

Automation Editor (IDE) User's Manual

Release PADS VX.2.6

Document Revision 4

© 2007-2019 Mentor Graphics Corporation
All rights reserved.

This document contains information that is proprietary to Mentor Graphics Corporation. The original recipient of this document may duplicate this document in whole or in part for internal business purposes only, provided that this entire notice appears in all copies. In duplicating any part of this document, the recipient agrees to make every reasonable effort to prevent the unauthorized use and distribution of the proprietary information.

This document is for information and instruction purposes. Mentor Graphics reserves the right to make changes in specifications and other information contained in this publication without prior notice, and the reader should, in all cases, consult Mentor Graphics to determine whether any changes have been made.

The terms and conditions governing the sale and licensing of Mentor Graphics products are set forth in written agreements between Mentor Graphics and its customers. No representation or other affirmation of fact contained in this publication shall be deemed to be a warranty or give rise to any liability of Mentor Graphics whatsoever.

MENTOR GRAPHICS MAKES NO WARRANTY OF ANY KIND WITH REGARD TO THIS MATERIAL INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

MENTOR GRAPHICS SHALL NOT BE LIABLE FOR ANY INCIDENTAL, INDIRECT, SPECIAL, OR CONSEQUENTIAL DAMAGES WHATSOEVER (INCLUDING BUT NOT LIMITED TO LOST PROFITS) ARISING OUT OF OR RELATED TO THIS PUBLICATION OR THE INFORMATION CONTAINED IN IT, EVEN IF MENTOR GRAPHICS HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

U.S. GOVERNMENT LICENSE RIGHTS: The software and documentation were developed entirely at private expense and are commercial computer software and commercial computer software documentation within the meaning of the applicable acquisition regulations. Accordingly, pursuant to FAR 48 CFR 12.212 and DFARS 48 CFR 227.7202, use, duplication and disclosure by or for the U.S. Government or a U.S. Government subcontractor is subject solely to the terms and conditions set forth in the license agreement provided with the software, except for provisions which are contrary to applicable mandatory federal laws.

TRADEMARKS: The trademarks, logos and service marks ("Marks") used herein are the property of Mentor Graphics Corporation or other parties. No one is permitted to use these Marks without the prior written consent of Mentor Graphics or the owner of the Mark, as applicable. The use herein of a third-party Mark is not an attempt to indicate Mentor Graphics as a source of a product, but is intended to indicate a product from, or associated with, a particular third party. A current list of Mentor Graphics' trademarks may be viewed at: mentor.com/trademarks.

The registered trademark Linux[®] is used pursuant to a sublicense from LMI, the exclusive licensee of Linus Torvalds, owner of the mark on a world-wide basis.

End-User License Agreement: You can print a copy of the End-User License Agreement from: mentor.com/eula.

Mentor Graphics Corporation
8005 S.W. Boeckman Road, Wilsonville, Oregon 97070-7777
Telephone: 503.685.7000
Toll-Free Telephone: 800.592.2210
Website: mentor.com
Support Center: support.mentor.com

Send Feedback on Documentation: support.mentor.com/doc_feedback_form

Revision History ISO-26262

Revision	Changes	Status/Date
4	Modifications to title page to reflect the latest product version supported. Approved by Regis Krug. All technical enhancements, changes, and fixes listed in the <i>Personal Automated Design System Release Notes</i> for this product are reflected in this document. Approved by Mike Bare.	Released September 2019
3	Modifications to title page to reflect the latest product version supported. Approved by Regis Krug. All technical enhancements, changes, and fixes listed in the <i>Personal Automated Design System Release Notes</i> for this product are reflected in this document. Approved by Mike Bare.	Released March 2019
2	Modifications to title page to reflect the latest product version supported. Approved by Regis Krug. All technical enhancements, changes, and fixes listed in the <i>Personal Automated Design System Release Notes</i> for this product are reflected in this document. Approved by Mike Bare.	Released September 2018
1	Modifications to improve the readability and comprehension of the content. Approved by Regis Krug. All technical enhancements, changes, and fixes listed in the <i>Personal Automated Design System Release Notes</i> for this product are reflected in this document. Approved by Mike Bare.	Released February 2018

Author: In-house procedures and working practices require multiple authors for documents. All associated authors for each topic within this document are tracked within the Mentor Graphics Technical Publication's source. For specific topic authors, contact Mentor Graphics Technical Publication department.

Revision History: Released documents include a revision history of up to four revisions. For earlier revision history, refer to earlier releases of documentation on Support Center.

Table of Contents

Revision History ISO-26262

Chapter 1

Script Creation	9
Using Scripts and Forms	10
How Do Scripts Work?	10
Distributing Scripts and Forms	10
Script Editor and Mini-Editor Functionality	12
Mini-Editor	13
Script Editor Only Functionality	15
Scripting for Applications	16
Scripts, Forms, and Automation	16
What Can I Do With Scripting?	16

Chapter 2

Scripting Object Usage	19
Adding Objects to a View	19
Adding Code to Objects	19
View Basics	21
Layers Sheet	22
Setting Tab Order for a Control in a View	23
Object Sheet	24

Chapter 3

Forms and the Form Editor	25
Form Editor	25
Setting the Scripting Language	26
Creating a Form	26
Opening an Existing Form	27
Setting the Form to Read-Only	27
Edit Mode at a Glance	28
View Mode Overview	30
Setting Up Your Edit Environment	31
Changing the Size of a Form	31
Ruler Bars	31
Grid Settings	32
Setting Snap to Grid in Active View (Edit Mode)	32
Changing the Layout Grid Size	32
Line Snapping	33
SnapToCenter	33
Testing a View	33
Returning to Edit Mode	34

Defining Mnemonics	34
Checking Mnemonics	34
Manipulating Objects	35
Selecting a Single Object	35
Selecting Multiple Objects	36
Moving Objects	36
Moving an Object Within a View	36
Moving or Copying Objects Between Views	36
Deleting an Object	37
Copying and Pasting an Object	37
Setting the Dominant Object in a Selection	37
Aligning Objects	39
Aligning Objects Along Edges	39
Aligning Objects On Their Centers	39
Centering Objects in the View	40
Spacing Objects	40
Resizing Objects	40
Sizing Individual Objects	41
Grouping and Ordering Objects	41
Rotation	43
Rotating an Object Using the Draw Menu Item	43
Rotating an Object Using the Format Bar Toolbar	44
Editing End Points	44
Hot Spots	44
Object Properties	45
Frame and View Properties	49
More Colors	50
Frame and View Events	51
Form Editor Menus	53
File Menu	53
Edit Menu	53
View Menu	54
Run Menu	55
Layout Menu	55
Draw Menu	56
Form Editor Toolbars	57
Standard Toolbar	58
Object Bar	59
Format Bars	61
Layout Bar	63
Script Bar	65
Chapter 4	
Built-in Objects	67
Text Object	68
TextVar Object	68
Connector Object	68
Line Object	69

Table of Contents

Freehand Object	69
Polyline Object	69
Border Object	70
Arc Object	71
Ellipse Object	71
Frame Object	71
Bitmap Object	72
Hilite Object	73
Pointer Object	73
Gauge Object	74
Graph Object	74
Button Object	75
CheckBox Object	76
RadioButton Object	76
ComboBox Object	77
ListBox Object	78
TextBox Object	78
MultiTextBox Object	79
Slider Object	79
Spinner Object	80
ActiveX Object	80
OLE Object	81

Third-Party Information

End-User License Agreement with EDA Software Supplemental Terms

Chapter 1

Script Creation

ActiveX Scripting engines (languages and interpreters) provided by Microsoft are VBScript (Visual Basic Scripting Edition) and JScript (Microsoft's implementation of the ECMAScript standard, similar to Netscape's JavaScript). The ActiveX Scripting standard allows third parties to develop their own ActiveX Scripting engines, and engines are already available on PC platforms for Python and PERL.

Note



In UNIX and Linux, only VBScript and JScript are available.

To download the latest version of the Microsoft scripting engines, VBScript documentation, Jscript documentation, and the Microsoft Script Debugger, use the following URL:

<http://www.microsoft.com/downloads>

Type 'script' in the search box, and click 'Go' to find all script downloads.

ActiveX Scripting hosts are applications that allow scripting by means of ActiveX Scripting engines. Current examples include Internet Explorer, Internet Information Server (IIS), the Visual C++ development environment, and Windows Scripting Host (WSH). The application is an ActiveX Scripting host. In general, an ActiveX Scripting host can make use of any ActiveX Scripting engine that the user has installed.

Using Scripts and Forms	10
Script Editor and Mini-Editor Functionality	12
Scripting for Applications	16

Using Scripts and Forms

In most applications, the following variables are automatically defined:

- **Application** (and its members)

Represents the top-level object in the application. All the members of the Application object are also accessible.

- **ScriptEngine**

Represents the running script engine. Its members include the functions defined in your script. This is especially useful for setting the **CommandBarButton.Target** property.

Note



You should use **ScriptThis** instead of **ScriptEngine** when configuring menu and accelerator handlers in IDE.

- **Form** (or the ID property of the form if you've changed it with the form's Properties dialog box)

Represents the running form (a window, such as a dialog box).

You can reference these objects without qualifying with a container object. For example, in a script you can refer to **Visible**, rather than **Application.Visible**.

How Do Scripts Work?	10
Distributing Scripts and Forms	10

How Do Scripts Work?

A script communicates with its host and other applications through a COM technology called *Automation* (formerly OLE Automation). To support Automation, an application requires an object model that exposes certain objects, with their properties and methods, to external applications. An application that supports Automation is called an *Automation Server*. An Automation server makes its objects available. An *Automation Client* can manipulate the Automation Server's objects.

With Automation, these applications and their components become objects you can control programmatically. In addition to manipulating objects exposed by the scripting host, a script can also manipulate objects served by other Automation servers. For example, a script in the application can manipulate an Excel spreadsheet or a Word document. The reverse of this is also possible: a Word or Excel script can manipulate a PCB design.

Distributing Scripts and Forms

You can centrally distribute and update scripts and forms.

When you specify a form or script to run on startup, the pathname of the file is stored in a file named *scripts.ini*. By making this a shared directory pointed to by each user's WDIR environment variable, groups of users can run the same forms and scripts when an application starts up.

For example, some applications will try to run each script/form in every *scripts.ini* file that they find. They use all *scripts.ini* files that they find in the project directory and every directory in WDIR. The application must be specified for the AutoActive tools.

The AutoActive tools (Xpedition Layout, Xpedition FabLink, Xpedition Layout Team Server, and so on) also look in the *<mgc_home./release/SDD_HOME/standard/automation/startup* directory. Users can place company-wide information in the *scripts.ini* in this location.

Script Editor and Mini-Editor Functionality

VBScript can be added to your application by using either the Script Editor or the Mini-Editor. The Script Editor is the view behind the Form Editor and can be accessed by pressing **Ctrl+T** or by clicking the View Script icon in the Standard toolbar.

Table 1-1. Script Editor and Mini-Editor Functionality

Functionality	Description
Bookmarks	Set bookmarks to quickly jump to bookmarked lines in the code.
Breakpoints	Set breakpoints for debugging purposes.
ListBox Objects	IntelliSense provides a complete list of objects, properties/methods, functions, and tooltip info.
Comment/Uncomment	Comments can be placed in code using the apostrophe (').
Tab/Untab	Highlighted sections of code can be formatted using Tab or Shift-Tab.
Find, Find Next and Replace	Ctrl+F to find occurrences of a specified string.
Right Click Menu	Access functions such as Cut, Copy, and Paste, as well as set bookmarks and breakpoints.
Edit (Declarations)	Local declarations to the view can be edited.

