

Advanced Custom Study Interaction With Menus, Control Bar Buttons, Pointer Events

- [Advanced Custom Study Menu Interaction](#)
 - [sc.AddACSChartShortcutMenuItem\(\)](#)
 - [sc.RemoveACSChartShortcutMenuItem\(\)](#)
 - [sc.SetACSChartShortcutMenuItemDisplayed\(\)](#)
 - [sc.SetACSChartShortcutMenuItemEnabled\(\)](#)
 - [sc.SetACSChartShortcutMenuItemChecked\(\)](#)
 - [sc.PlaceACSChartShortcutMenuItemsAtTopOfMenu](#)
 - [sc.AddACSChartShortcutMenuSeparator\(\)](#)
 - [sc.ChangeACSChartShortcutMenuItemText\(\)](#)
- [Advanced Custom Study Control Bar Buttons and Pointer Events](#)
 - [Introduction](#)
 - [Advanced Custom Study Control Bar Buttons](#)
 - [Receiving Pointer Events](#)
 - [sc.SetCustomStudyControlBarButtonText\(\)](#)
 - [sc.SetCustomStudyControlBarButtonHoverText\(\)](#)
 - [sc.SetCustomStudyControlBarButtonEnable\(\)](#)
 - [sc.SetCustomStudyControlBarButtonColor\(\)](#)
 - [sc.GetCustomStudyControlBarButtonEnableState\(\)](#)
 - [Advanced Custom Study Control Bar Button Keyboard Shortcuts](#)
 - [sc.BlockChartDrawingSelection](#)
 - [Maintaining Multiple On States For Advanced Custom Study Control Bar Buttons](#)
 - [Disabling Previously Selected Advanced Custom Study Control Bar Button](#)

Advanced Custom Study Menu Interaction

sc.AddACSChartShortcutMenuItem()

ACSIL Function

```
int AddACSChartShortcutMenuItem(long ChartNumber, const char * MenuText);
```

The **sc.AddACSChartShortcutMenuItem()** function is for adding a custom menu command to the Chart Shortcut menu that displays when you right-click on a chart. The name of the command will be the text specified by the **MenuText** parameter. The command is added to the chart specified by **ChartNumber**. The function returns a unique menu integer command ID when

successful.

You should only add a particular menu command item only once. Although when you add a menu item with the same **MenuText**, it will not be added again and you will be returned the very same menu item ID previously returned for that text.

Once a menu item has been added, the study function instance is called when the user selects the menu item from the Chart Shortcut Menu which is displayed when right clicking on a chart. The selected menu item identifier is obtained through the **sc.MenuEventID** ACSIL member given to the custom study function instance.

When a study is removed from a chart, the study should always remove any custom Chart Shortcut Menu commands by calling [sc.RemoveACSChartShortcutMenuItem\(\)](#). This can be done when **sc.LastCallToFunction** is nonzero.

Return Value

The function returns a unique menu integer command ID when successful.

Returns -1 on an error. An error can mean the **ChartNumber** parameter is not valid or the maximum limit of Chart Shortcut Menu custom commands has been reached. A value of 0 is never returned.

Example

The following code provides an example. For a complete working example, refer to the **scsf_ButtonAndMenuAndPointerExample()** function in the **/ACS_Source/studies.cpp** file in the folder where Sierra Chart is installed to.

```
int& MenuID = sc.PersistVars->i1;

if (sc.Index == 0) // Only do this when calculating the first bar.
{
    // Add chart short cut menu item
    if (MenuID <= 0)
        MenuID = sc.AddACSChartShortcutMenuItem(sc.ChartNumber, "Menu Example Menu Item 1");

    if (MenuID < 0)
        sc.AddMessageToLog("Add ACS Chart Shortcut Menu Item failed", 1);
}

if (sc.MenuEventID != 0 && sc.MenuEventID == MenuID)
    sc.AddMessageToLog("User selected ACS Chart Shortcut Menu Item", 1);
```

sc.RemoveACSChartShortcutMenuItem()

ACSL Function

```
bool RemoveACSChartShortcutMenuItem(long ChartNumber, int MenuID);
```

sc.RemoveACSChartShortcutMenuItem() is used to remove a custom chart shortcut menu

command, associated with **MenuItemID** previously added with **sc.AddACSChartShortcutMenuItem()**, from the specified chart (**ChartNumber**). The function returns 1 when the menu item command was successfully removed. Or returns 0 on an error removing the menu item command.

