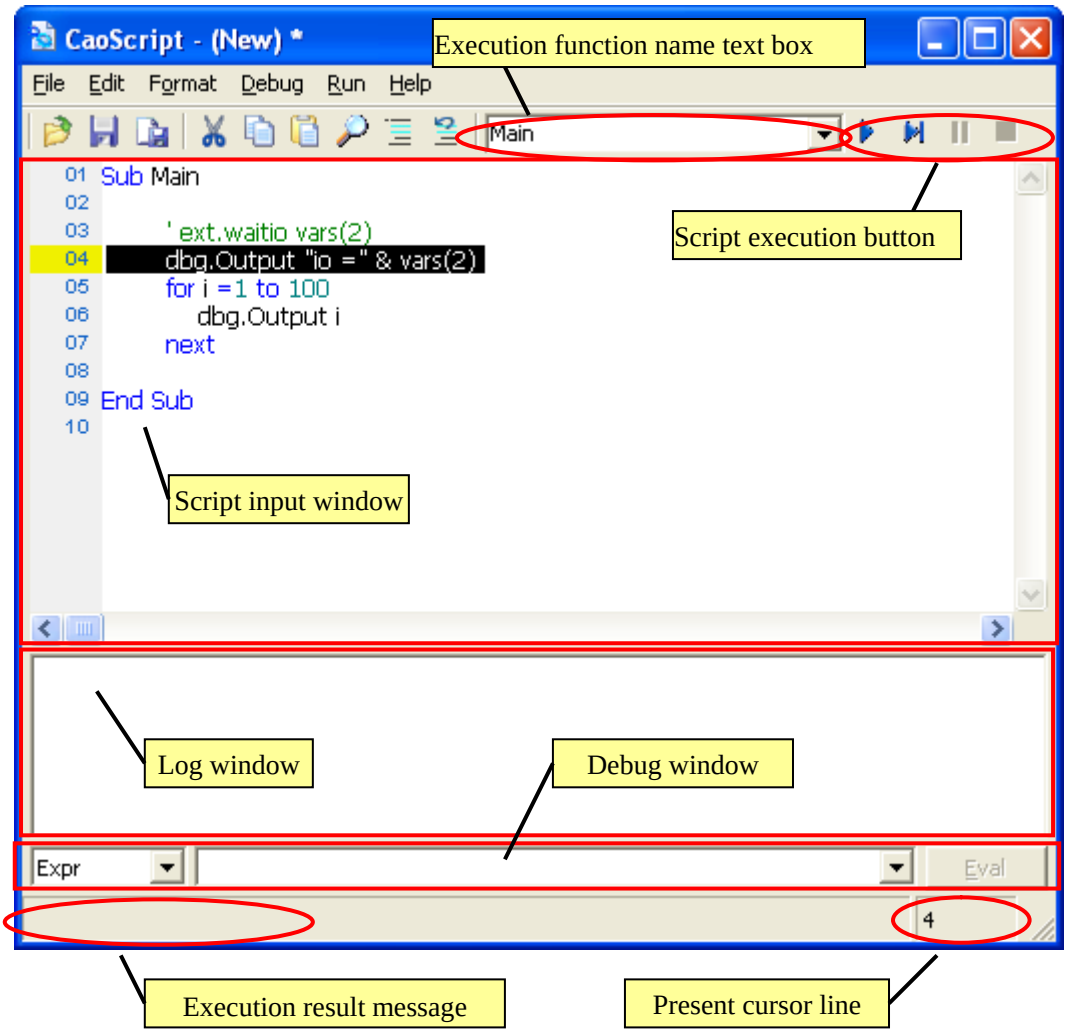


Figure2-1 CaoScript window layout

2.2. CaoScript function

CaoScript provides editing support functions, such as text search, replacement, line jump, and line marker.

また，実行する関数をスタートアップに指定することにより実行を開始し，実行中には Breakpoint 機能を使

用することで CaoScript を一時停止し, ステップ実行するなどのデバッグ実行することもできます・・・・(この部分、日本語の構成から見直すこと)

【こんなことが言いたいのか？】

その他にも、実行対象の関数をスタートアップとして指定する、「Breakpoint」機能を用いて、実行中のCaoScriptを一時停止する、CaoScriptを1ステップずつ実行する、といった機能があり、これらを組み合わせることでデバッグ作業が容易になります。

★これ以降、自動翻訳の英語原稿★edit functions like string search / replace, line jump and line marker. CaoScript starts its execution by specifying the executed function as a startup function. For debugging, functions like breakpoint for stopping CaoScript execution, or step execution are available.

2.2.1. CaoScript execution and stop

2.2.1.1. Execute

Once you program with CaoScript language, execute the program. From the menu bar, point to [Run], and then select [開始]. To run a program one step for each execution, from the menu bar, point to [Run], and then select [ステップオーバー]. You can select execution menu such as [Run] or [ステップオーバー] from the tool bar as well.

Operations described above are effective to the function displayed in the [Execution function name text box] which is introduced in Figure 2-1 CaoScript window layout. If incorrect function name is entered, an error dialog appears. If you run a CaoScript with the textbox empty, a program written out of the function scope will be executed. You can neither edit programs nor start-up while CaoScript is running.

2.2.1.2. Stop

To stop CaoScript execution, select [Run] – [Stop] menu. To pause CaoScript execution, select [Run] – [中断] menu.

2.2.1.3. Cycle start / Cycle stop

To run program cyclically, select [Run] – [サイクル起動] menu. If you run a program with Cycle start, the program restart automatically after the execution end. The program name to be repeated is displayed in the Execution function name text box.

2.2.2. CaoScript debug

CaoScript provides various functions for debugging, such as breakpoint, evaluation function on debug window, variable monitoring function, debug object log output. This subsection explains breakpoint function, debug window evaluation function and variable watch function.

2.2.2.1. Breakpoint function

Breakpoint function is to suspend program execution at the specified program step. This function enables

you to run a program to a certain step and then run the remaining steps with Step Over to examine the operation. It is very practical for debugging.

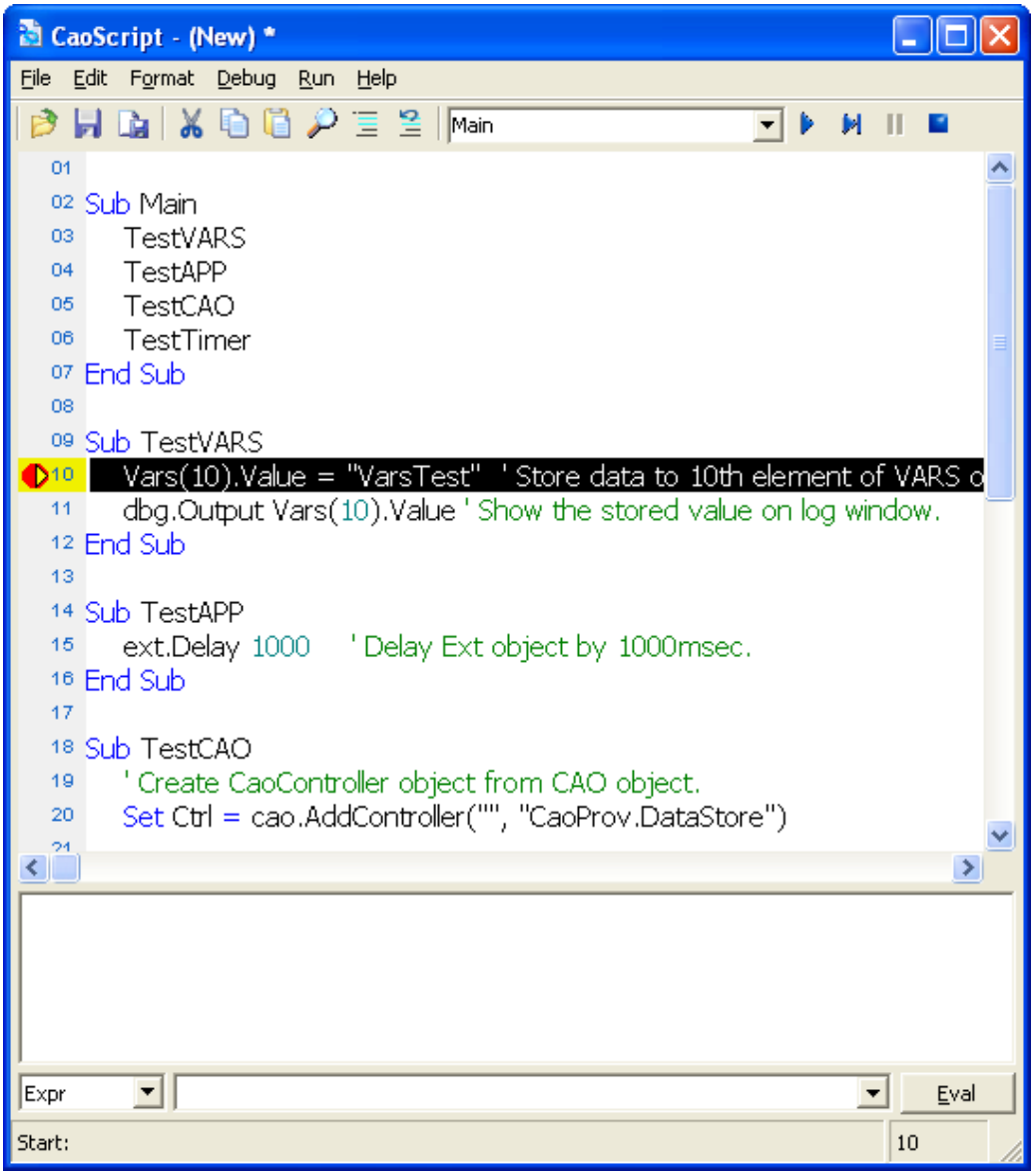


Figure 2-2 Program suspended status at breakpoint

The first step to use breakpoint function is to set breakpoint. There are two ways to set breakpoints.

The first way is, on the menu bar, point to [Debug], and then select [Breakpoint…]. Once a Breakpoint window appears, enter desired step (line) number. Users can set the breakpoint line (step) to pause program execution. This way is convenient to set several breakpoints.

When several breakpoints are set, the program pauses at the first breakpoint, and when the program

execution is resumed, the program pauses again at the next breakpoint.

The second way is to set a breakpoint one by one. Select a step (line) on the script input window, and then, from the menu bar, point to [Debug], and then select [ブレークポイントの設定/解除]. You can set/reset a breakpoint by pressing [F9] key on your keyboard as well.

To reset a breakpoint, on the menu bar, point to [Debug], and then select [Breakpoint…]. Once a Breakpoint window appears, select a step (line) number to reset a breakpoint and then click [解除], or click [すべて解除] if you reset all breakpoints. You can also reset a breakpoint by selecting a step (line) on the script input window and click [Debug]-[ブレークポイントの設定/解除], or press [F9] key on your keyboard.

2.2.2.2. Debug window evaluation function and variable watch function

There are other useful functions for debugging; global variable expression evaluation, expression evaluation, and the variable watch. You can use these functions on the debug window described in Fig.2-1 and the functions are available only when program is running or paused.

Evaluation of global variable expression is available to global variables in CaoScript.

For example, assuming that a CaoScript declares global variable "A", and value of "123" is substituted in a point in the program. In the debug window, select "Expr", enter "A=123", and then click "Eval" button. "True" is displayed in the log window if the substitution is done. On the other hand, select "Expr", enter "A" , and then click "Eval" button will display the current value of "A" in the log window.

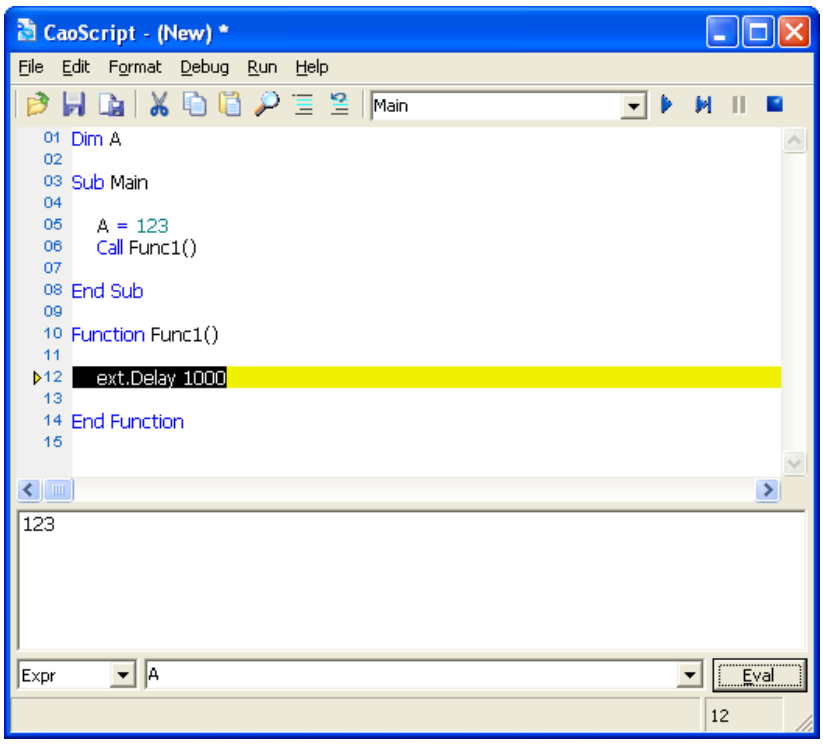


Figure2-3 Global value evaluation

★実際の動きがよくわからなかったので、解説してもらってから英訳する★これ以降は自動翻訳の英語★
For "Expression", a value returning function can by evaluated by specifying the function name. In this case, a function return value is shown in the log window by pressing "Eval" button, if the function returns a value. "Function name = Value" also returns evaluation result of "True" or "False".

Variable watch is to monitor local variable function value declared in a function, by registering it for watching.

For example, assuming that a local variable "A" is declared in the "Main" function. To register the variable for watching, on the debug window, select "Variable name", enter "A", and then click "Watch" button. If the variable is registered successfully, the log window displays "Watch on: A". Variable watch fails unless the target variable is registered correctly. Unlike the global variable expression evaluation, variable watch is affected by the scope of variable. Please note that the variable watch can monitor a local variable that is in the currently running function only.

