

Version

6.0



BAR-ONE[®]

Programmer's Guide

English

The information in this documentation is not contractual in nature. It is subject to modification without notice.

The software described in this manual is supplied under a user license. Its use, duplication, or reproduction on any media whatsoever, except as provided for under the terms of the license, is not authorized.

Windows™ is a registered trademark of **Microsoft® Corporation**.

IBM, PC, AT PS/2 are registered trademarks of **International Business Machines, Inc.**

Table of Contents

About this manual	vii
Welcome!	vii
The Programmer's Guide	vii
Typographical conventions	viii



Discover ActiveX Automation for your labeling software	Chapter 1 - 1
Introduction	Chapter 1 - 1
What is an ActiveX object?	Chapter 1 - 2
What is the type library?	Chapter 1 - 3
Mechanisms	Chapter 1 - 4
Server Activation	Chapter 1 - 4
Create Object function	Chapter 1 - 4
GetObject function	Chapter 1 - 5
New function	Chapter 1 - 6
Server Deactivation	Chapter 1 - 7
Quit method	Chapter 1 - 8
Data Type	Chapter 1 - 8
Application Object	Chapter 1 - 8
Document Object	Chapter 1 - 9
Collection Object	Chapter 1 - 9
Event management	Chapter 1 - 11
Handling an Object's Events	Chapter 1 - 11
Connecting a WithEvents variable to an object	Chapter 1 - 13
Compatibility with the previous version	Chapter 1 - 14
Particularity about access rights	Chapter 1 - 14

2

Reference Guide

Chapter 2 - 15

Hierarchy diagram	Chapter 2 - 15
Application Object	Chapter 2 - 17
Object Properties	Chapter 2 - 18
Object Methods	Chapter 2 - 23
PrinterSystem Object	Chapter 2 - 26
Object Methods	Chapter 2 - 26
Options Object	Chapter 2 - 29
Object Properties	Chapter 2 - 29
Dialogs Collection	Chapter 2 - 34
Object Properties	Chapter 2 - 34
Object Methods	Chapter 2 - 35
Dialog Object	Chapter 2 - 36
Object Properties	Chapter 2 - 36
Object Methods	Chapter 2 - 37
RecentFiles Collection	Chapter 2 - 38
Object Properties	Chapter 2 - 38
Object Methods	Chapter 2 - 39
RecentFile Object	Chapter 2 - 41
Object Properties	Chapter 2 - 41
Object Methods	Chapter 2 - 42
Documents Collection	Chapter 2 - 43
Object Properties	Chapter 2 - 43
Object Methods	Chapter 2 - 44
Document Object	Chapter 2 - 46
Object Properties	Chapter 2 - 46
Object Methods	Chapter 2 - 50
Database Object	Chapter 2 - 54
Object Properties	Chapter 2 - 54
Object Methods	Chapter 2 - 56
Printer Object	Chapter 2 - 58
Object Methods	Chapter 2 - 58
Object Properties	Chapter 2 - 60
Format Object	Chapter 2 - 63
Object Properties	Chapter 2 - 63
Object Methods	Chapter 2 - 67
DocumentProperties Collection	Chapter 2 - 68
Object Properties	Chapter 2 - 68
Object Methods	Chapter 2 - 69
DocumentProperty Object	Chapter 2 - 70
Object Properties	Chapter 2 - 70
DocObjects Collection	Chapter 2 - 72
Object Properties	Chapter 2 - 72
Object Methods	Chapter 2 - 74

DocObject Object	Chapter 2 - 76
Object Properties	Chapter 2 - 76
Object Methods	Chapter 2 - 79
Images Collection	Chapter 2 - 80
Object Properties	Chapter 2 - 80
Object Methods	Chapter 2 - 80
Image Object	Chapter 2 - 82
Object Properties	Chapter 2 - 82
Barcodes Collection	Chapter 2 - 84
Object Properties	Chapter 2 - 84
Object Methods	Chapter 2 - 85
Barcode Object	Chapter 2 - 86
Object Properties	Chapter 2 - 86
Code2D Object	Chapter 2 - 93
Object Properties	Chapter 2 - 93
Object Methods	Chapter 2 - 94
Texts Collection	Chapter 2 - 95
Object properties	Chapter 2 - 95
Object Methods	Chapter 2 - 95
Text Object	Chapter 2 - 97
Object Properties	Chapter 2 - 97
Object Methods	Chapter 2 - 99
TextSelection Object	Chapter 2 - 102
Object Properties	Chapter 2 - 102
Object Methods	Chapter 2 - 103
OLEObjects Collection	Chapter 2 - 104
Object Properties	Chapter 2 - 104
Object Methods	Chapter 2 - 104
OLEObject Object	Chapter 2 - 106
Object Properties	Chapter 2 - 106
Object Methods	Chapter 2 - 106
Shapes Collection	Chapter 2 - 108
Object Properties	Chapter 2 - 108
Object Methods	Chapter 2 - 109
Shape Object	Chapter 2 - 113
Object Properties	Chapter 2 - 113
Object Methods	Chapter 2 - 113
Variables Collection	Chapter 2 - 114
Object Properties	Chapter 2 - 114
Object Methods	Chapter 2 - 116
Variable Object	Chapter 2 - 118
Object Properties	Chapter 2 - 118
TableLookups Collection	Chapter 2 - 120
Object Properties	Chapter 2 - 120
Object Methods	Chapter 2 - 120
TableLookup Object	Chapter 2 - 122

Object Properties	Chapter 2 - 122
Object Methods	Chapter 2 - 123
Formulas Collection	Chapter 2 - 125
Object Properties	Chapter 2 - 125
Object Methods	Chapter 2 - 125
Formula Object	Chapter 2 - 127
Object Properties	Chapter 2 - 127
Object Methods	Chapter 2 - 128
Dates Collection	Chapter 2 - 129
Object Properties	Chapter 2 - 129
Object Methods	Chapter 2 - 129
Date Object	Chapter 2 - 131
Object Properties	Chapter 2 - 131
Counters Collection	Chapter 2 - 133
Object Properties	Chapter 2 - 133
Object Methods	Chapter 2 - 133
Counter Object	Chapter 2 - 135
Object Properties	Chapter 2 - 135
FreeVariables Collection	Chapter 2 - 139
Object Properties	Chapter 2 - 139
Object Methods	Chapter 2 - 140
Free Object	Chapter 2 - 141
Object Properties	Chapter 2 - 141
DatabaseVariables Collection	Chapter 2 - 143
Object Properties	Chapter 2 - 143
Object Methods	Chapter 2 - 144
FormVariables Collection	Chapter 2 - 145
Object Properties	Chapter 2 - 145
Object Methods	Chapter 2 - 146
Strings Collection	Chapter 2 - 147
Object Properties	Chapter 2 - 147
Object Methods	Chapter 2 - 147
Document Events	Chapter 2 - 149
Object Methods	Chapter 2 - 149
Application Events	Chapter 2 - 151
Object Methods	Chapter 2 - 151
3 Appendix	Chapter 3 - 152
Information on Visual C++ Data Type	Chapter 3 - 152
4 Index	Chapitre 4 - 155

About this manual

Welcome!

Welcome to the number one Windows based label design and printing software. It provides the simplest, yet highest performance solution for your labeling requirements.

This version of this labeling software integrates the ActiveX technology offering you the possibility to easily create a program to control your labeling software.

The Programmer's Guide

The purpose of this manual is to help you program your own application to control your labeling software. All you should know about using ActiveX with your labeling software is described in this manual. However, to get more information about the ActiveX technology, refer to the Microsoft reference manuals.

The Programmer's Guide is divided into three parts:

- **Discover ActiveX for your labeling software:** this part gives the bases for programming with ActiveX.
- **Reference Guide:** this part gives all the object, method and property definitions integrated by your labeling software.
- **Appendix:** this part gives you information on Visual C++ Data Type.

Typographical conventions This manual distinguishes different types of information by using the following conventions:

- terms taken from the interface itself, such as commands, appear in **bold**;
- keys appear in small caps, as in the following example: "Press the `SHIFT` key";
- numbered lists mean there is a procedure to follow;
- when the conjunction "or" appears next to a paragraph it means there is another procedure available for performing a given task;
- When a menu command contains submenus, the menu name followed by the command to select appears in bold. Thus, "Choose **File Open**" means choose the **File** menu, then the **Open** command.



This symbol provides tips for optimizing certain tasks, speeding up the execution of commands, etc.



This symbol highlights important information about a particular function or procedure.



This symbol highlights an example or an exercise.

Discover ActiveX Automation for your labeling software



Introduction

Using **ActiveX Automation**, you can control almost anything you create with your labeling software — even your labeling software itself.

The ActiveX technology lets you easily integrate your labeling software as a printing module or a designer module in your own organization.

ActiveX is object oriented and all of the ActiveX objects are programmable from any languages such as Visual Basic. So the user can control his labeling software using your own program.

Your labeling software becomes the server and your program is the client application.

Through this manual, you will find examples and references using Visual Basic 6.0.

What is an ActiveX object?

An ActiveX object is an instance of a class that exposes properties, methods, and events to ActiveX clients. ActiveX objects support the COM (Component Object Model) - Microsoft technology. An ActiveX component is an application or library that is able to create one or more ActiveX objects. In this case, your labeling software exposes many objects that you can use to create new applications and programming tools. Within your labeling software, objects are organized hierarchically, with an object named **Application** at the top of the hierarchy (see Chapter 2 - 15: the hierarchy diagram).

Each ActiveX object has its own member function definition. When the member functions are exposed, it makes the object programmable by an ActiveX client. Three types of members can be exposed for an object:

- **Methods** are actions that an object can perform. For example, the **Document** object in your labeling software provides a **Close** method that closes the current document.
- **Properties** are functions that access information about the state of an object. The **Application** object's **Visible** property determines whether the labeling software is visible or not.
- **Events** are actions recognized by an object, such as clicking the mouse or pressing a key. You can write code to respond to such actions. In Automation, an event is a method that is called, rather than implemented, by an object (see also Chapter 1 - 11).

Your labeling software often works with several instances of an object which together make up a **Collection object**. For example, since your labeling software is a multiple-document interface (MDI), it might have multiple documents. To provide an easy way to access and program the documents, your labeling software exposes an object named **Documents**, which refers to all of the already opened document objects. **Documents** is a collection object.

A Collection object lets you work as a group with the objects it manage (see Chapter 1 - 9).

All the methods, properties and events are defined in Chapter 2 - 17: Reference Guide.

What is the type library?

The type library supplied by the labeling software is a file (**Lppx2.tlb**) that describes the type of all the ActiveX objects. The type library does not store objects, it stores type information. By accessing the type library, your application can determine the characteristics of an object, such as the interfaces supported by the object and the names and parameters.

This library helps you to write your program because it contains all the definitions of object methods and properties that you can access. Using this library you optimize your job.

The name of this type library is **LabelManager2** with the **TK Labeling ActiveX 6.0** reference.

The procedures below show how to install and use the type library with Visual Basic 6.0.

► To install the type library

- 1 Choose **Project References**.
- 2 Activate **TK Labeling ActiveX 6.0** in the list of available references then validate the dialog box.

► To display the methods and properties

- 1 Use the **Object Explorer** by pressing the F2 key.
- 2 In the library list, select **LabelManager2**.

► To use the type library

- While writing code, you have just to enter a period "." after an object to get the associated methods and properties, or after a method to get the associated properties.

Mechanisms

Your labeling software offers you two main objects: the **Application** object that is at the top of the hierarchy and the **Document** object. These main objects provide access to the subordinated objects (see Chapter 2 - 15: the hierarchy diagram).

The first step to activate the server is the main object creation, in this case, the **Application** object.

The last step is the deactivation of the server with the **Quit** method.

Server Activation

Several methods are available to create an ActiveX object.

Create Object function This function creates and returns a reference to the **Application** object.

Syntax CreateObject(server name)



Defines an object variable. This object variable is meant to contain the object reference. **Dim as Object** creates a link at execution.

```
Dim MyApp as Object
Set MyApp = CreateObject("Lppx2.Application")
```

This code launches the application that creates the object. In this case, the labeling software. As soon as the object is created, you reference it in the code with the object variable that you have defined, i.e. MyApp.

Note

If you define an object variable with "As Object", a variable containing a reference for any object type is created. However, the access to an object via this variable is realized by a late bind, i.e. the link is created during the execution of your program. To create an object variable that induces an early bind, i.e. a link during the compilation of your program, define the object variable with a specific identifier (see below).



For example, you can define and create the reference using the code below:

```
Dim MyApp As Lppx2.Application
Set MyApp = CreateObject("Lppx2.Application")
```

The variable reference creating an early bind increases the performance but must only contain one reference.

GetObject function

This function returns a reference to an ActiveX object from a file.

Syntax `GetObject([pathname],[server name])`

The syntax of the **GetObject** function includes the following arguments:

Item	Description
pathname	Optional. Variable of Variant type (String). Complete pathname with the name of the file containing the object to get. If you don't define the pathname, you have to define the server name.
servername	Optional. Variable of Variant type (String). Name of the application that gives the object.

Note

Use the **GetObject** function to access an ActiveX object from a file and to assign this object, an object variable. Use the **Set** instruction to assign the object that is returned by the **GetObject** function at the object variable (see below).

Below are several examples showing the variations of the **GetObject** syntax.



```
Dim MyDoc As Object
Set MyDoc = GetObject("c:\ProgramFile\document.lab")
```

When this code is executed, the application associated with the pathname argument is launched and the object included in the file is activated.

Note

In the case where the server automation is already loaded in the system memory, the ActiveX mechanism selects it, then the document is activated.



In the example below, the server name is specified. Use this parameter if you have several versions of your labeling software to open the document.lab with the correct version.

```
Dim MyApp As Object
Set MyApp = GetObject ("c:\ProgramFile\document.lab", "Lppx2.Application")
```



Note that in the example below the Visual Basic for **Application** expression **GetObject(,"Lppx2.Application")** will fail unless the **Application** (the labeling software) is already running. If the **Application** is not already running, a new instance will not be launched.

```
Dim MyApp As Object
Set MyApp = GetObject (,"Lppx2.Application")
```



In this example, the variation of the **GetObject** syntax varies from the previous example in that a new instance of the application will always be launched even if the application is already running. This variation is equivalent to a **CreateObject** statement.

```
Dim MyApp As Object
Set MyApp = GetObject ("" , "Lppx2.Application")
```

New function

The key word **New** can only be used if you work with the Type Library (see: What is the type library).

New assigns an object reference to a variable or to a property.

Syntax

```
Set MyApp = {[New] objectexpression }
```



This example demonstrates reation of the MyApp object that represents **Application**. This is the standard access to get the subordinated objects of **Application**.

```
Dim MyApp As LabelManager2.Application
Set MyApp = New LabelManager2.Application
```

The syntax of the Set instruction contains the following items:

Item	Description
objectvar	Name of the variable or property.
New	Optional. This key word is generally used in the declarations to allow the implicit creation of an object. Used with Set, the New key word creates a new instance of the class. If the objectvar argument contains a reference to an object, this reference is lost when a new association is created.

Note

The objectvar must have an object type compatible with the object to which it is assigned.

Server Deactivation

The last step of your program is the deactivation of the server with the **Quit** method.

To correctly deactivate the server, you must:

- a. Close all the documents with the **CloseAll** method on the **Documents** collection.
- b. Call the **Quit** method of the **Application** object. This method means the process is ended.
- c. Ask Visual Basic to delete the **Application** from the system memory by setting the value of the variable to **Nothing**.

Quit method The **Quit** method is used to end the process. Before using this method, you must close all the documents.



