12 Automating Routine Tasks

SigmaPlot uses a VBA®-like macro language to access automation internally. However, whether you have never programmed, or are an expert programmer, you can take advantage of this technology by using the Macro Recorder.

This chapter describes how to use SigmaPlot's Macro Recorder and integrated development environment (IDE). It also contains descriptions of related features accessible in the Macro window, including the Sax Basic programming language, debugging tool, dialog box editor, and user-defined functions.

This chapter contains the following topics:

Creating Macros

Record a macro any time that you find yourself regularly typing the same keystrokes, choosing the same commands, or going through the same sequence of operations.

Before you Record

Before you record the macro:

1. Analyze the task you want to automate. If the macro has more than a few steps, write down an outline of the steps.
2. Rehearse the sequence to make sure you have included every single action.
3. Decide what to call the macro, where to assign it, and where to save it.

Recording a Macro

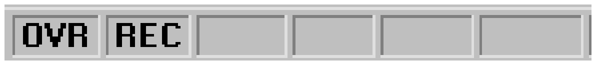
To learn how to place macros on the menu, see “Creating Macros as Menu Commands” on page 243.

To record a macro:

1. On the Tools menu, click Macro, and then click Record New Macro.

   The REC appears in the status area of SigmaPlot’s main window, indicating that the macro is recording your menu selections and keystrokes.

Figure 12–1
The Status Bar



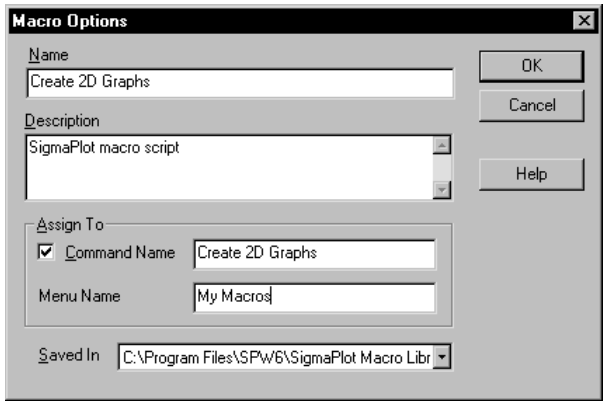
2. Complete the activity you want to include in this macro.

   Note that the Macro Recorder does not record cursor movements.

3. When you are finished recording the macro., on the Tools menu, click Macro, and then click Stop Recording.

   The Macro Recorder stops recording and the Macro Options dialog box appears.

Figure 12–2
Macros Options
Dialog Box



Name: Create 2D Graphs
Description: SigmaPlot macro script
☑ Command Name: Create 2D Graphs
Menu Name: My Macros
Saved In: C:\Program Files\SPW6\SigmaPlot Macro Libr

4. Type a name for the macro in the Name text box.

   Give the macro a descriptive name. You can use a combination of upper- and lowercase letters, numbers, and underscores. For example a macro that formats all of your graph legends to match a certain report might be called “Report1AddFormatToLegend”.

5. Enter a more detailed description in the Description text box.

6. Click OK.

   After you have finished recording the macro, save it globally (for use in all of SigmaPlot) or locally (for use in a particular notebook file).

When you return to the Notebook window, your macro appears in the Notebook.

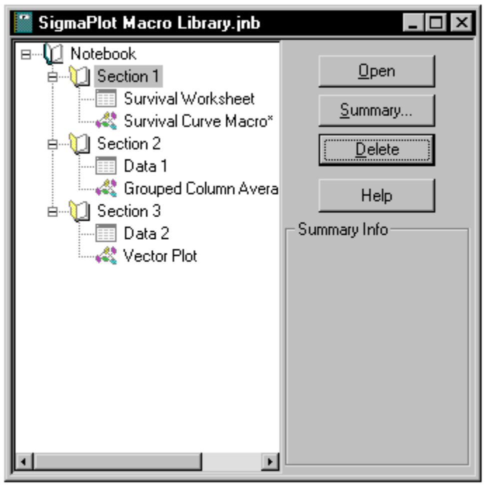


Figure 12–3
Macros associated with notebooks appear in the notebook. Double-click a macro icon to open the Macros dialog box.

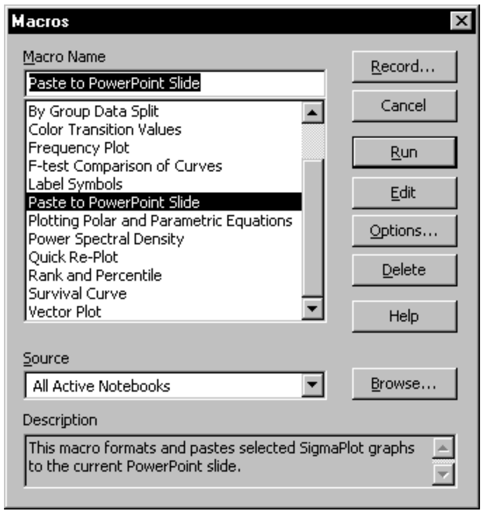
Creating Macros as Menu Commands

You can place your macro as a menu command on the main menu that you specify. For example, your new macro could appear on the main menu under the macro command "My Macros."

To create a new menu command:

1. On the Tools menu, click Macro, and then click Macros.

   The Macros dialog box appears.



This macro formats and pastes selected SigmaPlot graphs to the current PowerPoint slide.

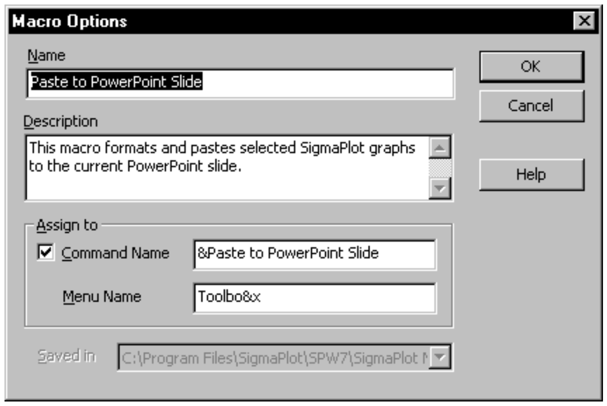
Figure 12–4
Macros Dialog Box

2. Select a macro from the Macro Name scroll-down list.

3. Click Options.

   The Macro Options dialog box appears.

Figure 12–5
Macros Options Dialog Box



Name: Paste to PowerPoint Slide

Description: This macro formats and pastes selected SigmaPlot graphs to the current PowerPoint slide.

Assign to

[✓] Command Name: &Paste to PowerPoint Slide

Menu Name: Toolbo&x

Saved in: C:\Program Files\SigmaPlot\SPW7\SigmaPlot M

4. Select Command Name.
5. Enter the name of the macro in the Command Name field.

   If the Command Name is cleared, the macro doesn't appear on a menu.
6. Enter the name of the menu under which you want the macro to appear in the Menu Name field.
7. Click OK.

   Your new macro appears under the menu command you've just created.
8. Enter the same menu command name in the Menu Name field of future macros if you want them to appear on your new macro command menu.

   By default, if the Menu Name field is left empty, the macro name appears on the Tools menu. You can also create your own menu by entering the menu name in the Menu Name field.

Running Your Macro

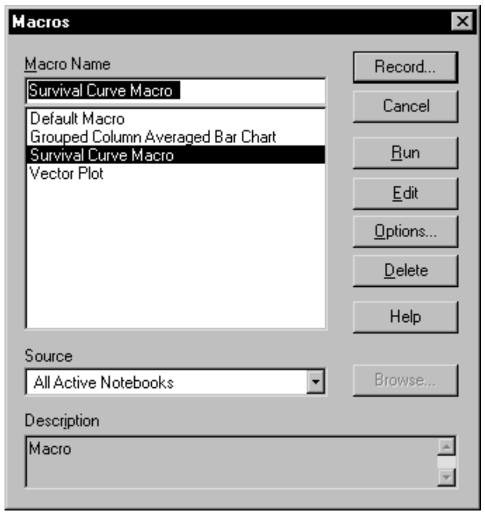
After you have recorded and saved a macro, it's ready to run.

To run a macro from the Macros dialog box:

1. On the Tools menu, click Macro, and then click Macros.

The Macros dialog box appears with a list of available macros.

Figure 12–6
Macros Dialog Box



2. Select the macro to run.
3. Click Run.

To run a macro from a notebook:

1. From within a notebook, double-click the macro icon.

   The Macro dialog box appears with the corresponding macro selected.

2. Click Run.

   If the macro does not have any errors or run into difficulties with your data, it will run to completion.

Σ You can also run a macro from the Macro script window. This is useful for debugging the macro script.