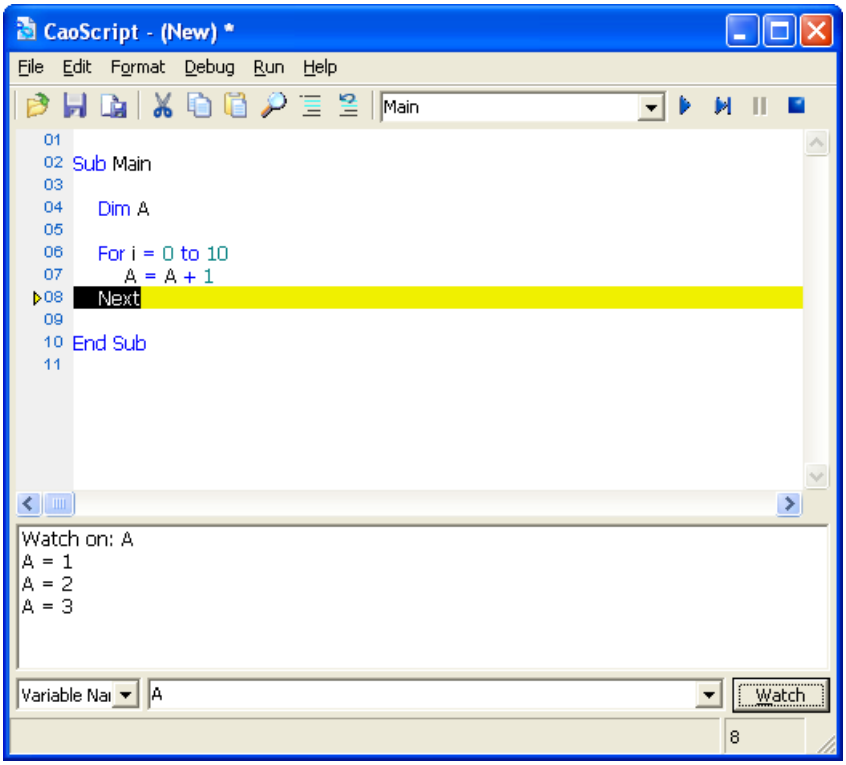


Figure2-4 Local variable watch display

For both "Expression" and "Variable Name", please notice that an error message appears if the variable does not exist on the program or evaluation equation syntax is wrong. Once the error message appears, the currently running program will stop.

★やってみたけど、記述されていない変数名を入れて Watch ON にしても、エラー出なかった。いいの？★

3. CAO script language

CAO script language is a Microsoft Script based-language, with 6 of embedded objects.

CAO script = VBScript + embedded objects (6 objects)

Since the grammar and functions of CAO script are exactly same as VBScript, which is the base language, it is easy to transport programs from CaoScript to Visual Basic. For about language reference of VBScript, refer to the following URL.

https://msdn.microsoft.com/library/d1wf56tt.aspx ★このリンク先でよいか確認してもらう★

3.1. Embedded object

CaoScript has two types of embedded objects; four global objects (Table3-1 Embedded global object list), and two local objects (Table3-2). The names of the global objects are the same as object names, and the system creates and registers only one object at startup. It is impossible to create global objects with programming. A local object is created by executing a function of a global object.

Table3-1 Embedded global object list

Object name	Description
CAO	CaoWorkspace object.
VARS	Collection object of VAR object.
DBG	Debug object.
APP	Application object.
DAT	Data conversion object.
EXT	Object for function expansion

Table3-2 Embedded local object list

Object	Description
CAOMESS	CaoMessage event sink object. Attach function of APP object creates this object.
VAR	Backup variable object. This object is acquired by Item property of VARS object. (Item property can be omitted because Item property is a default property of the collection object.) Values stored in VAR variable is permanently saved. VAR variable can store various types of data including array (one dimension only). The default array size is 100 (index range: 1 to 100). The array size can be changed by modifying [Parameter] of the DataStore provider with CaoConfig. For instance,

	to change the size to 500, enter "ItemMax=500" (see Figure3-1). [Attention] OnMessage event procedure interrupts the main process execution at any timing. Therefore, OnMessage procedure may accesses a VAR object while the interrupted main process is accessing the same VAR object, and this may result in an unexpected variable value. To avoid this problem, do not create programs where main process and OnMessage procedure may access the same object at the same time.
ERROR	Error object. The object can be acquired with Error property of APP object. When the process is trapped by OnErrorGoto, the error information is stored in this object, not in Err object of VBScript. Error information is a copy of the Err object of VBScript, and the Err object of VBScript is cleared.

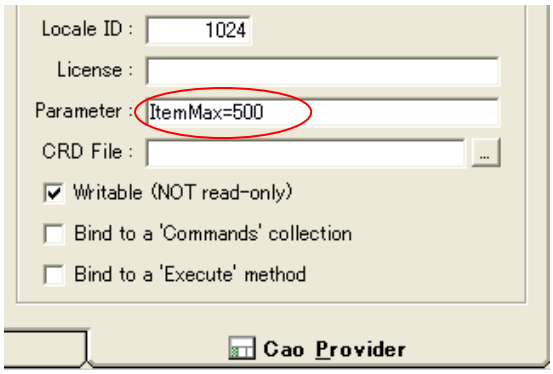


Figure3-1 CaoConfig VARS array number setting

The following tables show the function overview of the objects. For information of CAO object, refer to CAO specifications since these objects are the same as default CaoWorkspace object. The following tables explain other embedded objects.

Table3-3 VARS object member list

Member	Description
Property Get Item (Index As Variant) As clsVar	Set specified index number to ID, and acquire VAR object.
Property Get ItemValue (Index As Variant) As Variant	Acquire the value of the specified index number.

Property Let ItemValue (Index As Variant, newVal As Variant)	Set the value of the specified index number.

Table3-4 VAR object member list

Member	Description
Property Get Bin8 () As Byte	Convert eight consecutive VARS variables to Boolean type, pack them into Byte type, and acquire.
Property Let Bin8 (cVal As Byte)	Unpack Byte type variables and convert them to eight consecutive Boolean type VARS variables.
Property Get Bin16 () As Integer	Convert 15 consecutive VARS variables to Boolean type, pack them into byte type, and then acquire. (Note) The sign bit is ignored.
Property Let Bin16 (iVal As Integer)	Unpack Integer type variables and convert them to 15 consecutive Boolean type VARS variables. (Note) The sign bit is ignored.
Property Get Bin32 () As Long	Convert 31 consecutive VARS variables to Boolean type, pack them into Long integer type, and acquire. (Note) The sign bit is ignored.
Property Let Bin32 (lVal As Long)	Unpack Long integer type variables and convert to 31 consecutive Boolean type VARS variables. (Note) The sign bit is ignored.
Property Get Updated () As Boolean	Acquire whether the value is updated (True) or not (False). (Note) Local memory area is not supported.
Property Get Changed () As Boolean	Acquire whether the value is updated (True) or not (False). (Note 1) Local memory area is not supported. (Note 2) If your CaoScript version is 1.4 or lower, it works as Updated property.
Property Let Changed (bFlag As Boolean)	The change flag of the value is set. (Note) Local memory area is not supported.
Property Get DateTime () As Variant	Acquire the time when the value was updated. (Note 1) The resolution of DateTime is a millisecond. (Note 2) Local memory area is not supported.

Property Get Help () As String	Acquire the help character string. (Note 1) To edit help string, directly modify the vars table in datastore.mdb file that exists in the DataStore provider folder. (Note 2) Local memory area is not supported.
Property Get Macro () As String	Acquire the macro character string. (Note 1) To edit macro string, directly modify the vars table in datastore.mdb file that exists in the DataStore provider folder. (Note 2) Local memory area is not supported.
Property Get ID () As Variant	Acquire ID.
Property Let ID (newVal as Variant)	Set ID.
Property Get Value () As Variant	Acquire value.
Property Let Value (newVal As Variant)	Set values. The value stored in the VARS variable is permanently saved.

Table3-5 DBG object member list

Member	Description
Property Get Destination () As Integer	Acquire default output destination of Output.
Property Let Destination (iDest As Integer)	Set default output destination of Output. The initial value of the Destination property is 0. 0: Output to CaoScript log window. 1: Output using OutputDebugString() of Win32 API. (Example) Dbg. Destination = 1
Sub ClearLog ()	Clear log window display.
Sub Output (Optional vntMsg As Variant, Optional iDest as Integer)	Output message. The output destination is assigned by an argument iDest. If the destination is not specified, the value specified by the Destination property is used. (Example) Dbg.Output "hello", 1
Sub Stopoff ()	Suspend execution. It operates as the same as pressing [Pause] button..

Table3-6 APP object member list

Member	Description
Property Get Error () As clsError	Acquire error object.
Property Get ExitCode () As Variant	Acquire exit code.
Property Let ExitCode (vntRet As Variant)	Assign desired exit code. The assigned value is saved until next value is written.
Property Get FileName () As String	Acquire file name.
Property Get Path () As String	Acquire file path.
Property Get Priority() As Long	Acquire the current thread priority.
Property Let Priority(newVal As Long)	Set the thread priority. - Priority (low priority to high priority) THREAD_PRIORITY_IDLE = -15 THREAD_PRIORITY_LOWEST = -2 THREAD_PRIORITY_BELOW_NORMAL = -1 THREAD_PRIORITY_NORMAL = 0 THREAD_PRIORITY_ABOVE_NORMAL = 1 THREAD_PRIORITY_HIGHEST = 2 THREAD_PRIORITY_TIME_CRITICAL = 15
Property Get ThreadId() As Long	Acquire current thread ID.
Property Get TimerInterval() As Long	Acquire currently enabled timer event interval in the unit of millisecond. Acquired 0 when timer event is disabled.
Property Let TimerInterval(newVal As Long)	Set timer event interval in the unit of millisecond. Setting 0 disables timer event.
Sub AddScript(strFilename As String, Optional vntOpt As Variant)	Dynamically add script saved in a specified file. (Example) app.AddScript "Share.vbs"
Function Attach (caoCtrl As Variant) As clsCaoMess	Make program possible to receive OnMessage event of CaoController object that is added to CAO object. App_OnMessage event is generated after this function is executed.
Sub Clear()	Clear timer event generation interval. This function disables timer event.
Sub OnErrorGoto (strSub As String)	Set subroutine to which program execution jumps when an error occurs. Argument assignment is possible. Assigning blank string ("") clears the setting. (Example) app.OnErrorGoto "ErrHandler 123"

Function RunScript(strFilename As String, strStartup As String, Optional lStartMode As Long = 1, Optional bStopUnload As Boolean = True) As CaoScript.clsEngine	Start scripts saved in a specified file asynchronously. Argument strStartup has the same meaning as Startup property of Automation interface, and lStartMode has the same meanings as the first argument of the Startup property. If bStopUnload is True, the running program is stopped when the created CaoScript.clsEngine object is deleted. If bStopUnload is False, the program execution is continued. (Example) obj = app.RunScript ("Multi.vbs", "Main") [Remarks] It is more practical to execute StartScript with Script property of Mgr object than using this member.
Function ShellExec (vntPath As Variant, Optional vntStyle As Variant) As Double	Start an executable program, and return task ID of the program. When a problem occurs on the program execution, 0 is returned. A detailed specification is the same as the Shell function of the Visual Basic. (Example) app.ShellExec("notepad.exe") 'start note pad
Sub ActivateWindow (vntTitle As Variant, Optional bWait As Boolean = False)	Activate an application window. A detailed specification is the same as the AppActivate statement of the Visual Basic.
Sub SendKeyCode (strKeys As String, Optional bWait As Boolean = False)	Send a keystroke or a combination of keystrokes to the active window, just like inputting from a keyboard. A detailed specification is the same as the SendKeys statement of the Visual Basic.

Table3-7 DAT object member list

Member	Description
Function BstrFromVariant (vntVal as Variant) as Bstr	Convert Variant object to a RAC character string. (Example) strTest = dat.BstrFromVariant(vntValue)
Function ChangeType (vntSrc As Variant,vt As Integer) As Variant	Change to an arbitrary data type. This supports an array but one dimensional array only. See Table3-11 for supported data type. (Example) dat.ChangeType(Array(1,2,3), 4) 'convert to R4(4) array.
Function ToVar (vntSrc As Variant) As Variant	A macro function using ChangeType function, and defined as ChangeType(vntSrc, 12).

Function VariantFromBstr (strVal as String) as Variant	Create Variant object from a RAC character string. (Example) vntVal = dat.VariantFromBstr("8,TestString")

Table3-8 EXT object member list

Member	Description
Sub Delay (dwMilliseconds As Long)	Pause program execution for specified time (ms).
Function GetValueEx (objVar As Variant, Optional defaultVal As Variant, Optional lTimeout As Long = -1) As Variant	Acquire value with retrying. The function retries until timeout limit or successfully read value (objVar.Value). When defaultVal is specified, the value is returned on timeout. The function retries forever when lTimeout =-1. (Example) ext.GetValueEx(objVar, "default", 1000)
Sub PutValueEx (objVar As Variant, newVal As Variant, Optional lTimeout As Long = -1)	Set value with retrying. The function retries until timeout limit or successfully set value (objVar.Value). The function retries forever when lTimeout =-1. (Example) ext.PutValueEx objVar, 123, 1000
Sub WaitIo (objVar As Object, Optional bPositive as Boolean = True, Optional lTimeout As Long = -1, Optional bHandShake As Boolean = False)	Wait until one of the following condition is satisfied. - objVar.Value becomes other than zero value (when bPositive = True) - objVar.Value becomes zero (when bPositive = False) - execution time reaches timeout limit. If timeout limit is not specified, the function waits forever until one of the following condition is satisfied. Handshaking function bPositive = True & bHandShake = True: WaitAndReset operation bPositive = False & bHandShake = True: SetAndWait operation
Sub BeepEx (Optional iCount As Integer = 1, Optional lInterval As Long = 0)	Beep sound. The argument specifies the beep count and beep interval (ms). When argument is omitted, the beep count will be one time and interval will be 0 msec. (Note) Beep time and frequency varies on each computer, because they depend on the computer hardware and system software.

Table3-9 CAOMESS object member list

Member	Description
Property Get Index() As Long	Get index number to identify CaoController object.
Sub Attach (caoCtrl As Variant)	Register CaoController object to receive OnMessage event of CaoController. (Remarks) CAOMESS object is at first generated in the Attach function of the APP object. Use this function when you explicitly reuse the same object after Detach it.
Sub Detach()	Detach event sink of OnMessage for not receiving the event. To reactivate it, re-register with Attach function.

Table3-10 ERROR object member list

Member	Description
Property Get Description () As String	Get an error description.
Property Get Handler() As String	Get a value set with App.OnErrorGoto.
Property Let Handler (newVal As String)	Same as App.OnErrorGoto.
Property Get Line () As Long	Get an error line number.
Property Get ExtLine () As Long	Get an error line number when the error occurs in the program merged with App.AddScript.
Property Get Number () As Long	Get an error number.

Table3-11 Type list that can use ChangeType

VT type	Value	Type information
VT_BOOL	11	Boolean
VT_I1	16	One byte signed integer
VT_UI1	17	One byte unsigned integer
VT_I2	2	Two byte signed integer
VT_UI2	18	Two byte unsigned integer
VT_I4	3	Four byte signed integer
VT_UI4	19	Four byte unsigned integer

VT_R4	4	Four byte floating-point real
VT_R8	5	Eight byte floating-point real
VT_CY	6	Currency
VT_DATE	7	Date
VT_BSTR	8	Character string
VT_VARIANT	12	VARIANT

3.2. Embedded event

CaoScript has three types of events as shown in Table3-12.