The following example shows how to deactivate the server. Use the **CloseAll** method on the **Documents** collection to close all the documents. Then, use the **Quit** method on the **Application** object to end the process. At the end, the Set instruction delete the **Application** from the system memory.

```
MyApp.Documents.CloseAll False
MyApp.Quit
Set MyApp = Nothing
```

Data Type

There are three data types corresponding to the three main objects: **Application**, **Document** and **Collection**.

Application Object

The **Application** object represents the labeling software. The **Application** object contains the properties and the methods that return the first level objects. For example, the **ActiveDocument** property returns a **Document** object.

► Using the Application object

To return the **Application** object, use the **Application** property. The following sample shows how to display the path defined for the labeling software.



```
Dim MyApp As LabelManager2.Application
Set MyApp = New LabelManager2.Application

MsgBox MyApp.Path
```

Most of the properties and methods that return the common user interface objects, such as the active document (**ActiveDocument** property), can be used without the identifier of the **Application** object by using the **With** keyword.



```
Dim MyApp As LabelManager2.Application
...
    With MyApp
        .ActiveDocument
        .Print
    end With
```

The properties and methods that can be used without the **Application** object are called “global.”

- To display the global properties and methods in the object explorer (F2 key), click on **global** at the beginning of the list displayed in the **Classes** zone.

Document Object

The **Document** object represents an open document. Each open document in the labeling software is represented by a **Document** object. This object has members (properties, methods, and events) that you can use to manipulate the document.

You can access the current document if there is an open document by using the **ActiveDocument** property of the **Application** object.

All open documents that belong to the documents collection are represented by the **Documents** object. You can find a particular document by moving through this collection.

Collection Object

A **Collection** object is an ordered set of items that can be referred to a unit.

Note

The **Collection** object provides a convenient way to refer to a related group of items as a single object. The items, or members, in a collection need only be related by the fact that they exist in the collection. Members or items of a collection don't have to share the same data type (see Chapter 1 - 8).



A collection can be created the same way other objects are created. For example:

```
Dim X As New Collection
```

Once a collection is created, members can be added using the **Add** method and removed using the **Remove** method. Specific members can be returned from the collection using the **Item** method, while the entire collection can be returned using the **For Each...Next** statement.

► Collection methods

Methods for collection are described in the following table. The **Item** method is required; other methods are optional.

Method name	Return type	Description
Add	VT_DISPATCH or VT_EMPTY	Adds an item to a collection. Returns VT_DISPATCH if object is created (object cannot exist outside the collection) or VT_EMPTY if no object is created (object can exist outside the collection).
Item	Varies with type of collection	Returns the indicated item in the collection. Required. The Item method may take one or more arguments to indicate the element within the collection to return. This method is the default member for the collection object.
Remove	VT_EMPTY	Removes an item from a collection. Uses indexing arguments in the same way as the Item method.

The **Item** method takes one or more arguments to indicate the index. Indexes can be numbers or strings.



Because **Item** is the default method, you could write either:

```
MyObject.Item(3).Name
-Or-
MyObject(3).Name
```

► Count Property

Returns a **Long** (long integer) containing the number of objects in a collection. Read-only.

Event management

When a program detects that something has happened, it can notify its clients. For example, if a stock ticker program detects a change in the price of a stock, it can notify all clients of the change. This notification process is referred to as firing an event.

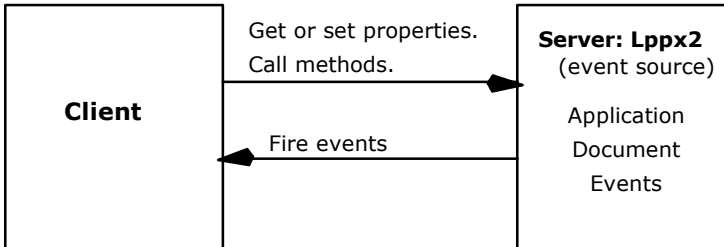


Figure 1 Interaction between the Client and the Labeling software

Handling an Object's Events

An object that triggers events is called an **event source**. To handle the events triggered by an event source, you can declare a variable of the object's class using the **WithEvents** keyword.



For example, to handle the **ProgressPrinting** event of a **Document**, place the following code in the Declarations section:

```
Option Explicit
Private WithEvents MyDoc As LabelManager2.Document
Private mblnCancel As Boolean
```



In this case, the client application must set the **EnableEvents** property of the application to True in order to trigger the events.

The **WithEvents** keyword specifies that the variable **MyDoc** will be used to handle an object's events. You specify the kind of object by supplying the name of the class from which the object will be created.

The variable **MyDoc** is declared in the Declarations section because **WithEvents** variables must be module-level variables. This is true regardless of the type of module you place them in.

The variable **mblnCancel** will be used to cancel the LongTask method.

► Limitations on WithEvents variables

You should be aware of the following limitations on the use of **WithEvents** variables:

- A **WithEvents** variable cannot be a generic object variable. That is, you cannot declare it As Object - you must specify the class name when you declare the variable.
- You cannot declare a **WithEvents** variable As New. The event source object must be explicitly created and assigned to the **WithEvents** variable.
- You cannot declare **WithEvents** variables in a standard module. You can declare them only in class modules, form modules, and other modules that define classes.
- You cannot create arrays of **WithEvents** variables.

► Writing code to handle an event

As soon as you declare a variable **WithEvents**, the variable name appears in the left-hand drop down of the module's code window. When you select **MyDoc**, the **Document** class events will appear in the right-hand drop down, as shown in Figure 2 below:

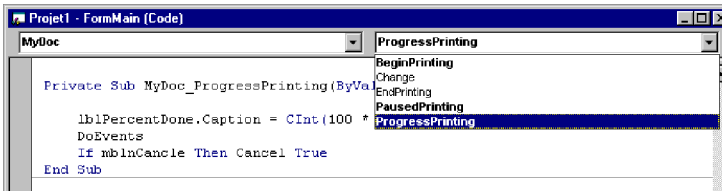


Figure 2 An event associated with a **WithEvents** variable

Selecting an event will display the corresponding event procedure, with the prefix **MyDoc_**. All the event procedures associated with a **WithEvents** variable will have the variable name as a prefix.



For example, add the following code to the **MyDoc_ProgressPrint** event procedure:

```
Private Sub MyDoc_ProgressPrinting (ByVal Percent as integer,Cancel as integer)
    lblPercentDone.caption = CInt (100 * Percent) & "%"
    DoEvents
    If mblnCancel Then Cancel = True
End Sub
```

Whenever the **ProgressPrinting** event is raised, the event procedure displays the percent complete in a Label control. The **DoEvent** statement allows event processing to occur. The module-level variable `mblnCancel` is set to `True`, and the **MyDoc_ProgressPrinting** event then tests it and sets the `ByRef Cancel` argument to `True`.

Connecting a **WithEvents** variable to an object

When you declare a variable **WithEvents** at design time, there is no object associated with it. A **WithEvents** variable is just like any other object variable. You have to create an object and assign a reference to the object to the **WithEvents** variable.



Add the following code to the **Form_Load** event procedure to create the **LabelManager2.Application**.

```
Private Sub Form_Load()
    Set MyApp = New LabelManager2.Application
    Set MyDoc = MyDoc.Documents.Add ("My Document")
    MyApp.EnableEvents = True
End Sub
```

When the code above is executed, Visual Basic creates a **LabelManager2.Application** and a new document called "My Document" then connects its events to the event procedures associated with **MyDoc**. From that point on, whenever the **MyDoc** raises its **ProgressingPrinting** event, the **MyDoc_PrintProgressing** event procedure will be executed.

Compatibility with the previous version

This version is compatible with the previous version of the label design software.

However, the labeling software includes new features and certain processes have changed.

To ensure your program can be executed with this version, verify your code by referring to the User's Guide for information on the functions that have changed.



For example, the previous version of your labeling software uses a simple-document interface (SDI) and, the **ActiveDocument** property always refers to a document. This version is a multiple-document interface (MDI) and there isn't always an open document. If you use this property, verify that there is an open document after the server is activated.

To remain compatible with the previous version :

- a document is automatically created at initialization,
- the **Open** method will close the current document (if one exists), before a new document is created (The **Close** method functions the same way).



However, if the **Application** object is visible, the user has control of the active document management. For example, if the user closes the active document, a new document is not automatically created.

Particularity about access rights

Certain versions of the labeling software include a **User manager** module. This module controls access to certain functions of the labeling software.

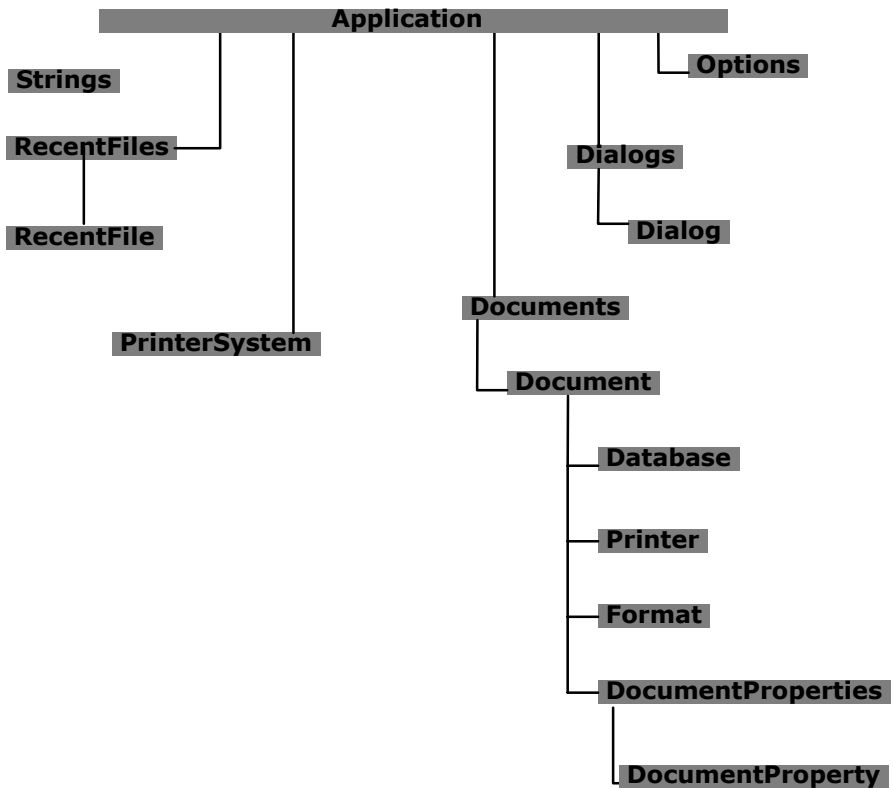
For example, if calling a function through your ActiveX interface fails, verify your rights in the **User manager** module. An error message is displayed and provides information about the nature of the error (see the **Reference Guide**, Chapter 2 - 24: Error code table).

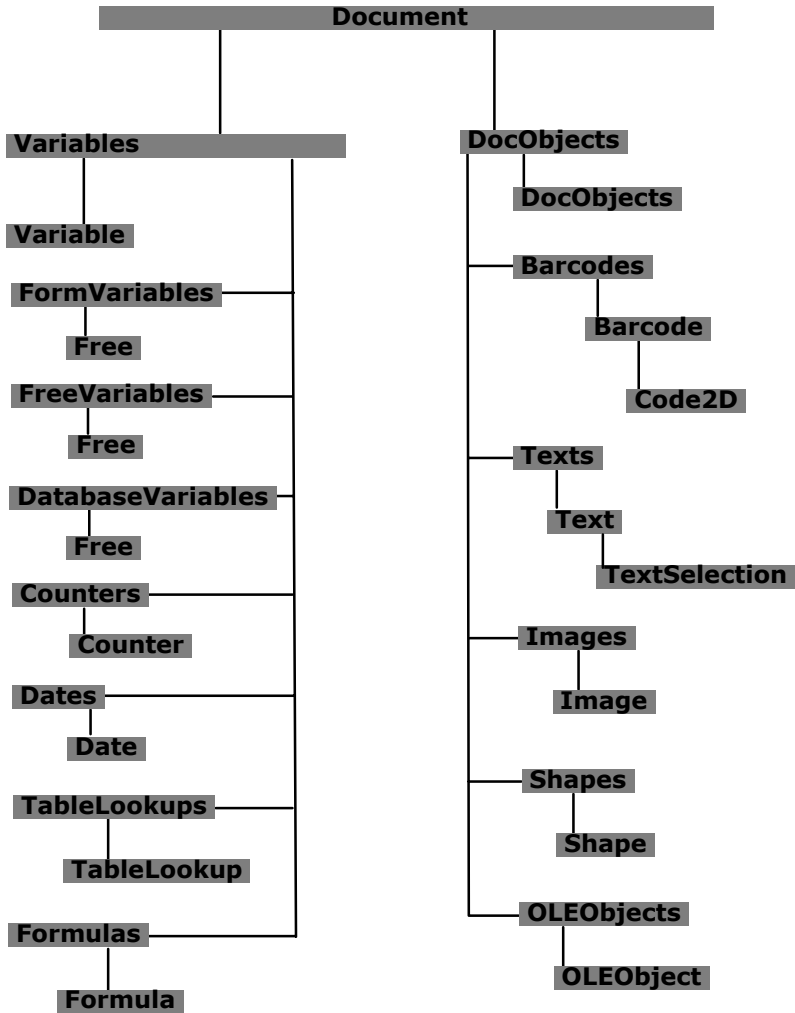
Reference Guide



Hierarchy diagram

The diagram below shows the object hierarchy:





Application Object

Properties

ActiveDocument
ActivePrinterName
Application
Caption
DefaultFilePath
Dialogs
Documents
EnableEvents
FullName
Height
Left
Locked
Name (Default)
Options
Parent
Path
PrinterSystem
RecentFiles
Top
UserControl
Version
Visible
Width

Methods

ErrorMessage
GetLastError
ShowHelp
Move
Resize
Quit

**Object
Properties****► Application.ActiveDocument**

This property allows you to access the document object interface (refer to the document which has the focus in the main application).

Returns an error if no document in application.

Access Read-Only.

Type VT_DISPATCH or Document.

► Application.ActivePrinterName

Returns the current pair <Printer, Port> of the active document, if any, empty string if none.

Access Read-only.

Type VT_BSTR or String.

► Application.Application

This property returns the Application object that represents the root object of the hierarchy.

Access Read-only.

Type VT_DISPATCH or Application.

► Application.Caption

Returns or sets the caption text for the application window. To change the caption of the application window into the default text, set this property to an empty string ("").

Access Read/Write.

Type VT_BSTR or String.

▶ Application.DefaultFilePath

Sets or returns the default path specification used by the application for opening document files.

Access Read/Write.

Type VT_BSTR or String.

▶ Application.Dialogs

Returns the Dialogs collection that represents all the built-in dialog boxes of the application.

Access Read-only.

Type VT_DISPATCH or Dialogs.

▶ Application.Documents

Returns the Documents collection that represents all the open documents.

Access Read-only.

Type VT_DISPATCH or Documents.

▶ Application.EnableEvents

Enables or disables Automation events notification (Default: False) (see Appendix).

Access Read/Write.

Type VT_BOOL or Boolean.

▶ Application.FullName

Returns the file specification for the application, including path. (Ex : c:\drawdir\scribble).

Access Read-Only.

Type VT_BSTR or String.

► Application.Height

Returns or sets the height of the main window of the application (in pixel unit).

Access Read/Write.

Type VT_I4 or Long.

► Application.Left

Returns or sets the distance between the left edge of the main window of the application and the left edge of the screen (in pixel unit).

Access Read/Write.

Type VT_I4 or Long.

