Stacker ActiveX

Version 18.0.0.947

User Guide

Original Instructions

Agilent Technologies

Notices

Guide Part Number

G5407-90002

Edition

Revision 00, September 2010

Contact Information

Agilent Technologies Inc.
Automation Solutions
5301 Stevens Creek Blvd.
Santa Clara, CA 95051
USA

Technical Support: 1.800.979.4811
or +1.408.345.8011
service.automation@agilent.com

Customer Service: 1.866.428.9811
or +1.408.345.8356
orders.automation@agilent.com

European Service: +44 (0)1763850230
euroservice.automation@agilent.com

Documentation feedback:
documentation.automation@agilent.com

Web:
www.agilent.com/lifesciences/automation

Acknowledgements

Warranty

The material contained in this document is provided "as is," and is subject to being changed, without notice, in future editions. Further, to the maximum extent permitted by applicable law, Agilent disclaims all warranties, either express or implied, with regard to this manual and any information contained herein, including but not limited to the implied warranties of merchantability and fitness for a particular purpose. Agilent shall not be liable for errors or for incidental or consequential damages in connection with the furnishing, use, or performance of this document or of any information contained herein. Should Agilent and the user have a separate written agreement with warranty terms covering the material in this document that conflict with these terms, the warranty terms in the separate agreement shall control.

Technology Licenses

The hardware and/or software described in this document are furnished under a license and may be used or copied only in accordance with the terms of such license.

Restricted Rights Legend

If software is for use in the performance of a U.S. Government prime contract or subcontract, Software is delivered and licensed as "Commercial computer software" as defined in DFAR 252.227-7014 (June 1995), or as a "commercial item" as defined in FAR 2.101(a) or as "Restricted computer software" as defined in FAR 52.227-19 (June 1987) or any equivalent agency regulation or contract clause. Use, duplication or disclosure of Software is subject to Agilent Technologies' standard commercial license terms, and non-DOD Departments and Agencies of the U.S. Government will receive no greater than Restricted Rights as defined in FAR 52.227-19(c)(1-2) (June 1987). U.S. Government users will receive no greater than Limited Rights as defined in FAR 52.227-14 (June1987) or DFAR 252.227-7015 (b)(2) (November 1995), as applicable in any technical data.

Safety Notices

A WARNING notice denotes a hazard. It calls attention to an operating procedure, practice, or the like that, if not correctly performed or adhered to, could result in personal injury or death. Do not proceed beyond a WARNING notice until the indicated conditions are fully understood and met.

A **CAUTION** notice denotes a hazard. It calls attention to an operating procedure, practice, or the like that, if not correctly performed or adhered to, could result in damage to the product or loss of important data. Do not proceed beyond a **CAUTION** notice until the indicated conditions are fully understood and met.

Contents

Preface

This preface contains the following topics:

About this guide

What this guide covers

This guide describes the Stacker ActiveX controls for the Labware Stacker.

This guide does not provide instructions for setting up and using the Labware Stacker. For these details, see the *Labware Stacker User Guide*.

Accessing Agilent Technologies Automation Solutions user guides

You can search the online knowledge base or download the latest version of any PDF file from the Agilent Technologies website at www.agilent.com/lifesciences/automation.

Safety information for the devices appears in the corresponding device user guide. You can also search the knowledge base or the PDF files for safety information.

Related topics

For information about...	See...
How to set up and use the Labware Stacker	*Labware Stacker User Guide*
Reporting problems	“Reporting problems” on page vii

Reporting problems

Contacting Automation Solutions Technical Support

If you find a problem with the Labware Stacker, contact Automation Solutions Technical Support at one of the following:

Europe

Phone: +44 (0)1763850230

email: euroservice.automation@agilent.com

US and rest of world

Phone: 1.800.979.4811 (US only) or +1.408.345.8011

email: service.automation@agilent.com

Reporting hardware problems

When contacting Agilent Technologies, make sure you have the serial number of the device ready.