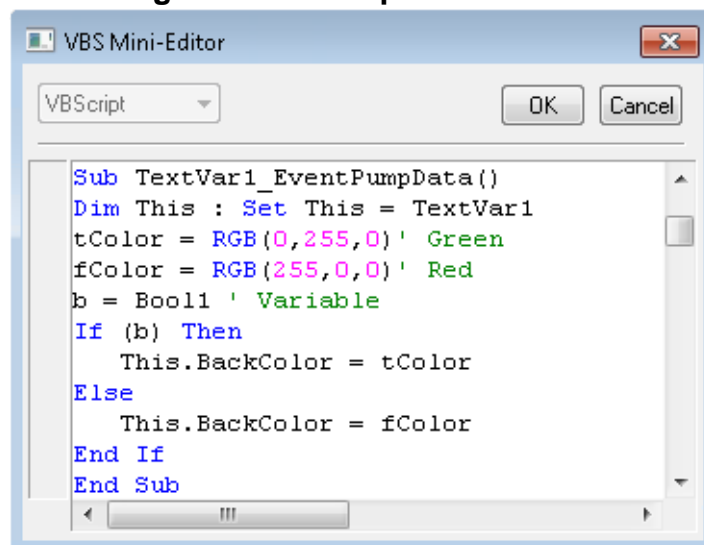
To see more functions available to only the Script Editor, see “[Script Editor Only Functionality](#)” on page 15.

Mini-Editor

To Access: When there is an error in running a script, the Mini-Editor window displays.

The figure shows the Mini-Editor attached to a VBScript property defining a subroutine (event):

Figure 1-1. Example Mini-Editor



Description

In this case, a "TextVar1" object's "EventPumpData()" property is being edited. The mini-editor is described according to its three sections.

Upper Part - Selection. The controls are used as follows

Fields

Table 1-2. VBS Mini-Editor Contents

Field	Description
VBScript	Shows that the Mini-Editor is in VBS mode. There are other modes (covered below) which bypass VBS and just return a constant or a variable value. The other modes run much faster than running a VBS function.
OK	OK. Also causes "check syntax" to run.
Cancel	Cancel and disregard all changes.

Usage Notes

Lower Part- VBScript Code. The following show the script for the "EventPumpData" function:

```
Sub MyTextVar1_EventPumpData()  
:Dim This As Object : Set This = MyTextVar1  
tColor = RGB(0,255,0)' Green  
fColor = RGB(255,0,0)' Red  
b = Bool1 ' Variable  
If (b) Then  
    This.BackColor = tColor  
Else  
    This.BackColor = fColor  
End If  
End Sub
```

This subroutine is run whenever new data is available to the view. The mini-editor automatically generates the code in italics (the first two and last lines).

- **First two lines** — The subroutine name is a combination of the object code ("MyTextVar1" is this case) and the given property ("EventPumpData"), stripped of characters illegal in function names. The name should be unique to the view. You can call other object/property functions (and subroutines) using the function name. The *This* variable is set to equal the object code of the object, and is serves as a convenient handle to the object (in this case you could also use "MyTextVar1" directly instead of *This*).
- **Last line(s)** — "End Sub" terminates the subroutine's definition.

Tip

For functions you must set the *Retn* variable equal to the return value so that it can be assigned to the function return. The *This* and *Retn* variables serve certain purposes:

- They are easy handles to more complex representations.
 - More importantly, using *This* and *Retn* allows you to change the object code without changing any VBScript code!
 - Notice that the "MyTextVar1" object code is not used outside of the automatically generated code (in this case).
-
- **Typed in code** — This function is getting the value for variable "Bool1". It also setting the background color of the object according to "Bool1": green for true and red for false. RGB() is a function pre-defined in DECLARE.BAS which must be in the boot directory. You can add your own function definitions into DECLARE.BAS (if available).

Here are the actions tied to the VBS Mini-Editor's shortcut menu:


Table 1-3. VBS Mini-Editor Context Menu

Field	Description
Cut / Copy / Paste	Clipboard functions for text in the buffer.

Table 1-3. VBS Mini-Editor Context Menu (cont.)

Field	Description
List Objects	Brings up a listbox of objects specified by All, Form, Variables, or Business.
List Properties/ Methods	Brings up a listbox of properties/methods associated with the specified object.
List Functions	Brings up a listbox of functions associated with the specified object.
Quick Info	Brings up information about the variable specified.
Check Syntax...	Compile the buffer and check for syntax errors. Errors are flagged with a message box and the offending text is highlighted. This check does not catch run time errors (such a calling an undefined function). Run time errors are trapped causing the view to switch back into edit mode.
Revert to Saved	Will load the previously saved script
Toggle Breakpoint	Sets or removes a breakpoint.
Toggle Bookmark	Sets or removes a bookmark


Tip

 Pressing '.' in the mini-editor also can be used to bring up a list of objects.

Script Editor Only Functionality

The following lists the functionality found in the Script Editor only.

Table 1-4. Script Editor Only Functionality

Item	Description
View Modes	Using the  icons, it is possible to view script by-function, by-object, or full module.
Drop Down Combos	You can switch between functions via Object/Event combos (similar to VB).
Edit (Global Declarations)	In addition to local declarations, global declarations can be edited and saved in DECLARE.BAS.
Script Editor Options Dialog	Set syntax colorization colors, font, etc. To access this dialog box click the Options icon located in the Layout toolbar.

Scripting for Applications

The applications give you the ability to programmatically complete tasks using scripts. These application add-ins work seamlessly within the framework of each application, enhancing and leveraging its functionality to suit your exact requirements.

At its simplest, you can use scripting to combine commands to perform repetitive operations quickly and easily. However, by using the application's event-driven object model, you can build complete custom solutions, including new user interfaces.

Many applications expose objects for the purpose of Automation. You use those exposed objects to perform tasks using scripts and forms. The details of these Automation interfaces vary from application to application. In addition, the applications offer full-featured form creation, in which you can use scripting to handle the events associated with each object in the form. Scripting languages that the applications support include VBScript and JScript.

Scripts, Forms, and Automation	16
What Can I Do With Scripting?.....	16

Scripts, Forms, and Automation

You can run scripts automatically when you start running applications. You can also run scripts by opening them from within these applications. You can write scripts to be self-contained custom commands or you can write script code to be run through forms. Forms are interfaces (windows, including dialog boxes) that you design by adding controls to a form and writing the script code to handle events associated with each control.

When a user acts on the form's controls, as when selecting an item in a listbox or clicking a button, the appropriate functions in the form's script run. Functions also run for other reasons (as when the form loads or because of events fired by other objects). Each script can contain many subroutines or functions.

Scripts by themselves run and then exit immediately. They do not wait for events, so event handlers only apply in the context of forms. If you want a particular subroutine to run in response to an event, you must include the appropriate script code as part of a form. Once invoked, forms remain open (and running) until you switch to edit mode or close the form.

What Can I Do With Scripting?

A script is a series of Automation instructions that you group together as a single command to accomplish a task automatically. You can use scripting to automate time-consuming, repetitive operations in most applications, but you can also create scripts and forms that provide brand-new functionality, such as design-navigation aids and web browsing add-ins.

Some typical uses for scripts and forms are:

- To automate a complex series of tasks
- To perform custom checks on a schematic.
- To add a new command to a popup menu.
- To add a new menu with custom commands bound to other scripts.
- To build a new dialog box that adds new functionality.

In addition to running scripts at application startup, you can bind them to a toolbar icon or menu command. You can create forms and call the scripts from the form. After you've assigned a script to a toolbar or menu, running the script is as simple as clicking the toolbar icon or menu item. You can also run standalone scripts, using the run command from the command line. Included in your applications installation are samples of scripts and forms that you can test.

Chapter 2

Scripting Object Usage

You can add control objects such as buttons, text boxes, and images to a form view. Add code to specify action when a control is clicked.

Adding Objects to a View	19
Adding Code to Objects	19
View Basics	21

Adding Objects to a View

You can add objects to a view using point and click, or by dragging the object with the mouse.

Procedure

1. On the Object Bar palette, click the icon for the object you want.
2. Select one of the following methods to place the object in the view.

Table 2-1. Add Objects to a View

If you want to...	Do the following...
Use point and click	Move the mouse pointer to the view and click at the position you want. The object is given a default size as previously defined for the type of object.
Drag with the mouse	<ol style="list-style-type: none">1. Place the mouse pointer where you want the upper-left corner of the object to be located.2. Hold down the left mouse button.3. Move the mouse pointer to the right and down; a dotted outline of the object appears.4. When the object is the size you want, release the mouse button.

Adding Code to Objects

You can add code to an object by double-clicking the object.

Prerequisites

- You must have added an object to the view.

Procedure

1. Click the View Script icon on the Standard toolbar, or double-click the object. Either action will display the code for the object.
2. You can also change the object properties in the Property Sheet, by right-clicking the object, and selecting the **Object Properties** menu item.


View Basics

You can add a control to a view, specify its insertion layer, and set the tab order and drawing order for controls within a view.

Layers Sheet	22
Setting Tab Order for a Control in a View	23
Object Sheet	24

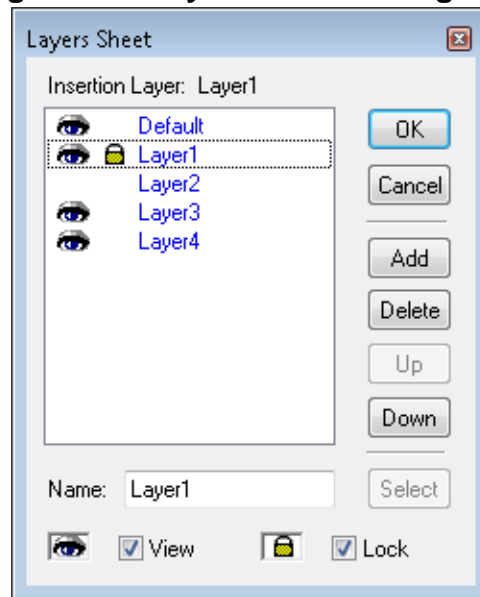
Layers Sheet

To access:

- IDE (form-editing mode) **Layout > Layers**
- IDE Layout bar Layer Sheets icon 
- IDE (background) context menu **Layers**

Use the Layers Sheet dialog box to add or delete view layers (except the Default layer which is always first in the list and cannot be deleted).

Figure 2-1. Layers Sheet Dialog Box



Fields

Table 2-2. Layers Sheet Dialog Box Contents

Field	Description
Up	Move an individual layer up the list.
Down	Move an individual layer down the list.
Select	Select all objects on a particular layer.
Name	Change the name of the layer.
View	Toggle the layer's View state (Visible/Invisible).
Lock	Toggle the layer's Lock state (Locked/Invisible).

Usage Notes

Moving an individual layer up or down in the list has no effect on the drawing of the view (objects are always drawn according to their Z-order in the Object Sheet dialog box).

The current Insertion Layer determines which layer new objects are inserted into. Click on a layer name in the list to select it as the Insertion Layer.

Double-click the View and Lock items to the left of the layer name in the list to toggle these states.

Setting Tab Order for a Control in a View

Tab order is the order in which the tab key moves the input focus from one control to the next within a view. To make a control part of the tab order, set the Tabstop property in the Property Sheet to **Yes**. Non-control objects do not have the Tabstop property.