► Application.Locked

Locks the User Interface if True.

Access Read/Write.

Type VT_BOOL or Boolean.

► Application.Name

Returns the name of the application (for example, "Microsoft Word"). Default property.

Access Read-Only.

Type VT_BSTR or String.

► Application.Options

Represents application and general document options. Many of the properties for the Options object correspond to items in the Options dialog box (Tools menu). Use the Options property to return the Options object.

The following example sets two application options:

```
With Options
    .LoadPrinterSetup = True
    .MeasureSystem = 1ppxInch
End With
```

Access Read-only.

Type VT_DISPATCH or Options.

▶ **Application.Parent**

Returns the parent object of the specified object.

Access Read-only.

Type VT_DISPATCH.

▶ **Application.Path**

Returns the path of the Application (with « \ » character).

Access Read-only.

Type VT_BSTR or String.

▶ **Application.PrinterSystem**

Returns the PrinterSystem object that represents all printers in the system.

Access Read-only.

Type VT_DISPATCH or PrinterSystem.

▶ **Application.RecentFiles**

Returns the RecentFiles collection that represents the list of last recent files used (File menu in UI).

Access Read-only.

Type VT_DISPATCH or RecentFiles.

▶ **Application.Top**

Returns or sets the distance between the top edge of the main window of the application and the top edge of the screen (in pixel unit).

Access Read/Write.

Type VT_I4 or Long.

► Application.UserControl

True if the application was created by the user.

False if the application was created in programming (with the **CreateObject** or **GetObject** method in Visual Basic).

Note

If the application is visible to the user, this property will always return **True**.

Access Read-Only.

Type VT_BOOL or Boolean.

► Application.Version

Returns the software version number.

Access Read-Only.

Type VT_BSTR or String.

► Application.Visible

True if the application is visible. (Default: False, if application was launched with CreateObject).

Access Read/Write.

Type VT_BOOL or Boolean.

► Application.Width

Returns or sets the width of the main window of the application (in pixel unit).

Access Read/Write.

Type VT_I4 or Long.

Object Methods

► Application.ErrorMessage

VTS_BSTR or String **ErrorMessage(intErrorCode)**

Returns the string message error associated with the error code parameter.

Return value: Message associated.

Parameters:

intErrorCode Required VT_I2 or Integer. Error code to process.

► Application.GetLastError

VTS_I2 or Integer **GetLastError()**

Returns the last error code generated.

Return value: Error code (see Error code table below).

Parameters: None.

No error	0
Can't open data file	1200
Can't open query file	1201
Can't open descriptor file	1202
Can't open label file	1203
Can't open POC file	1204
Can't open log file	1205
Printer not found	1300
Driver not found	1301
Incorrect Datasource enum value	1400
Incorrect Rotation enum value	1401
Incorrect HRAlign enum value	1402
Incorrect HRPosition enum value	1403
Incorrect HR check digit enum value	1404
Incorrect Anchor point enum value	1405
Incorrect counter base enum value	1406
Incorrect Label object enum value	1407
Incorrect view size enum value	1408
Incorrect view mode enum value	1409
Incorrect MeasureSystem enum value	1410
Incorrect dialog type enum value	1411

Incorrect language enum value	1412
Incorrect symbology enum value	1413
Incorrect built in document property enum value	1414
Incorrect view orientation enum value	1415
Incorrect form prompt mode enum value	1416
Object not found	1500
Can't create object	1501
Variable not found	1502
Can't create variable	1503
Invalid font object	1504
Invalid variable object	1505
Name of item already used	1506
Database not connected	1600
Database connection failed	1601
Number must be positive	2000
Data type must be a boolean	2001
Invalid path	2002
File already exists	2003
Can't prompt dialog box (no active document)	2100
Not sufficient access rights to perform this operation	3000

Figure 3 Error code table

► Application.ShowHelp

VTS_NONE **ShowHelp**(*strHelpFile*, *longHelpContext*)

This method activates a help file.

Parameters:

strHelpFile Optional VT_BSTR or String. Specifies the help file to open (.HLP or .CHM). If not specified, associated help file is opened.

longHelpContext Optional VT_I4 or Long. Specifies the id context to jump to. If not specified, general index is prompted.

► Application.Move

VTS_NONE **Move**(*longposLeft*, *longposTop*)

Moves the application window to the specified position (posLeft, posTop), in pixel unit.

Parameters:

longposLeft Required VT_I4 or Long. Sets the distance between the left edge of the main window of the application and the left edge of the screen(in pixel unit).

longPosTop Required VT_I4 or Long. Sets the distance between the top edge of the main window of the application and the top edge of the screen (in pixel unit).

► Application.Resize

VTS_NONE **Resize**(*longWidth*, *longHeight*)

Resizes the application window (Width, Height), in pixel unit.

Parameters:

longWidth Required VT_I4 or Long. Sets the width of the main window of the application (in pixel unit).

longHeight Required VT_I4 or Long. Sets the height of the main window of the application (in pixel unit).

► Application.Quit

VTS_NONE **Quit**()

Quits the current application. No effect if the application has been launched manually. First executes a Document.CloseAll (True) then releases the application.

PrinterSystem Object

Properties

(None)

Methods

Families
Models
Printers
Ports
Add
Remove
Rename

Object Methods

► PrinterSystem.Families

VTS_DISPATCH or Strings **Families** ()

Retrieves printer families list.

► PrinterSystem.Models

VTS_DISPATCH or Strings **Models** (*strFamilyName*)

Retrieves models associated with a family.

Parameters:

strFamilyName Optional VT_BSTR or String. It specifies the Family for which the models list is needed. If none, it returns the full models list.

► PrinterSystem.Printers

VTS_DISPATCH or Strings **Printers** (*intKindOfPrinters*)

Retrieves installed printers as string pairs < Printer, Port > .

Parameters:

intKindOfPrinters Optional VT_I2 or Integer or enumKindOfPrinters. (default lppxInternalPrinters).

The value can be one of the following:

```
lppxInternalPrinters= 1
lppxWindowsPrinters= 2
lppxAllPrinters= 3
```

► PrinterSystem.Ports

VTS_DISPATCH or Strings **Ports** ().

Retrieves all installed ports on the system.

► PrinterSystem.Add

VTS_BSTR or String **Add** (*strPrinterName*, *strPortName*, *boolDirectAccess*)

Installs a new printer and returns the full name assigned to it.

Parameters:

strPrinterName Required VT_BSTR or String. PrinterName to install (got with Printers.InternalPrinters).

strPortName Required VT_BSTR or String. PortName associated with the printer.

boolDirectAccess Optional VT_BOOL or Boolean(default value FALSE). Is the port used with direct access or not.

Note

Only models can be installed with this method.

► PrinterSystem.Remove

VTS_NONE **Remove** (*strPrinterPortName*).

Removes an installed printer.

Parameters:

strPrinterPortName Required VT_BSTR or String. Full name of an installed printer (got with PrinterSystem.Printers(lppxInternalPrinters) method).

Note

Only models can be removed.

If an active document uses this printer, the operation fails.

► PrinterSystem.Rename

VTS_NONE **Rename** (*strPrinterName*, *strNewPrinterName*).

Renames a model.

Parameters:

strPrinterName Required VT_BSTR or String. Name of the installed printer to rename.

strNewPrinterName Required VT_BSTR or String. New name to assign.

Note

Only models can be renamed.

Options Object

Properties

Application
CreateBackup
DefaultDescriberPath
DefaultSharedVarPath
DefaultImagePath
DefaultPrintOutFilePath
DefaultQueryPath
DefaultUserSettingsPath
EuroConversionRate
Language
LoadPrinterSetup
LoadPrinter
MeasureSystem
OpenMergeDatabase
Parent
OpenReadOnly
SharedFileAccessTimeout
TrayNotification

Methods

(None)

Object Properties

► Options.Application

This property returns the [Application](#) object that represents the root object of the hierarchy.

Access Read-Only.

Type VT_DISPATCH or [Application](#) object.

► Options.CreateBackup

Returns or sets the CreateBackup option. (Default: True).

Access Read/Write.

Type VT_BOOL or Boolean.

► Options.DefaultDescriberPath

Returns or sets the DefaultDescriberPath option.

Access Read/Write.

Type VT_BSTR or String.

► Options.DefaultSharedVarPath

Returns or sets the DefaultSharedVarPath option.

Access Read/Write.

Type VT_BSTR or String.

► Options.DefaultImagePath

Returns or sets the DefaultImagePath option.

Access Read/Write.

Type VT_BSTR or String.

► Options.DefaultPrintOutFilePath

Returns or sets the DefaultPrintOutPath option.

Access Read/Write.

Type VT_BSTR or String.

► Options.DefaultQueryPath

Returns or sets the DefaultQueryPath option.

Access Read/Write.

Type VT_BSTR or String.

► Options. DefaultUserSettingsPath

Returns or sets the DefaultUserSettingsPath option.

Access Read/Write.

Type VT_BSTR or String.

► Options. EuroConversionRate

Returns or sets the EuroConversionRate option. (Default 6.55957).

Access Read/Write.

Type VT_R4 or Single.

► Options.Language

Returns or sets the Language option.

Access Read/Write.

Type VT_I4 or Long or enumLanguage type.

The value can be one of the following:

lppxEnglish	= 1
lppxFrench	= 2
lppxGerman	= 3
lppxItalian	= 4
lppxSpanish	= 5
lppxDanish	= 6
lppxSwedish	= 7
lppxJapanese	= 8
lppxHungarian	= 9
lppxDutch	= 10
lppxCzech	= 11
lppxNorwegian	= 12
lppxFinnish	= 13
lppxPortuguese	= 14
lppxSimplifiedChinese	= 15
lppxTraditionalChinese	= 16
lppxKorean	= 17

Note

Depending on the product, not all languages are available.

► Options.LoadPrinter

Returns or sets the LoadPrinter option. (Default : False).

Access Read/Write.

Type VT_BOOL or Boolean.

► Options.LoadPrinterSetup

Returns or sets the LoadPrinterSetup option. (Default : True).

Access Read/Write.

Type VT_BOOL or Boolean.

► Options.MeasureSystem

Returns or sets the MeasureSystem option. (Default : lppxMillimeter).

Access Read/Write.

Type VT_I2 or Integer or enumMeasureSystem type.

The value can be one of the following:

lppxMillimeter = 0
lppxInch = 1

Note

Using lppxMillimeter unit means that values entered are in Millimeter per cent.

Using lppxInch unit means that values entered are in Inch per thousand.

► Options.OpenMergeDatabase

Returns or sets the OpenMergeDatabase option. (Default: False)

Access Read/Write.

Type VT_BOOL or Boolean.

► Options.Parent

Returns the parent object of the specified object.

Access Read-Only.

Type VT_DISPATCH.

► Options.OpenReadOnly

Returns or sets the OpenReadOnly option. (Default : False)

Access Read/Write.

Type VT_BOOL or Boolean.

► Options.SharedFileAccessTimeout

Returns or sets the SharedFileAccessTimeout option. (Default: 10000 ms)

Access Read/Write.

Type VT_I4 or Long.

► Options.TrayNotification

Enables or disables notification of printing in System Tray Bar. (Default: True)

Access Read/Write.

Type VT_BOOL or Boolean.

Dialogs Collection

Properties

Application
Count
Parent

Methods

Item (Default)

Object Properties

► Dialogs.Application

This property returns the [Application](#) object that represents the root object of the hierarchy.

Access Read-Only

Type VT_DISPATCH or [Application](#) object.

► Dialogs.Count

Returns the number of items in the specified collection.

Access Read-Only.

Type VT_I2 or [Integer](#).

► Dialogs.Parent

Returns the parent object of the specified object.

Access Read-Only.

Type VT_DISPATCH.

**Object
Methods****► Dialogs.Item**

VTS_DISPATCH or Dialog **Item**(*intIndex*).

Returns a member of a collection, by position.

Note

If the value provided as Index does not match any existing member of the collection, no object is returned.

Parameters:

intIndex Required VT_I2 or Integer or enumDialogType. The index number of a member of the collection.

The index must be a numeric expression (a number from 1 to the value of the collection's Count property), or a constant.

Note

If the value provided as Index doesn't match any existing member of the collection, an error occurs.

The Item method is the default method for collections. Therefore, the following two lines of code are equivalent.

```
Object.Dialogs(1)  
Object.Dialogs.Item(1)
```

The value can be one of the following:

lppxPrinterSelectDialog=	1
lppxOptionsDialog=	2
lppxFormDialog=	3
lppxPrinterSetupDialog=	4
lppxPageSetupDialog=	5
lppxDocumentPropertiesDialog=	6

Dialog Object

Properties

Application
Parent
Type

Methods

Show

Object Properties

► Dialog.Application

Returns the **Application** object that represents the root object of the hierarchy.

Access Read-Only.

Type VT_DISPATCH or Application object.

► Dialog.Parent

Returns the parent object of the specified object.

Access Read-Only.

Type VT_DISPATCH.

► Dialog.Type

Returns the type of the prompted dialog box.

Access Read-Only.

Type VT_I2 or Integer or enumDialogType type.

**Object
Methods****► Dialog.Show**

VTS_I2 or Integer **Show()**.

Prompts the dialog box associated.

Return value 1 if the user has clicked on OK.
 2 if the user has clicked on Cancel.

If application is not visible, dialog box is prompted at the top level of all windows.

If there is no document open, the dialog boxes (except Options dialog box) can't be displayed because they depend on the document.

RecentFiles Collection

Properties

Application
Count
Maximum
Parent

Methods

Add
Item (Default)
Clear
Remove

Object Properties

► RecentFiles.Application

This property returns the Application object that represents the root object of the hierarchy.

Access Read-Only.

Type VT_DISPATCH or Application object.

► RecentFiles.Count

Returns the number of items in the specified collection.

Access Read-Only.

Type VT_I2 or Integer.

► RecentFiles.Maximum

Returns or sets the maximum number of items in the specified collection (from 0 to 16).

Access Read/Write.

Type VT_I2 or Integer.

► RecentFiles.Parent

Returns the parent object of the specified object.

Access Read-Only.

Type VT_DISPATCH.

Object Methods

► RecentFiles.Add

VTS_DISPATCH or RecentFile **Add**(*DocumentReference*, *boolReadOnly*).

Adds a document reference to the collection and in the File menu.

Parameters:

DocumentReference Required VT_VARIANT or Variant. This reference must be unique in the collection.

boolReadOnly Optional VT_BOOL or Boolean. If document must be opened, it will be opened with ReadOnly attribute.

► RecentFiles.Clear

VTS_NONE **Clear**.

Resets the collection and clears menu.

► RecentFiles.Item

VTS_DISPATCH or RecentFile **Item**(*intIndex*).

Returns a member of a collection, by position (default method).

Note

If the value provided as Index does not match any existing member of the collection, no object is returned.

Parameters:

Index Required VT_I2 or Integer. The index number of a member of the collection.

The index must be a numeric expression (a number from 1 to the value of the collection's Count property), or a constant.

Note

If the value provided as Index doesn't match any existing member of the collection, an error occurs. The **Item** method is the default method for collections. Therefore, the following two lines of code are equivalent.

```
Object.RecentFiles(1)
Object.RecentFiles.Item(1)
```

► RecentFiles.Remove

VTS_NONE **Remove**(*intIndex*).

Deletes object with *intIndex* index.

RecentFile Object

Properties

Application
Parent
Path
Name

Methods

Open

Object Properties

► RecentFile.Application

Returns the Application object that represents the root object of the hierarchy.

Access Read-Only.

Type VT_DISPATCH or Application object.

► RecentFile.Parent

Returns the parent object of the specified object.

Access Read-Only.