Chart shortcut items should always be removed when the study is removed. This can be done by calling this function when **sc.LastCallToFunction** is nonzero.

Example

The following code provides an example. For a working example, refer to the **scsf_MenuAndPointerExample()** function in the **/ACS_Source/studies.cpp** file in the folder where Sierra Chart is installed to.

```
if (sc.LastCallToFunction)
{
    // Be sure to remove the menu command when study is removed
    sc.RemoveACSChartShortcutMenuItem(sc.ChartNumber, MenuItemID);
}
```

sc.SetACSChartShortcutMenuItemDisplayed()

ACSIL Function

```
int SetACSChartShortcutMenuItemDisplayed(long ChartNumber, int MenuItemID, bool DisplayItem);
```

sc.SetACSChartShortcutMenuItemEnabled()

ACSIL Function

```
int SetACSChartShortcutMenuItemEnabled(long ChartNumber, int MenuItemID, bool Enabled);
```

sc.SetACSChartShortcutMenuItemChecked()

ACSIL Function

```
int SetACSChartShortcutMenuItemChecked(long ChartNumber, int MenuItemID, bool Checked);
```

The **sc.SetACSChartShortcutMenuItemChecked** function adds or removes a checkmark beside the ACSIL menu command specified with **MenuItemID** for the chart specified with **ChartNumber**. Set **Checked** to FALSE to remove the check mark . Set **Checked** to TRUE to add a check mark.

sc.PlaceACSChartShortcutMenuItemsAtTopOfMenu

ACSIL boolean variable

The **sc.PlaceACSChartShortcutMenuItemsAtTopOfMenu** variable can be set anywhere within the custom study function.

When it is set to TRUE, then all menu items added with **sc.AddACSChartShortcutMenuItem()**, will be listed at the top of the Chart Shortcut menu.

sc.AddACSChartShortcutMenuSeparator()

ACSIL Function

```
int AddACSChartShortcutMenuSeparator(long ChartNumber);
```

The **sc.AddACSChartShortcutMenuSeparator** function will add a separator line to the Chart Shortcut menu below the last added ACSIL menu command.

sc.ChangeACSChartShortcutMenuItemText()

ACSIL Function

```
int ChangeACSChartShortcutMenuItemText(int ChartNumber, int MenuIdentifier, const char * NewMenuText);
```

The **sc.ChangeACSChartShortcutMenuItemText** function modifies the text of the menu item for the Chart Shortcut menu identified by the **MenuIdentifier** parameter.

The new text is specified by the **NewMenuText** parameter.

The **ChartNumber** parameter specifies particular chart the menu is associated with.

The function returns TRUE if the menu item text was successfully changed.

Advanced Custom Study Buttons and Pointer Events

Introduction

There are 150 Advanced Custom Study buttons available for custom studies to interact with on the [Control Bar](#).

These buttons can be added to any of the Control Bars.

These buttons serve 2 purposes.

They can be used to signal to an Advanced Custom Study, that is monitoring for them, whether a particular button is selected or not. In other words whether the button is in a pushed in state or not (on/off).

So this allows a study to monitor for an On/Off state using a particular Control Bar button. This provides a lot of useful functionality for a custom study including even one-time button events where the button does not remain in an On state, which is explained further in this section.

The second purpose of these Advanced Custom Study Control Bar buttons is that when one of

them is selected (On state), the study function instances on the chart that have requested to receive Pointer events by setting **sc.ReceivePointerEvents = ACS_RECEIVE_POINTER_EVENTS_WHEN_ACS_BUTTON_ENABLED** in the study function, will receive those events as the Pointer is moved around or used on the chart that contains the study function instance. These events include Pointer Button Down, Pointer Button Up, Pointer Move.

For a working example for all of this, refer to the **scsf_MenuAndPointerExample()** function in the **/ACS_Source/studies.cpp** file in the folder where Sierra Chart is installed to.