Table3-12 Embedded event list

Member	Description
Sub App_BeforeExit (lExitMode as Long)	Execution exit event. If implemented, this function is called when program execution ends. [lExitMode] 0: Exit normally 1: Exit by stop 2: Exit by abnormal error
Sub App_OnMessage (lIndex as Long, caoMess as Variant)	CaoMessage event. By implementing this function, program can receive OnMessage event of CaoController class. The types of the event of CaoController object need to be registered beforehand with Attach method of APP object. [lIndex] Index number to identify CaoController object is returned. By comparing it with Index property of the CAOMESS object returned at Attach execution, the event source object can be distinguished. [objCaoMess] Received CaoMessage object is returned. (Example) Dim bb Dim mess Sub Main Set bb = cao.AddController("BB",…)

	Set mess = app.Attach(bb) … End Sub ' OnMessage event handler Sub App_OnMessage(lIndex, caoMess) dbg.output caoMess.Value End Sub
Sub App_OnTimer()	Timer Event. If this function is implemented, the function is repeatedly called at the specified interval. To disable timer function, execute Clear method of APP object.

3.3. Error handling

CaoScript language is based on VBScript language. Therefore, for On Error statement of VBScript, On Error Resume Next (ignore error) can be used. However, if an error occurs during On Error Resume Next execution, the error will be ignored and the execution will continue. In addition to that, even if STOP button is pressed (or ScriptStop method of Automation IF is used), the execution will continue until it reaches one of the following statements.

1. Do, For, While statement
2. End statement

As explained above, since OnError statement of VBScript cannot describe flexible error processing, we prepare expanded OnError statement for CaoScript. In this expansion, OnErrorGoto method of App object is used for error handling. This method sets a subroutine to be jumped when an error occurs. Please note that this error handling process does not call the subroutine but jump to (GOTO) the subroutine. Because this is a jump process, the execution control is handed to the subroutine, and the control cannot return (Resume) to the original error generated point. This means that error-handling setting is automatically cleared after the program jumps on error. If error handling is necessary after the jump, use OnErrorGoto method to set error handling again in the jumped subroutine. In this case, please be careful to avoid infinite loop.

When the program is trapped with OnErrorGoto, to get error information, use Err object of App object because Err object of VBScript is already cleared.

To share a variable between main part of the program and error handling program after an error occurrence, declare he variable at the outside of the function (in the file scope). Since variables declared in the file scope can be shared in the files, error handler can refer the variable value that is set in the main part of the program before OnError jump occurs.

Because only one OnErrorGoto setting is allowed, the latest setting value is effective. Setting null string ("") disables error handling. Passing argument is also possible. Please refer the following examples.

```
' ------------------------------
Sub Main
	...
	app.OnErrorGoto "ErrHandle" 'Error on or after this line is jumped to ErrHandle.
	...
	app.OnErrorGoto "ErrHandleEx 123" 'Error on or after this line is jumped to
ErrHandleEx(123).
	...
	app.OnErrorGoto ""          'Error after this point is not handled.
	...
End Sub

Sub ErrHandle
	Msgbox "Error Handler"
End Sub

Sub ErrHandleEx(Arg1)
	Msgbox Arg1
End Sub
' ------------------------------
```

3.4. VARS variable types

VARS variables are for storing backup data or sharing data between applications. VARS variable has several areas, and each area has different functions and accessing speed. Please choose appropriate type of the variable for your purpose. For example, when the operation panel function is used, the operation panel function runs as a separate process, therefore, it cannot use the local memory.

Vars variable	Data sharing between tasks	Data sharing between applications	Backup	Speed	Usage
Database	Yes	Yes	Yes	Low	It uses to save data after the application program ends. For instance, configuration parameter etc.
Shared memory	Yes	Yes	No	Normal	When the data that wants to share other applications (The example: Operation Panel) and data (share between the processes) when it is time when it need not be backed up, it uses it.
Local memory	Yes	No	No	High	When it only has to share data between tasks, it uses it.

Figure3-2 VARS variable types

【Measurement condition】 Pentium 4, 1.6GHz, 1GB memory

【Measurement program】

```
For i=1 to 10000
    Vars(1001).Value = i     ' DB=1, Mem=501, LMem=1001
    Vars(1002).Value = i     ' DB=2, Mem=502, LMem=1002
Next
```

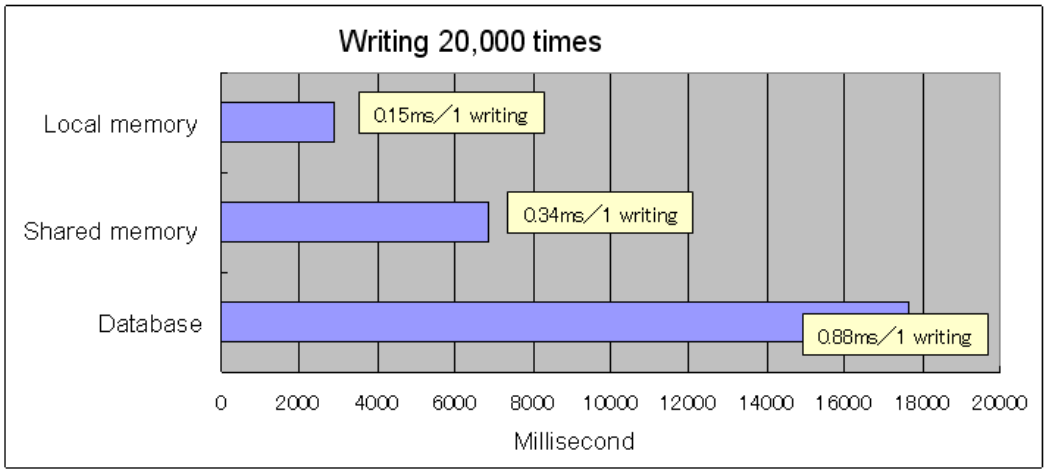
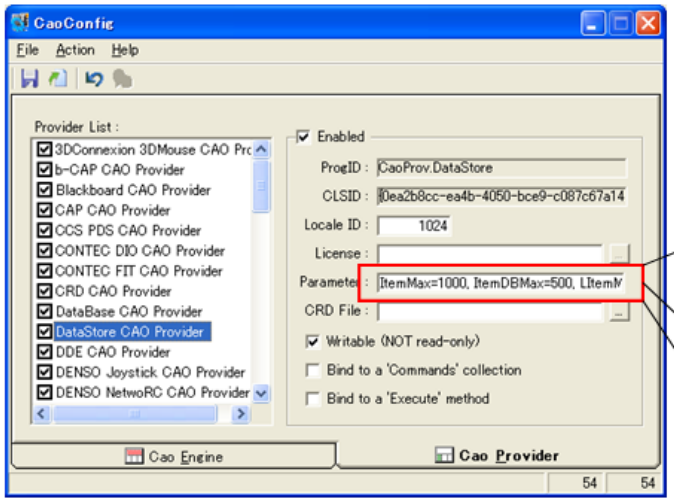


Figure3-3 VARS variables access speed comparisons

VARS variable uses the @VARS variable of the DataStore provider, and the area size is defined with Parameter of the DataStore provider. To set Parameter of the DataStore provider, please use CaoConfig tool.

● Because the Vars variable of CaoScript uses the @VARS variable of the DataStore provider, the area size is defined with Parameter of the DataStore provider.



(Note1) This area is increased in the end of the Vars variable in CaoScript though LItemMax of the DataStore provider is @LVARS originally size of a variable.

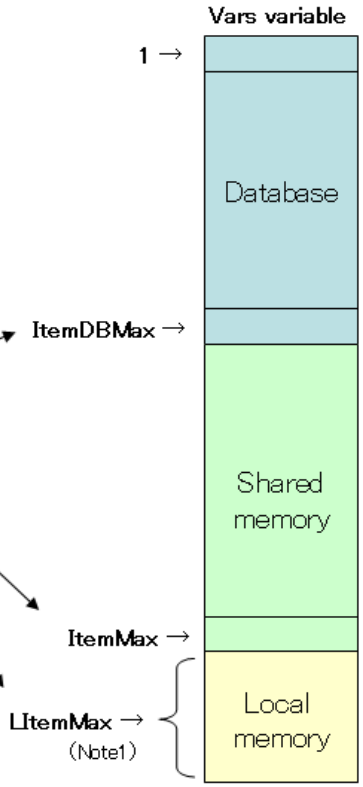


Figure3-4 Area size of VARS variable

4. Automation interface

CaoScript can be controlled from the external program by using the Automation interface.

4.1. Program sample

The following shows an Automation interface-used sample program. This sample program is written with Visual Basic 6.0. To execute this program, "CaoScript" should be made effective in the reference configuration setting. For other detailed examples, please refer to the Tester source code in the following folder.

<ORiN2>/CAO/Tools/CaoScript/Src.Tester

(Example) For VB6

```
Set scr = New CaoScript.clsEngine          ' Create CaoScript engine
With scr
  .ScriptText = "Msgbox ""Hello!"""        ' Set script
  .Startup = ""                            ' Set entry function
  .StartScript 1                           ' Start script
End With
```

4.2. .NET client

To control CaoScript from .NET language (ex. VB2005), set the reference to the following RCW (Runtime Callable Wrapper) instead of referring it as a COM. With this setting, you can avoid creating RCW of CaoScript for each client. For published functions, please refer to Table4-1.

<ORiN2>/CAO/Tools/CaoScript/Bin/CaoScrRCW.dll

RCW of CaoScript Manager is the following file. For published functions, please refer shared object Mgr (Table6-4).

<ORiN2>/CAO/Tools/CaoScript/Bin/CaoScrManRCW.dll

4.3. Automation interface specifications

The following table shows the members of the Automation interface.

Table4-1 Automation interface member list

Member	Description
Property Get ExitCode () As Variant	Acquire exit code set with App.ExitCode.
Property Get FileName () As String	Acquire currently opening file name.
Property Get FileNameChanged () As Boolean	Acquire a flag to change a file name.
Property Let FileNameChanged (bChanged As Boolean)	Set a flag to change a file name.

Property Let GUIEnabled (strObjPath As String, bEnabled As Boolean)	Set Enable of the specified GUI object.
Property Get GUIEnabled (strObjPath As String) As Boolean	Acquire Enable of the specified GUI object.
Property Get hWnd () As Long	Acquire window handle.
Property Get LastLog () as String	Acquire the latest log message on the log window.
Property Get LineNumber () As Long	Acquire current execution line number.
Property Get LogText () as String	Acquire entire log window.
Property Get Modified () As Boolean	Acquire script text modification status.
Property Let Modified (bModified As Boolean)	Set script text modification status.
Property Get ReadOnly () As Boolean	Acquire script text edit permission status.
Property Let ReadOnly (bReadOnly As Boolean)	Set script text edit permission status.
Property Get ScriptState () As Long	Acquire CaoScript status. The acquired value is as follows. -1: Initializing. 0:Script not running 1:Script running. 2:Script is generating error
Property Get ScriptText () As String	Acquire script text.
Property Let ScriptText (strText As String)	Set script text.
Property Get Startup () As String	Acquire execution function name.
Property Let Startup (strText As String)	Set execution function name.
Property Get Tag () As Variant	Acquire tag information.
Property Let Tag (varTag As Variant)	Set tag information.
Property Get Visible () As Boolean	Acquire Visible/invisible state of CaoScript.
Property Let Visible (bVisible As Boolean)	Set Visible/invisible state of CaoScript.
Sub AddObject (strName As String, objAddin As Object)	Register user defined object to a script. This enables to expand CaoScript language function freely. strName is not case-sensitive.

Function CallScript (strText As String) As Variant	Synchronously function call. ExitCode is returned as a result. (Example1) Ret = .CallScript("Sub1(100)") ' Argument array 100 (Example 2) Ret = scr.CallScript("Sub2(""" & text & """)") ' Specify character string as an argument.
Sub ClearLog ()	Clear log window.
Sub Execute (strCmd As String, Optional vntParam As Variant)	Execute extended functions. Refer to Table4-3.
Sub Hide (Optional vntOpt As Variant)	Hide CaoScript.
Sub Initialize (Optional vntOpt As Variant)	Initialize. (Remarks) This function is currently not used.
Sub OpenFile (strFilename As String, Optional vntOpt As Variant)	Open specified file.
Sub RemoveObject (strName As String)	Remove object added with AddObject. strName is not case sensitive. (Remarks) Objects are automatically removed when CaoScript engine is removed.
Sub SaveFile (strFilename As String, Optional vntOpt As Variant)	Save present content to the specified file.
Sub Show (Optional vntOpt As Variant)	Show CaoScript.
Sub StartScript (lMode As Long, Optional vntOpt As Variant)	Execute scripts asynchronously. 1: Single-cycle start 2: Continuous start 3: Single step start 4: Single step back 5: High speed execution 6: Single step start (not displayed)
Sub StopScript (lMode As Long, Optional vntOpt As Variant)	Stop script asynchronously. 0: Default stop (same as 1). 1: Halt 2: Step stop 3: Cycle stop 4: Initialization stop (same as 1). 5: Step stop (not displayed)

Sub Terminate (Optional vntOpt As Variant)	Execute termination preparation process.

- A GUI object name of the GUIEnabled property is specified by the following formats.

<parent object name>[.<child object name>[.<grandchild object name>]]

For parent object name, menu or tool bar can be assigned. Assign "Menu" for menu, or "Tlbar" for tool bar. For child object name or grandchild object name, refer to Table4-2.

Table4-2 Object name list for GUIEnabled

Parent object name	Child object name	Grandchild object name
menu[1]	file	new
		open
		save
		save_as
		save_log_as
		exit
	edit	undo
		cut
		copy
		paste
		delete
		select_all
		find
		find_next
		replace
		goto
		break_point
		line_trace
		clear_log
	form	font
	run	start
		step_over
		pause
		stop

[1] Acquiring or setting "Enabled" of the top menu is not possible

		fast_start
		cycle_start
		cycle_stop
	help	version
tlbar	open	
	save	
	save_log	
	Cut	
	copy	
	paste	
	find	
	comment	
	del_comment	
	startup	
	start	
	step_over	
	pause	
	stop	

For instance, to specify [File]–[Save As…], GUI object name is "menu.file.save_as". To specify [Cut] on the toolbar, GUI object name is "tlbar.cut".

Table4-3 Command list available with Execute

Command name	Argument	Description
GetScriptStarted	None	When it is asynchronously executed by StartScript, True will be returned if the script is already started.

5. Command line option

On starting CaoScript.exe, the following option can be specified.