Caution




If your view contains overlapping objects, changing the tab order may change the way the controls display. Controls which come first in the tab order always display on top of any overlapping controls and non-control objects that follow them in the tab order.

Prerequisites


- You must have at least one control in the view.

Procedure

1. If the Layout Bar is not visible, choose **View > Toolbars > Layout Bar**, or use the shortcut: **Shift + F2**.
2. On the Layout Bar, click on **Object Sheet** icon . The Object Sheet (Tab Order) dialog box appears. A number at the upper left of each control shows its current tab order. Non-control objects do not have numbers, but they are listed according to their tab order.
3. Change the tab order by clicking on the desired object in the Object Sheet dialog box, then clicking the **Up** or **Down** button.
4. Click the **OK** button to save changes to the tab order, or click the **Cancel** button to exit the Object Sheet dialog box without saving changes.

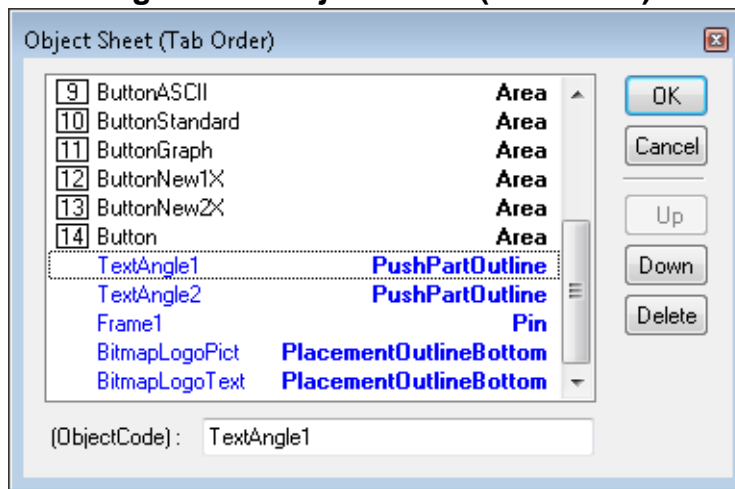
Object Sheet

To access:

- IDE (form-editing mode) **Layout > Objects**
- IDE Layout bar Object Sheet icon 
- IDE (background) context menu **Objects**
- Shortcut **Ctrl+I**

The Object Sheet (Tab Order) dialog box lists the objects in a view. The object order can be moved up or down. This affects the control drawing order and tab order.

Figure 2-2. Object Sheet (Tab Order)



Fields

Table 2-3. Object Sheet Dialog Box Contents

Field	Description
Up	Move an individual control up in tab order and drawing order.
Down	Move an individual control down in tab order and drawing order.
(ObjectCode)	Edit an individual control's object code.

Usage Notes

Objects are listed in blue text, controls are in black text with their tabbing sequence number at left. Object codes can also be edited on the fly.

Chapter 3

Forms and the Form Editor

Some applications allow you to create forms within the application to launch scripts. Using a form gives you a visual front end for each script. In some applications, for example, you could create a form that asks for an attribute name and visibility setting.

After you input the data in the form, it could run code that performs some action on all attributes that match the search criteria. Forms are also useful for displaying information. In other applications you could create a form that asks the user to enter a component name. The form could then run a script that gathers data about that component and then displays it to you.

Forms consist of controls that you select and define properties for at design time. You determine the runtime behavior of objects using the form's script code.

You can run a form in one of the following distinct ways:

- When you start an application.
- While the application is running, by opening the script or form file.
- By calling the form from a script or another form.


Form Editor	25
Setting the Scripting Language	26
Creating a Form	26
Opening an Existing Form	27
Setting the Form to Read-Only	27
Edit Mode at a Glance	28
Setting Up Your Edit Environment	31
Manipulating Objects	35
Frame and View Properties	49
Frame and View Events	51
Form Editor Menus	53
Form Editor Toolbars	57

Form Editor

The Form Editor is a WYSIWYG design environment that allows you to create forms and add controls (such as command buttons, check boxes, text boxes, and so on). The Form Editor

includes a code window, in which you write the script code (event handlers) associated with each object.

Important Form Editor commands, available from the Standard toolbar, include:

Run Script command  Runs the open form. Shortcut: F5

Stop Script command  Stops the open form.

Object View / View Script command  Toggles the view in the Form Editor between the form objects and the scripting code.

Setting the Scripting Language

The default scripting language of forms is VBScript. If you want to write event handlers in JScript you must select it in the Select Script Language dialog box as follows.

Procedure

1. In the application, choose **File > New Script Form** from the menu bar. The **Select Script Language** dialog box opens.
2. Select either **VBScript** or **JScript** in the Form Language dropdown list and click **OK**. The Form Editor appears.

Creating a Form

You can create a new form from within an application.

Procedure

1. Choose **File > New Script Form** from the menu bar. The Select Script Language dialog box appears.
2. Choose either VBScript or JScript option in the Form Language dropdown list and click **OK**. The Form Editor appears.
3. Design your form by dragging objects from the Object Bar palette and defining their properties as needed.
4. Switch to the Script Editor (Click the View Script icon in the Standard toolbar) and write the script code (event handlers) for each object.
5. Any custom functions must be placed in the Declarations section of the script.
6. Select **File > Save** to save the form and associated script.


Opening an Existing Form

You can open an existing form to run it and to edit it from within an application.

Procedure

1. Select **File > Open Script Form**.
2. Select or browse to the *.efm* file you want to open and click **Open**. You can also press the **Ctrl** key while opening the form to open in Edit mode.

Results

- The form runs. To stop the form, click the Stop Form icon on the Standard toolbar. 
- To edit the form, drag and drop objects from the Object Bar palette, or click on the View Script icon on the Standard toolbar, to edit the code.

Setting the Form to Read-Only

You can set the form to read-only to prevent users from modifying the form.

Procedure

1. Select **File > Open Script Form**.
2. Select, type, or browse to the *.efm* file you want to open, right-click, and select properties.
3. On the General tab, to the right of Attributes, click in the box for Read-only, and then click **OK**.
4. Select the same file again, and click **Open**. The file runs, but the menu options are limited.

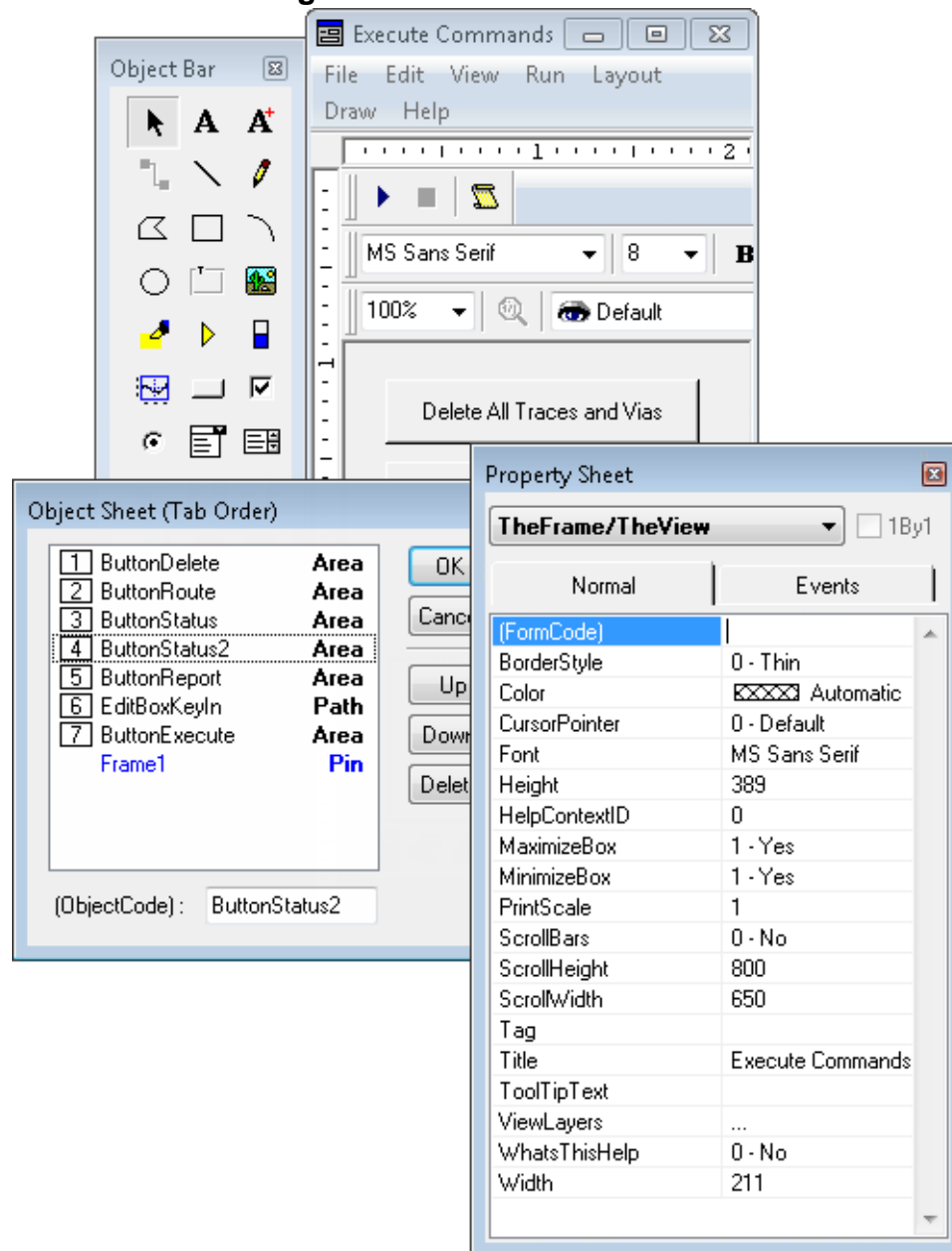
Edit Mode at a Glance

Edit mode consists of a full featured dialog box / view / forms.

Table 3-1. Edit Mode Components

Component	Where	Description
Format Bars	Top (moveable)	Property formatting toolbars.
Objects	Throughout	Objects and controls within the view.
Object Bar	Upper left (moveable)	Palette of available objects. Objects can be inserted by select & draw or by drag & drop.
Layout Bar	Bottom (moveable)	Layout operations on hyper-object.
Property Sheet	Lower right	Property editing for a particular object, set of objects, the view object or the frame object.
Object Sheet	Lower right	Setting of draw/tab order and as well as access to an object's Property Sheet.

Figure 3-1. View in Edit Mode



Format bars for easy access to property values:



Property formatting toolbar:



View Mode Overview

This is a quick overview of how a view works and how it can be used.

Table 3-2. View Elements

Element	Description
Objects	Windows controls (Common and Custom), ActiveX controls, OLE objects, and other lightweight graphical objects.
Properties	Control how objects look and behave. Usually set in edit mode, but can also be accessed by VBScript from run mode.
VBScript	Embedded VB compatible scripting engine can access objects in the form and pre-registered variables (including local variables).
Define Layout	By built-in Editor; defined while program running.
Member Variables	Bound to objects through Property Sheet (if used).
Printing	Printing support built in.

A view has two operating modes:

- **Edit Mode** — The user interface for the view is built in this mode. Objects can be added or deleted, and objects' properties can be edited.
- **Run Mode** — Most users only see this mode. Controls are created and data is connected to dynamic objects.

Setting Up Your Edit Environment

This section contains topics about modifying your environment to optimize it for your automation tasks.