Editing Macros

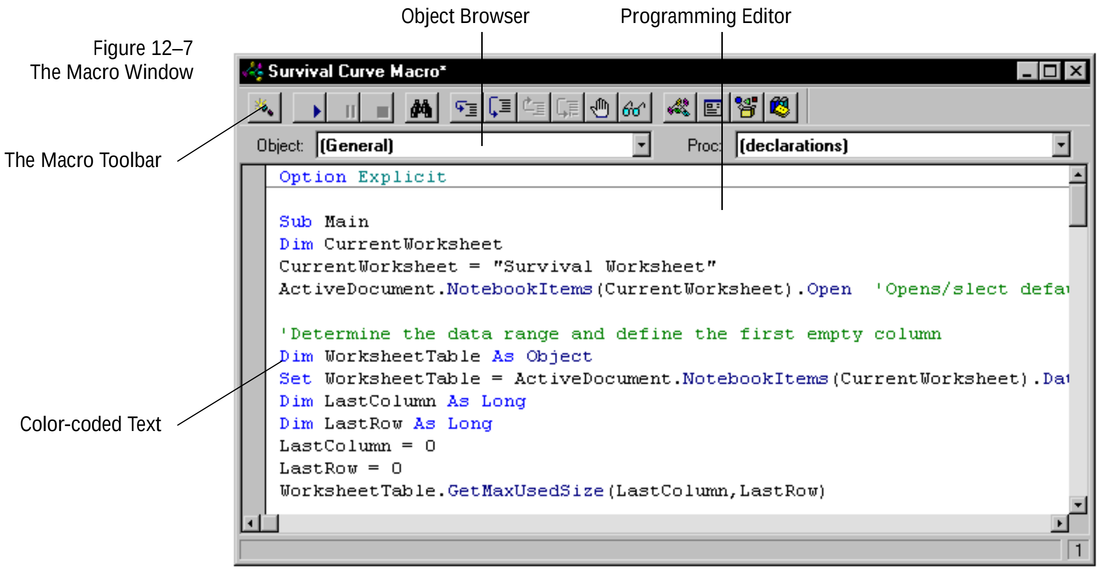
When you record a macro, SigmaPlot generates a series of program statements that are equivalent to the actions that you perform. These statements are in a form of SigmaPlot language that has custom extensions specifically for SigmaPlot automation and appear in the Macro Window. You can edit these statements to modify the actions of the macro. You can also add comments to describe code.

To edit a macro:

1. On the Tools menu, click Macro, and then click Macros.

   The Macros dialog box appears.
2. Select a macro from the Macro list.
3. Click Edit.
4. The Macro Window appears.

Figure 12–7
The Macro Window

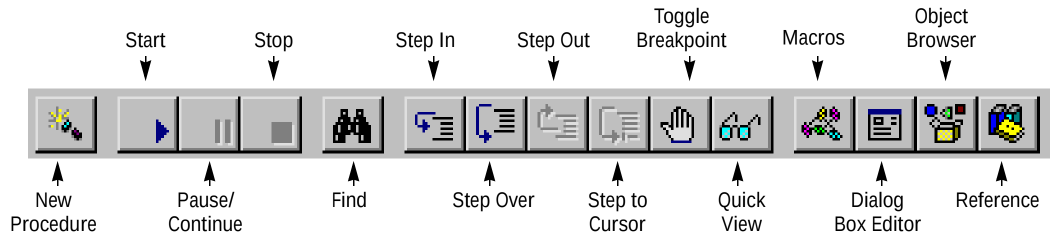


```
Option Explicit

Sub Main
Dim CurrentWorksheet
CurrentWorksheet = "Survival Worksheet"
ActiveDocument.NotebookItems(CurrentWorksheet).Open  'Opens/slect defa

'Determine the data range and define the first empty column
Dim WorksheetTable As Object
Set WorksheetTable = ActiveDocument.NotebookItems(CurrentWorksheet).Da
Dim LastColumn As Long
Dim LastRow As Long
LastColumn = 0
LastRow = 0
WorksheetTable.GetMaxUsedSize(LastColumn,LastRow)
```

Using the Macro Window Toolbar

The Macro Window toolbar appears at the top of the Macro Window. It contains buttons grouped by function.

Figure 12–8
The Macro Window Toolbar



The following table describes the functions of the toolbar buttons in the Macro Window.

Toolbar button	**Description**
New Procedure	Opens the Add Procedure dialog box that lets you name the procedure and paste procedure code into your macro file.
Start	Runs the active macro and opens the Debug Window.
Pause/Continue	Pauses and restarts a running macro. This button also pauses and restarts recording of SigmaPlot commands while using the Macro Recorder.
Stop	Terminates recording of SigmaPlot commands in the Macro Recorder. Also, stops a running macro.
Find	Opens the Find dialog where you can define a search for text strings in the Macro Window.
Step In	Executes the current line. If the current line is a subroutine or function call, execution will stop on the first line of that subroutine or call.
Step Over	Executes to the next line. If the current line is a subroutine or a function call, execution of that subroutine or function call will complete.
Step Out	Steps execution out of the current subroutine or function call.
Step to Cursor	Steps execution out to the current subroutine or function call.
Toggle Breakpoint	Toggles the breakpoint on the current line. The breakpoint stops program execution.
Quick View	Shows the value of the expression under the cursor in the Immediate Window.
Macros	Opens the Macros dialog box.
Dialog Box Editor	Opens the Dialog Box Editor.
Object Browser	Opens the Object Browser.
Reference	Opens the Reference dialog box which contains a list of all programs that are extensions of the SigmaPlot Basic language.

Color-Coded Display

The color-coding of text in the Macro Window indicates what type of code you are viewing.

The following table describes the default text colors used in the script text:

Text Color	**Description**
Blue	Identifies reserved words in Visual Basic (for example, Sub End Sub, and Dim).
Magenta	Identifies SigmaPlot macro commands and functions.
Green	Identifies comments in your macro code. Separates program documentation from the code as you read through your macros.

Object and Procedure Lists

The Object and Procedure lists show SigmaPlot objects and procedures for the current macro. These lists are useful when your macros become longer and more complex.

➤ The object identified as “(General)” groups all of the procedures that are not part of any specific object.

➤ The Procedure list shows all of the procedures for the currently selected object.

Setting Macro Window Options

You can set appearance options for the Macro window in the Macros tab of the Options dialog box.

To set the options of the Macro Window:

1. With a macro window open, on the Tools, click Options.

The Options dialog box appears.

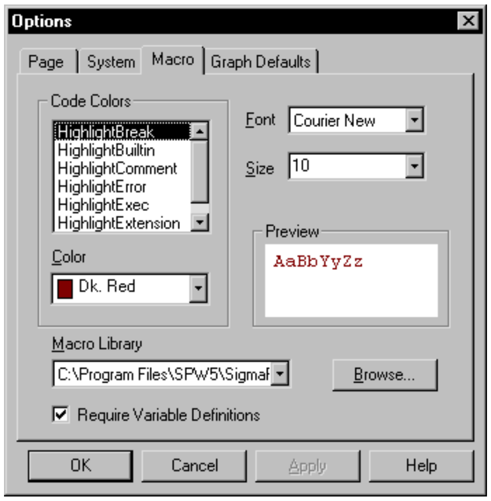


Figure 12–9
The Options Dialog Box
Macro Tab

2. Click the Macros tab.
3. Set text colors for different types of macro code and Debug Window output.
4. Change font characteristics.
5. Set the location for the macro library.

Parts of the Macro Programming Language

The following topics list the parts of the macro programming language:

➤ *Statements* are instructions to SigmaPlot to perform an action(s). Statements can consist of *keywords*, *operators*, *variables*, and *procedure* calls.

Keywords are terms that have special meaning in SigmaPlot. For example, the Sub and End Sub keywords mark the beginning and end of a macro. By default, keywords appears as blue text on color monitors. To find out more about a specific keyword in a macro, select the keyword and press F1. When you do this, a topic in the SigmaPlot on-line reference appears and presents information about the term.

➤ You can add optional comments to describe a macro command or function, and how it interacts in the script. When the macro is running, comment lines are ignored. Indicate a comment by beginning a line with an apostrophe. Comments always must end the line they're on. The next program line must go on a new line. By default, comment lines appear as green text.

Scrolling and Moving the Insertion Point

When you use the scroll bars the insertion point does not change. To edit the macro code that you are viewing in the macro window, you must move the insertion point manually.

To edit macro code manually:

1. In the Macro window, click where you want to edit.
2. You can also use arrows and key combinations to move the insertion point; when you do this the window scrolls automatically.

Editing Macro Code

You can edit macro code in the same way you edit text in most word-processing and text editing programs. You add select and delete text, type over code, or insert text by moving the insertion point and then typing in new text. As with other programming languages, you can also add comments to code.

To edit macro code:

➤ Open the macro code window and select the text to edit.

Adding Comments to Code

Adding comments to code is an excellent way to identify the purpose of the various parts of a macro and to map locations as you edit a complex macro. Comments can be inserted that fully document how to use and how to understand the macro code.

Deleting Unnecessary Code

The Macro Recorder creates code corresponding exactly to the actions that you make in SigmaPlot while the recorder was turned on. You may need to edit out unwanted steps.

Moving and Copying Code You can cut, copy, and paste selected text.

Finding and Replacing Code When you need to find and change text in a macro that you have written, use the Find commands. For example, if you change the name of a file that is referenced in your macro, you need to change every instance of the file name in your macro. Use Find to locate the instances of the filename in the macro and replace using cut and paste edit commands.