CaoScript.exe <file name>[<function name> [start]]

< file name >: File name to open

< function name >: Startup function name

< start>: Start script automatically

(Example) > CaoScript.exe d:¥temp¥test.vbs PRO1 start

6. CaoScriptManager

6.1. Outline

CaoScript Manager is a tool to manage several CaoScript.

CaoScript Manager displays one CaoScript in one line, changes the settings, and executes scripts.

★2016/6/29 ここまで作業済★

Screen information can be saved in a CaoScript project file (extension: csp) to save and reproduce the manager environment.

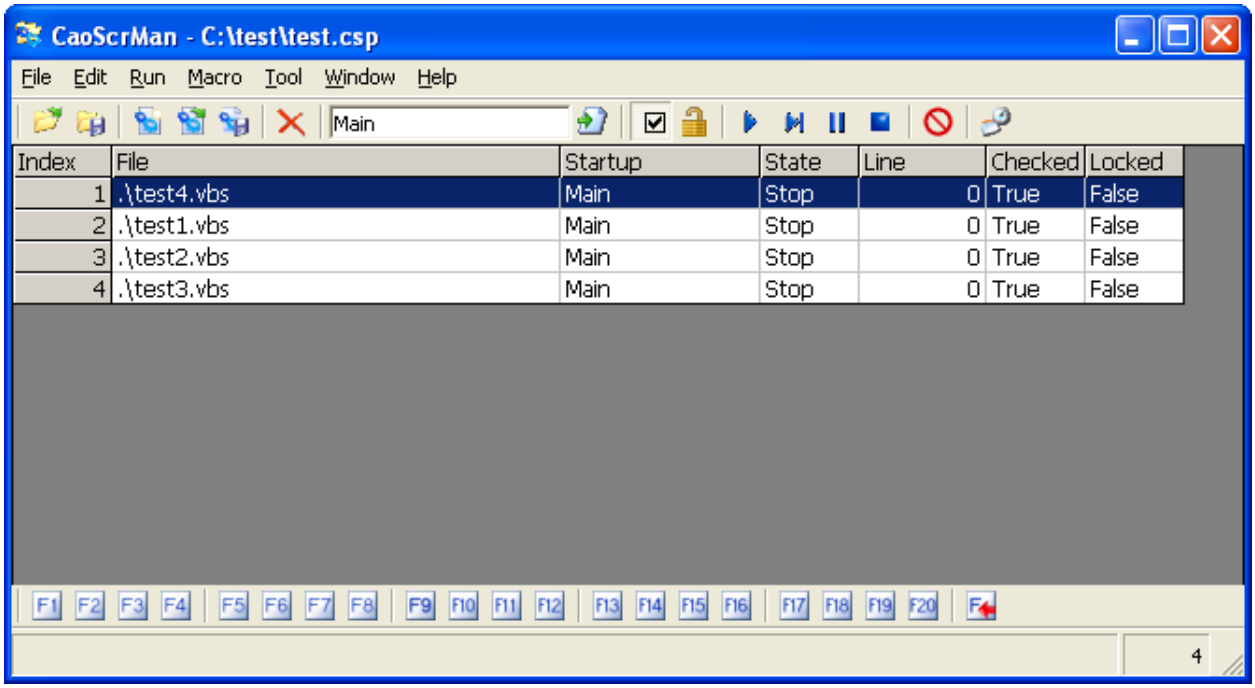


Figure6-1 CaoScriptManager start window

6.2. Window layout

6.2.1. CaoScript display grid

Index	File	Startup	State	Line	Checked	Locked
1	¥test1.vbs	main2	0	0	False	True
2	¥test1.vbs	main5	0	0	True	False

Figure6-2 CaoScript display grid

For information shown in each row, refer to Table6-1.

Table6-1 CaoScript display grid information

Column name	Description	Corresponding CaoScript property name
Index	Line number. When two or more lines are selected, execution order is ascending from the smallest Index number line.	(none)
File	File name opened with CaoScript.	FileName
Startup	Startup function name.	Startup
State	CaoScript status value	ScriptState
LineNumber	Current execution line of CaoScript.	LineNumber
Checked	For details, refer Target of execution menu (6.2.4) and window menu (6.2.7).	(none)
Locked	Script window edit enabled/disabled setting.	ReadOnly

6.2.2. File menu

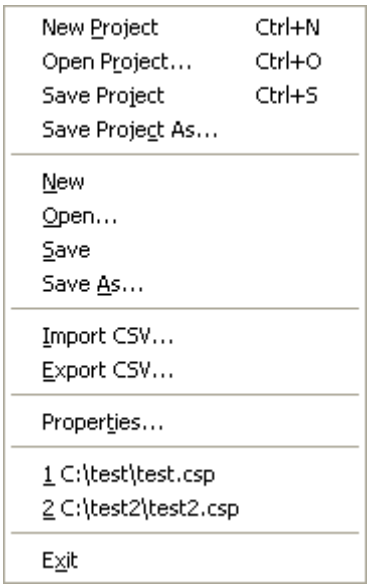


Figure 6-3 File menu

[New Project]

Create new project

[Open Project…]

Open CaoScript project file.

[Save Project]

Overwrite currently opened project.

[Save Project As…]

Save currently opened project with assigned name.

[New]

Add CaoScript without file name.

[Open]

Open the specified file and add it as CaoScript.

[Save]

Save CaoScript of currently selected line.

[Save As…]

Save CaoScript of currently selected line with specified file name.

6.2.3. Edit menu

The Edit menu operation is applied to all selected lines.

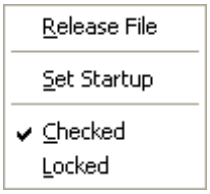


Figure6-4 Edit menu

[Up]

Move up selected line by one line.

[Down]

Move down selected line by one line.

[Release]

Delete currently selected line.

[Set Startup]

Set startup function of selected line CaoScript as a function shown in the "startup function name setting box."

Figure6-5 Startup function name setting box

[Checked](Initial value "True")

To handle plural numbers of files, check "True". The figure shows the checked status. The status becomes false by choosing it again. For example, if [Run] – [target] – [check] is selected, all files with check button set to "True" will be executed.

[Locked]

"Locked" is to prohibit file handling and other purposes. By setting "Locked" to "True", editing and running the file become not possible.

6.2.4. Run menu

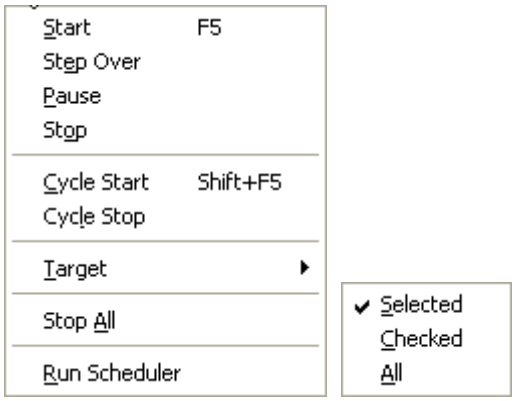


Figure6-6 Run menu

[Start]

Execute Start method of CaoScript specified in **[Target]** condition.

[Step Over]

Execute Step Over method of CaoScript specified in **[Target]** condition.

[Pause]

Execute Pause method of CaoScript specified in **[Target]** condition.

[Stop]

Execute Stop method of CaoScript specified in **[Target]** condition.

[Target]

Select execution target line condition.

[Selected] : Currently selected line

[Checked] : Lines where Checked row is True.

[All] : All lines

[Run Scheduler]

Run scheduler task.

6.2.5. Macro menu

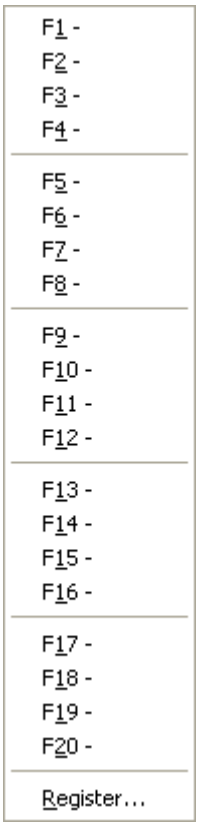


Figure6-7 Macro menu

[F1-20]

The registered macro is executed.

[Register...]

Macro window (6.2.10) is opened.

6.2.6. Tool menu

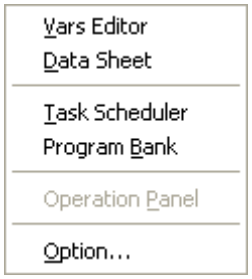


Figure6-8 Tool menu

[VARS Monitor]

Open VARS Monitor window (6.2.9).

[Task Scheduler]

Open Task Scheduler window (6.2.14).

[Option]

Open Option window (6.2.15).

6.2.7. Window menu

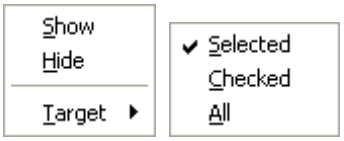


Figure6-9 Window menu

[Show]

Execute Show method of CaoScript specified in **[Target]** condition line.

[Hide]

Execute Hide method of CaoScript specified in **[Target]** condition line.

[Target]

Select execution target line condition.

[Selected] : Currently selected line

[Checked] : Lines where Checked row is True.

[All] : All lines

6.2.8. Help menu

Figure6-10 Help menu

[License]

Display license setting window.

[Version]

Display version information.

6.2.9. Project Properties window

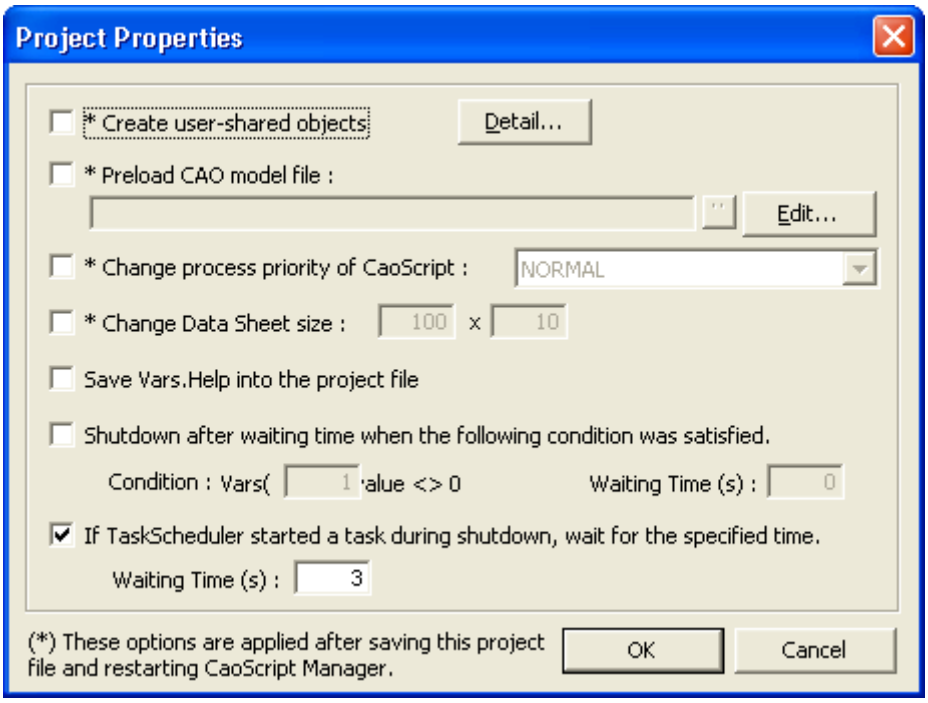


Figure6-11 Project Properties window

[**Create user-shared objects**]

Choose whether user shared object is created. By checking the box, user predefined shared objects of CaoScript are additionally registered. 5.2.9.1 explains how to set the shared objects.

[**Preload CAO model file**]

CRD files is used to specify the object structure that are created when "Cao" starts. By pressing "Edit" button, CRD file can be edited by associated editor program. Property setting cannot be finished while CRD file is edited.

[Change process priority of CaoScript]

Select CaoScript priority from IDLE, BELLOW NORMAL, NORMAL, ABOVE NORMAL, or REAL TIME.

[Change Data Sheet size]

Specify data sheet raw and column size.

[Save Vars.Help info the project file]

Choose whether Help property of Vars object is saved into the project file.

[Shutdown after waiting time when the following condition was satisfied.]

The end condition of CaoScriptMan is set. The value of the specified Vars variable ends when the check box is turned on and CaoScrMan ends when the values other than 0 continue at specified time (The unit: second).

(note)When this setting is newly done, the value of the specified Vars variable is initialized by 0.

[If TaskScheduler started a task during shutdown, wait for the specified time.]

Wait specified time if task scheduler started a task during shutdown.

To activate (*) marked setting, save project file and restart CaoScriptManager.

6.2.9.1. User shared object

User shared object is to add user created shared object to CaoScript. By using the function, CaoScript can use object without calling CreateObject.

Created object can be shared between script files. Following is the way for sharing setup.

At first, to setup user-shared object, select "Create CaoSQL object" in the property window of Fig. 5-11. Then, press "…" button for setup details.

Figure6-12 User CaoSQL object

Setup dialog is shown. Setup each items and press "Add" button.

When added, object name is shown at the last line of the list. Check only necessary objects.

To delete objects, select object names from the list, and press "Delete" button.

Share object names shown in blue color (only "Csq" for current implementation) cannot be deleted because it is CaoScrMan default setting. An warning is shown when the same name shared name is added. Following is an explanation for setup dialogue items.

[Shared name]

Specify shared name. The name is used as shared object name in CaoScript.

[Class name]

Specify class name for shared object creation. (The name is used as an argument for "CreateObject")

6.2.10. Macro window

The content of the registration of the macro is managed.

Figure6-13 Macro window

[Macro No.]

Macro number displayed in macro window

[Enabled]

It makes it to the having efficacy of the specified macro number.

[File]

The script file that the macro executes is specified.

[Startup]

The start-up function that the macro executes is specified.

[Tool Hint]

The hint of the macro is specified.

[Clear]

The content of the specified macro number set is cleared.

[Clear All]

All the macro settings are cleared.

6.2.11. VARS Editor window

Vars Editor manages @VARS system variable of the DataStore provider.

About the @VARS system variable, refer to DataStore Provider Guide.

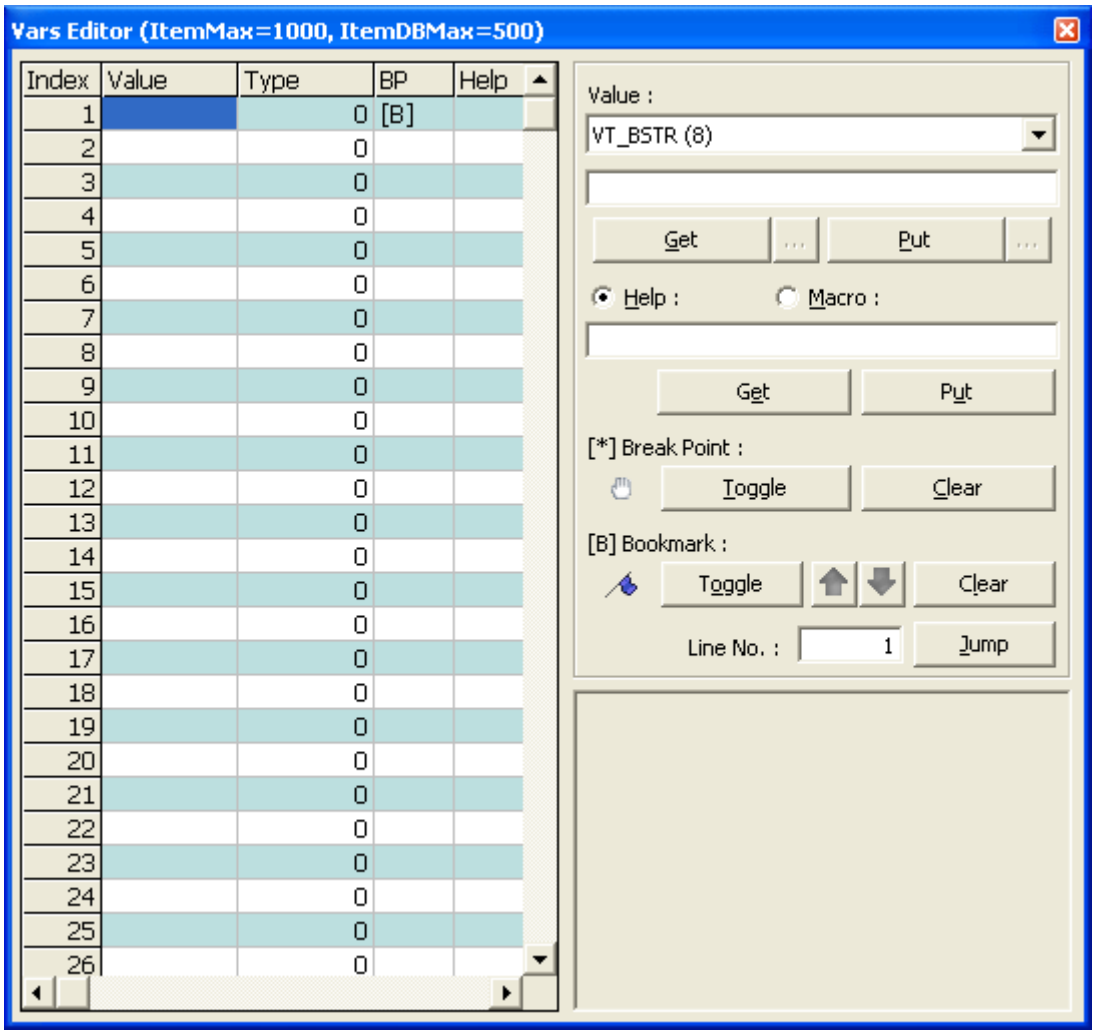


Figure6-14 VARS Monitor window

[Value]-[Get]

The value of the line that has been selected is acquired. When two or more lines are selected, only the selected row is acquired in the beginning.

[Value]- [Put]

The value is set to the line that has been selected. When two or more lines are selected, it sets it to all lines that have been selected.

[Help]-[Get]

Help of the line that has been selected is acquired. When two or more lines are selected, only the selected row is acquired in the beginning.

[Help]-[Put]

Help is set to the line that has been selected. When two or more lines are selected, it sets it to all lines that

have been selected.

[Macro]-[Get]

The macro of the line that has been selected is acquired. When two or more lines are selected, only the selected row is acquired in the beginning.

[Macro]-[Put]

The macro is set to the line that has been selected. When two or more lines are selected, only the selected row is set in the beginning.

[Break Point]-[Toggle]

The breakpoint is set and it releases it to the line that has been selected. When two or more lines are selected, only the selected row is set in the beginning.

[Break Point]-[Clear]

All breakpoints are released.

[Bookmark]-[Toggle]

The bookmark is set and it releases it to the line that has been selected. When two or more lines are selected, only the selected row is set in the beginning.

[Bookmark]-[Clear]

All bookmarks are released.

[Jump]

The cursor is moved to the specified line.

6.2.12. Vars event handler window

The Vars event handler is set.

When the value of registered Vars changes, the Vars event handler executes the specified handler.

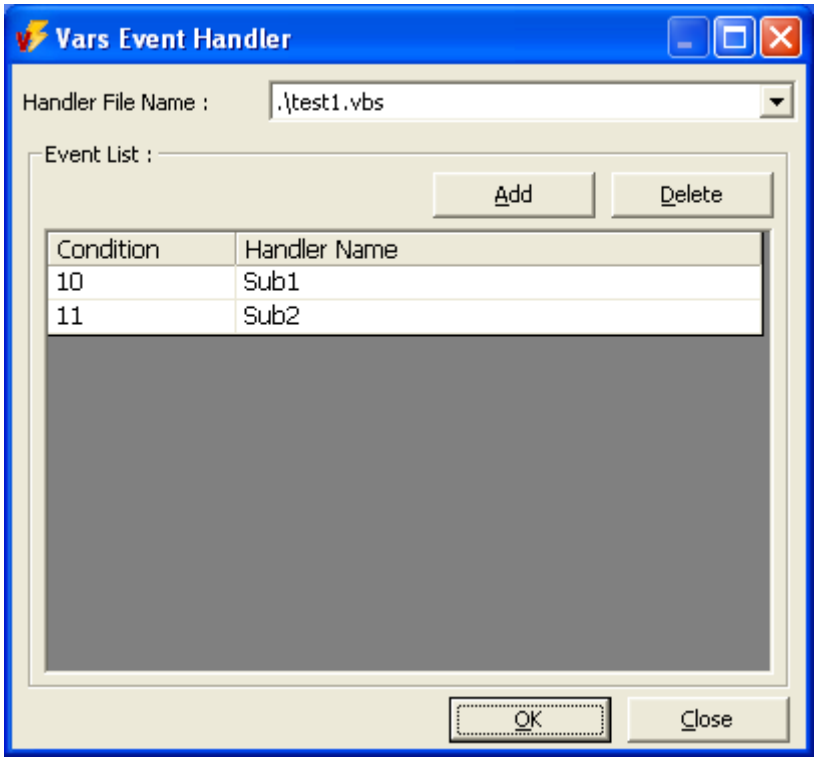


Fig. 6-1 Vars event handler window

[Handler file name]

Select an event handler script file from a list of CaoScrMan script list.

[Add]

Add empty Vars Event line at the bottom.

[Delete]

Delete the selected Vars Event.

[Event list]

Vars ID and the handler name are input. Only the numerical value is set to Vars ID. Please specify the handler that exists in the specified script for a handler name. When the handler that doesn't exist is specified, registered Vars cannot call the event handler even if the value changes.

6.2.13. Data Sheet window

The data stored in the Csv object is displayed and edited.

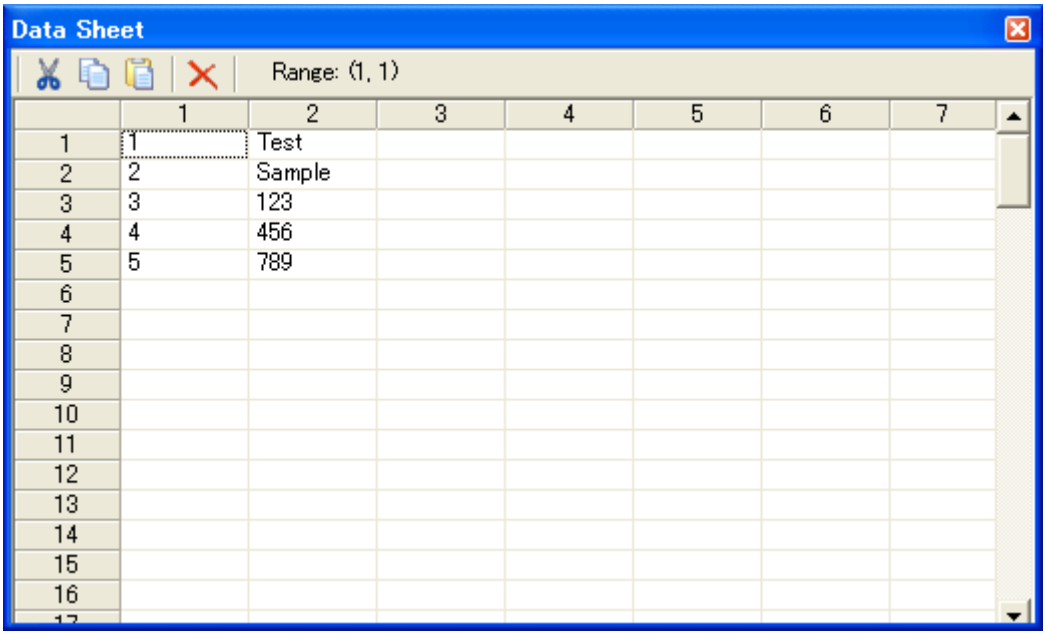


Figure6-15 Data Sheet window

6.2.14. Task Scheduler window

Task scheduler displays a list of task schedule. Check box sets ON/OFF of task scheduling.

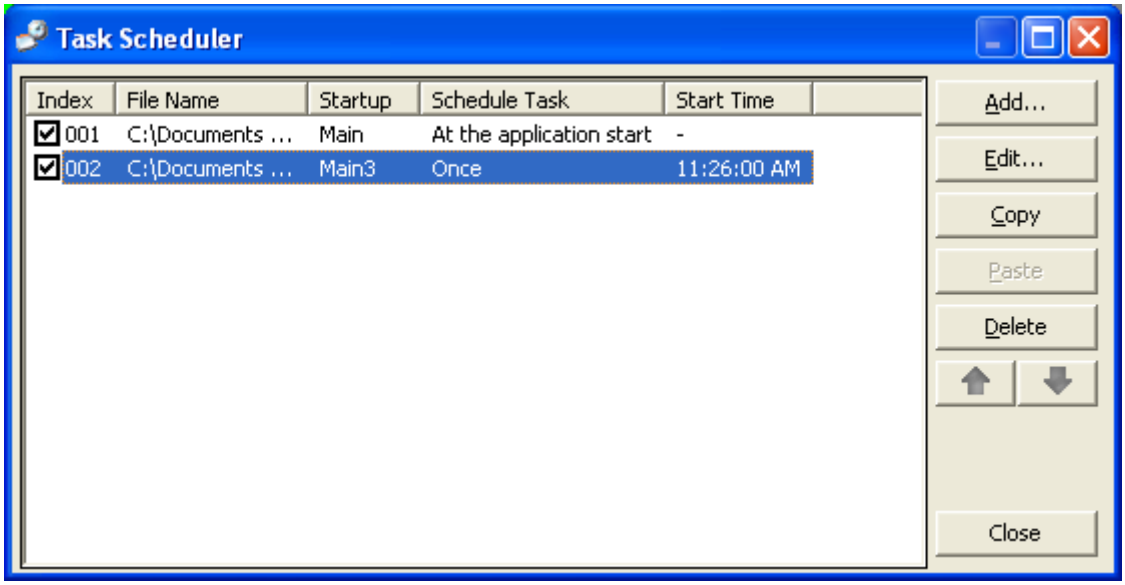


Figure6-16 Task Scheduler window

[Addition]

The task schedule is added.

[Edit]

The task schedule that has been selected is edited.

[Copy]

The task schedule that has been selected is copied.

[Putting]

The copied task schedule is put.

[Deletion]

The task schedule that has been selected is deleted.

To add and edit task schedule, following window is used for detailed setting.

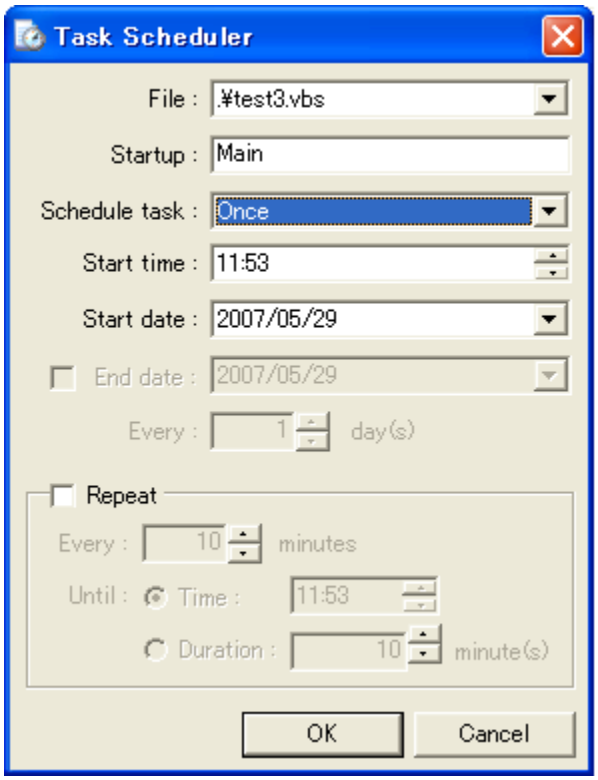


Figure6-17 Task Scheduler edit display (schedule setting)

[File]

File name to set task schedule

[Startup]

Startup function that executes task schedule

[Schedule task]

Task schedule execution type

[Once] : execute at specified time.

[At Startup] : execute at when task schedule starts.

[Daily] : execute every day.

[By Vars] : execute by Vars variable trigger setting

[Start time]

Task schedule start time

[Start date]

Task schedule start date

[End date]

Task schedule end date

[Every x day(s)]

Execution days interval when "Daily" is specified

[Repeat]

Specify repeated task execution

[Every]

Task execution interval on repeated task execution

[Until]

Execution end time on repeated task execution

[Time]

Specification by time. If the specified time is before the current time, the repeated task stops on the specified time of the next day.

[Duration]

Specification the duration before stopping the repeated

[Condition]

Vars variable setting as task starting trigger

[Action]

Triggered task action

[Post Proc]

Vars variable value setting after the trigger-started repeated task execution is ended.