Changing the Size of a Form	31
Ruler Bars	31
Grid Settings	32
Setting Snap to Grid in Active View (Edit Mode)	32
Changing the Layout Grid Size	32
Line Snapping	33
SnapToCenter	33
Testing a View	33
Returning to Edit Mode	34
Defining Mnemonics	34
Checking Mnemonics	34

Changing the Size of a Form

You can change the size of a form by setting its size property. Using the form's Property Sheet, you can also determine a form's language engine, color, border style, control menu buttons, title, and window placement.

Procedure

1. In the Form Editor, click and drag the bottom-right corner of the form to the required height and width.
2. Right-click on the form background and choose **Form Properties** from the popup menu. The Property Sheet opens. The new height and width are listed.
3. Close the Property Sheet (by clicking on the Close Window button) to set the new size.

Ruler Bars

Ruler Bars show the dimensions of an object in the Form Editor.

Choose **View > Ruler Bars** to toggle between viewing and hiding the ruler bars. When a check mark appears next to the Ruler Bars menu item, the ruler bars are displayed in the Form Editor.

Ruler bars run vertically, along the left side and horizontally, below the main menu bar at the top of the Form Editor.

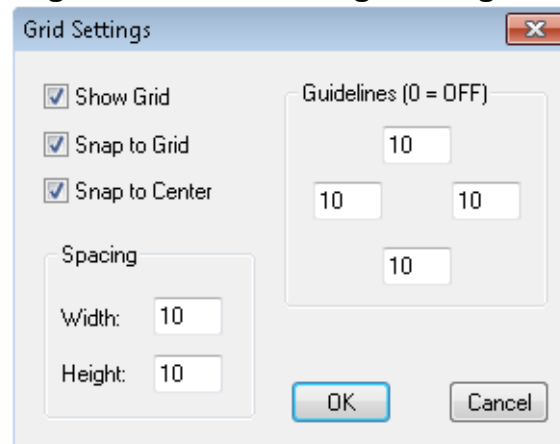
Dimensions may be displayed in either inches or centimeters, by right-clicking on the Ruler Bar and choosing the desired measurement units from the popup menu.

Grid Settings

When you are placing or arranging objects in a view, you can use the alignment grid for more precise positioning. When the grid is turned on, objects appear to "snap to" the dotted lines of the grid as if magnetized. You can turn this "snap to grid" feature on and off, and change the size of the grid cells.

In addition to defining a grid size, and whether to show, or snap to the grid, or Snap to Center, a guidelines margin can be defined around the perimeter of the view, such that objects can not be moved outside the guidelines margin.

Figure 3-2. Grid Settings Dialog Box



Setting Snap to Grid in Active View (Edit Mode)

You can turn the "snap to grid" feature on and off.

Procedure

1. From the menu bar, choose **Layout > Grid Settings**. The Grid Settings dialog box appears.
2. Select the Snap to Grid option.

Changing the Layout Grid Size

You can change the size of the grid cells.

Procedure

1. From the menu bar, choose **Layout > Grid Settings**. The Grid Settings dialog box appears.
2. Enter the height and width in pixels for the grid using the Spacing field. The grid spacing must be the same for both height and width.

Line Snapping

Objects can be permanently connected by line snapping. When they are moved, the lines (or polylines) are moved to maintain the connections. The endpoints of lines and polylines are "snap ties" which are matched up with "anchor points" in other objects. Built-in objects with anchor points include ellipse and (rectangular) border, text and bitmap, although other objects and controls can be made to contain anchor points by system integrators.

Objects with anchor points also have the "AnchorSnaps" property. An unlimited number of anchor points can be defined. Each is defined by a pixel offset (by default by 0,0) from one of eight base points around the perimeter and one base point in the middle of the object.

The checking of new line snapping connections can be turned on and off by the button on the Layout Bar. If turned off, existing connections are maintained, but new ones are not checked for. Line snapping is not performed for multi-selected or grouped objects.


SnapToCenter

New grid mode. Snap-to-center assumes that all objects and symbols moved around have been pre-sized and are smaller than the grid cell (convenient for flowcharting). To implement this, choose **Layout > Grid Settings** to open the Grid Settings dialog box. Select the Snap to Center option. Note that the Snap to Grid option must be selected to enable Snap to Center.

Testing a View

This editor is actually part of your program. By switching out of edit mode into run mode you can test the behavior of a view. This gives you immediate feedback on how the layout of objects appears and performs, and speeds up the user-interface design process.


Procedure

Choose **Run > Run**; or, click the **Run Form** icon  in the Standard toolbar. (Shortcut: **Ctrl+E** or **F5**)

Returning to Edit Mode

You can switch out of run mode and return to edit mode to resume editing the view.

Procedure

Choose **Run > Stop**; or, click the **Stop Form** icon  in the Standard toolbar (if available).

Defining Mnemonics

Normally keyboard users move the input focus from one control to another in a view with the Tab and arrow keys. However, you can define a mnemonic key that allows users to choose the control by pressing a specific key sequence (**Alt** + mnemonic key). To define a mnemonic key for a control with its own visible caption (push buttons, check boxes, and radio buttons) use the following procedure.

Procedure

1. Select the control and open the Property sheet.
2. In the Text property, type an ampersand (&) in front of the letter you want as the mnemonic for that control.

Results

An underline appears in the displayed caption to indicate the mnemonic key.

Checking Mnemonics

To verify that none of the controls in your view have mnemonic key conflict use the following procedure.

Procedure

1. Choose **Layout > Check Mnemonics**.
2. You receive warning of any conflicts via a message box, and are given the opportunity to select the group of objects which have the mnemonic key conflict.

Manipulating Objects

You can move, copy, group, and delete objects, as well as adjust their size, alignment, spacing and rotation angle.

Selecting a Single Object	35
Selecting Multiple Objects	36
Moving Objects	36
Moving an Object Within a View.....	36
Moving or Copying Objects Between Views	36
Deleting an Object	37
Copying and Pasting an Object	37
Setting the Dominant Object in a Selection.....	37
Aligning Objects.....	39
Spacing Objects	40
Resizing Objects.....	40
Sizing Individual Objects	41
Grouping and Ordering Objects	41
Rotation.....	43
Editing End Points.....	44
Hot Spots.....	44
Object Properties.....	45


Selecting a Single Object

You can select a single object in a view.

Procedure

1. In the Object Bar palette, choose the **Select** icon.
2. Point to the object you want to select and Left click. The object is selected, and displays green sizing handles.

Tip

 Use Tab to move forward or Shift-Tab to move backward through the objects in the view.


Selecting Multiple Objects

You can select multiple objects in a view.

Procedure

1. In the Object Bar palette, choose the **Select** icon.
2. Hold down the Left mouse button and drag to draw a selection box around the objects you want to select. Objects partially outside the selection box are not selected.
3. Release the Left mouse button; all objects inside the selection box are selected.

Tip

 Hold down the Shift key and click each object you want to remove from or add to the existing selection.

Moving Objects

You can use the following procedures to move one or more objects from one location to another in a view or from one view to another. If Snap to Grid is on, the object(s) snaps to the alignment grid when moved by the mouse.

For information on other ways to align multiple objects, see “[Aligning Objects](#)” on page 39.

Moving an Object Within a View

You can move an object within a view by dragging it or using the arrow keys.


Procedure

1. Select the object.
2. Drag the object to its new location.

-Or-

Use **Ctrl+Arrow** keys to move the object one grid unit at a time, or use the **Arrow** keys to move the object one pixel at a time

Note

 When moving objects one pixel at a time, Snap to Grid must be turned off.

Moving or Copying Objects Between Views

You can use Copy and Paste commands to move one or more objects between views.

Procedure

1. Select one or more objects.
2. Choose **Edit > Cut** or **Edit > Copy** in the original view.
3. Choose **Edit > Paste** to paste the objects in the new view.

Deleting an Object

You can delete an object from a view.

Procedure

1. Select the object.
2. Choose **Edit > Cut** or **Edit > Delete**.

Copying and Pasting an Object

You can copy and paste objects from the clipboard.

Procedure

1. Select the object.
2. To copy the object, choose the **Edit > Copy** menu item or **Ctrl+C**.
3. To paste the object, choose the **Edit > Paste** menu item or **Ctrl+V**.

Note



When you paste an object into a new view, it is placed in the same position as it was in the original view.

Setting the Dominant Object in a Selection

When you are resizing or aligning multiple objects, the Editor uses the *dominant object* to determine how the other objects are sized or aligned. By default, the dominant object is the first object selected. When multiple objects are selected, the dominant object has green sizing handles; all the other selected objects have aqua sizing handles.

Sizing handle colors may vary on different platforms, hardware or software.

Procedure

1. Select multiple objects.
2. Hold down the Ctrl key and click the object you want to make the dominant object.

Results

All further resizing or alignment is based on this object.

Aligning Objects

You can align objects along edges or on their centers.





Aligning Objects Along Edges	39
Aligning Objects On Their Centers	39
Centering Objects in the View	40

Aligning Objects Along Edges

Align objects along their left, right, top, or bottom edges.

Procedure

1. Select the objects you want to align.
2. Make sure the correct dominant object is selected. The final position of the group of objects depends on the position of the dominant object.
3. Choose one of the following icons on the Layout Bar:

Action	Description	Shortcut	Icon
Align Left	Aligns the selected objects along their left side.	Ctrl+Left Arrow	
Align Right	Aligns the selected objects along their right side.	Ctrl+Right Arrow	
Align Top	Aligns the selected objects along their top edges.	Ctrl+Up Arrow	
Align Bottom	Aligns the selected objects along their bottom edges.	Ctrl+Down Arrow	

Aligning Objects On Their Centers

Align objects vertically or horizontally on their centers.

Procedure



1. Select the objects you want to center.
2. Make sure the correct dominant object is selected. The final position of the group of objects depends on the position of the dominant object.
3. To align objects vertically on centers choose **Layout > Align Objects > Vert. Center** or to align objects horizontally on centers choose **Layout > Align Objects > Horz. Center**.

Centering Objects in the View

You can center objects vertically or horizontally in the view.

Procedure


1. Select the object or objects you want to center.
2. Choose one of the following icons on the Layout Bar:

Action	Description	Icon
Center Vertically	Centers the selected objects vertically in the form view.	
Center Horizontally	Centers the selected objects horizontally in the form view.	

Spacing Objects

You can space objects evenly either down or across.

Procedure

1. Select three or more objects you want to space.
2. To space objects evenly across, choose **Layout > Space Evenly > Across** or the **Space Across** icon  on the Layout Bar.

-Or-

To space objects evenly down, choose **Layout > Space Evenly > Down** or the **Space Down** icon  on the Layout Bar.

Resizing Objects


You can make objects the same size, height, or width.

Procedure


1. Select the objects you want to resize.
2. Choose one of the following icons on the Layout Bar:

Make Same Width 

Make Same Height 

Make Same Size 

Note

 When you are sizing or aligning multiple objects, the Editor uses the "dominant object" to determine how the other objects are sized or aligned. When multiple objects are selected, the dominant object has green sizing handles; all the other selected objects have aqua sizing handles. The dominant object is the first object selected. To change the dominant object press the Ctrl key while clicking the left mouse button.

All selected objects will be resized to match the selected dimension of the dominant object.

Sizing Individual Objects

Use the sizing handles to resize an object. When the mouse pointer is positioned on a sizing handle, it changes shape to indicate the direction in which the object will be resized. Active sizing handles are solid; if a sizing handle is hollow, the object cannot be resized along that axis.

For information on sizing multiple objects, see “[Resizing Objects](#)” on page 40.

When you change the size of an object, its final shape may be affected by whether you have Snap to Grid turned on.

Procedure

1. Click the object or select it with the Tab key.
2. Use the sizing handles to change the size of the object:
 - Sizing handles at the top and sides change the horizontal or vertical size.
 - Sizing handles at the corners change both horizontal and vertical size.
3. Or, use the Shift key plus the arrow keys to resize the object one pixel at a time, or the Ctrl+Shift keys plus the arrow keys to resize the object one grid unit at a time.


Grouping and Ordering Objects

Objects can be grouped so that they always are selected and moved together. The objects can also be ungrouped or regrouped (after ungrouping). Members of the same group can also share a set of group-specific properties. Multiple groups and objects can be combined to form a single group.

Procedure

1. Select more than one object.
2. Choose the **Draw > Group > Group** menu item.

Use the Group, Ungroup, or Regroup icons on the Layout Bar:

Group 

Ungroup 

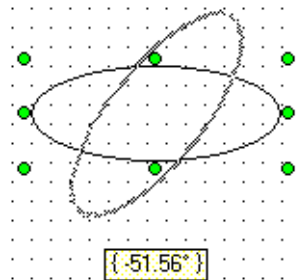
Regroup 

Rotation

Some built-in objects can be rotated. Additional objects can be added.

Some objects (but not controls) can be rotated. Built-in rotate-capable objects include line, polyline, ellipse and (rectangular) border, but system integrators may add in their own rotate-capable objects (or reduce the built-in set.) Besides being able to set the "RotateAngle" property directly in the **Property Sheet**, it also available in the Format Bar in a editable combo box which also contains rotations to 0, +90 and -90 degrees.

Figure 3-3. An Ellipse Undergoing Rotation



In free rotate mode (toggled by clicking the Free Rotate icon in the Format Bar), the drag handles of the selected object turn into circles, and the object can be rotated about its center by dragging any one of these handles. Free Rotate mode is exited by completing a rotation, by toggling the same icon or by making another object selection.

"Unrotate" sets the object back to 0 degrees. "Rotate Right" rotates the object clockwise by an additional 90 degrees.

Rotating an Object Using the Draw Menu Item 43

Rotating an Object Using the Format Bar Toolbar 44

Rotating an Object Using the Draw Menu Item

Use the Rotate item on the Draw menu to rotate an object clockwise in 90 degree increments or to free rotate.

Procedure

1. Select one object.
2. Choose the **Draw > Rotate > Free Rotate** menu item to free rotate the object.


-Or-

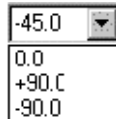
Choose the **Draw > Rotate > Unrotate** menu item to reset the object back to 0 degrees or choose the **Draw > Rotate > Rotate Right** menu item to rotate the object clockwise by 90 degrees.

Rotating an Object Using the Format Bar Toolbar

Use the Free Rotate or Rotate Angle controls in the Format Bar.

Procedure

1. To free rotate an object use the  Free Rotate icon.
2. To unrotate an object use the Rotate Angle combo box and choose 0.
3. To rotate an object a specific angle use the Rotate Angle combo box and enter the degrees into the field and press Enter.



Editing End Points

You can add or delete vertices.

Procedure

1. While in Edit Points Mode (which works in the same manner as MS Draw), hold down the Ctrl key and click the vertices to delete them. An X will appear over the vertex to be deleted.
2. Drop a vertex on another vertex to delete the dropped vertex.
3. To add a vertex, click on the line and drag the new vertex to the desired position.

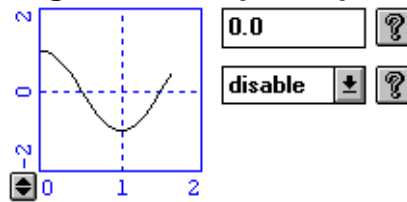
Hot Spots

Hot spots are a convenient way to add user interaction to objects (or controls) without going to the overhead of using Windows controls (standard, custom, or otherwise). With hot spots you can add clickable bitmap areas (the '?' help button in check box controls), or even Right mouse click short-cut menus.

Figure 3-4. Mouse Arrow Over a Help Hotspot.



Clickable bitmap areas can also be seen below in the edit, combo and graph objects.

Figure 3-5. Help Hotspots

Most objects with hotspots provide a property to turn the hotspot off.

Object Properties

Objects (as well as the view and frame) have properties to make them programmable. Properties are accessed by right-clicking on the object (or view) in edit mode.

All objects have a number of fundamental properties:


Table 3-3. Object Properties

Property	Description
(ObjectCode)	Identifies the object for convenient access.
Bottom	Bottom position of containing rectangle.
CursorPointer	Cursor pointer displayed while mouse pointer is over object.
Enable	Enable the object's hotspots and event handling, and, additionally, for a control, its state.
HelpContextID	Help context ID used if the Frame Object 's WhatsThisHelp property is set to Yes .
Left	Left position of containing rectangle.
Right	Right position of containing rectangle.
Tag	(VARIANT) stores user defined data.
ToolTipText	Tooltip text displayed while mouse pointer is over object.
Top	Top position of containing rectangle.
Visible	Determine whether an object or control is visible.

Tip

 A color of "Nil" represents the default Windows color. For example, setting a [Button Object](#)'s BackColor to "Nil" implies COLOR_BTNFACE and setting TextColor to "Nil" implies COLOR_BTNTEXT.

Tip

 A font with height 0 (nil) implies that the font should be adopted from the view object.

There are also a number of common properties that most objects implement. (draw-only) indicates that the properties are used with drawing objects only, and not controls.

Table 3-4. Draw-only Object Properties

Property	Description
BackColor	Background color. Nil represents transparent.
ForeColor	Foreground or Text color.
Font	Font: Name, Style & Size.
Layer	Layer where the object or control resides. Layers are defined in the Layers Sheet and may individually have their enabled or visibility state set.
AnchorSnaps	If the object has anchor points, then this property allows them to be added, deleted or edited. See Line Snapping .
RotateAngle	(draw-only) If the object can be rotated, this is its angle of rotation (between -180 and 180 degrees). See Rotation .

Table 3-5. Object Events

Events	Description
EventClick (or EventChange)	VBScript action callback procedure run when a user clicks over the object (or for a Windows control, when it changes value).
EventInitialize	VBScript action callback procedure run when view is opened to allow the object to further initialize itself.
EventPumpData	VBScript action callback procedure run when new data is pumped to the view.

There are also a number of optional events that may be linked in:

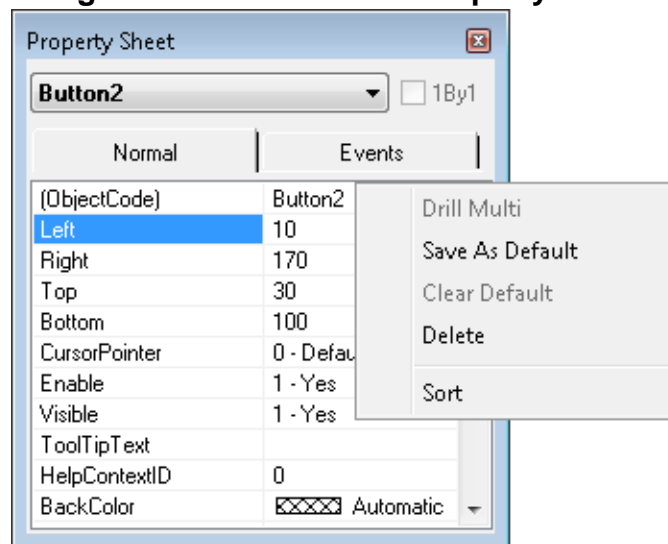
Table 3-6. Optional Object Events

Property	Description
EventDbClick	VBScript action callback procedure run when a user double-clicks over an object.
EventMouseDown	VBScript action callback procedure run when a user clicks the Left mouse button down over an object.
EventMouseMove	VBScript action callback procedure run when a user moves the mouse pointer over an object.

Table 3-6. Optional Object Events (cont.)

Property	Description
EventMouseUp	VBScript action callback procedure run when a user releases the Left mouse button up over an object.
EventDragOver	VBScript action callback procedure run when a user drags (text) over an object during drag & drop. You must set "Effect = 1" to enable EventDragDrop.
EventDragDrop	VBScript action callback procedure run when a user drops (text) over an object during drag & drop.

For example, here is a property sheet for the Window's push button control:

Figure 3-6. Push Button Property Sheet

The push button's properties are described in a following section. In general, you can edit properties values in line. Here are the actions tied to the property sheet's shortcut menu:

Table 3-7. Property Sheet Shortcuts

Item	Description
OK	Save the new property values.
Cancel	Abort (don't save) new property values.
Drill-Multi	multi-select only: if objects within the selection have different values for the same property, this action "drills" the dominant object's property values to be the universal value for that particular property for all of the object's in the selection.

Table 3-7. Property Sheet Shortcuts (cont.)

Item	Description
Save As Default	Set current property configuration as default when creating as a new object. The allows you to default appearance of the object when it is created from the Object Bar palette.
Delete	Delete the object.
Sort	Sort the property list in ascending order?

The topics that follow describe the properties defined for default objects. Note that the programmer (but not the user) can add new objects, with new, custom properties.

Frame and View Properties

Bringing up the properties dialog box with no objects selected allows the view and frame properties to be set.

Table 3-8. View Properties

View Properties	Description
(FormCode)	Form code. If entered, allows a form to be accessed by this handle using the FormFind() method.
Color	Background color of the view
CursorPointer	Cursor pointer displayed while mouse pointer is over object
Font	Default font for the view. If an object's font property is cleared, this font is used.
HelpContextID	Help context ID used if the frame object's WhatsThisHelp property is set to Yes
PrintScale	Default = 1.0. Changes the scaling of the view when printed.
ScrollBars	Make view scrollable.
ScrollHeight	View scroll size, bottom (scroll height).
ScrollWidth	View scroll size, right (scroll width).
Tag	(VARIANT) Stores user defined data.
ToolTipText	Tooltip text displayed while mouse pointer is over object.
ViewLayers	Definition of layers (there is one layer named Default by default).

Table 3-9. Frame Properties

Frame Properties	Description
BorderStyle	Frame border style: "thin" or "resizing". If this property is set to "thin" the window cannot be resized in run mode.
Height	Height of frame.
MaximizeBox	Frame style: display window Maximize button on Title bar.
MinimizeBox	Frame style: display window Minimize button on Title bar.
Title	Frame title.

Table 3-9. Frame Properties (cont.)

Frame Properties	Description
WhatsThisHelp	Puts the frame into "What's This Help" context sensitive help mode. The HelpContextID's defined in the Objects are used.
Width	Width of frame.

Tip

i A color of "Nil" represents the default Windows color. So, for example, setting a Button Object's BackColor to "Nil" implies COLOR_BTNFACE and setting ForeColor to "Nil" implies COLOR_BTNTEXT.

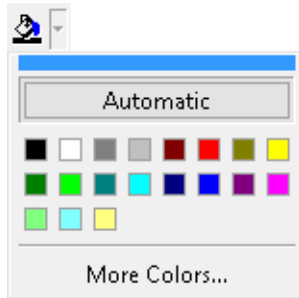
Tip

i A font with height 0 (nil) implies that the font should be adopted from the view object

More Colors

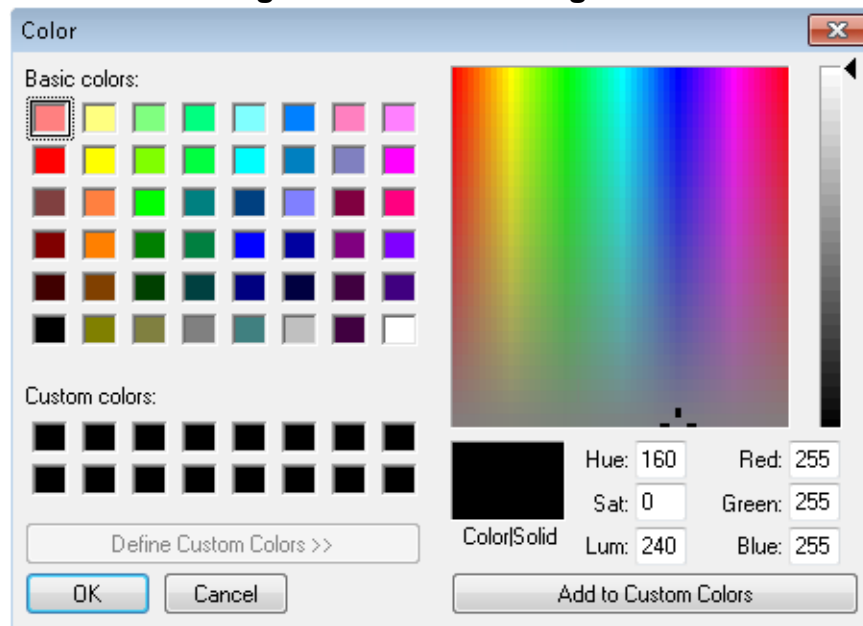
There are additional colors to choose from and these colors are saved with the view. To choose a color, click the down arrow icon for the fill icon on the Format Bar.

Figure 3-7. Color Palette.



For more colors, click on the More Colors option to activate the Color dialog box.

Figure 3-8. Color Dialog Box



Frame and View Events

Bringing up the properties dialog box with no objects selected and clicking the Events tab allows you to access and edit the view and frame event handlers.

Table 3-10. View Events

View Event	Description
EventClick	VBScript action callback procedure run when a user clicks over the object (or for a Windows control, when it changes value).
EventInitialize	VBScript action callback procedure run when the form first starts up (used for setup).
EventTerminate	VBScript action callback procedure run when the form first shuts down (used for cleanup).

There are also a number of optional events that may be linked in:

Table 3-11. Optional Events

Event	Description
EventDblClick	VBScript action callback procedure run when a user double-clicks over an object.
EventDragDrop	VBScript action callback procedure run when a user drops (text) over an object during drag & drop.

Table 3-11. Optional Events (cont.)

Event	Description
EventDragEnter	VBScript action callback procedure run when a user first enters a view while dragging during drag & drop.
EventDragOver	VBScript action callback procedure run when a user drags (text) over an object during drag & drop. You must set "Effect = 1" to enable EventDragDrop.
EventMouseDown	VBScript action callback procedure run when a user clicks the Left mouse button down over an object.
EventMouseMove	VBScript action callback procedure run when a user moves the mouse pointer over an object.
EventMouseUp	VBScript action callback procedure run when a user releases the Left mouse button up over an object.

Form Editor Menus

The Form Editor Window uses a standard menu format. The View, Layout, and Draw menus have dropdown submenus that appear when you select them.

File Menu	53
Edit Menu	53
View Menu	54
Run Menu	55
Layout Menu	55
Draw Menu	56

File Menu

The File menu controls document open and create operations, save options, and printing.

From the File menu, the commands in the pulldown menu are:

Menu Item	Description
New Ctrl+N	Not Defined
Open Ctrl+O	Not Defined
Save Ctrl+S	Allows you to save any changes that have been made to the open form.
Save As	Allows you to save the open form, with any changes that have been made, as a new form, with a different name.
Save Encrypted	Allows you to save any changes that have been made to the open form, in an encrypted file, which can only be opened with the password that is assigned to that file.
Print Ctrl+P	Allows you to print the open form on the designated printer.
Print Setup	Allows you to set printing preferences for the open form.
Close	Closes the open form.

Edit Menu

The Edit menu contains standard edit commands.

Edit menu commands include:

Menu Item	Description
Undo Ctrl+Z	Undo reverses the effect of each action in the Form Editor, from the last action to the first action. You can reverse to the first action the Form Editor performed.
Redo Ctrl+A	Redo reverses the effect of each Undo command, from the last Undo to the first Undo.
Cut Ctrl+X	Cut removes the selected items and retains them in a clipboard.
Copy Ctrl+C	Copy makes a duplicate of the selected items and retains them in the clipboard.
Paste Ctrl+V	Paste places the last items copied into the Form Editor window.
Paste Special	Not Defined
Delete Del	Delete deletes the selected items.

View Menu

From the View menu, the options in the pulldown menu are:

- Toolbars
- Status Bar
- Ruler Bars
- Snap Points

Choose **Status Bar**, **Ruler Bars** or **Snap Points** to toggle each feature on or off. Each item displays a check mark to its left when it is on.

When you choose **View > Toolbars**, the sub-menu displays:

- Standard
- Object Bar F2
- Font Bar
- Format Bar2
- Layout Bar Shift+F2

- Script Bar

Run Menu

The Run menu provides commands for running and stopping the current script form. These items perform the same functions as the Run Form and Stop Form icons on the Standard Toolbar.

The Run menu has the following commands:

Menu Item	Description	Shortcut
Run	Starts the current script	(F5)
Stop	Stops the running script	

Layout Menu

The Layout menu commands provide a means to organize and arrange items in the Form Editor.

The Layout menu also provides access to the Property Sheet for a selected object.

Menu Item	Description
Align Objects	Arranges selected objects according to the options listed in the sub-menu: Left , Right , Top , Bottom , Vertical Center , or Horizontal Center .
Space Evenly	Arranges selected objects evenly Across or Down .
Center in View	Aligns selected objects on a Vertical or Horizontal line in the view.
Make Same Size	Resizes selected objects to similar dimensions, based on Width , Height , or Both .
Objects Ctrl+I	Opens the Object Sheet dialog box, and allows arrangement of tabs.
Layers	Opens the Layers Sheet dialog box, that allows you to control layers.
Properties Ctrl+L	Displays the Property Sheet for the selected object.
Grid Settings	Opens the Grid Settings dialog box.
Check Mnemonics	Checks for duplicate Mnemonics (redundant key definitions).

Draw Menu

The Draw menu commands are used to manipulate objects in the Form Editor.

The Draw menu also controls the Zoom level within the view.

Menu Item	Sub-Menu Item	Description
Order	Bring to Front Ctrl+F	Moves selected items in front of other items.
	Send to Back Ctrl+B	Moves selected items behind other items.
	Bring Forward	Moves selected items one level forward.
	Send Backward	Moves selected items one level backward.
Zoom	Zoom In	Selects the next higher zoom level.
	Zoom Out	Selects the next lower zoom level.
	Zoom 1:1	Returns zoom level to 100%.
Group	Group	Group selected objects and groups.
	Ungroup	Ungroup selected group.
	Regroup	Regroup previous ungroup. Not available after an object is added or deleted.
Rotate	Free Rotate	Rotates an object about its center by dragging any handle.
	Unrotate	Resets the object back to 0 degrees.
	Rotate Right	Rotates the object clockwise by an additional 90 degrees.
Contour	Edit Points	Displays edit points, on a line, polyline, or freehand object, which can be added, deleted, or moved.
	Open/Close	Opens or closes the outline of a line, polyline, or freehand object.

Form Editor Toolbars

The Form Editor toolbars provide fast access to objects, layout, scripting tools, and property settings.

Standard Toolbar	58
Object Bar	59
Format Bars	61
Layout Bar	63
Script Bar	65




Standard Toolbar

To access: **View > Toolbars > Standard**

The Standard toolbar offers access to script controls.

Fields

Table 3-12. Standard Toolbar

Object	Icon	Description
Run Form(F5)		Runs the current script.
Stop Form		Stops the running script.
View Script		Toggles the view between Form Editor and Script Editor.

Object Bar

To access: **View > Toolbars > Object Bar**

New objects are accessible from the Object Bar palette. They can be inserted by select & draw or by drag & drop. You can show or hide the Object Bar palette. The built-in objects include:

Description

In addition to the built-in objects, developers can add their own custom objects and controls.

Figure 3-9. Adding, Deleting or Customizing ActiveX Controls




















The Object Bar palette is dynamic in that new ActiveX controls may be added or deleted from it, or have their entries customized. The ActiveX control configuration is restored when the application is restarted. Also, go to Frame and View Objects to see their descriptions.

Fields

Table 3-13. Object Bar

Field	Icon	Description
Select		Select tool.
Text		Text Object . Static text.
TextVar		TextVar Object . Dynamic text connected to variable.
Connector		Connector Object .
Line		Line Object .
Freehand		Freehand Object .
Polyline		Polyline Object .
Border		Border Object / Rectangular border.
Arc		Arc Object . Elliptical arc.
Ellipse		Ellipse Object .

Table 3-13. Object Bar (cont.)

Field	Icon	Description
Frame		Frame Object . Multi-purpose frame.
Bitmap		Bitmap Object .
Hilite		Hilite Object .
Pointer		Pointer Object . Pointer moves up-down or left-right.
Gauge		Gauge Object . Gauge moves up-down or left-right.
Graph		Graph Object . Graph with user definable X & Y.
Button		Button Object . Push button with pre-defined action.
CheckBox		CheckBox Object .
RadioButton		RadioButton Object .
ComboBox		ComboBox Object .
ListBox		ListBox Object .
EditBox		EditBox Object . Edit control for all variables.
MultiEditBox		MultiEditBox Object . Multi-line edit control for all variables.
Slider		Slider Object . Slider button common control.
Spinner		Spinner Object . Spin button common control.
ActiveX		ActiveX Object .
OLEObject		OLE Object .

Format Bars

To access: **View > Toolbars > Format Bar2**

The Format Bars offer fast access to the property values of selected object(s), and some extra functionality not available in the Layout Bar (to avoid overcrowding). You can show or hide the Format Bars. By default there are two Format Bars: Font Bar and Format Bar2.

Note


 The Format Bars are customizable. Click the right mouse button on them to bring up the Customize Toolbar dialog box. By holding down the **Ctrl** key and clicking **Reset**, the individual Format Bar is restored to its default state.

Figure 3-10. Format Bar1 (Font Bar)



Figure 3-11. Format Bar2



Fields

Table 3-14. Format Bar1 (Font Bar) Contents


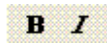



Object	Description
	Font name and size.
	Font bold and italic.
	Text (and general) left / center / right alignment.
	Background and Foreground colors.
	Line width, style and arrowhead style.

Table 3-15. <Format Bar2 Contents

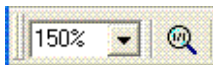
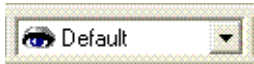


Object	Description
	Zoom percentage and zoom to 100%.
	Set default insertion layer.
	Rotation angle and Free Rotate mode (See rotation).

Table 3-15. <Format Bar2 Contents (cont.)

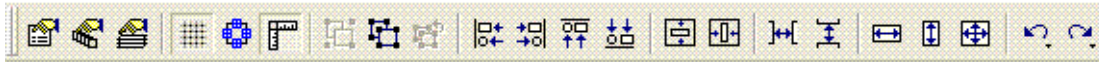
Object	Description
	Edit Polyline and Freehand points.

Layout Bar

To access: **View > Toolbars > Layout Bar**

The Layout Bar offers fast access to some important items from the Layout menu. You can show or hide the Layout Bar from the View menu.

