

ACSIL Interface Members - sc.Subgraph Array

Related Documentation

- [ACSIL Interface Members - Introduction](#)
 - [ACSIL Interface Members - Variables and Arrays](#)
 - [ACSIL Interface Members - sc.Input Array](#)
 - **ACSIL Interface Members - sc.Subgraph Array**
 - [ACSIL Interface Members - Functions](#)
-

On This Page

- [sc.Subgraph\[\]](#)
- [sc.Subgraph\[\] Structure Members](#)
 - [sc.Subgraph\[\].Data\[\] / sc.Subgraph\[\]\[\]](#)
 - [sc.Subgraph\[\].GetArraySize\(\)](#)
 - [sc.Subgraph\[\].Arrays\[\]\[\]](#)
 - [sc.Subgraph\[\].Name](#)
 - [sc.Subgraph\[\].PrimaryColor](#)
 - [sc.Subgraph\[\].SecondaryColor](#)
 - [sc.Subgraph\[\].SecondaryColorUsed](#)
 - [sc.Subgraph\[\].DataColor\[\]](#)
 - [sc.Subgraph\[\].DrawStyle](#)
 - [sc.Subgraph\[\].LineStyle](#)
 - [sc.Subgraph\[\].LineWidth](#)
 - [sc.Subgraph\[\].LineLabel](#)
 - [sc.Subgraph\[\].DisplayNameValueInWindowsFlags](#)
 - [sc.Subgraph\[\].AutoColoring](#)
 - [sc.Subgraph\[\].DrawZeros](#)
 - [sc.Subgraph\[\].GraphicalDisplacement](#)
 - [sc.Subgraph\[\].ExtendedArrayElementsToGraph](#)
 - [sc.Subgraph\[\].TextDrawStyleText](#)
 - [sc.Subgraph\[\].ShortName](#)
 - [sc.Subgraph\[\].IncludeInStudySummary](#)
 - [sc.Subgraph\[\].UseStudySummaryCellBackgroundColor](#)
 - [sc.Subgraph\[\].StudySummaryCellBackgroundColor](#)
 - [sc.Subgraph\[\].StudySummaryCellText](#)
 - [sc.Subgraph\[\].UseLabelsColor](#)

- [sc.Subgraph\[\].LabelsColor](#)
- [sc.Subgraph\[\].DisplayNameValueInDataLine](#)
- [Numeric Information Table Graph Draw Type](#)
 - [s_NumericInformationGraphDrawTypeConfig](#)
 - [sc.SetNumericInformationGraphDrawTypeConfig\(\)](#)
 - [sc.SetNumericInformationDisplayOrderFromString\(\)](#)
 - [Numeric Information Graph Example](#)

sc.Subgraph[]

Type: Array of study Subgraph structures.

sc.Subgraph[] is an array of the subgraphs available to the study. There is currently a maximum of **SC_SUBGRAPHS_AVAILABLE** (60) subgraphs available for your study to use.

Subgraphs have two purposes. The first is to display data which is part of the study onto the chart. The individual drawings in a study graph are considered Subgraphs. The second purpose is to hold data for background calculations or to hold data that needs to be maintained between function calls.

If you are using the [sc.Subgraph\[\].Data\[\]](#) member array of a Subgraph for the second purpose, then do not name a Subgraph unless you want the data in the `sc.Subgraph[].Data` array to appear on the chart. By default, Subgraphs do not have names unless you set them. If you do want to make the background data visible for debugging purposes, then a Subgraph can have a name. However, in this case set its draw style to **DRAWSTYLE_IGNORE**. This is very useful for debugging. The data can be viewed in the **Window >> Chart Values Window**.

There are also the Extra Arrays to hold data for background calculations and hold data that needs to be maintained between function calls. Refer to the [Extra Arrays](#) member of this Subgraph structure.

References

A useful method to make it easier to work with a **sc.Subgraph[]** and the **sc.Subgraph[].Data** array is to use a C++ reference. A reference is defined with **SCSubgraphRef**. **SCSubgraphRef** is a reference to the `sc.Subgraph[]` type. Below is an example of defining and using a reference.

Example

```
// Make a reference to the second Subgraph and c  
SCSubgraphRef PlotB = sc.Subgraph[1];
```

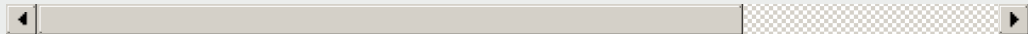
```
// Now the PlotB reference can be used in place of
```

```
// Set the value of the element in the Subgraph Data  
// current index to 10.
```

```
// This is the same as sc.Subgraph[1][sc.Index] = 1  
PlotB[sc.Index] = 10.0f;
```

```
// Calculate the simple moving average and store it  
// the Data array of PlotB (sc.Subgraph[1]).
```

```
sc.SimpleMovAvg(sc.BaseDataIn[SC_LAST], PlotB
```



sc.Subgraph[] Structure Members

sc.Subgraph[].Data[] / sc.Subgraph[][]

Read/Write. Array of float variables (SCFloatArray).

sc.Subgraph[].Data[] is the array of values for the study Subgraph. This is where you will store the results of the study calculations, and this is the data that will be graphed on the chart if the sc.Subgraph has the **Name** member set and has a visible **DrawStyle**.

The size of this array is equal to [sc.ArraySize](#). If you have set [sc.IsCustomChart](#), the size of this array is equal to [sc.OutArraySize](#).

When a chart is reloaded, when the study is first added to a chart, or when a Chartbook is opened and the study exists on one of the charts, then all of the **sc.Subgraph[].Data[]** array elements are initialized to 0.

If you are familiar with the Sierra Chart Spreadsheet Studies, it may help to think of this Subgraph Data array as a column of data in a Spreadsheet (formula columns K through Z). The Spreadsheet Studies uses these same Subgraph Data arrays to hold the results from the formula columns.