Adding Existing Macros to a Macro

If you have another macro that already does what you want, you can just paste it into your new macro. Copy and paste the macro into your new macro, test it in the new code and run it.

What Macro Recorder Records

The Macro Recorder does not record the following types of activity:

➤ Cursor movement

If you want to include this type of activity in your macro, you can use the IDE features.

About User-Defined Functions

A user-defined function is a combination of math expressions and Basic code. The function always requires input data values and always returns a value. You supply the function with a value; it performs calculations on the values and returns a new value as the answer. Functions can work with text, dates, and codes, not just numbers.

A user-defined function is similar to a macro but there are differences. Some of the differences are listed in the following table.

Recorded macros	**User-defined functions**
Performs a SigmaPlot action, such as creating a new chart. Macros change the state of the program.	Returns a value; cannot perform actions. Functions return answers based on input values.
Can be recorded.	Must be created in Macro code.
Are enclosed in the Sub and End Sub keywords.	Are enclosed in the keywords Function and End Function.

For More Information

The on-line help has an extensive section on user-defined functions. From anywhere in the Macro window, press F1, or choose Help Topics from the Help menu.

Creating User-Defined Functions

A user-defined function is like any of the built-in SigmaPlot function. Because you create the user-defined function, however, you have control over exactly what it does. A single user-defined function can replace database and spreadsheet data manipulation with a single program that you call from inside SigmaPlot. It is a lot easier to remember a single program than it is to remember several spreadsheet macros.

For a full explanation of User Defined Functions, see the Automation on-line reference Help file.

Using the Dialog Box Editor

The Dialog Box Editor lets you design and customize your own dialog boxes. When you are designing and creating SigmaPlot macros, you can automatically create the necessary dialog box code and dialog monitor function code with the Dialog Box

Editor. Like the other automated coding features in SigmaPlot, the code may require further customization.

To create a custom dialog box:

1. In the Macro Window, place the insertion point where you want to put the code for the dialog box.
2. On the Macro Window toolbar click the Dialog Box Editor button. A blank dialog grid appears.
3. Now you can select a tool, such as a button or check boxes, from the Toolbox. The cursor changes to a cross when you move it over the grid.
4. To place a tool on the dialog box, click a position on the grid. A default tool will be added to the dialog grid.
5. Resize the dialog box by dragging the handles on the sides and the corners.
6. Right-click any of the controls that you have placed on the dialog surface (after selecting the control) and enter a name for the control.
7. Right-click the dialog form (with no control selected) and enter a name for the dialog monitor function in the DialogFunc field.
8. To finish, click OK. The code for the dialog box with controls will be written to the Macro Window.
9. Finally, and in most cases, you must edit the code for dialog box monitor function to define the specific behavior of the elements in your dialog box.

For more information, see the Automation on-line help reference.

Using the Object Browser

The Object Browser displays all SigmaPlot object classes. The methods and properties associated with each SigmaPlot macro object class are listed. A short description of each object appears in the dialog box as you select them from the list. By clicking F1, you can access extensive Help that includes example code for the individual properties and methods. The Paste feature lets you insert generic code based on your selection into a macro.

The Object Browser will be familiar and useful if you are comfortable with object oriented programming. If you are not, consult one of the excellent introductory texts on Visual Basic.

For full details on using the Object Browser, press F1 from anywhere in the Macro window.

Using the Add Procedure Dialog Box

Organizing your code in procedures makes it easier to manage and reuse. SigmaPlot macros, like Visual Basic programs, must have at least one procedure (the main subroutine) and often they have several. The main procedure may contain only a few statements, aside from calling subroutines that do the work.

SigmaPlot provides a dialog box that generates procedure code for your macros.

Using the Add Procedure Dialog Box

By using the Add Procedure dialog box, you can define a sub, function, or property using the Name, Type, and Scope boxes. Clicking OK pastes the code for a new procedure into your macro at the insertion point.

For full details on using the Add Procedure, press F1 from anywhere in the Macro window.

Using the Debug Window

The Debug Window contains a group of features that are helpful when you are trying to locate and resolve errors in your macro code. The debugging tools in SigmaPlot will be familiar if you have used one of the modern visual programming languages or Microsoft Visual Basic for Applications. Essentially, the Debug Window gives you incremental control over the execution of your program so that you can sleuth errors in your programs. The Debug Window also gives you a precise way to determine the contents of your variables. Again, a series of buttons is used to select the operation mode of the Debug Window.

Debug Toolbar Buttons

The debugging features of the Debug Window are controlled by buttons on the Macro Window toolbar. To review:

- ➤ The four Step buttons provide methods for controlling the execution of commands. They offer various ways of responding to subroutines and functions.
- ➤ The Breakpoint button lets you set a point and execute the program until it reaches that point.
- ➤ The Quick View button displays the value of the expression in the immediate window.

The inclusion of these features for controlling program execution are a standard but powerful combination of tools for writing and editing macros.

Debug Window Tabs

The output from the Debug Window is organized in four tabs that allow you to type in statements, observe program execution responses, and iteratively modify your code using this feedback. If you have never used a debugging tool and are new to programming, it would be a good idea to supplement the following description with further study.

Immediate Tab

The Immediate Tab lets you evaluate an expression, assign a specific value to a variable or call a subroutine and evaluate the results. Trace mode prints the code in the tab when the macro is running.

- ➤ Type "?expr" and press Enter to show the value of "expr".
- ➤ Type "var = expr" and press Enter to change the value of "var".
- ➤ Type "set var = expr" and press Enter to change the reference of "var".
- ➤ Type "subname args" and press Enter to call a subroutine or built-in expression “subname” with arguments “args”.
- ➤ Type "trace" and press Enter to toggle trace mode. Trace mode prints each statement in the Immediate Tab when a macro is running.

Watch Tab

The Watch Tab lists variables, functions, and expressions that are calculated during execution of the program.

- ➤ Each time program execution pauses, the value of each line in the window is updated.
- ➤ The expression to the left of the "->" may be edited.
- ➤ Pressing Enter updates all the values immediately.
- ➤ Pressing Ctrl+Y deletes the line.

Stack Tab

The output from the Stack Tab lists the program lines that called the current statement. This is a macro command audit and is helpful to determine the order of statements in you program.

- ➤ The first line is the current statement. The second line is the one that called the first, and so on.
- ➤ Clicking a line brings that macro into a sheet and highlights the line in the edit window.

13 SigmaPlot Automation Reference

OLE Automation is a technology that lets other applications, development tools, and macro languages use a program. SigmaPlot Automation allows you to integrate SigmaPlot with the applications you have developed. It also provides an effective tool to customize or automate frequent tasks you want to perform.

Automation uses objects to manipulate a program. Objects are the fundamental building block of macros; nearly all macro programs involve modifying objects. Every item in SigmaPlot-graphs, worksheets, axes, tick marks, reports, notebooks, etc.-can be represented by an object.

SigmaPlot uses a VBAÆ-like macro language to access automation internally. For more information on recording and editing SigmaPlot macros, see About Macros.

This chapter contains the following topics:

➤ "Opening SigmaPlot from Microsoft Word or Excel" on page 255
➤ "SigmaPlot Objects and Collections" on page 256
➤ "SigmaPlot Properties" on page 266
➤ "SigmaPlot Methods" on page 275

Opening SigmaPlot from Microsoft Word or Excel

To open SigmaPlot from Microsoft Word or Excel, you must first create a macro from within either application.

To create the macro:

1. In Excel or Word, choose Tools/Macro/Visual Basic Editor.

   Visual Basic appears.

2. Choose Insert/Module.

3. Type:

```
Sub SigmaPlot()
'
```

```
' SigmaPlot Macro
'
'
    Dim SPApp as Object
    Set SPApp = CreateObject("SigmaPlot.Application.1")
    SPApp.Visible = True

    SPApp.Application.Notebooks.Add
End Sub
```

4. Choose Run/Run Sub/User Form to run the macro.

   SigmaPlot appears with an empty worksheet and notebook window.

To open SigmaPlot from Word or Excel in the future:

1. Choose Tools/Macro/Macros to open the Macros dialog box.
2. Select SigmaPlot.
3. Click Run.

SigmaPlot Objects and Collections

An *object* represents any type of identifiable item in SigmaPlot. Graphs, axes, notebooks, worksheets, and worksheet columns are all objects.

A *collection* is an object that contains several other objects, usually of the same type; for example, all the items in a notebook are contained in a single collection object. Collections can have methods and properties that affect the all objects in the collection.

Properties and *methods* are used to modify objects and collections of objects. To specify the properties and methods for an object that is part of a collection, you need to return that individual object from the collection first.

For more information, refer to SigmaPlot Automation Help from the SigmaPlot Help menu.

Application Object

An Application object represents the SigmaPlot application, within which all other objects are found. (Most other objects must exist inside higher-level objects. You access objects by applying properties and methods on these higher-level objects.) It is a "user-creatable" object, that is, outside programs can run SigmaPlot and access its properties directly, and will be registered in registry as SPW32.Application.

The Application object properties and methods return or manipulate attributes of the SigmaPlot application and main window, and access the list of notebooks and from there all other objects.

