Notice

Trademarks

Southern Computer Systems, Inc.
P.O. Box 1888
Birmingham, Alabama 35201
251- 2985
Fax (205)322-4851

SCS Europe Ltd.
Suite 5A, Douglas House, Simpson Road
Fenny Stratford
Milton Keynes, Buckinghamshire
Telex 755714
England MK1 1BA 44 1908 630611

VistaStat ControlUser GuidePublication No. 1997.02.06.15.24.00 Revision 1.8

Table of Contents

Introduction to VistaStat

What is VistaStat?

VistaStat is an easy-to-use tool for capturing application, operator and system performance data. This 32-bit ActiveX Control (OCX) will provide time-based statistics for each critical process or event within your developed application.

Once installed, VistaStat can be integrated into the Visual Basic toolbar and quickly placed on a form(s) within your application. It provides automatic or predefined counters for capturing the number of keystrokes, number of mouse clicks, total time the mouse was in motion and total time the application was running. In addition, you can create customized event counters.

The number of custom defined application counters is virtually unlimited and their implementation is quick and easy. Customized event counters can capture such details as:

- Keystrokes required per document
- Total fields keyed per screen
- Number of documents processed
- Number of pages scanned
- Count of document types processed
- Amount of time to complete an automated process (e.g. ICR, host lookup, file transfer)
- Number of times the operator clicked a mouse button while in an application

VistaStat will report data on a time-increment basis, so that operators and managers can easily monitor the performance of the system. Some examples of these types of reports include:

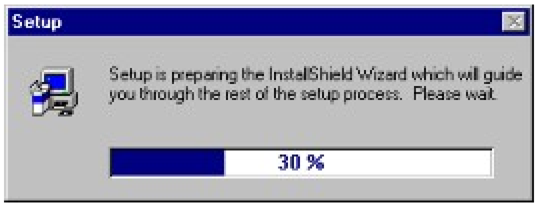
- Average keystrokes per hour
- Average operator keystrokes per hour
- Number of ICR characters recognized per second
- Documents processed per day
- Pages scanned per minute

Productivity statistics from VistaStat are stored in any database compatible with Visual Basic (e.g. formatted text, SQL Server, Access, Excel, ODBC, etc.). This information can be accessed and viewed by a number of common tools, such as Crystal Reports, Excel, Lotus 1-2-3, SQL, Access, as well as customer-developed report application.

How to Install VistaStat

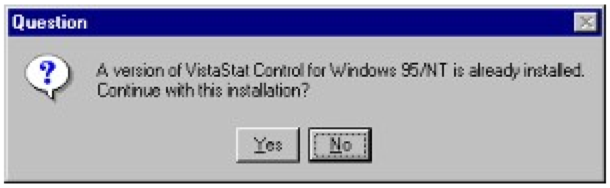
Follow the steps below to install VistaStat:

1. Insert the Installation diskette.
2. From the start menu, select the RUN option and enter A:\SETUP to execute the VistaStat setup program on the install diskette and Click OK.
3. The system will prepare the InstallShield wizard that will guide the installation process. The following setup screen is shown:



Setup is preparing the InstallShield Wizard which will guide you through the rest of the setup process. Please wait.

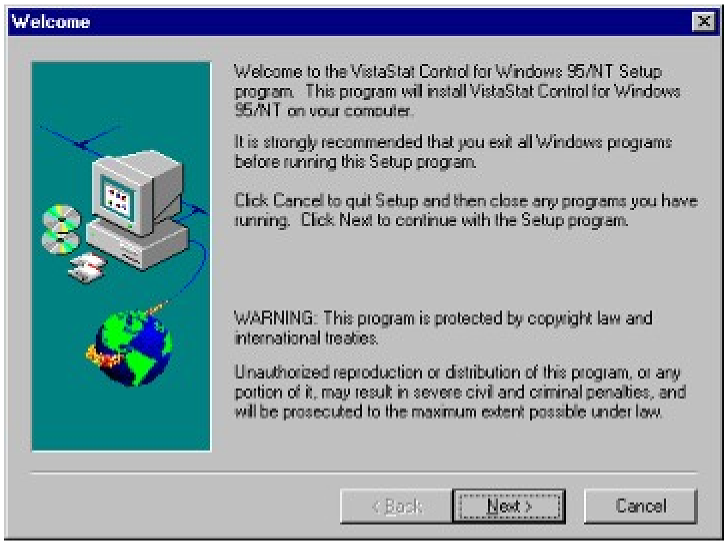
4. If you've already installed VistaStat on your computer, the following message will be displayed:



A version of VistaStat Control for Windows 95/NT is already installed. Continue with this installation?

5. Click Yes to continue. The existing version of VistaStat will be overwritten. If you do not want to continue, click No.

6. Once the wizard has been loaded, the following Welcome screen is displayed:



Welcome to the VistaStat Control for Windows 95/NT Setup program. This program will install VistaStat Control for Windows 95/NT on your computer.

It is strongly recommended that you exit all Windows programs before running this Setup program.

Click Cancel to quit Setup and then close any programs you have running. Click Next to continue with the Setup program.

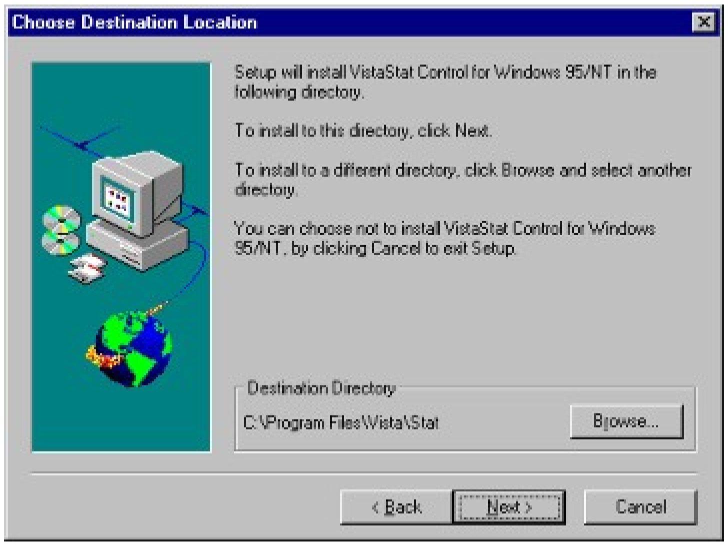
WARNING: This program is protected by copyright law and international treaties.

Unauthorized reproduction or distribution of this program, or any portion of it, may result in severe civil and criminal penalties, and will be prosecuted to the maximum extent possible under law.

7. If you want to exit the install program, click the Cancel button. Otherwise, click Next to

continue.

8. The Choose Destination Location screen is shown. The location entered is where the VistaStat sample program will be placed.



Setup will install VistaStat Control for Windows 95/NT in the following directory.

To install to this directory, click Next.

To install to a different directory, click Browse and select another directory.

You can choose not to install VistaStat Control for Windows 95/NT, by clicking Cancel to exit Setup.

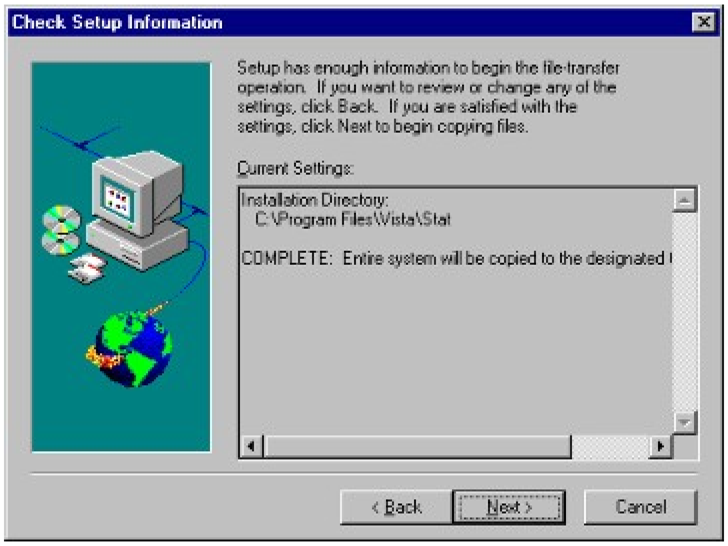
Destination Directory

C:\Program Files\Vista\Stat

9. The default directory where the sample programs will be installed is indicated in the Destination Directory area of the screen. To select another directory, click the Browse button.