A shorthand method to access a Subgraph Data array element exists. Example: **sc.Subgraph[0]**

[sc.Index] is equivalent to **sc.Subgraph[0].Data[sc.Index]**. When passing a Subgraph Data array to an array based study function such as **sc.Highest()**, you do not need to use the second set of brackets. For example, use **sc.Subgraph[0]** or **sc.Subgraph[0].Data**.

For more information about the **sc.Subgraph** structure and the purpose of Subgraphs, refer to [sc.Subgraph\[\]](#).

For information about indexing and array sizes refer to [Array Indexing and Sizes](#).

Whenever first accessing an element of the **sc.Subgraph[].Data** array at **sc.Subgraph[]** index 1 or higher, at that time causes an allocation of the memory required for that Data array.

The use of the **[]** operator on the **Data** member (**sc.Subgraph[].Data[]**), returns a reference to that particular element and you can both get and set the element.

Example


```
// Set the value of the element at the current index in the third Subgraph to 12.5
sc.Subgraph[2][sc.Index] = 12.5f;

// Get the value of the element of the current index in the third Subgraph
float SubgraphValue = sc.Subgraph[2][sc.Index];
```

Example

```
// Calculate the exponential moving average with a
// prices from the chart and store the result in the s
sc.ExponentialMovAvg(sc.BaseDataIn[SC_LAST],

// Calculate the simple moving average with a Len
// moving average that was calculated above. To d
// Subgraph (sc.Subgraph[1]) as the input for the s
// Store the result in the first Subgraph (sc.Subgrap
sc.SimpleMovAvg(sc.Subgraph[1], sc.Subgraph[0];
```



sc.Subgraph[].GetArraySize()

Type: Function.

The **sc.Subgraph[].GetArraySize()** function gets the number of **sc.Subgraph[].Data[]** arrays within the **sc.Subgraph[]**. Normally, this will return the maximum number of Subgraphs. As of 2013-8 this is 60.

sc.Subgraph[].Arrays[][]

Type: Read/Write array of float arrays (SCFloatArray).

Arrays[Index1][Index2]: This member is an array of arrays to be used for storing background or intermediate calculations and to store data that needs to be held between function calls.

As of 2016-02 there are 12 extra arrays for each **sc.Subgraph[]**.

The arrays referred to by **sc.Subgraph[].Arrays[]** are of type SCFloatArray. The reference type to them is **SCFloatArrayRef**.

These extra array elements can be accessed by using **sc.Subgraph[].Arrays[Index1][Index2]**. Where **Index1** can be from 0 to (12 - 1) and represent these internal Subgraph arrays. Where **Index2** is the same index value as used with the **sc.Subgraph[].Data[]** arrays (This accesses the primary graphable Data array for the Subgraph).

Each of the **sc.Subgraph[].Arrays[][]** arrays has the same size as **sc.Subgraph[].Data[]**. A working example can be found in the **scsf_ExtraArraysExample** function in the **/ACS_Source/studies.cpp** file.

As with all arrays used in ACSIL, these are safe and using an index which is out of bounds does not cause any harm. It is simply adjusted to be within the bounds. For example, if Index2 is -1, it will be adjusted to 0.

Note: Some study functions that take arrays for input and output require a reference to a **sc.Subgraph** and not a reference to a **SCFloatArray** contained within a **sc.Subgraph**. An example is **sc.MACD()**.


These functions will use the extra arrays contained within the passed **sc.Subgraph** (**sc.Subgraph[].Arrays[]**). Usually they will use 2 or 3 extra arrays, but it could be up to 6 to 8 extra arrays. After passing a Subgraph to one of these functions, you do not want to use one of these extra arrays in the **sc.Subgraph** for another purpose by writing to it.

References

To do a reference to an Extra Array to make accessing it easier, define a reference like this:

Example

```
SCFloatArrayRef myArray = sc.Subgraph[0].Array;  
  
// Set the array element at the current index to 10.  
myArray[sc.Index] = 10;
```



Passing Extra Arrays to Functions

To pass an Extra Array to a function you simply define the parameter type as **SCFloatArrayRef**. Refer to the **scsf_PassingExtraArray** function in the /ACS_Source/studies.cpp file in the folder Sierra Chart is installed to.

sc.Subgraph[].Name

Read/Write string variable.

Initial value: "" (empty string)

sc.Subgraph[].Name is the name of the Subgraph. If there is no name, the Subgraph will not be drawn, and it will not be displayed on the list of Subgraphs in the **Subgraphs** tab on the Study Settings window. It is useful to use a sc.Subgraph to store some background or intermediate calculation that should not be displayed. This is a good example of when you would want to leave the **Name** blank, so that it will not be graphed on the chart.

Example:

```
// Set the name of the first Subgraph  
sc.Subgraph[0].Name = "First Subgraph";
```

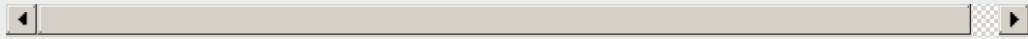
sc.Subgraph[].PrimaryColor

Read/Write color variable.

sc.Subgraph[].PrimaryColor is the primary color for the Subgraph. This is the only color for the Subgraph if the secondary color is not used. For more information on colors, refer to [RGB Color Values](#).

Example

```
// Set the primary color for the first Subgraph to red  
sc.Subgraph[0].PrimaryColor = RGB(255,0,0);
```



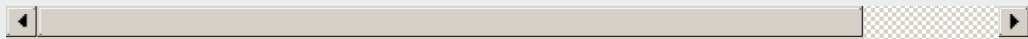
sc.Subgraph[].SecondaryColor

Read/Write color variable.

sc.Subgraph[].SecondaryColor is the secondary color of the Subgraph. For more information on colors, refer to [RGB Color Values](#).

Example

```
// Set the secondary color for the first Subgraph to blue  
sc.Subgraph[0].SecondaryColor = RGB(255,255,0)
```



sc.Subgraph[].SecondaryColorUsed

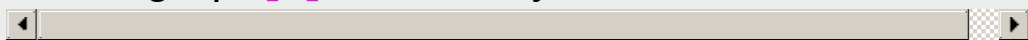
Read/Write variable.

