
TECHNICAL SPECIFICATION

isystem.connect Automation Quick Start

Introduction

The winIDEA *isystem.connect* automation server is designed to provide access to winIDEA functionality from an assortment of different programming languages and technologies.

The automation server is implemented in C++ and ATL (advanced template library), it is small and fast with no unnecessary overhead. It has been tested to work from Visual Basic, ASP pages, VBScript, Jscript and C#. Other languages and technologies supporting automation and/or COM should work without problems. The server implements a dual interface; this allows flexibility and fast code.

The automation server exposes the functionality provided by the winIDEA *isystem.connect* automation server. A detailed reference of the API is provided in a separate document.

Organization

The server interface is divided in two different sets of methods: hard-type output parameters and soft-type output parameters.

The hard-type output parameter set of functions uses hard automation types for input and output parameters, providing a safer type checking environment. It is better suited for languages where types can or must be specified: C++, C#, Visual Basic, etc.

The soft-type output parameter set of functions uses variants to accommodate output parameters. Functions in this set are identified by the prefix "Script" pre-pended to each function name. This allows typeless languages to interact properly with the server. It is recommended for: VBScript, ASP pages, Jscript, etc.

Functions are divided in four different functional groups: Connect, IDE, Debug, Profiler.

The Connect group of functions includes functions to connect and disconnect from winIDEA.

The IDE group of functions include functions to perform IDE operations like creating a document, saving a workspace with a different name or repositioning the winIDEA main window.

The Debug group of functions allows tracing the program and accessing the target. Functionality includes access to the target's memory and registers, breakpoint setting and clearing, program stepping, expression evaluation, etc.

For detailed information consult the winIDEA *isystem.connect* automation server reference.

Example 1: reading the value of variables

This example in VBScript reads the value of the given variable a few times:

Source code

```
` Makes a shortcut to the standard output stream
Dim StdOut
Set StdOut = WScript.StdOut

` Get the input arguments
Dim ObjArgs
Set ObjArgs = WScript.Arguments

` Check the command line
If ObjArgs.Count < 1 Then
    StdOut.WriteLine "Usage:"
    StdOut.WriteLine "DebugEvaluate.vbs <symbol> [<symbol> [...]]"
    WScript.Quit(1)
End If

` Create the automation server object
Set iConnect = CreateObject("WinIdeaAuto.WinIdeaAutoSvr")
iConnect.Connect

` Declare and initialize the output parameter. Initialization is optional.
Dim strResult
strResult = ""

` Fore each passed parameter, gets its address and print the info
For I = 0 To ObjArgs.Count - 1
    iConnect.ScriptDebugEvaluateString 0, CStr(ObjArgs(I)), strResult
    StdOut.WriteLine "The expression '" & ObjArgs(I) & "' has the value '" & strResult & "'."
Next

` Disconnect from winIDEA
iConnect.Disconnect
```

Running the program

To run the example use the following command line:

```
cscript DebugEvaluate.vbs <arg1> <arg2> ... <argN>
```

The program output would be:

```
D:\WinIdeaAutoSamples\VBScript>cscript DebugEvaluate.vbs iCounter  
Microsoft (R) Windows Script Host Version 5.6  
Copyright (C) Microsoft Corporation 1996-2001. All rights reserved.  
The expresion 'iCounter' has the value '0x00000015'
```

Example 2: getting the address of symbols

This example in VBScript retrieves the address of one or more symbols on the target's memory:

