User's Manual

RENESAS

HewTargetServer

User's Manual

All information contained in these materials, including products and product specifications, represents information on the product at the time of publication and is subject to change by Renesas Electronics Corporation without notice. Please review the latest information published by Renesas Electronics Corporation through various means, including the Renesas Electronics Corporation website (http://www.renesas.com).

Renesas Electronics
www.renesas.com

Rev.8.00   Nov 2010

Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations.
6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
7. Renesas Electronics products are classified according to the following three quality grades: “Standard”, “High Quality”, and “Specific”. The recommended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as “Specific” without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as “Specific” or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is “Standard” unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.

   “Standard”: Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots.

   “High Quality”: Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.

   “Specific”: Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

Abstract

HEW Target Server (COM) is provided for the purpose of extending the functions of the High-performance Embedded Workshop. Using Windows application development tools available on the market, you can customize the High-performance Embedded Workshop and operate in conjunction with other applications.

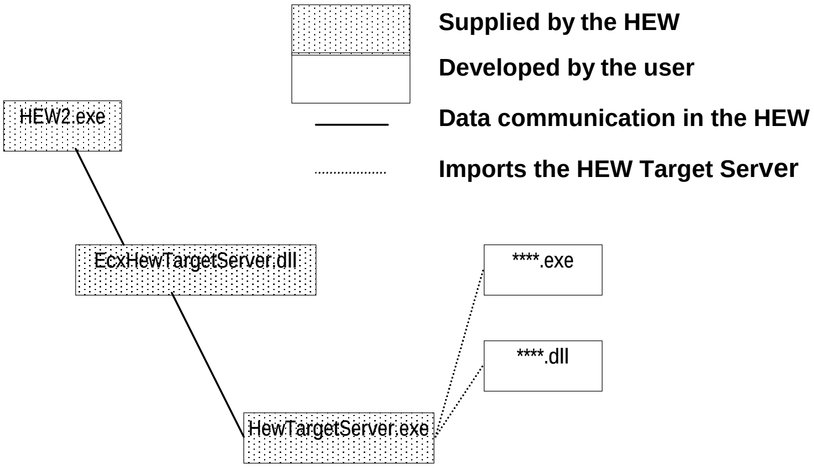
This user's manual shows the basic information necessary to use the HEW Target Server (COM). For details about the language specifications of and the method for using Windows application development tools, refer to the user's manual included with your product or online help.

Trademarks

[Contents]

1.Abstract

HEW Target Server (COM) provides the interface to extend the functions of the High-performance Embedded Workshop. Using this interface, you can create the customize window (application) for the High-performance Embedded Workshop, and operate in conjunction with other applications.

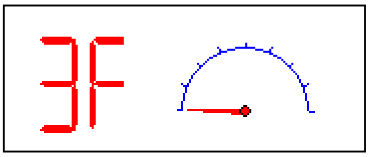


1.1 Development Tools Used

To create customize windows or operate in conjunction with other applications, you need to use Windows application development tools which support Microsoft's Visual Basic or Visual C++ or other COM.

- **Many reference books are available on the market, as is the information necessary to create applications.**

- **The kit comes standard with abundant GUI components. These GUI components can be used as simulate components for the user system. Freeware and shareware control components (ActiveX control) can also be used. Or you can create your original components using Visual Basic or Visual C++.**



1.2 Methods To Be Called

Various methods can be called through the HewTargetServer's COM interface, including those to control execution of the microcomputer, set/reference memory or register contents, and set software breakpoints.

2.Preparing for Use of the HEW Target Server (COM)

To use the HEW Target Server (COM) to work in cooperation with external applications, you must first enable it in the High-performance Embedded Workshop environment you are using.

This chapter explains how to register and enable the HEW Target Server (COM) functions.

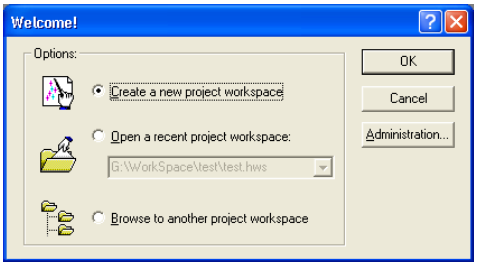
2.1 Registering the HEW Target Server (COM)

In the initial state of the High-performance Embedded Workshop or Renesas' integrated development environment, that is installed in your computer, the facilities necessary to use the HEW Target Server (COM) functions, i.e., HewTargetServer.exe and EcxHewTargetServer.dll, are not registered yet. Therefore, even when you launch an external application you've created by using the HEW Target Server (COM) you cannot control the High-performance Embedded Workshop with it.

The following describes how to register HewTargetServer.exe in your Windows registry and how to register EcxHewTargetServer.dll.

2.1.1 Registering EcxHewTargetServer.dll

1. **Launch the High-performance Embedded Workshop, and the "Welcome" dialog box shown below will appear. When this dialog box is displayed, click the Administration button in it.**



Welcome!

Options:

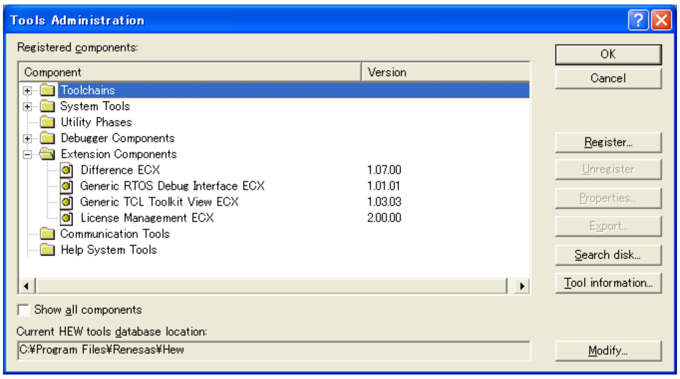
● Create a new project workspace

○ Open a recent project workspace:

G:\WorkSpace\test\test.hws

○ Browse to another project workspace

2. **The Tool Administration dialog box shown below will appear. In its registered component list, select the Extension Components folder to open. In the initial state, you will see that HewTargetServer is not registered. Next, click the Search Disk button.**



Tools Administration

Registered components:

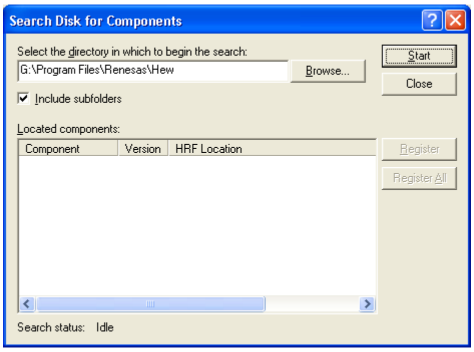
Component	Version
Toolchains	
System Tools	
Utility Phases	
Debugger Components	
Extension Components	
Difference ECX	1.07.00
Generic RTOS Debug Interface ECX	1.01.01
Generic TCL Toolkit View ECX	1.03.03
License Management ECX	2.00.00
Communication Tools	
Help System Tools	

□ Show all components

Current HEW tools database location:

C:¥Program Files¥Renesas¥Hew

3. **When the Search Component Disk dialog box shown below is displayed, select the folder in which the High-performance Embedded Workshop, or Renesas' integrated development environment, is installed and click the Start button. The components that are installed in your computer will be listed.**



Search Disk for Components

Select the directory in which to begin the search:

G:\Program Files\Renesas\Hew

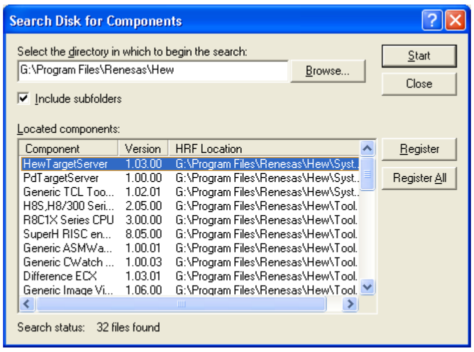
☑ Include subfolders

Located components:

Component	Version	HRF Location

Search status: Idle

4. **Select HewTargetServer from the listed components and click the Register button.**



Search Disk for Components
Select the directory in which to begin the search:
G:\Program Files\Renesas\Hew
Browse...
Start
Close
☑ Include subfolders
Located components:
Register
Register All

Component	Version	HRF Location
HewTargetServer	1.03.00	G:\Program Files\Renesas\Hew\Syst...
PdTargetServer	1.00.00	G:\Program Files\Renesas\Hew\Syst...
Generic TCL Too...	1.02.01	G:\Program Files\Renesas\Hew\Syst...
H8S,H8/300 Seri...	2.05.00	G:\Program Files\Renesas\Hew\Tool.
R8C1X Series CPU	3.00.00	G:\Program Files\Renesas\Hew\Tool.
SuperH RISC en...	8.05.00	G:\Program Files\Renesas\Hew\Tool.
Generic ASMWa...	1.00.01	G:\Program Files\Renesas\Hew\Tool.
Generic CWatch ...	1.00.03	G:\Program Files\Renesas\Hew\Tool.
Difference ECX	1.03.01	G:\Program Files\Renesas\Hew\Tool.
Generic Image Vi...	1.06.00	G:\Program Files\Renesas\Hew\Tool.

Search status: 32 files found

Registration of EcxHewTargetServer.dll is completed. Close the dialog boxes sequentially.

Note: If EcxHewTargetServer.dll becomes unnecessary after you registered it, be sure to unregister it.

2.1.2 Registering HewTargetServer.exe in Your Registry

From High-performance Embedded Workshop V.4.05, the installer automatically registers or removes HewTargetServer.exe. You do not need to register or remove HewTargetServer.exe manually.
Install Manager Version 1.03 or later also automatically registers or removes HewTargetServer.exe as required in switching to another High-performance Embedded Workshop.

However, if you have installed multiple High-performance Embedded Workshop by Install Manager and intend to use Install Manager Version 1.02 or earlier to switch the active High-performance Embedded Workshop, you should remove and register HewTargetSever.exe manually.

To remove and register HewTargetSever.exe:
Double-click on the ALL_UNREGISTERSERVER.bat file stored in the folder where you have installed the currently active High-performance Embedded Workshop. HewTargetServer.exe will be removed from the registry.
Then find the ALL_REGISTERSERVER.bat file stored in the folder in which you have installed the High-performance Embedded Workshop that you wish to activate and double-click on it to execute. HewTargetServer.exe will be registered in your registry.

If you are running the High-performance Embedded Workshop under Windows Vista, invoke the command prompt as an administrator and enter the cd command to go to the folder where ALL_REGISTERSERVER.bat/ ALL_UNREGISTERSERVER.bat is stored. Then execute the batch file.
(To invoke the command prompt as an administrator, select [All Programs -> Accessories] from the Start menu, right-click on [Command Prompt], and select [Run as administrator].)

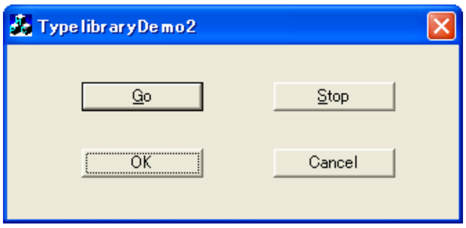
3.Using the HEW Target Server (COM)

This chapter describes how to use the HEW Target Server (COM) that is supplied for connection with external applications. For details about the methods of the HEW Target Server (COM), refer to Section 5.2, "Details of Methods. "

In this chapter specifically, you will learn how to create a customized window of the High-performance Embedded Workshop by using Visual C++6.0. The explanation here uses the simple sample window shown below as an example.

3.1 Sample Program

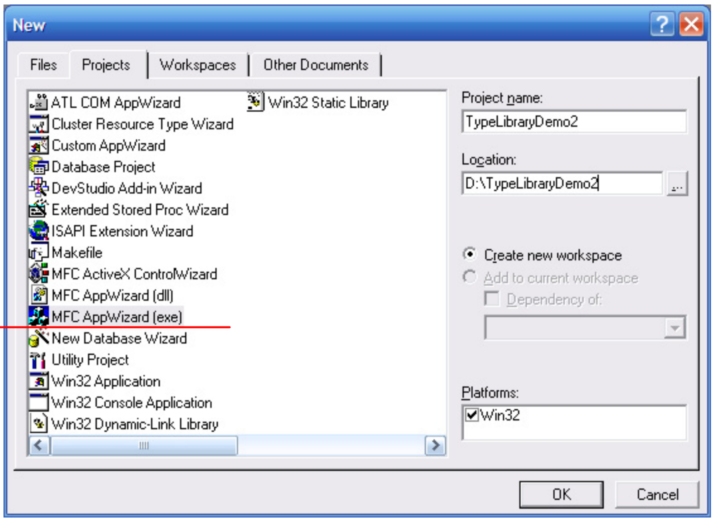
This sample program creates a window that when simulating the operation of a program created with the High-performance Embedded Workshop you can use to start or stop execution of simulation from an external window.



3.2 Creating a Program (Visual C++)

3.2.1 Generating a Project

Generate a new project with Visual C++. Select New from the File menu of Visual C++. The New wizard will start. In this wizard, select MFC AppWizard (exe) and click the OK button.



Project name: TypeLibraryDemo2

Location: D:\TypeLibraryDemo2

● Create new workspace

○ Add to current workspace

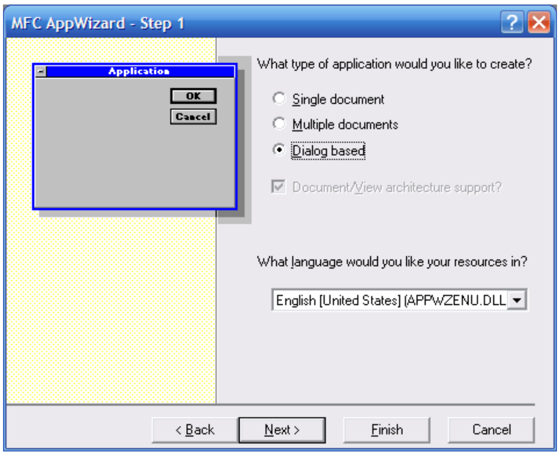
□ Dependency of:

Platforms: ☑ Win32

Select MFC AppWizard (exe)

The project name in this example is TypeLibraryDemo2.

In Step 1 of MFC AppWizard, specify the type of application you want to create. In the example here we'll create a "Dialog based" application. So select it and click the Next button.



MFC AppWizard - Step 1

What type of application would you like to create?

○ Single document
○ Multiple documents
● Dialog based

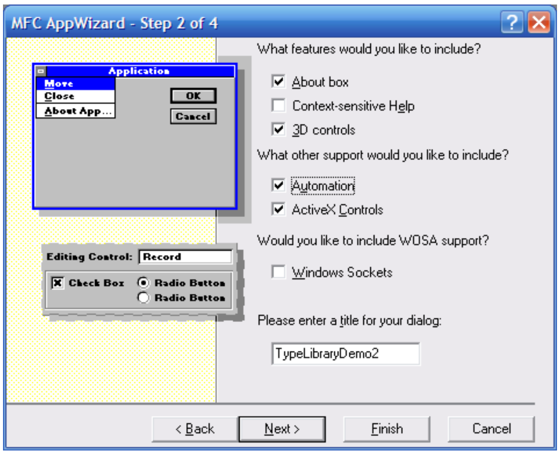
☑ Document/View architecture support?

What language would you like your resources in?

English [United States] (APPWZENU.DLL

< Back | Next > | Finish | Cancel

In Step 2, select the check box titled "Automation. " Leave other options as set by default.



MFC AppWizard - Step 2 of 4

What features would you like to include?

☑ About box
□ Context-sensitive Help
☑ 3D controls

What other support would you like to include?

☑ Automation
☑ ActiveX Controls

Would you like to include WOSA support?

□ Windows Sockets

Please enter a title for your dialog:

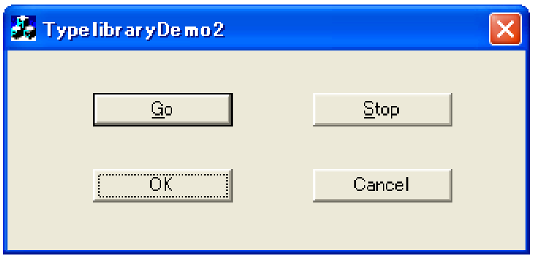
TypeLibraryDemo2

< Back | Next > | Finish | Cancel

In the steps that follow, you can proceed with default settings without causing any problem.

3.2.2 Creating Buttons

When you finished creating a project, create buttons in a dialog box. The IDs and captions set for each button you created are shown below.



ID	Caption	Notes
IDC_BUTTON_GO	&Go	-
IDC_BUTTON_STOP	&Stop	-
IDOK	OK	Default
IDCANCEL	Cancel	Default

3.2.3 Creating Source Code

Next, add statements to the source code that was generated when you created a project and use the HEW Target Server (COM).

(1) Import HewTargetServer.exe.

File to correct: TypelibraryDemo2Dlg.h

```
//import type library
#import "..\Hew2\HewTargetServer.exe"                no_namespace
```

The path to HewTargetServer.exe specified here differs with each environment used. Specify the folder in which the compiler package is installed.

(2) Declare a smart pointer as a member variable.

File to correct: TypelibraryDemo2Dlg.h

```
class CTypelibraryDemo2Dlg : public CDialog
{
...
public:
        //declare smart pointer
        IHewServer1Ptr    pHewServer1;

};
```

(3) Create and initialize the smart pointer by a constructor.

File to correct: TypelibraryDemo2Dlg.cpp

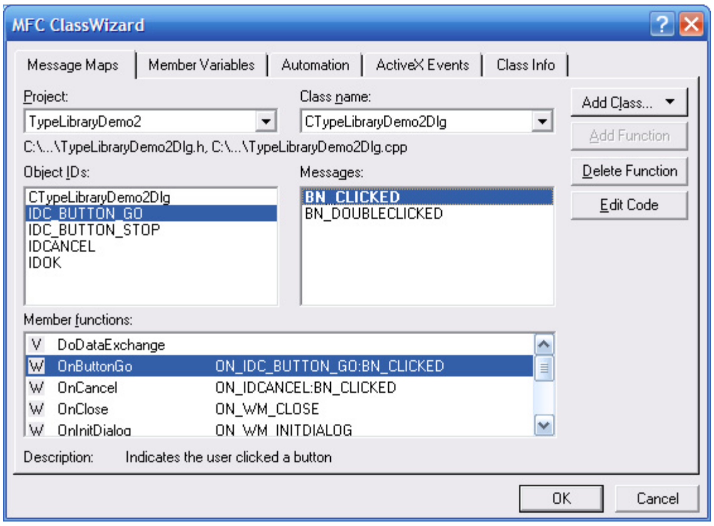
```
// TODO: Add to this place when special initialization is desired.
// TODO: Add extra initialization here
try{
        //create smart pointer
        IHewServer1Ptr ptr(_uuidof(HewServer1));
        pHewServer1 = ptr;
}
```

(4) To ensure that the smart pointer will be discarded when a client terminates, add the statement shown below.

File to correct: TypelibraryDemo2Dlg.cpp

```
CTest1Dlg::~CTest1Dlg()
{
        // If there is an automation proxy for this dialog,
        // the pointer to this dialog is returned to NULL,
        // which indicates that the dialog has been deleted.
        if (m_pAutoProxy != NULL)
                m_pAutoProxy->m_pDialog = NULL;
        //destroy smart pointer
        pHewServer1 = NULL;

}
...
void CTypelibraryDemo2Dlg::OnClose()
{
        if (CanExit())
                CDialog::OnClose();

        //destroy smart pointer
        pHewServer1 = NULL;
}
...
void CTypelibraryDemo2Dlg::OnOK()
{
        if (CanExit())
                CDialog::OnOK();

        //destroy smart pointer
        pHewServer1 = NULL;
}
...
void CTypelibraryDemo2Dlg::OnCancel()
{
        if (CanExit())
                CDialog::OnCancel();

        //destroy smart pointer
        pHewServer1 = NULL;
}
```

(5) Next, add a function in ClassWizard that you want to be called when a button is clicked.



Project: TypeLibraryDemo2

Class name: CTypeLibraryDemo2Dlg

C:\...\TypeLibraryDemo2Dlg.h, C:\...\TypeLibraryDemo2Dlg.cpp

Object IDs:
CTypeLibraryDemo2Dlg
IDC_BUTTON_GO
IDC_BUTTON_STOP
IDCANCEL
IDOK

Messages:
BN_CLICKED
BN_DOUBLECLICKED

Member functions:

```
V  DoDataExchange
W  OnButtonGo      ON_IDC_BUTTON_GO:BN_CLICKED
W  OnCancel        ON_IDCANCEL:BN_CLICKED
W  OnClose         ON_WM_CLOSE
W  OnInitDialog    ON_WM_INITDIALOG
```

Description: Indicates the user clicked a button

Select a button object and message and click the Add Function button. A null function like the one shown below will be inserted into TypelibraryDemo2Dlg.cpp.

```
void CTypelibraryDemo2Dlg::OnButtonGo()
{
        // TODO: Add code for the control notification handler at this position.

}
```

Precautions:
To call the functions published for the High-performance Embedded Workshop, you must always use try{} and/or catch{}. If an error occurs in an interface function call, you can use catch{} to get a COM error from the COM system. If a COM error is issued when not using catch{}, the client program will cause an application error to occur.
There are following three types of COM errors.

Custom error (errors issued by HewTargetServer.exe)
This error is included in the error returned by HewTargetServer.exe when it is invoked while the High-performance Embedded Workshop is inactive, no targets are connected, or no load modules are downloaded.

HEW error (errors issued by HEW2.exe)
If the High-performance Embedded Workshop returns an error, it is possible that some parameter of the called interface is invalid. When the High-performance Embedded Workshop returned an error you can call GetErrorString() to get the content of the error.

System error (errors issued by the COM system)
If an error is returned by the COM system, it means that the RPC (Remote Procedure Call) environment has a problem or the communication between the client and HewTargetServer.exe has a problem.