Type VT_DISPATCH.

► RecentFile.Path

Returns the path associated with the current filename (always without « \ » character).

Access Read-Only.

Type VT_BSTR or String.

► RecentFile.Name

Returns the name associated with the current filename.

Access Read-Only.

Type VT_BSTR or String.

**Object
Methods****▶ RecentFile.Open**

VTS_DISPACTCH or Document **Open()**.

Opens the document associated with the current filename.

Documents Collection

Properties

Application
Count
DefaultExt
Parent

Methods

Add
CloseAll
Item (Default)
Open
SaveAll

Object Properties

► Documents.Count

Returns the number of items in the specified collection.

Access Read-Only.

Type VT_I2 or Integer.

► Documents.Application

This property returns the Application object that represents the root object of the hierarchy.

Access Read-Only.

Type VT_DISPATCH or Application object.

► Documents.DefaultExt

This property returns the default document filename extension for the application.

Access Read-Only.

Type VT_BSTR or String.

► Documents.Parent

Returns the parent object of the specified object.

Access Read-Only.

Type VT_DISPATCH.

Object Methods

► Documents.Add

VTS_DISPATCH or Document **Add**(*strDocumentName*).

Adds a new Document to the collection.

Return value: Returns a Document object if succeeded.

Parameters:

strDocumentName Optional VT_BSTR or String.
Specifies the name of the new document to add.
If none, system automatically assigns one.

► Documents.CloseAll

VTS_NONE **CloseAll** (*boolSaveChanges*).

Closes all documents.

Parameters:

boolSaveChanges Optional VT_BOOL or Boolean. Specifies the save action for all documents. (default: True).

If *boolSaveChanges* is True and a document has not been previously saved, the Saves As dialog box is automatically prompted.

► Documents.Item

VTS_DISPATCH or Document **Item**(*varIndex*).

Returns a Document of a collection, either by position or by name.

Note

If the value provided as Index does not match any existing member of the collection, no object is returned.

Parameters:

varIndex Required VT_VARIANT or Variant. The name or index number of a member of the collection.

The index can be a numeric expression (a number from 1 to the value of the collection's Count property), a constant, or a string.

Note

If the value provided as Index doesn't match any existing member of the collection, an error occurs.
The Item method is the default method for collections. Therefore, the following two lines of code are equivalent.

```
Object.Documents(1)
Object.Documents.Item(1)
```

► Documents.Open

VTS_DISPATCH or Document **Open**(*strFileName*,
boolReadOnly).

Opens the specified document and adds it to the Documents collection.

Return value: Returns a Document object.

Parameters:

strFileName Required VT_BSTR or String. The name of the document (paths are accepted).

boolReadOnly Optional VT_BOOL or Boolean. True to open the document as read-only. By default, set to False.

► Documents.SaveAll

VTS_NONE **SaveAll**(*boolAlwaysPrompt*).

Saves all the documents in the Documents collection.

If a document hasn't been previously saved, the Save As dialog box is prompted even if the Prompt parameter is assigned to False.

Parameters:

boolAlwaysPrompt Optional VT_BOOL or Boolean. True if user wants to prompt the save dialog box (default: False).

Document Object

Properties

Application
 BuiltInDocumentProperties
 Database
 Format
 TriggerForm
 FullName
 Name (Default)
 DocObjects
 Parent
 Printer
 ReadOnly
 Variables
 ViewMode
 ViewOrientation
 WindowState
 IsModified

Methods

Close
 CopyToClipboard
 Merge
 FormFeed
 GeneratePOF
 Insert
 PrintDocument
 PrintLabel
 Save
 SaveAs
 Activate
 CopyImageToFile

Object Properties

► Document.Application

Returns the [Application](#) object that represents the root object of the hierarchy.

Access Read-Only.

Type VT_DISPATCH or [Application](#) Object.

► Document.BuiltInDocumentProperties

Returns the [DocumentProperties](#) collection that represents document properties.

Access Read-Only.

Type VT_DISPATCH or [DocumentProperties](#) collection.

▶ Document.Database

Returns the Database object associated with the document.

Access Read-Only.

Type VT_DISPATCH or Database object.

▶ Document.Format

Returns the Format object that represents the format of the document.

Access Read-Only.

Type VT_DISPATCH or Format object.

▶ Document.TriggerForm

Sets or returns the TriggerForm in printing situation.

Access Read/Write.

Type VT_I2 or Integer or enumTriggerForm type.

The value can be one of the following:

lppxNever	= 1
lppxForEachSerie	= 2
lppxForEachLabel	= 3

▶ Document.FullName

Returns the file specification for the document, including path.

Access Read-Only.

Type VT_BSTR or String.

Ex : c:\drawdir\scribble.

▶ Document.Name

Returns the document's name. Default property.

Access Read-Only.

Type VT_BSTR or String.

► Document.IsModified

Tests that the document has been modified since the last save operation.

Access Read-Only.

Type VT_BOOL or Boolean.

The possible modifications are : creating, deleting and editing DocObjects; creating and deleting variables...

► Document.DocObjects

Returns the DocObjects collection that represents all the created objects in the document.

Access Read-Only.

Type VT_DISPATCH or DocObjects object.

► Document.Parent

Returns the parent object of the specified object.

Access Read-Only.

Type VT_DISPATCH.

► Document.Printer

Returns the Printer object that represents the associated printer.

Access Read-Only.

Type VT_DISPATCH or Printer object.

► Document.ReadOnly

True, if the changes of the current document cannot be saved to the original document.

Access Read-Only.

Type VT_BOOL or Boolean.

► Document.Variables

Returns the Variables collection that represents all the created Variable objects in the document.

Access Read-Only.

Type VT_DISPATCH or Variables collection.

► Document.ViewMode

Sets or retrieves the current mode of visual display.

Access Read/Write.

Type VT_I2 or Integer or enumViewMode type.

The value can one of the following:

lppxViewModeName	= 1
lppxViewModeSize	= 2
lppxViewModeValue	= 3
lppxViewModeForm	= 4

► Document.ViewOrientation

Sets or retrieves the orientation of the view of the document.

Access Read/Write.

Type VT_I2 or Integer or enumRotation type.

The value can be one of the following:

lppxNoRotation	= 0
lppx90DegreeRight	= 1
lppxUpSideDown	= 2
lppx90DegreeLeft	= 3

► Document.WindowState

Sets or retrieves the current size of the visual display of the document.

Access Read/Write.

Type VT_I2 or Integer or enumWindowState type.

The value can one of the following:

lppxNormal=	1
lppxMinimized=	2
lppxMaximized=	3

**Object
Methods****► Document.Close**

VTS_I2 **Close**(*boolSave*).

Closes document.

Parameters:

boolSave Optional VT_BOOL or Boolean.(default false) If True, saves document.

► Document.CopyToClipboard

VTS_BOOL **CopyToClipboard**().

Copies an image of the document to the Clipboard.

► Document.FormFeed

VTS_I2 **FormFeed** ().

Ends the process job.

► Document.GeneratePOF

VTS_I2 **GeneratePOF** (*strDestinationFileName*, *strModelFileName*).

Generates a POF file.

Parameters:

strDestinationFileName Required VT_BSTR or String. Name of the file to print to.

strModelFileName Optional VT_BSTR or String. Name or FullName of Configuration file (.POC) to use. If none, default POC file is used.

► Document.Insert

VTS_I2 **Insert** (*strDocumentFileName*).

Inserts a document in the current document.

Parameters:

strDocumentFileName Required VT_BSTR or String. Name of the document to insert.

► Document.Merge

VTS_I2 **Merge** (*longLabelQuantity*, *longLabelCopy*, *longInterCut*, *longPageCopy*, *longLabelNoPrintedFrom*, *strFileName*).

Merges document with the associated Database.

Parameters:

<i>longLabelQuantity</i>	Required VT_I4 or <u>Long</u> .
<i>longLabelCopy</i>	Optional VT_I4 or <u>Long</u> (default 1).
<i>longInterCut</i>	Optional VT_I4 or <u>Long</u> (default 1).
<i>longPageCopy</i>	Optional VT_I4 or <u>Long</u> (default 1).
<i>longLabelNoPrintedFrom</i>	Optional VT_I4 or <u>Long</u> (default 1).
<i>strFileName</i>	Optional VT_BSTR or <u>String</u> (default empty string).

Note

Parameters are described in the User's Guide.

► Document.PrintDocument

VTS_I2 **PrintDocument** (*longLabelQuantity*).

Prints document and executes an automatic FormFeed.

Parameters:

longLabelQuantity Optional VT_I4 or Long. Quantity of labels to print (Default : 1).

► Document.PrintLabel

VTS_I2 **PrintLabel** (*longLabelQuantity*, *longLabelCopy*, *longInterCut*, *longPageCopy*, *longLabelNoPrintedFrom*, *strFileName*).

Prints document.

Parameters:

<i>longLabelQuantity</i>	Required VT_I4 or <u>Long</u> .
<i>longLabelCopy</i>	Optional VT_I4 or <u>Long</u> (default 1).
<i>longInterCut</i>	Optional VT_I4 or <u>Long</u> (default 1).
<i>longPageCopy</i>	Optional VT_I4 or <u>Long</u> (default 1).
<i>longLabelNoPrintedFrom</i>	Optional VT_I4 or <u>Long</u> (default 1).
<i>strFileName</i> (empty string).	Optional VT_BSTR or <u>String</u> (default empty string).

Note

Parameters are described in the User's Guide.

► Document.Save

VTS_I2 **Save** ().

Saves the document.

If the document has not been saved, a dialog box is prompted automatically.

► Document.SaveAs

VTS_I2 **SaveAs** (*strDocumentFileName*).

Saves the document with a new name.

Parameters:

strDocumentFileName Required VT_BSTR or String.

► Document.Activate

VTS_NONE **Activate** ().

Causes the document object to be activated, being the ActiveDocument.

► Document.CopyImageToFile

VTS_BSTR or String **CopyImageToFile**(*Colors, Extension, Rotation, Percent, strFilename*).

Generates file that contains the Image of the document.

Return value: Returns a string that represents the full name of the generated Bitmap file.

Parameters:

Colors Optional VT_I2 or Integer. (Default 8). Specifies the number of bits-per-pixel. Depending of the generated file, the values must be : 1, 4, 8, 16, 24, 32.

Extension Optional VT_BSTR or String (Default "BMP"). Specifies the extension of the file to generate. For a complete list of extensions, refer to the labeling software documentation.

Rotation Optional VT_I2 or Integer (Default 0). Rotation in geometrical degree. The values must be between 0 and 360.

Percent Optional VT_I2 or Integer (Default 100). Scaling factor. The values must be between 1 and 400.

strFileName Optional VT_BSTR or String (Default ""). If specified: name of the generated Bitmap file.

Database Object

Properties

Application
 AutoVariables
 BOF
 EOF
 IsOpen
 Parent
 Name (Default)
 DocObjects
 Parent

Methods

Close
 MoveFirst
 MoveLast
 MoveNext
 MovePrevious
 OpenASCII
 OpenODBC
 OpenQuery
 Save

Object Properties

► Database.Application

This property returns the [Application](#) object that represents the root object of the hierarchy.

Access Read-Only.

Type VT_DISPATCH or [Application](#) object.

► Database.AutoVariables

Automatic creation of database variables when database connects. (Default: True).

Access Read/Write.

Type VT_BOOL or [Boolean](#).

► Database.BOF

Returns a value that indicates whether the current row position is before the first row in the current recordset.

Return value:

True The current row position is before the first row.
False The current row position is on or after the first row.

Access Read-Only.

Type VT_BOOL or Boolean.

► Database.EOF

Returns a value that indicates whether the current row position is after the last row in the current recordset.

Return value:

True The current row position is after the last row.
False The current row position is on or before the last row.

Access Read-Only.

Type VT_BOOL or Boolean.

► Database.IsOpen

Tests if the Database object has been open successfully.

Access Read-Only.

Type VT_BOOL or Boolean.

► Database.Parent

Returns the parent object of the specified object.

Access Read-Only.

Type VT_DISPATCH.

**Object
Methods****► Database.Close**

VT_NONE **Close()**.

Closes an open database.

► Database.MoveFirst

VT_BOOL or Boolean **MoveFirst()**.

Repositions the current row pointer in the first row of the current recordset and makes that row the current row.

► Database.MoveLast

VT_BOOL or Boolean **MoveLast()**.

Repositions the current row pointer in the last row of the current recordset object and makes that row the current row.

► Database.MoveNext

VT_BOOL or Boolean **MoveNext()**.

Repositions the current row pointer in the next row of the current recordset object and makes that row the current row.

► Database.MovePrevious

VT_BOOL or Boolean **MovePrevious()**.

Repositions the current row pointer in the previous row of the current recordset object and makes that row the current row.

► Database.OpenASCII

VT_BOOL or Boolean **OpenASCII**(*strTextFileName*,
strDescriberFileName).

Opens ASCII database.

Return value: Returns a boolean that indicates whether the opening fails or not.

Parameters:

strTextFileName Required VT_BSTR or String. The database text file.

strDescriberFileName Required VT_BSTR or string.
Describer associated with the text file.

► Database.OpenQuery

VT_BOOL or Boolean **OpenQuery**(*strQueryFileName*).

Opens a CSQ query file.

Return value: returns a boolean that indicates whether the opening fails or not.

Parameters:

strQueryFileName Required VT_BSTR or String. The file which contains the query.

► Database.OpenODBC

VT_BOOL or Boolean **OpenODBC**(*strDatasourceConnexion*,
strQueryString).

Opens an ODBC database.

Return value: Returns a boolean that indicates whether the opening fails or not.

Parameters:

strDatasourceConnexion Required VT_BSTR or String. The database string connection. For the *strDatasourceConnexion* parameter, refer to Microsoft ODBC documentation.

strQueryString Required VT_BSTR or String. SQL query.

Printer Object

Properties

Application
DeviceCodeNames
DeviceFontNames
FullName (Default)
Name
Parent
WindowsFontNames
WindowsCodeNames
XDPI
YDPI

Methods

ShowSetup
Send
SetParameter
SwichTo

Object Methods

► **Printer.ShowSetup**

VTS_NONE **ShowSetup**().

Prompts the Printer Setup dialog box, in order to change the current printer settings.

► **Printer.Send**

VTS_BOOL **Send** (*strEscapeSequence*).

Sends an escape sequence to the physical device.

Parameters:

EscapeSequence Required VT_BSTR or String. Escape sequence to send.

► **Printer.SetParameter** (not yet implemented)

VTS_BOOL or Boolean **SetParameter**(*strParameter*, *varValue*).

Changes the current printer settings.

Parameters:

strParameter Required VT_BSTR or String. Parameter name to use.

varValue Required VT_VARIANT or Variant. Value to set.

► **Printer.SwitchTo ()**

VTS_BSTR or String **SwitchTo**(*strPrinterName*, *strPortName*, *boolDirectAccess*).

Changes the current printer.

Return value: Returns the name of the installed printer.

Automatically installs a printer if no printer is already installed. You don't need to add a printer through the user interface.

Parameters:

strPrinterName Required VT_BSTR or String. Printer 's name to switch to.

strPortName Optional VT_BSTR or String. Port's name to switch to.

boolDirectAccess Optional VT_BOOL or Boolean. Is the connection of the port direct or not.



Form 1

SwitchTo(« THTPrinter L-1234 », « LPT1: », FALSE).
Result of this instruction is « THTPrinter L-1234,LPT1: ».

Form 2

SwitchTo(« THTPrinter L-1234, ->COM3: »).
Result of this instruction is « Copy of THTPrinter L-1234, ->COM3: » because printer's name is unique.

You can either use form 1 or form 2.