It is also supported to assign keyboard shortcuts to these Advanced Custom Study Control Bar buttons. To do this, select **Global Settings >> Customize Keyboard Shortcuts**. In the **Commands** list, expand the **Custom Study Control Bar Button Keyboard Shortcuts** list. For complete documentation, refer to [Customizing Keyboard Shortcuts](#).

The On/Off state of Advanced Custom Study Control Bar buttons is remembered per individual chart. So depending upon what chart is active, effects the on/off state of these buttons, depending upon their usage with that chart or how an Advanced Custom Study study has set them.

Advanced Custom Study Control Bar Buttons

To use the Advanced Custom Study Control Bar buttons, they need to be added to the Control Bar. To do this, select **Global Settings >> Customize Control Bars >> Control Bar #** to add them to the Control Bar.

Look in the **Advanced Custom Study Buttons** list. You will find listed **Custom Study Button 1-150**. Add any of these to the Control Bar that you require.

Once added to the Control Bar, these Advanced Custom Study Control Bar buttons toggle between On and Off, and visually show the state by being pushed in or out on the Control Bar.

This has changed effective with version 2268: Only one Advanced Custom Study Control Bar button can be active at any time for a particular chart. Therefore, selecting an Advanced Custom Study Control Bar button also effectively disables the currently active Advanced Custom Study Control Bar button.

Effective with version 2268 and higher, any number of Advanced Custom Study Control Bar buttons can be in an On state at the same time.

The Advanced Custom Study Control Bar button state changes caused by pressing one of these buttons are sent to all of the studies on the active chart. This occurs by the study functions being called and the setting of the ACSIL variable **sc.PointerEventType**.

The possible values for **sc.PointerEventType** are: **SC_ACS_BUTTON_ON**, **SC_ACS_BUTTON_OFF**. The specific ACS Control Bar button that is toggled on or off is determined through ACSIL variable **sc.MenuEventID**. The possible values for **sc.MenuEventID** are **ACS_BUTTON_1** through **ACS_BUTTON_150**.

Receiving Pointer Events

An Advanced Custom Study on a chart can receive Pointer events (**SC_POINTER_BUTTON_DOWN**, **SC_POINTER_MOVE**, **SC_POINTER_BUTTON_UP**) when the Pointer is moved around or clicked on the chart the study function instance is applied to.

To have the study function instance receive these events, set **sc.ReceivePointerEvents** to one of the following constants: **ACS_RECEIVE_NO_POINTER_EVENTS**, **ACS_RECEIVE_POINTER_EVENTS_WHEN_ACS_BUTTON_ENABLED**, **ACS_RECEIVE_POINTER_EVENTS_ALWAYS**, **ACS_RECEIVE_POINTER_EVENTS_ALWAYS_FOR_ALL_TOOLS** anywhere within the study function.

If **sc.ReceivePointerEvents** is set to **ACS_RECEIVE_NO_POINTER_EVENTS**, the Pointer events will not be received by the study function instance.

If **sc.ReceivePointerEvents** is set to **ACS_RECEIVE_POINTER_EVENTS_WHEN_ACS_BUTTON_ENABLED** and one of the Advanced Custom Study Control Bar Buttons 1-150 is selected (On state) for the chart, then the Pointer events will be received by the study function .

If **sc.ReceivePointerEvents** is set to **ACS_RECEIVE_POINTER_EVENTS_ALWAYS**, then the Pointer events will be received by the study function whether or not an Advanced Custom Study Control Bar Buttons 1-150 is selected (On state) for the chart or not.

These Pointer events are only passed to custom studies on the chart when the active Tool is **Tools >> Pointer**, **Tools >> Chart Values**, or **Tools >> Hand**. Otherwise, they are not unless **sc.ReceivePointerEvents** is set to **ACS_RECEIVE_POINTER_EVENTS_ALWAYS_FOR_ALL_TOOLS**.

Additionally, Pointer events into a custom study will not be received when adjusting the size of a Chart Region.

Once a particular **Advanced Custom Study Control Bar Button 1-150** is enabled (On state) and **sc.ReceivePointerEvents** is set to **ACS_RECEIVE_POINTER_EVENTS_WHEN_ACS_BUTTON_ENABLED** or **ACS_RECEIVE_POINTER_EVENTS_ALWAYS** / **ACS_RECEIVE_POINTER_EVENTS_ALWAYS_FOR_ALL_TOOLS**, Pointer events will begin to be received into the custom study function instance as the events occur.