(6) Create the OnButtonGo() function

File to correct: TypelibraryDemo2Dlg.cpp

```
void CTypelibraryDemo2Dlg::OnButtonGo()
{

        HRESULT         hr = E_FAIL, hrErr = E_FAIL;
        CString                 s1;

        //calling HewTargetServer function
        try
        {
                hr = pHewServer1->GoTargetExec(); //   Write the method for executing a program.

        }
        catch(_com_error e1)
        {
                if(e1.Description().length()>0)         //display custom COM error
                        AfxMessageBox(e1.Description());
                else
                {

                        BSTR    bstrErrStr;

                        try
                        {
                                hrErr = pHewServer1->GetErrorString(e1.Error(), &bstrErrStr);
                                // Write the method for getting the content of an error.
                        }
                        catch(_com_error e)
                        {
                        }
                        if(SUCCEEDED(hrErr))
                        {
                                s1.Format("%s", CString(bstrErrStr));
                                AfxMessageBox(s1);
                        }
                        else{   //display system error
                                AfxMessageBox(e1.ErrorMessage());
                        }
                }
        }
}
```

(7) Create the OnButtonStop() function

File to correct: TypelibraryDemo2Dlg.cpp

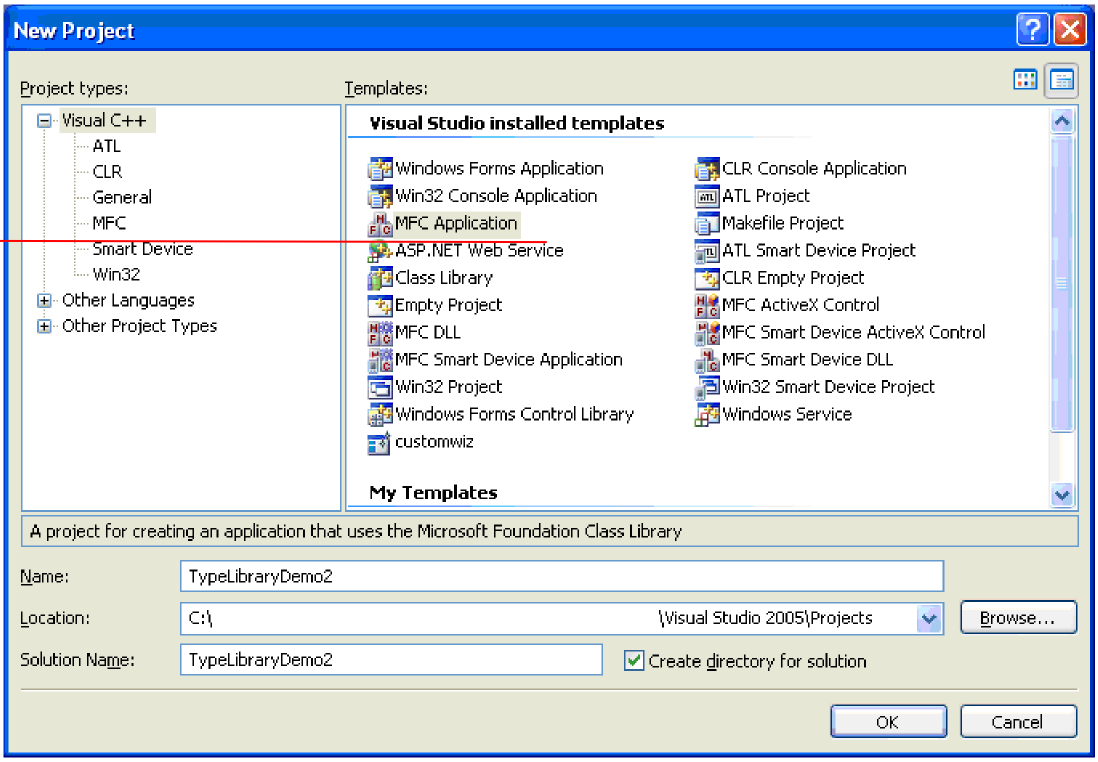
```
void CTypelibraryDemo2Dlg::OnButtonStop()
{

    HRESULT     hr = E_FAIL, hrErr = E_FAIL;
    CString         s1;

    //calling HewTargetServer function
    try
    {
        hr = pHewServer1->StopTargetExec(); // Add the method for stopping program execution.
    }
    catch(_com_error e1)
    {
        if(e1.Description().length()>0)        //display custom COM error
            AfxMessageBox(e1.Description());
        else
        {

            BSTR    bstrErrStr;

            try
            {
                hrErr = pHewServer1->GetErrorString(e1.Error(), &bstrErrStr);
                // Write the method for getting the content of an error.
            }
            catch(_com_error e)
            {
            }
            if(SUCCEEDED(hrErr))
            {
                s1.Format("%s", CString(bstrErrStr));
                AfxMessageBox(s1);
            }
            else
            {   //display system error
                AfxMessageBox(e1.ErrorMessage());
            }
        }
    }
}
```

3.3 Creating a Program (Visual C++ 2005)

3.3.1 Generating a Project

Generate a new project with Visual C++. Select [New]->[Project...] from the File menu of Visual C++. The New Project wizard will start. In this wizard, select MFC Application and click the OK button.

Select:
MFC Application


New Project

Project types:
- Visual C++
  - ATL
  - CLR
  - General
  - MFC
  - Smart Device
  - Win32
- Other Languages
- Other Project Types

Templates:

Visual Studio installed templates

Windows Forms Application	CLR Console Application
Win32 Console Application	ATL Project
MFC Application	Makefile Project
ASP.NET Web Service	ATL Smart Device Project
Class Library	CLR Empty Project
Empty Project	MFC ActiveX Control
MFC DLL	MFC Smart Device ActiveX Control
MFC Smart Device Application	MFC Smart Device DLL
Win32 Project	Win32 Smart Device Project
Windows Forms Control Library	Windows Service
customwiz	

My Templates

A project for creating an application that uses the Microsoft Foundation Class Library

Name: TypeLibraryDemo2

Location: C:\ \Visual Studio 2005\Projects

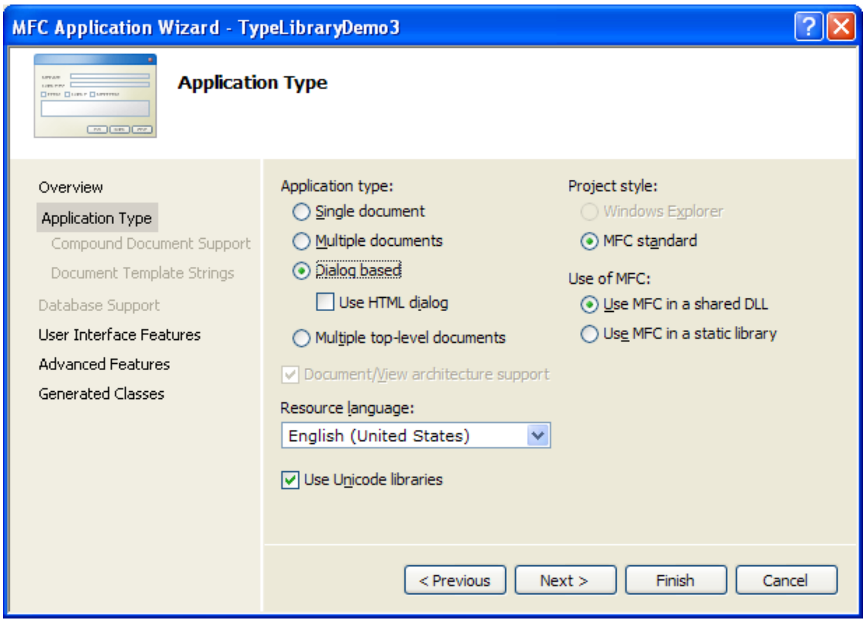
Solution Name: TypeLibraryDemo2

☑ Create directory for solution

The project name in this example is TypeLibraryDemo2.

In the "Welcome to the MFC Application Wizard", click the Next button.

In the "Application Type", select "Dialog based" and click the Next button.


MFC Application Wizard - TypeLibraryDemo3

Application Type

Application type:
○ Single document
○ Multiple documents
● Dialog based
□ Use HTML dialog
○ Multiple top-level documents

☑ Document/View architecture support

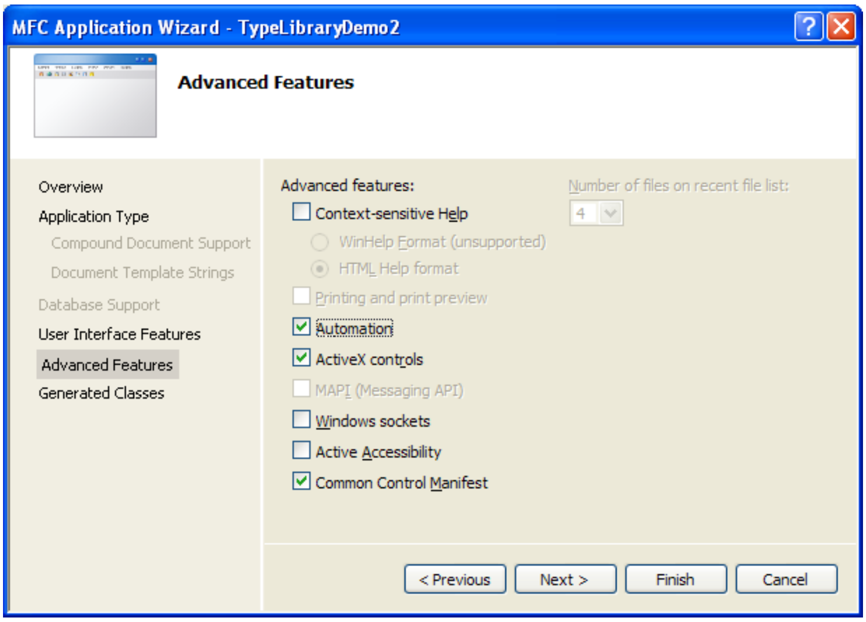
Resource language:
English (United States)

☑ Use Unicode libraries

Project style:
○ Windows Explorer
● MFC standard

Use of MFC:
● Use MFC in a shared DLL
○ Use MFC in a static library

In the "Advanced Features", select the check box titled "Automation. " Leave other options as set by default.


MFC Application Wizard - TypeLibraryDemo2

Advanced Features

Advanced features:

□ Context-sensitive Help

○ WinHelp Format (unsupported)

● HTML Help format

□ Printing and print preview

☑ Automation

☑ ActiveX controls

□ MAPI (Messaging API)

□ Windows sockets

□ Active Accessibility

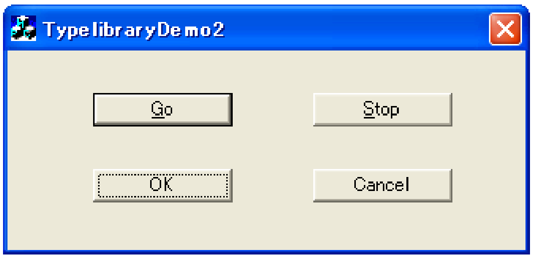
☑ Common Control Manifest

Number of files on recent file list: 4

In the steps that follow, you can proceed with default settings without causing any problem.

3.3.2 Creating Buttons

When you finished creating a project, create buttons in a dialog box. The IDs and captions set for each button you created are shown below.



ID	Caption	Notes
IDC_BUTTON_GO	&Go	-
IDC_BUTTON_STOP	&Stop	-
IDOK	OK	Default
IDCANCEL	Cancel	Default

3.3.3 Creating Source Code

Next, add statements to the source code that was generated when you created a project and use the HEW Target Server (COM).

(1) Import HewTargetServer.exe.

File to correct: TypelibraryDemo2Dlg.h

```
//import type library
#import "..\Hew2\HewTargetServer.exe"                no_namespace
```

The path to HewTargetServer.exe specified here differs with each environment used.
Specify the folder in which the compiler package is installed.

(2) Declare a smart pointer as a member variable.

File to correct: TypelibraryDemo2Dlg.h

```
class CTypelibraryDemo2Dlg : public CDialog
{
...
public:
        //declare smart pointer
        IHewServer1Ptr    pHewServer1;

};
```

(3) Create and initialize the smart pointer by a constructor.

File to correct: TypelibraryDemo2Dlg.cpp

```
// TODO: Add to this place when special initialization is desired.
// TODO: Add extra initialization here
try{
        //create smart pointer
        IHewServer1Ptr ptr(_uuidof(HewServer1));
        pHewServer1 = ptr;
}
```

(4) To ensure that the smart pointer will be discarded when a client terminates, add the statement shown below.

File to correct: TypelibraryDemo2Dlg.cpp

```
CTest1Dlg::~CTest1Dlg()
{
        // If there is an automation proxy for this dialog,
        // the pointer to this dialog is returned to NULL,
        // which indicates that the dialog has been deleted.
        if (m_pAutoProxy != NULL)
                        m_pAutoProxy->m_pDialog = NULL;
        //destroy smart pointer
        pHewServer1 = NULL;

}
...
void CTypelibraryDemo2Dlg::OnClose()
{
        if (CanExit())
                        CDialog::OnClose();

        //destroy smart pointer
        pHewServer1 = NULL;
}
...
void CTypelibraryDemo2Dlg::OnOK()
{
        if (CanExit())
                        CDialog::OnOK();

        //destroy smart pointer
        pHewServer1 = NULL;
}
...
void CTypelibraryDemo2Dlg::OnCancel()
{
        if (CanExit())
                        CDialog::OnCancel();

        //destroy smart pointer
        pHewServer1 = NULL;
}
```

(5) Next, add a function that you want to be called when a button is clicked.

(5-1) Right-click on the [CTypeLibraryDemo2Dlg] class in the [Class] view and select [Properties] from the popup menu.

(5-2) Click on ⚡ in the [Properties] pane to view a list of events.

(5-3) Click on [+] to the left of IDC_BUTTON_GO and select BN_CLICKED.

(5-4) Select <Add>OnBnClickedButtonGo from the drop-down menu.

(5-5) Add IDC_BUTTON_STOP in the same way (as in steps 5-3 and 5-4).

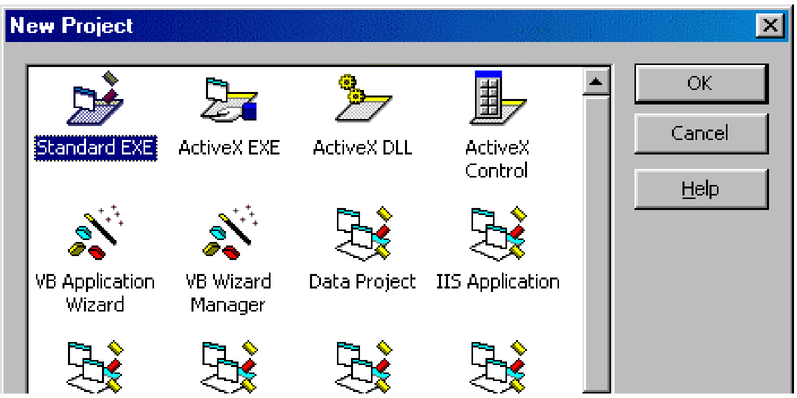
For Precautions, (6) Create the OnButtonGo() function, and subsequent procedures, see the descriptions about Visual C++ 6.0 (3.2.3, Creating Source Code). If you wish to use a Unicode library, use the _T() macro as a constant in the string.

Example: s1.Format(_T("%s"), CString(bstrErrStr));

3.4 Creating a Program (Visual Basic 6.0)

3.4.1 Generating project

Select Visual Basic Menu [File]->[New Project]. The "New Project" dialog box opens. Select "Standard EXE" and click the "OK" button.



3.4.2 Specification of Type Library

Select Visual Basic Menu [Project]->[References...] and check "HEWTargetServer 1.7 Type Library". Type library specification must be set for each project of Visual Basic.

3.4.3 Generating Object

Describe as follows on the VB code window. This code is the basic one for accessing the COM interface of HEWTargetServer.

```
1:   Dim WithEvents hts As HEWTargetServerLib.HewServer1
2:
3:   Private Sub Form_Load()
4:            Set hts = New HEWTargetServerLib.HewServer1
5:   End Sub
6:
7:   Private Sub Form_Unload(Cancel As Integer)
8:            Set hts = Nothing
9:   End Sub
```

Explanation of Each Line

1st line: Here, it is declared that the type of variable hts is "HEWTargetServerLib.HewServer1". This is the COM interface name.
Also, designate the description of "WithEvents" to obtain the event occurring on the High-performance Embedded Workshop side, such as Program execution start and program stop.
It is possible to change the variable name hts to any character string. The variable name hts can be any name.

3rd to 5th lines: This procedure (function) is called at applications startup (form open). Here, the object of "HEWTargetServerLib.HewServer1" is substituted for variable hts. The method of HEWTargetServer is accessed via this variable hts.

7th to 9th lines: This procedure (function) is accessed at applications end (form closing). Here, the object of variable hts is cancelled. If the object is cancelled, it will become impossible to call the method of HEWTargetServer.

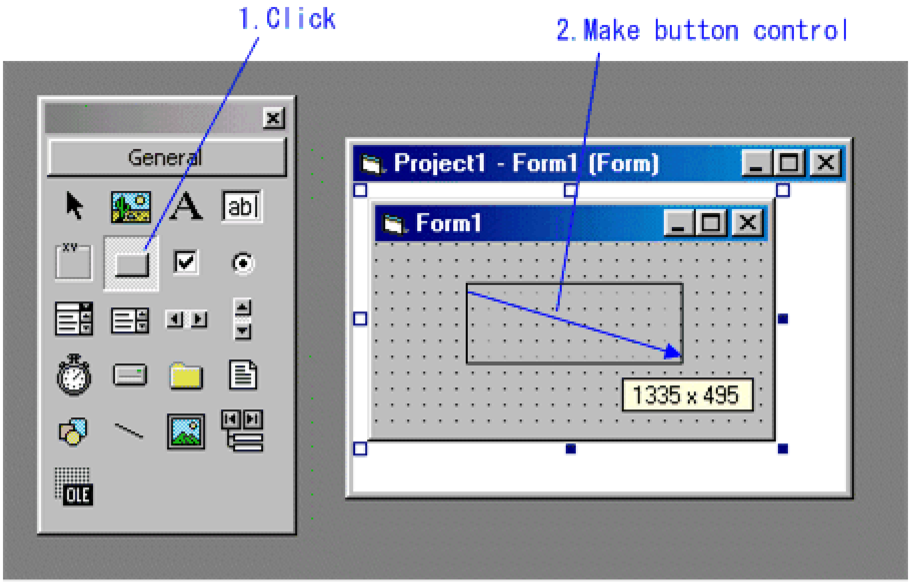
3.4.4 Method Access

Sample: Reset User Target

The following is the method of preparing the customized window for resetting the user target. In this application, a single button control is used.

(1) Adding Button Control

Click the Command button of the tool box to create one button control on the form.



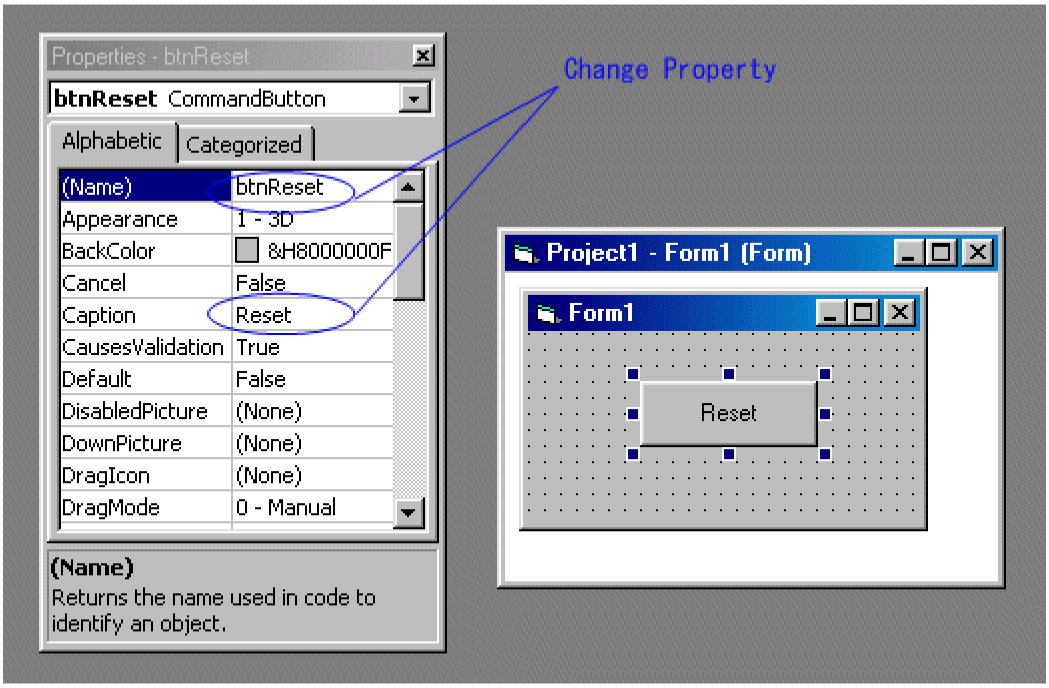
1. Click

2. Make button control

(2) Button Property Change

In the property window, alter the properties for the button control you created.

Property	Contents
(Name)	btnReset
Caption	Reset



Change Property

Properties - btnReset

btnReset CommandButton

(Name)	btnReset
Appearance	1 - 3D
BackColor	□ &H8000000F
Cancel	False
Caption	Reset
CausesValidation	True
Default	False
DisabledPicture	(None)
DownPicture	(None)
DragIcon	(None)
DragMode	0 - Manual

(Name)
Returns the name used in code to identify an object.

(3) Describe the button operation

Describe as follows in the code window. The bold-faced place indicates the additional part.

```
Dim WithEvents hts As HEWTARGETSERVERLib.HewServer1
Private Sub Form_Load()
      Set hts = New HEWTARGETSERVERLib.HewServer1
End Sub

Private Sub Form_Unload(Cancel As Integer)
      Set hts = Nothing
End Sub

Private Sub btnReset_Click()
      Dim rtn As Long
      On Error GoTo HTS_error
      rtn = HTS.GoTargetExec2
      Exit Sub
HTS_error:
      MsgBox Err.Description
End Sub
```

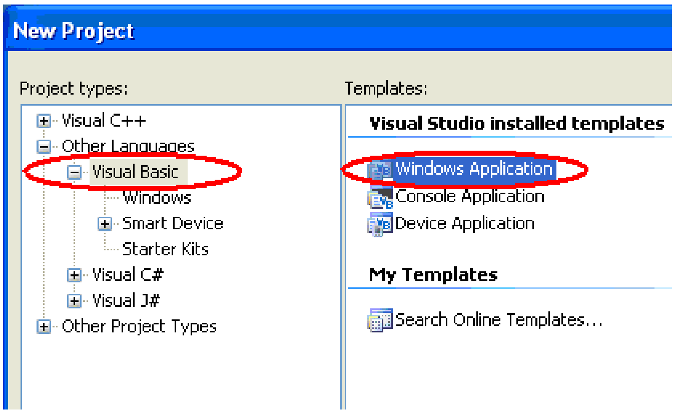
(4) Operation Check

Check for applications performance. First, startup High-performance Embedded Workshop. Next, select Visual Basic Menu [EXEC]->[START], and execute applications. By clicking the Reset button in the applications, the user target is reset.

3.5 Creating a Program (Visual Basic 2005)

3.5.1 Generating project

Select Visual Basic Menu [File]->[New Project]. The "New Project" dialog box opens. Select "Windows Application" and click the "OK" button.



3.5.2 Specification of Type Library

Select Visual Basic Menu [Project]->[Add References...] and open the [COM] tabbed page. Then select "HEWTargetServer 1.7 Type Library" and click the "OK" button. Type library specification must be set for each project of Visual Basic.

3.5.3 Generating Object

Describe as follows on the VB code window. This code is the basic one for accessing the COM interface of HEWTargetServer.

```
 1:    Public Class Form1
 2:          Dim WithEvents hts As HEWTARGETSERVERLib.HewServer1
 3:
 4:           Private Sub Form1_Load(ByVal sender As Object, ByVal e As System.
EventArgs) Handles Me.Load
 5:                hts = New HEWTARGETSERVERLib.HewServer1
 6:          End Sub
 7:
 8:          Private Sub Form1_FormClosed(ByVal sender As Object, ByVal e As
System.Windows.Forms.FormClosedEventArgs) Handles Me.FormClosed
 9:                HTS = Nothing
10:          End Sub
11:    End Class
```

Explanation of Each Line

2nd line: Here, it is declared that the type of variable hts is "HEWTargetServerLib.HewServer1". This is the COM interface name.
Also, designate the description of "WithEvents" to obtain the event occurring on the High-performance Embedded Workshop side, such as Program execution start and program stop.
It is possible to change the variable name hts to any character string. The variable name hts can be any name.

4th to 6th lines: This procedure (function) is called at applications startup (form open). Here, the object of "HEWTargetServerLib.HewServer1" is substituted for variable hts. The method of HEWTargetServer is accessed via this variable hts.

8th to 10th lines: This procedure (function) is accessed at applications end (form closing). Here, the object of variable hts is cancelled. If the object is cancelled, it will become impossible to call the method of HEWTargetServer.

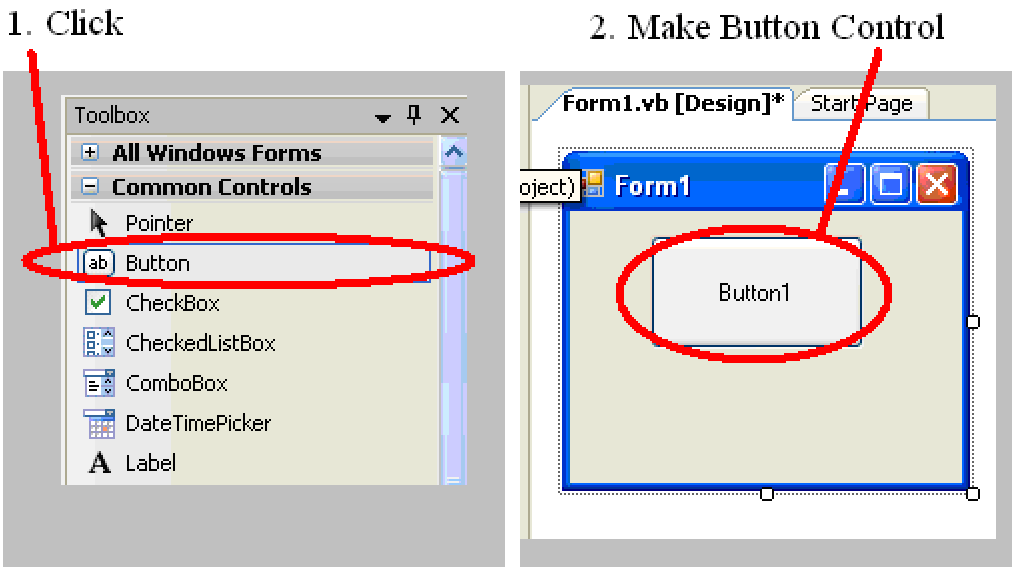
3.5.4 Method Access

Sample: Reset User Target

The following is the method of preparing the customized window for resetting the user target. In this application, a single button control is used.

(1) Adding Button Control

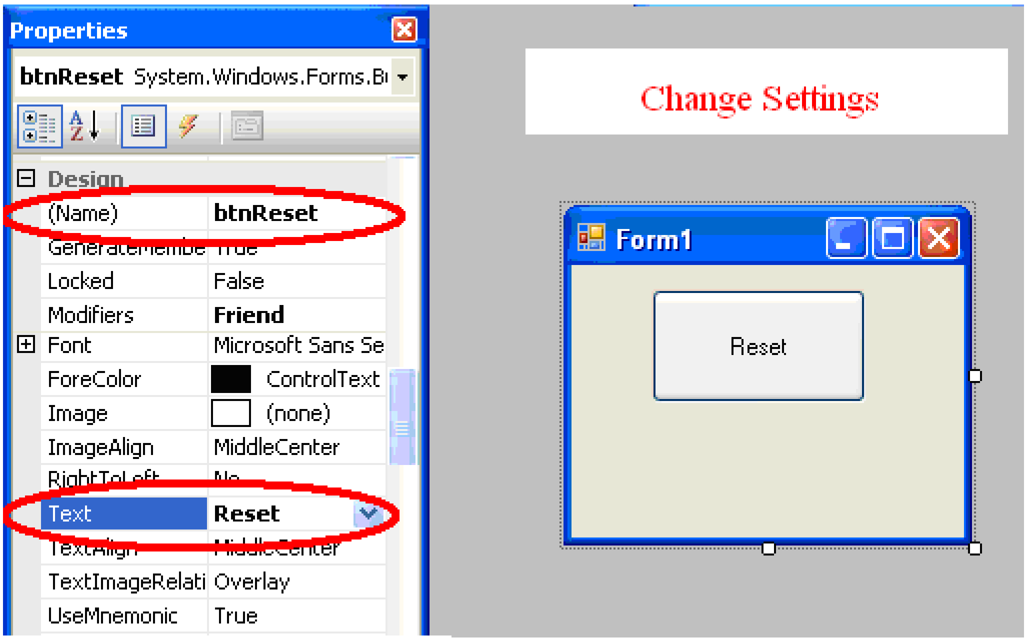
Click the Command button of the tool box to create one button control on the form.



(2) Button Property Change

In the property window, alter the properties for the button control you created.

Property	**Contents**
(Name)	**btnReset**
Text	**Reset**



Properties

btnReset System.Windows.Forms.Bu

Design

(Name)	btnReset
GenerateMembe	True
Locked	False
Modifiers	Friend
Font	Microsoft Sans Se
ForeColor	ControlText
Image	(none)
ImageAlign	MiddleCenter
RightToLeft	No
Text	Reset
TextAlign	MiddleCenter
TextImageRelati	Overlay
UseMnemonic	True

(3) Describe the button operation

Add a function to be executed in response to clicking on the btnReset button (for details, see the help information on VisualBasic).

Describe as follows in the code window. The bold-faced place indicates the additional part.

```
Public Class Form1
    Dim WithEvents hts As HEWTARGETSERVERLib.HewServer1

    Private Sub Form1_Load(ByVal sender As Object, ByVal e As System.EventArgs)
Handles Me.Load
        hts = New HEWTARGETSERVERLib.HewServer1
    End Sub

    Private Sub Form1_FormClosed(ByVal sender As Object, ByVal e As System.Windows.
Forms.FormClosedEventArgs) Handles Me.FormClosed
        HTS = Nothing
    End Sub

    Private Sub btnReset_Click(ByVal sender As System.Object, ByVal e As System.
EventArgs) Handles btnReset.Click
        Dim rtn As Integer
        Try
            rtn = hts.ResetTargetExec2()

        Catch ex As System.Runtime.InteropServices.COMException
            If ex.ErrorCode = &H8004FFFF Then
                MessageBox.Show("Hew Target is not linked up")
            ...
        Catch ex As Exception
            MessageBox.Show(ex.ToString())
        End Try

    End Sub
End Class
```

(4) Operation Check

Check for applications performance. First, startup High-performance Embedded Workshop. Next, select Visual Basic Menu [Debug]->[Start with debugging], and execute applications. By clicking the Reset button in the applications, the user target is reset.

3.6 Note on a Shift from Visual Basic 6.0 to Visual Basic .NET

If you have shifted from Visual Basic 6.0 to Visual Basic .NET and wish to use unstructured exception handling (On Error) available in Visual Basic 6.0, modify the exception-handling section as follows.

(1) Unstructured Exception Handling in Visual Basic 6.0

```
HTS_error:
    HTSErrorMsgBox Err.Description
End Sub
```

(2) Unstructured Exception Handling in Visual Basic .NET

```
HTS_error:
    strErrorMessage = ""
    If Err.Number = &H8004FFFF Then
      strErrorMessage = "Hew Target is not linked up"
    ElseIf Err.Number = &H80050000 Then
      strErrorMessage = "No module is downloaded"
    ElseIf Err.Number = &H80050001 Then
      strErrorMessage = "Invalid Break point handle"
    ElseIf Err.Number = &H80050002 Then
      strErrorMessage = "Error in pass in parameters"
    ElseIf Err.Number = &H80050003 Then
      strErrorMessage = "Invalid Begin and End address"
    ElseIf Err.Number = &H80050004 Then
      strErrorMessage = "No Interface"
    ElseIf Err.Number = &H80050005 Then
      strErrorMessage = "HEW was not found."
    Else
      HTS.GetErrorString2(Err.Number, strErrorMessage)
    End If

    HTSErrorMsgBox(strErrorMessage)
End Sub
```

4. Event Acquisition from the High-performance Embedded Workshop

When you've created a dialog-based application, you can use the method described below to get an event (Event3_ToClient_Stop). The sample here displays a dialog box for the case where the target program has stopped in the High-performance Embedded Workshop.

Note on acquisition of generated events:
The HEW target server uses flags not to issue specific events that frequently occur during execution of the target program. The following events are not issued if the flag is active after the same event has already been detected.

Event1_ToClient_TargetReset/Event5_ToClient_RegisterReset/Event8_ToClient_PlatformInitialize

Calling GetHewStatus() clears the flags. To acquire all events, call GetHewStatus() after each of the events has been issued.

4.1 Visual C++ Event Acquisition

(1) Import HewTargetServer.exe.

File to correct: StdAfx.h

```
#import "HewTargetServer.exe"          no_namespace named_guids
```

(2) Add AfxOleInit().

File to correct: CclientApp.cpp

```
BOOL CclientApp::InitInstance()
{
        AfxOleInit();
        ...
```

(3) Create and initialize a smart pointer by a constructor.

File to correct: CclientDig.cpp

```
#include "Afxctl.h"

...

CClientDlg::CClientDlg()
{
        ...
        try{
                //creating smart pointer from new i/f: IhewServer2
                IHewServer2Ptr ptr(_uuidof(HewServer1));
                pHewServer1 = ptr;
        }
        catch(_com_error e)
        {
        }
```

(4) Declare the smart pointer as a member variable.

File to correct: ClientDig.h

```
#include "EventHandler.h"
...
public:
  IHewServer2Ptr   pHewServer1;               //using smart pointer from new interface: IHewServer2
  protected:
        CEventHandler*     m_pHandler;
        DWORD              m_dwCookie;
...
```

(5) Creation of an event acquisition file 1

File to correct: EventHandler.cpp (Create a new file)

```
// EventHandler.cpp : implementation file

#include "stdafx.h"
#include "EventHandler.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

IMPLEMENT_DYNCREATE(CEventHandler, CCmdTarget)

CEventHandler::CEventHandler()
{
        EnableAutomation();
}

CEventHandler::~CEventHandler()
{
}

void CEventHandler::OnFinalRelease()
{
        CCmdTarget::OnFinalRelease();
}

BEGIN_MESSAGE_MAP(CEventHandler, CCmdTarget)
        //{{AFX_MSG_MAP(CEventHandler)
        //}}AFX_MSG_MAP
END_MESSAGE_MAP()

BEGIN_DISPATCH_MAP(CEventHandler, CCmdTarget)
        //{{AFX_DISPATCH_MAP(CEventHandler)
        DISP_FUNCTION_ID(CEventHandler, "Event3_ToClient_Stop", 3, OnHewStatusStop, VT_EMPTY,
0)      //}}AFX_DISPATCH_MAP
END_DISPATCH_MAP()

BEGIN_INTERFACE_MAP(CEventHandler, CCmdTarget)
        INTERFACE_PART(CEventHandler, DIID__IHewServer2Events, Dispatch)
END_INTERFACE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CEventHandler message handlers
void CEventHandler::OnHewStatusStop()
{
        AfxMessageBox("Event3_ToClient_Stop");
}
```

(6) Creation of an event acquisition file 2

File to correct: EventHandler.h (Create a new file)

```
#if !defined(AFX_EVENTHANDLER_H__0F96FDDD_7167_457D_8069_73D9AEFCDF49__INCLUDED_)
#define AFX_EVENTHANDLER_H__0F96FDDD_7167_457D_8069_73D9AEFCDF49__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
// EventHandler.h : header file

/////////////////////////////////////////////////////////////////////////////
// CEventHandler command target

class CEventHandler : public CCmdTarget
{
        DECLARE_DYNCREATE(CEventHandler)
        CEventHandler();            // protected constructor used by dynamic creation
public:
// Overrides
        // ClassWizard generated virtual function overrides
        //{{AFX_VIRTUAL(CEventHandler)
        public:
        virtual void OnFinalRelease();
        //}}AFX_VIRTUAL

// Implementation
public:
        virtual ~CEventHandler();

        // Generated message map functions
        //{{AFX_MSG(CEventHandler)
                afx_msg void OnHewStatusStop();             //event call back function
        //}}AFX_MSG

        DECLARE_MESSAGE_MAP()
        // Generated OLE dispatch map functions
        //{{AFX_DISPATCH(CEventHandler)
                // NOTE - the ClassWizard will add and remove member functions here.
        //}}AFX_DISPATCH
        DECLARE_DISPATCH_MAP()
        DECLARE_INTERFACE_MAP()
};
/////////////////////////////////////////////////////////////////////////////
//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately before the previous line.

#endif // !defined(AFX_EVENTHANDLER_H__0F96FDDD_7167_457D_8069_73D9AEFCDF49__INCLUDED_)
```

4.2 Visual Basic Event Acquisition

To obtain an event (program execution start, etc.) arising on the High-performance Embedded Workshop side, use procedure hts_GotEventMessage. This procedure is a subroutine called out when the event arising on the High-performance Embedded Workshop side was received. ("hts" at the head of the procedure name denotes the variable name of the object designated at the head of the program.)

Visual Basic 6.0

```
Private Sub hts_GotEventMessage(ByVal action As Long)
End Sub
```

Visual Basic 2005

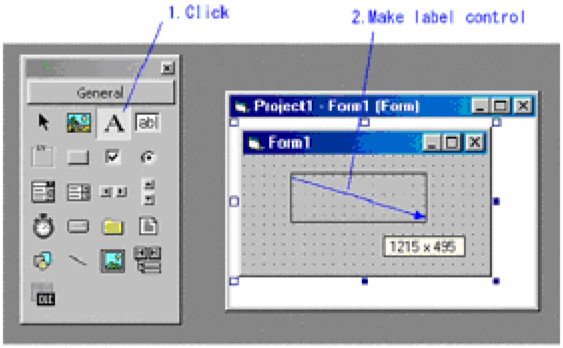
```
Private Sub hts_GotEventMessage(ByVal action As Integer)
End Sub
```

Event No. arising on the High-performance Embedded Workshop side is stored in parameter "action". With this procedure, describe the processing to be executed where the event was received from High-performance Embedded Workshop.

Sample: Get an event arising on High-performance Embedded Workshop side. The following is the method of creating the customized window for getting an event arising on the High-performance Embedded Workshop side and displaying its number (the figures below show examples of Visual Basic 6.0). For this application, one label control is used.

(1) Add label control

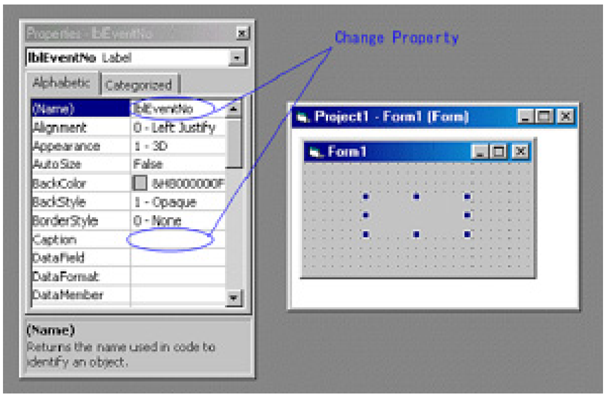
Click on "Label" of the tool box and create one label control on the form.



(2) Change the property

In the property window, change the properties for the label control you created.

Property	Contents
(Name)	lblEventNo
Caption	(nothing)



(Name)	lblEventNo
Alignment	0 - Left Justify
Appearance	1 - 3D
AutoSize	False
BackColor	&H8000000F
BackStyle	1 - Opaque
BorderStyle	0 - None
Caption	
DataField	
DataFormat	
DataMember	

(Name)
Returns the name used in code to identify an object.

(3) Describe the operation when an event occurred
Describe as follows in the code window. The section in red indicates the addition part.

Visual Basic 6.0

```
Dim WithEvents hts As HEWTARGETSERVERLib.HewServer1
Private Sub Form_Load()
    Set hts = New HEWTARGETSERVERLib.HewServer1
End Sub
Private Sub Form_Unload(Cancel As Integer)
    Set hts = Nothing
End Sub
Private Sub hts_GotEventMessage(ByVal action As Long)
    lblEventNo.Caption = action
End Sub
```

Visual Basic 2005

```
Public Class Form1
    Dim WithEvents hts As HEWTARGETSERVERLib.HewServer1
    Private Sub Form1_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles Me.Load
        hts = New HEWTARGETSERVERLib.HewServer1
    End Sub
    Private Sub Form1_FormClosed(ByVal sender As Object, ByVal e As System.Windows.
Forms.FormClosedEventArgs) Handles Me.FormClosed
       hts = Nothing
    End Sub
    Private Sub hts_GotEventMessage(ByVal action As Integer)
        lblEventNo.Caption = action
    End Sub
End Class
```

(4) Check for action
Check the applications action. Startup High-performance Embedded Workshop, select Visual Basic Menu [EXEC]->[START], and execute applications. By manipulating High-performance Embedded Workshop (go/stop, etc.), the number of an event that occurred in its High-performance Embedded Workshop is displayed.

5.Method List

HEW Target Server (COM) is disclosing the following method (function).

Notes:

(1) Support for VisualBasic6.0

The HEW target server for High-performance Embedded Workshop V.4.02 and later versions supports VisualBasic6.0 (hereafter referred to as VB). Parameter types for existing methods have been changed so that they can be used from VB. Names of these methods with their parameter type changed include "2".

Example 1: Step(2, 1); // Existing VC++ method with DWORD-type parameters
Example 2: Step2(2, 1); // New VB/VC++ method with long-type parameters
Methods added from the High-performance Embedded Workshop can also be used from VB and VC++.

(2) Difference between GetMemory() and GetDirectMemory()

GetMemory() acquires data from the cache memory, while GetDirectMemory() acquires data from memory in the MCU. To acquire data during execution of the user program, use GetDirectMemory(). When the debugger in use does not support caching, the operation of GetMemory() and GetDirectMemory() is the same.

(3) Invoking multiple High-performance Embedded Workshop

When multiple High-performance Embedded Workshop are invoked, the HEW target server cannot distinguish them. These High-performance Embedded Workshop will thus perform the same operation.

(4) Path for HewTargetServer.exe

If the path for HewTargetServer.exe has changed (e.g., by re-installation of the High-performance Embedded Workshop), use the registration tool (ALL_REGISTERSERVER.bat) to register HewTargetServer.exe again.
(For details, refer to Section 2.1.2, " Registering HewTargetServer.exe in Your Registry".)

5.1 Method Outline (for only VC++)

5.1.1 CPU Control

Method Name	Parameter	Description	Page
GoTargetExec	(Nothing)	Executes a program.	P38
StopTargetExec	(Nothing)	Stops program execution.	P39
ResetTargetExec	(Nothing)	Resets the debugger environment.	P40
InitializeTarget	(Nothing)	Initializes the debugger environment.	P41
Step	[in] DWORD _eMode [in] DWORD _dWStep	Step executes the target program.	P42
StepRate	[in] DWORD _dwRate	Sets a speed at which the program is single-stepped.	P43
StepOver	[in] DWORD _eMode [in] DWORD _dwStep	Runs a program by stepping-over instructions or source lines.	P44
StepOut	[in] DWORD _eMode	Runs a program by stepping-out instructions or source lines.	P45
IsRunning	[out] long* p_bRunning	Determines whether or not the current user program is running.	P46

5.1.2 Register

Method Name	Parameter	Description	Page
GetPC	[out] DWORD *p_dwPC	Gets the current program counter value.	P47
SetPCAddress	[in] ADDR aPCAddr	Sets the program counter.	P48
SetPCSource	[in] BSTR bstrFileName [in] LINENO LineNum	Sets the program counter by specifying a source file and line.	P49
TestSetPC	[out] long* p_bSetPCState	Determines whether or not the PC (program counter) value can be set.	P50

5.1.3 Memory

Method Name	Parameter	Description	Page
GetMemory	[in] ADDR _aBegin [in] ADDR _aEnd [in] WORD _wDisplayWidth [out] SAGEARRAY(BYTE)* _ppbyBuff	Gets a memory data.	P51
SetMemory	[in] ADDR _aBegin [in] ADDR _aEnd [in] WORD _wDisplayWidth [in] SAGEARRAY(BYTE) _ppbyBuff	Sets a memory data.	P52
GetDirectMemory	[in] ADDR _aBegin [in] ADDR _aEnd [in] WORD _wDisplayWidth [out] SAGEARRAY(BYTE)* _ppbyBuff	Gets a direct memory data.	P53

5.1.4 Software Breaks

Method Name	Parameter	Description	Page
SetPCBreakPt	[in] ADDR _aPCBreakAddr [out] BHANDLE* p_Bhandle	Registers the software breakpoint.	P54
EnableBreakPt	[in] BHANDLE p_Bhandle [in] VARIANT_BOOL bEnable	Enable or disable the software breakpoint.	P55
DeleteBreakPt	[in] BHANDLE BHandle	Delete the software breakpoint.	P56
GetAllBreakPt	[out] long *p_index [out] VARIANT *p_vAllBreakPt	Gets the software breakpoints that have been set.	P57
DeleteAllBreakPt	(Nothing)	Deletes the software breakpoints that have been set.	P58

5.1.5 Variable Break

Method Name	Parameter	Description	Page
SetDataBreakpoint	[in] DWORD _aSymbol [in] DWORD _eSize [in] DWORD _eType [in] DWORD _dwData [out] DWORD *p_dwBreakDataNo	Registers the data breakpoint.	P59
EnableDataBreakpoint	[in] DWORD dwBreakDataNo [in] VARIANT_BOOL _bEnable	Enables or disables the data breakpoint.	P60
DeleteDataBreakpoint	[in] DWORD dwBreakDataNo	Delete the data breakpoint.	P61

5.1.6 Variable Trace

Method Name	Parameter	Description	Page
SetSymbolTrace	[in] ADDR _aSymbol [in] DWORD _eConditon [in] DWORD _eSize [in] DWORD _eType [in] DWORD _dwData [out] DWORD *p_dwTraceNo	Sets the variable trace conditions.	P62
ExecuteSymbolTrace	[in] VARIANT_BOOL _bEnable	Enables or disables the variable traces.	P63
DeleteSymbolTrace	[in] DWORD _dwTrace	Deletes the variable trace conditions.	P64
SaveSymbolTraceData	[in] BSTR _bstrFileName	Saves the variable trace result to a specified file.	P65

5.1.7 Interrupt Condition

Method Name	Parameter	Description	Page
SendTrigger	[in] DWORD _dwTriggerNo [in] DWORD _dwTriggerType1 [in] DWORD _dwTriggerType2 [in] DWORD _dwPriority	Sets interrupt conditions.	P66

5.1.8 Symbol

Method Name	Parameter	Description	Page
GetRealTimeWatch	[in] ADDR _aSymbol [in] DWORD _eSize [out] DWORD *p_dwValue	Gets the real-time watch.	P67
GetQuickWatch	[in] BSTR bstrVarName [out] DWORD* p_dwValueSize [out] BSTR* bstrByValue [out] EobjectTypeServer* p_eType [out] BSTR* bstrVariableAllocation	Gets the value that corresponds to a string character.	P68
SymbolToAddress	[in] BSTR bstrSymbolName [out] ADDR* p_aSymbolAddr	Converts from symbol to address.	P69
AddressToSymbol	[in] ADDR aSymbolAddr [out] BSTR* p_bstrSymbolName	Converts from address to symbol.	P70
GetLineFromAddr	[in] ADDR _aLineAddr [out] BSTR* p_bstrFileName [out] LINEO* p_LineNo	Gets the source file name and the line number corresponding to the specified address.	P71
GetAddrFromLine	[in] BSTR bstrFileName [in] LINENO LineNo [out] ADDR* p_aLineAddr	Gets the address of specified source line information.	P72

5.1.9 Downloads

Method Name	Parameter	Description	Page
Download	[in] BSTR _bstrFileName	Downloads the target program.	P73
Unload	[in] BSTR _bstrFileName	Unloads the target program.	P74

5.1.10 Start/Stop

Method Name	Parameter	Description	Page
InvokeHew	(Nothing)	Starts a High-performance Embedded Workshop application.	P75
QuitHew	(Nothing)	Closes a High-performance Embedded Workshop application.	P76

5.1.11 Workspace

Method Name	Parameter	Description	Page
OpenWorkspace	[in] BSTR _bstrFileName	Opens a workspace.	P77
CloseWorkspace	[in] DWORD _dwIgnoreChanges	Closes the workspace.	P78
SaveWorkspace	(Nothing)	Saves the workspace.	P79

5.1.12 Configuration and session

Method Name	Parameter	Description	Page
SaveSession	(Nothing)	Saves the session file.	P80
GetCurrentConfiguration	[out] BSTR *p_bstrCurrentConfigurationName	Gets the current build configuration.	P81
SetCurrentConfiguration	[in] BSTR _bstrConfiguration	Sets a build configuration.	P82
GetConfigurations	[out] BSTR *p_strConfigurations	Gets registered build configurations.	P83
GetCurrentSession	[out] BSTR *p_bstrCurrentSessionName	Gets the current debug session.	P84
SetCurrentSession	[in] BSTR _bstrSession	Sets a debug session.	P85
GetSessions	[out] BSTR *p_bstrSessions	Gets registered debug sessions.	P86
GetCurrentProject	[out] BSTR *p_bstrCurrentProjectName	Gets the current project.	P87
GetProjects	[out] BSTR *p_bstrProjectNames	Gets all project names.	P89
SetCurrentProject	[in] BSTR _bstrProjectName	Sets the active project.	P88

5.1.13 Project

Method Name	Parameter	Description	Page
AddFile	[in] BSTR _bstrFileName	Adds a file to the project.	P90
AddFiles	[in] BSTR _bstrFileName	Adds multiple files to the project.	P91
DeleteFile	[in] BSTR _bstrFileName	Deletes a file from the project.	P92
DeleteFiles	[in] BSTR _bstrFileName	Deletes multiple files from the project.	P93

5.1.14 Build

Method Name	Parameter	Description	Page
BuildProject	(Nothing)	Builds a project.	P94
RebuildProject	(Nothing)	Rebuilds a project.	P95
UpDateAllDependency	(Nothing)	Updates all dependency relations.	P96
AddFileWithCompilerOption	[in] BSTR _bstrFileName [in] BSTR _bstrIncludeDirectories [in] BSTR _bstrDefines	Adds a file after setting compiler options for the project.	P97

5.1.15 Files

Method Name	Parameter	Description	Page
OpenFileAtLine	[in] BSTR _bstrOpenFileName [in] int _iLine	Opens a file by specifying the file name and line number.	P98
GetSourceFiles	[out] BSTR *p_bstrSourceFiles	Gets source file names.	P99
GetDownloadModules	[out] BSTR *p_bstrDownloadModules	Gets module file names.	P100
GetDependentFiles	[out] BSTR *p_bstrDependentFiles	Gets dependent file names.	P101

5.1.16 Coverage

Method Name	Parameter	Description	Page
SetCoverageRange	[in] DWORD dwStartAddress [in] DWORD dwEndAddress	Sets a coverage range.	P102
GetCoverageRange	[out] DWORD *p_dwStartAddress [out] DWORD *p_dwEndAddress	Gets data from a coverage range.	P103
SetCoverageDisable	(Nothing)	Disables the coverage function.	P104
SetCoverageEnable	(Nothing)	Enables the coverage function.	P105
ClearCoverage	(Nothing)	Clears the coverage information.	P106
GetCoverageStatus	[out] int *p_iStatus	Gets the coverage status information.	P107
LoadCoverage	[in] BSTR _bstrLoadFileName	Loads the coverage information.	P108
SaveCoverage	[in] BSTR _bstrSaveFileName	Saves the coverage information.	P109

5.1.17 Others

Method Name	Parameter	Description	Page
GetErrorString	[in] HRESULT _lError [out] BSTR* _pbstrError	Gets an error string occurred in a method call.	P110
GetHewStatus	[out] int* p_iTargetReset [out] int* p_iTaStatus [out] int* p_iMemoryReset [out] int* p_iRegisterReset [out] int* iPlatformInitialize [out] int* p_iLoadingStatus	Gets status.	P111
GetHewStatusEx	[out] int *p_iInvokeHew [out] int *p_iOpenWorkspace [out] int *p_iBuildProject	Gets the status information (on initiation, opening a workspace, and build).	P112
GetTargetName	[out] BSTR* p_bstrName	Gets a target name.	P113

5.2 Method Outline (for VB, VC++)

5.2.1 CPU Control

Method Name	Parameter	Description	Page
GoTargetExec2	(Nothing)	Executes a program.	P114
StopTargetExec2	(Nothing)	Stops program execution.	P115
ResetTargetExec2	(Nothing)	Resets the debugger environment.	P116
InitializeTarget2	(Nothing)	Initializes the debugger environment.	P117
Step2	[in] long _lMode [in] long _lStep	Step executes the target program.	P118
StepRate2	[in] long _lRate	Sets a speed at which the program is single-stepped.	P119
StepOver2	[in] long _lMode [in] long _lStep	Runs a program by stepping-over instructions or source lines.	P120
StepOut2	[in] long _lMode	Runs a program by stepping-out instructions or source lines.	P121
IsRunning2	[out] long * p_lRunning	Determines whether or not the current user program is running.	P122

5.2.2 Register

Method Name	Parameter	Description	Page
GetPC2	[out] long *p_lPC	Gets the current program counter value.	P123
SetPCAddress2	[in] long lPCAddr	Sets the program counter.	P124
SetPCSource2	[in] BSTR bstrFileName [in] long lLineNum	Sets the program counter by specifying a source file and line.	P125
TestSetPC2	[out] long* p_lSetPCState	Determines whether or not the PC (program counter) value can be set.	P126

5.2.3 Memory

Method Name	Parameter	Description	Page
GetMemory2	[in] long lBegin [in] long lEnd [in] long lDisplayWidth [out] VARIANT *p_vMemData	Gets a memory data.	P127
SetMemory2	[in] long lBegin [in] long lEnd [in] long lDisplayWidth [in] VARIANT vMemData	Sets a memory data.	P128
GetDirectMemory2	[in] long lBegin [in] long lEnd [in] long lDisplayWidth [out] VARIANT *p_vMemData	Gets a direct memory data.	P130

5.2.4 Software Breaks

Method Name	Parameter	Description	Page
SetPCBreakPt2	[in] long lPCBreakAddr [out] long *p_lHandle	Registers the software breakpoint.	P131
EnableBreakPt2	[in] long lHandle [in] long lEnable	Enable or disable the software breakpoint.	P132
DeleteBreakPt2	[in] long lHandle	Delete the software breakpoint.	P133
GetAllBreakPt2	[out] long *p_index [out] VARIANT *p_vAllBreakPt	Gets the software breakpoints that have been set.	P134
DeleteAllBreakPt2	(Nothing)	Deletes the software breakpoints that have been set.	P135

5.2.5 Variable Break

Method Name	Parameter	Description	Page
SetDataBreakpoint2	[in] long _lSymbol [in] long _lSize [in] long _lType [in] long _lData [out] long *p_lBreakDataNo	Registers the data breakpoint.	P136
EnableDataBreakpoint2	[in] long lDataBreakNo [in] long _lEnable	Enables or disables the data breakpoint.	P137
DeleteDataBreakpoint2	[in] long lDataBreakNo	Delete the data breakpoint.	P138

5.2.6 Variable Trace

Method Name	Parameter	Description	Page
SetSymbolTrace2	[in] long _lSymbol [in] long _lConditon [in] long _lSize [in] long _lType [in] long _lData [out] long *p_lTraceNo	Sets the variable trace conditions.	P139
ExecuteSymbolTrace2	[in] long _lEnable	Enables or disables the variable traces.	P140
DeleteSymbolTrace2	[in] long _lTraceNo	Deletes the variable trace conditions.	P141
SaveSymbolTraceDeta2	[in] BSTR _bstrFileName	Saves the variable trace result to a specified file.	P142

5.2.7 Interrupt Condition

Method Name	Parameter	Description	Page
SendTrigger2	[in] long _lTriggerNo [in] long _lTriggerType1 [in] long _lTriggerType2 [in] long _lPriority	Sets interrupt conditions.	P143

5.2.8 Symbol

Method Name	Parameter	Description	Page
GetRealTimeWatch2	[in] long _lSymbol [in] long _lSize [out] long *p_lValue	Gets the real-time watch.	P144
GetQuickWatch2	[in] BSTR bstrVarName [out] long *p_lValueSize [out] BSTR *bstrByValue [out] long *p_lType [out] BSTR *bstrTypeName [out] BSTR *bstrVarAllocation	Gets the value that corresponds to a string character.	P145
SymbolToAddress2	[in] BSTR bstrSymbolName [out] long *p_lSymbolAddr	Converts from symbol to address.	P146
AddressToSymbol2	[in] long lSymbolAddr [out] BSTR *p_bstrSymbolName	Converts from address to symbol.	P147
GetLineFromAddr2	[in] long lLineAddr [out] BSTR *p_bstrFileName [out] long *p_lLineNo	Gets the source file name and the line number corresponding to the specified address.	P148
GetAddrFromLine2	[in] BSTR bstrFileName [in] long lLineNo [out] long *p_lLineAddr	Gets the address of specified source line information.	P149

5.2.9 Downloads

Method Name	Parameter	Description	Page
Download2	[in] BSTR _bstrFileName	Downloads the target program.	P150
Unload2	[in] BSTR _bstrFileName	Unloads the target program.	P151

5.2.10 Start/Stop

Method Name	Parameter	Description	Page
InvokeHew2	(Nothing)	Starts a High-performance Embedded Workshop application.	P152
QuitHew2	(Nothing)	Closes a High-performance Embedded Workshop application.	P153
InvokeHewWithNoDialog	(Nothing)	Invokes the High-performance Embedded Workshop application without opening the [Welcome!] dialog box.	P154

5.2.11 Workspace

Method Name	Parameter	Description	Page
OpenWorkspace2	[in] BSTR _bstrFileName	Opens a workspace.	P155
CloseWorkspace2	[in] long _lIgnoreChanges	Closes the workspace.	P156
SaveWorkspace2	(Nothing)	Saves the workspace.	P157
GetWorkSpaceDirectory	[out] BSTR *_pbstrCurrentWorkspaceDirectory	Gets the absolute path of the current workspace.	P158

5.2.12 Configuration and session

Method Name	Parameter	Description	Page
SaveSession2	(Nothing)	Saves the session file.	P159
GetCurrentConfiguration2	[out] BSTR *p_bstrCurrentConfigurationName	Gets the current build configuration.	P160
SetCurrentConfiguration2	[in] BSTR _bstrConfiguration	Sets a build configuration.	P161
GetConfigurations2	[out] BSTR *p_strConfigurations	Gets registered build configurations.	P162
GetCurrentSession2	[out] BSTR *p_bstrCurrentSessionName	Gets the current debug session.	P163
SetCurrentSession2	[in] BSTR _bstrSession	Sets a debug session.	P164
GetSessions2	[out] BSTR *p_bstrSessions	Gets registered debug sessions.	P165
GetCurrentProject2	[out] BSTR *p_bstrCurrentProjectName	Gets the current project.	P166
GetProjects2	[out] BSTR *p_bstrProjectNames	Gets all project names.	P168
SetCurrentProject2	[in] BSTR _bstrProjectName	Sets the active project.	P167

5.2.13 Project

Method Name	Parameter	Description	Page
AddFile2	[in] BSTR _bstrFileName	Adds a file to the project.	P169
AddFiles2	[in] BSTR _bstrFileName	Adds multiple files to the project.	P170
DeleteFile2	[in] BSTR _bstrFileName	Deletes a file from the project.	P171
DeleteFiles2	[in] BSTR _bstrFileName	Deletes multiple files from the project.	P172
AddProjectFileFolder	[in] BSTR _bstrFolderName	Adds a folder to the Projects tree.	P173
RemoveProjectFileFolder	[in] BSTR _bstrFolderName	Deletes a folder from the Projects tree.	P174
AddFileToFolder	[in] BSTR _bstrFileName [in] BSTR _bstrFolderName	Adds a file to a specific folder under the Projects tree.	P175

5.2.14 Build

Method Name	Parameter	Description	Page
BuildProject2	(Nothing)	Builds a project.	P176
RebuildProject2	(Nothing)	Rebuilds a project.	P177
UpDateAllDependency2	(Nothing)	Updates all dependency relations.	P178
AddFileWithCompilerOption2	[in] BSTR _bstrAddFileName [in] BSTR _bstrIncludeDirectories [in] BSTR _bstrDefines	Adds a file after setting compiler options for the project.	P179
GetLibraryOptions	[out] BSTR *p_bstrLibraryOption	Acquires the library options for the linker in the current project.	P180
SetLibraryOptions	[in] BSTR _bstrLibraryOption	Sets library options for the linker in the current project.	P181
GetLibraryFilesForConfiguration	[in] BSTR _bstrProjectName [in] BSTR _bstrConfiguration [out] BSTR *_pbstrLibraryFiles	Gets library options from a specific configuration in a specific project.	P182
SetLibraryFilesForConfiguration	[in] BSTR _bstrProjectName [in] BSTR _bstrConfiguration [in] BSTR _bstrLibraryFiles	Sets library options for a specific configuration in a specific project.	P183
GetIncludeFileDirectories	[in] BSTR _bstrProjectName [in] BSTR _bstrConfiguration [in] BSTR _bstrFileName [out] VARIANT *_pvtIncludeDirectories	Gets include file options from a file of a specific configuration in a specific project.	P184
SetIncludeFileDirectories	[in] BSTR _bstrProjectName [in] BSTR _bstrConfiguration [in] BSTR _bstrFileName [in] VARIANT _vtIncludeDirectories [in] long _lSettingMode	Sets include file options for a file of a specific configuration in a specific project.	P185
GetCpuAndToolChainData	[in] BSTR _bstrProjectName [out] BSTR *_pbstrCPUFamily [out] BSTR *_pbstrCPUSeries [out] BSTR *_pbstrCPUType [out] BSTR *_pbstrToolChainFamily [out] BSTR *_pbstrToolChainName [out] BSTR *_pbstrToolChainVersion	Gets the family name, series name, and type name of the CPU, and the family name, name, and version number of the compiler in a specific project.	P186
SetBuildExcludeFiles	[in] BSTR _bstrFileNames	Excludes the specified file from building.	P188
SetBuildIncludeFiles	[in] BSTR _bstrFileNames	Includes the specified file in building.	P189

5.2.15 Files

Method Name	Parameter	Description	Page
OpenFileAtLine2	[in] BSTR _bstrOpenFileName [in] long _lLine	Opens a file by specifying the file name and line number.	P190
GetSourceFiles2	[out] BSTR *p_bstrSourceFiles	Gets source file names.	P191
GetDownloadModules2	[out] BSTR *p_bstrDownloadModules	Gets module file names.	P192
GetDependentFiles2	[out] BSTR *p_bstrDependentFiles	Gets dependent file names.	P193

5.2.16 Coverage

Method Name	Parameter	Description	Page
SetCoverageRange2	[in] long _lStartAddress [in] long _lEndAddress	Sets a coverage range.	P194
GetCoverageRange2	[out] long *p_lStartAddress [out] long *p_lEndAddress	Gets data from a coverage range.	P195
SetCoverageDisable2	(Nothing)	Disables the coverage function.	P196
SetCoverageEnable2	(Nothing)	Enables the coverage function.	P197
ClearCoverage2	(Nothing)	Clears the coverage information.	P198
GetCoverageStatus2	[out] long *p_lStatus	Gets the coverage status information.	P199
LoadCoverage2	[in] BSTR _bstrLoadFileName	Loads the coverage information.	P200
SaveCoverage2	[in] BSTR _bstrSaveFileName	Saves the coverage information.	P201

5.2.17 Others

Method Name	Parameter	Description	Page
GetErrorString2	[in] long lError [out] BSTR *p_bstrError	Gets an error string occurred in a method call.	P202
GetHewStatus2	[out] long *p_lTargetReset [out] long *p_lTagetExecStatus [out] long *p_lMemoryReset [out] long *p_lRegisterReset [out] long *p_lLinkStatus [out] long *p_lPlatformInitialize [out] long *p_lLoadingStatus	Gets status.	P203
GetHewStatusEx2	[out] long *p_lInvokeHew [out] long *p_lOpenWorkspace [out] long *p_lBuildProject	Gets the status information (on initiation, opening a workspace, and build).	P205
GetTargetName2	[out] BSTR *p_bstrName	Gets a target name.	P206
GetHewVersion	[out] BSTR *p_bstrHewVersion	Gets the version number of the High-performance Embedded Workshop.	P207
Command	[in] BSTR _bstrCommandLine [out]BSTR *p_bCommandMessage	Executes a High-performance Embedded Workshop command.	P208

5.3 Method Details (for only VC++)

5.3.1 CPU Control

GoTargetExec

Description

Executes a program from the current program position.

Parameters

There is no parameter.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULThr = E_FAIL;

try
{
            hr = pHewServer1->GoTargetExec();
}
```

StopTargetExec

Description

Stops program execution.

Parameters

There is no parameter.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULThr = E_FAIL;

try
{
        hr = pHewServer1->StopTargetExec();
}
```

ResetTargetExec

Description

Resets the debugger environment that is run as the target.

Parameters

There is no parameter.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULThr = E_FAIL;

try
{
        hr = pHewServer1->ResetTargetExec();
}
```

InitializeTarget

Description

Initializes the debugger environment that is run as the target.

Parameters

There is no parameter.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULThr = E_FAIL;

try
{
        hr = pHewServer1->InitializeTarget();
}
```

Step

Description

Step executes the target program.

Parameters

Attribute	Type	Content
[in]	DWORD _eMode	Description 0x00000001 Steps through assembler instructions 0x00000002 Steps through source code lines
[in]	DWORD _dwStep	Number of steps executed

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
DWORD _eMode = 1;               //assembler:1, source:2
DWORD _dwStep;

try
{
            hr = pHewServer1->Step(_eMode, _dwStep);
}
```

StepRate

Description

Sets a speed at which the program is single-stepped.

Parameters

Attribute	Type	Content
[in]	DWORD _dwRate	Set a stepping rate in the range 0-6. 0 : 3 seconds 1 : 2.5 seconds 2 : 2 seconds 3 : 1.5 seconds 4 : 1 seconds 5 : 0.5 seconds 6 : 0 seconds

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT                 hr = E_FAIL;
DWORD _dwRate;

try
{
            hr = pHewServer1->StepRate(_dwRate);
}
```

StepOver

Description

Runs a program by stepping-over instructions or source lines.

Parameters

Attribute	Type	Content
[in]	DWORD _eMode	Description 0x00000001 Steps through assembler instructions 0x00000002 Steps through source code lines
[in]	DWORD _dwStep	Number of steps executed

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
DWORD _eMode = 1;               //assembler:1, source:2
DWORD _dwStep;

try
{
            hr = pHewServer1->StepOver(_eMode, _dwStep);
}
```

StepOut

Description

Runs a program by stepping-out instructions or source lines.

Parameters

Attribute	Type	Content
[in]	DWORD _eMode	Description 0x00000001 Steps through assembler instructions 0x00000002 Steps through source code lines

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
DWORD _eMode = 1;              //assembler:1, source:2

try
{
            hr = pHewServer1->StepOut(_eMode);
}
```

IsRunning

Description

Determines whether or not the user program is running.

Parameters

Attribute	Type	Content
[out]	long* p_bRunning	1 when the user program is running or 0 otherwise

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
long _bRunning;

try
{
            hr = pHewServer1->IsRunning(&_bRunning);
}
```

5.3.2 Register

GetPC

Description

Gets the program counter value.

Parameters

Attribute	Type	Content
[out]	DWORD *p_dwPC	PC (program counter) value

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
DWORD _dwPC = 0x0;

try
{
            hr = pHewServer1->GetPC(&_dwPC);
}
```

SetPCAddress

Description

Sets the program counter.

Parameters

Attribute	Type	Content
[in]	ADDR aPCAddr	PC (program counter) value to be set

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
ADDR aPCAddr;

try
{
            hr = pHewServer1->SetPCAddress(aPCAddr);
}
```

SetPCSource

Description

Sets the program counter by specifying a source file and line.

Parameters

Attribute	Type	Content
[in]	BSTR bstrFileName	File name
[in]	LINENO LineNum	Line number

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT     hr = E_FAIL;
BSTR bstrSetPCSourceFile;
LINENO LineNum;

try
{
            hr = pHewServer1->SetPCSource(bstrSetPCSourceFile,   LineNum);
}
```

TestSetPC

Description

Determines whether or not the program counter value can be set.

Parameters

Attribute	Type	Content
[out]	long* p_bSetPCState	1 when the PC value can be set or 0 otherwise

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
long        lSetPC = NULL;
HRESULT     hr = E_FAIL;

try
{
            hr = pHewServer1->TestSetPC(&lSetPC);
}
```

5.3.3 Memory

GetMemory

Description

Gets memory content according to specified start and end addresses and access size. If the memory content of this specified area is held in the High-performance Embedded Workshop, the memory content which is added in the High-performance Embedded Workshop is returned directly without accessing the target memory.

Parameters

Attribute	Type	Content
[in]	ADDR _aBegin	Start address of the area from which memory contents will be acquired
[in]	ADDR _aEnd	End address of the area from which memory contents will be acquired
[in]	WORD _wDisplayWidth	Size in which memory is accessed (1, 2, 4, or 8 specifiable)
[out]	SAFEARRAY(BYTE)* _ppbyBuff	Memory content

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
    HRESULT   hr = E_FAIL;
    DWORD     dwAddrBegin = (DWORD)strtol(m_GetMemoryStartAddress, NULL, 16);
    DWORD     dwAddrEnd = (DWORD)strtol(m_GetMemoryEndAddress, NULL, 16);
    WORD      wDisplayWidth = (WORD)m_GetMemorySize.GetCurSel();

    if      (wDisplayWidth == 0) wDisplayWidth = 1;
    else if(wDisplayWidth == 1) wDisplayWidth = 2;
    else if(wDisplayWidth == 2) wDisplayWidth = 4;
    else if(wDisplayWidth == 3) wDisplayWidth = 8;
    else wDisplayWidth = 1;

    BYTE bTemp;

    //array for storing data obtained from HewTargetServer
    ...
    SAFEARRAY FAR*   pHewArray = NULL;

    try
    {
              hr   =   pHewServer1->GetMemory(dwAddrBegin,   dwAddrEnd,   wDisplayWidth,
&pHewArray);
    }
```

SetMemory

Description

Sets memory content according to specified start and end addresses and access size.