Don't insert spaces between the components in Form 2. Notice that the names are case sensitive !

Object Properties

► **Printer.Application**

This property returns the Application object that represents the root object of the hierarchy.

Access Read-Only.

Type VT_DISPATCH or Application object.

► **Printer.DeviceFontNames**

Returns the Strings collection that represents all the printer fonts names.

Access Read-Only.

Type VT_DISPATCH or Strings collection.

► **Printer.DeviceCodeNames**

Returns the Strings collection that represents all the printer code names.

Access Read-Only.

Type VT_DISPATCH or Strings collection.

► **Printer.FullName**

Returns the full name of the pair <Printer, Port>.

Access Read-Only.

Type VT_BSTR or String.

▶ Printer.Name

Returns the simple name of the current printer.

Access Read/Write.

Type VT_BSTR or String.

▶ Printer.Parent

Returns the parent object of the specified object.

Access Read-Only.

Type VT_DISPATCH.

▶ Printer.XDPI

Returns the horizontal resolution of the printer (in DPI).

Access Read-Only.

Type VT_I4 or Long.

▶ Printer.YDPI

Returns the vertical resolution of the printer (in DPI).

Access Read-Only.

Type VT_I4 or Long.

▶ Printer.WindowsFontNames

Returns the Strings collection that represents all the windows font names.

Access Read-Only.

Type VT_DISPATCH or Strings collection.

▶ Printer.WindowsCodeNames

Returns the Strings collection that represents all the windows code names.

Access Read-Only.

Type VT_DISPATCH or Strings collection.

Format Object

Properties

Application
AutoSize
ColumnCount
HorizontalGap
LabelHeight
LabelWidth
MarginLeft
MarginTop
StockName
StockType
PageHeight
PageWidth
Parent
Portrait
Corner
RowCount
VerticalGap

Methods

SaveStock

Object Properties

► **Format.Application**

Returns the [Application](#) object that represents the root object of the hierarchy.

Access Read-Only.

Type VT_DISPATCH or [Application](#) object.

► Format. AutoSize

Automatically adjusts the page based on the dimension of the label, the number of labels, margins, and the amount of space between labels.

This option is available only for customized page formats. However, it is always possible to disable the automatic option to enter the Height and Width values manually.

Access Read/Write.

Type VT_BOOL or Boolean.

► Format.ColumnCount

Retrieves or sets the number of labels per row (horizontal count).

Access Read/Write.

Type VT_I4 or Long.

► Format.HorizontalGap

Retrieves or sets the amount of empty space between the columns (Horizontal) of labels on a page. (in MeasureSystem unit).

Access Read/Write.

Type VT_I4 or Long.

► Format.VerticalGap

Retrieves or sets the amount of empty space between the rows (Vertical) of labels on a page (in MeasureSystem unit).

Access Read/Write.

Type VT_I4 or Long.

► Format.LabelHeight

Retrieves or sets the height of the label (in MeasureSystem unit).

Access Read/Write.

Type VT_I4 or Long.

► Format.LabelWidth

Retrieves or sets the width of the label (in MeasureSystem unit).

Access Read/Write.

Type VT_I4 or Long.

► Format.MarginLeft

Retrieves or sets the left margin of the page (in MeasureSystem unit).

Access Read/Write.

Type VT_I4 or Long.

► Format.MarginTop

Retrieves or sets the top margin of the page (in MeasureSystem unit).

Access Read/Write.

Type VT_I4 or Long.

► Format.StockName

Retrieves or sets the name of the format model, if any.

Access Read/Write.

Type VT_BSTR or String.

► Format.StockType

Retrieves or sets the type of the format type, if any.

Access Read/Write.

Type VT_BSTR or String.

► Format.PageHeight

Retrieves or sets the height of the page (in MeasureSystem unit).

Access Read/Write.

Type VT_I4 or Long.

► Format.PageWidth

Retrieves or sets the width of the page (in MeasureSystem unit).

Access Read/Write.

Type VT_I4 or Long.

► Format.Parent

Returns the parent object of the specified object.

Access Read-Only.

Type VT_DISPATCH.

► Format.Portrait

Retrieves or sets the orientation of the document.

Access Read/Write.

Type VT_BOOL or Boolean.

► Format.Corner

Retrieves or sets the radius corner of the document (in MeasureSystem unit).

Access Read/Write.

Type VT_I4 or Long.

► Format.RowCount

Retrieves or sets the number of labels per column (vertical count).

Access Read/Write.

Type VT_I4 or Long.

**Object
Methods****► Format.SaveStock**

VTS_NONE Format.**SaveStock**().

Saves the current stock Name/Type. (In order to reuse it with others documents)

DocumentProperties Collection

Properties

Application
Count
Parent

Methods

Item (Default)

Object Properties

► DocumentProperties.Application

This property returns the [Application](#) object that represents the root object of the hierarchy.

Access Read-Only.

Type VT_DISPATCH or [Application](#) object.

► DocumentProperties.Count

Returns the number of items in the specified collection.

Access Read-Only.

Type VT_I2 or [Integer](#).

► DocumentProperties.Parent

Returns the parent object of the specified object.

Access Read-Only.

Type VT_DISPATCH.

Object Methods

► DocumentProperties.Item

VTS_DISPATCH or DocumentProperty **Item**(*longIndex*).

Returns a member of a collection, either by position or by name.

Note

If the value provided as Index does not match any existing member of the collection, no object is returned.

Parameters:

longIndex Required VT_VARIANT or Variant. The name or index number of a member of the collection.

The index can be a numeric expression (a number from 1 to the value of the collection's Count property), a constant, or a string.

Note

If the value provided as Index doesn't match any existing member of the collection, an error occurs.

The Item method is the default method for collections. Therefore, the following two lines of code are equivalent.

```
Object.DocumentProperties(1)
Object.DocumentProperties.Item(1)
```

An enumerated type enumBuiltInDocumentProperty is enabled:

lppxPropertyManager	= 1
lppxPropertyCompany	= 2
lppxPropertyCategory	= 3
lppxPropertyTitle	= 4
lppxPropertySubject	= 5
lppxPropertyAuthor	= 6
lppxPropertyKeywords	= 7
lppxPropertyComments	= 8

DocumentProperty Object

Properties	Methods
Application Name Parent Type Value (Default)	(None)

Object Properties

► DocumentProperty.Application

This property returns the [Application](#) object that represents the root object of the hierarchy.

Access Read-Only.

Type VT_DISPATCH or [Application](#) object.

► DocumentProperty.Name

Returns the name of the variable.

Access Read-Only.

Type VT_BSTR or [String](#).

► DocumentProperty.Parent

Returns the parent object of the specified object.

Access Read-Only.

Type VT_DISPATCH.

► DocumentProperty.Type

Returns the type of the property.

Access Read-Only.

Type VT_I2 or Integer or enumProperty type.

The value can be one of the following:

lppxPropertyTypeNumber = 1

lppxPropertyTypeBoolean = 2

lppxPropertyTypeDate = 3

lppxPropertyTypeString = 4

lppxPropertyTypeFloat = 5

► DocumentProperty.Value

Returns the current value of the DocumentProperty.

Access Read-Only.

Type VT_VARIANT or Variant.

DocObjects Collection

Properties

Application
 Count
 Parent
 Barcodes
 Shapes
 OLEObjects
 Images
 Texts

Methods

Add
 Item (Default)
 Remove

Object Properties

► DocObjects.Application

This property returns the [Application](#) object that represents the root object of the hierarchy.

Access Read-Only.

Type VT_DISPATCH or [Application](#) object.

► DocObjects.Count

Returns the number of items in the specified collection.

Access Read-Only.

Type VT_I2 or [Integer](#).

► DocObjects.Parent

Returns the parent object of the specified object.

Access Read-Only.

Type VT_DISPATCH.

► DocObjects.BarCodes

Returns the BarCodes collection that represents all the created Barcode objects in the document.

Access Read-Only.

Type VT_DISPATCH or BarCodes collection.

► DocObjects.Shapes

Returns the Shapes collection that represents all the created Shape objects in the document.

Access Read-Only.

Type VT_DISPATCH or Shapes collection.

► DocObjects.OLEObjects

Returns the OLEObjects collection that represents all the created OLEObject objects in the document.

Access Read-Only.

Type VT_DISPATCH or OLEObjects collection.

► DocObjects.Images

Returns the Images collection that represents all the created Image objects in the document.

Access Read-Only.

Type VT_DISPATCH or Images collection.

► DocObjects.Texts

Returns the Texts collection that represents all the created Text objects in the document.

Access Read-Only.

Type VT_DISPATCH or Texts collection.

Object Methods

► DocObjects.Add

VT_DISPATCH or DocObject **Add**(*longDocObjectType*, *strDocObjectName*).

Adds a new DocObject object to the current document.

Return value: Returns a DocObject object.

Parameters:

longDocObjectType Required VT_I4 or Long or enumDocObject. The type of object to add.

The value can be one of the following:

<u>lppxObjectText</u>	= 1
<u>lppxObjectBarCode</u>	= 2
<u>lppxObjectImage</u>	= 3
<u>lppxObjectLine</u>	= 4
<u>lppxObjectRectangle</u>	= 5
<u>lppxObjectEllipse</u>	= 6
<u>lppxObjectPolygon</u>	= 7
<u>lppxObjectOblique</u>	= 8
<u>lppxObjectRoundRect</u>	= 9
<u>lppxObjectOLEObject</u>	= 10

strDocObjectName Optional VT_BSTR or String. The name of the object to add.

► DocObjects.Item

VTS_DISPATCH or DocObject **Item**(*varIndex*).

Returns a member of a collection, either by position or by name.

Note

If the value provided as Index does not match any existing member of the collection, no object is returned.

Parameters:

varIndex Required VT_VARIANT or Variant. The name or index number of a member of the collection.

The index can be a numeric expression (a number from 1 to the value of the collection's Count property), a constant, or a string.

Note

If the value provided as Index doesn't match any existing member of the collection, an error occurs. The Item method is the default method for collections. Therefore, the following two lines of code are equivalent.

```
Object.DocObjects (1)
Object.DocObjects.Item(1)
```

► DocObjects.Remove

VTS_NONE **Remove**(*varIndex*).

Removes a member from the DocObjects object.

Parameters:

varIndex Required VT_VARIANT or Variant. An expression that specifies the position of a collection member. If numeric expression: index must be a number from 1 to the value of the collection's Count property. If a string expression: index must correspond to the key argument specified when this member referred to was added to the collection.

DocObject Object

Properties

AnchorPoint
 Application
 BackColor
 ForeColor
 Height
 Left
 Locked
 Name
 Parent
 Printable
 Rotation
 Top
 Type
 Width

Methods

Bound
 Move

Object Properties

► DocObject.AnchorPoint

Returns or sets the anchor point of the current object.

Access Read/Write.

Type VT_I2 or Integer or enumAnchorPoint type.

lppxTopLeft = 1
 lppxTopCenter = 2
 lppxTopRight = 3
 lppxCenterLeft = 4
 lppxCenter = 5
 lppxCenterRight = 6
 lppxBottomLeft = 7
 lppxBottomCenter = 8
 lppxBottomRight = 9

► DocObject.Application

This property returns the Application object that represents the root object of the hierarchy.

Access Read-Only.

Type VT_DISPATCH or Application object.

► DocObject.Height

Returns or sets the height of the object (in MeasureSystem unit).

Access Read/Write.

Type VT_I4 or Long.

► DocObject.Left

Returns or sets the distance between the left edge of the anchor point of the object and the left edge of the document (in MeasureSystem unit).

Access Read/Write.

Type VT_I4 or Long.

► DocObject.Name

Returns or sets the name of the DocObject.

Access Read/Write.

Type VT_BSTR or String.

► DocObject.Printable

Sets or not whether the object is printable.

Access Read/Write.

Type VT_BOOL or Boolean.

► DocObject.Rotation

Sets or retrieves the rotation of the object.

Access Read/Write.

Type VT_I2 or Integer.

The value can be one of the following: 0, 900, 1800, 2700.

► DocObject.Top

Returns or sets the distance between the top edge of the anchor point of the object and the top edge of the document. (in MeasureSystem unit).

Access Read/Write.

Type VT_I4 or Long.

► DocObject.Type

Returns the type of the object (in enumDocObject type).

Access Read-Only.

Type VT_I2 or Integer.

► DocObject.Width

Returns or sets the width of the object (in enumMeasureSystem unit).

Access Read/Write.

Type VT_I4 or Long.

Object Methods

► **DocObject.Bound**

VTS_NONE **Bound** (*longLeftPosition*, *longTopPosition*, *longRightPosition*, *longBottomPosition*).

Sets the bounding rectangle of an object.

Parameters:

longLeftPosition Required VT_I4 or Long. Distance between the left edge of an object and the left edge of the document (in MeasureSystem unit).

longTopPosition Required VT_I4 or Long. Distance between the top edge of an object and the top edge of the document (in MeasureSystem unit).

longRightPosition Required VT_I4 or Long. Distance between the right edge of an object and the left edge of the document (in MeasureSystem unit).

longBottomPosition Required VT_I4 or Long. Distance between the bottom edge of an object and the top edge of the document (in MeasureSystem unit).

► **DocObject.Move**

VTS_NONE **Move** (*longLeftPosition*, *longTopPosition*).

Moves an object in its window.

Parameters:

longLeftPosition Required VT_I4 or Long. Distance between the left edge of an object and the left edge of the document (in MeasureSystem unit).

longTopPosition Required VT_I4 or Long. Distance between the top edge of an object and the top edge of the document (in MeasureSystem unit).

Images Collection

Properties

Application
Count
Parent

Methods

Add
Item (Default)

Object Properties

► Images.Count

Returns the number of items in the specified collection.

Access Read-Only.

Type VT_I2 or Integer.

► Images.Application

This property returns the Application object that represents the root object of the hierarchy.

Access Read-Only.

Type VT_DISPATCH or Application object.

► Images.Parent

Returns the parent object of the specified object.

Access Read-Only.

Type VT_DISPATCH.

Object Methods

► Images.Add

VT_DISPATCH or Image **Add**(*strImageName*).

Adds a new Image object to the collection.

Return value: Returns a Image object.

Parameters:

strImageName Optional VT_BSTR or String. The name of the object to add.

► Images.Item

VTS_DISPATCH or Image **Item**(*varIndex*).

Returns a member of a collection, either by position or by name.

Note

If the value provided as Index does not match any existing member of the collection, none object returned.

Parameters:

varIndex Required VT_VARIANT or Variant. The name or index number of a member of the collection.

The index can be a numeric expression (a number from 1 to the value of the collection's Count property), a constant, or a string.

Note

If the value provided as Index doesn't match any existing member of the collection, an error occurs. The Item method is the default method for collections. The following two lines of code are equivalent.

```
Object. Images (1)  
Object. Images.Item(1)
```

► Images.Remove

VTS_NONE **Remove** (*varIndex*).

Removes a member from the collection.

Parameters:

varIndex Required VT_VARIANT or Variant. An expression that specifies the position of a member of the collection. If a numeric expression, index must be a number from 1 to the value of the collection's Count property. If a string expression, index must correspond to the key argument specified when the member referred to was added to the collection.

Image Object

Properties

DocObject object properties

Brightness

FileName

HorzFlip

VertFlip

Negative

VariableName

VariableObject

Methods

DocObject object methods

Object Properties

► Image.Brightness

This adjustment influences the color reduction process. Use this property to print a color image on a noncolor printer.

Access Read/Write.

Type VT_I2 or Integer (between -255 and +255).

► Image.FileName

Retrieves or sets the filename of the image.

Access Read/Write.

Type VT_BSTR or String.