Initial value: 0 (FALSE)

sc.Subgraph[].SecondaryColorUsed can be a TRUE (1) or FALSE (0) value indicating that the secondary color of the Subgraph is used. When this is set to 1 (TRUE) the secondary color is made available for the Subgraph in the Subgraphs tab on the Technical Study Settings window. Setting this does not automatically color your Subgraph based on it's slope, however if the Auto-Coloring option is on for the Subgraph, then this secondary color is used.

Example

```
// Enable the secondary color for the first Subgraph  
sc.Subgraph[0].SecondaryColorUsed = 1;
```



sc.Subgraph[].DataColor[]

Read/Write array of Integer color values.

sc.Subgraph[].DataColor[] is an array of RGB (unsigned Integer) color values associated with each element of a **sc.Subgraph[].Data[]** array.

If you use this array, the **sc.Subgraph[].Data** elements will be drawn using the colors in this array rather than the primary color of the **sc.Subgraph[]**.

The **DataColor** array has the same number of elements as the **sc.Subgraph[].Data** array. The color in each element of this array will line up directly with the value in each element of the **sc.Subgraph[].Data** array.

The colors in this array are unset unless you set them. For more information on colors, refer to [RGB Color Values](#).

For information about indexing and array sizes, refer to [Array Indexing and Sizes](#).

For a code example, refer to **scsf_SimpMovAvgColored** in **studies.cpp** inside the **ACS_Source** folder inside of the Sierra Chart installation folder.

Colors for Price Bar Graph Draw Types

In the case where you are using a **sc.GraphDrawType** other than **GDT_CUSTOM**, then the following details how the **sc.Subgraph[].DataColor[]** arrays affect the elements of each price bar Graph Draw Type.

- **Candlesticks**
 - **CandleUpOutlineColor** = **sc.Subgraph[SC_OPEN].DataColor[]**
 - **CandleUpFillColor** = **sc.Subgraph[SC_HIGH].DataColor[]**
 - **CandleDownOutlineColor** = **sc.Subgraph[SC_LOW].DataColor[]**
 - **CandleDownFillColor** = **sc.Subgraph[SC_LAST].DataColor[]**

Using **sc.Subgraph[].DataColor[]** Array for **GDT_NUMERIC_INFORMATION** **sc.GraphDrawType**

The **sc.Subgraph[].DataColor[]** array can be used to set the foreground and background colors of each element of a Subgraph displayed in a **GDT_NUMERIC_INFORMATION** table. **GDT_NUMERIC_INFORMATION** is set through [sc.GraphDrawType](#).

To be able to set the foreground and background color requires that these colors be combined into a single 4 byte color value by using the [sc.CombinedForegroundColorRef](#) function.

Example


```
// Set the color of the Data element at the current I  
// for the third Subgraph (Subgraph[2]) to Blue.
```

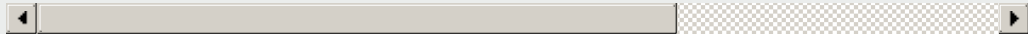
```
sc.Subgraph[2].DataColor[sc.Index] = RGB(0,0,25
```

```
// Set the color of the Data element at the current I  
// for the third Subgraph (Subgraph[2]) to the Prima
```

```
sc.Subgraph[2].DataColor[sc.Index] = sc.Subgrap
```

```
// Set the color of the Data element at the current I  
// for the third Subgraph (Subgraph[2]) to the Seco
```

```
sc.Subgraph[2].DataColor[sc.Index] = sc.Subgrap
```



sc.Subgraph[].DrawStyle

Read/Write Integer variable.

Initial value: **DRAWSTYLE_LINE** or **DRAWSTYLE_IGNORE**

sc.Subgraph[].DrawStyle is the Draw Style that is used to draw the Subgraph. These are relevant when **sc.GraphDrawType** is set to **GDT_CUSTOM**. This is the default setting. The Draw Styles you can use are as follows:

- **DRAWSTYLE_LINE**
- **DRAWSTYLE_BAR**
- **DRAWSTYLE_POINT**
- **DRAWSTYLE_DASH**
- **DRAWSTYLE_HIDDEN**
- **DRAWSTYLE_IGNORE**
- **DRAWSTYLE_STAIR_STEP**
- **DRAWSTYLE_SQUARE**
- **DRAWSTYLE_STAR**
- **DRAWSTYLE_PLUS**
- **DRAWSTYLE_ARROW_UP**
- **DRAWSTYLE_ARROW_DOWN**
- **DRAWSTYLE_ARROW_LEFT**
- **DRAWSTYLE_ARROW_RIGHT**
- **DRAWSTYLE_FILL_TOP**
- **DRAWSTYLE_FILL_BOTTOM**