Parameters

Attribute	Type	Content
[in]	ADDR _aBegin	Start address of the area from which memory contents will be acquired
[in]	ADDR _aEnd	End address of the area from which memory contents will be acquired
[in]	WORD _wDisplayWidth	Size in which memory is accessed (1, 2, 4, or 8 specifiable)
[in]	SAFEARRAY(BYTE)* _ppbyBuff	Memory content

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT                 hr = E_FAIL;

dwAddrBegin = (DWORD)strtol(m_SetMemoryStartAddress, NULL, 16);
dwAddrEnd = (DWORD)strtol(m_SetMemoryEndAddress, NULL, 16);


...


try
{
          hr = pHewServer1->SetMemory(dwAddrBegin, dwAddrEnd, wDisplayWidth,
         pHewArray);
}
```

GetDirectMemory

Description

Gets memory content according to specified start and end addresses and access size. Regardless of whether the memory content of this specified area is held in the High-performance Embedded Workshop, the target memory is accessed to get the memory content to be returned.

Parameters

Attribute	Type	Content
[in]	ADDR _aBegin	Start address of the area from which memory contents will be acquired
[in]	ADDR _aEnd	End address of the area from which memory contents will be acquired
[in]	WORD _wDisplayWidth	Size in which memory is accessed (1, 2, 4, or 8 specifiable)
[out]	SAFEARRAY(BYTE)* _ppbyBuff	Memory content

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
SAFEARRAY FAR*   pHewArray = NULL;

...

try
{
            hr  =  pHewServer1->GetDirectMemory(dwAddrBegin,  dwAddrEnd,  wDisplayWidth,
&pHewArray);
}
```

5.3.4 Software Breaks

SetPCBreakPt

Description

Sets a breakpoint at a specified address and returns its handle value.

Parameters

Attribute	Type	Content
[in]	ADDR _aPCBreakAddr	Address value
[out]	BHANDLE* p_BHandle	Breakpoint handle value

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
ADDR _aPCBreakAddr;
BHANDLE Bhandle;

try
{
            hr = pHewServer1->SetPCBreakPt(dwAddr, &BHandle);
}
```

EnableBreakPt

Description

Enables or disables a breakpoint according to the handle value of the breakpoint.

Parameters

Attribute	Type	Content
[in]	BHANDLE BHandle	Breakpoint handle value
[in]	VARIANT_BOOL bEnable	Enables or disables a breakpoint according to the handle value of the breakpoint.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT             hr = E_FAIL;
BHANDLE BHandle;
VARIANT_BOOL        bEnable;

try
{
           hr = pHewServer1->EnableBreakPt(BHandle, bEnable);
}
```

DeleteBreakPt

Description

Deletes the breakpoint that has a specified breakpoint handle value.

Parameters

Attribute	Type	Content
[in]	BHANDLE BHandle	Breakpoint handle value

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT             hr = E_FAIL;
BHANDLE BHandle;

try
{
            hr = pHewServer1->DeleteBreakPt(BHandle);
}
```

GetAllBreakPt

Description

Gets the software breakpoints that have been set.

Parameters

Attribute	Type	Content
[out]	long *p_index	Number of software breakpoints
[out]	VARIANT *p_vAllBreakPt	Array of software breakpoints

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
long index;
VARIANT vAllBreakPt;
VariantInit( &vAllBreakPt );

//calling HewTargetServer function
try
{
  hr = pHewServer1->GetAllBreakPt(&index, &vAllBreakPt);
}
```

DeleteAllBreakPt

Description

Deletes the software breakpoints that have been set.

Parameters

There is no parameter.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT hr = E_FAIL;

//calling HewTargetServer function
try
{
        hr = pHewServer1->DeleteAllBreakPt();
}
```

5.3.5 Variable Break

SetDataBreakpoint

Description

Sets a data breakpoint.

Parameters

Attribute	Type	Content
[in]	DWORD _aSymbol	Symbol address
[in]	DWORD _eSize	Symbol size (1/2/4) 0x00000001 - 1 0x00000002 - 2 0x00000004 - 4
[in]	DWORD _eType	Type of break (Equal/Not Equal) 0x00000001 - Equal 0x00000002 - Not Equal
[in]	DWORD _dwData	Symbol value
[out]	DWORD *p_dwBreakDataNo	Variable break No.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
DWORD _aSymbol;
DWORD _eSize;
DWORD _eType;
DWORD _dwData;
DWORD _dwBreakDataNo;

...

try
{
            hr = pHewServer1->SetDataBreakpoint(_aSymbol, _eSize, _eType, _dwData,
&_dwBreakDataNo);
}
```

EnableDataBreakpoint

Description

Enables or disables a data breakpoint.

Parameters

Attribute	Type	Content
[in]	DWORD dwBreakDataNo	Variable break No.
[in]	VARIANT_BOOL _bEnable	Enabled (True)/ Disabled (False)

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
DWORD dwBreakDataNo;
VARIANT_BOOL        bEnable;

try
{
            hr = pHewServer1->EnableDataBreakpoint(dwBreakDataNo, bEnable);
}
```

DeleteDataBreakpoint

Description

Deletes the data breakpoint.

Parameters

Attribute	Type	Content
[in]	DWORD dwBreakDataNo	Variable break No.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
DWORD   dwBreakDataNo;

try
{
            hr = pHewServer1->DeleteDataBreakpoint(dwBreakDataNo);
}
```

5.3.6 Variable Trace

SetSymbolTrace

Description

Sets variable trace conditions.

Parameters

Attribute	Type	Content
[in]	ADDR _aSymbol	Symbol address
[in]	DWORD _eCondition	Trace condition (Read/Write) 0x00000001 - Read 0x00000002 - Write 0x00000003 - Read_Write
[in]	DWORD _eSize	Symbol size (1/2/4) 0x00000001 - 1 0x00000002 - 2 0x00000004 - 4
[in]	DWORD _eType	Type of trace (Equal/Not Equal/No Specific) 0x00000001 - Equal 0x00000002 - Not Equal 0x00000003 - Not Specified
[in]	DWORD _dwData	Symbol value
[out]	DWORD * p_dwTraceNo	Variable trace No.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
ADDR _aSymbol;
DWORD _eSize;
DWORD _eType;
DWORD _dwData;
DWORD _dwTraceNo;

 ...

try
{
            hr = pHewServer1->SetSymbolTrace(_aSymbol, 0x00000001, _eSize, _eType, _dwData,
  &_dwTraceNo);
}
```

ExecuteSymbolTrace

Description

Enables or disables variable trace.

Parameters

Attribute	Type	Content
[in]	VARIANT_BOOL _bEnable	Enabled (True)/ Disabled (False)

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
VARIANT_BOOL _bEnable;

try
{
            hr = pHewServer1->ExecuteSymbolTrace(_bEnable);
}
```

DeleteSymbolTrace

Description

Deletes variable trace conditions.

Parameters

Attribute	Type	Content
[in]	DWORD _dwTraceNo	Variable trace No. to be deleted

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
DWORD _dwTraceNo;

try
{
            hr = pHewServer1->DeleteSymbolTrace(_dwTraceNo);
}
```

SaveSymbolTraceData

Description

Saves the result of variable trace to a specified file.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrFileName	File in which variable trace data is saved

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT                hr = E_FAIL;
BSTR bstrSaveSymbolTraceData;

try
{
            hr = pHewServer1->SaveSymbolTraceData(bstrSaveSymbolTraceData);
}
```

Example of an output format

The trace result consists of the following contents which are separated by a space when output.

- Accessed time (in cycles for simulator)
- Accessed address
- Access attribute (Read/Write/Read_Write)
- Access value
- Access size

Sample

```
1287539 0XFFFE5DC Write 0XEA 1
1287553 0XFFFE5DC Write 0X30 1
1288170 0XFFFE5DC Write 0XEA 1
1445327 0XFFFE5DC Write 0XE0 1
1445341 0XFFFE5DC Write 0X30 1
1445958 0XFFFE5DC Write 0XE0 1
1605377 0XFFFE5DC Write 0X4C 1
1605391 0XFFFE5DC Write 0X30 1
1606008 0XFFFE5DC Write 0X4C 1
1760876 0XFFFE5DC Write 0XF6 1
```

5.3.7 Interrupt Condition

SendTrigger

Description

Sets trigger conditions.

Parameters

Attribute	Type	Content
[in]	DWORD _dwTriggerNo	Trigger No.
[in]	DWORD _dwTriggerType1	Trigger interrupt condition 1
[in]	DWORD _dwTriggerType2	Trigger interrupt condition 2
[in]	DWORD _dwPriority	Interrupt priority (0-17)

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;

DWORD _dwTriggerNo;
DWORD _dwTriggerType1;
DWORD _dwTriggerType2;
DWORD _dwPriority

try
{
            hr = pHewServer1->SendTrigger(
                      _dwTriggerNo,
                      _dwTriggerType1,
                      _dwTriggerType2,
                      _dwPriority
            );
}
```

5.3.8 Symbol

GetRealTimeWatch

Description

Gets the specified data value.

Parameters

Attribute	Type	Content
[in]	ADDR _aSymbol	Symbol address
[in]	DWORD _eSize	Symbol size (1/2/4) 0x00000001 - 1 0x00000002 - 2 0x00000004 - 4
[out]	DWORD *p_dwValue	Symbol value

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULThr = E_FAIL;
ADDR _aSymbol;
DWORD _eSize;
DWORD p_dwValue;

try
{
                    hr = GetRealTimeWatch( aSymbol, eSize, &p_dwValue);
}
```

GetQuickWatch

Description

Gets the variable size, variable value, type, and allocated area from the variable name.

Parameters

Attribute	Type	Content
[in]	BSTR bstrVarName	Variable name
[out]	DWORD* p_dwValueSize	Variable size
[out]	BSTR* bstrByValue	String of variable value
[out]	EObjectTypeServer* p_eType	Variable type
[out]	BSTR* bstrTypeName	String of variable type
[out]	BSTR* bstrVariableAllocation	String of allocated variable area

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
BSTR bstrVarName;
DWORD _dwValueSize;
BSTR bstrByValue;
EObjectTypeServer _eType;
BSTR bstrTypeName;
BSTR bstrVariableAllocation;

try
{
            hr = pHewServer1->GetQuickWatch(bstrVarName,
                      &_dwValueSize,
                      & bstrByValue,
                      &_eType,
                      & bstrTypeName,
                      & bstrVariableAllocation
            );
}
```

SymbolToAddress

Description

Converts label/symbol from a symbol name to its corresponding address value.

Parameters

Attribute	Type	Content
[in]	BSTR bstrSymbolName	Symbol name
[out]	ADDR* p_aSymbolAddr	Symbol address

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
BSTR bstrSymbolName;
ADDR _aSymbolAddr;

try
{
                hr = pHewServer1->SymbolToAddress(bstrSymbolName, &_aSymbolAddr);
}
```

AddressToSymbol

Description

Converts label/symbol from an address value to its corresponding symbol name.

Parameters

Attribute	Type	Content
[in]	ADDR aSymbolAddr	Address value
[out]	BSTR* p_bstrSymbolName	Symbol name

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
ADDR aSymbolAddr;
BSTR         bstrSymbolName;

try
{
             hr = pHewServer1->AddressToSymbol(aSymbolAddr, &bstrSymbolName);
}
```

GetLineFromAddr

Description

Converts label/symbol from an address value to its corresponding file and line.

Parameters

Attribute	Type	Content
[in]	ADDR _aLineAddr	Line address
[out]	BSTR* p_bstrFileName	File name
[out]	LINENO* p_LineNo	Line number

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
ADDR _aLineAddr;
BSTR bstrFileName;
LINENO _LineNo;

try
{
            hr = pHewServer1->GetLineFromAddr(_aLineAddr, &bstrFileName, &_LineNo);
}
```

GetAddrFromLine

Description

Converts a label/symbol from file and line to its corresponding address value.

Parameters

Attribute	Type	Content
[in]	BSTR bstrFileName	File name
[in]	LINENO LineNo	Line number
[out]	ADDR* p_aLineAddr	Line address

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
BSTR bstrFileName;
LINENO LineNo;
ADDR _aLineAddr;

try
{
            hr = pHewServer1->GetAddrFromLine(bstrFileName,LineNo,&_aLineAddr);
}
```

5.3.9 Downloads

Download

Description

Downloads a load module.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrFileName	Load module (including path name)

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
BSTR      bstrDownloadFile;

try
{
          hr = pHewServer1->Download(bstrDownloadFile);
}
```

Unload

Description

Unloads a load module.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrFileName	Unload module (including path name)

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
BSTR bstrUnloadFile;

try
{
            hr = pHewServer1->Unload(bstrUnloadFile);
}
```

5.3.10 Start/Stop

InvokeHew

Description

Starts a High-performance Embedded Workshop application. (Workspace is not opened.)

Parameters

There is no parameter.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;

try
{
        hr = pHewServer1->InvokeHew();
}
```

QuitHew

Description

Terminates a High-performance Embedded Workshop application.

Parameters

There is no parameter.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;

try
{
          hr = pHewServer1->QuitHew();
}
```

OpenWorkspace

Description

Opens a workspace.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrFileName	File name (including path name)

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
BSTR bstrOpenWorkspace;

try
{
            hr = pHewServer1->OpenWorkspace(bstrOpenWorkspace);
}
```

CloseWorkspace

Description

Closes a workspace.

Parameters

Attribute	Type	Content
[in]	DWORD _dwIgnoreChanges	0x00000000:Workspace is not closed when changed 0x00000001:Workspace is closed without saving changes

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL
DWORD _dwIgnoreChanges;

try
{
            hr = pHewServer1->CloseWorkspace(_dwIgnoreChanges);
}
```

SaveWorkspace

Description

Saves a workspace.

Parameters

There is no parameter.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;

try
{
            hr = pHewServer1->SaveWorkspace();
 }
```

SaveSession

Description

Saves a session file.

Parameters

There is no parameter.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL

try
{
          hr = pHewServer1->SaveSession();
 }
```

GetCurrentConfiguration

Description

Gets the current build configuration.

Parameters

Attribute	Type	Content
[out]	BSTR *p_bstrCurrentConfigurationName	Name of the build configuration

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
CString       strCurrentConfigurationName = _T("");
BSTR bstrCurrentConfigurationName = strCurrentConfigurationName.AllocSysString();

//calling HewTargetServer function
try
{
  hr = pHewServer1->GetCurrentConfiguration(&bstrCurrentConfigurationName);
  strCurrentConfigurationName = bstrCurrentConfigurationName;
}
```

SetCurrentConfiguration

Description

Sets a currently active build configuration.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrConfiguration	Build configuration name

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
BSTR bstrSetCurrentConfiguration;

try
{
            hr = pHewServer1->SetCurrentConfiguration(bstrSetCurrentConfiguration);
 }
```

GetConfigurations

Description

Gets all build configurations that have a project in each.

Parameters

Attribute	Type	Content
[out]	BSTR *p_bstrConfigurations	Build configuration name (multiple names, if any, are separated by a comma) (Example) "DefaultSession, SimSessionSH-4"

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT	hr = E_FAIL;
Cstring	strTmp = _T("");
BSTR o1 = strTmp.AllocSysString();	//CString -> BSTR converted

//calling HewTargetServer function
CString so1;
try
{
		hr = pHewServer1->GetConfigurations(&o1);
		so1 = o1;
}
```

GetCurrentSession

Description

Gets the current debug session.

Parameters

Attribute	Type	Content
[out]	BSTR *p_bstrCurrentSessionName	Name of the debug session

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
CString strCurrentSessionName = _T("");
BSTR bstrCurrentSessionName = strCurrentSessionName.AllocSysString();

//calling HewTargetServer function
try
{
  hr = pHewServer1->GetCurrentSession(&bstrCurrentSessionName);
  strCurrentSessionName = bstrCurrentSessionName;
}
```

SetCurrentSession

Description

Sets a currently active debug session.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrSession	Debug session name

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
        BSTR   bstrSetCurrentSession   =   m_SetCurrentSession.AllocSysString();     //CString   ->   BSTR
converted
        HRESULT   hr = E_FAIL;

        //calling HewTargetServer function
        try
        {
                    hr = pHewServer1->SetCurrentSession(bstrSetCurrentSession);
        }
```

GetSessions

Description

Gets all debug sessions that are included in a project.

Parameters

Attribute	Type	Content
[out]	BSTR *p_bstrSessions	Debug session name (multiple names, if any, are separated by a comma) (Example) "DefaultSession, SimSessionSH-4"

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULThr = E_FAIL;

//calling HewTargetServer function
CString strTmp = _T("");
BSTR o2 = strTmp.AllocSysString();    //CString -> BSTR converted

CString so2;
try
{
            hr = pHewServer1->GetSessions(&o2);
            so2 = o2;
}
```

GetCurrentProject

Description

Gets the current project.

Parameters

Attribute	Type	Content
[out]	BSTR *p_bstrCurrentProjectName	Name of the project

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
    HRESULT   hr = E_FAIL;
    CString strCurrentProjectName = _T("");
    BSTR bstrCurrentProjectName = strCurrentProjectName.AllocSysString();

    //calling HewTargetServer function
    try
    {
      hr = pHewServer1->GetCurrentProject(&bstrCurrentProjectName);
      strCurrentProjectName = bstrCurrentProjectName;
  }
```

SetCurrentProject

Description

Enables a specified project to make it active.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrProjectName	Project name

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT hr = E_FAIL;

BSTR bstrSetCurrentProject = m_SetCurrentProject.AllocSysString();
//calling HewTargetServer function
try
{
                    hr = pHewServer1->SetCurrentProject(bstrSetCurrentProject);
}
```

GetProjects

Description

Gets all project names.

Parameters

Attribute	Type	Content
[out]	BSTR *p_bstrProjectNames	Project name. If there are two or more project names, they should be delimited by a comma. Example: "Project1, Project2"

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
CString      strProjectNames = _T("");
BSTR bstrProjectNames = strProjectNames.AllocSysString();

//calling HewTargetServer function
try
{
  hr = pHewServer1->GetProjects(&bstrProjectNames);
  strProjectNames = bstrProjectNames;
}
```

5.3.13 Project

AddFile

Description

Adds a file to the currently active project.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrFileName	File name (including path name)

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
BSTR bstrAddFiles;

try
{
            hr = pHewServer1->AddFile(bstrAddFiles);
}
```

AddFiles

Description

Adds multiple files to the currently active project.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrFileName	File name (multiple names, if any, are separated by a comma) (including path name)

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
BSTR bstrAddFile;

try
{
            hr = pHewServer1->AddFiles(bstrAddFile);
}
```

DeleteFile

Description

Deletes a file from the currently active project.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrFileName	File name (including path name)

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT              hr = E_FAIL

BSTR bstrDeleteFile = m_DeleteFile.AllocSysString();
//calling HewTargetServer function
try
{
                     hr = pHewServer1->DeleteFile(bstrDeleteFile);
}
```

DeleteFiles

Description

Deletes multiple files from the currently active project.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrFileName	File name (multiple names, if any, are separated by a comma) (including path name)

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT               hr = E_FAIL

BSTR bstrDeleteFiles = m_DeleteFiles.AllocSysString();
//calling HewTargetServer function
try
{
                      hr = pHewServer1->DeleteFiles(bstrDeleteFiles);
}
```

5.3.14 Build

BuildProject

Description

Builds a project.

Parameters

There is no parameter.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;

//calling HewTargetServer function
try
{
            hr = pHewServer1->BuildProject();
}
```

RebuildProject

Description

Rebuilds a project.

Parameters

There is no parameter.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;

//calling HewTargetServer function
try
{
            hr = pHewServer1->RebuildProject();
}
```

UpDateAllDependency

Description

Updates all dependency relations.

Parameters

There is no parameter.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT hr = E_FAIL;
//calling HewTargetServer function
try
{
                    hr = pHewServer1->UpDateAllDependency();
}
```

AddFileWithCompilerOption

Description

Adds a file after setting compiler options for the project.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrFileName	File name (including path name)
[in]	BSTR _bstrIncludeDirectories	Include directory name. If there are two or more directories, they should be delimited by a comma. Example: "C:\tmp, D:\work"
[in]	BSTR _bstrDefines	Definition. If there are two or more definitions, they should be delimited by a comma. Example: "TMP1=C:\tmp, TMP2=D:\work"

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
BSTR bstrFileName;
BSTR bstrIncludeDirectories;
BSTR bstrDefines;

//calling HewTargetServer function
try
{
  hr = pHewServer1->AddFileWithCompilerOption(bstrFileName, bstrIncludeDirectories, bstrDefines);
}
```

OpenFileAtLine

Description

Opens a file by specifying the file name and line number.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrOpenFileName	File name (including path name)
[in]	int _iLine	Line number

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
CString     strOpenFileName = _T("");
BSTR bstrOpenFileName = strOpenFileName.AllocSysString();
int iLine = 1;

//calling HewTargetServer function
try
{
  hr = pHewServer1->OpenFileAtLine(bstrOpenFileName, iLine);
}
```

GetSourceFiles

Description

Gets all source file names (such as *.cpp or *.src) in a project.
The file name is output as an absolute path.

Parameters

Attribute	Type	Content
[out]	BSTR *p_bstrSourceFiles	Source file names (if there are two or more file names, they should be delimited by a comma). Example: "c:\sample1.cpp, c:\sample2.cpp"

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
CString     strSourceFiles = _T("");
BSTR bstrSourceFiles = strSourceFiles.AllocSysString();

//calling HewTargetServer function
try
{
  hr = pHewServer1->GetSourceFiles(&bstrSourceFiles);
  strSourceFiles = bstrSourceFiles;
}
```

GetDownloadModules

Description

Gets all module file names (such as *.abs) in a project.
The file name is output as an absolute path.

Parameters

Attribute	Type	Content
[out]	BSTR *p_bstrDownloadModules	Module file names (if there are two or more file names, they should be delimited by a comma). Example: "c:\sample1.abs, c:\sample2.abs"

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
CString      strDownloadModules = _T("");
BSTR bstrDownloadModules = strDownloadModules.AllocSysString();

//calling HewTargetServer function
try
{
  hr = pHewServer1->GetDownloadModules(&bstrDownloadModules);
  strDownloadModules = bstrDownloadModules;
}
```

GetDependentFiles

Description

Gets all dependent file names (such as *.h or *.inc) in a project.
The file name is output as an absolute path.

Parameters

Attribute	Type	Content
[out]	BSTR *p_bstrDependentFiles	Dependent file names (if there are two or more file names, they should be delimited by a comma). Example: "c:\sample1.h, c:\sample2.h"

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
CString      strDependentFiles = _T("");
BSTR bstrDependentFiles = strDependentFiles.AllocSysString();

//calling HewTargetServer function
try
{
  hr = pHewServer1->GetDependentFiles(&bstrDependentFiles);
  strDependentFiles = bstrDependentFiles;
}
```

5.3.16 Coverage

SetCoverageRange

Description

Sets a coverage range.

Parameters

Attribute	Type	Content
[in]	DWORD dwStartAddress	Start address
[in]	DWORD dwEndAddress	End address

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
DWORD                    dwStartAddress;
DWORD                    dwEndAddress;

//calling HewTargetServer function
try
{
  hr = pHewServer1->SetCoverageRange(dwStartAddress, dwEndAddress);
}
```

Precautions

The coverage facility is enabled as soon as a coverage range is set.

GetCoverageRange

Description

Gets data from a coverage range.

Parameters

Attribute	Type	Content
[out]	DWORD *p_dwStartAddress	Start address
[out]	DWORD *p_dwEndAddress	End address

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
DWORD                   dwStartAddress;
DWORD                   dwEndAddress;

//calling HewTargetServer function
try
{
  hr = pHewServer1->GetCoverageRange(&dwStartAddress, &dwEndAddress);
}
```

SetCoverageDisable

Description

Disables the coverage function.