Source code

```
' Makes a shortcut to the standard output stream
Dim StdOut
Set StdOut = WScript.StdOut

' Get the input arguments
Dim ObjArgs
Set ObjArgs = WScript.Arguments

' Check the command line
If ObjArgs.Count < 1 Then
    StdOut.WriteLine "Usage:"
    StdOut.WriteLine "DebugGetAddress.vbs <symbol> [<symbol> [...]]"
    WScript.Quit(1)
End If

' Create the automation server object
Set iConnect = CreateObject("WinIdeaAuto.WinIdeaAutoSvr")
iConnect.Connect

' Declare and initialize the output parameters. Initialization is optional.
Dim MemArea
MemArea = 0
Dim Address
Address = 0
Dim SizeMau
SizeMau = 0
Dim ExpType
ExpType = 0

' Fore each passed parameter, gets its address and print the info
For I = 0 To ObjArgs.Count - 1
    iConnect.ScriptDebugGetAddress 0, CStr(ObjArgs(I)), MemArea, Address, SizeMau, ExpType
    StdOut.WriteLine "The expression '" & ObjArgs(I) & "' is located in the memory area " & MemArea & " at the address " & Address & ", has a size of " & SizeMau & " memory allocation units and is of the type 0x" & Hex(ExpType) & "."
Next
```

```
` Disconnect from winIDEA
```

```
iConnect.Disconnect
```

Running the program

To run the example use the following command line:

```
cscript DebugGetAddress.vbs <arg1> <arg2> ... <argN>
```

The program output would be:

```
D:\WinIdeaAutoSamples\VBScript>cscript DebugGetAddress.vbs main uCounter
```

```
Microsoft (R) Windows Script Host Version 5.6
```

```
Copyright (C) Microsoft Corporation 1996-2001. All rights reserved.
```

```
The expression 'main' is located in the memory area 0 at the address 67109996, has a size of 336 memory allocation units and is of the type 0x0.
```

```
The expression 'uCounter' is located in the memory area 0 at the address 68157504, has a size of 4 memory allocation units and is of the type 0x2001.
```

Example 3: reading the value of an expression into Microsoft Excel

This example reads the value of an expression a specified number of times, fills an excel table and makes a graph.

Source code

```
Dim StdOut
Set StdOut = WScript.StdOut

Dim ObjArgs
Set ObjArgs = WScript.Arguments

If ObjArgs.Count <> 3 Then
    StdOut.WriteLine "Usage:"
    StdOut.WriteLine "DemoExcel <symbol> <count> <interval>"
    WScript.Quit(1)
End If

Dim SymbolName
SymbolName = CStr(ObjArgs(0))

Dim AccessCount
AccessCount = CLng(ObjArgs(1))

Dim WaitTime
WaitTime = CLng(ObjArgs(2))

Dim oExcelApp
Dim oWorkBook
Dim oWorkSheet
Dim oChart

Set oExcelApp = CreateObject("Excel.Application")
oExcelApp.Visible = True
Set oWorkBook = oExcelApp.Workbooks.Add
Set oWorkSheet = oWorkBook.ActiveSheet

Set iConnect = CreateObject("WinIdeaAuto.WinIdeaAutoSvr")
iConnect.Connect

For I = 1 to AccessCount
    Dim SymbolValue
```

```

SymbolValue = 0
iConnect.ScriptDebugEvaluateVariant (1 + 0 + 0), SymbolName, SymbolValue
oWorkSheet.Cells(I, 1).Value = "T" & I
oWorkSheet.Cells(I, 2).Value = SymbolValue
WScript.Sleep(WaitTime)

```

Next

```

iConnect.Disconnect

oExcelApp.Range("B" & 1, "B" & AccessCount).Select
Set oChart = oExcelApp.Charts.Add

```

Running the program

To run the example use the following command line:

```
cscript DemoExcelSimple.vbs <symbol> <count> <interval>
```