- **DRAWSTYLE_FILL_RECTANGLE_TOP**
- **DRAWSTYLE_FILL_RECTANGLE_BOTTOM**
- **DRAWSTYLE_COLOR_BAR**
- **DRAWSTYLE_BOX_TOP**
- **DRAWSTYLE_BOX_BOTTOM**
- **DRAWSTYLE_COLOR_BAR_HOLLOW**
- **DRAWSTYLE_COLOR_BAR_CANDLE_FILL**
- **DRAWSTYLE_CUSTOM_TEXT** (This is for internal Sierra Chart use only. It is not an actual visible Draw Style. It is used only to set the Color and Font height for text drawn on the chart.)
- **DRAWSTYLE_BAR_TOP**
- **DRAWSTYLE_BAR_BOTTOM**
- **DRAWSTYLE_LINE_SKIP_ZEROS**
- **DRAWSTYLE_TRANSPARENT_FILL_TOP**
- **DRAWSTYLE_TRANSPARENT_FILL_BOTTOM**
- **DRAWSTYLE_TEXT** (Supports the use of the newline character, \n, in the text string)
- **DRAWSTYLE_POINT_ON_LOW**
- **DRAWSTYLE_POINT_ON_HIGH**
- **DRAWSTYLE_TRIANGLE_UP**
- **DRAWSTYLE_TRIANGLE_DOWN**
- **DRAWSTYLE_TRANSPARENT_FILL_RECTANGLE_TOP**
- **DRAWSTYLE_TRANSPARENT_FILL_RECTANGLE_BOTTOM**
- **DRAWSTYLE_BACKGROUND** (example function: `scsf_BackgroundDrawStyleExample` in `studies8.cpp`)
- **DRAWSTYLE_DIAMOND**
- **DRAWSTYLE_LEFT_PRICE_BAR_DASH**
- **DRAWSTYLE_RIGHT_PRICE_BAR_DASH**
- **DRAWSTYLE_TRIANGLE_LEFT**
- **DRAWSTYLE_TRIANGLE_RIGHT**
- **DRAWSTYLE_TRIANGLE_RIGHT_OFFSET**
- **DRAWSTYLE_TRIANGLE_RIGHT_OFFSET_FOR_CANDLESTICK**
- **DRAWSTYLE_CANDLESTICK_BODY_OPEN**
- **DRAWSTYLE_CANDLESTICK_BODY_CLOSE**
- **DRAWSTYLE_FILL_TO_ZERO**
- **DRAWSTYLE_TRANSPARENT_FILL_TO_ZERO**
- **DRAWSTYLE_SQUARE_OFFSET_LEFT**
- **DRAWSTYLE_SQUARE_OFFSET_LEFT_FOR_CANDLESTICK**
- **DRAWSTYLE_VALUE_ON_HIGH**
- **DRAWSTYLE_VALUE_ON_LOW**
- **DRAWSTYLE_VALUE_OF_SUBGRAPH**
- **DRAWSTYLE_SUBGRAPH_NAME_AND_VALUE_LABELS_ONLY**
- **DRAWSTYLE_LINE_AT_LAST_BAR_TO_EDGE**
- **DRAWSTYLE_FILL_RECTANGLE_TO_ZERO**
- **DRAWSTYLE_TRANSPARENT_FILL_RECTANGLE_TO_ZERO**
- **DRAWSTYLE_X**

- **DRAWSTYLE_CUSTOM_VALUE_AT_Y**: This Draw Style is meant to be used with ACSIL. The `sc.Subgraph[].Data[]` array contains the custom value for a bar index. The Chart Region y-coordinate is controlled through the `sc.Subgraph[].Arrays[0][]` array. The y-coordinate is based on the study scale values.

As of version 2526, text alignment for **DRAWSTYLE_CUSTOM_VALUE_AT_Y** and **DRAWSTYLE_TRANSPARENT_CUSTOM_VALUE_AT_Y**, can be optionally controlled through the `sc.Subgraph[].Arrays[2][]` array. Can be one of: **TA_LEFT**, **TA_RIGHT**, **TA_CENTER** and one of **TA_TOP**, **TA_BOTTOM** separated by the bitwise OR operator |.

A vertical offset based on text height can be controlled through the `sc.Subgraph[].Arrays[1][]` array. A positive value offsets the text up by this set value times the text height. A negative value offsets the text down by this set value times the text type.

When the Subgraph Secondary Color is used, then the background color is going to be the secondary color.

- **DRAWSTYLE_CUSTOM_VALUE_AT_Y_LEFT_ALIGNED**: For more details, refer to **DRAWSTYLE_CUSTOM_VALUE_AT_Y**.
- **DRAWSTYLE_CUSTOM_VALUE_AT_Y_RIGHT_ALIGNED**: For more details, refer to **DRAWSTYLE_CUSTOM_VALUE_AT_Y**.
- **DRAWSTYLE_CUSTOM_VALUE_AT_Y_WITH_BORDER**: For more details, refer to **DRAWSTYLE_CUSTOM_VALUE_AT_Y**.
- **DRAWSTYLE_TRANSPARENT_CUSTOM_VALUE_AT_Y**: For more details, refer to **DRAWSTYLE_CUSTOM_VALUE_AT_Y**.
- **DRAWSTYLE_TRANSPARENT_CUSTOM_VALUE_AT_Y_LEFT_ALIGNED**: For more details, refer to **DRAWSTYLE_CUSTOM_VALUE_AT_Y**.
- **DRAWSTYLE_TRANSPARENT_CUSTOM_VALUE_AT_Y_RIGHT_ALIGNED**: For more details, refer to **DRAWSTYLE_CUSTOM_VALUE_AT_Y**.
- **DRAWSTYLE_TRANSPARENT_BAR_TOP**
- **DRAWSTYLE_TRANSPARENT_BAR_BOTTOM**
- **DRAWSTYLE_LEFT_OFFSET_BOX_TOP**
- **DRAWSTYLE_LEFT_OFFSET_BOX_BOTTOM**
- **DRAWSTYLE_RIGHT_OFFSET_BOX_TOP**
- **DRAWSTYLE_RIGHT_OFFSET_BOX_BOTTOM**
- **DRAWSTYLE_HORIZONTAL_PROFILE**
- **DRAWSTYLE_HORIZONTAL_PROFILE_HOLLOW**
- **DRAWSTYLE_SQUARE_OFFSET_RIGHT**
- **DRAWSTYLE_SQUARE_OFFSET_RIGHT_FOR_CANDLESTICK**
- **DRAWSTYLE_TRANSPARENT_CIRCLE**
- **DRAWSTYLE_CIRCLE_HOLLOW**
- **DRAWSTYLE_TRANSPARENT_CIRCLE_VARIABLE_SIZE** (The `sc.Subgraph[].Data[]` array contains the Chart Region y- coordinate. The circle size in pixels is controlled through the `sc.Subgraph[].Arrays[0][]` array.)

- **DRAWSTYLE_CIRCLE_HOLLOW_VARIABLE_SIZE**
- **DRAWSTYLE_POINT_VARIABLE_SIZE**
- **DRAWSTYLE_LINE_EXTEND_TO_EDGE**
- **DRAWSTYLE_BACKGROUND_TRANSPARENT**
- **DRAWSTYLE_LEFT_SIDE_TICK_SIZE_RECTANGLE**
- **DRAWSTYLE_RIGHT_SIDE_TICK_SIZE_RECTANGLE**
- **DRAWSTYLE_TRANSPARENT_TEXT** (Supports the use of the newline character, \n, in the text string)
- **DRAWSTYLE_TRANSPARENT_TEXT_WITH_ALIGNMENT**: This draw style is meant to be used with ACSIL. It is the same as **DRAWSTYLE_TRANSPARENT_TEXT**.

Text alignment can be optionally controlled through the `sc.Subgraph[].Arrays[0][]` array. Can be one of: `TA_LEFT`, `TA_RIGHT`, `TA_CENTER` and one of `TA_TOP`, `TA_BOTTOM` separated by the bitwise OR operator `|`.

- **DRAWSTYLE_TEXT_WITH_BACKGROUND**: (Supports the use of the newline character, \n, in the text string)

For descriptions and more information about each of the above Draw Styles, refer to [Draw Style](#) on the **Chart Studies** documentation page.

When using the Color Bar type of styles (**DRAWSTYLE_COLOR_BAR**, **DRAWSTYLE_COLOR_BAR_HOLLOW**, **DRAWSTYLE_COLOR_BAR_CANDLE_FILL**), these will color the existing chart bars. Typically you will set the **sc.GraphRegion** to 0 when using these draw styles. To color a particular bar in the chart, you will set a **sc.Subgraph[].Data[]** array element to any nonzero value to color the corresponding chart bar. The color that will be used will be the **sc.Subgraph [].PrimaryColor** unless you are using the [sc.Subgraph\[\].DataColor](#) array.

For more information, refer to the [Color Bar](#) style. For an example, see the `scsf_ColorBarOpenClose` in the `studies.cpp` file inside the `/ACS_Source` folder inside of the Sierra Chart installation folder.

When you use a **sc.GraphDrawType** setting value other than `GDT_CUSTOM`, then the `sc.Subgraph[].DrawStyle` variable is automatically set for each of the relevant `sc.Subgraphs` needed by the **sc.GraphDrawType**. You cannot change them. Additionally, it is not possible when you are drawing a price bar type of graph (`GraphDrawType` not equal to `GDT_CUSTOM`), to also use standard study lines or other Draw Styles using the other available `sc.Subgraphs` which are not used by the **sc.GraphDrawType**. In this case you will need to use a separate study for those.

When you use **DRAWSTYLE_TEXT** this means the specified text is drawn at each bar/column in the chart at the value specified in the corresponding Subgraph Data element. The actual text is specified with **sc.Subgraph[].TextDrawStyleText**. The font height is specified through **sc.Subgraph[].LineWidth**. If **sc.Subgraph[].DrawZeros** is 0, and the `sc.Subgraph Data` element for a bar/column in the chart is set to zero, no text will be drawn.

The **sc.Subgraph[].DrawStyle** for a study Subgraph sets the draw style for that entire Subgraph for every chart column the Subgraph is drawn in. Any changes to the Draw Style affect all chart

column elements of the Subgraph when the chart is drawn. Therefore, it is not possible to use different Draw Styles for different elements of a single study Subgraph. If you want different Draw Styles, it is necessary to use separate Subgraphs for each Draw Style that you want to use.

Example

```
// Set the draw style of the first Subgraph to the st  
sc.Subgraph[0].DrawStyle = DRAWSTYLE_STAIF
```

sc.Subgraph[].LineStyle

Read/Write variable.

Initial value: **LINESTYLE_SOLID**

sc.Subgraph[].LineStyle is the style with which lines are drawn. This only applies to subgraphs where [sc.Subgraph\[\].DrawStyle](#) is **DRAWSTYLE_LINE**, **DRAWSTYLE_BAR**, **DRAWSTYLE_DASH**, or **DRAWSTYLE_STAIR_STEP**. The line styles you can use are as follows:

- **LINESTYLE_SOLID**
- **LINESTYLE_DASH**
- **LINESTYLE_DOT**
- **LINESTYLE_DASHDOT**
- **LINESTYLE_DASHDOTDOT**

Line styles only work when [sc.Subgraph\[\].LineWidth](#) is set to 0 or 1. If the line width is greater than 1, the line will appear solid.

Example

```
// Set the line style of the first Subgraph to the "do  
sc.Subgraph[0].LineStyle = LINESTYLE_DOT;
```

sc.Subgraph[].LineWidth

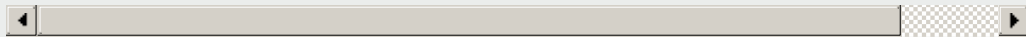
Read/Write variable.

Initial value: 1

sc.Subgraph[].LineWidth is the width in pixels for the Subgraph Draw Style. Not all the available Draw Styles will support a Line Width. When the Draw Style is set to DRAWSTYLE_TEXT, this controls the font height. In this case, setting this to 10 will mean a 10 point height.

Example

```
// Set the line width of the second Subgraph to 2 pixels  
sc.Subgraph[1].LineWidth = 2;
```



sc.Subgraph[].LineLabel

Read/Write variable.

Initial value: 0

sc.Subgraph[].LineLabel can be set to a set of flags to enable displaying and positioning of the Name and/or Value of the Subgraph. You can set this to a combination of the following flags:

- LL_DISPLAY_NAME
- LL_NAME_ALIGN_CENTER
- LL_NAME_ALIGN_FAR_RIGHT
- LL_NAME_ALIGN_ABOVE
- LL_NAME_ALIGN_BELOW
- LL_NAME_ALIGN_LEFT
- LL_NAME_ALIGN_RIGHT
- LL_NAME_ALIGN_VALUES_SCALE
- LL_NAME_ALIGN_LEFT_EDGE
- LL_DISPLAY_VALUE
- LL_VALUE_ALIGN_CENTER
- LL_VALUE_ALIGN_FAR_RIGHT
- LL_VALUE_ALIGN_ABOVE
- LL_VALUE_ALIGN_BELOW
- LL_VALUE_ALIGN_RIGHT
- LL_VALUE_ALIGN_VALUES_SCALE
- LL_VALUE_ALIGN_LEFT_EDGE
- LL_VALUE_ALIGN_LEFT
- LL_NAME_REVERSE_COLORS
- LL_VALUE_REVERSE_COLORS_INV
- LL_NAME_ALIGN_DOM_LABELS_COLUMN
- LL_VALUE_ALIGN_DOM_LABELS_COLUMN

- LL_DISPLAY_CUSTOM_VALUE_AT_Y
- LL_NAME_ALIGN_LEFT_SIDE_VALUES_SCALE
- LL_VALUE_ALIGN_LEFT_SIDE_VALUES_SCALE

Example

```
// Set the second Subgraph to display it's name on  
sc.Subgraph[1].LineLabel = LL_DISPLAY_NAME
```



sc.Subgraph[].DisplayNameValueInWindowsFlags

Read/Write Integer variable.

Initial value: **SNV_DISPLAY_IN_WINDOWS | SNV_DISPLAY_IN_DATA_LINE**

sc.Subgraph[].DisplayNameValueInWindowsFlags can be set to a set of flags to enable displaying of the Subgraph's Name and Value on the Region Data Line on the chart window and/or in the Chart Values Windows.

- SNV_DISPLAY_IN_WINDOWS
- SNV_DISPLAY_IN_DATA_LINE

Example

```
sc.Subgraph[1].DisplayNameValueInWindowsFlag
```



sc.Subgraph[].AutoColoring

Read/Write variable.

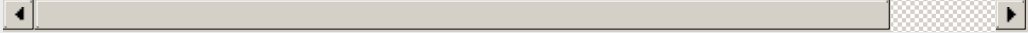
Initial value: **0**

AutoColors a Subgraph. Can be one of the following constants:

- AUTOCOLOR_NONE
- AUTOCOLOR_SLOPE
- AUTOCOLOR_POSNEG
- AUTOCOLOR_BASEGRAPH

Example

```
sc.Subgraph[1].AutoColoring = AUTOCOLOR_SL
```



sc.Subgraph[].DrawZeros

Type: Read/Write variable.

Initial value: **0** (disabled)

sc.Subgraph[].DrawZeros can be a TRUE (1) or FALSE (0) value to enable or disable the drawing of Subgraph Data array elements that have a value of zero.

Set this value to 1 to enable the drawing of zero values. Set this value to 0 to disable the drawing of zero values.

When this is disabled, the Subgraph Draw Style of **DRAWSTYLE_LINE** will draw a continuous line between the chart columns that have non-zero values.

The Subgraph Draw Style of **DRAWSTYLE_LINE_SKIP_ZEROS** can be used to skip over zero values and not draw a line between the last nonzero value and the next nonzero value as an alternative to using **sc.Subgraph[].DrawZeros = 0**.

Example

```
sc.Subgraph[0].DrawZeros = 1;
```

sc.Subgraph[].GraphicalDisplacement

Type: Read/Write Integer variable.

Initial value: **0**

This is either the positive or negative displacement, in chart columns, to shift the **sc.Subgraph []** forward or backward by. A positive number will shift the Subgraph forward and a negative number will shift the Subgraph backward. For more information, refer to [Displacement](#) in the Chart Studies documentation.

Example


```
sc.Subgraph[0].GraphicalDisplacement = 1;
```

sc.Subgraph[].ExtendedArrayElementsToGraph

Type: Read/Write variable.

Initial value: 0

ExtendedArrayElementsToGraph is the number of **sc.Subgraph[].Data[]** array elements at and after **sc.ArraySize** that will be graphed into the extended area on the chart.

Example: **sc.Subgraph[0].Data[sc.ArraySize] = 10;** This line of code will set the value of 10 for Subgraph 0 at the element after the last bar in the chart.

The extended area on the chart are the columns after the very last bar in the chart. You can see this area by scrolling past the right edge of the chart. This is also known as the [Right Side Fill Space/Forward Projection area](#).

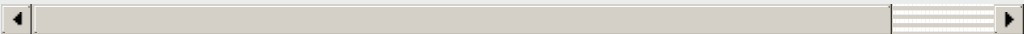
To increase the number of columns after the last bar in the chart, use the

Chart >> Chart Settings >> [Number of Forward Columns](#) setting. The default is 150.

For a code example, refer to the **scsf_ExtendedArrayExample** function in the **/ACS_Source/Studies6.cpp** file in the Sierra Chart installation folder.

Example

```
sc.Subgraph[0].ExtendedArrayElementsToGraph :
```



sc.Subgraph[].TextDrawStyleText

Type: Read/Write SCString variable.

TextDrawStyleText is used to specify the actual text to use with the DRAWSTYLE_TEXT Draw Style.

This Subgraph member can be changed at any time even outside of the **sc.SetDefaults** code block. When it is changed, it applies to all elements of the particular Subgraph it is set on. It is not possible to use different text for each chart bar/column with the DRAWSTYLE_TEXT Draw Style.

Example

```
sc.Subgraph[0].DrawStyle = DRAWSTYLE_TEXT;  
sc.Subgraph[0].TextDrawStyleText = "Buy";  
sc.Subgraph[0].LineWidth = 10; // Use a font height
```



sc.Subgraph[].ShortName

Type: Read/Write SCString variable.

ShortName is used to specify the Subgraph Short Name.

Example

```
// Set the short name of the first Subgraph  
sc.Subgraph[0].ShortName = "FSG";
```

sc.Subgraph[].IncludeInStudySummary

Type: Read/Write Integer variable.

IncludeInStudySummary can be set to 1 to include the Subgraph in the [Study Summary Window](#) or 0 to not include it.