Parameters

There is no parameter.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT	hr = E_FAIL;

//calling HewTargetServer function
try
{
		hr = pHewServer1->SetCoverageDisable();
}
```

SetCoverageEnable

Description

Enables the coverage function.

Parameters

There is no parameter.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT hr = E_FAIL;

//calling HewTargetServer function
try
{
        hr = pHewServer1->SetCoverageEnable();
}
```

ClearCoverage

Description

Clears the coverage information.

Parameters

There is no parameter.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT hr = E_FAIL;

//calling HewTargetServer function
try
{
        hr = pHewServer1->ClearCoverage();
}
```

GetCoverageStatus

Description

Gets the coverage status information.

Parameters

Attribute	Type	Content
[out]	int *p_iStatus	Coverage status (1: Enabled or 0: Disabled)

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
int iStatus;

//calling HewTargetServer function
try
{
  hr = pHewServer1->GetCoverageStatus(&iStatus);
}
```

LoadCoverage

Description

Loads the coverage information.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrLoadFileName	File name

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
BSTR bstrLoadFileName;

//calling HewTargetServer function
try
{
  hr = pHewServer1->LoadCoverage(bstrLoadFileName);
}
```

SaveCoverage

Description

Saves the coverage information.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrSaveFileName	File name

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
BSTR bstrSaveFileName;

//calling HewTargetServer function
try
{
  hr = pHewServer1->SaveCoverage(bstrSaveFileName);
}
```

5.3.17 Others

GetErrorString

Description

Gets an error message corresponding to a specified error number.

Parameters

Attribute	Type	Content
[in]	HRESULT _lError	Error number
[out]	BSTR* _pbstrError	Error message

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hrErr = E_FAIL;
HRESULT _lError;
BSTR        bstrErrStr;

try
{
                hrErr = pHewServer1->GetErrorString(_lError, &bstrErrStr);
}
```

GetHewStatus

Description

Gets the current High-performance Embedded Workshop status.

Parameters

Attribute	Type	Content
[out]	int* p_iTargetReset	Returns 1 when the target is reset or 0 otherwise*
[out]	int* p_iTargetExecStatus	Returns 1 when the user program is under execution or 0 otherwise
[out]	int* p_iMemoryReset	Returns 1 when memory contents are updated or 0 otherwise*
[out]	int* p_iRegisterReset	Returns 1 when register values are updated or 0 otherwise*
[out]	int* p_iLInkStatus	Returns 1 when the target is connected or 0 otherwise
[out]	int* p_iPlatformInitialize	Returns 1 after the target is initialized or 0 otherwise*
[out]	int* p_iLoadingStatus	Returns 1 after a program is loaded or 0 otherwise

*: These flags are reset to 0 when this function is called.

Returned value

The returned value is 1 when the method was terminated successfully or 0 when there is error.

Description example

```
int         iTargetReset;
int         iTargetExecStatus;
int         iMemoryReset;
int         iRegisterReset;
int         iLinkStatus;
int         iPlatformInitialize;
int         iLoadingStatus;
HRESULT     hr;

//calling HewTargetServer function
try
{
            hr = pHewServer1->GetHewStatus(&iTargetReset, &iTargetExecStatus,
                        &iMemoryReset,
                        &iRegisterReset,
                        &iLinkStatus,
                        &iPlatformInitialize,
                        &iLoadingStatus
            );
}
```

GetHewStatusEx

Description

Gets the High-performance Embedded Workshop status information (on initiation, opening a workspace, and build).

Parameters

Attribute	Type	Content
[out]	int *p_iInvokeHew	Initiation of the High-performance Embedded Workshop (0: Not initiated or 1: Initiated)
[out]	int *p_iOpenWorkspace	Opening of a workspace (0: Not open or 1: Open) Note: The acquired value can be 1 only when the HEW is connected to the target. To check whether a workspace is open, call GetWorkspaceDirectory instead.
[out]	int *p_iBuildProject	Build (0: Build stopped or 1: Build being performed)

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
int iInvokeHew;
int iOpenWorkspace;
int iBuildProject;

//calling HewTargetServer function
try
{
  hr = pHewServer1->GetHewStatusEx(&iInvokeHew, &iOpenWorkspace, &iBuildProject);
}
```

GetTargetName

Description

Gets the target name that is currently connected.

Parameters

Attribute	Type	Content
[out]	BSTR* p_bstrName	Target name

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example

```
HRESULT   hr = E_FAIL;
  BSTR      bStrTargetName;

//calling HewTargetServer function
try
{
            //get target name
            hr = pHewServer1->GetTargetName(&bStrTargetName);
}
```

5.4 Method Details (for VB, VC++)

5.4.1 CPU Control

GoTargetExec2

Description

Executes a program from the current program position.

Parameters

There is no parameter.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT hr = E_FAIL;

try
{

        hr = pHewServer1->GoTargetExec2();

}
```

Description example (Visual Basic)

```
Dim ret As Long

ret = hts.GoTargetExec2
```

StopTargetExec2

Description

Stops program execution.

Parameters

There is no parameter.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT hr = E_FAIL;

try
{
        hr = pHewServer1->StopTargetExec2();
}
```

Description example (Visual Basic)

```
Dim ret As Long

ret = hts.StopTargetExec2
```

ResetTargetExec2

Description

Resets the debugger environment that is run as the target.

Parameters

There is no parameter.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT hr = E_FAIL;

try
{
            hr = pHewServer1->ResetTargetExec2();
}
```

Description example (Visual Basic)

```
Dim ret As Long

ret = hts.ResetTargetExec2
```

InitializeTarget2

Description

Initializes the debugger environment that is run as the target.

Parameters

There is no parameter.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT hr = E_FAIL;

try
{
        hr = pHewServer1->InitializeTarget2();
}
```

Description example (Visual Basic)

```
Dim ret As Long

ret = hts.InitializeTarget2
```

Step2

Description

Step executes the target program.

Parameters

Attribute	Type	Content
[in]	long _lMode	Description 0x00000001 Steps through assembler instructions 0x00000002 Steps through source code lines
[in]	long _lStep	Number of steps executed

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
DWORD _lMode = 1;              //assembler:1, source:2
DWORD _lStep = 1;

try
{
            hr = pHewServer1->Step2(_lMode, _lStep);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim lMode As Long
Dim lStep As Long
lMode = 1
lStep = 1

ret = hts.Step2(lMode, lStep)
```

StepRate2

Description

Sets a speed at which the program is single-stepped.

Parameters

Attribute	Type	Content
[in]	long _lRate	Set a stepping rate in the range 0-6.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT                hr = E_FAIL;
int _lRate;

try
{
           hr = pHewServer1->StepRate2(_lRate);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim lRate As Long
lRate = 0

ret = hts.StepRate2(lRate)
```

StepOver2

Description

Runs a program by stepping-over instructions or source lines.

Parameters

Attribute	Type	Content
[in]	long _lMode	Description 0x00000001 Steps through assembler instructions 0x00000002 Steps through source code lines
[in]	long _lStep	Number of steps executed

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
long _lMode = 1;                    //assembler:1, source:2
long _lStep = 1;

try
{
            hr = pHewServer1->StepOver2(_lMode, _lStep);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim lMode As Long
Dim lStep As Long
lMode = 1
lStep = 1

ret = hts.StepOver2(lMode, lStep)
```

StepOut2

Description

Runs a program by stepping-out instructions or source lines.

Parameters

Attribute	Type	Content
[in]	long _lMode	Description 0x00000001 Steps through assembler instructions 0x00000002 Steps through source code lines

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
long _lMode = 1;                    //assembler:1, source:2

try
{
            hr = pHewServer1->StepOut2(_lMode);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim lMode As Long
lMode = 1

ret = hts.StepOut2(lMode)
```

IsRunning2

Description

Determines whether or not the user program is running.

Parameters

Attribute	Type	Content
[out]	long* p_lRunning	1 when the user program is running or 0 otherwise

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
long _lRunning;

try
{
              hr = pHewServer1->IsRunning2(&_lRunning);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim p_lRunning As Long      'In Visual Basic 2005, "As Long" will be replaced with "As Integer".

ret = hts.IsRunning2(p_lRunning)
```

GetPC2

Description

Gets the program counter value.

Parameters

Attribute	Type	Content
[out]	long *p_lPC	PC (program counter) value

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
long _lPC = 0x0;

try
{
            hr = pHewServer1->GetPC2(&_lPC);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim p_lPC As Long              'In Visual Basic 2005, "As Long" will be replaced with "As Integer".
p_lPC = 0

ret = hts.GetPC2(p_lPC)
```

SetPCAddress2

Description

Sets the program counter.

Parameters

Attribute	Type	Content
[in]	long lPCAddr	PC (program counter) value to be set

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
long lPCAddr = 0x800;

try
{
            hr = pHewServer1->SetPCAddress2(lPCAddr);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim lPCAddr As Long
lPCAddr = &H800

ret = hts.SetPCAddress2(lPCAddr)
```

SetPCSource2

Description

Sets the program counter by specifying a source file and line.

Parameters

Attribute	Type	Content
[in]	BSTR bstrFileName	File name
[in]	long lLineNum	Line number

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT     hr = E_FAIL;
BSTR bstrFileName;
long lLineNum = 100;

try
{
            hr = pHewServer1->SetPCSource2(bstrFileName, lLineNum);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim bstrFileName As String
Dim lLineNum As Long
lLineNum = 100

ret = hts. SetPCSource2(bstrFileName, lLineNum)
```

TestSetPC2

Description

Determines whether or not the program counter value can be set.

Parameters

Attribute	Type	Content
[out]	long *p_lSetPCState	1 when the PC value can be set or 0 otherwise

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
long      lSetPCState = 0;

try
{
          hr = pHewServer1->TestSetPC2(&lSetPCState);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim p_lSetPCState As Long    'In Visual Basic 2005, "As Long" will be replaced with "As Integer".
p_lSetPCState = 100

ret = hts.TestSetPC2(p_lSetPCState)
```

5.4.3 Memory

GetMemory2

Description

Gets memory content according to specified start and end addresses and access size. If the memory content of this specified area is held in the High-performance Embedded Workshop, the memory content which is added in the High-performance Embedded Workshop is returned directly without accessing the target memory.

Parameters

Attribute	Type	Content
[in]	long lBegin	Start address of the area from which memory contents will be acquired
[in]	long lEnd	End address of the area from which memory contents will be acquired
[in]	long lDisplayWidth	Size in which memory is accessed (1, 2, 4, or 8 specifiable)
[out]	VARIANT vMemData	Memory content

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
long      lBegin = strtol(m_GetMemoryStartAddress, NULL, 16);
long      lEnd = strtol(m_GetMemoryEndAddress, NULL, 16);
long      lDisplayWidth = m_GetMemorySize.GetCurSel();

//array for string data obtained from HewTargetServer
...

try
{
          hr = pHewServer1->GetMemory2(lBegin, lEnd, lDisplayWidth, &vMemData);
}
```

Description example (Visual Basic 6.0)

```
Dim ret As Long
Dim lBegin As Long
Dim lEnd As Long
Dim lDisplayWidth As Long
Dim vMemData As Variant

...

ret = hts.GetMemory2(lBegin, lEnd, lDisplayWidth, vMemData)
```

Description example (Visual Basic 2005)

```
Dim ret As Integer
Dim iBegin As Integer
Dim iEnd As Integer
Dim iDisplayWidth As Integer
Dim vMemData As Object

...

ret = hts.GetMemory2(iBegin, iEnd, iDisplayWidth, vMemData)
```

SetMemory2

Description

Sets memory content according to specified start and end addresses and access size.

Parameters

Attribute	Type	Content
[in]	long lBegin	Start address of the area from which memory contents will be acquired
[in]	long lEnd	End address of the area from which memory contents will be acquired
[in]	long lDisplayWidth	Size in which memory is accessed (1, 2, 4, or 8 specifiable)
[in]	VARIANT vMemData	Memory content

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT                hr = E_FAIL;

lBegin = strtol(m_SetMemoryStartAddress, NULL, 16);
lEnd = strtol(m_SetMemoryEndAddress, NULL, 16);

...

long length = lEnd - lBegin + 1;
long *plDataArray;
SAFEARRAY* psaData;
VARIANT vMemData;
VARIANT *p_vMemData = &vMemData;

SAFEARRAYBOUND bounds = {length, 0};
VariantInit(p_vMemData);
p_vMemData->vt = VT_ARRAY | VT_I4;
psaData = SafeArrayCreate(VT_I4, 1, &bounds);
SafeArrayAccessData(psaData, (void**)&plDataArray);

for (long j = 0 ; j < length ; j++ ) {
    CString strWork;
    // e.g. Set 0x00, 0x01, 0x02, 0x03, 0x00, 0x01, ...
    plDataArray[j] = j % 4;
}
SafeArrayUnaccessData(psaData);
p_vMemData->parray = psaData;

try
{
            hr = pHewServer1->SetMemory2(lBegin, lEnd, lDisplayWidth, vMemData);
}
...
SafeArrayDestroy(psaData);
```

Description example (Visual Basic 6.0)

```
Dim ret As Long
Dim lBegin As Long
Dim lEnd As Long
Dim lDisplayWidth As Long
Dim i As Long
Dim length As Long
Dim vMemData As Variant
Dim memData(65535) As Long
...

length = lEnd - lBegin + 1
For i = 0 To length - 1
    ' e.g. Set 0x00, 0x01, 0x02, 0x03, 0x00, 0x01, ...
    memData(i) = i Mod 4
Next i
vMemData = memData
...

ret = hts.SetMemory2(lBegin, lEnd, lDisplayWidth, vMemData)
```

Description example (Visual Basic 2005)

```
Dim ret As Integer
Dim iBegin As Integer
Dim iEnd As Integer
Dim iDisplayWidth As Integer
Dim i As Integer
Dim length As Integer
Dim vMemData As Object
Dim memData(65535) As Integer
...

length = iEnd - iBegin + 1
For i = 0 To length - 1
    ' e.g. Set 0x00, 0x01, 0x02, 0x03, 0x00, 0x01, ...
    memData(i) = i Mod 4
Next i
vMemData = memData
...

ret = hts.SetMemory2(iBegin, iEnd, iDisplayWidth, vMemData)
```

GetDirectMemory2

Description

Gets memory content according to specified start and end addresses and access size. Regardless of whether the memory content of this specified area is held in the High-performance Embedded Workshop, the target memory is accessed to get the memory content to be returned.

Parameters

Attribute	Type	Content
[in]	long lBegin	Start address of the area from which memory contents will be acquired
[in]	long lEnd	End address of the area from which memory contents will be acquired
[in]	long lDisplayWidth	Size in which memory is accessed (1, 2, 4, or 8 specifiable)
[out]	VARIANT *p_vMemData	Memory content

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
long      lBegin = strtol(m_GetMemoryStartAddress, NULL, 16);
long      lEnd = strtol(m_GetMemoryEndAddress, NULL, 16);
long      lDisplayWidth = m_GetMemorySize.GetCurSel();

//array for string data obtained from HewTargetServer
...

try
{
          hr = pHewServer1->GetDirectMemory2(lBegin, lEnd, lDisplayWidth, &vMemData);
}
```

Description example (Visual Basic 6.0)

```
Dim ret As Long
Dim lBegin As Long
Dim lEnd As Long
Dim lDisplayWidth As Long
Dim vMemData As Variant

...

ret = hts.GetDirectMemory2(lBegin, lEnd, lDisplayWidth, vMemData)
```

Description example (Visual Basic 2005)

```
Dim ret As Integer
Dim iBegin As Integer
Dim iEnd As Integer
Dim iDisplayWidth As Integer
Dim vMemData As Object

...

ret = hts.GetDirectMemory2(iBegin, iEnd, iDisplayWidth, vMemData)
```

5.4.4 Software Breaks

SetPCBreakPt2

Description

Sets a breakpoint at a specified address and returns its handle value.

Parameters

Attribute	Type	Content
[in]	long lPCBreakAddr	Address value
[out]	long *p_lHandle	Breakpoint handle value

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
long      lPCBreakAddr;
long      lHandle;

try
{
          hr = pHewServer1->SetPCBreakPt2(lPCBreakAddr, &lHandle);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim lPCBreakAddr As Long
Dim p_lHandle As Long       'In Visual Basic 2005, "As Long" will be replaced with "As Integer".


ret = hts.SetPCBreakPt2(lPCBreakAddr, p_lHandle)
```

EnableBreakPt2

Description

Enables or disables a breakpoint according to the handle value of the breakpoint.

Parameters

Attribute	Type	Content
[in]	long lHandle	Breakpoint handle value
[out]	long lEnable	Enables or disables a breakpoint according to the handle value of the breakpoint.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT                hr = E_FAIL;
long lHandle;
long lEnable = 1;

try
{
           hr = pHewServer1->EnableBreakPt2(lHandle, lEnable);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim lHandle As Long
Dim lEnable As Long
lEnable = 1

ret = hts.EnableBreakPt2(lHandle, lEnable)
```

DeleteBreakPt2

Description

Deletes the breakpoint that has a specified breakpoint handle value.

Parameters

Attribute	Type	Content
[in]	long lHandle	Breakpoint handle value

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT             hr = E_FAIL;
long        lHandle;

try
{
            hr = pHewServer1->DeleteBreakPt2(lHandle);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim lHandle As Long

ret = hts.DeleteBreakPt2(lHandle)
```

GetAllBreakPt2

Description

Gets the software breakpoints that have been set.

Parameters

Attribute	Type	Content
[out]	long *p_index	Number of software breakpoints
[out]	VARIANT *p_vAllBreakPt	Array of software breakpoints

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
long index;
VARIANT vAllBreakPt;
VariantInit( &vAllBreakPt );

//calling HewTargetServer function
try
{
  hr = pHewServer1->GetAllBreakPt2(&index, &vAllBreakPt);
}
```

Description example (Visual Basic 6.0)

```
Dim ret As Long
Dim p_index As Long
Dim p_vAllBreakPt As Variant

ret = hts.GetAllBreakPt2(p_index, p_vAllBreakPt)
```

Description example (Visual Basic 2005)

```
Dim ret As Integer
Dim p_index As Integer
Dim p_vAllBreakPt As Variant

ret = hts.GetAllBreakPt2(p_index, p_vAllBreakPt)
```

DeleteAllBreakPt2

Description

Deletes the software breakpoints that have been set.

Parameters

There is no parameter.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT hr = E_FAIL;

//calling HewTargetServer function
try
{
        hr = pHewServer1->DeleteAllBreakPt2();
}
```

Description example (Visual Basic)

```
Dim ret As Long

ret = hts.DeleteAllBreakPt2
```

5.4.5 Variable Break

SetDataBreakpoint2

Description

Sets a data breakpoint.

Parameters

Attribute	Type	Content
[in]	long _lSymbol	Symbol address
[in]	long _lSize	Symbol size (1/2/4) 0x00000001 - 1 0x00000002 - 2 0x00000004 - 4
[in]	long _lType	Type of break (Equal/Not Equal) 0x00000001 - Equal 0x00000002 - Not Equal
[in]	long _lData	Symbol value
[out]	long *p_lBreakDataNo	Variable break No.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
long      _lSymbol;
long      _lSize;
long      _lType;
long      _lData;
long      _lBreakDataNo;

...

try
{
          hr = pHewServer1->SetDataBreakpoint2(_lSymbol, _lSize, _lType, _lData,
&_lBreakDataNo);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim lSymbol As Long
Dim lSize As Long
Dim lType As Long
Dim lData As Long
Dim p_lBreakDataNo As Long  'In Visual Basic 2005, "As Long" will be replaced with "As Integer".

...

ret = hts.SetDataBreakpoint2(lSymbol, lSize, lType, lData, p_lBreakDataNo)
```

EnableDataBreakpoint2

Description

Enables or disables a data breakpoint.

Parameters

Attribute	Type	Content
[in]	long lBreakDataNo	Variable break No.
[in]	long _lEnable	Enabled (True)/ Disabled (False)

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
long lDataBreakNo;
long _lEnable = 1;

try
{
            hr = pHewServer1->EnableDataBreakpoint2(lDataBreakNo, _lEnable);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim lDataBreakNo As Long
Dim lEnable As Long
lEnable = 1

ret = hts.EnableDataBreakpoint2(lDataBreakNo, lEnable)
```

DeleteDataBreakpoint2

Description

Deletes the data breakpoint.

Parameters

Attribute	Type	Content
[in]	long lBreakDataNo	Variable break No.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
long lDataBreakNo;

try
{
            hr = pHewServer1->DeleteDataBreakpoint2(lDataBreakNo);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim lDataBreakNo As Long

ret = hts.DeleteDataBreakpoint2(lDataBreakNo)
```

5.4.6 Variable Trace

SetSymbolTrace2

Description

Sets variable trace conditions.

Parameters

Attribute	Type	Content
[in]	long _lSymbol	Symbol address
[in]	long _lCondition	Trace condition (Read/Write) 0x00000001 - Read 0x00000002 - Write 0x00000003 - Read_Write
[in]	long _lSize	Symbol size (1/2/4) 0x00000001 - 1 0x00000002 - 2 0x00000004 - 4
[in]	long _lType	Type of trace (Equal/Not Equal/No Specific) 0x00000001 - Equal 0x00000002 - Not Equal 0x00000003 - Not Specified
[in]	long _lData	Symbol value
[out]	long *p_lTraceNo	Variable trace No.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
long _lSymbol;
long _lCondition;
long _lSize;
long _lType;
long _lData;
long _lTraceNo;

...

try
{
            hr = pHewServer1->SetSymbolTrace2(_lSymbol, _lCondition, _lSize, _lType, _lData, &
  _lTraceNo);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim lSymbol As Long
Dim lCondition As Long
Dim lSize As Long
Dim lType As Long
Dim lData As Long
Dim p_lTraceNo As Long        'In Visual Basic 2005, "As Long" will be replaced with "As Integer".

...

ret = hts.SetSymbolTrace2(lSymbol, lCondition, lSize, lType, lData, p_lTraceNo)
```

ExecuteSymbolTrace2

Description

Enables or disables variable trace.

Parameters

Attribute	Type	Content
[in]	long _lEnable	Enabled (True)/ Disabled (False)

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
long _lEnable = 1;

try
{
            hr = pHewServer1->ExecuteSymbolTrace2(_lEnable);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim lEnable As Long
lEnable = 1

ret = hts.ExecuteSymbolTrace2(lEnable)
```

DeleteSymbolTrace2

Description

Deletes variable trace conditions.

Parameters

Attribute	Type	Content
[in]	long _lTraceNo	Variable trace No. to be deleted

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
long _lTraceNo;

try
{
            hr = pHewServer1->DeleteSymbolTrace2(_lTraceNo);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim lTraceNo As Long

ret = hts.DeleteSymbolTrace2(lTraceNo)
```

SaveSymbolTraceData2

Description

Saves the result of variable trace to a specified file.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrFileName	File in which variable trace data is saved

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT             hr = E_FAIL;
BSTR bstrFileName;

try
{
          hr = pHewServer1->SaveSymbolTraceData2(bstrFileName);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim bstrFileName As String

ret = hts.SaveSymbolTraceData2(bstrFileName)
```

Example of an output format

The trace result consists of the following contents which are separated by a space when output.