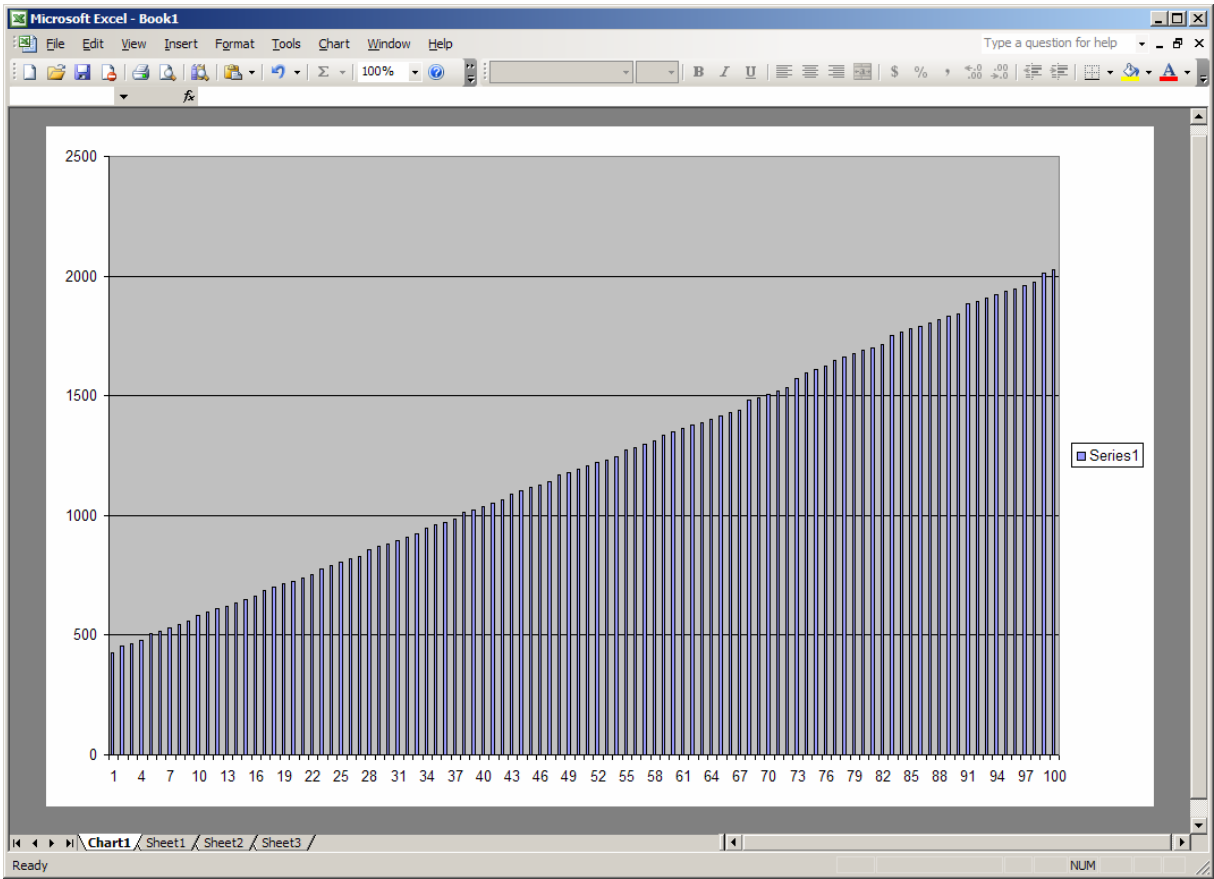
The program output would be:

```
D:\WinIdeaAutoSamples\VBScript>cscript DemoExcelSimple.vbs iCounter 100 100
```

Microsoft (R) Windows Script Host Version 5.6

Copyright (C) Microsoft Corporation 1996-2001. All rights reserved.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	T1	427														
2	T2	453														
3	T3	466														
4	T4	479														
5	T5	505														
6	T6	518														
7	T7	531														
8	T8	544														
9	T9	557														
10	T10	583														
11	T11	596														
12	T12	609														
13	T13	622														
14	T14	635														
15	T15	648														
16	T16	661														
17	T17	687														
18	T18	700														
19	T19	713														
20	T20	726														
21	T21	739														
22	T22	752														
23	T23	778														
24	T24	791														
25	T25	804														
26	T26	817														
27	T27	830														
28	T28	856														
29	T29	869														
30	T30	882														
31	T31	895														
32	T32	908														
33	T33	921														
34	T34	947														
35	T35	960														
36	T36	973														
37	T37	986														



Epiloge

As is seen in the examples above, the general structure of a program using the winIDEA *isystem.connect* automation server will consist of: the creation of the automation server object, connecting to winIDEA, operating with the interface functions and finally disconnecting.

Additional examples in Visual Basic, VBScript, ASP pages, Jscript and C# are provided in the winIDEA distribution media.

Notes: