What is ActiveBar?

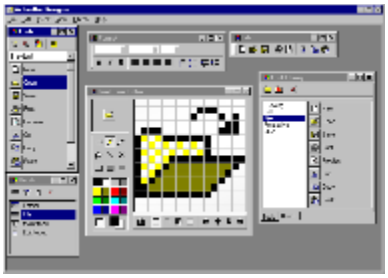
Description

With ActiveBar™ any application or web site you create can sport cool toolbars and menus in mere minutes. If you liked the flexibility Office97 and Internet Explorer offers, wait until you test drive ActiveBar.

ActiveBar™ is an ActiveX control that provides dockable toolbars and menus with full runtime customization. Advanced UI features will boost your applications look and feel, make them easier to use and give your end users the power to reconfigure toolbars and menus.

All this and more come in a lightweight ActiveX package that offers easy distribution and high performance. ActiveBar™ lets you effortlessly add a state-of-the-art user interface to your existing and future projects.

Create menu and toolbar layouts in seconds using ActiveBar Designer. The integrated tool library provides a repository of commonly used tools and bands for reuse.
Adding a new tool is a snap, just drag&drop tools from the library or to create a new one just fill in the property sheet and use the icon editor.
Visual Menu Builder lets you create application menus and popup menus on the fly.



Features

1. Homogenous menu and toolbar architecture.
2. Multiple layout styles (Docking, Floating, Popup).
3. Flexible runtime behavior (enable/disable docking, floating, moving and resizing).
4. Detachable menus & toolbars.
5. Tabbed toolbars for organizing large number of commands in categories.
6. Sliding tabs.
7. Over 100 properties, methods, and events.
8. Automatic rendering of icon states: Clicked, Enabled, Disabled.
9. Multistate buttons, Group buttons, & Separators.
10. Horizontal, vertical and floating menus.
11. Alignment and position control over text and icon elements.
12. Built-in support for Combobox, Edit, and DropDown controls.
13. Custom icon sizes and text styles.
14. ToolTips, keyboard shortcuts & grab handles.
15. Intelligent resizing & wrapping.
16. Truly lightweight ActiveX control (less than 300k).
17. Requires no external DLL's for easier deployment and eliminates potential DLL conflicts.
18. **Visual menu builder** allows you to quickly design or modify tools and bands, using a drag & drop interface.
19. **Integrated tool library** serves as a repository for your commonly used tools and bands, for example, the predefined standard File, Edit, Window, Help tools.
20. **Advanced icon editor** includes an **icon image capture tool** and state of the art drawing tools.
21. ActiveBar seamlessly attaches to both your MDI and standard forms to fully integrate toolbars & menus with your applications.
22. Allows context switching of menus and toolbars depending on the your applications program flow. You can easily save and restore toolbar layout information as you move between different application states.

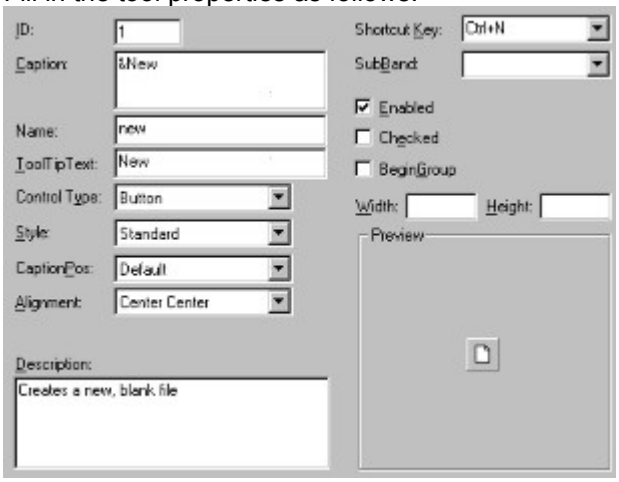
23. Users can easily customize and reconfigure their toolbars and menus to create a more productive layout to suite their specific needs.
24. ActiveBar has an open architecture with extensible COM interfaces, which will carry your applications well into the future.

QuickTour

Creating a simple toolbar at design time

ActiveBar makes it very easy to create toolbars and menus using its designer. The designer is available in your development environment by selecting Designer from the control's context menu. Here are the steps to create a simple toolbar using the ActiveBar designer.

1. Start by adding ActiveBar control to your project.
2. Select the ActiveBar Icon and place the ActiveBar on your form.
3. Right-Click on the icon  to bring up the context menu
4. Select Designer from the menu
5. The Designer window and the tools and bands windows are opened.
6. First we create a category for our new toolbar. (optional)
7. Click on the categories icon  to bring up the "Manage Categories"
8. Type "Category 1" in the name text box and click add.
9. Select "Category 1" from the categories drop down in the tools window.
10. Click the new tool icon.
11. Fill in the tool properties as follows:



12. You can draw an icon or capture it using the icon capture tool  .
13. Close the tool properties window.
14. Click New Band in the Bands Window
15. Double-click on the Band1 list item to bring up the band window.
16. Drag "New" tool and drop it in the Band1 window.
17. Right-Click inside the band for a context menu and select Band Properties.
18. Set the band properties as follows:
    Caption = "First Toolbar"
19. Close "First Toolbar " band.

20. Close the designer.
21. Run your project.

Creating a toolbar at runtime

While the designer is quickest way to create tools and tool bands for your application, ActiveBar gives you complete control to create and modify your toolbars at runtime using its properties and methods.

Below is the code listing of creating toolbars with inline comments explaining the process.

```
Dim tool As New tool

' Add the standard toolbar
barDemo.Bands.Add "Standard"

' Set wrapping to true (Default is False)
barDemo.Bands("Standard").Wrappable = True

With barDemo.Bands("Standard").Tools Do
' Add First Tool with id 101 and Name tiNew
      Set tool =.Add(101, "tiNew")
      ' Set the caption with an accelerator key
      tool.Caption = "&New"
      ' Load the normal picture (index 0) with a gray mask color
      tool.SetPicture _
            Index:=0, _
            Picture:=LoadPicture("new.bmp"), _
            color:=QBColor(8)
' No separator line before this icon
      tool.BeginGroup = False
      ' Display style as a standard button.
      tool.Style = DDTSStandard     ' 0

      ' Repeat for the second tool, ….
      Set tool =.Add(102, "tiOpen")
      tool.Caption = "&Open"
      tool.SetPicture _
            Index:=0, _
            Picture:=LoadPicture("open.bmp"), _
            color:=QBColor(8)
      tool.BeginGroup = False
      tool.Style = 0
End With

' Force the toolbar to recalculate its layout and refresh
barDemo.RecalcLayout
```

ActiveBar Designer

The ActiveBar designer provides a complete environment for designing bands, icons and managing tool libraries and layouts. A default library with standard icons (default.bli) comes with ActiveBar.

The following is a description of the designer window and its menu options.

File

Open Layout

Opens a previously saved layout file. Layout files store tools, bands and their properties for archiving or loading at runtime using ActiveBar's Load Method.

Loading a layout destroys current tools and bands, make sure you save your work if you do not want it destroyed.

Save Layout

Saves current set of tools and bands to a disk file.

Exit

Saves all changes closes the designer window.

Edit

Delete (Del)

Deletes current tool or band from the active tool or band window.

Properties (F4)

Displays current tool or band property window.

Insert

Tool

Adds a new tool to the ActiveBar tools collection. Adds a new tool item to the tools windows within the current selected category and displays the tool property dialog box.

Band

Adds a new band to the ActiveBar bands collection. Adds a new band Band1 item to the bands window.

View

Tools

Opens the tools window.

Bands

Opens the bands window.

Library

Opens the tools library window.

Image

Load Picture

Enabled for the icon editor window, allows you to load a picture from file.

Options

Enabled for the icon editor window, allows you to set the size of the icon in pixels.

Library

New Category

Creates a new category for the tools in the library.

Delete Category

Deletes current category from the tools library.

Delete Tool

Deletes current tool from the tools library.

Help

Contents

Displays this windows help file.

About ActiveBar

Displays the program's Information dialog box.

Tools Window

The tools window displays all tools in the ActiveBar tools collection by their category. You can add new tools, modify tool properties, delete tools, drag tools from this window and drop them on a band, page, or into the tools library. Dragging tools from the window into other windows creates a copy of the tool and inserts it into the dropped window tools collection.

 New tool. Add a new tool to the window. Displays new tool properties dialog box.

 Delete tool. Deletes current tool from the window and removes it from the collection.