Reporting software problems

When you contact Automation Solutions Technical Support, make sure you provide the following:

- Short description of the problem
- Software version number
- Error message text (or screen capture of the error message dialog box)
- Relevant software files

Reporting user guide problems

If you find a problem with this user guide or have suggestions for improvement, send your comments via an email to documentation.automation@agilent.com.

Related topics

For information about...	See...
How to set up and use the Labware Stacker	*Labware Stacker User Guide*
Accessing user information	"Accessing Agilent Technologies Automation Solutions user guides" on page vi

Stacker ActiveX control

This chapter gives integrators the ActiveX control information required to integrate the Labware Stacker in another company's lab automation system.

The Stacker ActiveX has been verified to work with both Visual Studio 6 and Visual Studio .NET.

This chapter contains the following topics:

About ActiveX controls

What is the Stacker ActiveX control

The Stacker ActiveX control is the software component that allows the Labware Stacker to interact with a third-party lab automation system.

How the Stacker ActiveX control is used

In an Agilent Technologies automation system, the VWorks software runs in standalone mode, and the Stacker ActiveX control is not used. The operator uses the VWorks software, which is already configured to control the devices in the system. However, some integrations, such as those with LIMS, require that a third-party application control the Labware Stacker. The Stacker ActiveX control enables third-party applications to interface with the Labware Stacker.

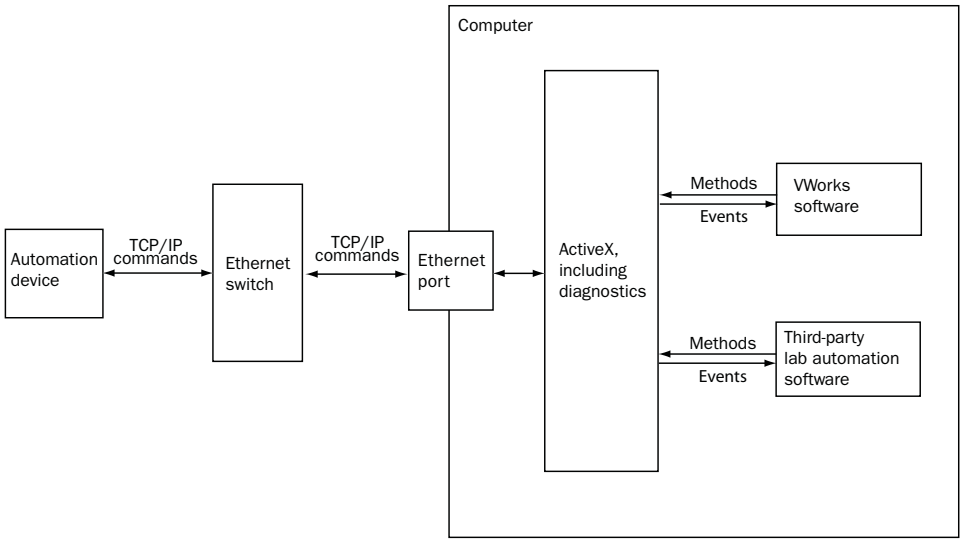
Each ActiveX control consists of a collection of the following:

- *Methods*. Functions that can be called to invoke individual operations
- *Properties*. Attributes or features of the ActiveX control
- *Events*. Notifications that methods have completed or resulted in errors

To ensure proper integration, you must know the available methods and properties for the ActiveX control.

The following diagram illustrates the use of the Stacker ActiveX control in a lab automation system environment. Actions you perform are conducted through ActiveX methods. System responses are relayed back through ActiveX events.

Note: Although the Stacker ActiveX control generates events, the third-party application must implement handlers for them.