This variable only has an effect, if the study itself is included in the Study Summary window.

sc.Subgraph[].UseStudySummaryCellBackgroundColor

Type: Read/Write Integer variable.

sc.Subgraph[].UseStudySummaryCellBackgroundColor can be set to 1 to cause the color set with [sc.Subgraph\[\].StudySummaryCellBackgroundColor](#) to be used for the cell background color when the study Subgraph is included in the [Study Summary](#) window.

This variable only has an effect, if the study instance itself is included in the Study Summary window.

sc.Subgraph[].StudySummaryCellBackgroundColor

Type: Read/Write Integer color variable.

sc.Subgraph[].StudySummaryCellBackgroundColor only applies when [sc.Subgraph\[\].UseStudySummaryCellBackgroundColor](#) is set to 1.

sc.Subgraph[].StudySummaryCellBackgroundColor is the [RGB Color Value](#) of the cell background color for the study Subgraph when it is displayed in the [Study Summary Window](#).

sc.Subgraph[].StudySummaryCellText

Type: Read/Write SCString variable.

The **sc.Subgraph[].StudySummaryCellText** text string sets the text for the study Subgraph to display in the [Study Summary Window](#) instead of the Subgraph value.

This variable is not supported in the 32-bit version of Sierra Chart.

sc.Subgraph[].UseLabelsColor

Type: Read/Write integer variable.

The **sc.Subgraph[].UseLabelsColor** can be set to 1 or 0 and controls whether a separate color is used for [Subgraph Name and Value Labels](#) instead of the Subgraph primary and secondary color settings.

If it is set to 1, the color is set through the [sc.Subgraph\[\].LabelsColor](#) variable.

sc.Subgraph[].LabelsColor

Type: Read/Write Integer variable.

The **sc.Subgraph[].LabelsColor** only applies when [sc.Subgraph\[\].UseLabelsColor](#) is set to 1 and controls the color of [Subgraph Name and Value Labels](#).

sc.Subgraph[].DisplayNameValueInDataLine

Type: Read/Write Integer variable.

The **sc.Subgraph[].DisplayNameValueInDataLine** variable can be set to 1 (or any nonzero value) or 0 and controls whether to display the Subgraph Name and Value in the Region Data Line of the Chart Region the study is displayed in.

When it is set to 1, the Subgraph Name and Values are displayed. Otherwise, they are not.

sc.Subgraph[].IncludeInSpreadsheet

Type: Read/Write Integer variable.

The **sc.Subgraph[].IncludeInSpreadsheet** variable

sc.Subgraph[].UseTransparentLabelBackground

Type: Read/Write Integer variable.

The **sc.Subgraph[].UseTransparentLabelBackground** variable can be set to 1 (or any nonzero value) or 0 and controls whether the Subgraph labels are transparent. When they are

transparent, only the text is displayed without a background color.

sc.Subgraph[].GradientAngleUnit

Type: Read/Write Float variable.

The `sc.Subgraph[].GradientAngleUnit`

sc.Subgraph[].GradientAngleMax

Type: Read/Write Float variable.

The `sc.Subgraph[].GradientAngleMax`

Numeric Information Table Graph Draw Type

Sierra Chart supports an `sc.GraphDrawType` for ACSIL which displays a table of values in a Chart Region. Refer to the screenshot below which shows the table generated from the `scsf_NumericInformationGraphDrawTypeExample` function. This table can consist of multiple rows which represents `sc.Subgraphs` within a study. And each column relates to a specific chart column and chart bar And corresponds to a particular element within a specific study Subgraph.



The background color of the text.

Default value = **COLOR_BLACK**

- **bool TransparentTextBackground**

Turns on or off the background coloring of the text. A value of **true** removes the background coloring and a value of **false** displays the background coloring.

Default value = **true**.

- **bool LabelsOnRight**

When set to **true** the row labels are displayed on the right, otherwise they are displayed on the left.

Default value = **false**.

- **bool AllowLabelOverlap**

When set to **true** the row values are drawn in the same location as the labels, potentially overwriting the row labels. Otherwise the row values are not displayed in the same location as the labels.

Default Value = **false**.

- **bool DrawGridlines**

When set to **true** the gridlines separating the rows and columns are drawn.

Otherwise the gridlines are not drawn.

Default value = **true**.

- **int GridlineStyleSubgraphIndex**

Defines the index of the subgraph that will be used to define the Gridline Style.

Default value = **-1**.

- **int FontSize**

Allows for a custom font size to be defined. A value of **0** means that it will use the defined Chart Text Font.

Default value = **0**.

- **bool ShowPullback**

A value of **true** will display the pullback column and any data that is available for the pullback column. Otherwise, the pullback column is not displayed. If a subgraph does not have any data defined for the pullback column when it is displayed, then the cell will be blank.

Default value = **false**.

- **bool HideLabels**

A value of **true** will hide the labels so they are not displayed. Otherwise, the labels will be displayed.

Default value = **false**.

- **int ValueFormat[SC_SUBGRAPHS_AVAILABLE]**

Allows a subgraph to be displayed in a specific [Value Format](#).

Default value = **VALUEFORMAT_INHERITED**.

- **int SubgraphOrder[SC_SUBGRAPHS_AVAILABLE]**

Defines the display order of the subgraphs. It is easier to use the **sc.SetNumericInformationDisplayOrderFromString()** function (defined below) rather than set this order directly in this variable.

Default value = Subgraph index order.

- **bool ColorBackgroundBasedOnValuePercent**

A value of **true** will color the background of the cell according to the percentage value of the cell as determined from the **HighestValue[]** and **LowestValue[]** and uses the **PercentCompareThresholds[]** to determine the **Range#UpColor** or **Range#DownColor** to use. For more information refer to [Numbers Bars Calculated Values Background Coloring Logic](#).

Default value = **false**.

- **int DetermineMaxMinForBackgroundColoringFrom**

When **ColorBackgroundBasedOnValuePercent** is set to **true**, then this variable sets whether the Maximum and Minimum values are determined from **All Data** or **Daily Data**.

This variable can not be changed with ACSIL, as there is no ability to control the Maximum and Minimum daily values through ACSIL.