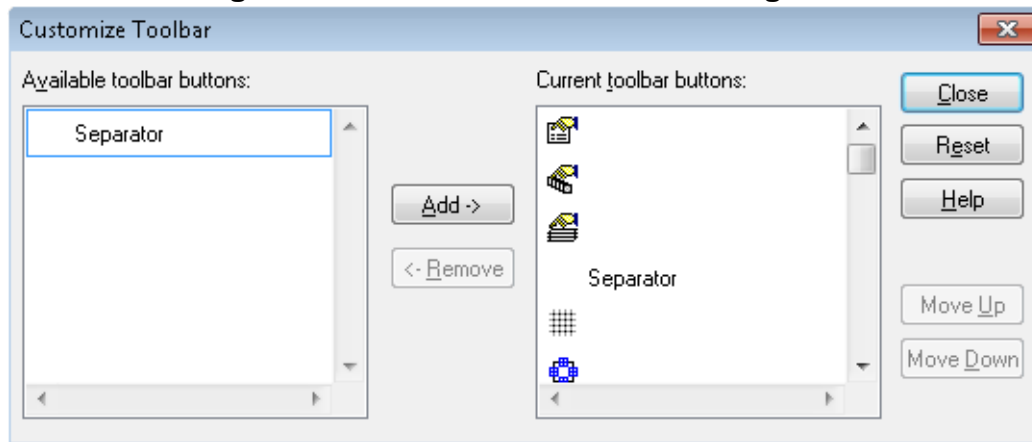
Figure 3-12. Layout Bar



Description

The Layout Bar is customizable. Right click on it to bring up the Customize Toolbar dialog box. By holding down the Ctrl key and clicking **Reset**, the Layout Bar is restored to its default state.

Figure 3-13. Customize Toolbar Dialog Box



Fields

Table 3-16. Layout Bar Contents























Object	Icon	Description	
Property Sheet		Opens the Property Sheet.	Enter
Object Sheet		Opens the Object Sheet (Tab Order) dialog box.	Ctrl+I
Layers Sheet		Opens the Layers Sheet dialog box.	
Toggle Grid		Turns grid on and off.	Ctrl+G
View Snap Points		Turns visibility of anchor / snap points on and off. Also, turn new snapping on and off.	
Toggle Ruler		Turns ruler bars on and off. Rulers measure pixels as (1 = 100).	

Table 3-16. Layout Bar Contents (cont.)

Object	Icon	Description	
Group		Group selected objects and groups.	
Ungroup		Ungroup selected group.	
Regroup		Regroup previous ungroup. This action is enabled until an object is added or deleted.	
Align Left		Aligns the selected objects along their left side.	
Align Right		Aligns the selected objects along their right side.	
Align Top		Aligns the selected objects along their top edges.	
Align Bottom		Aligns the selected objects along their bottom edges.	
Center Vertically		Centers objects vertically in the view.	
Center Horizontally		Centers objects horizontally in the view.	
Space Across		Spaces objects horizontally in the view.	
Space Down		Spaces objects vertically in the view.	
Make Same Width		Sizes objects to be the same width.	
Make Same Height		Sizes objects to be the same height.	
Make Same Size		Sizes objects to be the same size.	
Undo		Undo the last action.	
Redo		Redo the previously undone action.	

Script Bar

To access: **View > Toolbars > Script Bar**

The Script toolbar (Script Bar) is only displayed when in the Script Editor, and offers fast access to some important tools used for scripting.

Figure 3-14. Script Bar



Fields

Table 3-17. Script Bar Contents

Toolbar Item	Icon	Description
Members		Show list of member properties and methods.
Objects		Bring up list of objects.
Tooltip Info		Show Tooltip information.
Toggle Breakpoint		Toggles between showing and hiding breakpoints.
Clear Breakpoints		Clears the selected breakpoint.
Indent		Indents the selected text items.
Outdent		Removes indents from selected text items.
Comment		Adds comment to selected lines of text.
Uncomment		Uncomments selected lines of text.
Find		Find the specified Net, Net Class, or Part.
Repeat		Repeat the last action.
Replace		Replace specific text with different text.
Options		Set editor options.
Undo		Reverses last action.
Redo		Reverses last undo.

The Script Bar is customizable. Right-click on it to bring up the Customize Toolbar dialog box. By holding down the Ctrl key and clicking the Reset button, the Script Bar is restored to its default state.

Chapter 4

Built-in Objects

The Object Bar contains icons representing built-in objects that you can add to a form.

Text Object	68
TextVar Object	68
Connector Object	68
Line Object	69
Freehand Object	69
Polyline Object	69
Border Object	70
Arc Object	71
Ellipse Object	71
Frame Object	71
Bitmap Object	72
Hilite Object	73
Pointer Object	73
Gauge Object	74
Graph Object	74
Button Object	75
CheckBox Object	76
RadioButton Object	76
ComboBox Object	77
ListBox Object	78
EditBox Object	78
MultiEditBox Object	79
Slider Object	79
Spinner Object	80
ActiveX Object	80
OLE Object	81

Text Object

Static text object. It is derived from the border object, so it has all of the same properties as the border object, plus those shown in the table.

Static Text

Property	Description
Alignment	Text alignment within the rectangle: Left/Center/Right.
RotateAngle	Text rotation (horizontal or vertical). Note that some fonts can only be displayed horizontally (e.g., MS SANS SERIF).
Text	The actual text to display.

TextVar Object

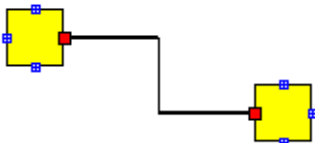
Dynamic text object. The "TextVar" object builds on the "Text" object by deleting the "Text" property, which is set automatically, and adding the two properties in the table.

Dynamic Text

Property	Description
Format	Follows 'C' language standards. Use only numeric portion, e.g., "%8.2lf" becomes "8.2", "%8d" becomes "8". One exception: for BOOL variables the "\$" string is converted into "NO"/"YES".
ValueEq	Equation (Simple or VBScript) attached to the object. The value is formatted and displayed at run time.

Connector Object

The connector object can only be used to make a connection between two anchor snaps (snap points). It is not possible to use a connector unless both ends are anchored to a snap point (the connector will auto-delete itself if both ends are not snapped).



Property	Description
Arrowhead	Arrowhead direction.

Property	Description
ArrowheadHeight	Arrowhead height in pixels.
PenStyle	Line style: solid, dotted, etc.
PenWidth	Line width in pixels

Line Object

Static line object.



Property	Description
Arrowhead	Arrowhead direction.
ArrowheadHeight	Arrowhead height in pixels.
PenStyle	Line style: solid, dotted, etc.
PenWidth	Line width in pixels.

Freehand Object

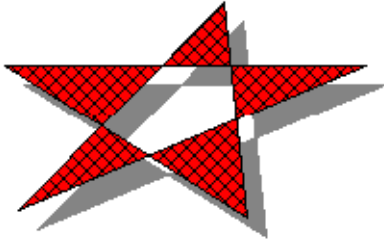
The freehand object can be used to create curved lines or freehand polyline shapes.



Property	Description
PenStyle	Line style: solid, dotted, etc.
PenWidth	Line width in pixels

Polyline Object


Static polyline object. If you perform Undo (^Z) after inserting a polyline object, instead of disappearing entirely, you will see a multi-select of line objects. Contains same properties as the Line object, plus those in the table.

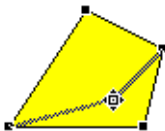


Property	Description
HatchStyle	Hatching pattern fill with.
PenWidth	Border width in pixels.
ShadowStyle	Shadow style: top/left, bottom/right or none.

Tip

i If you select a series of lines on a continuous contour in an ordered fashion, and "Group" them, you will instead get a polyline. Ctrl+Group overrides this behavior and results in a group.

Edit points mode : In this mode you can move the vertices of the polyline around, create new vertices by dragging lines, and delete vertices by dropping them on top of one another or by holding down the Ctrl key and clicking:



Border Object

Static line object.



Property	Description
BorderStyle	Normal, 3D, Sunken or Raised.
HatchStyle	Hatching pattern fill with.
HiliteColor	Used only for BorderStyle of Sunken or Raised.
PenWidth	Border width in pixels.
ShadowStyle	Shadow style: Top/Left, Bottom/Right, Bottom/Left, Top/Right, or None.

Arc Object

Static elliptical arc object. Angle1, Angle2 and IsWedge properties can be edited from the property sheet only. The arc is measured in degrees going counter-clockwise.



Property	Description
Angle1	Start angle, for example 0 degrees.
Angle2	End angle, for example -90 degrees.
IsWedge	Draw arc as a wedge?
PenStyle	Border line style: solid, dotted, etc.
PenWidth	Border width in pixels.

Ellipse Object

Static ellipse object.



Property	Description
HatchStyle	Hatching pattern fill with.
PenStyle	Border line style: solid, dotted, etc.
PenWidth	Border width in pixels.

Frame Object

2D/3D static frame object.



Property	Description
Alignment	Alignment of title within the frame: Left/Center/Right.
BorderColor	Border color.
BorderStyle	Frame border style: Normal, 3D, Sunken, Raised.
BorderWidth	Border width in pixels.

Property	Description
HiliteColor	3D highlighting color.
Text	Frame title.

Bitmap Object

Static bitmap object.

You can add a graphic image to a view.

Table 4-1. Bitmap Object Properties

Property	Description
Bitmap	Bitmap to display.
Layout	Alignment of the bitmap within the rectangle: Center, Left, Right.

The bitmap property has a special interface, described below:

Figure 4-1. Bitmap Dialog Box



Table 4-2. Bitmap Dialog Box Contents

Button	Description
OK	Save the new property value.
Cancel	Abort (don't save) new property value.
Copy	Copies the specified bitmap to the clipboard.
Paste	Allows a bitmap to be pasted in from the clipboard. "RLE" specifies whether the pasted bitmap is compressed when saved with the view.

Table 4-2. Bitmap Dialog Box Contents (cont.)

Button	Description
Load	Loads a bitmap from a file. "Link" saves only the directory path with the view. If unchecked, the entire bitmap file is saved in the view file.
Clear	Clears any bitmap specification.
Transparent	Specifies a color within the bitmap to be made transparent.

Hilite Object

Draws a semi-transparent yellow, magenta, or cyan rectangle to work as a highlighter when placed on top of other objects.

Hilite Object

Tip

- i** For more colors, group two hilite objects of different colors. For example, to create a green hilite object, group yellow and cyan together.

Property	Description
HiliteColor	Selects the color of the hilite object.

Pointer Object

Dynamic, movable pointer object. Works like a gauge/status bar.



Property	Description
MaximumEq	Maximum (Simple or VBScript) equation (-\$, \$, CONST).
MinimumEq	Minimum (Simple or VBScript) equation (-\$, \$, CONST).
ValueEq	Value equation (Simple or VBScript) (-\$, \$, CONST).

Tip

- i** Reversing rectangle in left/right or top/bottom directions reverses the direction the arrow points to. The relative dimensions (height to width) determines pointer direction.


Gauge Object

Dynamic gauge/status bar object. Has same properties as "Pointer" object plus those in the table.



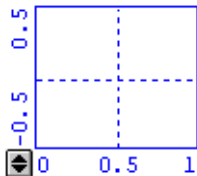
Property	Description
Font	Font of optionally displayed text.
ShowExtra	Display text representing percentage.

Tip

 Reversing rectangle in either the left/right or top/bottom direction reverses the direction of movement of the arrow.

