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.

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 ........................................................................................................... v
About this guide ................................................................................................. vi
Reporting problems ............................................................................................. vii

1. PlateLoc ActiveX control .............................................................................. 1
   About ActiveX controls ................................................................................... 2
   Properties ........................................................................................................ 4
   Methods ........................................................................................................... 6
   Events ............................................................................................................ 16
Preface

This preface contains the following topics:

- “About this guide” on page vi
- “Reporting problems” on page vii
About this guide

What this guide covers

This guide describes the ActiveX controls for the PlateLoc Thermal Microplate Sealer.

This guide does not provide instructions for setting up and using the PlateLoc Sealer. For these details, see the PlateLoc Thermal Microplate Sealer 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

<table>
<thead>
<tr>
<th>For information about...</th>
<th>See...</th>
</tr>
</thead>
<tbody>
<tr>
<td>How to set up and use the PlateLoc Sealer</td>
<td>PlateLoc Thermal Microplate Sealer User Guide</td>
</tr>
<tr>
<td>Reporting problems</td>
<td>“Reporting problems” on page vii</td>
</tr>
</tbody>
</table>
Reporting problems

Contacting Automation Solutions Technical Support

If you find a problem with the PlateLoc Sealer, 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

<table>
<thead>
<tr>
<th>For information about...</th>
<th>See...</th>
</tr>
</thead>
<tbody>
<tr>
<td>How to set up and use the PlateLoc Sealer</td>
<td>PlateLoc Thermal Microplate Sealer User Guide</td>
</tr>
<tr>
<td>Accessing user information</td>
<td>“Accessing Agilent Technologies Automation Solutions user guides” on page vi</td>
</tr>
</tbody>
</table>
PlateLoc ActiveX control

This chapter gives integrators the ActiveX control information required to integrate another company's lab automation device into the PlateLoc Sealer.

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

This chapter contains the following topics:

- “About ActiveX controls” on page 2
- “Properties” on page 4
- “Methods” on page 6
- “Events” on page 16
About ActiveX controls

What is the PlateLoc ActiveX control

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

How the PlateLoc ActiveX control is used

In an Agilent Technologies automation system, the VWorks software runs in standalone mode, and the PlateLoc 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 PlateLoc Sealer. The PlateLoc ActiveX control enables third-party applications to interface with the PlateLoc Sealer.

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 PlateLoc 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 PlateLoc ActiveX control generates events, the third-party application must implement handlers for them.
## Related topics

<table>
<thead>
<tr>
<th>For information about...</th>
<th>See...</th>
</tr>
</thead>
<tbody>
<tr>
<td>PlateLoc Sealer ActiveX properties</td>
<td>“Properties” on page 4</td>
</tr>
<tr>
<td>PlateLoc Sealer ActiveX methods</td>
<td>“Methods” on page 6</td>
</tr>
<tr>
<td>PlateLoc Sealer ActiveX events</td>
<td>“Events” on page 16</td>
</tr>
<tr>
<td>How to use the PlateLoc Sealer</td>
<td><em>PlateLoc Thermal Microplate Sealer</em></td>
</tr>
<tr>
<td></td>
<td><em>User Guide</em></td>
</tr>
<tr>
<td>VWorks software</td>
<td><em>VWorks Automation Control User Guide</em></td>
</tr>
<tr>
<td>Reporting problems</td>
<td>“Reporting problems&quot; on page vii</td>
</tr>
</tbody>
</table>
Properties

Blocking

VARIANT_BOOL Blocking;

Type
VARIANT_BOOL

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 6)

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 PlateLoc in blocking mode
sres = m_PlateLoc.SetBlocking(VARIANT_TRUE);

// set the PlateLoc in non-blocking mode
sres = m_PlateLoc.SetBlocking(VARIANT_FALSE);

// user should handle events if non-blocking!