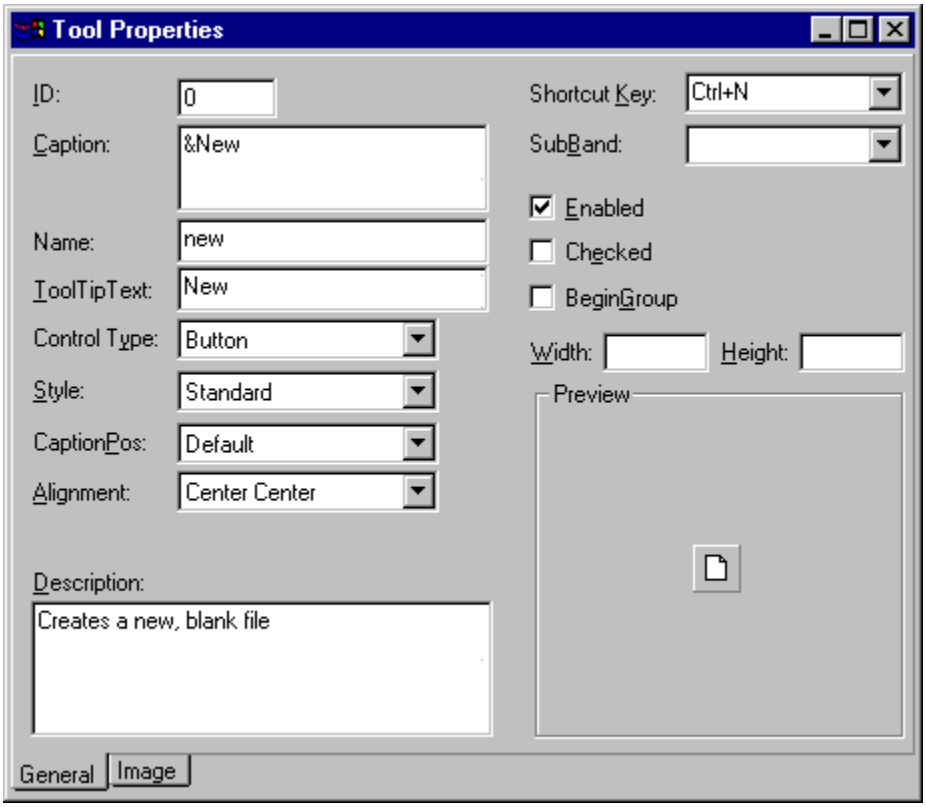
 Properties. Displays tool properties and icon editor dialog box.

 Manage categories. Allows you to add and delete tool categories.

Category selection. New tools are added into the selected category.

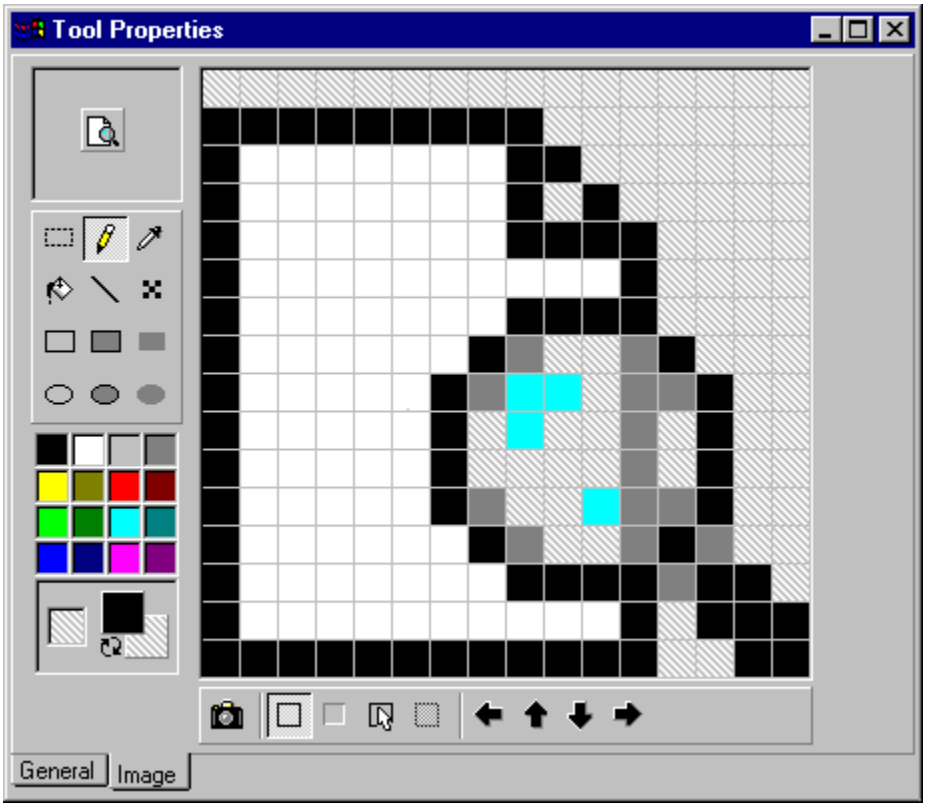
Tool Properties

The properties dialog box allows you to modify tool display and state properties. It is called from the tools window or the bands window.



Icon Designer

The icon designer displays current tool icon and provides editing tools to create or modify the image.



Icon Preview
Displays a real size view of the current icon.

Selection Tool
Selects a rectangular area within the icon editor.

Draw Tool

Draws pixels using the current foreground color.

Eye Dropper Tool
Picks the color at the specified pixel and selects it as the foreground color.

Fill Tool
Fills a contiguous area of a certain color with the current selected foreground color.

Line Tool

Draws a line between two points. Holding the Ctrl key while dragging the mouse draws a straight horizontal line. Holding the Alt key while dragging the mouse draws a straight vertical line.

Pattern Tool
Draws a 3x3 pixel pattern using the selected foreground color. Holding the control key switched the pattern from odd pixels to even.

Rectangle Tool
Draw a rectangular shape with a border of the selected foreground color.

Bordered Filled Rectangle Tool
Draws a rectangular shape filled with the selected background color and has a one-pixel border of the selected foreground color.

Filled Rectangle Tool
Draws a rectangular shape filled with the selected background color.

Ellipse Tool
Draw an ellipse shape with a border of the selected foreground color.

Bordered Filled Ellipse Tool
Draws an ellipse shape filled with the selected background color and has a one-pixel border of the selected foreground color.

Filled Ellipse Tool
Draws an ellipse shape filled with the selected background color.

Color Picker
Selects the current foreground color.

Background Foreground
Displays current foreground and background colors. Clicking on the round arrow icon  switches the colors around.

Icon Capture Tool

Minimizes the designer window and allows you to capture any portion of your screen with the exact size of your current icon. A zoom window follows around displaying a zoomed view of the to-be-captured area.

Image Selection

Each tool has four different icons that can be designed using the icon editor. Clicking on each of the icons saves the current icon and switches to the clicked one. The icon images are used as follows:

Normal  - This is the main icon displayed with the tool. ActiveBar can automatically generate the remaining states from this icons if they were not available.

Pressed

 - Displayed when the mouse is clicked on the tool (optional).

Mouse-Hover

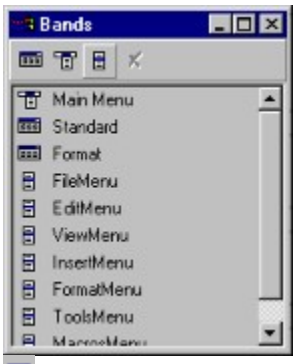
 - Displayed when the mouse hovers over the tool (optional).

Disabled

 - Displayed when the icon is disabled (Enabled = False) (optional).

Shift Tools
Used to move the image in the specified direction one pixel at a time.

Bands Window



New toolbar band - Creates a new band of type DDBTNormal and adds it to the ActiveBar bands collection. Double clicking on a band opens it up for editing and the new band is a drop target for tools from the tools and tool library windows.

New menu bar band - Creates a new band of type DDBTMenuBar and adds it to the ActiveBar bands collection.

New popup menu band - Creates a new band of type DDBTPopup and adds it to the ActiveBar bands collection.

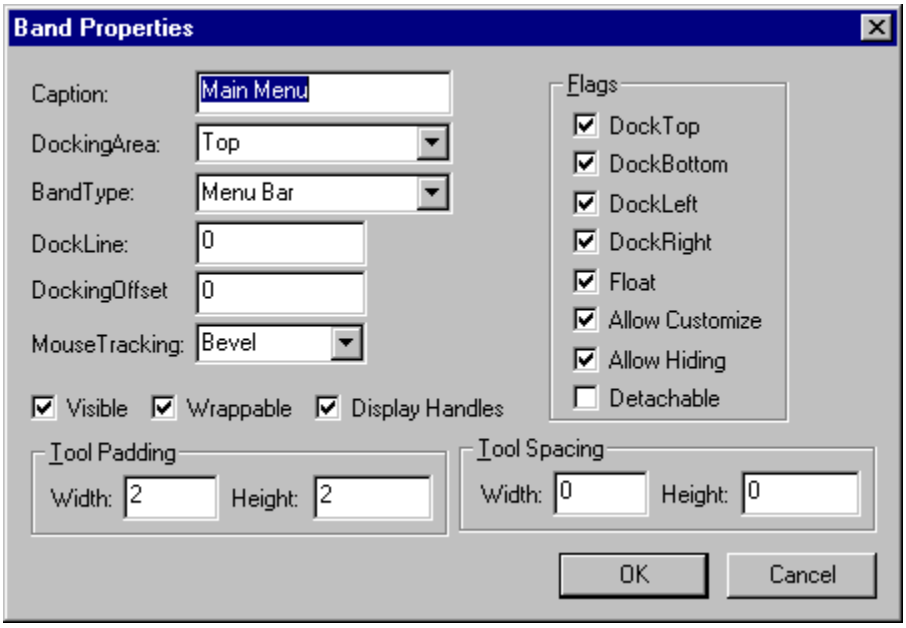
Delete band - deletes current band from the bands collection.

Band Window Menu

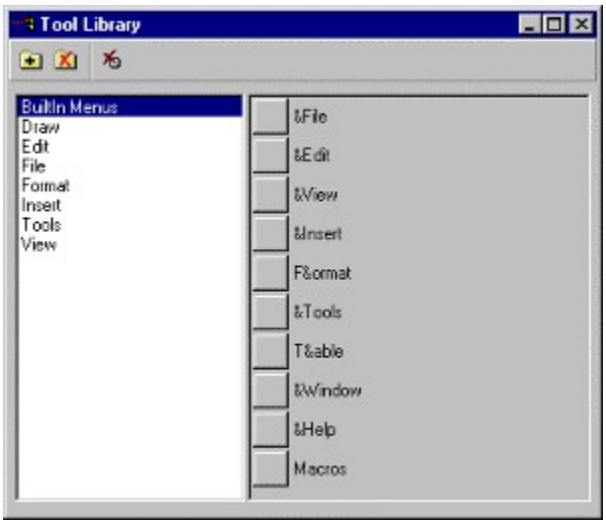
Edit Tool - Displays the current tool property dialog box.
Delete - Deletes a tool from the band.
Delete All - Deletes all tools from the band.
Add Page - Adds a new page to the band to create tabbed toolbars.
Band Properties - Displays the band properties dialog box.

Band Properties

Bands properties dialog box allows you to set band properties.



Library Window

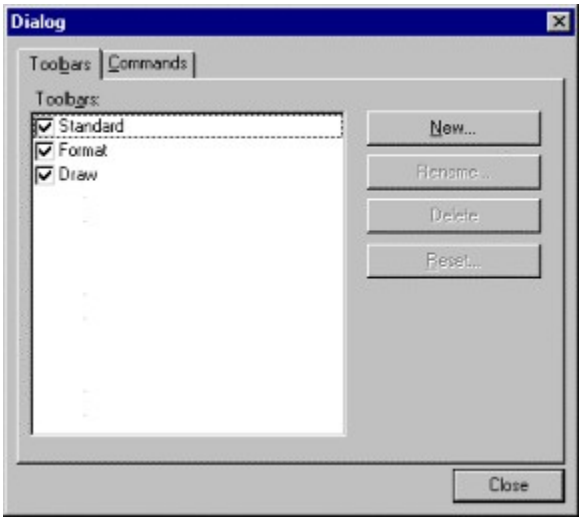


The library window allows you to keep categorized tools that are used frequently such as standard menus, toolbars, … You can add any tool to the tools window or an open band by dragging it from the library window.

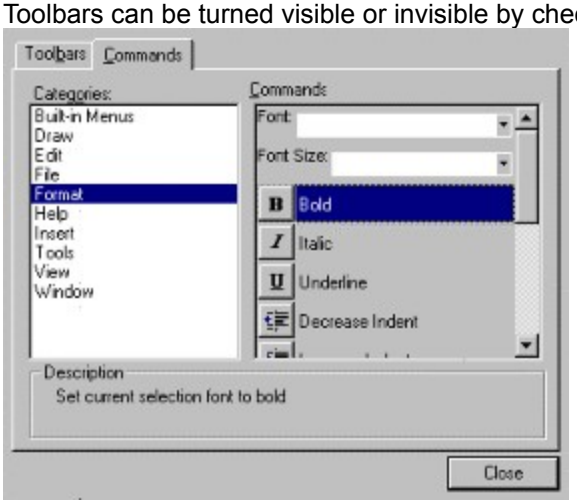
ActiveBar Runtime

ActiveBar run-time behavior can be customized through methods, properties and customization dialog box. The Customize dialog box is enabled by default for the user to show, hide, create or modify toolbars at runtime.

Customize



The toolbars page allows the user to create new toolbars, rename or delete toolbars. New.. and Reset.. fire ActiveBar events so you can display your custom dialog for adding a new toolbar. Reset allows you to re-load the original layout from a saved layout file.
Toolbars can be turned visible or invisible by checking the checkbox in the toolbars list.

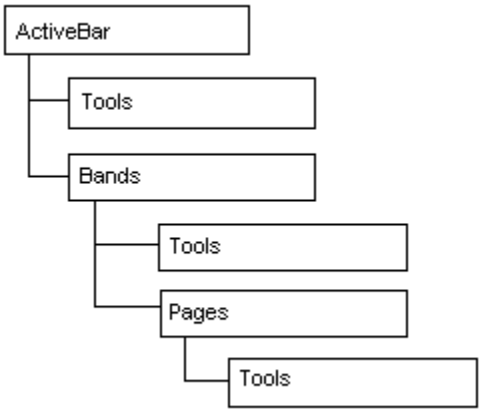


The categories and commands page allows the end-user to drag tools from the commands list and drop them on any open toolbar (standard or custom). Description displays a brief description of the tool as specified in the tool's property sheet at design time.

Resetting the toolbar from a saved layout.

You can keep a default layout of the toolbars in your application in a saved layout file and use the file to reset when your application starts or on the Reset event which is fired when the user clicks Reset.. in the Customize dialog box. Layout files are created at design-time in the designer window or at run-time using the save method. Layouts can be restored at run-time using the ActiveBar Load method.

Quick Reference



ActiveBar Architecture

ActiveBar contains three main internal objects that are needed to create all the different types of menus and toolbars. The Tool object, Band object and Page object. Bands and Pages are container objects that have Tools collection properties similar to the ActiveBar object.

The main tools collection in the ActiveBar object (ActiveBar.Tools) is the main tools collection and it contains all the tools used in the bar. When new tools are added to the tools window in designer, they are added to this collection. This collection is used to display the tools list in the customization window at run-time.

The band and page tools collections include every tool that appears in a band or a page. When a new tool is dropped into a band or page, ActiveBar creates a copy of the tool object and adds it to the respective band or page tools collection. The new copy can be customized separately from the original tool from which it was copied. Changing any of the properties does not effect the original tool even if the new tool had the same name and tool id. There is one exception to this rule. The Enabled and Checked properties which are called State Properties are changed across all tools collection for each tool with the same id or name. This is helpful when updating the tool's state at run time. For example, setting a Bold tool Checked to True will change both the toolbar and the menu item if they had the same id number.

Constants

Band Flags

1	DDBFDockLeft
2	DDBFDockTop
4	DDBFDockRight
8	DDBFDockBottom
16	DDBFFloat
32	DDBFCustomize
64	DDBFHide
128	DDBFDetach

Band Types

0	DDBTNormal
1	DDBTMenuBar
2	DDBTPopup

Color Depth

4	DD16Colors
8	DD256Colors
16	DD16BitColors
24	DD24BitColors

Caption Positions

0	DDCPStandard
1	DDCPLeft
2	DDCPRight
3	DDCPAbove
4	DDCPBelow
5	DDCPCenter

Docking Area

1	DDDATop
2	DDDABottom
4	DDDALeft
8	DDDARight
16	DDDAFloat
32	DDDAPopup

PageStyles

0	DDPSNone
1	DDPSTopTabs
2	DDPSBottomTabs

ToolAlignment

0	DDALeftTop
1	DDACenterTop
2	DDARightTop
3	DDALeftCenter
4	DDACenterCenter
5	DDARightCenter
6	DDALeftBottom
7	DDACenterBottom
8	DDARightBottom

ToolStyles

0	DDSStandard
1	DDSText
2	DDSIcon
3	DDSIconText

ToolTypes

0	DDTTButton
1	DDTTButtonDropDown
2	DDTTComboBox
3	DDTTEdit
4	DDTTStatic

TrackingStyle

0	DDTSNone
1	DDTSBevel
2	DDTSColor

Control Reference

{button ,JI(`ActiveBar.HLP',`ActiveBar_Events')} ActiveBar Events

{button ,JI(`ActiveBar.HLP',`ActiveBar_Methods')} ActiveBar Methods

{button ,JI(`ActiveBar.HLP',`ActiveBar_Properties')} ActiveBar Properties

{button ,JI(`ActiveBar.HLP',`Bands_Methods')} Band Methods

{button ,JI(`ActiveBar.HLP',`Band_Properties')} Band Properties

{button ,JI(`ActiveBar.HLP',`CBList_Methods')} CBList Methods

{button ,JI(`ActiveBar.HLP',`Tools_Methods')} Tools Collection Methods

{button ,JI(`ActiveBar.HLP',`Tool_Methods')} Tool Methods

{button ,JI(`ActiveBar.HLP',`Tool_Properties')} Tool Properties

ActiveBar Properties

ActiveBand	Band	Sets or returns the active band in the bands collection
Bands	Collection	Collection of all bands in the toolbar
ColorDepth	Integer	Sets or returns the number of colors used to save an icon's image.
DataPath	String	Starts an asynchrosous download of a layout file
DisplayKeysInTooltip	Boolean	Determines if shortcut keys will be displayed in the tool's ToolTip
DisplayToolTips	Boolean	Determines if ToolTips should be displayed when the mouse cursor hovers over the tool.
Font	Font	Sets or returns the font used in displaying text in toolbars and menus.
Tools	Collection	Collection of all tool objects in the toolbar

ActiveBand

Description

Returns a Band object representing the current active band. Read only.

Data Type

Band object

Syntax

```
<oBand> = <oActiveBar>.ActiveBand
```

oBand	Returned band object
oActiveBar	Valid reference to an active bar control

Example

```
‘ Hide the current band’s grab handles
oCurrBand = ActiveBar1.ActiveBand
oCurrBand.DisplayHandles = False
```

Bands

Description

Returns the collection of bands in the ActiveBar object.

Data Type

Collection

Syntax

```
<colBands> = <oActiveBar>.Bands
```

oActiveBar	Valid reference to an active bar control
colBands	A reference to the bands colloection

Example

```
‘ Show all bands
For Each oBand in ActiveBar1.Bands
oBand.Visible = True
Next
```

ColorDepth

Description

Sets or returns the number of color bits used to save and display the icon image. Once set, subsequent saved images will use that value.