► Image.VertFlip

Displays the image as if it is reflected in a mirror.

Reflection axis is vertical.

Access Read/Write.

Type VT_BOOL or Boolean.

► Image.HorzFlip

Displays the image as if it is reflected in a mirror.

Reflection axis is horizontal.

Access Read/Write.

Type VT_BOOL or Boolean.

► Image.Negative

Prints the image negatively.

Access Read/Write.

Type VT_BOOL or Boolean.

► Image.VariableName

Retrieves or sets the current variable name associated with the image.

Access Read/Write.

Type VT_BSTR or String.

► Image.VariableObject

Retrieves or sets the current Variable object associated with the image.

Access Read/Write.

Type VT_DISPATCH or Variable object.

Barcodes Collection

Properties

Application
Count
Parent

Methods

Add
Item (Default)

Object Properties

► **Barcodes.Count**

Returns the number of items in the specified collection.

Access Read-Only.

Type VT_I2 or Integer.

► **Barcodes.Application**

This property returns the Application object that represents the root object of the hierarchy.

Access Read-Only.

Type VT_DISPATCH or Application object.

► **Barcodes.Parent**

Returns the parent object of the specified object.

Access Read-Only.

Type VT_DISPATCH.

Object Methods

► **Barcodes.Add**

VTS_DISPATCH or Barcode **Add**(*strBarcodeName*).

Adds a new Barcode object to the collection.

Return value: Returns a Barcode object.

Parameters:

strBarcodeName Optional VT_BSTR or String. The name of the object to add.

► **Barcodes.Item**

VTS_DISPATCH or Barcode **Item**(*varIndex*).

Returns a member of a collection, either by position or by name.

Note

If the value provided as Index does not match any existing member of the collection, no object is returned.

Parameters:

varIndex Required VT_VARIANT or Variant. The name or index number of a member of the collection.

The index can be a numeric expression (a number from 1 to the value of the collection's Count property), a constant, or a string.

Note

If the value provided as Index doesn't match any existing member of the collection, an error occurs. The Item method is the default method for collections. The following two lines of code are equivalent.

```
Object. Barcodes (1)
Object. Barcodes.Item(1)
```

► **Barcodes.Remove**

VTS_NONE **Remove** (*varIndex*).

Removes a member from the collection.

Parameters:

varIndex Required VT_VARIANT or Variant. An expression that specifies the position of a member of the collection. If a numeric expression, index must be a number from 1 to the value of the collection's Count property. If a string expression, index must correspond to the key argument specified when the member referred to was added to the collection.

Barcode Object

Properties

DocObject object properties

BarHeight
 CheckMode
 Code2D
 Device
 HRAAlignment
 HRCheckCharacter
 HRFont
 HRFreeTextObject
 HRDevice
 HRGap
 HRPosition
 NarrowBarWidth
 Ratio
 Symbology
 Value
 VariableName
 VariableObject

Methods

DocObject object methods

Object Properties

► Barcode.BarHeight

Retrieves or sets the bar height of the barcode (in MeasureSystem unit).

Access Read/Write.

Type VT_I4 or Long.

▶ Barcode.CheckMode

Retrieves or sets the check control of the barcode (in enum-CheckMode type).

Access Read/Write.

Type VT_I4 or Long or enumCheckMode type.

The value can be one of the following:

<code>lppxCheckModeNone</code>	= 0
<code>lppxCheckMode1Digit</code>	= 1
<code>lppxCheckMode2Digit</code>	= 2
<code>lppxCheckModeMod11Mod10</code>	= 3

▶ Barcode.Code2D

Retrieves the Code2D object for 2D barcodes.

Access Read-Only.

Type VT_DISPATCH or Code2D object.

▶ Barcode.Device

Determines if the barcode is graphical or generated by the printer.

Access Read/Write.

Type VT_BOOL or Boolean.

▶ Barcode.HRAalignment

Retrieves or sets the current human readable alignment.

Access Read/Write.

Type VT_I4 or Long or enumAlignment type.

The value can be one of the following:

<code>lppxAlignLeft</code>	= 0
<code>lppxAlignCenter</code>	= 1
<code>lppxAlignRight</code>	= 2

▶ Barcode.HRCheckCharacter

Includes or not the check character control in the human readable.

Access Read/Write.

Type VT_BOOL or Boolean.

▶ Barcode.HRDevice

Determines if the human readable is printer generated or not.

Access Read/Write.

Type VT_BOOL or Boolean.

▶ Barcode.HRFont

Retrieves or sets the font of the human readable.

Access Read/Write.

Type VT_FONT or StdFont object.

▶ Barcode.HRFreeTextObject

Retrieves the Text object representing the text of the human readable.

Access Read-Only.

Type VT_DISPATCH or Text object.

▶ Barcode.HRGap

Retrieves or sets the gap between the barcode and its human readable (in MeasureSystem unit).

Access Read/Write.

Type VT_I4 or Long.

▶ Barcode.HRPosition

Sets or retrieves the position of the human readable.

Access Read/Write.

Type VT_I4 or Long or enumHRPosition type.

The value can be one of the following:

lppxHRPositionNone	= 0
lppxHRPositionBelow	= 1
lppxHRPositionAbove	= 2
lppxHRPositionFree	= 3

▶ Barcode.NarrowBarWidth

Retrieves or sets the narrow bar width of the barcode (in MeasureSystem unit).

Access Read/Write.

Type VT_I4 or Long.

▶ Barcode.Ratio

Retrieves or sets the ratio of the barcode (between 20 and 35).

Access Read/Write.

Type VT_I4 or Long.

▶ Barcode.Symbology

Retrieves or sets the symbology of barcode.

Access Read/Write.

Type VT_I4 or Long or enumSymbology type.

The value can be one of the following (depending on product and printer):

LppxCode11	49
lppx25Interleave	50
LppxCode39	51
LppxCode49	52
LppxMaxicode	53
LppxCode16K	54
LppxGermanPostcode	55
LppxEAN8	56
LppxUPCE	57
LppxBC412	58
LppxMicroPDF	59
LppxCode93	65
lppx25Beared	66
LppxCode128	67
LppxEAN128	68
LppxEAN13	69
LppxCode39Full	70
LppxCode128Auto	71
LppxCodablockF	72
lppx25Industrial	73
lppx25Standard	74
LppxCodabar	75
LppxLogmars	76
LppxMsi	77
LppxCodablockA	78
LppxPostnet	79
LppxPlessey	80
LppxCode128SSCC	81
LppxUPCExtended	83
LppxUPCA	85
LppxUPCEXT2	86
LppxUPCEXT5	87
LppxCode25PRDG	88
LppxUPCWEIGHT	89
LppxUPCEPLUS2	97

LppxUPCEPLUS5	98
LppxUPCAPLUS2	99
LppxUPCAPLUS5	100
LppxEAN8PLUS2	101
LppxEAN8PLUS5	102
LppxEAN13PLUS2	103
LppxEAN13PLUS5	104
LppxITF	105
lppx25MatrixEuropean	106
lppx25MatrixJapan	107
LppxDatamatrix	120
lppxItf14	121
LppxPdf	122
LppxQrcode	123
LppxRss	124
LppxComposite	125

Note

Depending on the product, not all symbologies are available.

► **Barcode.Value**

Retrieves or sets the value of the barcode.

Access Read/Write.

Type VT_BSTR or String.

► **Barcode.VariableName**

Retrieves or sets the current variable name associated with the barcode.

Access Read/Write.

Type VT_BSTR or String.

▶ Barcode.VariableObject

Retrieves or sets the current Variable object associated with the barcode.

Access Read/Write.

Type VT_DISPATCH or Variable object.

Code2D Object

Properties

Columns
ECC
ModuleX
ModuleY
Rows

Methods

SetOption

Object Properties

► Code2D.Columns

Sets or retrieves the column count of the current 2D code.

Access Read/Write.

Type VT_I2 or Integer.

► Code2D.ECC

Sets or retrieves the Security attribute of the current 2D code (see annexes).

Access Read/Write.

Type VT_I2 or Integer.

► Code2D.ModuleX

Sets or retrieves the thickness of the ModuleX attribute of the current 2D code.

Access Read/Write.

Type VT_I2 or Integer.

► Code2D.ModuleY

Sets or retrieves the thickness of the ModuleY attribute of the current 2D code.

Access Read/Write.

Type VT_I2 or Integer.

▶ Code2D.Rows

Sets or retrieves the row count of the current 2D code.

Access Read/Write.

Type VT_I2 or Integer.

**Object
Methods****▶ Code2D.SetOption**

VTS_NONE **Code2D.SetOption**(*strOptionName*, *varOptionValue*).

Sets option of the current 2D code.

Parameters:

strOptionName Required VT_BSTR or String. Name of the option to set.

varOptionValue Required VT_VARIANT or Variant. Value of the option to set.

Texts Collection

Properties

Application
Count
Parent

Methods

Add
Item (Default)

Object properties

► Texts.Application

This property returns the Application object that represents the root object of the hierarchy.

Access Read-Only.

Type VT_DISPATCH or Application object.

► Texts.Count

Returns the number of items in the specified collection.

Access Read-Only.

Type VT_I2 or Integer.

► Texts.Parent

Returns the parent object of the specified object.

Access Read-Only.

Type VT_DISPATCH.

Object Methods

► Texts.Add

VT_DISPATCH or Text **Add**(*strTextName*).

Adds a new Text object to the collection.

Return value: Returns a Text object.

Parameters:

strTextName Optional VT_BSTR or String. The name of the object to add.

► Texts.Item

VTS_DISPATCH or Text **Item**(*varIndex*).

Returns a member of a collection, either by position or by name.

Note

If the value provided as Index does not match any existing member of the collection, no object is returned.

Parameters:

varIndex Required VT_VARIANT or Variant. The name or index number of a member of the collection.

The index can be a numeric expression (a number from 1 to the value of the collection's Count property), a constant, or a string.

Note

If the value provided as Index doesn't match any existing member of the collection, an error occurs. The Item method is the default method for collections. Therefore, the following two lines of code are equivalent.

```
Object.Texts(1)  
Object.Texts.Item(1)
```

► Texts.Remove

VTS_NONE **Remove** (*varIndex*).

Removes a member from the collection.

Parameters:

varIndex Required VT_VARIANT or Variant. An expression that specifies the position of a member of the collection. If a numeric expression, index must be a number from 1 to the value of the collection's Count property. If a string expression, index must correspond to the key argument specified when the member referred to was added to the collection.

Text Object

Properties

Alignment
FitToFrame
Font
Value
VariableName
VariableObject
WordHyphenation
WordWrap
SelText

Methods

DocObject object methods
AppendCRLF
AppendString
AppendTextObject
AppendVariable
Copy
InsertCRLF
InsertString
InsertTextObject
InsertVariable
Paste

Object Properties

► Text.Alignment

Sets or retrieves current alignment (in [enumAlignment](#) type).

Access Read/Write.

Type VT_I2 or Integer or [enumAlignment](#) type.

► Text.FitToFrame

Sets or retrieves fit to frame option.

Access Read/Write.

Type VT_BOOL or [Boolean](#).

► Text.Font

Sets or retrieves text font.

Access Read/Write.

Type VT_FONT or [StdFont](#) object.

► Text.Value

Sets or retrieves global value of the object.

Access Read/Write.

Type VT_BSTR or String.

► Text.VariableName

Retrieves or sets the current variable name associated with the text.

Access Read/Write.

Type VT_BSTR or String.

► Text.VariableObject

Retrieves or sets the current Variable object associated with the text.

Access Read/Write.

Type VT_DISPATCH or Variable object.

► Text.WordHyphenation

Retrieves or sets the WordHyphenation option.

Access Read/Write.

Type VT_BOOL or Boolean.

► Text.WordWrap

Retrieves or sets the WordWrap option.

Access Read/Write.

Type VT_BOOL or Boolean.

► Text.Select

Retrieves the current selection of text if any.

Access Read-Only.

Type VT_DISPATCH or TextSelection object.

**Object
Methods****► Text.AppendCRLF**

VTS_NONE **AppendCRLF** (*fntFont*).

Appends a <CrLf> at the end of the text.

Parameters:

fntFont Optional VT_FONT or StdFont. Font associated with the Carriage return.

► Text.AppendString

VTS_NONE **AppendString**(*strString*, *fntFont*).

Appends a string at the end of the text.

Parameters:

strString Required VT_BSTR or String. String to append.

fntFont Optional VT_FONT or StdFont. Font associated with the string.

► Text.AppendTextObject

VTS_NONE **AppendTextObject**(*objectText*).

Appends a Text object at the end of the text.

Parameters:

objectText Required VT_DISPATCH or Text object. Text object to append.

► Text.AppendVariable

VTS_NONE **AppendVariable**(*objectVariable*, *fontFont*).

Appends a Variable object at the end of the text.

Parameters:

objectVariable Required VT_DISPATCH or Variable object.
Variable object to append.

fontFont Optional VT_FONT or StdFont. Font associated with the Variable object.

► Text.Copy

VTS_NONE **Copy**().

Copies the Text object to the clipboard.

Note

About « Insert » functions :
Positions starts at 1.
A variable object equals 1 position.
A CarriageReturn field equals 1 position.

► Text.InsertCRLF

VTS_NONE **InsertCRLF** (*longPosition*, *fontFont*).

Inserts a carriage return at the position *longPosition*.

Parameters:

longPosition Optional VT_I4 or Long. Position to insert CarriageReturn.

fontFont Optional VT_FONT or StdFont. Font associated with the CarriageReturn field.

► Text. InsertString

VTS_NONE **InsertString** (*strString*, *longPosition*, *fntFont*).

Inserts a string at the position *longPosition*.

Parameters:

strString Required VT_BSTR or String. String to insert.

longPosition Optional VT_I4 or Long. Position to insert string.

fntFont Optional VT_FONT or StdFont. Font associated with the string.

► Text. InsertTextObject

VTS_NONE **InsertTextObject** (*objectText*, *longPosition*).

Inserts a Text object at the position *longPosition*.

Parameters:

objectText Required VT_DISPATCH or Text object. Text to insert.

longPosition Optional VT_I4 or Long. Position to insert Text object.

► Text. InsertVariable

VTS_NONE **InsertVariable** (*objectVariable*, *longPosition*, *fntFont*).

Inserts a Variable object at the position *longPosition*.

Parameters:

objectVariable Required VT_DISPATCH or Variable object. Variable to insert.

longPosition Optional VT_I4 or Long. Position to insert Variable object.

fntFont Optional VT_FONT or StdFont. Font associated with the Variable object.

TextSelection Object

Properties

BackColor
Font
ForeColor
Value
IsEmpty

Methods

Copy
Cut
Paste
Select

Object Properties

► TextSelection.BackColor

Retrieves or sets back color of the selected text if any.

Access Read/Write.

Type VT_I4 or Long.

► TextSelection.Font

Retrieves or sets font of the select text if any.

Access Read/Write.

Type VT_DISPATCH or StdFont.

► TextSelection.ForeColor

Retrieves or sets fore color of the selected text if any.

Access Read/Write.

Type VT_I4 or Long.

► TextSelection.IsEmpty

Tests if the selected text exists.

Access Read/Write.

Type VT_BOOL or Boolean.

► TextSelection.Value

Retrieves or sets value of the selected text if any.

Access Read/Write.

Type VT_BSTR or String.

**Object
Methods****► TextSelection.Copy**

VTS_NONE **Copy**().

Copies the selected text, if any, to the clipboard.

► TextSelection.Cut

VTS_NONE **Cut**().

Cuts the selected text, if any, and copies it to the clipboard.

► TextSelection.Paste

VTS_NONE **Paste**().