Visual Basic 6 example

'set PlateLoc in blocking mode
PlateLoc1.Blocking = True
'set PlateLoc in non-blocking mode
PlateLoc1.Blocking = False
'user should handle events if non-blocking!
**ControlPicture**

**Type**
IPictureDisp

**Description**
A read-only picture of the PlateLoc 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 PlateLocPic;
// retrieve the picture
PlateLocPic = m_PlateLoc.GetControlPicture();
// paint the bitmap onto the button
button.SetBitmap((HBITMAP) PlateLocPic.GetHandle());

**Visual Basic 6 example**

' assume that there is a button named Command1 on the current form. You must set the Style property of 'Command1 to “Graphical”
Command1.Picture = PlateLoc1.ControlPicture

**Related topics**

<table>
<thead>
<tr>
<th>For information about...</th>
<th>See...</th>
</tr>
</thead>
<tbody>
<tr>
<td>PlateLoc Sealer ActiveX methods</td>
<td>“Methods” on page 6</td>
</tr>
<tr>
<td>PlateLoc Sealer ActiveX events</td>
<td>“Events” on page 16</td>
</tr>
<tr>
<td>Overview of the ActiveX controls</td>
<td>“About ActiveX controls” on page 2</td>
</tr>
<tr>
<td>Reporting problems</td>
<td>“Reporting problems” on page vii</td>
</tr>
</tbody>
</table>
Methods

Abort

LONG Abort(void)

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_PlateLoc.Abort();

Visual Basic 6 example
lres = PlateLoc.Abort

AboutBox

void AboutBox(void)

Description
Displays the About Box dialog, which includes the ActiveX version and, if initialized, the firmware version of the currently connected PlateLoc Sealer. The blocking mode does not affect the behavior of this method.

Parameters
None

Returns
None

Visual C++ example
m_PlateLoc.AboutBox();

Visual Basic 6 example
PlateLoc1.AboutBox

ApplySeal

LONG ApplySeal(void)

Description
Applies the seal to the microplate and keeps the door closed.

Parameters
None

Returns
0 if successful; other value if there was an error
**Visual C++ example**

```c++
lres = m_PlateLoc.ApplySeal();
```

**Visual Basic 6 example**

```vbnet
lres = PlateLoc1.ApplySeal
```

**Close**

LONG Close(void)

**Description**
Closes the initialized PlateLoc profile and disconnects from the device.

**Parameters**
None

**Returns**
0 if successful; other value if there was an error

**Visual C++ example**

```c++
lres = m_PlateLoc.Close();
```

**Visual Basic 6 example**

```vbnet
lres = PlateLoc1.Close
```

**EnumerateProfiles**

VARIANT EnumerateProfiles(void)

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

**Parameters**
None

**Returns**
An array of profile names

**Visual C++ example**

```c++
VARIANT vProfiles = m_PlateLoc.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]));
```
Visual Basic 6 example

```vbscript
profileNames = PlateLoc1.EnumerateProfiles
For i = LBound(profileNames) To UBound(profileNames)
    MsgBox profileNames(i)
Next
```

### GetActualTemperature

**LONG GetActualTemperature(SHORT* actual_temperature)**

**Description**
Retrieves the current temperature (°C) of the hot plate.

**Parameters**

<table>
<thead>
<tr>
<th>Argument Type</th>
<th>Argument Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SHORT*</td>
<td>actual_temperature</td>
<td>Contains the temperature value after returning from a call</td>
</tr>
</tbody>
</table>

**Returns**

0 if successful; other value if there was an error

**Visual C++ example**

```cpp
SHORT actual_temp;
lres = m_PlateLoc.GetActualTemperature(&actual_temp);
```

**Visual Basic 6 example**

```vbscript
DIM Actual_temp as INTEGER
lres = PlateLoc1.GetActualTemperature Actual_temp
```

### GetCycleCount

**LONG GetCycleCount(LONG* cycle_count)**

**Description**
Retrieves the number of seal cycles that have been performed.

**Parameters**

<table>
<thead>
<tr>
<th>Argument Type</th>
<th>Argument Name</th>
<th>Range</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LONG*</td>
<td>cycle_count</td>
<td>0</td>
<td>Stores the number of seal cycles after returning from the call</td>
</tr>
</tbody>
</table>
Returns
0 if successful; other value if there was an error

Visual C++ example
LONG cycle_count;
lres = m_PlateLoc.GetCycleCount(&cycle_count);

Visual Basic 6 example
DIM cycle_count as INTEGER
lres = PlateLoc1.GetCycleCount cycle_count

GetFirmwareVersion
BSTR GetFirmwareVersion(void)

Description
Retrieves the firmware version of the device.

Parameters
None

Returns
A firmware string

Visual C++ example
str = m_PlateLoc.GetFirmwareVersion();

Visual Basic 6 example
str = PlateLoc1.GetFirmwareVersion

GetLastError
BSTR GetLastError(void)

Description
Returns the last known error condition.