The custom study function instance will be called on an event and the **sc.PointerEventType** will be set with the particular event constant (**SC_POINTER_BUTTON_DOWN**, **SC_POINTER_MOVE**, **SC_POINTER_BUTTON_UP**).

The ACSIL variable **sc.MenuEventID** indicates which Advanced Custom Study Control Bar Button is currently selected. The possible values are the constants **ACS_BUTTON_1** through **ACS_BUTTON_150**.

The Pointer location can be determined through the [sc.ActiveToolIndex](#) member, which indicates the chart bar index the Pointer is hovering over. And [sc.ActiveToolYValue](#) member, which

indicates the vertical coordinate value of the Pointer position according to the scale of the Chart Region the Pointer is currently in.

The **sc.ActiveToolIndex** and the **sc.ActiveToolYValue** values are updated with Pointer movements and Pointer click operations. They are updated with the last movement or operation received. When an Advanced Custom Study Control Bar button is pressed, these values are not updated on that event and will be what they last were when using the Pointer over the chart.

sc.SetCustomStudyControlBarButtonText()

ACSIL Function

```
void SetCustomStudyControlBarButtonText(int ControlBarButtonNum, const char*  
ButtonText);
```

The **sc.SetCustomStudyControlBarButtonText()** function is used to change the text for the specified Advanced Custom Study Control Bar button to the **ButtonText** parameter.

There are 150 Advanced Custom Study Control Bar buttons (1-150), and **ControlBarButtonNum** specifies which is being changed. This function is typically paired with a call to **sc.SetCustomStudyControlBarButtonHoverText()** to also change the the text displayed when hovering over the button with the Pointer.

sc.SetCustomStudyControlBarButtonHoverText()

ACSIL Function

```
void SetCustomStudyControlBarButtonHoverText(int ControlBarButtonNum, const char*  
ToolTip);
```

The **sc.SetCustomStudyControlBarButtonHoverText()** function is used to change the text displayed when hovering over the specified Advanced Custom Study Control Bar button with the Pointer, to the **ToolTip** text parameter.

There are 150 Advanced Custom Study Control Bar buttons (1-150), and **ControlBarButtonNum** specifies which is being changed. The **ToolTip** text parameter is also used to provide a description of the button in the Control Bar Customization window. This function is typically paired with a call to **sc.SetCustomStudyControlBarButtonText()** to also change the Advanced Custom Study Control Bar button text.

sc.SetCustomStudyControlBarButtonEnable()

ACSIL Function

```
void SetCustomStudyControlBarButtonEnable(int ControlBarButtonNum, int Enable);
```

The **sc.SetCustomStudyControlBarButtonEnable()** function is used to set state of the Advanced Custom Study Control Bar button specified by the **ControlBarButtonNum** parameter to the value of the **Enable** parameter. **Enable** can be **TRUE** or **FALSE**.

There are 150 Advanced Custom Study Control Bar buttons (1-150), and

ControlBarButtonNum specifies which is being changed.

The Control Bar buttons, 1 through 150, can be toggled On (enabled) and Off (disabled). Visually this state is shown for the Advanced Custom Study Control Bar button by the button being pushed in (On/Enabled) or out (Off/Disabled) on the Control Bar.

Effective with version 2268, there can be multiple Advanced Custom Study Control Bar buttons on/enabled at the same time.

Using Advanced Custom Study Control Bar Buttons for One-Time Events and Not Staying On

It is possible to use Advanced Custom Study Control Bar buttons for one-time events and for the button not to remain in an on or pushed in state.

An Advanced Custom Study Control Bar button can be set to be On or Off with calls to **sc.SetCustomStudyControlBarButtonEnable()**, or it can be manually turned On or Off through the Control Bar itself by the user.