Pastes the text from clipboard into the current selected text.

► TextSelection.Select

VTS_NONE **Select**(*longFirstPosition*, *longLastPosition*).

Selects the text from *longFirstPosition* to *longLastPosition*.

Parameters:

longFirstPosition Optional VT_I4 or Long. Index of the first position of the text to select (starts with 1).

longLastPosition Optional VT_I4 or Long. Index of the last position of the text to select.

Note

If *longLastPosition* is omitted the text selection starts with *longFirstPosition* and ends with the end of the text.
If *longFirstPosition* and *longLastPosition* are omitted, all the text is selected.

OLEObjects Collection

Properties

Application
Count
Parent

Methods

Add
Item (Default)

Object Properties

► OLEObjects.Count

Returns the number of items in the specified collection.

Access Read-Only.

Type VT_I2 or Integer.

► OLEObjects.Application

This property returns the Application object that represents the root object of the hierarchy.

Access Read-Only.

Type VT_DISPATCH or Application object.

► OLEObjects.Parent

Returns the parent object of the specified object.

Access Read-Only.

Type VT_DISPATCH.

Object Methods

► OLEObjects.Add

VT_DISPATCH or OLEObject **Add**(*strOLEObjectName*).

Adds a new OLEObject object to the collection.

Return value: Returns a OLEObject object.

Parameters:

strOLEObjectName Required VT_BSTR or String. The name of the object to add.

► OLEObjects.Item

VTS_DISPATCH or OLEObject **Item**(*varIndex*).

Returns a member of a collection, either by position or by name.

Note

If the value provided as Index does not match any existing member of the collection, no object is returned.

Parameters:

varIndex Required VT_VARIANT or Variant. The name or index number of a member of the collection.

The index can be a numeric expression (a number from 1 to the value of the collection's Count property), a constant, or a string.

Note

If the value provided as Index doesn't match any existing member of the collection, an error occurs. The Item method is the default method for collections. Therefore, the following two lines of code are equivalent.

```
Object. OLEObjects (1)  
Object. OLEObjects.Item(1)
```

► OLEObjects.Remove

VTS_NONE **Remove** (*varIndex*).

Removes a member from the collection.

Parameters:

varIndex Required VT_VARIANT or Variant. An expression that specifies the position of a member of the collection. If a numeric expression, index must be a number from 1 to the value of the collection's Count property. If a string expression, index must correspond to the key argument specified when the member referred to was added to the collection.

OLEObject Object

Properties

DocObject object properties
Object

Methods

DocObject object methods
EmbedFile
LinkFile
ConnectServer

Object Properties

► OLEObject.Object

Returns directly the **IDispatch** interface of the object.

Access Read-Only.

Type VT_DISPATCH.

Object Methods

► OLEObject. EmbedFile

VTS_BOOL or Boolean **EmbedFile**(*strOLEFileNameToConnect*).

Connects the object to a file (see Insert OLE object dialog box).

Return value: Returns the result of the connection.

Parameters:

strOLEFileNameToConnect Required VT_BSTR or String. The name of the filename to connect to.

► OLEObject. LinkFile

VTS_BOOL or Boolean **LinkFile**(*strOLELinkFileNameToConnect*).

Connects the object to a file (see Insert OLE object dialog box).

Return value: Returns the result of the connection.

Parameters:

strOLELinkFileNameToConnect Required VT_BSTR or String. The name of the linkfilename to connect to.

▶ OLEObject. ConnectServer

VTS_BOOL or Boolean **ConnectServer**(*strCLSIDorPROGID-ServerName*).

Connects the object to an OLE server (see Insert OLE object dialog box).

Return value: Returns the result of the connection.

Parameters:

strCLSIDorPROGIDServerName Required VT_BSTR or String. The CLSID with brackets or directly the name of the OLE server.

Shapes Collection

Properties

Application
Count
Parent

Methods

AddEllipse
AddLine
AddOblique
AddPolygon
AddRectangle
AddRoundRect
Item (Default)
Remove

Object Properties

► Shapes.Count

Returns the number of items in the specified collection.

Access Read-Only.

Type VT_I2 or [Integer](#).

► Shapes.Application

This property returns the [Application](#) object that represents the root object of the hierarchy.

Access Read-Only.

Type VT_DISPATCH or [Application](#) object.

► Shapes.Parent

Returns the parent object of the specified object.

Access Read-Only.

Type VT_DISPATCH.

Object Methods

► Shapes.AddEllipse

VTS_DISPATCH or Shape **AddEllipse**(*longLeft*, *longTop*,
longRight, *longBottom*).

Adds a new Shape object to the collection.

Return value: Returns a Shape object.

Parameters:

longLeft Required VT_I4 or Long. Sets the left corner of the bounding rectangle of the object (in MeasureSystem unit).

longTop Required VT_I4 or Long. Sets the top corner of the bounding rectangle of the object (in MeasureSystem unit).

longRight Required VT_I4 or Long. Sets the right corner of the bounding rectangle of the object (in MeasureSystem unit).

longBottom Required VT_I4 or Long. Sets the bottom corner of the bounding rectangle of the object (in MeasureSystem unit).

► Shapes.AddLine

VTS_DISPATCH or Shape **AddLine**(*longLeft*, *longTop*,
longRight, *longBottom*).

Adds a new Shape object to the collection.

Return value: Returns a Shape object.

Parameters:

longLeft Required VT_I4 or Long. Sets the left corner of the bounding rectangle of the object (in MeasureSystem unit).

longTop Required VT_I4 or Long. Sets the top corner of the bounding rectangle of the object (in MeasureSystem unit).

longRight Required VT_I4 or Long. Sets the right corner of the bounding rectangle of the object (in MeasureSystem unit).

longBottom Required VT_I4 or Long. Sets the bottom corner of the bounding rectangle of the object (in MeasureSystem unit).

► Shapes.AddOblique

VTS_DISPATCH or Shape **AddOblique**(*longLeft*, *longTop*, *longRight*, *longBottom*).

Adds a new Shape object to the collection.

Return value: Returns a Shape object.

Parameters:

longLeft Required VT_I4 or Long. Sets the left corner of the bounding rectangle of the object (in MeasureSystem unit).

longTop Required VT_I4 or Long. Sets the top corner of the bounding rectangle of the object (in MeasureSystem unit).

longRight Required VT_I4 or Long. Sets the right corner of the bounding rectangle of the object (in MeasureSystem unit).

longBottom Required VT_I4 or Long. Sets the bottom corner of the bounding rectangle of the object (in MeasureSystem unit).

► Shapes.AddRectangle

VTS_DISPATCH or Shape **AddRectangle**(*longLeft*, *longTop*, *longRight*, *longBottom*).

Adds a new Shape object to the collection.

Return value: Returns a Shape object.

Parameters:

longLeft Required VT_I4 or Long. Sets the left corner of the bounding rectangle of the object (in MeasureSystem unit).

longTop Required VT_I4 or Long. Sets the top corner of the bounding rectangle of the object (in MeasureSystem unit).

longRight Required VT_I4 or Long. Sets the right corner of the bounding rectangle of the object (in MeasureSystem unit).

longBottom Required VT_I4 or Long. Sets the bottom corner of the bounding rectangle of the object (in MeasureSystem unit).

► Shapes.AddRoundRect

VTS_DISPATCH or Shape **AddRoundRect**(*longLeft*,
longTop, *longRight*, *longBottom*, *longCorner*).

Adds a new Shape object to the collection.

Return value: Returns a Shape object.

Parameters:

longLeft Required VT_I4 or Long. Sets the left corner of the bounding rectangle of the object (in MeasureSystem unit).

longTop Required VT_I4 or Long. Sets the top corner of the bounding rectangle of the object (in MeasureSystem unit).

longRight Required VT_I4 or Long. Sets the right corner of the bounding rectangle of the object (in MeasureSystem unit).

longBottom Required VT_I4 or Long. Sets the bottom corner of the bounding rectangle of the object (in MeasureSystem unit).

LongCorner Required VT_I4 or Long. Sets the radius of the corner (distance from bound) (in MeasureSystem unit).

► Shapes.AddPolygon

VTS_DISPATCH or Shape **AddPolygon**(*varSafeArrayOfPoints*).

Adds a new Shape object to the collection.

Return value: Returns a Shape object.

Parameters:

varSafeArrayOfPoints Required VT_VARIANT/VT_ARRAY or Variant. Sets the list of points of the object (in MeasureSystem unit).

Note

It can be a one-dimensional array of values or a two-dimensional array of values.

► **Shapes.Item**

VTS_DISPATCH or Shape **Item**(*varIndex*).

Returns a member of a collection, either by position or by name.

Note

If the value provided as Index does not match any existing member of the collection, no object is returned.

Parameters:

varIndex Required VT_VARIANT or Variant. The name or index number of a member of the collection.

The index can be a numeric expression (a number from 1 to the value of the collection's Count property), a constant, or a string.

Note

If the value provided as Index doesn't match any existing member of the collection, an error occurs. The Item method is the default method for collections. Therefore, the following two lines of code are equivalent.

```
Object.Shapes(1)
Object.Shapes.Item(1)
```

► **Shapes.Remove**

VTS_NONE **Remove** (*varIndex*).

Removes a member from the collection.

Parameters:

varIndex Required VT_VARIANT or Variant. An expression that specifies the position of a member of the collection. If a numeric expression, index must be a number from 1 to the value of the collection's Count property. If a string expression, index must correspond to the key argument specified when the member referred to was added to the collection.

Shape Object

Properties

DocObject object properties
LineWidth

Methods

DocObject object methods
SetPoints

Object Properties

► Shape.LineWidth

Sets or retrieves the thickness of the bounds of the object.

Access Read/Write.

Type VT_I4 or Long.

Object Methods

► Shape.SetPoints

VTS_NONE **SetPoints** (*varArrayOfPoints*).

Sets all points describing the current object.

Parameters:

varArrayOfPoints Required VT_VARIANT/VT_ARRAY or Variant. An expression that evaluates to an array of points.

Note

In order to have a closed polygon, last point must match first point.

Variables Collection

Properties

Application
 Count
 Parent
 Counters
 DatabaseVariables
 Dates
 FormVariables
 Formulas
 FeeVariables
 TableLookups

Methods

Add
 Item (Default)
 Remove

Object Properties

► Variables.Application

This property returns the [Application](#) object that represents the root object of the hierarchy.

Access Read-Only.

Type VT_DISPATCH or [Application](#) object.

► Variables.Count

Returns the number of items in the specified collection.

Access Read-Only.

Type VT_I2 or [Integer](#).

► Variables.Parent

Returns the parent object of the specified object.

Access Read-Only.

Type VT_DISPATCH.

► Variables.Counters

Returns the Counters collection that represents all the created Counter variables in the document.

Access Read-Only.

Type VT_DISPATCH or Counters collection.

► Variables.DatabaseVariables

Returns the DatabaseVariables collection that represents all the created Free variables with database link in the document.

Access Read-Only.

Type VT_DISPATCH or DatabaseVariables collection.

► Variables.Dates

Returns the Dates collection that represents all the created Date variables in the document.

Access Read-Only.

Type VT_DISPATCH or Dates collection.

► Variables.FormVariables

Returns the FormVariables collection that represents all the created Free variables with form attribute in the document.

Access Read-Only.

Type VT_DISPATCH or FormVariables collection.

► Variables.Formulas

Returns the Formulas collection that represents all the created Formula variables in the document.

Access Read-Only.

Type VT_DISPATCH or Formulas collection.

► Variables.FreeVariables

Returns the FreeVariables collection that represents all the created Free variables in the document.

Access Read-Only.

Type VT_DISPATCH or FreeVariables collection.

► Variables.TableLookups

Returns the TableLookups collection that represents all the created TableLookup variables in the document.

Access Read-Only.

Type VT_DISPATCH or TableLookups collection.

**Object
Methods****► Variables.Add**

VT_DISPATCH or Variable **Add**(*VariableDataSource*, *strVariableName*).

Adds a new Variable object to the current document.

Return value: Returns a Variable object.

Parameters:

strVariableDataSource Required VT_I4 or Long. The data source type of the variable to add.

Can be one of the following values (enumDataSource type):

<code>lppxDataSourceCounter</code>	= 1
<code>lppxDataSourceTableLookup</code>	= 2
<code>lppxDataSourceDate</code>	= 3
<code>lppxDataSourceFormula</code>	= 4
<code>lppxDataSourceFree</code>	= 5
<code>lppxDataSourceForm</code>	= 6
<code>lppxDataSourceDataBase</code>	= 7

strVariableName Optional VT_BSTR or String. The name of the variable to add.

► Variables.Item

VTS_DISPATCH or Variable **Item**(*varIndex*).

Returns a member of a collection, either by position or by name.

Note

If the value provided as Index does not match any existing member of the collection, no object is returned.

Parameters:

varIndex Required VT_VARIANT or Variant. The name or index number of a member of the collection.

The index can be a numeric expression (a number from 1 to the value of the collection's Count property), a constant, or a string.

Note

If the value provided as Index doesn't match any existing member of the collection, an error occurs. The Item method is the default method for collections. Therefore, the following two lines of code are equivalent.

```
Object.Variables(1)  
Object.Variables.Item(1)
```

► Variables.Remove

VTS_NONE **Remove** (*varIndex*).

Removes a member from the collection.

Parameters:

varIndex Required VT_VARIANT or Variant. An expression that specifies the position of a member of the collection. If a numeric expression, index must be a number from 1 to the value of the collection's Count property. If a string expression, index must correspond to the key argument specified when the member referred to was added to the collection.

Variable Object

Properties

Application
DataSource
Name
Parent
Value (Default)

Methods

(None)

Object Properties

► Variable.Application

This property returns the Application object that represents the root object of the hierarchy.

Access Read-Only.

Type VT_DISPATCH or Application object.

► Variable.DataSource

Returns the kind of data source of the variable (enumDataSource type).

Access Read-Only.

Type VT_I2 or Integer or enumDataSource type.

► Variable.Name

Returns the name of the variable.

Access Read/Write.

Type VT_BSTR or String.

► Variable.Parent

Returns the parent object of the specified object.

Access Read-Only.

Type VT_DISPATCH.

▶ Variable.Value

Returns the current value of the variable.

Access Read/Write.

Type VT_BSTR or String.

TableLookups Collection

Properties

Application
Count
Parent

Methods

Add
Item (Default)
Remove

Object Properties

▶ **TableLookups.Count**

Returns the number of items in the specified collection.

Access Read-Only.

Type VT_I2 or Integer.

▶ **TableLookups.Application**

This property returns the Application object that represents the root object of the hierarchy.

Access Read-Only.

Type VT_DISPATCH or Application object.

▶ **TableLookups.Parent**

Returns the parent object of the specified object.

Access Read-Only.

Type VT_DISPATCH.

Object Methods

▶ **TableLookups.Add**

VT_DISPATCH or TableLookup **Add**(*strLinkedTableName*).

Adds a new TableLookup object to the collection.

Return value: Returns a TableLookup object.

Parameters:

strLinkedTableName Optional VT_BSTR or String. The name of the object to add.

► TableLookups.Item

VTS_DISPATCH or TableLookup **Item**(*varIndex*).

Returns a member of a collection, either by position or by name.

Note

If the value provided as Index does not match any existing member of the collection, no object is returned.

Parameters:

varIndex Required VT_VARIANT or Variant. The name or index number of a member of the collection.

The index can be a numeric expression (a number from 1 to the value of the collection's Count property), a constant, or a string.

Note

If the value provided as Index doesn't match any existing member of the collection, an error occurs. The Item method is the default method for collections. Therefore, the following two lines of code are equivalent.

```
Object.TableLookups(1)
```

```
Object.TableLookups.Item(1)
```

► TableLookups.Remove

VTS_NONE **Remove** (*varIndex*).