Parameters
None

Returns
An error string

Visual C++ example
str = m_PlateLoc.GetLastError();

Visual Basic 6 example
str = PlateLoc1.GetLastError
GetSealingTemperature

LONG GetActualTemperature(SHORT* sealing_temperature)

**Description**
Retrieves the current desired sealing temperature (°C) entered by the user.

**Parameters**

<table>
<thead>
<tr>
<th>Argument Type</th>
<th>Argument Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SHORT*</td>
<td>sealing_temperature</td>
<td>Gets the user-defined sealing temperature (°C)</td>
</tr>
</tbody>
</table>

**Returns**
0 if successful; other value if there was an error

**Visual C++ example**
SHORT sealing_temp;
lres = m_PlateLoc.GetSealingTemperature(&sealing_temp);

**Visual Basic 6 example**
DIM sealing_temp as INTEGER
lres = PlateLoc1.GetSealingTemperature sealing_temp

GetSealingTime

LONG GetSealingTime(DOUBLE* sealing_time)

**Description**
Retrieves the current desired seal-cycle duration entered by the user.

**Parameters**

<table>
<thead>
<tr>
<th>Argument Type</th>
<th>Argument Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DOUBLE*</td>
<td>sealing_time</td>
<td>Returns the user-defined seal-cycle duration</td>
</tr>
</tbody>
</table>

**Returns**
0 if successful; other value if there was an error
PlateLoc ActiveX control
Methods

GetVersion

BSTR GetVersion(void)

Description
Retrieves the PlateLoc ActiveX Control version number.

Parameters
None

Returns
A version string

Visual C++ example
str = m_PlateLoc.GetVersion();

Visual Basic 6 example
str = PlateLoc1.GetVersion

Ignore

LONG Ignore(void)

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_PlateLoc.Ignore();

Visual Basic 6 example
lres = PlateLoc1.Ignore
Initialize

LONG Initialize(BSTR Profile)

Description
Initializes the profile and starts communication with the PlateLoc Sealer using the parameters set in the profile. The profile specifies the serial port used to communicate with the PlateLoc Sealer. The parameters for each profile can be adjusted in the Diagnostics dialog box (by a call to the ShowDiagsDialog method) on the Profiles page. Initialize then sets the initial seal time and temperature for the PlateLoc Sealer and, if in the non-blocking mode, will signal its completion.

Parameters

<table>
<thead>
<tr>
<th>Argument Type</th>
<th>Argument Name</th>
<th>Range</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BSTR</td>
<td>profile name</td>
<td>Valid profile name</td>
<td>The name of the profile to be used for initialization</td>
</tr>
</tbody>
</table>

Returns
0 if successful; other value if there was an error

Visual C++ example

```cpp
// connect via serial, using the com port specified in the profile
lres = m_PlateLoc.Initialize( "plateloc profile");
```

Visual Basic 6 example

```vbnet
'connect via serial, using the com port specified in the profile
lres = PlateLoc1.Initialize "plateloc profile"
```

MoveStageIn

LONG MoveStageIn(void)

Description
Moves the plate stage into the sealing chamber.

Parameters
None

Returns
0 if successful; other value if there was an error

Visual C++ example

```cpp
lres = m_PlateLoc.MoveStageIn();
```

Visual Basic 6 example

```vbnet
lres = PlateLoc1.MoveStageIn
```
MoveStageOut

LONG MoveStageOut(void)

Description
Moves the plate stage out of the sealing chamber.

Parameters
None

Returns
0 if successful; other value if there was an error

Visual C++ example
lres = m_PlateLoc.MoveStageOut();

Visual Basic 6 example
lres = PlateLoc1.MoveStageOut

Retry

LONG Retry(void)

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_PlateLoc.Retry();

Visual Basic 6 example
lres = PlateLoc1.Retry

SetSealingTemperature

LONG SetSealingTemperature(SHORT* sealing_temperature)

Description
Sets the sealing temperature of the hot plate to the desired value entered by the user.

Parameters

<table>
<thead>
<tr>
<th>Argument Type</th>
<th>Argument Name</th>
<th>Range</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SHORT*</td>
<td>sealing_temperature</td>
<td>20–235</td>
<td>Returns the user-defined sealing temperature (°C)</td>
</tr>
</tbody>
</table>

Returns
0 if successful; other value if there was an error
Visual C++ example

    SHORT sealing_temp;
    lres = m_PlateLoc.SetSealingTemperature(&sealing_temp);

Visual Basic 6 example

    DIM sealing_temp as INTEGER
    lres = PlateLoc1.SetSealingTemperature sealing_temp

SetSealingTime

LONG SetSealingTime(DOUBLE* sealing_time)

Description

Sets the seal-cycle duration (seconds) to the desired value entered by the user.