To use the Application Object

Use Application properties to return attributes of the SigmaPlot application. Note that when using the SigmaPlot macro window, all Application methods and properties are global, that is, you do not need to specify the Application object.

Notebooks Collection Object

The Notebooks collection represents the list of open notebooks in SigmaPlot. Use this collection to create new documents and open existing documents, as well as to specify and return individual notebooks as objects.

To use the NotebooksCollection

A Notebooks collection is returned using the Application object Notebooks property. Use the Add method to add a new notebook to the collection. You can return a specific Notebook object using either the Item property or the collection index

Notebook Object

Represents a SigmaPlot notebook file (including template and equation library files). Notebook properties and methods are used to set individual notebook file attributes and specify individual notebook items, e.g., worksheets, graph pages, reports, etc. Also used to return a collection of notebook items as objects.

To use the Notebook Object

Notebook objects are returned using the Notebooks or ActiveDocument Application object properties. Access individual notebook items using the NotebookItems property, which returns the NotebookItems collection.

NotebookItems Collection Object

This collection represents all the items in a notebook, and is used to create new items and open existing items. Also used to specify and return the different notebook items as objects. Worksheets, pages, equations, reports, macros, and section and notebook folders are all notebook items and can be returned as objects.

To use the NotebookItems Collection

The NotebookItems collection is returned using the NotebookItems property of a Notebook object. You can return individual notebook item objects using either the Item method or collection index, and add new notebook item objects, such as worksheets and graph pages, using the Add method.

NativeWorksheetItem Object

This object represents the SigmaPlot data worksheet. Use this object to perform worksheet edit operations, and to access the data using the DataTable property.

To use the NativeWorksheetItem Object

The NativeWorksheetItem object has the standard notebook item properties and methods. A NativeWorksheetItem is returned from the NotebookItems collection using the Item property or collection index, and created using the NotebookItems Add method. The NativeWorksheetItem object has an ItemType property and NotebookItems.Add method value of 1.

ExcelItem Object

Use this object to manipulate in-place activated Excel worksheets. In general, most NativeWorksheetItem properties and methods also apply to Excel worksheets.

To use the ExcelItem Object

The ExcelItem object also has the standard notebook item properties and methods. An ExcelItem is returned from the NotebookItems collection using the Item property or collection index, and created using the NotebookItems Add method. The ExcelItem object has an ItemType property and NotebookItems.Add method value of 8.

Σ Note that you can only create an ExcelItem if you have Excel for Office 95 or 97 installed.

DataTable Object

Represents a table of data as used by a worksheet or graph page. This object's properties and methods can be used to access the data in a worksheet or page, and also return the NamedRanges (row and column titles) collection object.

To use the DataTable Object

The DataTable objects is returned from NativeWorksheetItem, ExcelItem, and GraphItem objects using the DataTable method, and in turn accesses data using the GetData and PutData methods and the Cell property.

NamedDataRanges Collection Object

The NamedDataRanges collection contains all ranges in the DataTable object that have been assigned a name. Column and row titles are name ranges.

To use the NamedDataRanges Collection

The NamedDataRanges collection is mainly used to retrieve existing ranges and add new ranges to DataTable objects within NativeWorksheetItem and GraphItem objects.

NamedDataRange Object

Represents named data range objects (e.g. column and row titles) in the worksheet and page data tables.

To use the NamedDataRange Object

The NamedDataRange object is returned from the NamedRanges collection using an index or the Item property. The NamedRange object properties are mainly the range name, dimensions and other similar attributes.

GraphItem Object

The GraphItem object represents a SigmaPlot graph page. GraphItem properties can be used to return a collection of the graphs on the page using the GraphObjects property. It can also be used to create graphs using the CreateWizardGraph method.

To use the GraphItem Object

The GraphItem object has the standard notebook item properties and methods. A GraphItem is returned from the NotebookItems collection using the Item property or collection index, and created using the NotebookItems Add method. The GraphItem object has an ItemType property and NotebookItems Add method value of 2.

Pages Collection/ Page Object

The Page object represents a SigmaPlot graph page. Graph pages can be different sizes and colors, and a page object can be used to return the collections of objects on that page. Pages have an ObjectType value of 1 or GPT_PAGE.

To use the Page Object

A graph page is returned from a GraphItem object using the GraphPages property. Note that since there is currently only one page per graph item, you can always use .GraphPages(0) to return a page. Use the ChildObjects property to return the GraphObjects collection, or the Graphs property. Note that if a graph is part of a Group object, it can only be returned using the Graph property.

Many Page attributes and attribute values can only be returned or set using the GetAttribute and SetAttribute methods. Use the Page Attribute constants to specify these attributes.

Page GraphObjects Collection

The Page GraphObjects Collection represents a collection of the child objects returned from a Page object.

To use the Page GraphObjects Collection

A Page GraphObjects collection is returned from a Page object using the ChildObjects property. You can also return Page GraphObjects collections composed only of Graph objects using the Graphs property.

Graph Object

The Graph object represents a SigmaPlot graph. A graph is used to access the parts of a graph, e.g., plots, axes, etc., as well as to change graph attributes such as title, size, and position. Graph objects have an ObjectType value of 2 or GPT_GRAPH.

To use the Graph Object

A Graph object is returned from a Page GraphObjects collection. Note that you can create a GraphObjects collection composed only of the graphs using the Page object Graphs property. The ChildObjects or Plots properties are used to return the graph object's Plots collection. Other properties, such as Axes and AutoLegend, are used to return other graph objects.

Many Graph attributes and attribute values can only be returned or set using the GetAttribute and SetAttribute methods. Use the Graph Attribute constants to specify these attributes.

Plot GraphObjects Collection

Represents a collection of the child objects returned from a Graph object.

To use the Plot GraphObjects Collection

A Plot GraphObjects collection is returned from a Graph object using either the ChildObjects or Plots property. Use the Plots collection to return specific Plot objects.

Plot Object

The Plot object represents a data plot and all its attributes and child objects. Plots have an ObjectType value of 3 or GPT_PLOT.

To use the Plot Object

Plot properties and methods are mainly used to return the Plot child objects. Return specific Plot child objects using different Plot properties:

- ➤ Line returns the Line plot object
- ➤ Symbols returns the Symbol plot object
- ➤ Fill returns the Solid plot object (e.g. bars)
- ➤ DropLines returns a collection of the drop line Line objects
- ➤ Functions returns collection of the Function objects (e.g. regression and reference lines)

Use .ChildObjects(0) to return the Tuple GraphObjects collection. Tuples represent the individual plot curves.

Many Plot attributes and attribute values can only be returned or set using the GetAttribute and SetAttribute methods. Use the Plot Attribute constants to specify these attributes.

Axes GraphObjects Collection

The Axes collection corresponds to all the sets of axes available for a graph.

To use the Axes Collection

Use the index to return specific x, y or z axes from an Axes collection.

Axis Object

The Axis objects represents a SigmaPlot axis. Axes have an ObjectType value of 4 or GPT_AXIS.

To use the Axis Object

An axis has several Line and Text objects associated with it, including the axis line itself, grid lines, tick marks and labels, and the axis title. Use the LineAttributes property to return the collection of axis lines, and the AxisTitles and TickLabelAttributes property to return collection of axis text objects.

Most other Axis attributes, such as range, scale, breaks, etc., can only be returned or set using the GetAttribute and SetAttribute methods. Use the Axis Attribute constants to specify these attributes.

Text Object

All characters and labels found on a SigmaPlot change correspond to a text object and can be modified using text object properties and methods. Axes have an ObjectType value of 5 or GPT_TEXT.

To use the Text Object

The Text object for most graph objects can be returned using the NameObject property.

Text objects are also found below both AutoLegend and Axis objects. Use the TickLabelAttributes property to access the tick label Text object. Use the AxisTitles property to access the axis titles.

To access the text objects within a Page or an AutoLegend, use the ChildObjects property.

Use the Name property to change the string used for the text. Most other Text attributes and attribute values can only be returned or set using the GetAttribute and SetAttribute methods. Use the Text Attribute constants to specify these attributes.

Line Object

Objects that correspond to drawn lines. These lines include all lines used for axes and plots, regression and reference lines, drop lines, and manually drawn lines. Lines have an ObjectType value of 6 or GPT_LINE.

To use the Lines Object

Lines are returned from a number of different objects:

Drawn lines on a page are returned using the ChildObjects property. Lines are creating using the Page object Add method with an object value of 6 or GPT_LINE.

Plot lines are returned using the Line property. Collections of Lines can also be returned using the DropLines property.

To return the collection of axis lines, use the Axis object LineAttributes property. These include the axis lines, tick marks, grid lines, and axis break lines.

The collection of function lines is returned using the Plot object Functions property. Functions include linear regression and reference lines.

Many Line attributes and attribute values can only be returned or set using the GetAttribute and SetAttribute methods. Use the Line Attribute constants to specify these attributes.

Symbol Object

The symbol object controls the symbols used in a plot. Symbols have an ObjectType value of & or GPT_SYMBOL.

To use the Symbols Object

The Symbols object is returned from a Plot object using the Symbols property. The Symbols properties and methods are used to return or modify the individual symbols within the parent plot.

Many Symbol attributes and attribute values can only be returned or set using the GetAttribute and SetAttribute methods. Use the Symbol Attribute constants to specify these attributes.

Solid Object

A solid object can represent many different graph and page objects. All drawn shapes-ellipses and rectangles, as well as graph bars and boxes, pie slices, meshes, and any other "filled" object. Solids have an ObjectType value of 8 or GPT_SOLID.

To use the Solid Object

Solid objects can be returned from a number of other groups or collections using different properties:

To access the solid object(s) for a Plot, use the Fill property.

To access the solid object(s) within a Page or an AutoLegend, use the ChildObjects property.

Many Solid attributes and attribute values can only be returned or set using the GetAttribute and SetAttribute methods. Use the Solid Attribute constants to specify these attributes.

Tuple GraphObjects Collection

Represents the collection of tuples for a Plot object. A tuple is an individual curve or series representing a plotted column or column set. For example, an XY pair plotted as a scatter plot, a single column plotted as a bar series, or a column plotted as a mean are all tuples.

To use the Tuples GraphObjects Collection

The Tuples collection is returned from a Plot using the ChildObjects property. Use the Tuple collection return specific tuples or add new tuples.

Tuple Object

A tuple is an object that represents a plotted column or column pair, displayed as a curve, datapoint, or bar series. A plot always consists of a collection of one or more tuples. Tuples have an ObjectType value of 9 or GPT_TUPLE.

To use the Tuple Object

Tuples are returned from the Tuples GraphObjects collection. Most generic GraphObject properties are not retained by tuples; instead, use the SetAttribute and GetAttribute methods to return or change the tuple properties. Use the Tuple Attribute constants to specify these attributes.

Function GraphObjects Collection

The Functions collection consists of all regression, confidence, prediction, and reference lines for a plot.

To use the Functions Collection

The functions collection is basically used to return individual function objects. Return the Functions Collection from a Plot object using the Functions property.

Function Object

A Function object represents one of the various function lines of a Plot object. In addition to Line object properties, functions also have properties and attributes specific to regression and reference lines. Functions have an ObjectType value of 10 or GPT_FUNCTION.

To use the Function Object

The Function object is returned from the Functions collection as follows:

Index	Constant	Function
1	SLA_FUNC_REGR	Regression Line
2	SLA_FUNC_CONF1	Upper Confidence Intervals
3	SLA_FUNC_CONF2	Lower Confidence Interval
4	SLA_FUNC_PRED1	Upper Prediction Interval
5	SLA_FUNC_PRED2	Lower Prediction Interval
6	SLA_FUNC_QC1	1st Reference Line (Upper Specification)
7	SLA_FUNC_QC2	2nd Reference Line (Upper Control Line)
8	SLA_FUNC_QC3	3rd Reference Line (Mean)
9	SLA_FUNC_QC4	4th Reference Line (Lower Control Line)
10	SLA_FUNC_QC5	5th Reference Line (Lower Specification)

Most Function attributes and attribute values can only be returned or set using the GetAttribute and SetAttribute methods. Use the Function Attribute constants to specify these attributes. You can return the label for reference lines using the NameObject property, but only if the label is turned on first.

Note that Function lines are turned on or off with Plot object attributes.

DropLines Collection

The DropLines object is a special collection of lines that represent the drop lines for a plot. The DropLines object is returned using the Plot object DropLines property. There are three different sets of drop lines that can be retrieved from the DropLines collection:

DropLine property indexes:

1. xy plane (SLA_FLAG_DROPZ , 3D graphs only)
2. Y axis/x direction or yz plane (SLA_FLAG_DROPX)
3. X axis/y direction or zx plane (SLA_FLAG_DROPY)

Note that drop lines are turned on and off using the Plot object SetAttribute method, using the SLA_PLOTOPTIONS property coupled with the SLA_FLAG_DROPX, SLA_FLAG_DROPY, or SLA_FLAG_DROPZ value, and using the FLAG_SET_BIT to turn on drop lines, or the FLAG_CLEAR_BIT to turn off drop lines. Other drop line properties are set using Line object attributes.

Group Object

A group is any grouped collection of objects, generally created with the Format menu Group command. Grouped objects can be treated as a single object. AutoLegends are a special class of Group object. Groups have an ObjectType value of 12 or GPT_BAG.

Many Group attributes and attribute values can only be returned or set using the GetAttribute and SetAttribute methods. Use the Group (Bag) Attribute constants to specify these attributes.

AutoLegend Object

The AutoLegend object is really a specific Group object consisting of a solid object and text objects. The AutoLegend object is returned from a graph using the AutoLegend property.

To use the AutoLegend Object

Manipulate the AutoLegend object as you would a Group object. Use the ChildObjects property to return the members of the AutoLegend. .ChildObjects(0) always returns the Solid rectangle object used as the AutoLegend border/background, and indexes >0 to return the individual Text objects used for the legend keys.

Legends can also be manipulated with many Text attributes using the GetAttribute and SetAttribute methods.

GraphObject Object

The GraphObject object corresponds to non-SigmaPlot objects residing on a graph page, such as pasted bitmaps or metafiles, or embedded or linked OLE objects.

FitItem Object

The FitIItem object corresponds to the a SigmaPlot equation and all the equation code parameters and settings. FitItems are used not only for regressions, but for other nonlinear curve fitting applications, and function plotting and solving. The results of a FitItem are accessed from a FitResults object.

To use the FitItem Object

The FitItem object has the standard notebook item properties and methods. A FitItem is returned from the NotebookItems collection using the Item property or collection index, and created using the NotebookItems Add method. The FittItem object has an ItemType property and NotebookItems.Add method value of 6.

The complete list of FitItem properties and methods also can be found in FitItem and FitResults Properties and Methods.

FitResults Object

The FitResults object is used to return the different values computed by the nonlinear regression. These statistics and other results are specifically useful for computing additional statistics that can be derived from these results.

The complete list of FitResult properties can also be found in FitItem and FitResults Properties and Methods.

TransformItem Object

Represents either an open transform or opened transform file as an object. You can load transforms, specify the transform code, and replace variables before executing a transform.

To use the TransformItem Object

The TransformItem object has the standard notebook item properties and methods; however transforms cannot be currently saved as notebook objects, only created and opened. If you want to save a transform to a .xfm file, use the Name property to specify a file name and path before using the Save method.

When using a TransfomItem object, you must first declare a variable as an object and then define it as a newly added transform item. Create a new TransformItem collection using the NotebookItems Add method, using a value of 9. After defining the transform object, open it using the Open method.

Specify the transform code using the Text property. To change the value of a transform variable, use the AddVariableExpression method. Run transforms using the Execute method.

Note: After executing a transform, it is a good idea to close it, as you are limited to four concurrent transforms that can be open simultaneously.

ReportItem Object

A ReportItem represents the RTF (Rich Text Format) documents used for text and regression reports in SigmaPlot. You can use the ReportItem properties to add and remove block of text from a report.

To use the ReportItem Object

The ReportItem object has the standard notebook item properties and methods. A ReportItem is returned from the NotebookItems collection using the Item property or collection index and created using the NotebookItems Add method. SigmaPlot reports have an ItemType property and NotebookItems.Add method value of value of 5, and SigmaStat reports have a value of 4.

MacroItem Object

Represents a SigmaPlot macro. You can use this command to edit and run macros from within macros, or to run macros from outside applications.

To use the MacroItem Object

The MacroItem object has the standard notebook item properties and methods. A MacroItem is returned from the NotebookItems collection using the Item property or collection index, and created using the NotebookItems Add method. The MacroItem object has an ItemType property and NotebookItems.Add method value of 0.

NotebookItem Object

Represents the notebook item in the notebook window. You can use this object to rename the notebook item. The notebook item can always be reference with NotebookItems(0).

To use the NotebookItem Object

The NotebookItem object has most of the standard notebook item properties and methods, and created using the NotebookItems Add method. The NotebookItem object has an ItemType property and NotebookItems.Add method value of 7.

SectionItem Object

Represents the section folders within a SigmaPlot notebook.

To use the SectionItem Object

The SectionItem object most of the standard notebook item properties and methods. A SectionItem is returned from the NotebookItems collection using the Item property or collection index, and created using the NotebookItems Add method. The SectionItem object has an ItemType property and NotebookItems.Add method value of 3.

SigmaPlot Properties

A *property* is a setting or other attribute of an objectóthink of a property as an "adjective." For example, properties of a graph include the size, location, type and style of plot, and the data that is plotted. To change the settings of an object, you change the properties settings. Properties are also used to access the objects that are below the current object in the hierarchy.

To change a property setting, type the object reference followed with a period, then type the property name, an equal sign (=), and the property value.

For more information, refer to SigmaPlot Automation Help from the SigmaPlot Help menu.

Application Property

Used without an object qualifier, this property returns an Application object that represents the SigmaPlot application. Used with an object qualifier, this property returns an Application object that represents the creator of the specified object (you can use this property with an Automation object to return that object's application).

Σ Use the CreateObject and GetObject functions give you access to an Automation object.