Related topics

For information about...	See...
Stacker ActiveX properties	“Properties” on page 4
Stacker ActiveX methods	“Methods” on page 7
Stacker ActiveX events	“Events” on page 21
How to use the Labware Stacker	*Labware Stacker User Guide*
VWorks software	*VWorks Automation Control User Guide*
Reporting problems	“Reporting problems” on page vii

Properties

About this topic

This topic describes the following properties:

Blocking

```
VARIANT_BOOL Blocking
```

Description

Determines whether methods should block until completion or return immediately for asynchronous operation.

Acceptable values

VARIANT_TRUE or VARIANT_FALSE (C++) or True or False (Visual Basic .NET)

If set to VARIANT_TRUE or True, the ActiveX caller is forced to block or wait until a method completes before it returns control to the caller.

If set to VARIANT_FALSE or False, methods return immediately, and the caller should handle events accordingly.

Default value

VARIANT_FALSE or False

Blocking affects some methods differently. See each method's description for the effect. Unless otherwise noted:

- In non-blocking mode (Block = False), a method:
  - Starts another thread of execution to perform the given method, returning control to the application immediately.
  - Returns 0 on launching new thread successfully; otherwise returns nonzero, and an Error event is fired.
  - If the method is successful, an event indicating completion is fired; if unsuccessful, an Error event is fired.
- In blocking mode (Block = True), a method:
  - Is executed.
  - Returns 0 if it completes successfully; returns nonzero otherwise
- Error message can be reviewed by calling GetLastError().

Visual C++ example

```
// set the Stacker in blocking mode
sres = m_Stacker.SetBlocking(VARIANT_TRUE);
// set the Stacker in non-blocking mode
sres = m_Stacker.SetBlocking(VARIANT_FALSE);
// user should handle events if non-blocking!
```

Visual Basic .NET example

```
'set Stacker in blocking mode
Stacker1.Blocking = True
'set Stacker in non-blocking mode
Stacker1.Blocking = False
'user should handle events if non-blocking!
```

ControlPicture

```
IPictureDisp* ControlPicture
```

Description

A read-only picture of the Labware Stacker that can be used in the container's application.

Parameters

None

Visual C++ example

```
/* the CPicture class will be imported into your project
when the ActiveX is installed */
CButton button;
// create a button
CPicture Stacker;
// retrieve the picture
StackerPic = m_Stacker.GetControlPicture();
// paint the bitmap onto the button
button.SetBitmap( (HBITMAP) StackerPic.GetHandle());
```

Visual Basic .NET example

```
'assume that there is a button named Command1 on the current
form. You must set the Style property of 'Command1 to
"Graphical"
Dim iPicture As Image = Stacker1.ControlPicture()
```

DisplayPanelMessage

```
BSTR TopPanelMessage
```

Description

The message that is shown in the display panel at the top of the Labware Stacker.

Parameters

None

Visual C++ example

```
/* Sets the property value */
m_Stacker.TopPanelMessage = "message text";
/* Retrieves the property value */
sMsg = m_Stacker.TopPanelMessage;
```

Visual Basic .NET example

```
'Sets the property value'
Stacker1.TopPanelMessage = "message text"
'Retrieves the property value'
sMsg = Stacker1.TopPanelMessage
```

Related topics

For information about...	See...
Stacker ActiveX methods	"Methods" on page 7
Stacker ActiveX events	"Events" on page 21
Overview of the ActiveX controls	"About ActiveX controls" on page 2
Reporting problems	"Reporting problems" on page vii

Methods

About this topic

This topic describes the following methods:

Abort

```
LONG Abort()
```

Description

Aborts a current task that is in the error state and clears the error.

Parameters

None

Returns

0 if successful

Other value if there was an error

Visual C++ example

```
lres = m_Stacker.Abort();
```

Visual Basic .NET example

```
lres = Stacker1.Abort()
```

AboutBox

```
void AboutBox()
```

Description

Displays the About Box dialog, which includes the ActiveX version number and, if initialized, the firmware version number of the currently connected Labware Stacker.

The blocking mode does not affect the behavior of this method.