6.2.15. Option window

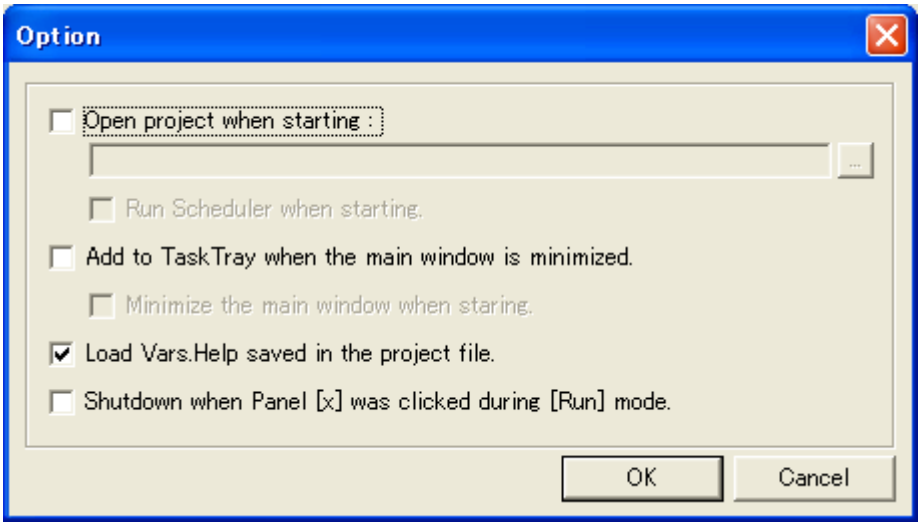


Figure6-18 Option window

[Open project when starting.]

The project file specified when CaoScriptManager is started is opened.

[Run Scheduler when starting.]

When CaoScriptManager is started, the task scheduler is executed.

[Add to TaskTray when the main window is minimized.]

When CaoScriptManeger is minimized, it adds it to the task tray.

[Minimize the main window when staring.]

When CaoScriptManager is started, it minimizes it.

[Load Vars.Help saved in the project file.]

The Help property of the Vars object preserved in the local is loaded.

[Shut down when Panes[x] was clicked during [Run] mode.]

When shutting while executing the Operation Panel window, CaoScriptManager is ended.

6.3. Vars event

The Vars event handler function can call an arbitrary handler by specifying the file that describes the handler, and setting observed Vars.

As for CaoScript set to the Vars event handler, Startup automatically becomes "Dispatcher__". Startup cannot be changed while it is setting it to the event handler.

When the setting ends, CaoScript set to the event handler is executed.

When the tenth Vars becomes "123" as an example, set-up steps of the example of calling the handler of "Sub1" described in "TextScr.vbs" are shown as follows.

"Vars event handler" is called, and set from "Tool" menu of CaoScrMan first.

In the setting of the Vars event handler, the file that first becomes an event handler is selected. Next, the event item is added to the event list. Because an empty condition is added when the add button is pushed, Vars ID and the handler name are input.

First of all, ID of Vars registered as an event is specified. The handler name describes the handler name that exists in the file specified for an event handler. If the handler name that doesn't exist here is specified, the handler is not correctly called even if the value of Vars changes.

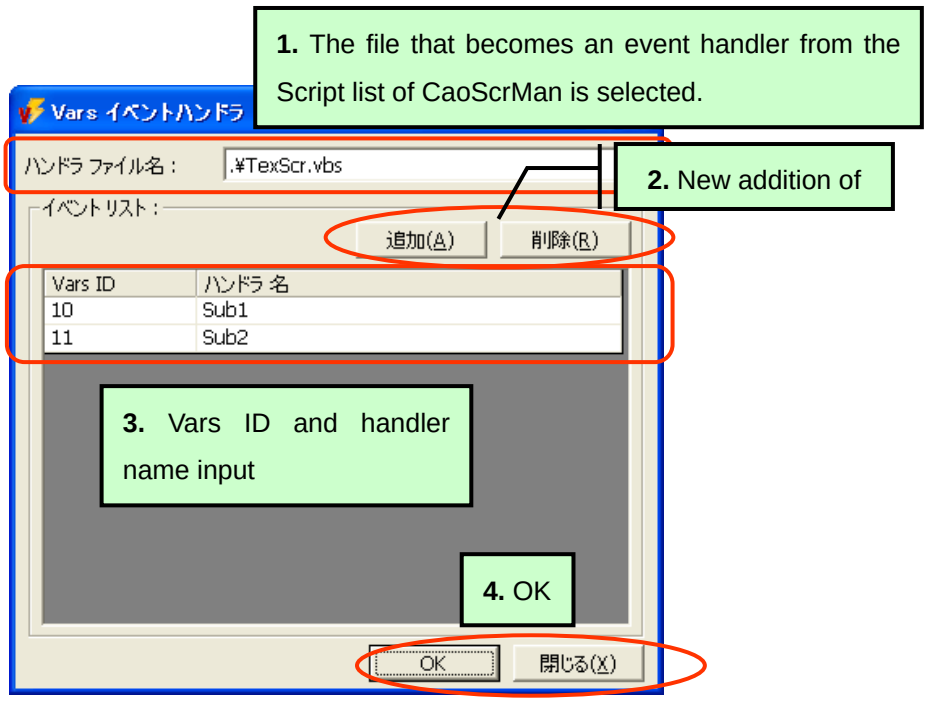


図 6-2 Setting of Vars event handler

Next, CaoScript set from the main window of CaoScrMan to the event handler is executed.

The tenth Vars is written in "123" in the Vars editor so that the Vars condition of setting it a little while ago may consist.

If the condition consists, "Sub1" of "TextScr.vbs" is executed, and it is confirmed that the message box is displayed.

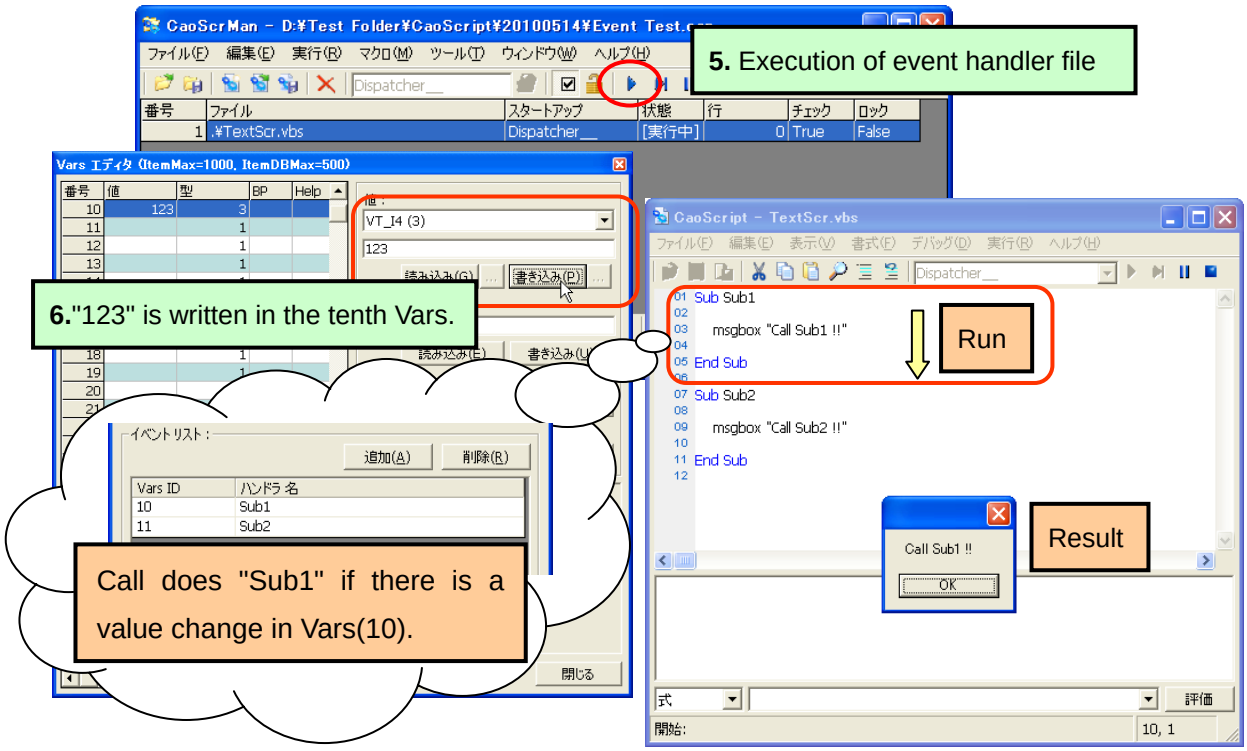


図 6-3 Approval result of Vars event condition

However, because the Vars event calls the handler by the change of the value, it is possible to achieve it by describing the condition in the handler to execute processing when changing into a specific value as follows.

Example. When assuming that it wants to execute processing when the value of Vars becomes 123

```
Sub Sub1

    If Vars(10).Value = 123 Then
            Msgbox "Call Sub1 !!"
    End If

End Sub
```

[Attention]

The Vars event uses the change of the value and the Changed property of the Vars object is used.

When the event is set, the code for the event is added to Dispatcher__.

```
If Vars(x).Changed Then Handler :Vars(x).Changed = False
```

The change of the value of Vars is confirmed in the Changed property, and the Changed property is cleared. Therefore, when the Changed property is treated in the handler, it is necessary to note it.

Moreover, when Vars registered in the Vars event is changed while Script file (Dispatcher) that describes the event handler has stopped, the Changed property becomes true. Therefore, the event is generated because the value change occurs in the Changed property if event Script (Dispatcher) is started next. This is movement of the specification of the Vars event.

6.4. Method of making user common object (For MicroSoft Visual Basic 6.0).

It introduces the making method here through 6.2.9.1 introduced the setting method of the user common object. It is possible to make it also with Visual C++ etc. similarly though it uses and it explains Visual Basic 6.0 (henceforth VB6) to making.

The made the user common object displays, and non-displays the form by calling the Show method and the Hide method from CaoScript.

6.4.1. Start of VB6

When you start VB firstFigure6-4The [noyouna] dialog is displayed. Because the user common object operates by the call from the outside, "ActiveX EXE" is selected here.

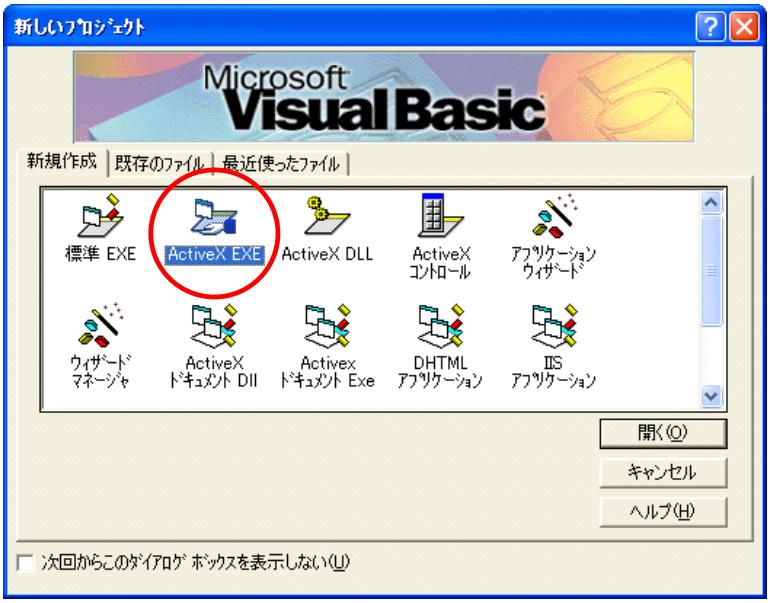


Figure6-4 New project window of VB

First of all, when the main window is opened, Form is added to the project. This time, the introduced the user common object uses Form. If it is ActiveX EXE, and ActiveX DLL without GUI, it is not necessary to add Form. The addition of Form selects addition (F) of a form module of project (P), and specifies preserving ahead.

Next, the whole project is set. When you select project (P) of the menuFigure6-5The [noyouna] dialog is displayed. This time, the made project name is input. Here, it is assumed, "Sample".

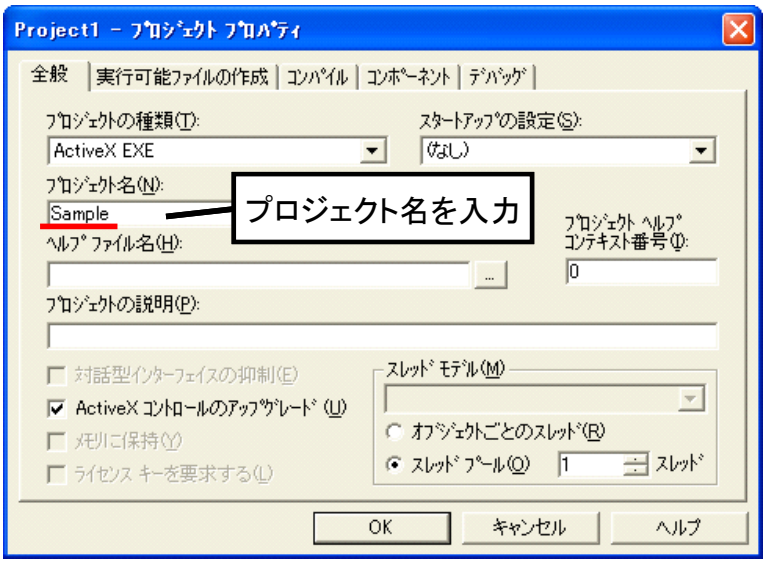


Figure6-5 Set whole tab of project

It continues, and making an executable file of the project is set. Here, a form title name is input. When the setting is completed, OK is done and the dialog is shut.