Author Property

A standard property of notebook files and all NotebookItems objects. Returns or sets the Author field in the Summary Information for all notebook items, or the Author field under the Summary tab of the Windows 95/98 file Properties dialog box.

Autolegend Property

Returns the Autolegend Group object for the specified Graph object. Autolegends have all standard group properties. The first ChildObject of a legends is always a solid; the successive objects are text objects with legend symbols.

Axis Property

The Axes property is used to return the collection of Axis objects for the specified graph object. Individual axis objects have a number of line and text objects that are returned with Axis object properties.

Axistitles Property

The AxisTitle property is used to return the collection of axis title Text objects for the specified Axis. Use the following index values to return the different titles. Note the specific title returned depends on the current axis dimension/direction selected.

0	Bottom/Left axis title
1	Right/Top axis title
2	Sub axis title (not currently shown)
3	Sub axis title (not currently shown)

Cell Property

Returns or sets the value of a cell with the specified column and row coordinates for the current DataTable object.

ChildObjects Property

Used by all page objects that contain different sub-objects to return the collection of those objects. The ChildObjects property returns different type of objects depending on the object type:

Object Returns	ChildObjects
Page	Page GraphObjects
Graph	Plots
Plot	Tuples
Tuples	Tuple
Group (including Autolegends)	all group objects

Color Property

Gets or sets the color for all drawn page objects. Use the different color constants for the standard VGA color set. For more information, refer to SigmaPlot Automation from the SigmaPlot Help menu.

Comments Property

Syntax: Notebook/NotebookItems object.Comments

A standard property of notebook files and all NotebookItems objects. Returns or sets the Description field in the Summary Information for all notebook items, or the Comments section under the Summary tab of the Windows 95/98 file Properties dialog box for notebook files.

Count Property

A property available to all collection objects that returns the number of objects within that collection.

CurrentDataItem Property

The CurrentDataItem property returns the worksheet window in focus as an object. You must still use the ActiveDocument property to specify the currently active notebook.

Note that if a worksheet is not in focus an error is returned.

CurrentItem Property

This property returns whatever notebook item currently has focus as an object. You must still use the ActiveDocument property to specify the currently active notebook.

CurrentPageItem Property

Returns the current graph page window as a GraphItem object. You must still use the ActiveDocument property to specify the currently active notebook.

If the current item in focus is not a page, an error is returned.

DataTable Property

Returns the DataTable object for the specified worksheet object.

DefaultPath Property

Sets or returns the default path used by the Application object to save and retrieve files. Files are opened using the Notebooks collection Open method and saved using the Notebook object Save or SaveAs methods.

DropLines Property

Returns the DropLines line collection for a Plot object. Line objects within the DropLines collection have standard line properties.

Use an index to return a specific set of drop lines from the DropLines collection:

1. XYplane (SLA_FLAG_DROPZ , 3D graphs only)
2. Y axis/X direction or YZ plane (SLA_FLAG_DROPX)
3. X axis/Y direction or ZX plane (SLA_FLAG_DROPY)

Some drop line properties are controlled from the Plot object; for example, use the SetAttribute(SLA_PLOTOPTIONS,SLA_FLAG_DROPX Or FLAG_SET_BIT) plot object method to turn on y axis drop lines. Other drop line properties are set using Line object attributes.

Expanded Property

A property of notebook window notebooks and sections, which opens or closes the tree for that notebook section, or returns a true or false value for the current view.

Fill Property

The Fill property is used to return the Solid object for the specified Plot object. Solid objects for plots include bars and boxes.

FullName Property

Returns the filename and path for either the application or the current notebook object. If the notebook object has not yet been saved to a file, an empty string is returned.

Functions Property

The Functions property is used to return the collection of Function objects for the specified Plot object. Plot functions include regression and confidence lines, and all reference (QC) lines. The individual function lines are specified using an index:

Index	Constant	Function
1	SLA_FUNC_REGR	Regression Line
2	SLA_FUNC_CONF1	Upper Confidence Intervals
3	SLA_FUNC_CONF2	Lower Confidence Interval
4	SLA_FUNC_PRED1	Upper Prediction Interval
5	SLA_FUNC_PRED2	Lower Prediction Interval
6	SLA_FUNC_QC1	1st Reference Line (Upper Specification)
7	SLA_FUNC_QC2	2nd Reference Line (Upper Control Line)
8	SLA_FUNC_QC3	3rd Reference Line (Mean)
9	SLA_FUNC_QC4	4th Reference Line (Lower Control Line)
10	SLA_FUNC_QC5	5th Reference Line (Lower Specification)

Note that most regression and reference lines options are controlled with different plot and line attributes. For example, to turn on a regression line, use SetAttribute(SLA_REGROPTIONS,SLA_REGR_FORPLOT Or FLAG_SET_BIT), and to turn on the third reference line, use SetAttribute(SLA_QCOPTIONS,SLA_QCOPTS_SHOWQC3 Or FLAG_SET_BIT)

Graphs Property

Returns the collection of graphs for the specified Page object. Use the index to select a specific Graph object. Graphs are used in turn to return the different graph items: Plots, Axes, the graph title, and the graph legend.

GraphPages Property

Returns the GraphPages collection of Page objects for a GraphItem object. However, since there is currently only one graph page for any given graph item, you can always use GraphPages(0). However, in order to access items within a GraphItem, you must always specify the GraphPage.

Height Property Sets or returns the height of the application window or specified notebook document window in pixels, or the size of pages and page objects in 1000ths of an inch.

InsertionMode Property Sets or returns a Boolean indicating whether or not Insert mode is on.

Interactive Property Sets or returns a Boolean indicating whether or not the user is allowed to interact with the running notebook window or application. Do not set the Application property to False from within SigmaPlot or you will lose access to the application.

IsCurrentBrowser Entry Property Returns whether or not the specified item is the currently selected item in the notebook tree. This is particularly useful when adding new objects to a notebook in s specific notebook location.

IsCurrentItem Property Returns whether or not the specified item is the currently selected item. This property is particularly useful when used in conjunction with the CurrentItem property.

IsOpen Property A property common to all NotebookItems objects. Returns a Boolean indicating whether or not the specified document or section is open. Open and close notebook items using the Open and Close methods.

ItemType Property A property common to all NotebookItems objects. Returns an integer denoting the item/object type.

1	CT_WORKSHEET	NativeWorksheetItem
2	CT_GRAPHICPAGE	GraphItem
3	CT_FOLDER	SectionItem
4	CT_STATTEST	ReportItem (SigmaStat)
5	CT_REPORT	ReportItem (SigmaPlot)
6	CT_FIT	FitItem
7	CT_NOTEBOOK	NotebookItem
8	CT_EXCELWORKSHEET	ExcelItem
9	CT_TRANSFORM	TransformItem
10		MacroItem

Keywords Property A standard property of notebook files and all NotebookItems objects. Sets the Keywords field under the Summary tab of the Windows 95/98 file Properties dialog box.

Note that the keywords for notebook items are not currently displayed or used. The default keywords used by SigmaPlot notebooks are "SigmaPlot" and "SigmaStat."

Left Property Sets or returns the left coordinate of the application window or specified notebook document window in pixels, or the size of pages and page objects in 1000ths of an inch.

Line Property Returns the Line object for the specified Plot object. Lines are available in both line plots and line and scatter plots.

LineAttributes Property Returns the collection of axis Line objects for the specified Axis object. Use the collection index to return a specific line object:

Index	Line
1	Axis Lines
2	Major Ticks
3	Minor Ticks
4	Major Grid
5	Minor Grid
6	Axis Break

Note that many axis line attributes are set with the different Axis object attributes, using the Axis object SetAttribute method.

Name Property A standard property of almost all SigmaPlot objects. Returns or sets the Title name and field in the Summary Information for all notebook items, the filename for a notebook file, and the object name or title for page objects.

To set the title used for a notebook, use the Notebook object Title property, or set the name for NotebookItems(0).

Note: If you attempt to set the name of a document to the existing name, you will receive an error message and the macro will halt.

NamedRanges Property Returns the collection of NamedDataRanges from a DataTable object. Use the NamedDataRanges collection to return a specific NamedDataRange object.

NameObject Property Returns the Text object that corresponds to the name of the specified object.

NameOfRange Property

Sets or returns the name for a NamedDataRange object. Useful for returning lists of column and row titles, which are named ranges.

NotebookItems Property

A Notebook object property that returns the collection of notebook items. Use the NotebookItems collection to access individual notebook items. Worksheets, pages, equations, reports, macros, and section and notebook folders are all notebook items and can be returned as objects.

Notebooks Property

An Application object property that returns the Notebooks collection object. Use the Notebooks collection to return individual Notebook objects and create new notebooks.

NumberFormat Property

Sets or returns the format used by the currently selected cells in the DataTable of the NativeWorksheetItem or ExcelItem object. If there is no selection, the format for the entire worksheet is assumed. If there are mixed formats, a NULL value is returned.

Both Number and Date and Time formats are set or returned using the standard number and date and time format designations.