Parameters

None

Returns

None

Visual C++ example

```
m_Stacker.AboutBox();
```

Visual Basic .NET example

```
Stacker1.AboutBox()
```

Close

```
LONG Close()
```

Description

Closes the initialized Stacker profile and disconnects from the device.

Parameters

None

Returns

0 if successful

Other value if there was an error

Visual C++ example

```
lres = m_Stacker.Close();
```

Visual Basic .NET example

```
lres = Stacker1.Close()
```

Downstack

```
LONG Downstack()
```

Description

Performs a downstack operation.

Parameters

None

Returns

0 if successful

Other value if there was an error

Visual C++ example

```
lres = m_Stacker.Downstack();
```

Visual Basic .NET example

```
lres = Stacker1.Downstack()
```

EnumerateProfiles

```
VARIANT EnumerateProfiles()
```

Description

Retrieves a list of defined profiles. The strings in this array are the profile names that should be used for the Initialize method.

Parameters

None

Returns

An array of profile names

Visual C++ example

```
VARIANT vPRofiles = m_Stacker.EnumerateProfiles();
SAFEARRAY *psa = vProfiles.parray;
BSTR* bstrArray;
if
(FAILED(SafeArrayAccessData(psa,reinterpret_cast<void**>(
&bstrArray))))
{
VariantClear(&vProfiles);
return;
}
for (ULONG i = 0; i < psa->rgsabound[0].cElements; i++)
{
MessageBox(CString(bstrArray[i])));
}
SafeArrayUnaccessData(psa); VaraintClear(&vProfiles);
```

Visual Basic .NET example

```
profileNames = Stacker1.EnumerateProfiles()
For i = 0 To profileNames.GetLength(0) - 1
MsgBox profileNames(i)
Next
```

GetActiveXVersion

```
BSTR GetActiveXVersion()
```

Description

Retrieves the Labware Stacker ActiveX version number.

Parameters

None

Returns

A version number (string)

Visual C++ example

```
CString strActiveXVer = m_Stacker.GetActiveXVersion();
```

Visual Basic .NET example

```
Version = Stacker1.GetActiveXVersion()
```

GetFirmwareVersion

```
BSTR GetFirmwareVersion()
```

Description

Retrieves the firmware version of the device.

Parameters

None

Returns

A firmware version string

Visual C++ example

```
str = m_Stacker.GetFirmwareVersion();
```

Visual Basic .NET example

```
str = Stacker1.GetFirmwareVersion()
```

GetLabwareNames;

```
VARIANT GetLabwareNames()
```

Description

Retrieves a list of defined labware. The strings in this array are the options that should be used for the SetLabware method.

Parameters

None

Returns

An array of labware names

Visual C++ example

```
VARIANT vLabware = m_Stacker.GetLabwareNames();
SAFEARRAY *psa = vLabware.parray;
if
(FAILED(SafeArrayAccessData(psa,
reinterpret_cast<void**>( &bstrArray))))
{
VariantClear(&vLabware);
return;
}
for (ULONG i = 0; i < psa->rgsabound[0].cElements; i++)
{
MessageBox(CString(bstrArray[i]));
}
SafeArrayUnaccessData(psa); VariantCLear(&vLabware);
```

Visual Basic .NET example

```
labwareNames  = Stacker1.EnumerateProfiles()
For i = 0 To labwareNames.GetLength(0) - 1
MsgBox labwareNames (i)
Next
```

GetLastError

```
BSTR GetLastError()
```

Description

Returns the last known error condition.

Parameters

None

Returns

An error string

Visual C++ example

```
str = m_Stacker.GetLastError();
```

Visual Basic .NET example

```
str = Stacker1.GetLastError()
```

Home

```
LONG Home()
```

Description

Homes the Labware Stacker stage.

Parameters

None

Returns

0 if successful

Other value if there was an error

Visual C++ example

```
lres = m_Stacker.Home();
```

Visual Basic .NET example

```
lres = Stacker1.Home()
```

Ignore

```
LONG Ignore()
```

Description

Ignores the previously issued error and moves to the next step in the task. This is not a recommended course of action, as the errors are issued for a reason. However, ignoring some errors can be appropriate if the operator understands the implications.