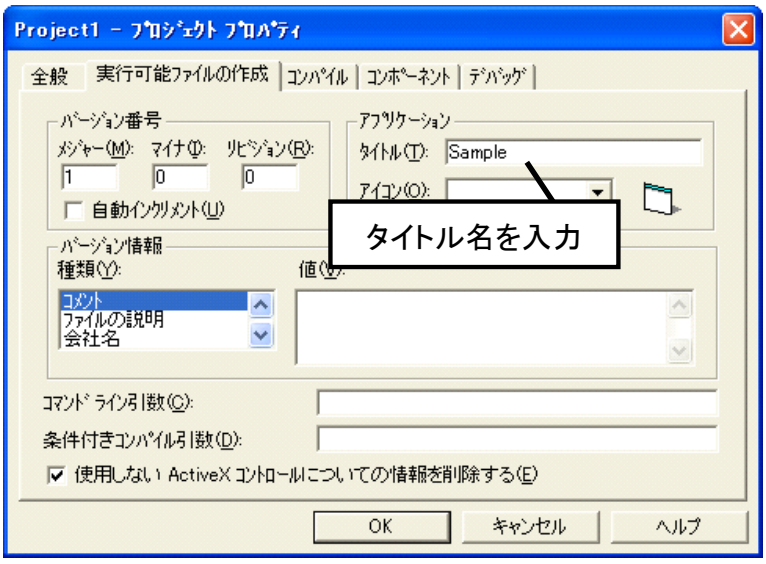


Figure6-6 Set making of executable file tab of project

6.4.2. Input of program (VB6)

The object name of the module is set before the program is input. The name of the class module is set to "ClsSample" this time. (Figure6-7)

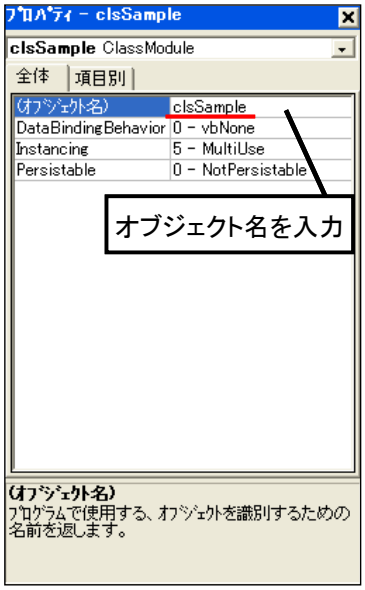
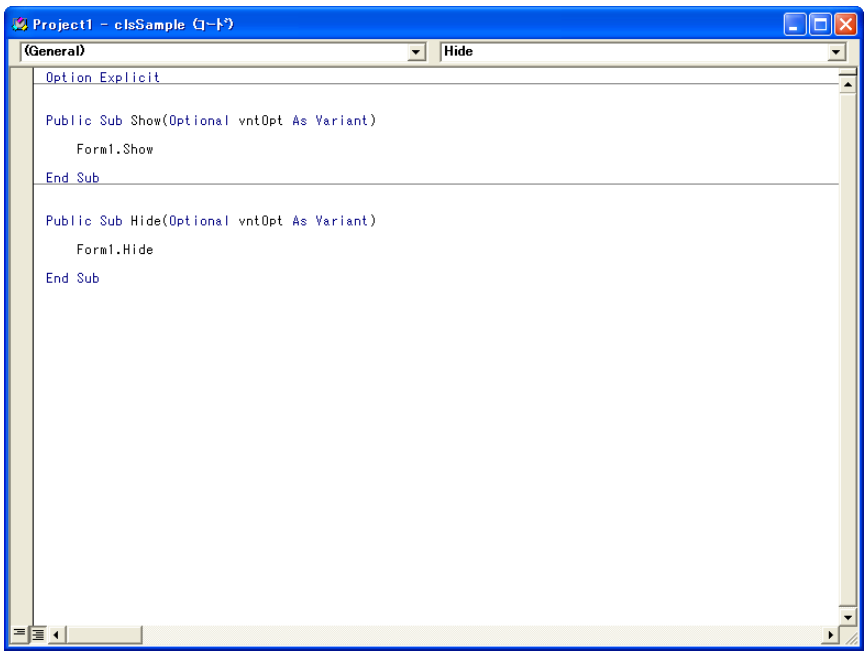


Figure6-7 Property of class module

The Show method and the Hide method that calls from CaoScript are mounted this time.

The function specification is as follows.

- Public Sub Show (Optional vntOpt As Variant)[2]
- Public Sub Hide (Optional vntOpt As Variant)



[2] Argument vntOpt is described for enhancing. This argument is not used in the method of this Show and Hide.

Figure6-8 Program input window

6.4.3. Making of execution file

Next, when the input of the program is completed, the execution file is made. To make the execution file, making "Sample.exe" of the File(F) menu is selected. This name depends by setting the project.

Figure6-9 Making of execution file

6.4.4. The made the user common object is added.

The user common object made in the foregoing paragraph is added. Please refer to 6.2.9.1for the procedure of the addition. Here, the shared name is assumed to be "Smp", and < project name > . and < class module name > are set by the name given when the user common object is made for the class name.

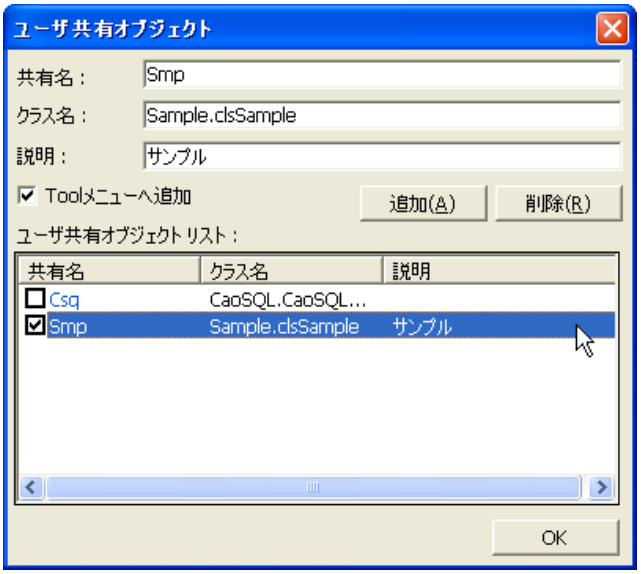


Figure6-10 Setting for user sharing window

6.4.5. Program input (CaoScript)

Next, the program that calls the user common object made from CaoScript is input.

When calling it, it is possible to call it by the method name that has been opened to the public to the shared name applied by registering the user common object ahead.

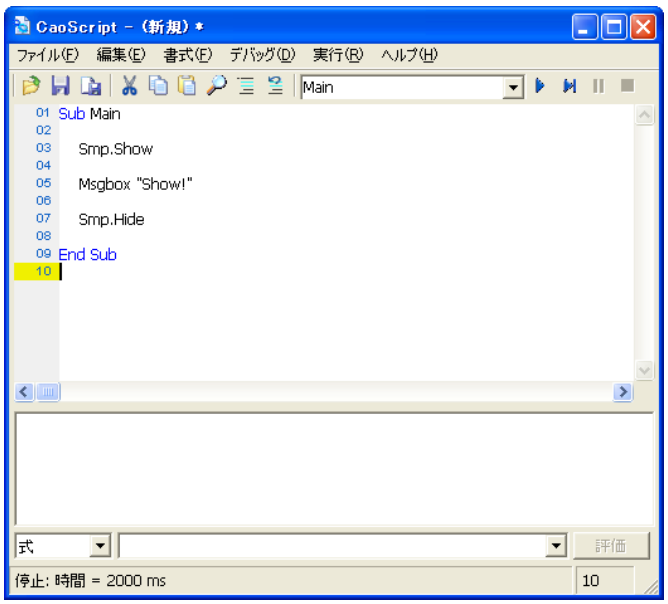


Figure6-11 CaoScript input window

6.4.6. Execution result

It is a result of doing, and executing the above-mentioned setting. Form is displayed, and when the message

box is done in OK, Form does Hide.

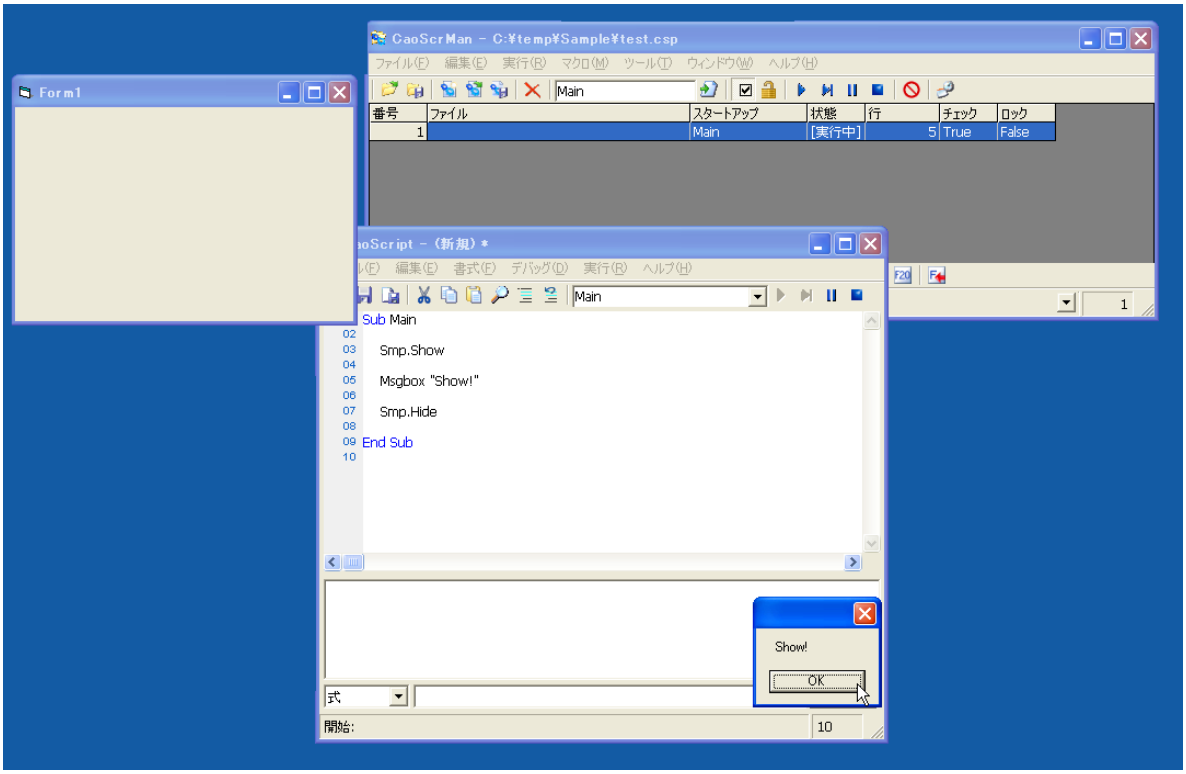


Figure6-12Result of execution for user sharing

The cooperation of the data of the user common object and CaoScript can be easily done by using Vars.

6.5. Add-In of external application by macro registration

It introduces the method of doing an external application in Add-In by using macro registration function () of CaoScriptManager. ShellExec of the App object is used to start an external application with CaoScript. ()

The function is described specifying the Exe passing of the application that starts. Here, "Note pad" installed in Windows by the standard is started.

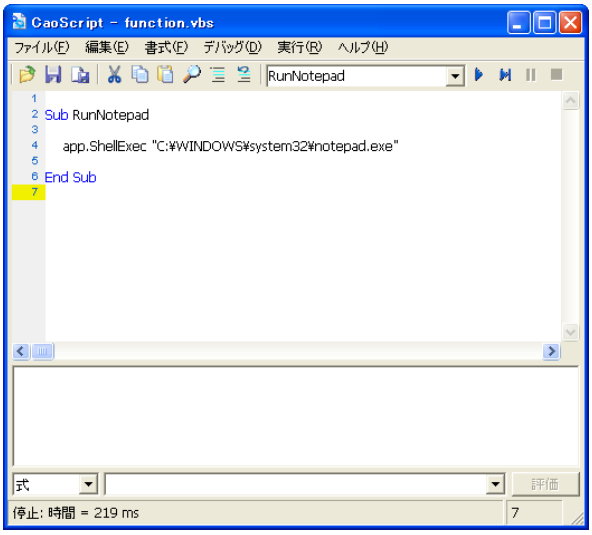


Figure6-13Notepad is started with CaoScript.

Next, described CaoScript (For instance,) is preserved by the name like function.vbs, and it registers in the macro.

Figure6-14Macro registration of Notepad

6.6. Shared object

CaoScriptManager has the following embedded object to be shared between each CaoScript. This object is one entity, and is shared between each CaoScript.

Table6-2 Shared object list

Name	Meaning	Remarks
Csq	CaoSQLEngine object	Provides CaoSQL access function. Use [Create a CaoSQL object] option to

		enable/disable the object.
Cao	CaoWorkspace object	Always enabled object to provide access function to CaoWorkspace.
Pnl	VarsPanel object	Provides Operation Panel accessing function. The access function to Operation Panel is offered. Enabled if necessary license is installed.
Mgr	CaoScriptManager object	Always enabled object to provide access function to scripts registered in CaoScriptManager.
Csv	DataSheet object	Always enabled object to provide data access function to Data Sheet.

Table6-3 Pnl object member list

Member	Description
Property Get Page () As Long	Acquire current page number
Property Let Page (lPage As Long)	Set current page number
Property Get TimeInterval () As Long	Acquire page update interval
Property Let TimeInterval (lInterval As Long)	Set page update interval
Property Get Visible () As Boolean	Acquire whether window is opened (True) or not (False).
Property Let Visible (bVisible As Boolean)	Set whether window is opened (True) or not (False).
Property Get WindowState () As Long	Acquire window status (<>2: Normal, 2:Maximized)
Property Let WindowState (lState As Long)	Set window status
Sub Execute (strCmd As String, Optional vntParam As Variant)	For future extension
Sub Hide (Optional vntOpt As Variant)	Hide panel window
Sub PanelStart (Optional vntOpt As Variant)	Change to Run mode
Sub PanelStop (Optional vntOpt As Variant)	Change to Design mode

Sub Show (Optional vntOpt As Variant)	Show panel window

Table6-4 Mgr object member list

Member	Description
Property Get Count() As Integer	Return the number of CaoScript objects
Property Get FileName() As String	Return CaoScrMan project file name.
Property Get Path() As String	Return CaoScrMan project folder path.
Property Get Script (vntID As Variant) As CaoScript.clsEngine	Acquire CaoScript object. About the members opened by object, refer to "Automation interface". vntID is specified by Index number or file name. Effective Index range is 1 to Count.
Sub Hide (Optional vntOpt As Variant)	Hide CaoScript Manager.
Sub Show (Optional vntOpt As Variant)	Show CaoScript Manager

Table6-5 Csv object member list

Member	Description
Property Get TextMatrix (ByVal lRow1 As Long, ByVal lCol1 As Long, Optional ByVal lRow2 As Long = -1, Optional ByVal lCol2 As Long = -1, Optional strRowDelimiter As String, Optional strColDelimiter As String) As String	Return the specified cell value. (Example) x = csv.TextMatrix(1,1) y = csv.TextMatrix(1,1,1,2,"",",") z = csv.TextMatrix(1,1,2,2,":",",")
Property Let TextMatrix (ByVal lRow1 As Long, ByVal lCol1 As Long, Optional ByVal lRow2 As Long = -1, Optional ByVal lCol2 As Long = -1, Optional strRowDelimiter As String, Optional strColDelimiter As String, strText As String)	Set the specified cell value. (Example) csv.TextMatrix(1,1) = "a" csv.TextMatrix(1,1,1,2,"",",") = "a,b" csv.TextMatrix(1,1,2,2,":",",") = "a1,b1:a2,b2:a3,b3"
Property Get ValueMatrix (ByVal	Return the specified cell value.

lRow1 As Long, ByVal lCol1 As Long, Optional ByVal lRow2 As Long = -1, Optional ByVal lCol2 As Long = -1) As Variant	(Example) x = csv.ValueMatrix(1,1) y = csv.ValueMatrix(1,1,1,2) z = csv.ValueMatrix(1,1,2,2)
Property Let ValueMatrix (ByVal lRow1 As Long, ByVal lCol1 As Long, Optional ByVal lRow2 As Long = -1, Optional ByVal lCol2 As Long = -1, vntValue As Variant)	Set the specified cell value. (Example) Dim x csv.ValueMatrix(1,1) = x Dim y(2,1) csv.ValueMatrix(1,1,1,2) = y Dim z(2,2) csv.ValueMatrix(1,1,2,2) = z

6.6.1. CSV object

The CSV object provides access function to data sheet managed by CaoScript Manager. The data sheet is saved in CSV file when the project is saved. File name is the same as the project file name with extension ".csv". The main purpose of the CSV file is to save setting parameters. The data sheet is in spreadsheet format. Following is an sample code. This program acquires robot current position for 10 times, and stores them into data sheet.

```
'Acquire current robot position for 10 times, and store them into data sheet.
Sub SaveCurrentAngle

        cao.Controllers.Clear
        set rc = cao.AddController("","CaoProv.DENSO.NetwoRC","","Conn=eth:192.168.0.1")
        set rob = rc.AddRobot("")
        set ang = rob.AddVariable("@CURRENT_ANGLE")

        for k=1 to 10
                 x = dat.ToVar(ang.Value)
                 for i=1 to 6
                          csv.TextMatrix(k,i) = x(i-1)
                 next
        next

End Sub
```

CSV object may be used for inter-task communication just like VARS object, but this is not recommended because of the following reasons.