Therefore, when the user manually selects an Advanced Custom Study Control Bar button and therefore causing it to be in the On state if it was previously Off, the study function instance will then be notified. After the study function has done its processing, then a call can be made to **sc.SetCustomStudyControlBarButtonEnable()** to set the state back to Off. Therefore, this allows a button to be used for a one-time event and the button will not remain On.

sc.SetCustomStudyControlBarButtonColor()

ACSIL Function

```
void SetCustomStudyControlBarButtonColor(int ControlBarButtonNum, const uint32_t Color);
```

The **sc.SetCustomStudyControlBarButtonColor()** function is used to set the background color of the Custom Study Control Bar button specified by the **ControlBarButtonNum** parameter to the color value specified by the **Color** parameter. **Color** is an [RGB color value](#).

There are 150 Custom Study Control Bar buttons (1-150), and **ControlBarButtonNum** specifies which is being changed.

sc.GetCustomStudyControlBarButtonEnableState()

ACSIL Function

```
int GetCustomStudyControlBarButtonEnableState(int ControlBarButtonNum);
```

The **sc.GetCustomStudyControlBarButtonEnableState()** function returns the state of the Advanced Custom Study Control Bar button specified by the **ControlBarButtonNum** parameter.

1 is returned when the button is in the enabled or On state. 0 is returned when the button is in the

not enabled or Off state.

There are 150 Advanced Custom Study Control Bar buttons (1-150), and **ControlBarButtonNum** specifies which is being changed.

ACS Control Bar Button Keyboard Shortcuts

Keyboard shortcuts can be set up for any of the 150 Advanced Custom Study Control Bar buttons.

To set the keyboard shortcuts, select **Global Settings >> Customize Keyboard Shortcuts**. The Control Bar buttons are listed in the **Custom Study Control Bar Button Keyboard Shortcuts** list. In that list you will see **Custom Study Control Bar Button #** items listed. These correspond to each of the Advanced Custom Study Control Bar buttons 1 through 150.

sc.BlockChartDrawingSelection

The default value for this variable is 0.

The **sc.BlockChartDrawingSelection** ACSIL variable when set to 1 will block Chart Drawing selection on the chart when using your system Pointer. When set to 0, the normal behavior applies to Chart Drawing selection.

Setting this to 1 may be useful when implementing support for Pointer events to prevent the selection of existing Chart Drawings when the user is interacting with the chart with the Pointer.

Be sure to set this variable to 0 when your study function is done with handling chart Pointer events.

When using [Advanced Custom Study Control Bar Buttons and Pointer Events](#) to implement a custom drawing tool when drawing a user drawn drawing on the chart, you should set **sc.BlockChartDrawingSelection** to 1 when you start the drawing on the first SC_POINTER_BUTTON_DOWN event, and then set it to 0 when user drawn drawing is complete after the final SC_POINTER_BUTTON_DOWN event. This will block the selection of the user drawn drawing that has been added, on the second SC_POINTER_BUTTON_DOWN event and allow the mouse clicks to be properly routed to the custom study.

Maintaining Multiple On States For Advanced Custom Study Control Bar Buttons

This section is not directly relevant with version 2268 and higher but this technique can still be useful and informative.

As is documented, only one Advanced Custom Study Control Bar button can be active at any time for a particular chart. Therefore, selecting an Advanced Custom Study Control Bar button also effectively disables the currently active Advanced Custom Study Control Bar button.

If you need to be able to support multiple control bar buttons to be enabled at the same time, an alternative is that when an Advanced Custom Study Control Bar button is enabled, internally invert a boolean state variable in your custom study that the button is associated with, to

remember the current state. After that [disable the button](#) so that it is no longer in an on and pushed in-state.

After this, [change the text](#) or the [background color](#) of the Control Bar button to indicate the current state (whether on or off).

Disabling Previously Selected Advanced Custom Study Control Bar Button

With version 2268 and higher, it is supported that multiple Advanced Custom Study Control Bar buttons can be active or in an On state at the same time.

Use the code below in your custom study function to use the old behavior to only support one button active or on at the same time.

```
if (sc.MenuEventID >= ACS_BUTTON_1 && sc.MenuEventID <= ACS_BUTTON_150)
{
    sc.SetCustomStudyControlBarButtonEnable(sc.PriorSelectedCustomStudyControlBarButtonNumber, 0);
}
```

*Last modified Wednesday, 22nd February, 2023.