Parameters

<table>
<thead>
<tr>
<th>Argument Type</th>
<th>Argument Name</th>
<th>Range</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DOUBLE*</td>
<td>sealing_time</td>
<td>0.5–12.0</td>
<td>Returns the user-defined seal-cycle duration (s)</td>
</tr>
</tbody>
</table>

Returns

0 if successful; other value if there was an error

Visual C++ example

    DOUBLE sealing_time;
    lres = m_PlateLoc.SetSealingTime(&sealing_time);

Visual Basic 6 example

    DIM sealing_time as DOUBLE
    lres = PlateLoc1.SetSealingTime sealing_time

ShowDiagsDialog

LONG ShowDiagsDialog (VARIANT_BOOL modal, SHORT security_level)

Description

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

Parameters

<table>
<thead>
<tr>
<th>Argument Type</th>
<th>Argument Name</th>
<th>Range</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VARIANT_BOOL</td>
<td>modal</td>
<td>1/0</td>
<td>Whether the Diagnostics dialog should be shown modally (1) or non-modally (0)</td>
</tr>
<tr>
<td>Argument Type</td>
<td>Argument Name</td>
<td>Range</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>---------------</td>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>SHORT</td>
<td>security_level</td>
<td>–1 – 3</td>
<td>The security level that the user has to operate the diagnostics: 0 = Administrator, 1 = Technician, 2 = Operator, 3 = Guest, –1 = No access</td>
</tr>
</tbody>
</table>

**Returns**

0 if successful; other value if there was an error

**Visual C++ example**

```c++
lres = m_PlateLoc.ShowDiagsDialog( VARIANT_TRUE, 0);
```

**Visual Basic 6 example**

```vbnet
lres = m_PlateLoc1.ShowDiagsDialog True, 0
```

**StartCycle**

**LONG StartCycle(void)**

**Description**

Starts a PlateLoc seal cycle.

**Parameters**

None

**Returns**

0 if successful; other value if there was an error

**Visual C++ example**

```c++
lres = m_PlateLoc.StartCycle();
```

**Visual Basic 6 example**

```vbnet
lres = PlateLoc1.StartCycle
```

**StopCycle**

**LONG StopCycle(void)**

**Description**

Stops the currently running PlateLoc seal cycle.

**Parameters**

None

**Returns**

0 if successful; other value if there was an error
Visual C++ example
lres = m_PlateLoc.StopCycle();

Visual Basic 6 example
lres = PlateLoc1.StopCycle

Related topics

<table>
<thead>
<tr>
<th>For information about...</th>
<th>See...</th>
</tr>
</thead>
<tbody>
<tr>
<td>PlateLoc Sealer ActiveX properties</td>
<td>“Properties” on page 4</td>
</tr>
<tr>
<td>PlateLoc Sealer ActiveX events</td>
<td>“Events” on page 16</td>
</tr>
<tr>
<td>Overview of the ActiveX controls</td>
<td>“About ActiveX controls” on page 2</td>
</tr>
<tr>
<td>Reporting problems</td>
<td>“Reporting problems” on page vii</td>
</tr>
</tbody>
</table>

Events

About events
The events listed in this topic occur only if Blocking is set to false or 0 (non-blocking mode).

ApplySealComplete

void ApplySealComplete(void)

Description
Occurs when the PlateLoc Sealer completes the sealing process.

Parameters
None

Returns
None

CycleComplete

void CycleComplete(void)

Description
Occurs when the PlateLoc Sealer successfully completes a non-blocking seal cycle.
Error

Description
Starts when an error occurs during any PlateLoc Sealer operation or initialization.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>SHORT</td>
<td>Not used.</td>
</tr>
<tr>
<td>Description</td>
<td>BSTR*</td>
<td>Description of the error.</td>
</tr>
<tr>
<td>Scode</td>
<td>LONG</td>
<td>Not used.</td>
</tr>
<tr>
<td>Source</td>
<td>BSTR</td>
<td>Not used.</td>
</tr>
<tr>
<td>HelpFile</td>
<td>BSTR</td>
<td>Not used.</td>
</tr>
<tr>
<td>HelpContext</td>
<td>LONG</td>
<td>Not used.</td>
</tr>
<tr>
<td>CancelDisplay</td>
<td>VARIANT_BOOL*</td>
<td>Set to VARIANT_TRUE (C++) or True (Visual Basic 6) to disable the stock event handler behavior, which is to display a dialog box with the description in it.</td>
</tr>
</tbody>
</table>

Returns
None

GetActualTemperatureComplete

```c
void GetActualTemperatureComplete(SHORT actual_temperature)
```

Description
Occurs when the PlateLoc Sealer successfully retrieves the current actual temperature of the hot plate. Current actual temperature is returned in the actual_temperature parameter in the non-blocking mode.