Parameters

None

Returns

0 if successful

Other value if there was an error

Visual C++ example

```
lres = m_Stacker.Ignore();
```

Visual Basic .NET example

```
lres = Stacker1.Ignore()
```

Initialize

```
LONG Initialize(BSTR Profile)
```

Description

Initializes the profile and starts communication with the Labware Stacker using the parameters set in the profile. The profile specifies the Ethernet connection used to communicate with the Labware Stacker. The parameters for each profile can be adjusted in the Diagnostics dialog box (by a call to the ShowDiagsDialog method) on the Profiles page.

Parameters

BSTR Profile

The name of the profile to be used for initialization.

Returns

0 if successful

Other value if there was an error

Visual C++ example

```
// connect via Ethernet connection specified in the profile
lres = m_Stacker.Initialize("stacker profile");
```

Visual Basic .NET example

```
'connect via Ethernet connection specified in the profile
lres = Stacker1.Initialize ("stacker profile")
```

IsGripperOpen

```
VARIANT_BOOL IsGripperOpen()
```

Description

Checks whether the grippers are open.

Parameters

None

Returns

VARIANT_TRUE (C++) or TRUE (Visual Basic) if grippers are open

VARIANT_FALSE (C++) or FALSE (Visual Basic) if grippers are closed

Visual C++ example

```
lres = m_Stacker.IsGripperOpen();
```

Visual Basic .NET example

```
lres = Stacker1.IsGripperOpen()
```

IsPlatePresent

```
VARIANT_BOOL IsPlatePresent()
```

Description

Checks whether labware is in the grippers.

Parameters

None

Returns

VARIANT_TRUE (C++) or TRUE (Visual Basic) if labware is in the grippers

VARIANT_FALSE (C++) or FALSE (Visual Basic) if no labware is in the grippers

Visual C++ example

```
lres = m_Stacker.IsPlatePresent();
```

Visual Basic .NET example

```
lres = Stacker1.IsPlatePresent()
```

IsStackLoaded

```
VARIANT_BOOL IsStackLoaded()
```

Description

Checks whether the bottom-most labware in the stack is loaded (held by grippers) in preparation for a downstack or upstack operation. When loaded, the rack cannot be removed from the device.

Parameters

None

Returns

VARIANT_TRUE (C++) or TRUE (Visual Basic) if the stack is loaded

VARIANT_FALSE (C++) or FALSE (Visual Basic) if the stack is not loaded

Visual C++ example

```
lres = m_Stacker.IsStackLoaded();
```

Visual Basic .NET example

```
lres = Stacker1.IsStackLoaded()
```

IsStackPresent

```
VARIANT_BOOL IsStackPresent()
```

Description

Checks whether a rack is present on the Labware Stacker.

Parameters

None

Returns

VARIANT_TRUE (C++) or TRUE (Visual Basic) if the rack is present

VARIANT_FALSE (C++) or FALSE (Visual Basic) if the rack is not present

Visual C++ example

```
lres = m_Stacker.IsStackPresent();
```

Visual Basic .NET example

```
lres = Stacker1.IsStackPresent()
```

IsStageEmpty

```
VARIANT_BOOL IsStageEmpty()
```

Description

Checks whether the Labware Stacker stage is empty by moving the stage to the sensor position and reading the sensors.

Parameters

None

Returns

VARIANT_TRUE (C++) or TRUE (Visual Basic) if the stage is empty

VARIANT_FALSE (C++) or FALSE (Visual Basic) if labware is on the stage

Visual C++ example

```
lres = m_Stacker.IsStageEmpty();
```

Visual Basic .NET example

```
lres = Stacker1.IsStageEmpty()
```

Jog

```
LONG Jog(float increment)
```

Description

Moves the stage up or down by the specified increment, where positive increments move up, and negative increments move down. Increment units are millimeters.