Data Type

Integer

Syntax

```
<oActiveBar>.ColorDepth = <iColorDepth>
```

oActiveBar	Valid reference to an active bar control
iColorDepth	Color depth enum

Example

```
ActiveBar1.ColorDepth = 4 ' 16 colors
ActiveBar1.Save "MyLayout.Lyt", ""
```

DataPath

Description

Starts asynchronous download of a bands layout file. Used in web browsers. Runtime only.

Data Type

String

Syntax

<oActiveBar>.DataPath = <sDataPath>

oActiveBar	Valid reference to an active bar control
sDataPath	Layout file URL

Example

```
bar.DataPath = "http://www.datadynamics.com/ barlayout.lyt"
```

DisplayKeysInTooltips

Description

When set to True, the ActiveBar object will display shortcut key names in the tooltip. This property is valid when DisplayTooltips is also True.

Data Type

Boolean

Syntax

```
<oActiveBar>.DisplayKeysInTooltips = <bFlag>
```

oActiveBar	Valid reference to an active bar control
bFlag	New DisplayKeysInTooltips setting.

Example

```
ActiveBar1.DisplayTooltips = True
ActiveBar1.DisplayKeysInTooltips = True
```

DisplayTooltips

Description

Determines whether Active Bar should display yellow instructional tooltips when the user hovers the mouse over the tool. Default is True.

Data Type

Boolean

Syntax

```
<oActiveBar>.DisplayTooltips = <bFlag>
```

oActiveBar	Valid reference to an active bar control
bFlag	New DisplayKeysInTooltips setting.

Example

```
ActiveBar1.DisplayTooltips = True
ActiveBar1.DisplayKeysInTooltips = True
```

Font

Description

Sets or returns the current font and properties used to display text tool captions. You can access the name and style of the font properties using this object reference.

Data Type

StandardFont

Syntax

```
<oActiveBar>.Font.<FontProperty> = <Setting>
```

oActiveBar	Valid reference to an active bar control
FontProperty	Any valid font property (Name, Bold, …)
Setting	A valid setting for the specified property

Example

```
ActiveBar1.Font.Name = “Arial”
ActiveBar1.Font.Size = 11
ActiveBar1.Font.Italic = True
```

Tools

Description

Returns a collection of all tools used in the specified ActiveBar control.

Data Type

Collection

Syntax

<colTools> = <oActiveBar>.Tools

oActiveBar	Valid reference to an active bar control
colTools	A returned collection of tools

Example

```
For Each oTool In ActiveBar1.Tools
oTool.Alignment = DDTACenterBottom
Next
```

ActiveBar Methods

AboutBox	Displays ActiveBar release information dialog.
Attach	Forces the ActiveBar to initialize and attach itself to the current form. `<oActiveBar>.Attach`
Customize	Displays runtime customize dialog. `<oActiveBar>.Customize`
Detach	Detaches any previously attached from the current form <oActiveBar>.Detach
GetLayoutData	Saves layout data to a variant variable `<vLayout> = <oActiveBar>.GetLayoutData(<oBand>)`
Load	Loads saved layout from a file `Load(<sFileName>, <sBandName>)`
OnKeyDown	Passthrough a form keydown event to ActiveBar `<bRetVal> = OnKeyDown(<iKeyCode>, <iShift>)`
OnKeyUp	Passthrough a form keydown event to ActiveBar `<bRetVal> = OnKeyUp(<iKeyCode>, <iShift>)`
RecalcLayout	Refreshes toolbars display and layout after modifying any of the properties at runtime. `<oActiveBar>.RecalcLayout`
ReleaseFocus	Releases focus from the toolbar `<oActiveBar>.ReleaseFocus`
Save	Saves current layout to a file `Save(<sFileName>, <sBandName>)`
SetLayoutData	Saves layout data into a variant variable `SetLayoutData(<sBandName>, <vSave>)`

AboutBox

Description

Displays ActiveBar's release information dialog box.

Attach

Description

Attaches ActiveBar to current form. This method is needed only in ActiveX containers that do not initialize the children ActiveX controls, such as Borland Delphi. You will need to detach every attached ActiveBar control before unloading the form. This method is not needed if you are using Microsoft Visual Basic.

Syntax

```
<oActiveBar>.Attach
```

Example

```
{Delphi:  form create event}
ActiveBar1.Attach();
```

Customize

Description

Displays an interactive customization dialog to the end user. The user can drag and drop tools from different tool categories to toolbars and menus.

Syntax

```
<oActiveBar>.Customize
```

Example

```
Private Sub cmdCustomizeBar_Click()
ActiveBar1.Customize
End Sub
```

Detach

Description

Detaches previously attached ActiveBar controls.

Syntax

```
<oActiveBar>.Detach
```

Example

```
{Delphi: form destory event}
ActiveBar1.Detach();
```

GetLayoutData

Description

Saves layout information to a variant variable. This method in combination with SetLayoutData can be used to how swap bands and toolbars based on current form context requirements. Passing an empty string for the band name argument saves all band layouts at once otherwise it saves the specified band.

Syntax

```
<vLayout> = GetLayoutData(<sBandName>)
```

Example

```
Private Sub Form_Activate()
gv_FormBandLayout = GetLayoutData("FormBand")
End Sub
```

Load

Description

Loads band layout information from a file. This method in combination with the Save method can be used to save user layout modification to disk or reset user modification to a system default layout. Passing an empty string for the band name argument loads all saved layouts in the file, otherwise it loads the specified band.

Syntax

```
<oActiveBar>.Load(<sFileName>, <sBandName>)
```

Example

```
Sub cmdReset_Click()
ActiveBar1.Load(App.Path & "\MYAPP.LYT")
End Sub
```

OnKeyDown

Description

Simulates a KeyDown event on the ActiveBar control. Since ActiveBar is a window-less control it cannot receive events on its own. You can simulate shortcut keys by setting you form's PreviewKeys property to True and programming the KeyUp and KeyDown events to call the OnKeyDown and OnKeyUp events in ActiveBar.

Syntax

```
<bHandled> = <oActiveBar>.OnKeyDown(<iKeyCode>, <iShift>)
```

Example

```
Private Sub Form_KeyDown(KeyCode, Shift)
bHandled = ActiveBar1.OnKeyDown
If bHandled Then
      KeyCode = 0
      Shift = 0
End If
End Sub
```

OnKeyUp

Description

Simulates a KeyUp event on the ActiveBar control. Since ActiveBar is a window-less control it cannot receive events on its own. You can simulate shortcut keys by setting you form's PreviewKeys property to True and programming the KeyUp and KeyDown events to call the OnKeyDown and OnKeyUp events in ActiveBar.

Syntax

```
<bHandled> = <oActiveBar>.OnKeyUp(<iKeyCode>, <iShift>)
```

Example

```
Private Sub Form_KeyUp(KeyCode, Shift)
Dim bHandled As Boolean
bHandled = ActiveBar1.OnKeyUp(KeyCode, Shift)
If bHandled Then
      KeyCode = 0
      Shift = 0
End If
End Sub
```

RecalcLayout

Description

Recalculates the bands (toolbars and menus) layout. This method is used to refresh the layout after a series of property changes, which affect the layout of the control, such as, width and height. This speeds up the display and reduces the continuous recalculation of new layouts.

Syntax

```
<oActiveBar>.RecalcLayout
```

Example

```
For Each oTool in ActiveBar1.Tools
oTool.Height = 32
Next
ActiveBar1.RefreshLayout
```

ReleaseFocus

Description

Release focus from the ActiveBar object.

Syntax

```
<oActiveBar>.ReleaseFocus
```

Example

```
ActiveBar1.ReleaseFocus
```

Save

Description

Saves current bands layout to a specified file. This method in combination with the Load method can be used to restore user preferences or reset layout to a system default layout. Passing an empty string for the band name argument saves all bands layouts otherwise it saves the specified band only.

Syntax

```
<oActiveBar>.Save(<sFileName>, <sBandName>)
```

Example

```
ActiveBar1.Save(App.Path & "MYAPP.LYT", "FileBand")
```

SetLayoutData

Description

Loads layout information from a previously saved variant variable. This methods in combination with the GetLayoutData method can be used hot swap bands based on current form context. Passing an empty string for the band name argument loads all bands in the variable, otherwise it loads the specified band layout.

Syntax

```
SetLayoutData <sBandName>, <vLayout>
```

Example

```
Private Sub Form_Activate()
SetLayoutData "", gv_FormLayout
End Sub
```

ActiveBar Events

BandClose	When a toolbar band is closed or hidden.
BandMove	When a toolbar band is moved across the screen.
BandOpen	When a toolbar band is opened.
BandResize	When a floating toolbar band window is resized
Click	When a tool button is clicked
ComboDrop	When a combobox tool is opened
ComboSelChange	When a combobox item selection changes
DataReady	When asynchronous data download is complete.
DblClick	When a toolbar tool is double clicked
Error	When an internal ActiveBar error occurs
GotFocus	When a tool gets focus
LostFocus	When a tool loses focus
NewToolbar	When a new toolbar is created by the user in Customize dialog box.
PreSysMenu	
Reset	When the user selects Reset in the Customize dialog box.
TextChange	When text changes in either a textbox or a combobox tool.

BandClose

Description

Triggered when user closes a band by either de-selecting it from the context list of bands or closing it while floating. The event is also fired when the visible property of a band is set to False.

Syntax

```
Private Sub BandClose(ByVal Band As Band)
```

Example

```
Private Sub BandClose(ByVal Band As Band)
‘ If the Band was docked
If Band.DockingArea And DDDAFloat <> DDDAFloat Then
      ‘ Adjust the children sizes
      ‘ Resize Controls on the form
End If
End Sub
```

BandMove

Description

Triggered when user moves a band by changing its docking level, docking position or floating position.

Syntax

```
Private Sub BandMove(ByVal Band As Band)
```

Example

```
Private Sub BandMove(ByVal Band As Band)
‘ If the Band was docked
If Band.DockingArea And DDDAFloat <> DDDAFloat Then
      ‘ Adjust the children sizes
      ‘ Resize Controls on the form
End If
End Sub
```

BandOpen

Description

Triggered when user opens a band by either selecting it from the context list of bands. The event is also fired when the visible property of a band is set to True.

Syntax

```
Private Sub BandOpen(ByVal Band As Band)
```

Example

```
Private Sub BandOpen(ByVal Band As Band)
‘ If the Band was docked
If Band.DockingArea And DDDAFloat <> DDDAFloat Then
      ‘ Adjust the children sizes
      ‘ Resize Controls on the form
End If
End Sub
```

BandResize

Description

Triggered when user resizes a band in docking or floating position.

Syntax

```
Private Sub BandResize(ByVal Band As Band)
```

Example

```
Private Sub BandResize(ByVal Band As Band)
‘ If the Band was docked
If Band.DockingArea And DDDAFloat <> DDDAFloat Then
      ‘ Adjust the children sizes
      ‘ Resize Controls on the form
End If
End Sub
```

Click

Description

Triggered when user presses primary mouse button on a tool or a menu option. This event is central to all ActiveBar action processing.

Syntax

```
Private Sub Click(ByVal Tool As Tool)
```

Example

```
Private Sub ActiveBar1_Click(ByVal Tool As Tool)
      Select Case Tool.Name
      Case "New"
      NewFile     ' Create a new document
Case "Open"
      OpenFile    ' Open an existing file
Case "Save"
      SaveFile    ' Save File
Case "Print"
      PrintFile   ' Print current open file
End Select
End Sub
```

ComboDrop

Description

Triggered when user clicks the drop-down arrow of a combo-box tool.

Syntax

```
Private Sub ComboDrop(ByVal Tool As Tool)
```

Example

```
Private Sub ActiveBar1_ComboDrop(ByVal Tool As Tool)
' Fill the list with values
Tool.CBList.AddItem "Item 1"
Tool.CBList.AddItem "Item 2"
End Sub
```

ComboSelChange

Description

Triggered when the current item selection is changed in a combobox tool.

Syntax

```
Private Sub ComboSelChange(ByVal Tool As Tool)
```

Example

```
Private Sub ActiveBar1_ComboSelChange(Tool As Tool)
If Tool.Name = "cboZoom" Then
      ZoomView(Tool.CBListIndex)
      End If
End Sub
```

DataReady

Description

Triggered when an asynchronous data download from datapath is complete.

Syntax

```
Private Sub DataReady()
```

Example

```
Private Sub ActiveBar1_DataReady()
activebar1.recalclayout
end sub
```

DblClick

Description

Triggered when double-clicking the primary mouse button on a tool.

Syntax

```
Private Sub DblClick(Tool As Tool)
```

Example

```
Private Sub ActiveBar1_DblClick(Tool As Tool)
If Tool.Name = "Borders" Then
' Open the borders band
End If
End Sub
```

Error

Description

Triggered when an internal error occurs in ActiveBar

Syntax

```
Private Sub Error( ByVal Number As Integer,
      ByVal Description As ReturnString,
      ByVal Scode As Long,
      ByVal Source As String,
      ByVal HelpFile As String,
      ByVal HelpContext As Long,
      ByVal CancelDisplay As ReturnBool)
```

GotFocus

Description

Triggered when a tool gets focus in ActiveBar

Syntax

```
Private Sub GotFocus(ByVal Tool As Tool)
```

Example

```
Private Sub ActiveBar1_GotFocus(ByVal Tool As Tool)
If Tool.Name = "txtURL" Then
      Tool.Text = ""
End If
End Sub
```

LostFocus

Description

Triggered when a tool loses focus in ActiveBar

Syntax

```
Private Sub LostFocus(Tool As Tool)
```

Example

```
Private Sub ActiveBar1_LostFocus(Tool As Tool)
If Tool.Name = "txtURL" Then
      ' Validate URL
      End If
End Sub
```

NewToolbar

Description

Triggered when user presses New Toolbar in the customize dialog. This event can be used to display a custom or localized toolbar name form. Returning an empty string for the Name argument displays the built-in dialog box.

Syntax

```
Private Sub NewToolbar(ByVal Name As ReturnString)
```

Example

```
Private Sub ActiveBar1_NewToolbar(ByVal Name As ReturnString)
Name = MyGetToolBarNameDialog()
End Sub
```

PreSysMenu

Description

Triggered in MDI parent forms when user clicks on an MDI child system menu or right-clicks on the toolbars area. You can use this event to modify or localize the system menu before it is displayed. MDI system menus have the caption "SysMDI" and the customize popup menu has the caption "SysCustomize" the caption property can be checked before customizing the band.

Syntax

```
Private Sub PreSysMenu(ByVal Band As Band)
```

Example

```
Private Sub ActiveBar1_PreSysMenu(ByVal Band As Band)
Dim oTool As Tool
If Band.Caption = "SysMDI" Then
      ' Add a close all command to the MDI system menu
      oTool = Band.Tools.Add "CloseAll"
      oTool.Caption = "Close &All"
End If
End Sub
```

Reset

Description

Triggered when pressing the Reset command in the customize dialog box or executing Reset method on the ActiveBar control. You can load the original band layouts from persistent layout files of your application.

Syntax

```
Private Sub Reset(ByVal BandName As String)
```

Example

```
Private Sub ActiveBar1_Reset(ByVal BandName As String)
ActiveBar1.Load  App.Path & "\MYAPP.LYT", BandName
End Sub
```

TextChange

Description

Triggered when text changes in text or combo box tool types.

Syntax

```
Private Sub TextChange(ByVal Tool As Tool)
```

Example

```
Private Sub ActiveBar1_TextChange(ByVal Tool As Tool)
If Tool.Name = "txtURL" Then
      '
End If
End Sub
```

Band Properties

Caption	String	Sets or returns band caption, which is displayed as the window title when the band is floating.
CreatedBy	Integer	Sets or returns an enum of the origin of a band. Origin can be System, Application, or User.
CurrentPage	Integer	Sets or return the index of the current page in a multi-page band pages collection.
CurrentTool	Integer	Returns the index of the current active tool in the band's Tools collection.
DisplayHandles	Boolean	Turns grab handles display on or off
DockingArea	Integer	Sets or returns the current docking area of the band
DockingOffset	Long	Sets or returns the docking offset.
DockLine	Integer	Sets or returns the docking line (level) within the band's docking area
Flags	Long	Sets or returns the bands Docking and Customization flags
GrabHandleStyle	Integer	Sets or returns the display style of grab handles. Applicable when DisplayGrabHandles is True.
Height	Long	Sets or returns the height of the band in pixels
Left	Long	Set or returns the left coordinate position of a floating band in pixels
MouseTracking	Integer	Sets or returns the tool display style when the mouse moves over it
Pages	Collection	Returns the collection of pages in the band
PageStyle	Integer	Sets or returns the tabs style for the band's pages
Tools	Collection	Returns the tools collection
ToolsHPadding	Long	Sets or returns the horizontal padding of tools in pixels.
ToolsHSpacing	Long	Sets or returns the horizontal spacing between tools in pixels.
ToolsVPadding	Long	Sets or returns the vertical padding of tools in pixels.
ToolsVSpacing	Long	Sets or returns the vertical spacing between tools in pixels.
Top	Long	Sets or returns the top coordinate of a floating band in pixels.
Type	Integer	Sets or returns the type of a band (Normal, Menubar, Popup)
Visible	Boolean	Hides or shows the band
Width	Long	Sets the width of a band in pixels
Wrappable	Boolean	Determines if the band should wrap its tools when it is sized below the sum width of all it tools.

Caption

Description

Sets or returns band caption, which is displayed as the window title when the band is floating.

Data Type

String

Syntax

```
<oBand>.Caption [= <sCaption>]
```

Example

```
ActiveBar1.Bands(0).Caption = "Main Toolbar"
```

CreatedBy

Description

Sets or returns an enum of the origin of a band. Origin can be System, Application, or User. This property can be used to limit end user's access to some bands context menus and customization.

Data Type

Integer

Syntax

```
<oBand>.CreatedBy [= <iCreatedBy>]
```

Example

```
' Save User defined bands to file
For Each oBand In ActiveBar1.Bands
If oBand.CreatedBy  = 1 Then  ' User Defined Band
      ActiveBar1.Save App.Path & _
                  "\BANDS" & Str(I) & ".LYT", _
                  oBand.Name
End If
Next
```

CurrentPage

Description

Sets or return the index of the current page in a multi-page band pages collection.

Data Type

Integer

Syntax

```
<oBand>.CurrentPage [= <iPage>]
```

Example

```
ActiveBar1.Bands("Toolbox").CurrentPage = 2 ' Set to Draw Page
```

CurrentTool

Description

Returns the index of the current active tool in the band's Tools collection.

Data Type

Integer

Syntax

```
<oBand>.CurrentTool
```

Example

```
If ActiveBar1.Bands(ActiveBar1.CurrentTool).Name = "txtURL"
' Change status bar message
End If
```

DisplayHandles

Description

Turns grab handles display on or off

Data Type

Boolean

Syntax

```
<oBand>.DisplayHandles [= <bDisplayHandles>]
```

Example

```
ActiveBar1.Bands("MainMenu").DisplayHandles = False
```

DockingArea

Description

Sets or returns the current docking area of the band

Data Type

Integer

Syntax

```
<oBand>.DockingArea [= <iDockingArea>]
```

Example

```
ActiveBar1.Bands(0).DockingArea = DDDAFloat
```

DockingOffset

Description

Sets or returns the docking offset.

Data Type

Long

Syntax

```
<oBand>.DockingOffset [= <lDockingOffset>]
```

Example

```
ActiveBar1.Bands(0).DockingOffset = 100 ' offset = 100 pixels
```

DockLine

Description

Sets or returns the docking line (level) within the band's docking area

Data Type

Integer

Syntax

```
<oBand>.DockLine [= <iDockLine>]
```

Example

```
ActiveBar1.Bands(0).DockLine = 1
```

Flags

Description

Sets the bands Docking and Customization flags. Determines the level of run-time control the end user will have to move, detach or dock the band.

Data Type

Long

Syntax

```
<oBand>.Flags [= <lFlags>]
```

Example

```
ActiveBar1.Bands(0).Flags = ActiveBar1.Bands(0).Flags Or DDBFCustomize
```

GrabHandleStyle

Description

Sets or returns the display style of a band's grab handles.

Data Type

Integer

Syntax

```
<oBand>.GrabHandleStyle [= <iGrabHandleStyle>]
```

Example

```
For Each oBand in ActiveBar1.Bands
oBand.GrabHandleStyle = 1
Next
```

Height

Description

Sets or returns the height of a band in pixels. The coordinate properties apply to the band when it is floating. They will not be set to the exact values supplied, ActiveBar will auto size the band's layout for best fit.

Data Type

Long

Syntax

```
<oBand>.Height [= <lHeight>]
```

Example

```
ActiveBar1.Bands(0).Height = 100
```

Left

Description

Sets or returns the left coordinate of a floating band in pixels. The coordinate properties apply to the band when it is floating. They will not be set to the exact values supplied, ActiveBar will auto size the band's layout for best fit.

Data Type

Long

Syntax

```
<oBand>.Left [= <lLeft>]
```

Example

```
ActiveBar1.Bands(0).Left = 100
```

MouseTracking

Description

Sets or returns the tool display style when the mouse moves over it

Data Type

Integer

Syntax

```
<oBand>.MouseTracking [= <iMouseTracking>]
```

Example

```
ActiveBar1.Bands("Format").MouseTracking = DDTSColor
```

Pages

Description

Returns a collection of pages in a multi-page band.

Data Type

Collection

Syntax

```
<oBand>.Pages
```

Example

```
With ActiveBar1.Bands("Toolbox") Do
For i = 0 To.Pages.Count -1
.Pages(i).Caption = aToolboxCaptions(i)
Next
End With
```

PageStyle

Description

Sets or returns the tabs style for the band's pages

Data Type

Integer

Syntax

```
<oBand>.PageStyle [= <iPageStyle>]
```

Example

```
ActiveBar1.Bands("Toolbox").PageStyle = DDPSBottomTabs
ActiveBar1.Bands().PageStyle = DDPSSliders
```

Tools

Description

Returns the tools collection in a band

Data Type

Collection

Syntax

```
<oBand>.Tools
```

Example

```
For Each oTool in ActiveBar1.Bands(0).Tools
' set oTool properties
Next
```

ToolsHPadding, ToolsVPadding

Description

Sets or returns the horizontal or vertical padding of tools in pixels. Padding is the number of transparent border pixels surrounding the actual tool image.

Data Type

Long

Syntax

```
<oBand>.ToolsHPadding [= <lPadding>]
<oBand>.ToolsVPadding [= <lPadding>]
```

Example

```
ActiveBar1.Bands("Format").ToolsHPadding = 20
ActiveBar1.Bands("Format").ToolVPadding  = 20
```

ToolsHSpacing, ToolsVSpacing

Description

Sets or returns the horizontal spacing between tools in pixels. Spacing is the number of pixels separating tools in a single band.

Data Type

Long

Syntax

```
<oBand>.ToolsHSpacing [= <lSpacing>]
<oBand>.ToolsVSpacing [= <lSpacing>]
```

Example

```
ActiveBar1.Bands("Toolbox").ToolsHSpacing = 20
ActiveBar1.Bands("Toolbox").ToolsVSpacing = 20
```

Top

Description

Sets or returns the top coordinate of a floating band in pixels. The coordinate properties apply to the band when it is floating. They will not be set to the exact values supplied, ActiveBar will auto size the band's layout for best fit.

Data Type

Long

Syntax

```
<oBand>.Top [= <lTop>]
```

Example

```
ActiveBar1.Bands("Toolbox").Top = 300
```

Type

Description

Sets or returns the type of a band (Normal, Menubar, Popup)

Data Type

Integer

Syntax

```
<oBand>.Type [= <iType>]
```

Example

```
If oBand.Type <> DDBTPopUp Then
lstBands.AddItems oBand.Caption
End If
```

Visible

Description

Hides or shows the band

Data Type

Boolean

Syntax

```
<oBand>.Visible [= <bVisible>]
```

Example

```
Private Sub cmdShowToolbox_Click()
ActiveBar1.Bands("Toolbox").Visible = True
End Sub
```

Width

Description

Sets or returns the width of a floating band in pixels. The coordinate properties apply to the band when it is floating. They will not be set to the exact values supplied, ActiveBar will auto size the band's layout for best fit.

Data Type

Long

Syntax

<oBand>.Width [= <lWidth>]

Example

ActiveBar.Bands("Toolbox").Width = 300

Wrappable

Description

Determines if the band should wrap its tools when it is sized below the sum width of all it tools.

Data Type

Boolean

Syntax

<oBand>.Wrappable [= <bWrappable>]

Example

ActiveBar1.Bands("Format").Wrappable = False

Band Methods

Refresh	Refreshes the band. `<oBand>.Refresh`
TrackPopup	Displays the band as a context menu at the specified coordinates `<oBand>.TrackPopup(<iXPos>, <iYPos>)`

Refresh

Description

Refresh the band. Call this method after changing any of the display properties of the band.

Syntax

```
<oBand>.Refresh
```

Example

```
ActiveBar1.Bands("Toolbox").Refresh
```

TrackPopup

Description

Display the band as a context menu at the specified coordinates. Passing -1 for iXPos and iYPos will display the band at the current mouse cursor position.

Syntax

```
<oBand>.TrackPopup(<iXPos>, <iYPos>)
```

Example

```
Private Sub txtNote_MouseUp(Button, X, Y)
ActiveBar1.Bands("NotePopup").TrackPopup X, Y
End Sub
```

Bands Methods

Add	Adds a new band object to the collection `Set <oBand> = <colBands>.Add(sName)`
Count	Returns the number of bands in the collection `<iCount> = <colBands>.Count`
Item	Returns the band object at the specified index `Set <oBand> = <colBands>.Item(<vBand>)`
Remove	Removes the band object at the specified index `<colBands>.Remove(<vBand>)`
RemoveAll	Removes all band items from the collection `<colBands>.RemoveAll`

Add

Description

Adds a new band to the bands collection.

Syntax

```
<oBand> = <colBands>.Add <sName>
```

Example

```
' Create toolbars
Set oBand = ActiveBar1.Bands.Add "Standard"
' Set band properties
```

Count

Description

Returns the number of bands in the collection

Syntax

```
<iCount> = <colBands>.Count
```

Example

```
For I = 0 to ActiveBar.Bands.Count - 1
' Set band properties
Next i
```

Item

Description

Returns the band object at the specified index or with the specified name. This method is not needed when using Visual Basic since Visual Basic allows you to directly use the bands collection as an array with index values or keys.

Syntax

```
<oBand> = <colBands>.Item(<vIndex>)
```

Example

```
oBand = ActiveBar1.Bands.Item(0)
' or
oBand = ActiveBar1.Bands.Item("Standard")
```

Remove

Description

Removes a band from the collection. You can specify the index or name of the band.

Syntax

```
<colBands>.Remove <vIndex>
```

Example

```
ActiveBar1.Bands.Remove 0
' or
ActiveBar1.Bands.Remove "Standard"
```

RemoveAll

Description

Removes all band items from the collection.

Syntax

```
<colBands>.RemoveAll
```

Example

```
ActiveBar1.Bands.RemoveAll
```

Page Properties

Caption	String	Sets or returns the caption of the page, displayed on the page's tab.
Tools	Collection	Returns a collection of tool objects belonging to the page.

Caption

Description

Sets or returns the caption of the page, displayed on the page's tab.

Data Type

String

Syntax

```
<oPage>.Caption [= <sCaption>]
```

Example

```
With ActiveBar1.Bands("Toolbox") Do
For i = 0 To.Pages.Count -1
.Pages(i).Caption = aToolboxCaptions(i)
Next
End With
```

Tools

Description

Returns a collection of tool objects belonging to the page.

Data Type

Collection

Syntax

```
<oPage>.Tools
```

Example

```
oTool = ActiveBar1.Bands("Toolbox").Pages(0).Tools.Add "Line"
```

Pages Methods

Add	Adds a new page object to the collection
Count	Returns the number of pages in the collection
Item	Returns a page object at the specified index or key in the collection
Remove	Removes a page object at the specified index or key in the collection

Add

Description

Adds a new page to a pages collection.

Syntax

```
<oPage> = <colPages>.Add <sPageName>
```

Example

```
' Add General page to the controls band
Set oPage = ActiveBar1.Bands("Controls").Pages.Add "General"
' set oPage properties
```

Count

Description

Returns the number of pages in a pages collection.

Syntax

```
<iPages> = <colPages>.Count
```

Example

```
For I = 0 to ActiveBar1.Bands("Controls").Pages.Count - 1
' Set page properties
Next i
```

Item

Description

Returns a page object at the specified index or the specified key.

Syntax

```
<oPage> = <colPages>.Item <vIndex>
```

Example

```
Set oPage = ActiveBar1.Bands("Controls").Pages.Item "General"
' or
Set oPage = ActiveBar1.Bands("Controls").Pages.Item 0
```

Remove

Description

Removes specified page from a pages collection

Syntax

```
<colPages>.Remove <vIndex>
```

Example

```
ActiveBar1.Bands("Controls").Pages.Remove "Custom"
' or
ActiveBar1.Bands("Controls").Pages.Remove 1
```

Tool Properties

Alignment	Integer	Sets or returns the alignment of the tool within its display area
BeginGroup	Boolean	Determines if the tool marks the beginning of a group. Setting this to true would display a separator before the tool
Caption	String	Sets or return the caption of the tool. The caption is displayed as menu text or button tool text
CaptionPosition	Integer	Sets or returns the position of the caption relative to its icon
Category	String	Sets or returns the category of the tool
CBLines	Integer	Number of lines displayed in dropdown combo
CBList	ComboList	ComboBox items list
CBListIndex	Integer	Current item index in the combo box list
CBStyle	Long	ComboBox tool style
CBWidth	Integer	Width of the dropdown of a combobox tool
Checked	Boolean	Sets or returns the checked status of the tool
ControlType	Integer	Sets or returns the type of the tool (Button, Combo, Text, …)
Custom	Object	Pointer to custom ITool interface object. (Pro Edition)
Description	String	Sets or returns a description of the tool. The description is displayed to the end user in the customization dialog box
Enabled	Boolean	Enabled or disables the tool or menu item
Height	Long	Sets or returns the height of the tool in pixels
HelpContextID	Long	Sets or returns a help context ID used for context sensitive help implementation
Name	String	Sets or returns the name of the tool. Used as a key in the tools collections
ShortcutKey	Long	Sets or returns the ANSI shortcut keycode for the tool
Style	Integer	Sets or returns the display style of the tool.
SubBand	String	Sets or returns the name of sub band to be displayed on mouse click.
Tag	Long	Use defined field
Text	String	Text property of the combo box or edit type tools
ToolID	Long	Unique tool identifier
TooltipText	String	Text displayed on the fly by tooltip
Width	Long	Width of the tool in pixels

Alignment

Description

Sets or returns the alignment of the tool within its display area.

Data Type

Integer

Syntax

```
<oTool>.Alignment [= <iAlignment>]
```

Example

```
ActiveBar1.Tools("Zoom").Alignment = DDTACenterBottom
```

BeginGroup

Description

Determines if the tool marks the beginning of a group. Setting this to true would display a separator before the tool.

Data Type

Boolean

Syntax

```
<oTool>.BeginGroup [= <bBeginGroup>]
```

Example

```
ActiveBar1.Tools("Print").BeginGroup = True
```

Caption

Description

Sets or return the caption of the tool. The caption is displayed as menu text or button tool text.

Data Type

String

Syntax

```
<oTool>.Caption [= <sCaption>]
```

Example

```
ActiveBar1.Tools("Print").Caption = "P&rint"
```

CaptionPosition

Description

Sets or returns the position of the caption relative to its icon

Data Type

Integer

Syntax

```
<oTool>.CaptionPosition [= <iCaptionPosition>]
```

Example

```
ActiveBar1.Tools("New").CaptionPosition = DDCPBelow
```

Category

Description

Sets or returns the category of the tool

Data Type

String

Syntax

```
<oTool>.Category [= <sCategory>]
```

Example

```
ActiveBar1.Tools("New").Category = "File"
ActiveBar1.Tools("Save").Category = "File"
```

CBLines

Description

Number of lines displayed in dropdown combo

Syntax

```
<oTool>.CBLines [= <iCBLines>]
```

Example

```
ActiveBar1.Tools("FontDrop").CBLines = 10
```

CBList

Description

Returns the ComboBox items list, applies to tools of type ComboBox only.

Syntax

```
<oTool>.CBList
```

Example

```
For I = 25 to 200 Step 25
      ActiveBar1.Tools("Zoom").CBList.AddItem Str(i)
Next
```

CBListIndex

Description

Sets or returns the current item index in the combo box list. Applies to ComboBox tools only.

Syntax

```
<oTool>.CBListIndex
```

Example

```
Private Sub AvtiveBar1_ComboSelChange(ByVal Tool As Tool)
      If Tool.Name = "Zoom" Then
            ZoomView(Tool.CBList(Tool.CBListIndex))
      End If
End Sub
```

CBStyle

Description

Sets or returns a Combobox tool style.

Settings: 0-Normal 1-Readonly 2-Sorted

Syntax

```
<oTool>.CBStyle [= <iCBStyle>]
```

Example

```
oTool = ActiveBar1.Add 0, "Zoom"
oTool.ControlType = DDTTComboBox
For I = 25 to 200 Step 25
      oTool.CBList.AddItem Str(i)
Next
oTool.CBStyle = 1 ' Readonly
oTool.CBWidth = 150
oTool.Caption = "&Zoom"
```

CBWidth

Description

Sets or returns the width of the ComboBox drop down listbox in pixels.

Syntax

```
<oTool>.CBWidth [= <iCBWidth>]
```

Example

```
oTool = ActiveBar1.Add 0, "Zoom"
oTool.ControlType = DDTTComboBox
For I = 25 to 200 Step 25
      oTool.CBList.AddItem Str(i)
Next
oTool.CBStyle = 1 ' Simple
oTool.CBWidth = 150
oTool.Caption = "&Zoom"
```

Checked

Description

Sets or returns the checked status of the tool. A checked button icon is depressed and a menu item is checked (Checkmark to its right) if it does not have an icon associated with it.

Data Type

Boolean

Syntax

```
<oTool>.Checked [= <bChecked>]
```

Example

```
Private Sub ActiveBar1_Click(ByVal Tool As Tool)
      If Tool.Name = "Bold" Then
      Tool.Checked = Not Tool.Checked
      SetBold Tool.Checked
      End If
End Sub
```

ControlType

Description

Sets or returns the type of the tool (Button, Combo, Text, …)

Data Type

Integer

Syntax

```
<oTool>.ControlType [= <iControlType>]
```

Example

```
oTool = ActiveBar1.Add 0, "Zoom"
oTool.ControlType = DDTTComboBox
For I = 25 to 200 Step 25
      oTool.CBList.AddItem Str(i)
Next
oTool.CBStyle = 1 ' Simple
oTool.CBWidth = 150
oTool.Caption = "&Zoom"
```

Custom

Description

This is an advanced feature for Visual C++ developers to create their own custom tools using the ITool interface.

Data Type

Object

Description

Description

Sets or returns a description of the tool. The description is displayed to the end user in the customization dialog box

Data Type

String

Syntax

<oTool>.Description [= <sDescription>]

Example

```
oTool = ActiveBar1.Add 0, "Zoom"
oTool.ControlType = DDTTComboBox
For I = 25 to 200 Step 25
      oTool.CBList.AddItem Str(i)
Next
oTool.CBStyle = 1 ' Simple
oTool.CBWidth = 150
oTool.Caption = "&Zoom"
oTool.Description = "Image Zoom Level"
```

Enabled

Description

Enabled or disables the tool or menu item

Data Type

Boolean

Syntax

```
<oTool>.Enabled [= <bEnabled>]
```

Example

```
Clipboard.Clear
ActiveBar1.Tools("Paste").Enabled = False
```

Height

Description

Sets or returns the height of the tool in pixels

Data Type

Long

Syntax

```
<oTool>.Height [= <lHeight>]
```

Example

```
oTool.Height = 30
```

HelpContextID

Description

Sets or returns a help context ID used for context sensitive help implementation

Data Type

Long

Syntax

```
<oTool>.HelpContextID [= <lContextID>]
```

Example

```
Private Sub cmdHelp_Click()
GetHelp ActiveBar1.CurrentTool.HelpContextID
End Sub
```

Name

Description

Sets or returns the name of the tool. Used as a key in the tools collections

Data Type

String

Syntax

```
<oTool>.Name [= <sName>]
```

Example

```
ActiveBar1.Tools(0).Name = "New"
```

ShortcutKey

Description

Sets or returns the ANSI shortcut keycode for the tool. ActiveBar will not use the shortcut key automatically, you must capture the key on your form by setting PreviewKeys to True and calling the OnKeyDown and OnKeyUp in your form's KeyDown and KeyUp events.

Data Type

Long

Syntax

```
<oTool>.ShorcutKey [= <lKey>]
```

Style

Description

Sets or returns the display style of the tool.

Data Type

Integer

Syntax

```
<oTool>.Style [= <iStyle>]
```

Example

```
For Each oTool in ActiveBar1.Bands("Navigate").Tools
oTool.Style = DDSIconText     ' Display Icon and Text
Next
```

SubBand

Description

Sets or returns the name of sub band to be displayed on mouse click.

Data Type

String

Syntax

```
<oTool>.SubBand [= <sSubBand>]
```

Example

```
ActiveBar1.Tools("mnuFile").SubBand = "FileMenu"
```

Tag

Description

User defined property.

Data Type

Long

Syntax

<oTool>.Tag [= <lTag>]

Text

Description

Text property of the combo box or edit type tools

Data Type

String

Syntax

<oTool>.Text [= <sText>]

Example

ActiveBar1.Tools("txtURL").Text = "http://www.datadynamics.com"

ToolID

Description

Unique tool identifier

Data Type

Long

Syntax

```
<oTool>.ToolID
```

Example

```
If Tool.ToolID = 201 Then DoSomething
```

TooltipText

Description

Text displayed on the fly by tooltip

Data Type

String

Syntax

```
<oTool>.TooltipText [= <sText>]
```

Example

```
ActiveBar1.Tools("New").TooltipText = "Create a New File"
```

Width

Description

Width of the tool in pixels

Data Type

Long

Syntax

```
<oTool>.Width [= <lWidth>]
```

Example

```
For Each oTool In ActiveBar1.Bands("Navigate").Tools
oTool.Width = 48
Next
```

Tool Methods

SetPicture	Sets the icon of the tool. You can use it to set enabled, pressed, and disabled icons. `SetPicture(<iIndex>, <picHandle>, <clr>)`
SetPictureMask	Sets the masking bitmap for the tools icon `SetPictureMask(<iIndex>, <picMask>)`

SetPicture

Description

Sets the icon of the tool. You can use it to set enabled, pressed, and disabled icons.

Syntax

```
<oTool>.SetPicture(<iIndex>, <picHandle>,  <clrTransparent>)
```

Example

```
With ActiveBar1.Tools("New") Do
.SetPicture(0, LoadPicture("new.bmp"), vbGrey)
End With
```

SetPictureMask

Description

Sets the masking bitmap for the tools icon

Syntax

```
SetPictureMask(<iIndex>, <picMask>)
```

Example

```
ActiveBar1.Tools("stoplight").SetPictureMask(0,
LoadPicture("lightmask.bmp"))
```

Tools Methods

Add	Add a new tool to the collection.
Count	Returns the number of tools in the collection
Item	Returns the tool object at the specified index
Remove	Removes the tool at the specified index from the collection
RemoveAll	Removes all tools from the collection

Add

Description

Adds a new tool to a tools collection.

Syntax

```
<oTool> = <colTools>.Add <lToolID>, <sName>
```

Example

```
Set oTool = ActiveBar1.Tools.Add 0, "FileNew"
' Set oTool properties
```

Count

Description

Returns the number of tools in a tools collection.

Syntax

```
<iCount> = <colTools>.Count
```

Example

```
For I = 0 to ActiveBar1.Tools.Count - 1
' Set tool properties
Next i
```

Item

Description

Returns tool object at the specified index or key.

Syntax

```
<oTool> = <colTools>.Item <vIndex>
```

Example

```
Set oTool = ActiveBar1.Tools.Item 0
' or
Set oTool = ActiveBar1.Tools.Item "FileNew"
```

Remove

Description

Removes the tool at the specified index or key from a tools collection.

Syntax

```
<colTools>.Remove <vIndex>
```

Example

```
ActiveBar1.Tools.Remove 0
' or
ActiveBar1.Tools.Remove "FileNew"
```

RemoveAll

Description

Removes all tools from a tool collection

Syntax

```
<colTools>.RemoveAll
```

Example

```
ActiveBar1.Tools.RemoveAll
```

ComboList Methods

AddItem	Add a new item to the list.
Clear	Removes all items in the list
Count	Returns the number of items in the collection
InsertItem	Adds a new item to the list
Item	Returns the value at the specified index
Remove	Removes an item from the list at the specified index

AddItem

Description

Adds a new item to the ComboBox list.

Syntax

```
<cblist>.AddItem <sItem>
```

Example

```
' Add fonts to the font tool
oTool.CBList.AddItem "Arial"
```

Clear

Description

Removes all items from the ComboBox list.

Syntax

```
<cblist>.Clear
```

Example

```
' Clear the list
oTool.CBList.Clear
oTool.AddItem "Arial"
```

Count

Description

Returns the number of item in the list.

Syntax

```
<iCount> = <cblist>.Count
```

Example

```
For I = 0 to oTool.CBList.Count -1
Debug.Print oTool.CBList.Item(I)
Next i
```

InsertItem

Description

Adds a new item to the ComboBox list at the specified position.

Syntax

```
<cblist>.InsertItem(<iIndex>, <sItem>)
```

Example

```
oTool.CBList.InsertItem 0, "Item 1"
```

Item

Description

Returns the item value at the specified index.

Syntax

```
<sItem> = <cbList>.Item(<iIndex>)
```

Example

```
For I = 0 to oTool.CBList.Count -1
Debug.Print oTool.CBList.Item(I)
Next
```

Remove

Description

Removes an item at the specified index.

Syntax

```
<cblist>.Remove(<iIndex>)
```

Example

```
oTool.CBList.Remove(0)
```

System Requirements

Windows95 or Windows NT development environment.

Included Files

Program Files

ACTBAR.OCX	ActiveBar ActiveX control
ACTBAR.LIC	ActiveBar License File - Registered copies only.
ACTBAR.CAB	ActiveBar signed cabinet file for web downloads.
BARDES.DLL	ActiveBar Designer DLL
DESIGN.EXE	ActiveBar Designer Executable

Documentation

ACTBAR.HLP	ActiveBar Online Help
ACTBAR.RTF	ActiveBar Manual - RTF Format

Sample Programs

VB*.*	Visual Basic sample programs.
VC*.*	Visual C++ sample programs.
DELPHI*.*	Borland Delphi sample programs.

Distribution

ActiveBar is a lightweight ActiveX that requires no run-time DLLs. The only distributable component is ACTBAR.OCX and ACTBAR.CAB.

Technical Support

Problems?

If you are having problems using the ActiveBar ActiveX, please make sure that the control was registered by the installation program. If not, use the RegSvr32.exe program to register the ACTBAR.OCX file and the BARDES.DLL. The REGSVR32.EXE is included with your Visual Basic installation.

If the problem is not fixed, and a solution is not listed in the README.TXT file included with your installation, please contact our technical support staff. You can reach Data Dynamics Technical Support via:

Email:	support@datadynamics.com
Internet:	http://www.datadynamics.com
Fax:	614.899.2943
Telephone:	614.895.3142 (8 a.m. to 5 p.m. EST, M-F)

Please include a complete description of the problem and the version of the ActiveBar.

We welcome suggestions

We at Data Dynamics welcome your suggestions for improving ActiveBar. Much of the initial feedback has been included in this version of ActiveBar.

Fax your suggestions to us at 614.899.2943, email to support@datadynamics.com, or write to Data Dynamics, 2600 Tiller Lane, Columbus, OH 43231.

Disk Defects Warranty

Data Dynamics is committed to producing a quality product that undergoes an extensive series of tests and refinements at both the manufacturing and development levels. In the unfortunate case that you receive a damaged disk, Data Dynamics will replace your disk free of charge. Please contact us at the above address to get your replacement disks.

License Agreement

The product in this package (libraries and object code) is proprietary to Data Dynamics, Ltd. and is protected by Federal Copyright Law. Data Dynamics retains the title to and ownership of the Product. You are licensed to use this Product on the following terms and conditions:

LICENSE - The licensee is defined as the individual software developer utilizing the Product**. This license is not for an entire company but for a single developer.** Data Dynamics hereby grants the licensee a nonexclusive license authorizing the licensee to use the enclosed Product on one computer at a time for development purposes. The licensee is also permitted to distribute this product to one, and only one web server to host the ActiveX control. Please contact Data Dynamics if you require additional licenses. You may incorporate the sample code into your applications. Use of this product by more than one individual or by anyone other than the licensee terminates, without notification, this license and the right to use this product.

YOU MAY NOT: Distribute, rent, sub-license or otherwise make available to others the software or documentation or copies thereof, except as expressly permitted in this License without prior written consent from Data Dynamics. In the case of an authorized transfer, the transferee must agree to be bound by the terms and conditions of this License Agreement.

RESTRICTIONS: - You may use this Product in your business application for sale or distribution as long as:

The product that you produce and/or distribute is NOT a software development product, a product that is sold primarily to software developers or system integrators or a development environment of any kind. Please contact Data Dynamics, Ltd. for special commercial licensing provisions in these circumstances.

The software serial number and user must be registered with Data Dynamics in order to receive support or distribution rights.

You may not remove any proprietary notices, labels, trademarks on the software or documentation.

You may not modify, decompile, disassemble, reverse engineer or translate the software.

FILES THAT MAY BE DISTRIBUTED WITH YOUR APPLICATION: ACTBAR.OCX and ACTBAR.CAB

US GOVERNMENT RESTRICTED RIGHTS - Use, duplication or disclosure by the United States Government is subject to restrictions as set forth under DFARS 252.227-7013 or in FARS 52.227-19 Commercial Computer Software - Restricted Rights.

TERM - You may terminate your License and this Agreement at anytime by destroying all copies of the Product and Product Documentation. They will also terminate automatically if you fail to comply with any term or condition in this Agreement.

LIMITED WARRANTY - This software and documentation are sold ‘‘as is’’ without any warranty as to their performance, merchantability or fitness for any particular purpose. The entire risk as to the quality and performance of the software is assumed by the licensee. Data Dynamics warrants that the diskettes on which the Program is furnished will be free from any defects in materials. Exclusive remedy in the event of a defect is expressly limited to the replacements of diskettes. In no event shall Data Dynamics or anyone else who has been involved in the creation, development, production, or delivery of this software be liable for any direct, incidental or consequential damages, such as, but not limited to, loss of anticipated profits, benefits, use, or data resulting from the use of this software, or arising out of any breach of warranty.