10. To go back to the previous screen, click Back. To exit from the Install program, click Cancel.

11. Otherwise, click Next to continue. The system will then display the following screen:



Setup has enough information to begin the file-transfer operation. If you want to review or change any of the settings, click Back. If you are satisfied with the settings, click Next to begin copying files.

Current Settings:

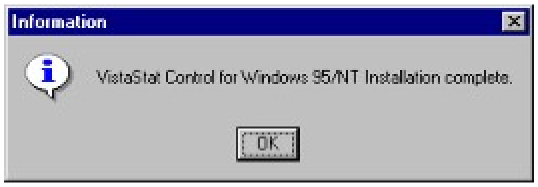
Installation Directory:
C:\Program Files\Vista\Stat

COMPLETE: Entire system will be copied to the designated

12. Click Back to go to the previous screen or Cancel to exit the install program.

13. Otherwise, click Next to continue. The program and associated files are then installed into the specified directory.

14. Once the installation is complete, the following screen is shown:

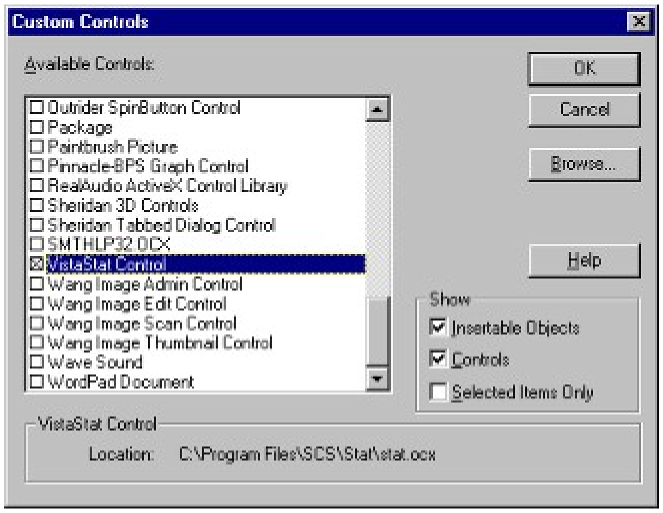


VistaStat Control for Windows 95/NT Installation complete.

15. Click the OK button. All **files** necessary to run VistaStat have now been installed.

16. Before using VistaStat you must enable the control From within Visual Basic.. Select Custom Controls from the Tools pull-down menu. The Custom Controls dialog will be displayed.

17. Then select the VistaStat Control from the list of Available Controls, as shown below and click the OK botton:



Available Controls:

- ☐ Outrider SpinButton Control
- ☐ Package
- ☐ Paintbrush Picture
- ☐ Pinnacle-BPS Graph Control
- ☐ RealAudio ActiveX Control Library
- ☐ Sheridan 3D Controls
- ☐ Sheridan Tabbed Dialog Control
- ☐ SMTHLP32.OCX
- ☒ VistaStat Control
- ☐ Wang Image Admin Control
- ☐ Wang Image Edit Control
- ☐ Wang Image Scan Control
- ☐ Wang Image Thumbnail Control
- ☐ Wave Sound
- ☐ WordPad Document

Show
- ☑ Insertable Objects
- ☑ Controls
- ☐ Selected Items Only

VistaStat Control
Location: C:\Program Files\SCS\Stat\stat.ocx

18. The VistaStat icon appears on the Visual Basic tool bar containing the list of controls available for use.

What is Installed

The following files have been **installed** on your computer in the directory specified during the install procedure (the default directory for the Install program is \Program Files\Vista\Stat):

- Samples\StatSamp.frm
- Samples\StatSamp.frx
- Samples\StatSamp.vbp

The following files are installed in your Windows95 System directory or Windows NT System32 directory:

- VistaStat.dll
- VistaStat.ocx
- VistaStat.hlp
- VistaStat.cnt
- mfc42.dll
- msvcrt.dll
- regsvr32.exe
- olepro32.dll

Customer Support

First, look in the printed product documentation or consult Help. If you cannot find the answer, you can obtain product support in several ways:

Electronic Services

The following electronic customer support services are available from Southern Computer Systems (SCS) 24 hours a day, seven days a week, including holidays.

Fax

You can fax support questions to SCS at (205) 322-4851. There is a per-incident charge for questions that are not related to a software defect.

SCS Electronic Bulletin Board

You can upload or download the most current files and technical notes via modem (no parity; 8 data bits; 1 stop bit). The modem number is (205) 251-1330.

Internet

World Wide Web

Support for our Vista software is available through our World Wide Web site at www.scsinc.com. Choose the "Support" option from our home page. Here you will find FAQs, the most current files and technical notes.

FTP

We support anonymous logon. Instructions for using FTP are available from our web site or electronic bulletin board. When logging on as anonymous, use your complete e-mail address as your password.

Standard Support

Standard product support is available via fax, e-mail, or phone. There is a per-incident charge for questions that are not related to a software defect. Payment must be by VISA or MasterCard. Standard Support charges are subject to SCS prices, terms and conditions in place at the time the service is used.

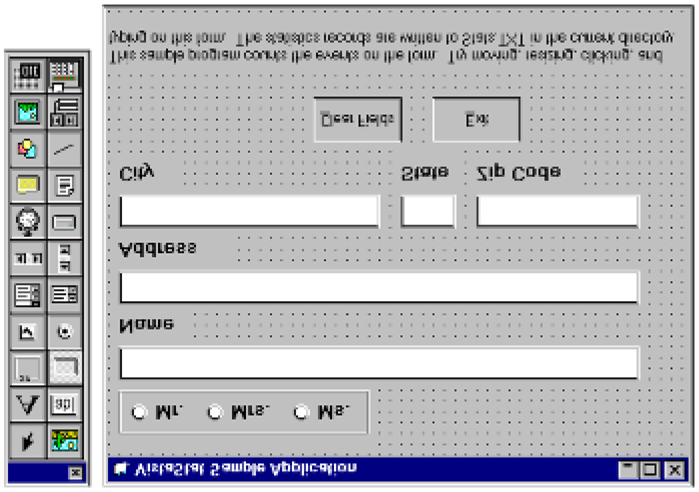
Priority Support

Priority product support requires an annual support agreement. Call an SCS sales representative for more information at (205)251-2985.

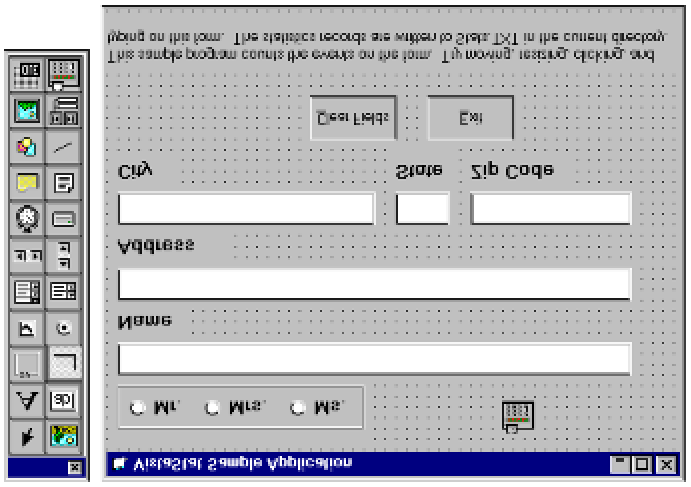
Placing VistaStat on a Form

Once VistaStat is properly **installed** and registered, the VistaStat control will appear in the Visual Basic toolbar. Follow the steps below to place VistaStat in your application.

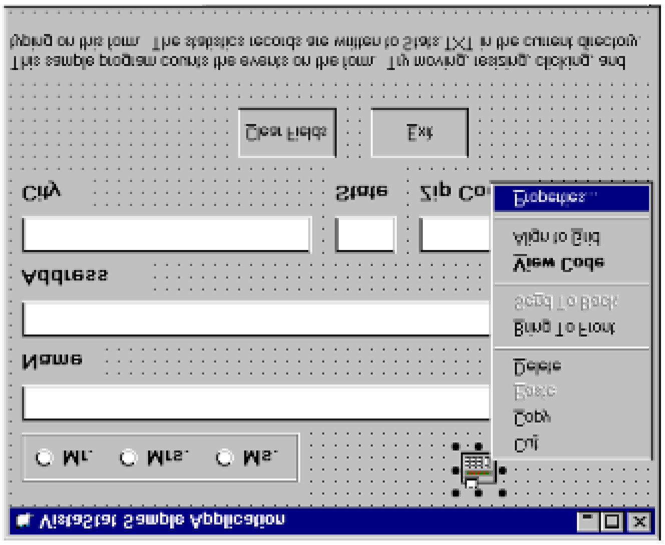
1. Start a Visual Basic project (the illustrations in this procedure show the sample application that is included with VistaStat).
2. Click the VistaStat control on the Visual Basic toolbar. The cursor will change to a cross-hair when on the Visual Basic form.


This sample program counts the events on the form. Try moving, resizing, clicking, and typing on this form. The statistics records are written to Stats.TXT in the current directory.

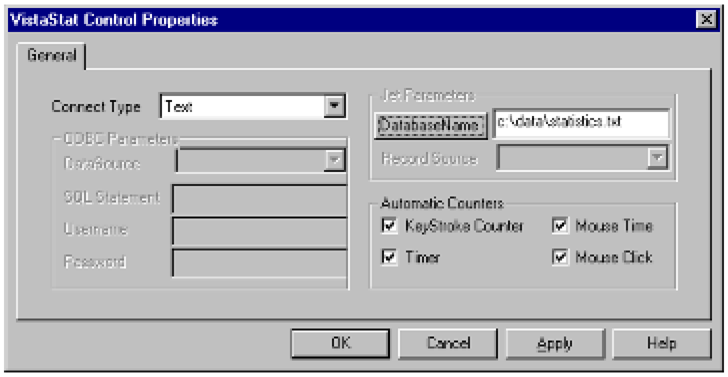
3. Move the cross-hair to the desired location on the form.
4. Holding down the left mouse button, drag the mouse downward until a small box is drawn. Then, release the mouse button.
5. The VistaStat control will appear on the form.


This sample program counts the events on the form. Try moving, resizing, clicking, and typing on this form. The statistics records are written to Stats.TXT in the current directory.

6. To establish the properties for VistaStat, move the cursor over the VistaStat control and click the right mouse button. Select Properties from the displayed menu.


This sample program counts the events on the form. Try moving, resizing, clicking, and typing on this form. The statistics records are written to Stats.TXT in the current directory.

7. The VistaStat Control Properties screen is displayed, as shown below. This screen is used to specify a variety of parameters about your application. Refer to the description of VistaStat Control Properties in this manual for more information on selecting parameters on this screen.


VistaStat Control Properties

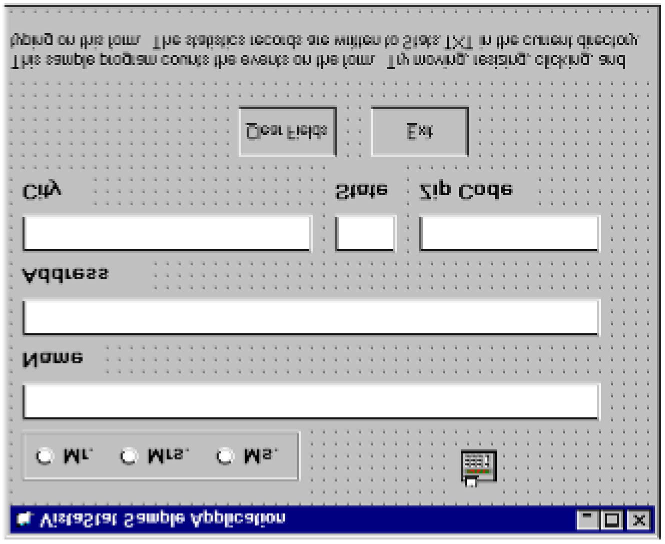
Connect Type: Text
DatabaseName: c:\data\statistics.txt

Automatic Counters: [✓ KeyStroke Counter] [✓ Mouse Time] [✓ Timer] [✓ Mouse Click]

Sample Application

The sample application that is included with VistaStat contains custom counters. All files associated with this sample application are automatically copied into the VistaStat install directory during the installation process.

The form for the sample application looks like the illustration below:



This sample program counts the events on the form. Try moving, resizing, clicking, and typing on this form. The statistics records are written to Stats.TXT in the current directory.

The output records that contain the accumulated statistics are stored in a file called Stats.TXT. This file is located in the current directory.

When the sample application is installed, all of the predefined counters are selected. This includes the following:

- KeyStroke
- MouseClick
- MouseTime
- Timer

The sample application also contains the following custom counters:

- Activate Events
- Click Events
- DblClick Events
- Deactivate Events
- DragDrop Events
- DragOver Events
- GotFocus Events
- KeyDown Events

- KeyPress Events
- KeyUp Events
- LostFocus Events
- MouseDown Events
- MouseMove Events
- MouseUp Events
- Paint Events
- QueryUnload Events
- Resize Events

The records for each session are appended to the file Stats.TXT.

The Visual Basic code associated with the custom counters in the sample application is shown below:

```
Option Explicit

Private Sub cmdButton_Click()
    txtText1.Text = ""
    txtText2.Text = ""
    txtText3.Text = ""
    txtText4.Text = ""
    txtText5.Text = ""
    Option1.Value = False
    Option2.Value = False
    Option3.Value = False
    Option1.SetFocus
End Sub

Private Sub Command1_Click()
    End
End Sub

Private Sub Form_Activate()
    With vistaStat.Counters("Test Session" & "Activate Events")
        .CounterValue = .CounterValue + 1
    End With
    Option1.Value = True
    Option2.Value = False
    Option3.Value = False
    Option1.SetFocus
End Sub

Private Sub Form_Click()
    With vistaStat.Counters("Test Session" & "Click Events")
        .CounterValue = .CounterValue + 1
    End With
End Sub

Private Sub Form_DblClick()
    With vistaStat.Counters("Test Session" & "DblClick Events")
        .CounterValue = .CounterValue + 1
    End With
End Sub

Private Sub Form_Deactivate()
    With vistaStat.Counters("Test Session" & "Deactivate Events")
```

```
        .CounterValue = .CounterValue + 1
    End With
End Sub

Private Sub Form_DragDrop(Source As Control, X As Single, Y As Single)
    With vistaStat.Counters("Test Session" & "DragDrop Events")
        .CounterValue = .CounterValue + 1
    End With
End Sub

Private Sub Form_DragOver(Source As Control, X As Single, Y As Single, State As
Integer)
    With vistaStat.Counters("Test Session" & "DragOver Events")
        .CounterValue = .CounterValue + 1
    End With
End Sub

Private Sub Form_GotFocus()
    With vistaStat.Counters("Test Session" & "GotFocus Events")
        .CounterValue = .CounterValue + 1
    End With
End Sub

Private Sub Form_KeyDown(KeyCode As Integer, Shift As Integer)
    With vistaStat.Counters("Test Session" & "KeyDown Events")
        .CounterValue = .CounterValue + 1
    End With
End Sub

Private Sub Form_KeyPress(KeyAscii As Integer)
    With vistaStat.Counters("Test Session" & "KeyPress Events")
        .CounterValue = .CounterValue + 1
    End With
End Sub

Private Sub Form_KeyUp(KeyCode As Integer, Shift As Integer)
    With vistaStat.Counters("Test Session" & "KeyUp Events")
        .CounterValue = .CounterValue + 1
    End With
End Sub

Private Sub Form_Load()
    With vistaStat.Counters
        .Add "Test Session", "Activate Events"
        .Add "Test Session", "Click Events"
        .Add "Test Session", "DblClick Events"
        .Add "Test Session", "Deactivate Events"
        .Add "Test Session", "DragDrop Events"
        .Add "Test Session", "DragOver Events"
        .Add "Test Session", "GotFocus Events"
        .Add "Test Session", "KeyDown Events"
        .Add "Test Session", "KeyPress Events"
        .Add "Test Session", "KeyUp Events"
        .Add "Test Session", "LostFocus Events"
        .Add "Test Session", "MouseDown Events"
        .Add "Test Session", "MouseMove Events"
        .Add "Test Session", "MouseUp Events"
        .Add "Test Session", "Paint Events"
        .Add "Test Session", "QueryUnload Events"
        .Add "Test Session", "Resize Events"
        .Add "Test Session - txtText1", "GotFocus Events"
        .Add "Test Session - txtText1", "LostFocus Events"
        .Add "Test Session - txtText2", "GotFocus Events"
        .Add "Test Session - txtText2", "LostFocus Events"
     End With
End Sub

Private Sub Form_LostFocus()
    With vistaStat.Counters("Test Session" & "LostFocus Events")
        .CounterValue = .CounterValue + 1
    End With
End Sub

Private Sub Form_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As
Single)
    With vistaStat.Counters("Test Session" & "MouseDown Events")
        .CounterValue = .CounterValue + 1
    End With
End Sub

Private Sub Form_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As
Single)
    With vistaStat.Counters("Test Session" & "MouseMove Events")
        .CounterValue = .CounterValue + 1
    End With
End Sub

Private Sub Form_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As
Single)
    With vistaStat.Counters("Test Session" & "MouseUp Events")
        .CounterValue = .CounterValue + 1
    End With
End Sub

Private Sub Form_Paint()
    With vistaStat.Counters("Test Session" & "Paint Events")
        .CounterValue = .CounterValue + 1
    End With
End Sub

Private Sub Form_QueryUnload(Cancel As Integer, UnloadMode As Integer)
    With vistaStat.Counters("Test Session" & "QueryUnload Events")
        .CounterValue = .CounterValue + 1
    End With
End Sub

Private Sub Form_Resize()
    With vistaStat.Counters("Test Session" & "Resize Events")
        .CounterValue = .CounterValue + 1
    End With
End Sub

Private Sub txtText1_GotFocus()
    With vistaStat.Counters("Test Session - txtText1" & "GotFocus Events")
        .CounterValue = .CounterValue + 1
    End With
End Sub

Private Sub txtText1_LostFocus()
    With vistaStat.Counters("Test Session - txtText1" & "LostFocus Events")
        .CounterValue = .CounterValue + 1
    End With
End Sub

Private Sub txtText2_GotFocus()
    With vistaStat.Counters("Test Session - txtText2" & "GotFocus Events")
```

```
        .CounterValue = .CounterValue + 1
    End With
End Sub

Private Sub txtText2_LostFocus()
    With vistaStat.Counters("Test Session - txtText2" & "LostFocus Events")
        .CounterValue = .CounterValue + 1
    End With
End Sub
```

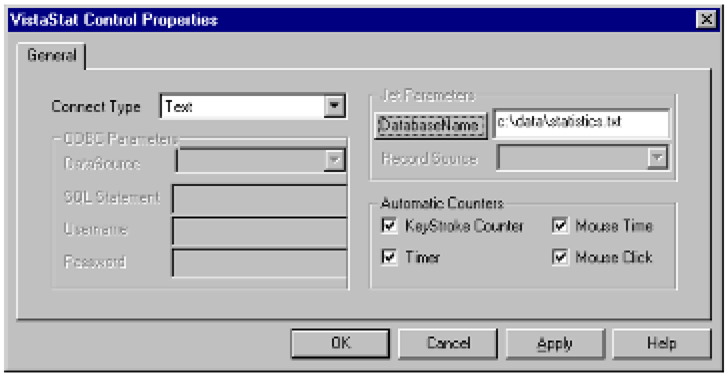
VistaStat Control Properties

Connect Type

You can specify a Connect Type by clicking in the down arrow at the end of the Connect Type drop-down list box. Then select the desired database type.

There are three categories of Connect Types, as described below:

Text: This connect type category is typically used when either Jet or ODBC are not installed on your computer. This means that the statistics are saved in a comma-delimited flat ASCII text file. The name of the file can be specified in the Database Name field (described later in this section of the manual). The following is an example of the VistaStat Control Properties dialog with a Text Connect Type selected.



Connect Type: Text

DatabaseName: c:\data\statistics.txt

☑ KeyStroke Counter ☑ Mouse Time

☑ Timer ☑ Mouse Click

ODBC: This type of connection is used for a network database. When this type of connection is selected, you must also specify the appropriate ODBC parameters (described later in this section of the manual).

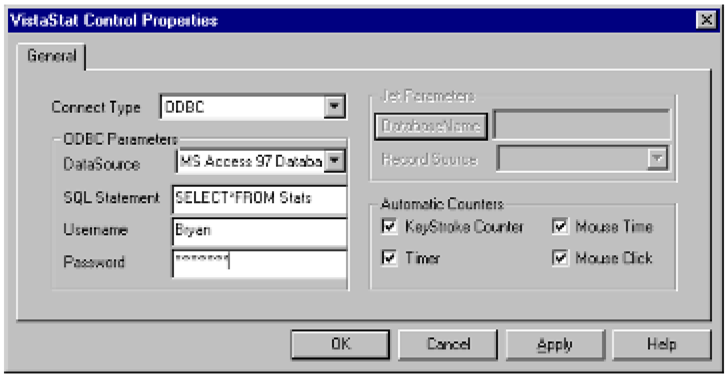
Jet: This type of connection is used for a local database. You can select, for example, Access, dBase or Excel. The output will then be appropriately formatted for use by the selected Jet engine. When this type of connection is selected, you must also specify the appropriate Jet parameters (described later in this section of the manual).

ODBC Parameters

This portion of the VistaStat Control Properties screen is only active when the ODBC connect type is specified.

DataSource	This field is used to specify the server on which your VB application database is to be located. The appropriate datasource must be installed on your computer.
SQL Statement	This field is used to specify the table to which you want the statistics written. This table must be created by the System Administrator.
Username	Use of this field depends on the security setup of your particular site.
Password	Use of this field depends on the security setup of your particular site.

The following is an example of the VistaStat Control Properties dialog with the ODBC connect type selected:



VistaStat Control Properties

General

Connect Type: ODBC

ODBC Parameters
- DataSource: MS Access 97 Databa
- SQL Statement: SELECT*FROM Stats
- Username: Bryan
- Password: *******

Jet Parameters
- DatabaseName
- Record Source

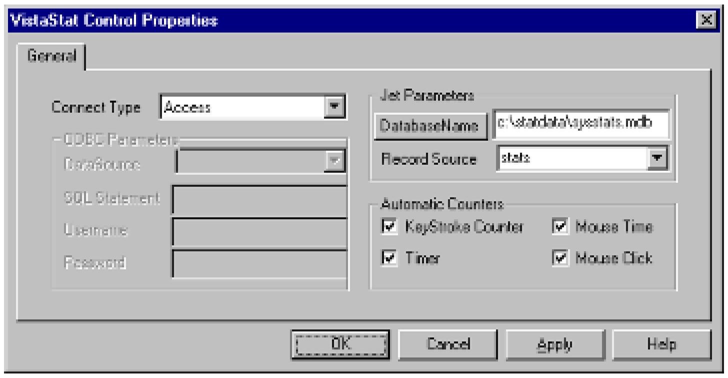
Automatic Counters
- [✓] KeyStroke Counter
- [✓] Mouse Time
- [✓] Timer
- [✓] Mouse Click

Jet Parameters

This portion of the screen is only active when a Jet connect type (e.g. Access, dBase, FoxPro or Excel) is specified.

DatabaseName	This field is used to specify the path and filename of your VB application database.
Record Source	This field is used to specify the name of the table to which you want the statistics written.

The following is an example of the VistaStat Control Properties dialog with a Jet connect type selected. For this example, Access is shown.



VistaStat Control Properties

General

Connect Type: Access

ODBC Parameters
DataSource:
SQL Statement:
Username:
Password:

Jet Parameters
DatabaseName: c:\statdata\sysstats.mdb
Record Source: stats

Automatic Counters
[✓] KeyStroke Counter
[✓] Mouse Time
[✓] Timer
[✓] Mouse Click

Counters

Predefined Counters

VistaStat allows you to easily use up to four automatic or predefined counters in your application. To select a counter, simply click on the associated checkbox on the VistaStat Control Properties dialog. When a predefined counter is active, it is automatically incremented when the counter event occurs and does not require an explict **Item** method call to increment or decrement the CounterValue. Refer to the **Technical Reference** section of this User's Guide for more information on defining counters in your application.

The following is a description of the predefined VistaStat counters:

Keystroke Counter

The Keystroke counter increments for every keystroke. The developer has the ability to add or subtract to this counter if necessary. For example using the KeyPress event keystrokes can be decremented from this counter when a specific key is pressed. The Shift, Alt and Ctrl keys are not counted as keystrokes.

One or more Keystroke counters can can be started using the Add method with CounterName set to Keystroke, or a global application counter can be set using the KeyStroke property at design time. If the KeyStroke property is set at design time the counter is automatically started when the application is initiated. The SessionID property is set to an empty string in this case.

MouseTime Counter

The MouseTime counter adds up the total time the mouse was in use during a SessionID. This includes both mouse movements and mouse button pushes. When using this counter the CounterValue property contains the total time the mouse was used in milliseconds.

Individual MouseTime counters can be started using the Add method with CounterName set to MouseTime, or a global application MouseTime counter can be set at design time using the MouseTime property. If the MouseTime property is set at design time the counter is automatically started when the application is initiated. The SessionID property is set to an empty string in this instance.

The Mouse Time counters accumulate as long as the mouse is moving.

MouseClicks Counter

The MouseClicks counter adds up the total number of times a mouse button was pushed during a SessionID.

One or more MouseClicks counters can be started using the Add method with CounterName set to MouseClicks, or a global application MouseClicks counter can be started by setting the Mouseclicks property at design time. If the MouseClicks property is set at design time the counter is automatically started when the application is initiated. The SessionID property is set to an empty string in this case.

Timer Counter

The Timer counter adds up the total number of times a mouse button was pushed during a SessionID.

Individual counters can be started using the Add method with CounterName set to Timer, or a system counter can be started by setting the Timer property at design time. If the Timer property is set at design time the counter is automatically started when the application is initiated. The SessionID property is set to an empty string in this case.

The Timer counter accumulates as long as the application is running.

Custom Counters

In addition to the predefined counters for VistaStat, you can create custom counters that capture operator and system performance information. There are no limits to the number of counters that can be used.

The VistaStat control contains a collection of counter objects, each of which can be referenced by its index or key. A performance counter is created for every counter object that is added. This means the database containing the output records from VistaStat will contain a counter (and the accumulated data) for every defined counter object.

The Technical Reference section of this User's Guide contains detailed information on the properties, events and methods for creating custom counters.

Notes on Keystrokes and Controls

- The <Shift>, <Ctrl>, and <Alt> keys in combination with another key will only be recorded as a single keystroke.
- The <Insert>, <CapsLock>, <NumLock> and <ScrollLock> keys can be toggled on and off, and will be recorded as a single keystroke.
- The KeyStroke, Mouse Click and Mouse Time counters are only incremented when you are within the application using VistaStat.
- Using the keyboard (instead of the mouse) in your application to perform such functions as selecting a Command Button or clicking a checkbox, causes the KeyStroke counter to increment by the number of keys pressed.
- The Mouse Click counter is affected by the use of the left or right mouse buttons.
- When using items such as scrollbars or listboxes, as long as the mouse button is held down, the counter is only incremented by one.

The following table shows how some of the standard Visual Basic components that may be used in your application affect the KeyStroke and/or Mouse Click counters. A "Yes" indicates that the counter is incremented.

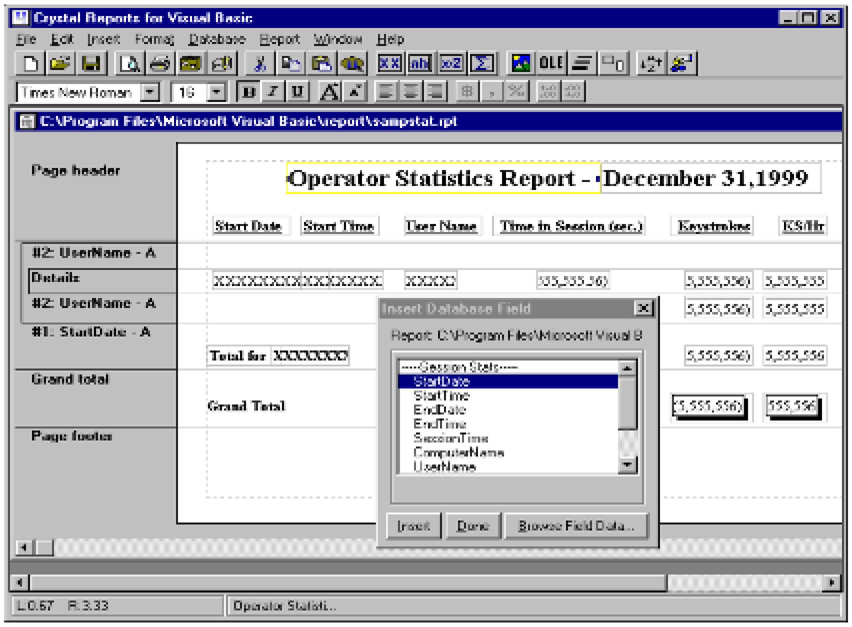
VB Control	**KeyStroke**	**Mouse Click**
PictureBox	No	No
Label	No	No
TextBox	Yes	Yes
Frame	No	No
CommandButton	Yes	Yes
Checkbox	Yes	Yes
OptionButton	Yes	Yes
ComboBox	Yes	Yes
ListBox	Yes	Yes
VScrollBar	No	Yes
HScrollBar	No	Yes
DriveListBox	Yes	Yes
DirListBox	Yes	Yes
FileListBox	Yes	Yes
Data	Yes	Yes
SSTab	Yes	Yes
RichTextBox	Yes	Yes
TabStrip	Yes	Yes

Toolbar	No	Yes
StatusBar	No	Yes
ProgressBar	No	Yes
TreeView	Yes	Yes
ListView	Yes	Yes
Slider	Yes	Yes
DBList	Yes	Yes
DBCombo	Yes	Yes
DBGrid	Yes	Yes
MSRDC	Yes	Yes

Reporting from VistaStat

The data captured by VistaStat is stored in a standard database output record (see the Statistics Record Format section). VistaStat can use all databases supported through Microsoft's 32-bit Jet interface and those supported by ODBC (except Borland's Paradox). The statistics database can be easily queried to generate reports using report writers such as Crystal Reports, ReportSmith, InfoReports, R&R Report Writer. The data can also be graphed using tools like ChartFX, Pinnacle's Graphics Server or Microsoft Excel.

The following image shows the Crystal Report 3.0 layout used to create the sample operator statistics report displayed on the following page.



The following report is an example operator statistics reports which presents individual operator keystroke rates per day and the combined average operator keystroke rates.

Operator Statistics Report - January 31, 1997

Start Date	Start Time	User Name	Time in Session (sec.)	Keystrokes	KS/Hr
1997/01/30	20:28:33.474	Dwight	55.18	216	14,091
	20:52:19.044		83.69	225	9,678
	22:44:32.156		156.25	419	9,654
			295.12	860	10,490
	20:59:14.721	Jeff	1,018.01	4,972	17,582
	22:31:09.371		825.38	1,700	7,414
			1,843.39	6,672	13,029
	20:58:36.826	Reena	1,049.74	1,415	4,852
	22:32:19.691		639.27	985	5,546
			1,689.01	2,400	5,115
Total for	**1997/01/30**		3,827.52	9,932	*9,342*

1997/01/31	15:34:44.251	Jeff	350.23	651	6,691
	20:08:58.221		610.57	1,422	8,384
			960.80	2,073	7,767
	15:33:03.596	Reena	304.03	415	4,913
	20:08:30.911		582.56	1,496	9,244
			886.60	1,911	7,759
Total for	**1997/01/31**		1,847.39	3,984	7,764
Grand Total			5,674.92	13,916	8,828

This sample report is based upon data generated from the sample application. This report shows the count of the various events that were captured using the customer counters created in the sample application.

January 31, 1997 - Event Totals

<u>CounterName</u>	**CounterValue**
Activate Events	11
Click Events	24
DblClick Events	7
Deactivate Events	0
DragDrop Events	0
DragOver Events	0
GotFocus Events	512
KeyDown Events	15975
KeyPress Events	12266
KeyStrokes	13916
KeyUp Events	13231
LostFocus Events	512
MouseClicks	272
MouseDown Events	26
MouseMove Events	2269
MouseTime	339
MouseUp Events	33
Paint Events	151
QueryUnload Events	0
Resize Events	256
Timer	5675

Many other reports can be created. Other possibilities include keystrokes per application, average mouse usage per user, transactions updated per computer, documents scanner per scan session and many more.

Technical Reference

Technical Overview

Purpose

VistaStat is a 32-bit OLE control which enables an application to capture many types of performance data. This includes both operator and system performance information. There are no limits to the number of counters that can be used, both predefined or developer-defined counters. The data to be captured is stored in a standardized database output record that can be easily queried and reported upon. At this time, all databases supported through the Microsoft Jet 3.0 32-bit interface as well as those supported by ODBC are available with the exception of **Paradox**.

Description

The VistaStat control contains a collection of Counter objects, each of which can be referred to by its index or key. A performance counter is created every time a counter object is added. A counter object is stopped when it is removed or cleared. At this time the output data record is written to the database. There can be as many counters running at one time as the developer desires. Using the defined output file a report program can filter through the desired records and create performance reports.

User defined counters can be used for timing various events, such as adding how many times an event occurs or calculating how many events occur within a time period. Other ideas include how long it takes to ICR a page, how many times a particular field was entered, how long it takes someone to key a particular form, timing how long the focus was off the application and how many documents are processed during a session.

Class Name & Icon

VistaStat

Properties, Events and Methods

The properties, events and methods for this control are listed in the following tables. Properties and events that apply only to this control, or require special consideration are marked with an asterisk (*) and are documented in the following sections. See the Visual Basic *Language Reference* or Help for documentation on the non-asterisked properties, events, and methods. This document describes the custom control properties, events and methods that pertain to this control.

VistaStat Control

Custom Properties

Property Name	**Design-Time**	**Run-Time**	
		Read	**Write**
*Connect	X	X	X
*DatabaseName	X	X	X
*DataSourceName	X	X	X
*KeyStroke	X	X	
*MouseClicks	X	X	
*MouseTime	X	X	
*Password	X		X
*RecordSource	X	X	X
*SQL	X	X	X
*Timer	X	X	
*UserName	X	X	X

Standard Properties

Index	Parent	Name	Tag

Events

There are no events for this control.

Methods

AboutBox	*Close	*Open

Counters Collection

Custom Properties

Property Name	Design-Time	Run-Time	
		Read	Write
*Count		X	

Events

There are no events for this collection.

Methods

*Add	*Clear	*Item	*Remove

Counter Object

Custom Properties

Property Name	Design-Time	Run-Time	
		Read	Write
*CounterName		X	
*CounterValue		X	X
*Index		X	
*Key		X	
*SessionID		X	
*TotalTime		X	

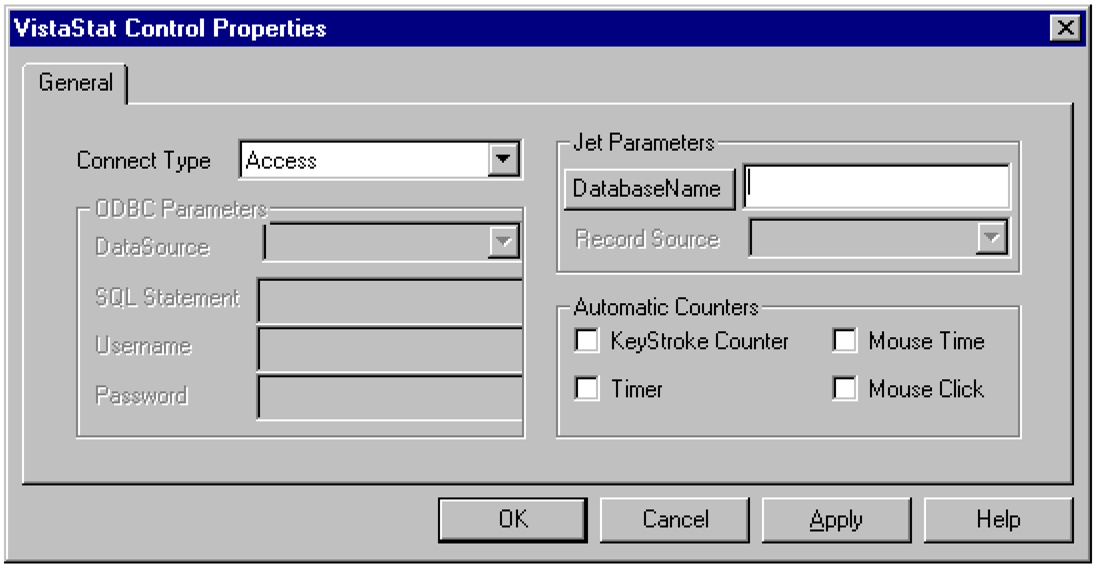
Events

There are no events for this object.

Methods

There are no methods for this object.

Custom Property Screen



This screen allows the design-time setting of the database properties and the automatic or predefined counters.

Connect Property

Description	Sets the connection type for the statistics database. This property is used for DAO and ODBC access to databases.
Syntax	*[form.]*VistaStat.**Connect** [= value]
Remarks	The **Connect** property contains the type of data source being used. ISAM data sources like Access, Foxpro, and dBase, can be used. If any of these are specified, the DAO properties are used to open the data source as a Jet database. If the Connect property contains ODBC, the specified ODBC properties are used to "attach" the external data source into a temporary Access table. This increases performance over direct connections to ODBC and allows the same code to work for ODBC data sources and Jet 3.0 data sources.
	This property is used by the **Open** method.

DataSourceName Property

Description	Sets the ODBC datasource name for the statistics database. This property is used for ODBC access to databases. ODBC is used for SQL Server, Oracle and other client/server database managers.
Syntax	*[form.]*VistaStat.**DataSourceName** [= value]
Remarks	DataSourceName specifies the name of the ODBC data source. This datasource must be installed using the ODBC icon in Control Panel. This property is used by the **Open** method.

SQL Property

Description	Sets the ODBC SQL statement for the statistics database. This property is used for ODBC access to databases. ODBC is used for SQL Server, Oracle 7 and other client/server database managers.
Syntax	*[form.]*VistaStat.**SQL** [= value]
Remarks	SQL contains the SELECT command necessary to return a Resultset for use as a table. To increase the speed of the connection to the database specify a SQL statement that returns zero rows. This is easily accomplished by including a “WHERE False” clause in the SQL statement.
	This property is used by the **Open** method.

UserName Property

Description	Sets the ODBC user name for the statistics database. This property is used for ODBC access to databases. ODBC is used for SQL Server, Oracle 7 and other client/server database managers.
Syntax	*[form.]*VistaStat.**UserName** [= value]
Remarks	UserName contains the name of the user which has access to the datasource on the ODBC server.
	This property is used by the **Open** method.

Password Property

Description	Sets the ODBC password for the statistics database. This property is used for ODBC access to databases. ODBC is used for SQL Server, Oracle 7 and other client/server database managers.
Syntax	*[form.]*VistaStat.**Password** [= value]
Remarks	Password contains the password which corresponds to the user name and allows access to the datasource on the ODBC server.
	This property is used by the **Open** method.

DatabaseName Property

Description	Sets the DAO DatabaseName for the statistics database. This property is used for DAO access to databases. DAO is used for Microsoft Access and Jet 3.0 ISAM tables.
Syntax	*[form.]*VistaStat.**DatabaseName** [= value]
Remarks	DatabaseName contains the actual location of the Access or Jet 3.0 database file. Microsoft Access databases require the file extension. dBase, Btrieve, Foxpro and other file-based ISAMs do not. This property is used by the **Open** method.

RecordSource Property

Description	Sets the DAO table name for the statistics database. This property is used for DAO access to databases. DAO is used for Microsoft Access and Jet 3.0 ISAM tables.
Syntax	*[form.]*VistaStat.**RecordSource** [= value]
Remarks	RecordSource specifies which table is to be used from the Access or Jet 3.0 database.
	This property is used by the **Open** method.

KeyStroke Property

Description	Sets the predefined Keystroke counter object ON or OFF.
Remarks	The **KeyStroke** property is a design-time property. When this property is set to True, the predefined Keystroke counter is automatically started when the **VistaStat** object is constructed.

Timer Property

Description	Sets the predefined Timer counter object ON or OFF.
Remarks	The **Timer** property is a design-time property. When this property is set to True, the predefined Timer counter is automatically started when the **VistaStat** object is constructed.

MouseTime Property

Description	Sets the predefined MouseTime counter object ON or OFF.
Remarks	The **MouseTime** property is a design-time property. When this property is set to True, the predefined MouseTime counter is automatically started when the **VistaStat** object is constructed.

MouseClicks Property

Description	Sets the predefined MouseClicks counter object ON or OFF.
Remarks	The **MouseClicks** property is a design-time property. When this property is set to True, the predefined MouseClicks counter is automatically started when the **VistaStat** object is constructed.

Counters Collection Properties

Count Property

Description	Returns the number of VistaStat Counter objects in the Counters collection.
Syntax	*[form.]*VistaStat.Counters.**Count**
Remarks	This is a run-time read-only property.

Counter Object Properties

CounterName Property

Description	Returns the name of the **Counter** object from the **Counters** collection.
Syntax	*object*.**CounterName**
	The *object* place holder represents an object expression that evaluates to a **Counter** object.
Remarks	The **CounterName** is a run-time, read-only property and is set by the **Add** method when a **Counter** object is added to a **Counters** collection.

CounterValue Property

Description	Returns or sets the current value of a **Counter** object from the **Counters** collection.
Syntax	*object*.**CounterValue** [= value]
	The *object* place holder represents an object expression that evaluates to a **Counter** object.
Remarks	This is a run-time property. A VistaStatGet function call is made to retrieve this property.

Index Property

Description	Returns the number that uniquely identifies a **Counter** object in the **Counters** collection.
Syntax	*object*.**Index**
	The *object* place holder represents an object expression that evaluates to a **Counter** object.
Remarks	This is a run-time read-only property.

Key Property

Description	Returns a string that uniquely identifies a **Counter** object in a **Counters** collection.
Syntax	*object*.**Key**
	The *object* place holder represents an object expression that evaluates to a **Counter** object.
Remarks	The Key property is a run-time, read-only property set by the Add method to add a **Counter** object to a **Counters** collection. The string defining the property is a concatenation of the SessionID and CounterName properties. If the string is not unique, an error will occur.

SessionID Property

Description	Returns the identifier for a **Counter** object categorization.
Syntax	*object*.**SessionID**
	The *object* place holder represents an object expression that evaluates to a **Counter** object.
Remarks	SessionID is a string identifier for the categorizing of counters. It can be a combination of various sorting criteria. (i.e. batch/folder name, field id, form type, record type, etc.)
	The **SessionID** is a run-time, read-only property and is set by the **Add** method when a **Counter** object is added to a **Counters** collection.

TotalTime Property

Description	Returns the TotalTime in seconds from the time a **Counter** object was added.
Syntax	*object*.**TotalTime**
	The *object* place holder represents an object expression that evaluates to a **Counter** object.
Remarks	Thisis a run-time, read-only property. It is returned as a floating point value in the form of ss.nnn. (ss = seconds, nnn = milliseconds.) A VistaStatGet function call is made to retrieve this property.

VistaStat Methods

Open Method

Description	Opens the statistics database in order to store the counter records. If the table does not exist, it is automatically created.
Syntax	*[form.]*VistaStat.**Open**
Remarks	The VistaStat object performs the Open method at instantiation to open the database specified at design time. If at runtime the statistics table should be changed, the Close method must be performed, the database properties must be changed, and then the Open method should be performed to use the new database property values. No developer or predefined counters are preserved by the Open and Close methods. All counters must be restarted once these methods have been performed. Uses of this method include departmentalized statistics, statistics stored in tables by month, or even batch specific tables.

Close Method

Description	Closes the statistics database previously opened by the Open method. All counters are cleared by this method.
Syntax	*[form.]*VistaStat.**Close**
Remarks	The VistaStat object performs the Close method when the object is destroyed to close the current statistics database. If at runtime the statistics table should be changed, the Close method must be performed, the database properties must be changed, and then the Open method should be performed to use the new database property values. No developer or predefined counters are preserved by the Open and Close methods. All counters must be restarted once these methods have been performed. Uses of this method include departmentalized statistics, statistics stored in tables by month, or even batch specific tables.

Add Method

Description	Start capturing statistics data for the named session and counter.
Syntax	*[form.]*VistaStat.Counters.**Add** *SessionID, Counter*
Remarks	Creates a counter object and calls the VistaStatStart() routine in the VistaStat DLL to create a new statistics record.

The **Add** method syntax has these parts:

Part	**Description**
SessionID	Required. The name identifying the capture session. It can be a combination of various sorting criteria. (i.e. batch/folder name, field id, form type, record type, etc.)
Counter	Required. The name identifying the data element being captured.

Remove Method

Description	Stops capturing statistics for a specific counter instance and writes the data to the output file.
Syntax	*[form.]*VistaStat.Counters.**Remove** *index*
Remarks	The Remove method clears the identified counter object from the collection. It calls the VistaStatStop routine in the VistaStat DLL to write the statistics record.

The **Remove** method syntax has these parts:

Part	**Description**
index	Required. An expression that specifies the position of a member of the collection. If a numeric expression, index must be a number from 0 to the value of the collection's Count property. If a string expression, index must correspond to the key generated when the member being referred to was added to the collection. This key is the combination of the SessionID and CounterName properties.

Clear Method

Description	Stops capturing statistics for all counter instances and writes the data to the output file.
Syntax	*[form.]*VistaStat.Counters.**Clear**
Remarks	The **Clear** method removes all the counter objects from the counters collection. For each of the **Counter** objects in the **Counters** collection the VistaStatStop routine in the VistaStat DLL is called to write a statistics record. Any counters remaining active when a form is unloaded are implicitly cleared when the VistaStat object is destroyed.

Item Method

Description	Returns a specific item of a collection object by position, index, or key.
Syntax	*[form.]*VistaStat.Counters.**Item**(index)
	*[form.]*VistaStat.Counters(index)
Remarks	If the value provided as index does not match any existing member of the collection, an error occurs. The **Item** method is the default method for a collection and has these parts:

Part	Description
index	An integer or unique string that specifies a member of the collection. The integer must be a number from 1 to the value of the collection's Count property. The string must correspond to the key generated when the member was added to the collection. This key is the combination of the SessionID and CounterName properties.

The primary use of this method is to change the value of an active counter. For example the following code increments the value of a counter by one.

```
VistaStat.Counter(index).CounterValue = _
       VistaStat.Counter(index).CounterValue + 1
```

Another use is to retrieve the current CounterValue property or current TotalTime property. For example the following code retrieves the current TotalTime

```
CurrentSeconds = VistaStat.Counter(index).TotalTime
```

Statistics Record Format

The statistics output data file is a fixed length record written to a data source independent database. The statistics file output structure is as follows:

Field	Max Length	FieldName	dBase/FoxPro FieldName	Field Type
Start Date	10	StartDate	STARTDATE	Text
Start Time	12	StartTime	STARTTIME	Text
Stop Date	10	EndDate	ENDDATE	Text
Stop Time	12	EndTime	ENDTIME	Text
Total Session time	20	SessionTime	SESSIONTIM	Currency
Computer name	255	ComputerName	COMPUTERNA	Text
Username	255	UserName	USERNAME	Text
Application name	255	ApplicationName	APPLICATIO	Text
SessionID	255	SessionID	SESSIONID	Text
Counter Name	40	CounterName	COUNTERNAM	Text
Counter value	20	CounterValue	COUNTERVAL	Currency

Start Date is the system date on which the counter was added. This field is stored in “yyyy/mm/dd” format.

Start Time is the system time in GMT at which the counter was added. This field is stored in “HH:MM:SS.NNN” format.

Stop Date is the system date on which the counter was removed or cleared. This field is stored in “yyyy/mm/dd” format.

Stop Time is the system time in GMT at which the counter was removed or cleared. This field is stored in “HH:MM:SS.NNN” format.

Total Session Time is the difference between the start date/time and stop date/time. This value is stored as a Currency data type. The Currency type provides a fixed decimal format with 4 decimal positions. VistaStat retains only three significant decimal digits. This value is stored as a number of seconds with milliseconds represented after the decimal.

Computer name is the operating system identity for the machine where the counter was added.

Username is the operating system identifier for the user which added the counter.

Application name is the operating system identifier for the application which added the counter.

SessionID is an identifier for the categorizing of counters. It can be a combination of various sorting criteria. (i.e. batch/folder name, field id, form type, record type, etc.)

Counter Name is the application supplied identifier for this counter.

Counter Value is the numeric amount for this counter. This value is stored as a Currency data type.

VistaStat Error Messages

Message	Message Number	Description
NoConnectType	1001	Returned from a call to Open, when No connect type has been specified
DBNotOpen	1002	Returned from a call to Close when the database has not been opened
CounterNotFound	1003	Returned when a SessionID and CounterName do not specify a counter that is active in this VistaStat Control.
DuplicateCounter	1004	Returned when the SessionID and CounterName in an Add operation specify a counter that is currently active in this VistaStat Control.
DatabaseOpen	1005	Returned from a call to Open when the database is already open.
DatabaseCreate	1006	Returned from a call to Open when the database does not exist and cannot be created.
DatabaseReadOnly	1007	Returned from a call to Open when the database can only be opened Read-Only.
DatabaseTable	1008	Returned from a call to Open when the specified Database Table is invalid.
DatabaseField	1009	Returned from a call to Open when a field in the database cannot be found or created.
DatabaseWrite	1010	Currently Not Used
NoDatabase	1011	Currently Not Used
NoRecordSource	1012	Currently Not Used
NoSQL	1013	Currently Not Used
NoDataSource	1014	Currently Not Used
InvalidOperation	1015	Returned from VistaStatGet (internally, and in DLL) when an invalid operation is specified.

VistaStat DLL Function Calls

VistaStatOpen

Description	Creates a connection to an Access, Jet 3.0, or ODBC table for storing generated statistics records.
Syntax	Retval = VistaStatOpen(connect, data name, record source/SQL, username, password)
Remarks	This function returns 0 if successful, or the DAO error number if an error occurs. The parameters define the database or external datasource to be opened. This function opens Access and Jet 3.0 databases directly and attaches ODBC datasources to temporary Access tables. In this way VistaStatOpen has only one type of database functionality and the speed of ODBC datasources is improved. Connect contains the type of database being used. This is either Access, a Jet 3.0 ISAM, or ODBC. The data name parameter specifies the database name when Connect is Access or a Jet 3.0 ISAM. Datasource name is specified in this parameter when Connect is set to ODBC. The record source/SQL parameter specifies the table name in the database when Connect is Access or a Jet 3.0 ISAM. This parameter contains the ODBC SQL command if Connect is set to ODBC. Username and Password are for use only with an ODBC Connect type. **For Microsoft Jet 3.0 compatible databases:** If the database does not exist, it is created. If the RecordSource does not exist in the database, it is created. If the fields are not found in the RecordSource, they are added. No previously defined information is lost by using this operation.

VistaStatClose

Description	Closes the database opened by VistaStatOpen.
Syntax	Retval = VistaStatClose()
Remarks	Returns 0 if successful, or the DAO error code if an error occurs. This function closes the Access, Jet 3.0, or ODBC table opened by VistaStatOpen.

VistaStatStart

Description	Starts a statistical counter with a given SessionID and counter name. This function allocates storage in memory for the start date/time, stop date/time, user name, computer name application, SessionID, counter name, total seconds, and counter value. No database access is performed by this function.
Syntax	Retval = VistaStatStart(SessionID, counter name)
Remarks	VistaStatStart returns an integer value of 0 if successful or 1 if an error occurs. The SessionID and counter name combination must be unique for each counter added. If the SessionID and counter name combination refer to an active counter, an error occurs. SessionID is a NULL terminated string of up to 255 characters. Counter name is a NULL terminated string of up to 40 characters.

VistaStatStop

Description	Stops a statistical counter with a given SessionID and counter name. A new record is added to the VistaStat database and all in memory values are written to the new record. This function releases storage in memory allocated by VistaStatStart.
Syntax	Retval = VistaStatStop(SessionID, counter name)
Remarks	VistaStatStop returns an integer value of 0 if successful or 1 if an error occurs. If the SessionID and counter name combination do not refer to an active counter, an error occurs. SessionID is a NULL terminated string of up to 255 characters. Counter name is a NULL terminated string of up to 40 characters. After completing this function, the SessionID and counter name combination no longer refer to an active counter.

If both SessionID and CounterName are NULL, all counters will be stopped at once then written to the database. This allows all counters to have valid times no matter how long it takes to write the records to the database.

VistaStatAdd

Description	Increments or decrements the value of an active statistical counter with a given SessionID and counter name. The value is changed in memory. No database access is performed by this function.
Syntax	Retval = VistaStatAdd(SessionID, counter name, currency pointer)
Remarks	VistaStatAdd returns an integer value of 0 if successful or 1 if an error occurs. If the SessionID and counter name combination do not refer to an active counter, an error occurs. SessionID is a NULL terminated string of up to 255 characters. Counter name is a NULL terminated string of up to 40 characters. Currency pointer is a pointer to a Currency data storage area. The Currency object pointed to by this pointer must contain the amount by which the value is to be changed. If the change is negative, value is decremented. If the change is positive, value is incremented.

VistaStatSet

Description	Sets the value of an active statistical counter with a given SessionID and counter name. The value is changed in memory. No database access is performed by this function.
Syntax	Retval = VistaStatSet(SessionID, counter name, currency pointer)
Remarks	VistaStatSet returns an integer value of 0 if successful or 1 if an error occurs. If the SessionID and counter name combination do not refer to an active counter, an error occurs. SessionID is a NULL terminated string of up to 255 characters. Counter name is a NULL terminated string of up to 40 characters. Currency pointer is a pointer to a Currency data storage area. The Currency object pointed to by this pointer must contain the amount to which the value is to be set.

VistaStatGet

Description	Retrieves the total time in milliseconds or the value of an active statistical counter with a given SessionID and counter name. The value returned is retrieved from memory. No database access is performed by this function.
Syntax	Retval = VistaStatGet(SessionID, counter name, property, currency pointer)
Remarks	VistaStatGet returns a long integer which indicates whether or not an error occurred. If an error occurs, a value of -1 is returned. If the SessionID and counter name combination do not refer to an active counter, an error occurs. SessionID is a NULL terminated string of up to 255 characters. Counter name is a NULL terminated string of up to 40 characters. Property is a single character which identifies the value to return: "T" for total milliseconds, and "V" for counter value. Currency pointer is a pointer to a Currency data storage area. The Currency object pointed to by this pointer is updated with the value of the number of milliseconds or the counter value.

Index

Error! No index entries found.