Graph Object


Dynamic graph object. This is the most complex default object, and the only one that stores an array of data points. X/Y Points specify the data points filling the graph.



Property	Description
ClearAllPoint	Variable used to clear all data points in the graph (if TRUE).
Clockwise	If No, top of y-axis is negative and bottom is positive (reversed).
ConnectPoints	Connect lines between consecutive data points?
DataPointColor	Data point color.
EnablePoint	Variable used to enable addition of new data point. If var. is FALSE new data is disabled. (None) means always enabled.
PenWidth	Data point diameter in pixels.
XDivisions	Number of x divisions.
XMaxEq	X axis maximum equation (Simple or VBScript).
XMinEq	X axis minimum equation (Simple or VBScript).

Property	Description
XPointEq	X point equation (Simple or VBScript).
YDivisions	Number of y divisions.
YMaxEq	Y axis maximum equation (Simple or VBScript).
YMinEq	Y axis minimum equation (Simple or VBScript).
YPointEq	Y point equation (Simple or VBScript).

Tip

 Making "YMinEq" and "YMaxEq" constants allows the scaling spinner hotspot to be used to rescale the Y-axis.

Button Object

Push button Windows control.



Property	Description
Bitmap	Button bitmap (see Bitmap Object for property description)
ButtonShape	Normal, or property tab (Active or Inactive) shapes the button like property controls
ButtonType	Cancel = close window & do nothing, EventClick = run VBScript action callback, Goto = go to MDI associated file, HELP = invoke WinHelp() using "HelpContextID", OK = record changes & close window, Record = just record changes.
Default	(WINDOWS) is the control the default button?
GotoPath	(N/A for none) MDI associated file to go-to if the button is pressed. Mostly used to implement property sheet-like controls.
Layout	Bitmap's position within the button. Bitmap and text are automatically laid out basis on this value.
Text	Button text string

CheckBox Object

Check box button Windows control.



Property	Description
AlignTextLeft	Align text left?
Bitmap	ON bitmap. (see Bitmap Object for property description)
BitmapOffState	OFF bitmap. (see Bitmap Object for property description)
HelpHotButton	Show hotspot help button?
OwnerDrawn	T/F: If TRUE check box is owner drawn using bitmaps specified by "Bitmap" and "Bitmap (Off)" properties.
Text	Button text.
TriState	Make check box 3 state (T/F/ambiguous)?
ValueID	Value ID get/set by the control.

RadioButton Object

Radio button Windows control(s). This hyper-control specifies a set of radio buttons.

- ☐ **Dog**
- ☐ **Pig**
- ☐ **Moose**

Property	Description
AlignTextLeft	Align text left?
Bitmap	ON bitmap.
BitmapOffState	OFF bitmap.
ListItems	Defines set of radio buttons and associated values (see below)
ValueID	Value ID get/set by the control.

The "ListItems" property has a special interface, described below:

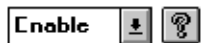
Figure 4-2. List Choices Dialog Box



This List Choices dialog box maintains a list of choices and their associated values. The hyper-variable specified by "Value ID" stores the associated value.

ComboBox Object

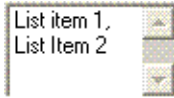
Combo box Windows control.



Property	Description
Accelerator	Windows mnemonic (accelerator) key.
ComboType	(WINDOWS) Only Droplist and Dropdown are supported by default.
HelpHotButton	Show hotspot help button?
ListItems	Defines set of radio buttons and associated values (see "Btn: Radio" Object)
NumDropped	Number of items in the control's drop list.
Sort	(WINDOWS) T/F: sort alphabetically?
UseColors	Use subsequent BackColor and TextColor properties (a little slow).
ValueID	Value ID get/set by the control.

ListBox Object

List box Windows control. The list box control currently only supports single selection. It operates in 2 modes: (1) copy selection from "List ID" into "Value ID" (2) move the selection.



Property	Description
Accelerator	Windows mnemonic (accelerator) key.
AddString	Adds the specified string as a single list item.
DeleteString	Removes the list item that corresponds to the specified index number (zero-based).
GetCurSel	Returns the zero-based index of the selected list item. Returns -1 if no item is selected.
HorizontalScroll	(WINDOWS) None/Always
ListID	List ID get/set by control. Should be in the format: "item 1\r\nitem 2\r\nitem N".
RemoveSelection	T/F remove selection from list box (List ID) when recorded.
Selection	Same as GetCurSel.
SetCurSel	Selects the list item that corresponds to the specified index number (zero-based).
Sort	(WINDOWS) T/F: sort alphabetically?
UseColors	Use subsequent BackColor and TextColor properties (a little slow).
UseTabstops	(WINDOWS) No/Yes
ValueID	Value ID get/set by control.
VerticalScroll	(WINDOWS) None/Always/Automatic
WantKeyInput	(WINDOWS) No/Yes

EditBox Object



Edit Windows control. Supports all types of variables.

Property	Description
Accelerator	Windows mnemonic (accelerator) key.
Alignment	(WINDOWS) Left/Center/Right
BorderDrawn	(WINDOWS) Draw border around edit control?
CaseOrPassword	(WINDOWS) None/Lower/Upper/Password
HelpHotButton	Show hotspot help button?
ReadOnly	(WINDOWS) Set edit control to read-only state
UseColors	Use subsequent BackColor and TextColor properties (a little slow).
ValueID	Value ID get/set by control.

MultiEditBox Object

Edit Windows control, but supports multi-line. Has all the properties of Edit Control, with the following additions:



Property	Description
HorizontalScroll	(WINDOWS) None/Always/Automatic
VerticalScroll	(WINDOWS) None/Always/Automatic

Slider Object

Slider Windows common controls.



Property	Description
Accelerator	Windows mnemonic (accelerator) key.
AutoTicks	(WINDOWS) display auto-ticks in run mode?
DisplayID	Display ID updated while the slider is being adjusted
HelpHotButton	Show hotspot help button?
MaximumEq	Maximum equation (Simple or VBScript)

Property	Description
MinimumEq	Minimum equation (Simple or VBScript)
Orientation	Horizontal, Vertical
TickFrequency	Number of tick intervals to display (2-100).
TickStyle	Both, Top/Left, Bottom/Right
ValueID	Value ID get/set by control.

Spinner Object

Spin button Windows common controls.



Property	Description
Accelerator	Windows mnemonic (accelerator) key.
DecimalBase	(WINDOWS) decimal/hexadecimal (NO =>Hex).
EditBuddy	Tie to previous edit control in tab order?
IncrAccelerator	Increment size.
MaximumEq	Maximum equation (Simple or VBScript).
MinimumEq	Minimum equation (Simple or VBScript)
Orientation	Horizontal, Vertical
ValueID	Value ID get/set by control.
WrapAround	Wrap around when decrementing below minimum or incrementing above maximum.

ActiveX Object

ActiveX objects and properties may vary, depending on the hardware and software platform being used.



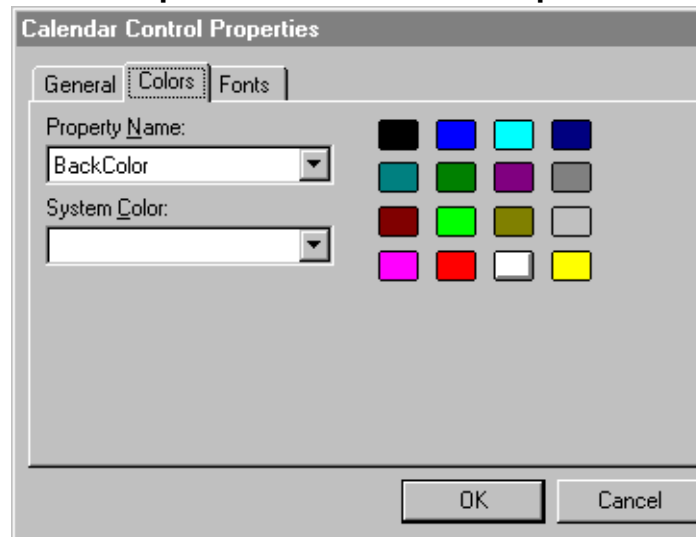
Sun	Mon	Tue	Wed	Thu	Fri	Sat
30	31	1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	1	2	3
4	5	6	7	8	9	10

ActiveX windows control. Allows any well-behaved ActiveX control to be inserted at run time, for its properties to be setup, and for event handlers to be connected.

Property	Description
(OCX Properties)	Brings up the ActiveX's properties dialog box (if implemented) which controls visual aspects of the ActiveX control.
Event...	Dynamically generated VBScript event handlers. The events listed are read in from the ActiveX control.

"(OCXProperties)" handles the ActiveX's properties dialog box. For example, this is for the MS Access Calendar control:

Figure 4-3. Example Calendar Control Properties Dialog Box




OLE Object

OLE 2.0 linked or embedded object.

Standard Option	\$1,495
Add 1 Yr Subscription	\$400
Extended Option	\$1,895

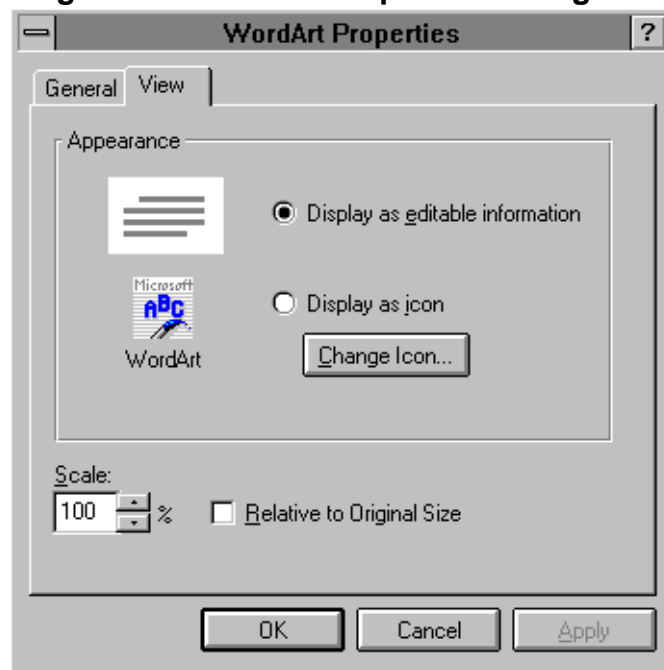
Note

 OLE objects and properties may vary, depending on the hardware and software platform being used.

Property	Description
BackColor	Background color.
DoVerbs	Enable/Disable do verbs in run mode. "Do verbs" is the list of operations exported by the OLE object itself. The verbs are available in run mode by right-clicking on the OLE object or from the Edit menu.
OLE Properties	Brings up standard OLE properties dialog box. Note changes made to this property can not be Canceled or Applied from the Property Sheet (changes can be applied from the OLE property dialog box).

"(OLEProperties)" handles the standard OLE 2.0 properties dialog box:

Figure 4-4. OLE 2.0 Properties Dialog Box



OLE objects have some extra special handling. You can "Paste Special" and directly paste an OLE 2.0 in edit mode. You can select an OLE object in run mode and run its verbs by right clicking and selecting from a popup menu.

Third-Party Information

For third-party information, refer to [*Third-Party Software*](#).

End-User License Agreement with EDA Software Supplemental Terms

Use of software (including any updates) and/or hardware is subject to the End-User License Agreement together with the Mentor Graphics EDA Software Supplement Terms. You can view and print a copy of this agreement at:

mentor.com/eula