ObjectType Property

Returns the type value for the specified object. The values returned and corresponding object types are:

Value	Constant	Object
1	GPT_PAGE	Page
2	GPT_GRAPH	Graph
3	GPT_PLOT	Plot
4	GPT_AXIS	Axis
5	GPT_TEXT	Text
6	GPT_LINE	Line
7	GPT_SYMBOL	Symbol
8	GPT_SOLID	Solid
9	GPT_TUPLE	Tuple
10	GPT_FUNCTION	Function
11	GPT_EXTERNAL	External
12	GPT_BAG	Group
13	GPT_DATATABLE	DataTable

OwnerGraphObject Property

Returns the object that the current object is contained within. This applies to the different graph page object hierarchies, where the Parent property is not supported.

Parent Property

Returns the object or collection immediately "above" the current object. For graph page items, use the OwnerGraphObject property instead.

Path Property

Returns the default path in which SigmaPlot looks for documents, or the path of the specified notebook file.

For notebooks, you can use the Name property to return the file name without the path, or use the FullName property to return the file name and the path together.

Plots Property

Returns the collection of plots for the specified Graph object. Use an index to return the individual Plot objects for the graph.

Saved Property

Returns a True or False value for whether of not the document has been saved since the last changes. Note that notebook items that are closed from within SigmaPlot are automatically saved to the notebook, but that the notebook file is only saved using a Save or Save As command or method.

SelectedText Property

Returns the text of the current selection from a ReportItem. You can set or return a text selection using the SelectionExtent property.

SelectionExtent Property

Returns the array of current selection extents from a ReportItem or ExcelItem. The start and stop indices for each selection are listed as individual members of the array, e.g., .SelectionExtent(0) is the start of the first selection, and SelectionExtent(1) is the end of the first selection.

ShowStatsWorksheet Property

If this Boolean property is set to “True,” SigmaPlot opens up a statistics window that displays statistics about the specified NativeWorksheetItem. Statistics include: mean, standard deviation, standard error, half-widths for 95% and 99% confidence intervals, sample size, total, minimum, maximum, smallest positive value, and number of missing values. If this property is set to “False,” the statistics window is closed if open.

This property returns “True” if the statistics worksheet window is open or “False” if the worksheet window is not open or the specified NativeWorksheet is not open.

If the specified NativeWorksheet object is not open, setting this property has no effect.

StatsWorksheetDataTable Property

Returns the Column Statistics worksheet as a DataTable object.

Returns an object expression representing the read-only data table belonging to the NativeWorksheetItemís statistics worksheet. If the worksheet has not been opened using the ShowStatsWorksheet property, this property returns nothing.

StatusBar Property

Sets or returns the SigmaPlot application window status bar text. Note that when a macro is running within SigmaPlot, it will also issue status messages that will overwrite messages set with the StatusBar property. A macro running in VB or VBA outside SigmaPlot will not create its own status bar messages other than those set with StatusBar.

StockScheme Property

Returns the property scheme value for a variable, which can then be assigned to a graph object.

STOCKSCHEME_COLOR_BW	&H00010001
STOCKSCHEME_COLOR_GRAYS	&H00020001
STOCKSCHEME_COLOR_EARTH	&H00030001
STOCKSCHEME_COLOR_FOREST	&H00040001
STOCKSCHEME_COLOR_OCEAN	&H00050001
STOCKSCHEME_COLOR_RAINBOW	&H00060001
STOCKSCHEME_COLOR_OLDINCREMENT	&H00070001

STOCKSCHEME_SYMBOL_DOUBLE	&H00010002
STOCKSCHEME_SYMBOL_MONOCHROME	&H00020002
STOCKSCHEME_SYMBOL_DOTTEDDOUBLE	&H00030002
STOCKSCHEME_SYMBOL_OLDINCREMENT	&H00040002

STOCKSCHEME_LINE_MONOCHROME	&H00010003
STOCKSCHEME_LINE_OLDINCREMENT	&H00020003

STOCKSCHEME_PATTERN_MONOCHROME	&H00010004
STOCKSCHEME_PATTERN_OLDINCREMENT	&H00020004

Subject Property

A standard property of notebook files and all NotebookItems objects. Sets the Subject field under the Summary tab of the Windows 95/98 file Properties dialog box.

Note that the Subject for notebook items is not currently displayed or used.

Symbols Property

Returns the Symbol object for the specified Plot object.

Template Property Returns the Notebook object used as the template source file. The template is used for new page creation. To create a graph page using a template file, use the ApplyPageTemplate method.

Text Property Specifies the text for the report, transform or macro code. The text is unformatted, plain text.

Σ Use the vbCrLf string data constant to insert a carriage-return and linefeed string.

Transforms: To change the value of a transform variable, use the AddVariableExpression method. Run transforms using the Execute method.

TickLabelAttributes Property Returns the tick label Text objects for the specified Axis object.

Title Property A Notebook object property. Sets the Name of the NotebookItem object of the Notebook file, and the Title field under the Summary tab of the Windows 95/98 file Properties dialog box. Does not affect the file name; to change the file name, use either the Name or FullName property.

Top Property Sets or returns the top coordinate of the application window or specified notebook document window.

Visible Property A property common to the Application, Notebook, and NotebookItems document objects. Sets or returns a Boolean indicating whether or not the application or specified document window is visible. Do not set the Application property to False from within SigmaPlot or you will lose access to the application.

Note that hidden document windows will still appear in the notebook window tree. Setting Visible=False for a notebook object hides all document windows for the notebook as well.

Width Property Sets or returns the width of the application window or specified notebook document window.

SigmaPlot Methods

Methods are an action that can be performed on or by an object-think of methods as "verbs." For example, the WorksheetEditItem object has Copy and Clear methods. Methods can have parameters that specify the action ("adverbs").

For more information, refer to SigmaPlot Automation Help from the SigmaPlot Help menu.

Activate Method Makes the specified notebook the object specified by the ActiveDocument property.

Add Method The Add method is used in collections to add a new item to the collection. The parameters depend on the collection type:

Collection	Value	Parameters
Notebooks		None
NotebookItems	1	CT_WORKSHEET
	2	CT_GRAPHICPAGE
	2	CT_FOLDER
	4	CT_STATTEST
	5	CT_REPORT
	6	CT_FIT
	7	CT_NOTEBOOK
	8	CT_EXCELWORKSHEET
	9	CT_TRANSFORMTransformItem
	10	
Graph Objects	2	GPT_GRAPH, more...
	3	GPT_PLOT, more...
	4	GPT_AXIS, more...
	5	GPT_TEXT, more...
	6	GPT_LINE, more...
	7	GPT_SYMBOL, more...
	8	GPT_SOLID, more...
	9	GPT_TUPLE, more...
	10	GPT_FUNCTION, more...
	11	GPT_EXTERNAL, more...
	12	GPT_BAG, more...
NamedRanges		Name string, Left long, Top long, Width long, Height long, NamedRange

The GraphObjects collection uses the CreateGraphFromTemplate and CreateWizardGraph methods to create new GraphObject objects.

AddVariable Expression Method

Allows the substitution of any transform variable with a value.

ApplyPageTemplate Method

Overwrites the current GraphItem using a new page template specified by the template name. Optionally, you can specify the notebook file to use as the source of the template page. If no template file is specified, the default template notebook is used, as returned by the Template property.

AddWizardAxis Method

Adds an additional axis to the current graph and plot on the specified GraphItem object, using the AddWizardAxis options. If there is only one plot for the current graph, SigmaPlot will return an error. Use the following parameters to specify the type of scale, the dimension, and the position for the new axis:

Scale Type
SAA_TYPE_LINEAR
SAA_TYPE_COMMON (Base 10)
SAA_TYPE_LOG (Base e)
SAA_TYPE_PROBABILITY
SAA_TYPE_PROBIT
SAA_TYPE_LOGIT

Dimension		
DIM_X	1	The X dimension
DIM_Y	2	The Y dimension
DIM_Z	3	The Z dimension (if applicable)

Position	
AxisPosRightNormal	0
AxisPosRightOffset	1
AxisPosTopNormal	2
AxisPosTopOffset	3

Position	
AxisPosLeftNormal	4
AxisPosLeftOffset	5
AxisPosBottomNormal	6
AxisPosBottomOffset	7

AddWizardPlot Method

Adds another plot to the current graph on the specified GraphItem object using the following parameters to define the plot:

Parameter	Values	Optional
graph type	any valid type name	no
graph style	any valid style name	no
data format	any valid data format name	no
column array	any column number/title array	no
columns per plot array	array of columns in each plot	yes
error bar source	any valid source name	error bar plots only
error bar computation	any valid computation name	error bar plots only
angular axis units	any valid angle unit name	polar plots only
lower range bound	any valid degree value	polar plots only
upper range bound	any valid degree value	polar plots only
ternary units	upper range of ternary axis scale	ternary plots only

Clear Method

Clears the selection in items that support this.

Close Method

The Close method is used to close notebooks and notebook items. The parameters for each object type depend on the object:

Notebook Save before closing Boolean, filename string

NotebookItems Save before closing Boolean

Specifying a Save before closing value of "False" closes the notebook or notebook item without saving changes made to the object.

Note that for NotebookItems and SectionItems, a Close corresponds to an Expanded = False.