Parameters

FLOAT increment

The jog increment, in millimeters.

Returns

0 if successful

Other value if there was an error

Visual C++ example

```
lres = m_Stacker.Jog(5.5);
```

Visual Basic .NET example

```
lres = Stacker1.Jog(5.5)
```

LoadStack

```
LONG LoadStack()
```

Description

Move the bottom-most labware in the stack to the grippers to prepare for a downstack or upstack action. When loaded, the grippers are holding the bottom-most labware in the stack, so the rack cannot be removed from the device.

Parameters

None

Returns

0 if successful

Other value if there was an error

Visual C++ example

```
lres = m_Stacker.LoadStack();
```

Visual Basic .NET example

```
lres = Stacker1.LoadStack()
```

OpenGripper

```
LONG OpenGripper(VARIANT_BOOL bOpen)
```

Description

Opens or closes the stacker grippers.

Parameters

VARIANT_BOOL bOpen

The open gripper option. Set bOpen to VARIANT_TRUE (C++) or TRUE (Visual Basic) to open grippers. Set bOpen to VARIANT_FALSE (C++) or FALSE (Visual Basic) to close grippers.

Returns

0 if successful

Other value if there was an error

Visual C++ example

```
lres = m_Stacker.OpenGripper(VARIANT_TRUE);
```

Visual Basic .NET example

```
lres = Stacker1.OpenGripper(TRUE)
```

ReleaseStack

```
LONG ReleaseStack()
```

Description

Unloads the labware from the grippers, thus allowing the rack to be removed. The Labware Stacker cannot downstack from or upstack to a released stack.

Parameters

None

Returns

0 if successful

Other value if there was an error

Visual C++ example

```
lres = m_Stacker.ReleaseStack();
```

Visual Basic .NET example

```
lres = Stacker1.ReleaseStack()
```

Retry

```
LONG Retry()
```

Description

Retries the last action after an error occurred.

Parameters

None

Returns

0 if successful

Other value if there was an error

Visual C++ example

```
lres =m_Stacker.Retry();
```

Visual Basic .NET example

```
lres = Stacker1.Retry()
```

SetButtonMode

```
LONG SetButtonMode(VARIANT_BOOL RunMode, BSTR Reply)
```

Description

Sets the action of the Load/Release button to permit loading or releasing of the stack. For example, use this method to prevent the operator from loading or releasing the stack during a running protocol.

Parameters

VARIANT_BOOL RunMode

The option used to set the button mode. Set RunMode to VARIANT_TRUE (C++) or TRUE (Visual Basic) to prevent the button from performing the load or release action.

BSTR Reply

The message string that is displayed if the operator presses the Load/Release button and the RunMode is set to VARIANT_TRUE (C++) or TRUE (Visual Basic).

Returns

0 if successful

Other value if there was an error

Visual C++ example

```
lres = m_Stacker.SetButtonMode(VARIANT_TRUE, "Cannot
release stack during protocol run");
```

Visual Basic .NET example

```
lres = Stacker1.SetButtonMode(TRUE, "Cannot release stack
during protocol run")
```

SetLabware

```
LONG SetLabware(BSTR bstrLabware, SHORT
plateDimensionSelect)
```

Description

Sets the labware for Labware Stacker operations. If the Diagnostics dialog box is open and the operator selects a different labware, the original labware will be restored when the Diagnostics dialog box is closed. This method should not be called when any movement is in progress.

Parameters

BSTR bstrLabware

The valid labware name.