Default value = **0**.

- **float HighestValue[SC_SUBGRAPHS_AVAILABLE]**

The highest value found for the subgraph. This is used to determine the color of the background when **ColorBackgroundBasedOnValuePercent** is **true**.

Default value = **-FLT_MAX**.

- **float LowestValue[SC_SUBGRAPHS_AVAILABLE]**

The lowest value found for the subgraph. This is used to determine the color of the background when **ColorBackgroundBasedOnValuePercent** is **true**.

Default value = **FLT_MAX**.

- **float PercentCompareThresholds[NUMBER_OF_THRESHOLDS]**

An array of percentages that is used to determine the color of the background when **ColorBackgroundBasedOnValuePercent** is **true**. These values need to be assigned as the decimal form of the percentage. For example, if the desired percentages are 25%, 50%, and 75%, these should be assigned as .25, .50, and .75.

Default value = **[0.0, 0.0, 0.0]**.

- **COLORREF Range3UpColor**

The background color that is used when **ColorBackgroundBasedOnValuePercent** is **true** and the percentage value of a cell falls into the Range 3 Up category.
Default value = **COLOR_BLACK**.

- **COLORREF Range2UpColor**

The background color that is used when **ColorBackgroundBasedOnValuePercent** is **true** and the percentage value of a cell falls into the Range 2 Up category.
Default value = **COLOR_BLACK**.

- **COLORREF Range1UpColor**

The background color that is used when **ColorBackgroundBasedOnValuePercent** is **true** and the percentage value of a cell falls into the Range 1 Up category.
Default value = **COLOR_BLACK**.

- **COLORREF Range0UpColor**

The background color that is used when **ColorBackgroundBasedOnValuePercent** is **true** and the percentage value of a cell falls into the Range 0 Up category.
Default value = **COLOR_BLACK**.

- **COLORREF Range0DownColor**

The background color that is used when **ColorBackgroundBasedOnValuePercent** is **true** and the percentage value of a cell falls into the Range 0 Down category.
Default value = **COLOR_BLACK**.

- **COLORREF Range1DownColor**

The background color that is used when **ColorBackgroundBasedOnValuePercent** is **true** and the percentage value of a cell falls into the Range 1 Down category.
Default value = **COLOR_BLACK**.

- **COLORREF Range2DownColor**

The background color that is used when **ColorBackgroundBasedOnValuePercent** is **true** and the percentage value of a cell falls into the Range 2 Down category.
Default value = **COLOR_BLACK**.

COLORREF Range3DownColor

- **COLORREF Range3DownColor**

The background color that is used when

ColorBackgroundBasedOnValuePercent is **true** and the percentage value of a cell falls into the Range 3 Down category.

Default value = **COLOR_BLACK**.

- **bool DifferentPullbackAndLabelsColor**

A value of **true** will use the color defined in **PullbackAndLabelsColor** (see below) to color the text in the Pullback and Labels columns. Otherwise, the pullback and labels text are colored the same as the subgraph for the row.

Default value = **false**.

- **COLORREF PullbackAndLabelsColor**

Defines the color of the text for the Pullback and Labels columns when the **DifferentPullbackAndLabelsColor** is set to **true**.

Default value = **COLOR_WHITE**.

- **bool ColorPullbackBackgroundBasedOnPositiveNegative**

A value of **true** will color the background of the Pullback Column based on whether the value is Positive or Negative. When the value in the Pullback Column is Positive, the Range3UpColor is used for the background, and when the value in the Pullback Column is Negative, the Range3DownColor is used for the background.

Default value = **false**

sc.SetNumericInformationGraphDrawTypeConfig()

Type: Function

```
void SetNumericInformationGraphDrawTypeConfig(const  
s_NumericInformationGraphDrawTypeConfig& NumericInformationGraphDrawTypeConfig)
```

The **sc.SetNumericInformationGraphDrawTypeConfig()** function sets the parameters for the Numeric Information Graph type, which are part of the [s_NumericInformationGraphDrawTypeConfig](#) structure.

The **sc.SetNumericInformationGraphDrawTypeConfig** function is used when **sc.GraphDrawType** is set to **GDT_NUMERIC_INFORMATION** for the study.

Parameters

- **NumericInformationGraphDrawTypeConfig:** The [s_NumericInformationGraphDrawTypeConfig](#) structure that is used to set the parameters for the Numeric Information Graph type.

sc.SetNumericInformationDisplayOrderFromString()

Type:Function

```
void SetNumericInformationDisplayOrderFromString(const SCString&
CommaSeparatedDisplayOrder)
```

The **sc.SetNumericInformationDisplayOrderFromString()** function takes a comma delimited text string, **CommaSeparatedDisplayOrder**, with the one-based `sc.Subgraph[]` indexes in the order that you want to display them in the Numeric Information Graph table. For example, "5,4,1,2" would display Subgraphs in the order given.

This function should be used to set the display order of the Subgraphs, rather than setting the display order using the **SubgraphOrder** member of the `s_NumericInformationGraphDrawTypeConfig` structure.

The **sc.SetNumericInformationDisplayOrderFromString** function is used when **sc.GraphDrawType** is set to **GDT_NUMERIC_INFORMATION**.

Any other `sc.Subgraphs` which will be displayed in the table that are left out of the **CommaSeparatedDisplayOrder** text string will be appended to the end of the table in their normal order.

Parameters

- **CommaSeparatedDisplayOrder:** A string of index numbers (integers between 0 and 59) that are separated by commas that define the order of the subgraphs to be displayed in the Numeric Information Graph. For example - "4, 45, 3, 18" would display the data for subgraphs 4, 45, 3, and 18 in that order from top to bottom.

Numeric Information Graph Example

The function **scsf_NumericInformationGraphDrawTypeExample** is an example of how to use the Numeric Information Graph Draw Type and can be found in the **studies8.cpp** file located in the `/ACS_Source` folder directly under the Sierra Chart installation folder.

*Last modified Monday, 24th July, 2023.