Copy Method

Copies the currently selected item within the specified notebook item. If no item is selected, then an error is returned.

CreateGraphFromTemplate Method

Creates a graph for a GraphItem from the Graph Style Gallery.

CreateWizardGraph Method

Creates a graph in the specified GraphItem object using the Graph Wizard options. These options are expressed using the following parameters:

Parameter	Values	Optional
graph type	any valid type name	no
graph style	any valid style name	no
data format	any valid data format name	no
columns plotted	any column number/title array	no
columns per plot	array of columns in each plot	yes
error bar source	any valid source name	error bar plots only
error bar computation	any valid computation name	error bar plots only
angular axis units	any valid angle unit name	polar plots only
lower range bound	any valid degree value	polar plots only
upper range bound	any valid degree value	polar plots only
ternary units	upper range of ternary axis scale	ternary plots only

Cut Method

Removes the current selection from the specified object, placing the contents on the clipboard. This method is equivalent to using the Copy method, followed by the Clear method. However, whereas Copy places OLE link formats on the clipboard for GraphItem objects, Cut does not.

Delete Method

Deletes a notebook item from a NotebookItems collection, as specified using an index number or name. If the item does not exist, an error is returned.

DeleteCells Method

Deletes the specified cells from the worksheet. The remaining cells can be moved in two different directions to fill in the deleted region:

1. Shift Cells Up
2. Shift Cells Left

To delete an entire column or row, simply set the column bottom or row right value to the system maximum:

Rows: 32,000,000

Columns: 32,000

Execute Method

Used to execute the specified TransformItem.

Export Method

SigmaPlot Automation supports export of NativeWorksheetItem objects, GraphItem objects, and NotebookItem objects of type CT_NOTEBOOK.

- ➤ If applied to a NativeWorksheetItem object, this method exports either the data in the worksheet to the specified data format or the entire notebook to a previous SPW file format.
- ➤ If applied to a GraphItem object, this method exports either the graphic data on the page to the specified graphic format or the entire notebook to a previous SPW file format.
- ➤ If applied to the first NotebookItem in the NotebookItemList, this method exports the entire notebook to a previous SPW file format.

GetAttribute Method

The GetAttribute method is used by all graph page objects to retrieve current attribute settings. Attributes are numeric values that also have constants assigned to them. For a list of all these attributes and constants, see SigmaPlot Constants.

Message Forwarding: If you use the GetAttribute method to retrieve an attribute that does not exist for the current object, the message is automatically routed to an object that has this attribute using the message forwarding table.

To use the Object Browser to view Constants You can view alternate values for attributes and constants by selecting the current attribute value, then clicking the Object Browser button. All valid alternate values will be listedóto use a different value, select the value and click Paste.

GetData Method

Returns the data within the specified range from a DataTable object as a variant. This variant is always returned as a one dimensional array. If a 2D array is specified, the data is stacked as a linear array. To ensure that GetData retrieves all data in a row or column, specify the worksheet maximum as the right of bottom parameter.

GetMaxLegalSize Method

Initializes the values of the maximum worksheet column and row values, so that they can be returned as a variables.

GetMaxUsedSize Method

Initializes the values of the maximum used worksheet column and row values, so that they can be returned as a variables.

Goto Method

Moves worksheet cursor position to the specified cell coordinate for the current NativeWorksheetItem or ExcelItem object.

Help Method

Opens an on-line Windows help file to a specific topic context map ID number (as a long) or search index keyword (K-word). You can use either the ID number or an index keyword. If any of the parameters are left empty, the SigmaPlot help file defaults are used.

Import Method

Imports a data file with the specified file name into an existing NativeWorksheetItem. You can specify both the import starting location in the SigmaPlot worksheet, as well as the range of data imported.

InsertCells Method

Inserts the specified block of cells into the worksheet. The existing cells can be moved in two different directions to accommodate the inserted region:

1. Shift Cells Down
2. Shift Cells Right

To insert an entire column or row, simply set the column bottom or row right value to the system maximum:

Rows: 32,000,000

Columns: 32,000

Mesh Method

Converts unsorted xyz triplet data to evenly incremented mesh data, as required by mesh and contour plots. The optional parameters control the results columns, mesh range and increment, and original datapoint weighting. Note that the output columns must be specified if the data is to be returned to the worksheet.

Item Method

Returns an object from the collection as specified by the object index number or name. Note that the index begins with 0 by default. The Item method is equivalent to specifying an object from the collection object using an index. If the item does not exist, an error is returned.

ModifyWizardPlot Method

Modifies the current plot on the specified GraphItem object using the following parameters:

Parameter	Values	Optional
graph type	any valid type name	no
graph style	any valid style name	no
data format	any valid data format name	no
column array	any column number/title array	no

Parameter	Values	Optional
columns per plot array	array of columns in each plot	yes
error bar source	any valid source name	error bar plots only
error bar computation	any valid computation name	error bar plots only
angular axis units	any valid angle unit name	polar plots only
lower range bound	any valid degree value	polar plots only
upper range bound	any valid degree value	polar plots only
ternary units	upper range of ternary axis scale	ternary plots only

NormalizeTernary Data Method

Normalize three columns of raw data to 100 or 1 for a ternary plot.

Open Method

Opens the notebook specified within the Notebooks collection, or the specified notebook item. The parameter depends upon whether you are opening a notebook or a notebook item.

Note that for NotebookItems and SectionItems, an Open corresponds to an Expanded = True.

Paste Method

Place the contents of the Windows Clipboard into the selected notebook item document, at the current position, if applicable. The format specified is an available clipboard format, as displayed by the Edit menu Paste Special command.

Print Method

Prints the selected item, including any items within specified NotebookItems and SectionItems. Specifying the Notebook prints all items in the notebook.

PrintStatsWorksheet Method

Prints the NativeWorksheetItemís statistics worksheet. If the worksheet has not been opened using the ShowStatsWorksheet property, this method fails.

PutData Method

Places the specified variant into the worksheet starting at the specified location. The data can be a 2D array.

Quit Method

Ends SigmaPlot. If SigmaPlot is in use, then this method is ignored.

Redo Method

Redoes the last undone action for the specified object. If redo has been disabled in SigmaPlot for either the worksheet or page, this method has no effect.

Remove Method

Deletes the specified object. The index can be a number or a name. If the specified index does not exist, an error is returned.

Run Method Runs a FitItem or Macro without closing the object.

SaveAs Method Save a notebook file for the first time, or to a new file name and path. Note that you need to provide the file extension. Recognized SigmaPlot notebook file extensions are .JNB, .JNT, and .JFL

Save Method Saves a Notebook object to disk using the current FullName , or a notebook item to the notebook (without saving the notebook file to disk). If no FullName exists for a notebook, an error occurs. To save a notebook that has not yet been saved, you must use the SaveAs method.

Note: Transform text can be saved to an .xfm file by naming the transform first with the full file name, extension, and path.

Select Method Selects all of the items within the specified selection region. In addition, if “Top” equals “Bottom” and “Right” equals “Left,” the resulting selection includes the object that the specified point lies within.

If “AddToSelection” is “False” then the previous selection list is replaced by the new list. If “True,” then the newly selected items are added to the existing selection list.

SelectAll Method Selects the entire contents of the item.

SelectObject Method Clears the current GraphItem selection list and selects the specified graph object so that it can be altered using the SetSelectedObjectsAttribute method. Line and Solid objects can only be selected if they are top level drawing objects (not child objects of other objects).

SetAttribute Method The SetAttribute method is used by all graph page objects to change current attribute settings. Attributes are numeric values that also have constants assigned to them. For a list of all these attributes and constants, see SigmaPlot Constants.

Message Forwarding: If you use the SetAttribute method to change an attribute that does not exist for the current object, the message is automatically routed to an object that has this attribute using the message forwarding table.

Using the Object Browser to view Constants You can view alternate values for attributes and constants by selecting the current attribute value, then clicking the Object Browser button. All valid alternate values will be listedóto use a different value, select the value and click Paste.

SetCurrentObject Attribute Method Changes the attribute specified by “Attribute,” of the “current object” on the graphics page. Use one of the following three techniques to set the “current object” on the graphics page:

➤ Click the object using the mouse.

- Use the SigmaPlot menus (e.g. “Select Graph”).
- Use the SetObjectCurrent method.

If the specified GraphItem is not open or there is no current object of the appropriate type on the page, the method will fail.

SetObjectCurrent Method

Sets the specified object to the “current” object for the purpose of the “SetCurrentObjectAttribute” command.

It the specified GraphItem is not open, the method will fail.

SetSelectedObjects Attribute Method

Changes the attribute specified by “Attribute” for all the selected objects on the graphics page. Select graphics page objects using one of the following two techniques:

- Click the object with the mouse.
- Use the SelectObject method.

TransposePaste Method

Pastes the data in the clipboard into the worksheet, transposing the row and column indices of the data such that rows and columns are swapped. If there is nothing in the clipboard or the data is not of the right type, nothing will happen.

Undo Method

Undoes the last performed action for the specified object. If undo has been disabled in SigmaPlot for either the worksheet or page, this method has no effect.