- Accessed time (in cycles for simulator)
- Accessed address
- Access attribute (Read/Write/Read_Write)
- Access value
- Access size

Sample

```
1287539 0XFFFE5DC Write 0XEA 1
1287553 0XFFFE5DC Write 0X30 1
1288170 0XFFFE5DC Write 0XEA 1
1445327 0XFFFE5DC Write 0XE0 1
1445341 0XFFFE5DC Write 0X30 1
1445958 0XFFFE5DC Write 0XE0 1
1605377 0XFFFE5DC Write 0X4C 1
1605391 0XFFFE5DC Write 0X30 1
1606008 0XFFFE5DC Write 0X4C 1
1760876 0XFFFE5DC Write 0XF6 1
```

5.4.7 Interrupt Condition

SendTrigger2

Description

Sets trigger conditions.

Parameters

Attribute	Type	Content
[in]	long _lTriggerNo	Trigger No.
[in]	long _lTriggerType1	Trigger interrupt condition 1
[in]	long _lTriggerType2	Trigger interrupt condition 2
[in]	long _lPriority	Interrupt priority (0-17)

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
long _lTriggerNo;
long _lTriggerType1;
long _lTriggerType2;
long _lPriority;

try
{
            hr = pHewServer1->SendTrigger2(
                        _lTriggerNo,
                        _lTriggerType1,
                        _lTriggerType2,
                        _lPriority
            );
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim lTriggerNo As Long
Dim lTriggerType1As Long
Dim lTriggerType2As Long
Dim lPriority As Long

ret = hts.SendTrigger2(lTriggerNo, lTriggerType1, lTriggerType2, lPriority)
```

5.4.8 Symbol

GetRealTimeWatch2

Description

Gets the specified data value.

Parameters

Attribute	Type	Content
[in]	long _lSymbol	Symbol address
[in]	long _lSize	Symbol size (1/2/4) 0x00000001 - 1 0x00000002 - 2 0x00000004 - 4
[out]	long *p_lValue	Symbol value

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT     hr = E_FAIL;
long _lSymbol;
long _lSize;
long _lValue;

try
{
            hr = pHewServer1->GetRealTimeWatch2(_lSymbol, _lSize, &_lValue);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim lSymbol As Long
Dim lSize As Long
Dim p_lValue As Long        'In Visual Basic 2005, "As Long" will be replaced with "As Integer".

ret = hts.GetRealTimeWatch2(lSymbol, lSize, p_lValue)
```

GetQuickWatch2

Description

Gets the variable size, variable value, type, and allocated area from the variable name.

Parameters

Attribute	Type	Content
[in]	BSTR bstrVarName	Variable name
[out]	long *p_lValueSize	Variable size
[out]	BSTR *bstrByValue	String of variable value
[out]	long *p_lType	Variable type
[out]	BSTR *bstrTypeName	String of variable type
[out]	BSTR *bstrVarAllocation	String of allocated variable area

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
BSTR      bstrVarName;
long      lValueSize;
BSTR      bstrByValue;
long      lType;
BSTR      bstrTypeName;
BSTR      bstrVarAllocation;

try
{
          hr = pHewServer1->GetQuickWatch2(bstrVarName,
                    &lValueSize,
                    &bstrByValue,
                    &lType,
                    &bstrTypeName,
                    &bstrVarAllocation
          );
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim bstrVarName As String
Dim p_lValueSize As Long     'In Visual Basic 2005, "As Long" will be replaced with "As Integer".
Dim p_bstrByValue As String
Dim p_lType As Long          'In Visual Basic 2005, "As Long" will be replaced with "As Integer".
Dim p_bstrTypeName As String
Dim p_bstrVarAllocation As String

ret = hts.GetQuickWatch2(bstrVarName, p_lValueSize, p_bstrByValue, p_lType, _
          p_bstrTypeName, p_bstrVarAllocation)
```

SymbolToAddress2

Description

Converts label/symbol from a symbol name to its corresponding address value.

Parameters

Attribute	Type	Content
[in]	BSTR bstrSymbolName	Symbol name
[out]	long *p_lSymbolAddr	Symbol address

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
BSTR bstrSymbolName;
long lSymbolAddr;

try
{
            hr = pHewServer1->SymbolToAddress2(bstrSymbolName, &lSymbolAddr);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim bstrSymbolName As String
Dim p_lSymbolAddr As Long   'In Visual Basic 2005, "As Long" will be replaced with "As Integer".

ret = hts.SymbolToAddress2(bstrSymbolName, p_lSymbolAddr)
```

AddressToSymbol2

Description

Converts label/symbol from an address value to its corresponding symbol name.

Parameters

Attribute	Type	Content
[in]	long lSymbolAddr	Address value
[out]	BSTR *p_bstrSymbolName	Symbol name

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
long lSymbolAddr;
BSTR        bstrSymbolName;

try
{
            hr = pHewServer1->AddressToSymbol2(lSymbolAddr, &bstrSymbolName);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim lSymbolAddr As Long
Dim p_bstrSymbolName As String

ret = hts.AddressToSymbol2(lSymbolAddr, p_bstrSymbolName)
```

GetLineFromAddr2

Description

Converts label/symbol from an address value to its corresponding file and line.

Parameters

Attribute	Type	Content
[in]	long lLineAddr	Line address
[out]	BSTR *p_bstrFileName	File name
[out]	long *p_lLineNo	Line number

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
long lLineAddr;
BSTR          bstrFileName;
long lLineNo;

try
{
              hr = pHewServer1->GetLineFromAddr2(lLineAddr, &bstrFileName, &lLineNo);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim lLineAddr As Long
Dim p_bstrFileName As String
Dim p_lLineNo As Long          'In Visual Basic 2005, "As Long" will be replaced with "As Integer".

ret = hts.GetLineFromAddr2(lLineAddr, p_bstrFileName, p_lLineNo)
```

GetAddrFromLine2

Description

Converts a label/symbol from file and line to its corresponding address value.

Parameters

Attribute	Type	Content
[in]	BSTR bstrFileName	File name
[in]	long lLineNo	Line number
[out]	long *p_lLineAddr	Line address

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
BSTR bstrFileName;
long lLineNo;
long lLineAddr;

try
{
                hr = pHewServer1->GetAddrFromLine2(
                       bstrFileName,
                       lLineNo,
                       &lLineAddr
                );
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim bstrFileName As String
Dim lLineNo As Long
Dim p_lLineAddr As Long        'In Visual Basic 2005, "As Long" will be replaced with "As Integer".

ret = hts.GetAddrFromLine2(bstrFileName, lLineNo, p_lLineAddr)
```

5.4.9 Downloads

Download2

Description

Downloads a load module.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrFileName	Load module (including path name)

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
BSTR      bstrDownloadFile;

try
{
          hr = pHewServer1->Download2(bstrDownloadFile);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim bstrDownloadFile As String

ret = hts.Download2(bstrDownloadFile)
```

Unload2

Description

Unloads a load module.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrFileName	Unload module (including path name)

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
BSTR bstrUnloadFile;

try
{
                hr = pHewServer1->Unload2(bstrUnloadFile);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim bstrUnloadFile As String

ret = hts.Unload2(bstrUnloadFile)
```

5.4.10 Start/Stop

InvokeHew2

Description

Starts a High-performance Embedded Workshop application. (Workspace is not opened.)

Parameters

There is no parameter.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;

try
{
            hr = pHewServer1->InvokeHew2();
}
```

Description example (Visual Basic)

```
Dim ret As Long

ret = hts.InvokeHew2
```

QuitHew2

Description

Terminates a High-performance Embedded Workshop application.

Parameters

There is no parameter.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;

try
{
          hr = pHewServer1->QuitHew2();
}
```

Description example (Visual Basic)

```
Dim ret As Long

ret = hts.QuitHew2
```

InvokeHewWithNoDialog

Description

Invokes the High-performance Embedded Workshop application without opening the [Welcome!] dialog box (no workspace is opened).

Parameters

There is no parameter.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;

try
{
          hr = pHewServer1->InvokeHewWithNoDialog();
}
```

Description example (Visual Basic)

```
Dim ret As Long

ret = hts.InvokeHewWithNoDialog
```

OpenWorkspace2

Description

Opens a workspace.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrFileName	File name (including path name)

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
BSTR bstrFileName;

try
{
            hr = pHewServer1->OpenWorkspace2(bstrFileName);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim bstrFileName As String

ret = hts.OpenWorkspace2(bstrFileName)
```

CloseWorkspace2

Description

Closes a workspace.

Parameters

Attribute	Type	Content
[in]	long _lIgnoreChanges	0x00000000:Workspace is not closed when changed 0x00000001:Workspace is closed without saving changes

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL
long _lIgnoreChanges = 1;

try
{
            hr = pHewServer1->CloseWorkspace2(_lIgnoreChanges);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim lIgnoreChanges As Long
lIgnoreChanges = 1

ret = hts.CloseWorkspace2(lIgnoreChanges)
```

SaveWorkspace2

Description

Saves a workspace.

Parameters

There is no parameter.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;

try
{
          hr = pHewServer1->SaveWorkspace2();
 }
```

Description example (Visual Basic)

```
Dim ret As Long

ret = hts.SaveWorkspace2
```

GetWorkSpaceDirectory

Description

Gets the absolute path of the current workspace.

Parameters

Attribute	Type	Content
[out]	BSTR *_pbstrCurrentWorkspaceDirectory	Absolute path of the current workspace

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULThr = E_FAIL;
CString strTmp = _T("");
BSTR bstrCurrentWorkspaceDirectory = strTmp.AllocSysString();
//Call HewTargetServer function
CString strCurrentWorkspaceDirectory;
try
{
        hr = pHewServer1->GetWorkSpaceDirectory(&bstrCurrentWorkspaceDirectory);
        strCurrentWorkspaceDirectory = bstrCurrentWorkspaceDirectory;
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim bstrCurrentWorkspaceDirectory As String

ret = hts.GetWorkSpaceDirectory(bstrCurrentWorkspaceDirectory)
```

SaveSession2

Description

Saves a session file.

Parameters

There is no parameter.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL

try
{
                hr = pHewServer1->SaveSession2();
 }
```

Description example (Visual Basic)

```
Dim ret As Long

ret = hts.SaveSession2
```

GetCurrentConfiguration2

Description

Gets the current build configuration.

Parameters

Attribute	Type	Content
[out]	BSTR *p_bstrCurrentConfigurationName	Name of the build configuration

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
CString      strCurrentConfigurationName = _T("");
BSTR bstrCurrentConfigurationName;

//calling HewTargetServer function
try
{
  hr = pHewServer1->GetCurrentConfiguration2(&bstrCurrentConfigurationName);
  strCurrentConfigurationName = bstrCurrentConfigurationName;
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim p_bstrCurrentConfigurationName As String

ret = hts.GetCurrentConfiguration2(p_bstrCurrentConfigurationName)
```

SetCurrentConfiguration2

Description

Sets a currently active build configuration.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrConfiguration	Build configuration name

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
BSTR bstrSetCurrentConfiguration;

try
{
                hr = pHewServer1->SetCurrentConfiguration2(bstrSetCurrentConfiguration);
 }
```

Description example (Visual Basic)

```
Dim ret As Long
Dim bstrSetCurrentConfiguration As String

ret = hts.SetCurrentConfiguration2(bstrSetCurrentConfiguration)
```

GetConfigurations2

Description

Gets all build configurations that have a project in each.

Parameters

Attribute	Type	Content
[out]	BSTR *p_bstrConfigurations	Build configuration name (multiple names, if any, are separated by a comma) (Example) "DefaultSession, SimSessionSH-4"

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT hr = E_FAIL;
Cstring    strTmp = _T("");
BSTR o1 = strTmp.AllocSysString();    //CString -> BSTR converted

//calling HewTargetServer function
CString so1;
try
{
            hr = pHewServer1->GetConfigurations2(&o1);
            so1 = o1;
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim p_bstrConfigurations As String

ret = hts.GetConfigurations2(p_bstrConfigurations)
```

GetCurrentSession2

Description

Gets the current debug session.

Parameters

Attribute	Type	Content
[out]	BSTR *p_bstrCurrentSessionName	Name of the debug session

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
CString strCurrentSessionName = _T("");
BSTR bstrCurrentSessionName;

//calling HewTargetServer function
try
{
  hr = pHewServer1->GetCurrentSession2(&bstrCurrentSessionName);
  strCurrentSessionName = bstrCurrentSessionName;
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim p_bstrCurrentSessionName As String

ret = hts.GetCurrentSession2(p_bstrCurrentSessionName)
```

SetCurrentSession2

Description

Sets a currently active debug session.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrSession	Debug session name

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
BSTR bstrSession;

//calling HewTargetServer function
try
{
            hr = pHewServer1->SetCurrentSession2(bstrSession);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim bstrSession As String

ret = hts.SetCurrentSession2(bstrSession)
```

GetSessions2

Description

Gets all debug sessions that are included in a project.

Parameters

Attribute	Type	Content
[out]	BSTR *p_bstrSessions	Debug session name (multiple names, if any, are separated by a comma) (Example) "DefaultSession, SimSessionSH-4"

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULThr = E_FAIL;

//calling HewTargetServer function
CString strTmp = _T("");
BSTR o2 = strTmp.AllocSysString();    //CString -> BSTR converted

CString so2;
try
{
        hr = pHewServer1->GetSessions2(&o2);
        so2 = o2;
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim p_bstrSessions As String

ret = hts.GetSessions2(p_bstrSessions)
```

GetCurrentProject2

Description

Gets the current project.

Parameters

Attribute	Type	Content
[out]	BSTR *p_bstrCurrentProjectName	Name of the project

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
  HRESULT   hr = E_FAIL;
  CString strCurrentProjectName = _T("");
  BSTR bstrCurrentProjectName;

  //calling HewTargetServer function
  try
  {
    hr = pHewServer1->GetCurrentProject2(&bstrCurrentProjectName);
    strCurrentProjectName = bstrCurrentProjectName;
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim p_bstrCurrentProjectName As String

ret = hts.GetCurrentProject2(p_bstrCurrentProjectName)
```

SetCurrentProject2

Description

Enables a specified project to make it active.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrProjectName	Project name

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT hr = E_FAIL;
BSTR bstrProjectName;

 //calling HewTargetServer function
try
{
            hr = pHewServer1->SetCurrentProject2(bstrProjectName);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim bstrProjectName As String

ret = hts.SetCurrentProject2(bstrProjectName)
```

GetProjects2

Description

Gets all project names.

Parameters

Attribute	Type	Content
[out]	BSTR *p_bstrProjectNames	Project name. If there are two or more project names, they should be delimited by a comma. Example: "Project1, Project2"

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
CString      strProjectNames = _T("");
BSTR bstrProjectNames;

//calling HewTargetServer function
try
{
  hr = pHewServer1->GetProjects2(&bstrProjectNames);
  strProjectNames = bstrProjectNames;
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim p_bstrProjectNames As String

ret = hts.GetProjects2(p_bstrProjectNames)
```

AddFile2

Description

Adds a file to the currently active project.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrFileName	File name (including path name)

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
BSTR bstrFileName;

try
{
              hr = pHewServer1->AddFile2(bstrFileName);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim bstrFileName As String

ret = hts.AddFile2(bstrFileName)
```

AddFiles2

Description

Adds multiple files to the currently active project.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrFileName	File name (multiple names, if any, are separated by a comma) (including path name)

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
BSTR bstrFileName;

try
{
            hr = pHewServer1->AddFiles2(bstrFileName);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim bstrFileName As String

ret = hts.AddFiles2(bstrFileName)
```

DeleteFile2

Description

Deletes a file from the currently active project.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrFileName	File name (including path name)

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT    hr = E_FAIL;
BSTR bstrFileName;

 //calling HewTargetServer function
try
{
            hr = pHewServer1->DeleteFile2(bstrFileName);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim bstrFileName As String

ret = hts.DeleteFile2(bstrFileName)
```

DeleteFiles2

Description

Deletes multiple files from the currently active project.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrFileName	File name (multiple names, if any, are separated by a comma) (including path name)

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT hr = E_FAIL;
BSTR bstrFileName;

//calling HewTargetServer function
try
{
            hr = pHewServer1->DeleteFiles2(bstrFileName);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim bstrFileName As String

ret = hts.DeleteFiles2(bstrFileName)
```

AddProjectFileFolder

Description

Adds a folder to the Projects tree in the current project.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrFolderName	Folder name (a folder and its subfolder should be separated by a backslash, e.g. Folder1\Subfolder)

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
BSTR bstrAddFolder = m_AddFolder.AllocSysString();
//Call HewTargetServer function
try
{
  hr = pHewServer1->AddProjectFileFolder(bstrAddFolder);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim bstrFolderName As String

ret = hts.AddProjectFileFolder(bstrFolderName)
```

RemoveProjectFileFolder

Description

Deletes a folder from the Projects tree in the current project.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrFolderName	Folder name (a folder and its subfolder should be separated by a backslash, e.g. Folder1\Subfolder)

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Precautions

You cannot delete any folders containing a file or subfolder.
When a folder and its subfolder are specified, only the subfolder is deleted.
If the specified folder name is Folder1\Subfolder, for example, Subfolder will be deleted.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
BSTR bstrRemoveFolder = m_RemoveFolder.AllocSysString();
//Call HewTargetServer function
try
{
  hr = pHewServer1->RemoveProjectFileFolder(bstrRemoveFolder);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim bstrFolderName As String

ret = hts.RemoveProjectFileFolder(bstrFolderName)
```

AddFileToFolder

Description

Adds a file to a specific folder under the Projects tree in the current project.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrFileName	File name (including path name)
[in]	BSTR _bstrFolderName	Folder name (a folder and its subfolder should be separated by a backslash, e.g. Folder1\Subfolder)

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT hr = E_FAIL;
BSTR bstrFileName = m_AddFile.AllocSysString();
BSTR bstrFolderName = m_AddFolder.AllocSysString();
//Call HewTargetServer function
try
{
        hr = pHewServer1->AddFileToFolder(bstrFileName, bstrFolderName);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim bstrFileName As String
Dim bstrFolderName As String

ret = hts.AddFileToFolder(bstrFileName, bstrFolderName)
```

BuildProject2

Description

Builds a project.

Parameters

There is no parameter.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;

//calling HewTargetServer function
try
{
            hr = pHewServer1->BuildProject2();
}
```

Description example (Visual Basic)

```
Dim ret As Long

ret = hts.BuildProject2
```

RebuildProject2

Description

Rebuilds a project.

Parameters

There is no parameter.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;

//calling HewTargetServer function
try
{
            hr = pHewServer1->RebuildProject2();
}
```

Description example (Visual Basic)

```
Dim ret As Long

ret = hts.RebuildProject2
```

UpDateAllDependency2

Description

Updates all dependency relations.

Parameters

There is no parameter.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT hr = E_FAIL;
//calling HewTargetServer function
try
{
            hr = pHewServer1->UpDateAllDependency2();
}
```

Description example (Visual Basic)

```
Dim ret As Long

ret = hts.UpDateAllDependency2
```

AddFileWithCompilerOption2

Description

Adds a file after setting compiler options for the project.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrFileName	File name (including path name)
[in]	BSTR _bstrIncludeDirectories	Include directory name. If there are two or more directories, they should be delimited by a comma. Example: "C:\tmp, D:\work"
[in]	BSTR _bstrDefines	Definition. If there are two or more definitions, they should be delimited by a comma. Example: "TMP1=C:\tmp, TMP2=D:\work"

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
BSTR _bstrAddFileName;
BSTR _bstrIncludeDirectories;
BSTR _bstrDefines;

//calling HewTargetServer function
try
{
  hr=pHewServer1->AddFileWithCompilerOption2(_bstrAddFileName,_bstrIncludeDirectories,_bstrDefines);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim bstrAddFileName As String
Dim bstrIncludeDirectories As String
Dim bstrDefines As String

ret = hts.AddFileWithCompilerOption2(bstrAddFileName, bstrIncludeDirectories, _
        bstrDefines)
```

GetLibraryOptions

Description

Acquires the library options for the linker in the current project.

Parameters

Attribute	Type	Content
[out]	BSTR *p_bstrLibraryOption	Library options for the linker in the current project Examples SHC: "LIB=c:\test\test1.lib, c:\temp\test2.lib" M16C: "-LD "D:\V540" -L "nc30lib""

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULThr = E_FAIL;
CString strTmp = _T("");
BSTR o1 = strTmp.AllocSysString();    //CSTring -> BSTR Conversion

//Call HewTargetServer function
CString so1;
try
{
            hr = pHewServer1->GetLibraryOptions(&o1);
            so1 = o1;
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim p_bstrLibraryOption As String

ret = hts.GetLibraryOptions(p_bstrLibraryOption)
```

SetLibraryOptions

Description

Sets library options for the linker in the current project. Existing library options that have been set will be deleted.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrLibraryOption	Library options for the linker set in the current project

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
BSTR bstrSetLibraryOption;
HRESULT hr = E_FAIL;

//Call HewTargetServer function
try
{
        hr = pHewServer1->SetLibraryOptions(bstrSetLibraryOption);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim bstrLibraryOption As String

ret = hts.SetLibraryOptions(bstrLibraryOption)
```

GetLibraryFilesForConfiguration

Description

Gets library options from a specific configuration in a specific project.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrProjectName	Project Name (the current project name when an empty string is entered)
[in]	BSTR _bstrConfiguration	Configuration Name (the current configuration name when an empty string is entered)
[out]	BSTR *_pbstrLibraryFiles	Library Option of Linker

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT hr = E_FAIL;
BSTR bstrProject = m_GetLibraryFilesForConfiguration_Project.AllocSysString();
BSTR bstrConfiguration = m_GetLibraryFilesForConfiguration_Configuration.AllocSysString();
CString strTmp = _T("");
BSTR bstrLibraryFiles = strTmp.AllocSysString();
CString strLibraryFiles;

//Call HewTargetServer function
try
{
   hr = pHewServer1->GetLibraryFilesForConfiguration(bstrProject, bstrConfiguration,
                                                     &bstrLibraryFiles);
   strLibraryFiles = bstrLibraryFiles;
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim bstrProject As String
Dim bstrConfiguration As String
Dim p_bstrLibraryFiles As String

ret = hts.GetLibraryFilesForConfiguration(bstrProject, bstrConfiguration, bstrLibraryFiles)
```

SetLibraryFilesForConfiguration

Description

Sets library options for a specific configuration in a specific project. If the selected library option has already been set, the older option is overwritten.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrProjectName	Project Name (the current project name when an empty string is entered)
[in]	BSTR _bstrConfiguration	Configuration Name (the current configuration name when an empty string is entered)
[in]	BSTR _bstrLibraryFiles	Library Option of Linker

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT hr = E_FAIL;
BSTR bstrProject = m_SetLibraryFilesForConfiguration_Project.AllocSysString();
BSTR bstrConfiguration = m_SetLibraryFilesForConfiguration_Configuration.AllocSysString();
BSTR bstrLibraryFiles = m_SetLibraryFilesForConfiguration_LibraryFiles.AllocSysString();

//Call HewTargetServer function
try
{
  hr = pHewServer1->SetLibraryFilesForConfiguration(bstrProject, bstrConfiguration,
                                                     bstrLibraryFiles);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim bstrProject As String
Dim bstrConfiguration As String
Dim bstrLibraryFiles As String

ret = hts. SetLibraryFilesForConfiguration(bstrProject, bstrConfiguration, bstrLibraryFiles)
```

GetIncludeFileDirectories

Description

Gets include file options from a file of a specific configuration in a specific project.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrProjectName	Project Name (the current project name when an empty string is entered)
[in]	BSTR _bstrConfiguration	Configuration Name (the current configuration name when an empty string is entered)
[in]	BSTR _bstrFileName	File Name (including path name)
[out]	VARIANT *_pvtIncludeDirectories	Include Option of Compiler which is VT_ARRAY|VT_VARIANT. Each element of the array is VARIANT of the VT_BSTR type.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT hr = E_FAIL;
BSTR bstrProject = m_GetIncludeFileDirectories_Project.AllocSysString();
BSTR bstrConfiguration = m_GetIncludeFileDirectories_Configuration.AllocSysString();
BSTR bstrFile = m_GetIncludeFileDirectories_File.AllocSysString();
CString strTmp = _T("");
VARIANT variantIncludeDirectories;

//Call HewTargetServer function
try
{
        hr =
        pHewServer1->GetIncludeFileDirectories(bstrProject,bstrConfiguration,bstrFile,
                                                            & variantIncludeDirectories);
}
```