SHORT plateDimensionSelect

A 16-bit value representing the set of labware dimensions. The possible values are:

0 = Normal dimensions

1 = Lidded dimensions

2 = Sealed dimensions

Returns

0 if successful

Other value if there was an error

Visual C++ example

```
lres = m_Stacker.SetLabware("384 Costar PS Sqr Well Flt
Btm",0);
```

Visual Basic .NET example

```
lres = Stacker1.SetLabware("384 Costar PS Sqr Well Flt
Btm",0)
```

ShowDiagsDialog

```
LONG ShowDiagsDialog (VARIANT_BOOL modal, SHORT
security_level)
```

Description

Displays the Diagnostics dialog box that allows the operator to troubleshoot and correct problems. This method can be called before the Initialize method to create a profile.

Parameters

VARIANT_BOOL modal

The mode of the dialog box. The dialog box displayed can be modal (does not permit operators to access the parent window) or modeless (permits operators to access the parent window). If the modal mode is desired, set modal to TRUE. If the modeless mode is desired, set modal to FALSE.

LONG securityLevel

The security level the user has in the Diagnostics dialog box:

0 = Administrator

1 = Technician

2 = Operator

3 = Guest

–1 - No access

Returns

0 if successful

Other value if there was an error

Visual C++ example

```
lres = m_Stacker.ShowDiagsDialog(VARIANT_TRUE,0);
```

Visual Basic .NET example

```
lres = m_Stacker1.ShowDiagsDialog (True,0)
```

Upstack

```
LONG Upstack()
```

Description

Performs an upstack operation.

Parameters

None

Returns

0 if successful

Other value if there was an error

Visual C++ example

```
lres = m_Stacker.Upstack();
```

Visual Basic .NET example

```
lres = Stacker1.Upstack()
```

Related topics

For information about...	See...
Stacker ActiveX properties	“Properties” on page 4
Stacker ActiveX events	“Events” on page 21
Overview of the ActiveX controls	“About ActiveX controls” on page 2
Reporting problems	“Reporting problems” on page vii

Events

About this topic

This topic describes the following events:

CloseComplete

```
void CloseComplete()
```

Description

This event occurs when the Close method is successful.

Parameters

None

Returns

None

DownstackComplete

```
void DownstackComplete()
```

Description

This event occurs when the Downstack method is successful.

Parameters

None

Returns

None

Error

```
void Error(short Number, BSTR* Description, long Scode, BSTR
Source, BSTR HelpFile, long HelpContext, boolean*
CancelDisplay)
```

Description

This event fires when an error occurs during any Labware Stacker operation or initialization.

Parameters

BSTR Description

The description of the error.

BOOLEAN* CancelDisplay

The option to hide the error message dialog box. Set to VARIANT_TRUE (C++) or TRUE (Visual Basic .NET).

Note: SHORT Number, LONG Scode, BSTR Source, BSTR HelpFile, and LONG HelpContext are not used.

Returns

None

HomeComplete

```
void HomeComplete()
```

Description

This event occurs when the Home method is successful.

Parameters

None

Returns

None

InitializeComplete

```
void InitializeComplete()
```

Description

This event occurs when the Initialize method is successful.

Parameters

None

Returns

None

JogComplete

```
void JogComplete()
```

Description

This event occurs when the Jog method is successful.

Parameters

None

Returns

None

LoadStackComplete

```
void LoadStackComplete()
```

Description

This event occurs when the LoadStack method is successful.

Parameters

None

Returns

None

OpenGripperComplete

```
void OpenGripperComplete()
```

Description

This event occurs when the OpenGripper method is successful.

Parameters

None

Returns

None

ReleaseStackComplete

```
void ReleaseStackComplete()
```

Description

This event occurs when the ReleaseStack method is successful.

Parameters

None

Returns

None

UpstackComplete

```
void UpstackComplete()
```

Description

This event occurs when the Upstack method is successful.

Parameters

None

Returns

None

Related topics

For information about...	See...
Stacker ActiveX properties	“Properties” on page 4
Stacker ActiveX methods	“Methods” on page 7
Overview of the ActiveX controls	“About ActiveX controls” on page 2
Reporting problems	“Reporting problems” on page vii

Agilent Technologies

User Guide

G5407-90002

Revision 00, September 2010