- Only character string type is supported.
- Processing speed is slow.

Therefore, please use VARS object for inter-task data sharing and synchronization.

6.7. Command line option

Following options are supported as CaoScrMan.exe command line options. More than one option specification is also possible. When using command line option, space separated character string is recognized as argument. Therefore, you need to be careful when the file path for project file name includes spaces.
When command line option is specified, insert spaces between options.

6.7.1. Start option

This option opens the specified project file at startup, and activates task scheduler. Following is the format.

CaoScrMan.exe [< project file name > [start]]

< **project file name** >: Specify the opened project file

< **Start** > : Start task scheduler at startup.

(example 1) >CaoScrMan.exe "d:¥temp¥test.csp"

(example 2) >CaoScrMan.exe "d:¥temp¥test.csp" start

6.7.2. Icon option

By specifying this option, CaoScriptMan is started with window is closed and icon is in the task tray.

CaoScrMan.exe < project file name > icon

< **project file name** >: Specify the opened project file name.

< **Icon** >: **Window** is minimized and put icon in the task tray

(Example 1) >CaoScrMan.exe "d:¥temp¥test.csp" icon

(Example 2) >CaoScrMan.exe "d:¥temp¥test.csp" start icon

7. Sample program

Following is sample programs to use each members of CAO object, Dbg object, VARS object and App object.

List 6-1 Sample.vbs

```
Sub Main
    TestVARS
    TestAPP
    TestCAO
    TestTimer
End Sub

Sub TestVARS
    Vars(10).Value = "VarsTest"    ' Store data to 10th element of VARS object.
    dbg.Output Vars(10).Value      ' Show the stored value on log window.
End Sub

Sub TestAPP
    ext.Delay 1000                 ' Delay Ext object by 1000msec.
End Sub

Sub TestCAO
    ' Create CaoController object from CAO object.
    Set Ctrl = cao.AddController("", "CaoProv.DataStore")

    ' Create CaoVariable object.
    Set Var1 = Ctrl.AddVariable("Var1","")

    vntAry = Array("test1", "test2", 1974)
    Var1.Value = vntAry
    ' Display all array element stored in CaoVariable in space separated format.
    dbg.Output Var1.Name & " : " & Join(Var1.Value)
End Sub

Sub TestTimer
    app.TimerInterval = 100        ' Event interval is set to 100msec.
    ext.Delay 100                  ' Wait event.
End Sub

' Event timer
Sub App_OnTimer
    dbg.Output "Event Test"
End Sub
```

Appendix A. Appendix

Appendix A.1. VBScript reference

◆ Event

Event	Description
Initialize	When the instance of the class is generated, it is generated.
Terminate	When the instance is ended, the class is generated.

◆ Statement

Statement	Description
Call	**It is a flow control statement that passes the control to the Sub procedure and the Function procedure.**
Class	The name of the class is declared.
Const	The constant used instead of the literal value is declared.
Dim	It declares a variable and the memory area is allocated.
Do...Loop	The flow control statement that between truth (True) the specified condition or the conditions repeat a series of statement to truth (True) and execute.
Erase	The memory that initializes a static array element again, and allocates it in a dynamic array is liberated.
Execute	One or more specified statements are executed.
ExecuteGlobal	One or more statements specified by the global name space of the script are executed.
Exit	**It is a flow control statement to slip out Do...Loop, For...Next, the Function procedure or the Sub procedure.**
For...Next	It is a flow control statement that only the specified frequency repeats a series of statement.
For Each...Next	It is a flow control statement that executes each element of the array and the collection repeating a series of statement.
Function	**The name and the argument of the Function procedure are declared, and the start of the Function procedure is shown.**
If...Then...Else	It is a flow control statement that does conditional execution based on the value of the expression.
On Error	Error processing effectively.
Option Explicit	A specified declaration is compelled to all variables in the script.
Private	The private variable is declared, and the storage area is allocated.

Property Get	The name, the argument, and the code that composes the Property procedure to acquire the value of the property are declared.
Property Let	The name, the argument, and the code that composes the Property procedure for which the value of the property is substituted are declared.
Property Set	The name, the argument, and the code that composes the Property procedure that sets the reference to the object are declared.
Public	The public variable is declared, and the storage area is allocated.
Randomize	The random numbers generator is initialized (The random numbers system is set again).
ReDim	The dynamic array variable is declared, and the allocation and the rediscount application in the memory area are done.
Rem	When the comment is described in the program, it specifies it.
Select Case	It is a flow control statement to execute either of two or more statement blocks according to the value of the conditional expression.
Set	The reference to the object is substituted for the variable or the property.
Sub	**The name and the argument of the Sub procedure are declared, and the start of the Sub procedure is shown.**
While...Wend	It is a flow control statement that repeats the execution of a series of statement between truth (True) the specified condition.
With	A series of statement is executed for one object.

◆ Method

Method	Description
Clear	**A set value of all the properties of the Err object is cleared.**
Execute	The specified character string is retrieved by the regular expression.
Raise	When executing it, the error is generated.
Replace	The text found by the retrieval by the regular expression is substituted.
Test	The specified character string is retrieved by the regular expression.

◆ Property

Property	Description
Description	The character string that explains the error related to the error is set.
FirstIndex	The place where the agreement was found in the character string to be retrieved is returned.
Global	The boolean value is set or it returns it.
HelpContext	The context number in which the topic of the Help file is shown is set. The

	acquisition of the value is also possible.
HelpFile	Passing to the Help file is set. The acquisition of the value is also possible.
IgnoreCase	Boolean (Boolean) value in which it is shown whether to distinguish the capital letter and the small letter by the pattern retrieval is set.
Length	The length of a character string corresponding in the character string to be retrieved is returned.
Number	The numerical value that specifies the error is set. The acquisition of the value is also possible.
Pattern	The pattern of the retrieved regular expression is set. The acquisition of the value is also possible.
Source	The name of the object or the application that causes the error is set first. The acquisition of the value is also possible.
Value	A value or a text corresponding in the character string to be retrieved is returned.

◆ Function

Property	Description
Abs	It is a numeric operation function that returns the absolute value of the specified numerical value.
Array	The value of variant type (Variant) where the array is stored is returned.
Asc	It is a transform function in which the ANSI code or the shifted JIS code of the first character in the specified character string is returned.
Atn	It is a numeric operation function to which the arc tangent of the specified angle is returned by the value of double precision floating point number type (Double).
CBool	The transform function of the specified expression that is the variant type (The internal processing form is Variant of Boolean type (Boolean)).
CByte	The transform function of the specified expression that is the variant type (The internal processing form is Variant of byte type (Byte)).
CCur	The transform function of the specified expression that is the variant type (The internal processing form is Variant of currency type (Currency)).
CDate	The transform function of the specified expression that is the variant type (The internal processing form is Variant of date type (Date)).
CDbl	The transform function of the specified expression that is the variant type (The internal processing form is Variant of double precision floating point number type (Double)).

Chr	The character corresponding to specified ANSI code or shifted JIS code is returned.
CInt	The transform function of the specified expression that is the variant type (The internal processing form is Variant of integer type (Integer)).
CLng	The transform function of the specified expression that is the variant type (The internal processing form is Variant of long integer (Long)).
Cos	It is a numeric operation function to which the cosine of the specified angle is returned by the value of double precision floating point number type (Double).
CreateObject	The automation object is made.
CSng	The transform function of the specified expression that is the variant type (The internal processing form is Variant of floating point of single precision type (Single)).
CStr	The transform function of the specified expression that is the variant type (The internal processing form is Variant of character string type (String)).
Date	The date of a present system is returned.
DateAdd	The date when the specified interval time was added is returned.
DateDiff	Two specified interval time of the date is returned.
DatePart	A specified part at the date is returned.
DateSerial	The date corresponding to the age and Monday and Sunday specified for the argument is returned by the value of the variant type (The internal processing form is Variant of date type (Date)).
DateValue	The specified date is returned by the value of the variant type (The internal processing form is Variant of date type (Date)).
Day	The integer within the range of 1-31 that shows several days of the moon is returned.
Eval	The expression is evaluated, and the result is returned.
Exp	It is a numeric operation function that calculates exponential (involution of the expression that makes e a bottom).
Filter	The array of zero base including the subset in the string array based on the specified filter condition is returned.
Fix	It is a numeric operation function that rounds down the fraction part of the expression specified for the argument, and returns only the integer part.
FormatCurrency	It is a character string processing function that sets the format to the character string of the currency form by using the currency sign defined by the control panel of the system and returns it.

FormatDateTime	It is a character string processing function that sets the format to the character string of the form of the date form or time and returns it.
FormatNumber	It is a character string processing function that sets the format to the character string of a numeric form and returns it.
FormatPercent	It is a character string processing function that sets the format to the character string of percent form (Multiplied it by 100) to which percent sign (%) was added and returns it.
GetLocale	The value of present locale ID is returned.
GetObject	The reference to the automation object acquired from the file is returned.
GetRef	The reference to the procedure that can do the event and bind is returned.
Hex	It returns it by the character string where the specified value is shown by the hexadecimal number.
Hour	1 The integer within the range of 0-23 that shows time of the day is returned.
InputBox	When the message and the text box are displayed in the dialog box, and the text is input or the button is clicked, the content of the text box is returned.
InStr	It is a character string processing function that retrieves specified character string (string2), and returns the character position (number of characters from the head to the position) found first from among a certain character string (string1).
InStrRev	The character string processing function of specified character string (string2) from from among a certain character string (string1) from which the retrieval begins from the last character position, and the character position (number of characters from the head to the position) found first is returned.
Int	It is a numeric operation function that rounds down the fraction part of the expression specified for the argument, and returns only the integer part.
IsArray	It is examined whether the variable is an array, and returns the result by the boolean value.
IsDate	It is examined whether the expression can be converted at the date, and returns the result by the boolean value.
IsEmpty	It is examined whether the variable was initialized. The result is returned by the boolean value.
IsNull	It is examined whether the Null value is included in the expression, and returns the result by the boolean value.
IsNumeric	It is examined to be appreciable of the expression as the numerical value, and returns the result by the boolean value.

IsObject	It is examined whether the expression refers to the automation object, and returns the result by the boolean value.
Join	The character string that unites internal character strings of each element included in the array and is made is returned.
LBound	The minimum value of the index number that can be used in the dimension for which the array is specified is returned.
LCase	It is a character string processing function that converts the capital letter of the alphabet into the small letter.
Left	The character string of a few minutes of the character specified from the left end of the character string is returned.
Len	The number of characters of specified character strings is returned.
LoadPicture	The picture object is returned. It is possible to use it only in 32 bit version platform.
Log	It is a numeric operation function that returns Napierian logarithm of the numerical value.
LTrim	The copy of the character string that space is not at the head is returned.
Mid	The character string of a few minutes of the character specified from the character string is returned.
Minute	The integer within the range of 0-59 that shows the amount at time is returned.
Month	The integer within the range of 0-12 shown whether it is in what month of one year is returned.
MonthName	The character string that shows the specified moon is returned.
MsgBox	The message is displayed in the dialog box, it is waited that the button is clicked, and returns the value in which which button was clicked is shown.
Now	A present date and time are returned based on the date of the system of the computer and the setting at time.
Oct	The specified value is returned by the octal number.
Replace	The character string for which a part of the specified character string is substituted by the frequency specified by another character string is returned.
RGB	The value in which RGB color value is shown is returned.
Right	The character string of a few minutes of the character specified from a right edge of the character string is returned.
Rnd function	Random numbers of single precision floating point number type (Single) are returned.
Round	The numerical value rounded at the specified decimal point position is returned.

RTrim	The copy of the character string that space is not in the end is returned.
ScriptEngine	The character string that shows the script language under use is returned.
ScriptEngineBuildVersion	The [birudoba-jon] number of the script engine under use is returned.
ScriptEngineMajorVersion	The major version number of the script engine under use is returned.
ScriptEngineMinorVersion	The [mainaba-jon] number of the script engine under use is returned.
Second	The integer of 0-59 that shows the second of time is returned.
Sgn	The sign of the expression specified for the argument is returned.
Sin	The signature of the specified angle is returned.
Space	The character string that consists of the space of the specified number is returned.
Split	One dimension of zero base array is made from the character string delimited to each element, and it returns it.
Sqr	The square root of the expression is returned.
StrComp	The value in which the result of the character-string comparison is shown is returned.
String	The character string to which only the specified number of characters arranged the heading character of the character or the character string that the specified character-code (ASCII or shifted JIS code) shows is returned.
StrReverse	The character string that reverses the order of the row of the character of the specified character string is returned.
Tan	The tangent of the specified angle is returned.
Time	The time of a present system is returned.
Timer	The number of seconds that passed after 0:00 the morning is returned.
TimeSerial	Time corresponding to the amount and the second is returned in specification for the argument.
TimeValue	Time is returned.
Trim	The copy of the character string that space is not in the head or the end is returned.
TypeName	The character string that offers information on the specified variable is returned.
UBound	The maximum value of the index number that can be used in the dimension specified by the array is returned.
UCase	The small letter of the specified alphabet is converted into the capital letter.
VarType	The value in which the internal processing form of the variable is shown is returned.

Weekday	(Sunday) 1 that shows what day of the week it is -7 The value within the range of (Saturday) is returned.
WeekdayName	The character string that shows a specified day of the week is returned.
Year	The integer in which the age is shown is returned.

◆ Object and collection

Method	Description
Class	The means to access the event of the class is offered.
Err	**When executing it, the Err object has information on the error.**
Match	The means to access the property only for reading a character string corresponding by the regular expression is offered.
Matches	It is a collection of the Match object of the regular expression.
RegExp	The function of the regular expression is offered.
SubMatches	It is a collection of the submatch character string of the regular expression.