Description example (Visual Basic 6.0)

```
Dim ret As Long
Dim bstrProject As String
Dim bstrConfiguration As String
Dim bstrFile As String
Dim vtIncludeDirectories As Variant
ret =
hts.GetIncludeFileDirectoriesGetIncludeFileDirectories(bstrProject,bstrConfiguration,bstrFile,
                                                                vtIncludeDirectories)
```

Description example (Visual Basic 2005)

```
Dim ret As Integer
Dim bstrProject As String
Dim bstrConfiguration As String
Dim bstrFile As String
Dim vtIncludeDirectories As Object
ret =
hts.GetIncludeFileDirectoriesGetIncludeFileDirectories(bstrProject,bstrConfiguration,bstrFile,
                                                                vtIncludeDirectories)
```

SetIncludeFileDirectories

Description

Sets include file options for a file of a specific configuration in a specific project.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrProjectName	Project Name (the current project name when an empty string is entered)
[in]	BSTR _bstrConfiguration	Configuration Name (the current configuration name when an empty string is entered)
[in]	BSTR _bstrFileName	File Name (including path name)
[in]	VARIANT _vtIncludeDirectories	Include Option of Compiler (VT_ARRAY|VT_BSTR)
[in]	long _lSettingMode	0: Append to the existing option(s) 1: Replace the existing option(s)

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT hr = E_FAIL;
long lMode;
BSTR bstrProject = m_SetIncludeFileDirectories_Project.AllocSysString();
BSTR bstrConfiguration = m_SetIncludeFileDirectories_Configuration.AllocSysString();
BSTR bstrFile = m_SetIncludeFileDirectories_File.AllocSysString();
VARIANT vtIncludeDirectories;

//Call HewTargetServer function
try
{
   hr = pHewServer1->SetIncludeFileDirectories(bstrProject, bstrConfiguration, bstrFile,
vtIncludeDirectories, lMode);
}
```

Description example (Visual Basic 6.0)

```
Dim ret As Long
Dim lMode As Long
Dim bstrProject As String
Dim bstrConfiguration As String
Dim bstrFile As String
Dim vtIncludeDirectories As Variant

ret = hts.SetIncludeFileDirectories(bstrProject, bstrConfiguration, bstrFile, vtIncludeDirectories,
                                    lMode)
```

Description example (Visual Basic 2005)

```
Dim ret As Integer
Dim iMode As Integer
Dim bstrProject As String
Dim bstrConfiguration As String
Dim bstrFile As String
Dim vtIncludeDirectories As Variant

ret = hts.SetIncludeFileDirectories(bstrProject, bstrConfiguration, bstrFile, vtIncludeDirectories,
                                    iMode)
```

GetCpuAndToolChainData

Description

Gets the family name, series name, and type name of the CPU, and the family name, name, and version number of the compiler in a specific project.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrProjectName	Project Name
[out]	BSTR *_pbstrCPUFamily	CPU Family Name
[out]	BSTR *_pbstrCPUSeries	CPU Series Name
[out]	BSTR *_pbstrCPUType	CPU Type Name
[out]	BSTR *_pbstrToolChainFamily	Compiler Family Name
[out]	BSTR *_pbstrToolChainName	Compiler Name
[out]	BSTR *_pbstrToolChainVersion	Compiler Version

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT hr = E_FAIL;
BSTR bstrProjectName = m_GetCpuAndToolChainData_Project.AllocSysString();
CString strTmp = _T("");
BSTR bstrCPUFamily = strTmp.AllocSysString();
BSTR bstrCPUSeries = strTmp.AllocSysString();
BSTR bstrCPUType = strTmp.AllocSysString();
BSTR bstrToolChainFamily = strTmp.AllocSysString();
BSTR bstrToolChainName = strTmp.AllocSysString();
BSTR bstrToolChainVersion = strTmp.AllocSysString();

//Call HewTargetServer function
CString strCPUFamily;
CString strCPUSeries;
CString strCPUType;
CString strToolChainFamily;
CString strToolChainName;
CString strToolChainVersion;

try
{
        hr = pHewServer1->GetCpuAndToolChainData(bstrProjectName, &bstrCPUFamily,
&bstrCPUSeries, &bstrCPUType, &bstrToolChainFamily, &bstrToolChainName, &bstrToolChainVersion);
        strCPUFamily = bstrCPUFamily;
        strCPUSeries        = bstrCPUSeries;
        strCPUType = bstrCPUType;
        strToolChainFamily = bstrToolChainFamily;
        strToolChainName = bstrToolChainName;
        strToolChainVersion = bstrToolChainVersion;
}
```

Description example (Visual Basic)

```
        Dim ret As Long
        Dim bstrProjectName As String
        Dim bstrCPUFamily As String
        Dim bstrCPUSeries As String
        Dim bstrCPUType As String
        Dim bstrToolChainFamily As String
        Dim bstrToolChainName As String
        Dim bstrToolChainVersion As String

        ret = hts.GetCpuAndToolChainData(bstrProjectName, bstrCPUFamily, bstrCPUSeries,
bstrCPUType, bstrToolChainFamily, bstrToolChainName, bstrToolChainVersion)
```

SetBuildExcludeFiles

Description

Excludes the specified file from building.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrFileNames	File name (multiple names, if any, are separated by a comma)

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT hr = E_FAIL;
 BSTR bstrBuildExcludeFiles;

//Call HewTargetServer function
try
{
            hr = pHewServer1->SetBuildExcludeFiles(bstrBuildExcludeFiles);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim bstrBuildExcludeFiles As String

ret = hts.SetBuildExcludeFiles(bstrBuildExcludeFiles)
```

SetBuildIncludeFiles

Description

Includes the specified file in building.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrFileNames	File name (multiple names, if any, are separated by a comma)

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT hr = E_FAIL;
 BSTR bstrBuildIncludeFiles;

//Call HewTargetServer function
try
{
		hr = pHewServer1->SetBuildIncludeFiles(bstrBuildIncludeFiles);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim bstrBuildIncludeFiles As String

ret = hts.SetBuildIncludeFiles(bstrBuildIncludeFiles)
```

5.4.15 Files

OpenFileAtLine2

Description

Opens a file by specifying the file name and line number.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrOpenFileName	File name (including path name)
[in]	long _lLine	Line number

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
BSTR bstrOpenFileName;
long _lLine = 1;

//calling HewTargetServer function
try
{
  hr = pHewServer1->OpenFileAtLine2(bstrOpenFileName, _lLine);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim bstrOpenFileName As String
Dim lLine As Long
lLine = 1

ret = hts.OpenFileAtLine2(bstrOpenFileName, lLine)
```

GetSourceFiles2

Description

Gets all source file names (such as *.cpp or *.src) in a project.
The file name is output as an absolute path.

Parameters

Attribute	Type	Content
[out]	BSTR *p_bstrSourceFiles	Source file names (if there are two or more file names, they should be delimited by a comma). Example: "c:\sample1.cpp, c:\sample2.cpp"

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
CString      strSourceFiles = _T("");
BSTR bstrSourceFiles = strSourceFiles.AllocSysString();

//calling HewTargetServer function
try
{
  hr = pHewServer1->GetSourceFiles2(&bstrSourceFiles);
  strSourceFiles = bstrSourceFiles;
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim p_bstrSourceFiles As String

ret = hts.GetSourceFiles2(p_bstrSourceFiles)
```

GetDownloadModules2

Description

Gets all module file names (such as *.abs) in a project.
The file name is output as an absolute path.

Parameters

Attribute	Type	Content
[out]	BSTR *p_bstrDownloadModules	Module file names (if there are two or more file names, they should be delimited by a comma). Example: "c:\sample1.abs, c:\sample2.abs"

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
CString      strDownloadModules = _T("");
BSTR bstrDownloadModules = strDownloadModules.AllocSysString();

//calling HewTargetServer function
try
{
  hr = pHewServer1->GetDownloadModules2(&bstrDownloadModules);
  strDownloadModules = bstrDownloadModules;
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim p_bstrDownloadModules As String

ret = hts.GetDownloadModules2(p_bstrDownloadModules)
```

GetDependentFiles2

Description

Gets all dependent file names (such as *.h or *.inc) in a project.
The file name is output as an absolute path.

Parameters

Attribute	Type	Content
[out]	BSTR *p_bstrDependentFiles	Dependent file names (if there are two or more file names, they should be delimited by a comma). Example: "c:\sample1.h, c:\sample2.h""

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
CString      strDependentFiles = _T("");
BSTR bstrDependentFiles = strDependentFiles.AllocSysString();

//calling HewTargetServer function
try
{
  hr = pHewServer1->GetDependentFiles2(&bstrDependentFiles);
  strDependentFiles = bstrDependentFiles;
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim p_bstrDependentFiles As String

ret = hts.GetDependentFiles2(p_bstrDependentFiles)
```

SetCoverageRange2

Description

Sets a coverage range.

Parameters

Attribute	Type	Content
[in]	long _lStartAddress	Start address
[in]	long _lEndAddress	End address

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
long _lStartAddress;
long _lEndAddress;

//calling HewTargetServer function
try
{
  hr = pHewServer1->SetCoverageRange2(_lStartAddress, _lEndAddress);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim lStartAddress As Long
Dim lEndAddress As Long

ret = hts.SetCoverageRange2(lStartAddress, lEndAddress)
```

Precautions

The coverage facility is enabled as soon as a coverage range is set.

GetCoverageRange2

Description

Gets data from a coverage range.

Parameters

Attribute	Type	Content
[out]	long *p_lStartAddress	Start address
[out]	long *p_lEndAddress	End address

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
long lStartAddress;
long lEndAddress;

//calling HewTargetServer function
try
{
  hr = pHewServer1->GetCoverageRange2(&lStartAddress, &lEndAddress);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim p_lStartAddress As Long  'In Visual Basic 2005, "As Long" will be replaced with "As Integer".
Dim p_lEndAddress As Long    'In Visual Basic 2005, "As Long" will be replaced with "As Integer".

ret = hts.GetCoverageRange2(p_lStartAddress, p_lEndAddress)
```

SetCoverageDisable2

Description

Disables the coverage function.

Parameters

There is no parameter.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT hr = E_FAIL;

//calling HewTargetServer function
try
{
        hr = pHewServer1->SetCoverageDisable2();
}
```

Description example (Visual Basic)

```
Dim ret As Long

ret = hts.SetCoverageDisable2
```

SetCoverageEnable2

Description

Enables the coverage function.

Parameters

There is no parameter.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT hr = E_FAIL;

//calling HewTargetServer function
try
{
        hr = pHewServer1->SetCoverageEnable2();
}
```

Description example (Visual Basic)

```
Dim ret As Long

ret = hts.SetCoverageEnable2
```

ClearCoverage2

Description

Clears the coverage information.

Parameters

There is no parameter.

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT hr = E_FAIL;

//calling HewTargetServer function
try
{
        hr = pHewServer1->ClearCoverage2();
}
```

Description example (Visual Basic)

```
Dim ret As Long

ret = hts.ClearCoverage2
```

GetCoverageStatus2

Description

Gets the coverage status information.

Parameters

Attribute	Type	Content
[out]	long *p_lStatus	Coverage status (1: Enabled or 0: Disabled)

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
long lStatus;

//calling HewTargetServer function
try
{
  hr = pHewServer1->GetCoverageStatus2(&lStatus);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim p_lStatus As Long           'In Visual Basic 2005, "As Long" will be replaced with "As Integer".

ret = hts.GetCoverageStatus2(p_lStatus)
```

LoadCoverage2

Description

Loads the coverage information.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrLoadFileName	File name

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
BSTR bstrLoadFileName;

//calling HewTargetServer function
try
{
  hr = pHewServer1->LoadCoverage2(bstrLoadFileName);
}
```

Description example (Visual Basic)

```
  Dim ret As Long
  Dim bstrLoadFileName As String

  ret = hts.LoadCoverage2(bstrLoadFileName)
```

SaveCoverage2

Description

Saves the coverage information.

Parameters

Attribute	Type	Content
[in]	BSTR _bstrSaveFileName	File name

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
BSTR bstrSaveFileName;

//calling HewTargetServer function
try
{
  hr = pHewServer1->SaveCoverage2(bstrSaveFileName);
}
```

Description example (Visual Basic)

```
  Dim ret As Long
  Dim bstrSaveFileName As String

  ret = hts.SaveCoverage2(bstrSaveFileName)
```

GetErrorString2

Description

Gets an error message corresponding to a specified error number.

Parameters

Attribute	Type	Content
[in]	long lError	Error number
[out]	BSTR *p_bstrError	Error message

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
long lError;
BSTR bstrErr;

try
{
                hr = pHewServer1->GetErrorString2(lError, &bstrErr);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim lError As Long
Dim p_bstrErr As String

ret = hts.GetErrorString2(lError, p_bstrErr)
```

GetHewStatus2

Description

Gets the current High-performance Embedded Workshop status.

Parameters

Attribute	Type	Content
[out]	long *p_lTargetReset	Returns 1 when the target is reset or 0 otherwise*
[out]	long *p_lTargetExecStatus	Returns 1 when the user program is under execution or 0 otherwise
[out]	long *p_lMemoryReset	Returns 1 when memory contents are updated or 0 otherwise*
[out]	long *p_lRegisterReset	Returns 1 when register values are updated or 0 otherwise*
[out]	long *p_lLInkStatus	Returns 1 when the target is connected or 0 otherwise
[out]	long *p_lPlatformInitialize	Returns 1 after the target is initialized or 0 otherwise*
[out]	long *p_lLoadingStatus	Returns 1 after a program is loaded or 0 otherwise

*: These flags are reset to 0 when this function is called.

Returned value

The returned value is 1 when the method was terminated successfully or 0 when there is error.

Description example (Visual C++)

```
HRESULT   hr;
long lTargetReset;
long lTargetExecStatus;
long lMemoryReset;
long lRegisterReset;
long lLinkStatus;
long lPlatformInitialize;
long lLoadingStatus;

//calling HewTargetServer function
try
{
            hr = pHewServer1->GetHewStatus2(&lTargetReset, & lTargetExecStatus,
                       &lMemoryReset,
                       &lRegisterReset,
                       &lLinkStatus,
                       &lPlatformInitialize,
                       &lLoadingStatus
            );
}
```

Description example (Visual Basic 6.0)

```
Dim ret As Long
Dim p_lTargetReset As Long
Dim p_lTargetExecStatus As Long
Dim p_lMemoryReset As Long
Dim p_lRegisterReset As Long
Dim p_lLinkStatus As Long
Dim p_lPlatformInitialize As Long
Dim p_lLoadingStatus As Long

ret = hts.GetHewStatus2(p_lTargetReset, p_lTargetExecStatus, p_lMemoryReset, _
        p_lRegisterReset, p_lLinkStatus, p_lPlatformInitialize, p_lLoadingStatus)
```

Description example (Visual Basic 2005)

```
Dim ret As Integer
Dim p_iTargetReset As Integer
Dim p_iTargetExecStatus As Integer
Dim p_iMemoryReset As Integer
Dim p_iRegisterReset As Integer
Dim p_iLinkStatus As Integer
Dim p_iPlatformInitialize As Integer
Dim p_iLoadingStatus As Integer

ret = hts.GetHewStatus2(p_iTargetReset, p_iTargetExecStatus, p_iMemoryReset, _
        p_iRegisterReset, p_iLinkStatus, p_iPlatformInitialize, p_iLoadingStatus)
```

GetHewStatusEx2

Description

Gets the High-performance Embedded Workshop status information (on initiation, opening a workspace, and build).

Parameters

Attribute	Type	Content
[out]	long *p_lInvokeHew	Initiation of the High-performance Embedded Workshop (0: Not initiated or 1: Initiated)
[out]	long *p_lOpenWorkspace	Opening of a workspace (0: Not open or 1: Open) Note: The acquired value can be 1 only when the HEW is connected to the target. To check whether a workspace is open, call GetWorkspaceDirectory instead.
[out]	long *p_lBuildProject	Build (0: Build stopped or 1: Build being performed)

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
long lInvokeHew;
long lOpenWorkspace;
long lBuildProject;

//calling HewTargetServer function
try
{
  hr = pHewServer1->GetHewStatusEx2(&lInvokeHew, &lOpenWorkspace, &lBuildProject);
}
```

Description example (Visual Basic 6.0)

```
Dim ret As Long
Dim p_lInvokeHew As Long
Dim p_lOpenWorkspace As Long
Dim p_lBuildProject As Long

ret = hts.GetHewStatusEx2(p_lInvokeHew, p_lOpenWorkspace, p_lBuildProject)
```

Description example (Visual Basic 2005)

```
Dim ret As Integer
Dim p_iInvokeHew As Integer
Dim p_iOpenWorkspace As Integer
Dim p_iBuildProject As Integer

ret = hts.GetHewStatusEx2(p_iInvokeHew, p_iOpenWorkspace, p_iBuildProject)
```

GetTargetName2

Description

Gets the target name that is currently connected.

Parameters

Attribute	Type	Content
[out]	BSTR* p_bstrName	Target name

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
BSTR      bstrName;

//calling HewTargetServer function
try
{
          //get target name
          hr = pHewServer1->GetTargetName2(&bstrName);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim p_bstrName As String

ret = hts.GetTargetName2(p_bstrName)
```

GetHewVersion

Description

Gets the version number of the High-performance Embedded Workshop.

Parameters

Attribute	Type	Content
[out]	BSTR*p_bstrHewVersion	Version

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT hr = E_FAIL;
CString strTmp = _T("");
BSTR bstrHewVersoin = strTmp.AllocSysString();    //CSTring -> BSTR Conversion
//Call HewTargetServer function
try
{
      hr = pHewServer1->GetHewVersion(&bstrHewVersoin);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim p_bstrHewVersion As String

ret = hts.GetHewVersion(p_bstrHewVersion)
```

Command

Description

Executes a High-performance Embedded Workshop command.

Parameters

Attribute	Type	Content
[out]	BSTR _bstrCommandLine	Command
[out]	BSTR *p_bstrCommandMessage	Message output by the command

Returned value

A value is returned in HRESULT type. The returned value is 0 when the function was successfully executed or other than 0 (e.g., E_FAIL (0x80004005L)) when an error occurred.

Description example (Visual C++)

```
HRESULT   hr = E_FAIL;
BSTR bstrCommandLine= m_Command.AllocSysString();
CString strTmp = _T("");
BSTR bstrCommandMessage = strTmp.AllocSysString();
//Call HewTargetServer function
try
{
hr = pHewServer1->Command(bstrCommandLine, &bstrCommandMessage);
}
```

Description example (Visual Basic)

```
Dim ret As Long
Dim bstrCommandLine As String
Dim p_bstrCommandMessage As String

ret = hts.Command(bstrCommandLine, p_bstrCommandMessage)
```

Precautions

(1) The Log command is not specifiable.

(2) In the edit mode of the Assemble command*, the High-performance Embedded Workshop does not automatically show the address of memory data to be assembled.
*Note: Support for this command depends on the debugger.

(3) The response in command execution using this feature will be slower than that in the Command Line window of the High-performance Embedded Workshop.

5.5 Events Acquirable in the High-performance Embedded Workshop

Type of event	Event issuance timing
Event1_ToClient_TargetReset	Issued when the target is reset
Event2_ToClient_Go	Issued when the target program is run
Event3_ToClient_Stop	Issued when the target program is halted
Event4_ToClient_MemoryReset	Issued when memory contents are updated through the HEW (e.g. by the SetMemory2 method, command line, or via the [Memory] window)
Event5_ToClient_RegisterReset	Issued when registers are updated through the HEW (e.g. by the SetPCAddress2 method, command line, or via the [Register] window)
Event6_ToClient_LinkUp	Issued when the target is up-linked
Event7_ToClient_LinkDown	Issued when the target is down-linked
Event8_ToClient_PlatformInitialize	Issued when the platform is initialized
Event9_ToClient_Download	Issued when a program is downloaded
Event10_ToClient_Unload	Issued when a program is unloaded
Event11_ToClient_HewInvoke	Issued at initiation of the High-performance Embedded Workshop
Event12_ToClient_WorkspaceOpen	Issued when a workspace is opened Note: This event is actually issued when the HEW is connected to the target. Make a workspace setting that allows the HEW to be connected to the target as soon as the workspace is opened.
Event13_ToClient_ProjectBuild	Issued at the build of a project

Note:
Events are not issued every time. For details, see "Note on acquisition of generated events" in secrion 4

Revision Record

Rev.	Date	Description	
		Page	**Summary**
1.00	Jun. 20, 2006	-	First Edition issued
2.00	Jul. 21, 2006	-	Added descriptions of the new methods
3.00	Oct. 20, 2006	20, 24	Modified the description of GetMemory() and GetDirectMemory()
4.00	Oct. 19, 2007	- 13	Added descriptions of the new methods and a note on a shift to Visual Basic .NET
5.00	Nov. 05, 2008	-	Added descriptions of Visual C++/Visual Basic 2005 and the new methods
6.00	Apr. 20, 2009	-	Updated the description example of SetMemory2() Updated the description of parameter FileName Removed the precautions on InvokeHew(WithNoDialog)
7.00	Jul. 01, 2010	-	Inserted a note on old company names and revised "Notice"
8.00	Nov. 01, 2010	-	Added notes on GetHewStatusEx2() and event WorkspaceOpen

HewTargetServer
User’s Manual

Publication Date: **Nov 01, 2010** **Rev.8.00**

Published by: **Renesas Electronics Corporation**

Edited by: **Microcomputer Tool Development Department 1**
Renesas Solutions Corp.

SALES OFFICES

Renesas Electronics Corporation

http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.

Renesas Electronics America Inc.
2880 Scott Boulevard Santa Clara, CA 95050-2554, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited
1101 Nicholson Road, Newmarket, Ontario L3Y 9C3, Canada
Tel: +1-905-898-5441, Fax: +1-905-898-3220

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-585-100, Fax: +44-1628-585-900

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-65030, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
7th Floor, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100083, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 204, 205, AZIA Center, No.1233 Lujiazui Ring Rd., Pudong District, Shanghai 200120, China
Tel: +86-21-5877-1818, Fax: +86-21-6887-7858 / -7898

Renesas Electronics Hong Kong Limited
Unit 1601-1613, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2886-9318, Fax: +852 2886-9022/9044

Renesas Electronics Taiwan Co., Ltd.
7F, No. 363 Fu Shing North Road Taipei, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
1 harbourFront Avenue, #06-10, keppel Bay Tower, Singapore 098632
Tel: +65-6213-0200, Fax: +65-6278-8001

Renesas Electronics Malaysia Sdn.Bhd.
Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics Korea Co., Ltd.
11F., Samik Lavied' or Bldg., 720-2 Yeoksam-Dong, Kangnam-Ku, Seoul 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 1.0

HewTargetServer
User’s Manual

R20UT0374EJ0800

Renesas Electronics Corporation