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>actual_temperature</td>
<td>SHORT</td>
<td>Actual temperature (°C) of the hot plate</td>
</tr>
</tbody>
</table>
Returns
None

**GetCycleCountComplete**

```c
void GetCycleCountComplete(LONG cycle_count)
```

**Description**
Occurs when the PlateLoc Sealer successfully retrieves the odometer value. The odometer value is returned in the cycle-count parameter in the non-blocking mode.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cycle_count</td>
<td>LONG</td>
<td>Odometer value</td>
</tr>
</tbody>
</table>

Returns
None

**GetSealingTemperatureComplete**

```c
void GetSealingTemperatureComplete(SHORT sealing_temperature)
```

**Description**
Occurs when the PlateLoc Sealer successfully retrieves the sealing temperature of the hot plate. The sealing temperature is returned in the sealing_temperature parameter in the non-blocking mode.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sealing_temperature</td>
<td>SHORT</td>
<td>Hot plate temperature (°C) entered by the user</td>
</tr>
</tbody>
</table>

Returns
None

**GetSealingTimeComplete**

```c
void GetSealingTimeComplete(DOUBLE sealing_time)
```

**Description**
Occurs when the PlateLoc Sealer successfully retrieves the sealing time. The sealing time is returned in the sealing_time parameter in the non-blocking mode.
## Events

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sealing_time</td>
<td>DOUBLE</td>
<td>Time duration (s) entered by the user</td>
</tr>
</tbody>
</table>

### Returns

None

#### InitializeComplete

```c
void InitializeComplete(void)
```

**Description**

Occurs when the PlateLoc Sealer successfully completes non-blocking initialization. This is necessary because devices without a barcode reader take longer to initialize.

**Parameters**

None

**Returns**

None

#### MoveStageInComplete

```c
void MoveStageInComplete(void)
```

**Description**

Occurs when the PlateLoc Sealer successfully moves the plate stage into the sealing chamber.

**Parameters**

None

**Returns**

None

#### MoveStageOutComplete

```c
void MoveStageOutComplete(void)
```

**Description**

Occurs when the PlateLoc Sealer successfully moves the plate stage out of the sealing chamber.

**Parameters**

None

**Returns**

None
**SetSealingTemperatureComplete**

void SetSealingTemperatureComplete(SHORT sealing_temperature)

**Description**
This event occurs when the PlateLoc Sealer successfully modifies the desired sealing temperature of the hot plate in the non-blocking mode. The new desired sealing temperature is returned in the sealing_temperature parameter.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sealing_temperature</td>
<td>SHORT</td>
<td>Hot plate temperature (°C) entered by the user</td>
</tr>
</tbody>
</table>

**Returns**
None

**SetSealingTimeComplete**

void SetSealingTimeComplete(DOUBLE sealing_time)

**Description**
This event occurs when the PlateLoc successfully modifies the desired sealing time duration in the non-blocking mode. The new sealing time duration is returned in the sealing_time parameter.

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sealing_time</td>
<td>DOUBLE</td>
<td>Time duration (s) entered by the user</td>
</tr>
</tbody>
</table>

**Returns**
None

**Related topics**

<table>
<thead>
<tr>
<th>For information about...</th>
<th>See...</th>
</tr>
</thead>
<tbody>
<tr>
<td>PlateLoc Sealer ActiveX properties</td>
<td>“Properties” on page 4</td>
</tr>
<tr>
<td>PlateLoc Sealer ActiveX methods</td>
<td>“Methods” on page 6</td>
</tr>
<tr>
<td>Overview of the ActiveX controls</td>
<td>“About ActiveX controls” on page 2</td>
</tr>
<tr>
<td>Reporting problems</td>
<td>“Reporting problems” on page vii</td>
</tr>
</tbody>
</table>