Removes a member from the collection.

Parameters:

varIndex Required VT_VARIANT or Variant. An expression that specifies the position of a member of the collection. If a numeric expression, index must be a number from 1 to the value of the collection's Count property. If a string expression, index must correspond to the key argument specified when the member referred to was added to the collection.

TableLookup Object

Properties

Counter object properties

CounterUse

DatabaseSource

ResultField

Keys

Length

PadLength

TableName

Methods

Counter object methods

AddKey

DeleteKey

Object Properties

► TableLookup.CounterUse

Activates or not a counter on the object.

Access Read/Write.

Type VT_BOOL or Boolean.

► TableLookup.DatabaseSource

Sets or retrieves the data source name of the linked ODBC database.

Access Read/Write.

Type VT_BSTR or String.

► TableLookup.Length

Sets or retrieves the length of the output value.

Access Read/Write.

Type VT_I4 or Long.

► TableLookup.PadLength

Sets or retrieves the number of characters to pad up to.

Access Read/Write.

Type VT_I4 or Long.

► TableLookup.ResultField

Sets or retrieves the name of the linked output field.

Access Read/Write.

Type VT_BSTR or String.

► TableLookup.Keys

Returns strings collection for keys.

Access Read-Only.

Type VT_DISPATCH or Strings.

► TableLookup.TableName

Sets or retrieves the linked table name of the current data source.

Access Read/Write.

Type VT_BSTR or String.

**Object
Methods****► TableLookup.AddKey**

VTS_NONE **AddKey** (*strSearchFieldName*, *strSearchFieldValue*).

Adds a pair (<SearchFieldName>,<SearchFieldValue>) in the Search Field List.

Parameters:

strSearchFieldName Required VT_BSTR or String. The field name.

strSearchFieldValue Required VT_BSTR or String. The value of the field (must be the name of an other variable).

▶ TableLookup.DeleteKey

VTS_NONE **DeleteKey**(*strFieldName*).

Deletes the search for the field *strFieldName*.

Parameters:

StrFieldName Required VT_BSTR or string. Fieldname to delete.

Formulas Collection

Properties

Application
 Counte
 Parent

Methods

Add
 Item (Default)
 Remove

Object Properties

► Formulas.Count

Returns the number of items in the specified collection.

Access Read-Only.

Type VT_I2 or Integer.

► Formulas.Application

This property returns the Application object that represents the root object of the hierarchy.

Access Read-Only.

Type VT_DISPATCH or Application object.

► Formulas.Parent

Returns the parent object of the specified object.

Access Read-Only.

Type VT_DISPATCH.

Object Methods

► Formulas.Add

VT_DISPATCH or Formula **Add**(*strFormulaName*).

Adds a new Formula object to the collection.

Return value: Returns a Formula object.

Parameters:

strFormulaName Optional VT_BSTR or String. The name of the object to add.

► Formulas.Item

VTS_DISPATCH or Formula **Item**(*varIndex*).

Returns a member of a collection, either by position or by name.

Note

If the value provided as Index does not match any existing member of the collection, no object is returned.

Parameters:

varIndex Required VT_VARIANT or Variant. The name or index number of a member of the collection.

The index can be a numeric expression (a number from 1 to the value of the collection's Count property), a constant, or a string.

Note

If the value provided as Index doesn't match any existing member of the collection, an error occurs. The Item method is the default method for collections. Therefore, the following two lines of code are equivalent.

```
Object.Formulas(1)  
Object.Formulas.Item(1)
```

► Formulas.Remove

VTS_NONE **Remove** (*varIndex*).

Removes a member from the collection.

Parameters:

varIndex Required VT_VARIANT or Variant. An expression that specifies the position of a member of the collection. If a numeric expression, index must be a number from 1 to the value of the collection's Count property. If a string expression, index must correspond to the key argument specified when the member referred to was added to the collection.

Formula Object

Properties

Counter object properties

CounterUse

Expression

Length

PadLength

Methods

Counter object methods

Test method

Object Properties

► **Formula.CounterUse**

Activates or not counting on the object.

Access Read/Write.

Type VT_BOOL or Boolean.

► **Formula.Expression**

Sets or retrieves the format of the Formula object.

Access Read/Write.

Type VT_BSTR or String.

► **Formula.Length**

Sets or retrieves the length of the output value.

Access Read/Write.

Type VT_I4 or Long.

► **Formula.PadLength**

Sets or retrieves the number of characters to pad up to.

Access Read/Write.

Type VT_I4 or Long.

**Object
Methods****► Formula.Test**

VTS_BOOL or Boolean **Test()**.

Tests the validity of the formula.

Return value: Returns a boolean which indicate if the Formula object format is valid or not.

Dates Collection

Properties

Application
Count
Parent

Methods

Add
Item (Default)
Remove

Object Properties

► Dates.Count

Returns the number of items in the specified collection.

Access Read-Only.

Type VT_I2 or [Integer](#).

► Dates.Application

This property returns the [Application](#) object that represents the root object of the hierarchy.

Access Read-Only.

Type VT_DISPATCH or [Application](#) object.

► Dates.Parent

Returns the parent object of the specified object.

Access Read-Only.

Type VT_DISPATCH.

Object Methods

► Dates.Add

VT_DISPATCH or [Date](#) **Add**(*strDateName*).

Adds a new [Date](#) object to the collection.

Return value: Returns a [Date](#) object.

Parameters:

strDateName Optional VT_BSTR or [String](#). The name of the object to add.

► Dates.Item

VTS_DISPATCH or Date **Item**(*varIndex*).

Returns a member of a collection, either by position or by name.

Note

If the value provided as Index does not match any existing member of the collection, no object is returned.

Parameters:

varIndex Required VT_VARIANT or Variant. The name or index number of a member of the collection.

The index can be a numeric expression (a number from 1 to the value of the collection's Count property), a constant, or a string.

Note

If the value provided as Index doesn't match any existing member of the collection, an error occurs. The Item method is the default method for collections. Therefore, the following two lines of code are equivalent.

```
Object.Dates(1)  
Object.Dates.Item(1)
```

► Dates.Remove

VTS_NONE **Remove** (*varIndex*).

Removes a member from the collection.

Parameters:

varIndex Required VT_VARIANT or Variant. An expression that specifies the position of a member of the collection. If a numeric expression, index must be a number from 1 to the value of the collection's Count property. If a string expression, index must correspond to the key argument specified when the member referred to was added to the collection.

Date Object

Properties

Variable object properties

Device

Format

Methods

Variable object methods

Object Properties

► Date.Device

Determines if the date is generated by the printer or not.

Access Read/Write.

Type VT_BOOL or Boolean.

► Date.Format

Sets or retrieves the format of the value.

Access Read/Write.

Type VT_BSTR or String.

Can contain a prefix code which indicates country code:

Prefix Code	Country
(UK)	English(default)
(GE)	German
(IT)	Italian
(FR)	French
(SW)	Swedish
(SP)	Spanish
(CH)	Chinese
(DK)	Danish
(JP)	Japanese

Followed by the real format:

Commands	Value
'm'	month as 1...12
'mm'	month as 01...12
'mmm'	month as Jan...Dec
'mmmm'	month as January...December
'd'	day of month as 1.31
'dd'	day of month as 01..31
'ddd'	day of week as 0..6 (0=Sunday, 1=Monday,...)
'dddd'	day of week as Sun...Sat
'ddddd'	day of week as Sunday...Saturday
'j'	Julian day as 1...366
'jj'	Julian day as 001...366
'y'	year as 0...9
'yy'	year as 00...99
'yyyy'	year as 1900...2040
'w'	week number as 1...53
'ww'	week number as 01...53
'h'	hour as 0...23
'hh'	hour as 00...23
'hhh'	hour as 0...12
'hhhh'	hour as 00..12
am pm	am or pm
'n'	Minutes as 0...59
'nn'	Minutes as 00...59
'c'	Separator (c = any non ambiguous character)
'ccc'	Separator string

Counters Collection

Properties

Application
Count
Parent

Methods

Add
Item (Default)
Remove

Object Properties

► Counters.Count

Returns the number of items in the specified collection.

Access Read-Only.

Type VT_I2 or Integer.

► Counters.Application

This property returns the Application object that represents the root object of the hierarchy.

Access Read-Only.

Type VT_DISPATCH or Application object.

► Counters.Parent

Returns the parent object of the specified object.

Access Read-Only.

Type VT_DISPATCH.

Object Methods

► Counters.Add

VT_DISPATCH or Counter **Add**(*strCounterName*).

Adds a new Counter object to the collection.

Return value: Returns a Counter object.

Parameters:

strCounterName Optional VT_BSTR or String. The name of the object to add.

► **Counters.Item**

VTS_DISPATCH or Counter **Item**(*varIndex*).

Returns a member of a collection, either by position or by name.

Note

If the value provided as Index does not match any existing member of the collection, no object is returned.

Parameters:

varIndex Required VT_VARIANT or Variant. The name or index number of a member of the collection.

The index can be a numeric expression (a number from 1 to the value of the collection's Count property), a constant, or a string.

Note

If the value provided as Index doesn't match any existing member of the collection, an error occurs.

The Item method is the default method for collections. Therefore, the following two lines of code are equivalent.

```
Object.Counters(1)
Object.Counters.Item(1)
```

► **Counters.Remove**

VTS_NONE **Remove** (*varIndex*).

Removes a member from the collection.

Parameters:

varIndex Required VT_VARIANT or Variant. An expression that specifies the position of a member of the collection. If a numeric expression, index must be a number from 1 to the value of the collection's Count property. If a string expression, index must correspond to the key argument specified when the member referred to was added to the collection.

Counter Object

Properties

Variable object properties

NumberOfDecimals

DecimalSeparator

DecimalUse

ThousandSeparator

Increment

ISO

BaseType

CustomSet

MaxValue

ResetToValue

PadCharacter

Prefix

Suffix

TriggerMode

TriggerParameter

Methods

Variable object methods

Object Properties

► Counter. NumberOfDecimals

Sets or retrieves the number of digits after the decimal point.

Access Read/Write.

Type VT_I4 or Long.

► Counter. DecimalSeparator

Sets or retrieves the decimal separator.

Access Read/Write.

Type VT_BSTR or String.

▶ Counter. DecimalUse

Uses or not decimal formatting.

Access Read/Write.

Type VT_BOOL or Boolean.

▶ Counter. ThousandSeparator

Sets or retrieves the thousand separator.

Access Read/Write.

Type VT_BSTR or String.

▶ Counter. Increment

Sets or retrieves the increment.

Access Read/Write.

Type VT_VARIANT or Variant.

▶ Counter. ISO

Uniquely numbers each copy of a label.

Access Read/Write.

Type VT_BOOL or Boolean.

▶ Counter. BaseType

Sets or retrieves the counting base used for counting.

Access Read/Write.

Type VT_I2 or Integer or enumBase type.

The value can be one of the following:

lppxBinary	= 2
lppxBaseOctal	= 8
lppxBaseDecimal	= 10
lppxBaseHexadecimal	= 16
lppxBaseAlphabetic	= 26
lppxBaseAlphaNumeric	= 36
lppxBaseCustom	= 255

► Counter. TriggerMode

Sets the Increment method:

Access Read/Write.

Type VT_I2 or Integer or enumTriggerMode type.

The value can be one of the following:

lppxNumberOfPrintedLabels = 1

lppxResetOfAnotherCounter = 2

(Default set to lppxNumberOfPrintedLabels)

► Counter. TriggerParameter

Sets the parameter for the trigger mode.

Access Read/Write.

Type VT_VARIANT or Variant.

Note

Can be the number of labels printed or the name of an other counter.

By default, it is the number of labels printed with a value set to 1.

► Counter.CustomSet

Sets or retrieves the characters list of the custom counter (ex: « 0123456789ABCD »).

The first element must be the neutral element.

Access Read/Write.

Type VT_BSTR or String.

► Counter. MaxValue

Sets or retrieves the Max value of the current object.

When reached, this value will trigger a reset of the counter.

Access Read/Write.

Type VT_VARIANT or Variant.

▶ Counter.ResetToValue

Sets or retrieves the reset value.

Access Read/Write.

Type VT_VARIANT or Variant.

▶ Counter.PadCharacter

Sets or retrieves the character used to pad the left of variable value.

Access Read/Write.

Type VT_BSTR or String.

▶ Counter.Prefix

Sets or retrieves the prefix string added to the variable.

Access Read/Write.

Type VT_BSTR or String.

▶ Counter.Suffix

Sets or retrieves the suffix string added to the variable.

Access Read/Write.

Type VT_BSTR or String.

FreeVariables Collection

Properties	Methods
Application Count Parent	Add item (Default) Remove

Object Properties

► FreeVariables.Count

Returns the number of items in the specified collection.

Access Read-Only.

Type VT_I2 or Integer.

► FreeVariables.Application

This property returns the Application object that represents the root object of the hierarchy.

Access Read-Only.

Type VT_DISPATCH or Application object.

► FreeVariables.Parent

Returns the parent object of the specified object.

Access Read-Only.

Type VT_DISPATCH.

Object Methods

► FreeVariables.Add

VTS_DISPATCH or Free **Add**(*strFreeName*).

Adds a new Free object to the collection with no specific attribute.

Return value: Returns a Free object.

Parameters:

strFreeName Optional VT_BSTR or String. The name of the object to add.

► FreeVariables.Item

VTS_DISPATCH or Free **Item**(*varIndex*).

Returns a member of a collection, either by position or by name.

Note

If the value provided as Index does not match any existing member of the collection, no object is returned.

Parameters:

varIndex Required VT_VARIANT or Variant. The name or index number of a member of the collection.

The index can be a numeric expression (a number from 1 to the value of the collection's Count property), a constant, or a string.

Note

If the value provided as Index doesn't match any existing member of the collection, an error occurs.

The Item method is the default method for collections. Therefore, the following two lines of code are equivalent.

```
Object.FreeVariables(1)
Object.Freevariables.Item(1)
```

► FreeVariables.Remove

VTS_NONE **Remove** (*varIndex*).

Removes a member from the collection.

Parameters:

varIndex Required VT_VARIANT or Variant. An expression that specifies the position of a member of the collection. If a numeric expression, index must be a number from 1 to the value of the collection's Count property. If a string expression, index must correspond to the key argument specified when the member referred to was added to the collection.

Free Object

Properties

Counter object properties

CounterUse

DisplayInForm

FormOrder

FormPrompt

Inputmask

Length

PadLength

Shared

Methods

Counter object methods

Object Properties

► **Free.CounterUse**

Activates or not a counter on the object.

Access Read/Write.

Type VT_BOOL or Boolean.

► **Free.DisplayInForm**

Includes or not the current object in the Form grid.

Access Read/Write.

Type VT_BOOL or Boolean.

► **Free.FormOrder**

Specifies the order of objects in the Form grid.

Access Read/Write.

Type VT_I2 or Integer.

▶ Free.FormPrompt

Specifies the prompt associated in the Form grid.

Access Read/Write.

Type VT_BSTR or String.

▶ Free.InputMask

Specifies the format prompt associated in the Form grid.

Access Read/Write.

Type VT_BSTR or String.

▶ Free.Length

Sets or retrieves the length of the output value.

Access Read/Write.

Type VT_I4 or Long.

▶ Free.PadLength

Sets or retrieves the number of characters to pad up to.

Access Read/Write.

Type VT_I4 or Long.

▶ Free.Shared

Specifies the list of proposed values for the prompt associated in the Form grid.

Access Read/Write.

Type VT_BOOL or Boolean.

DatabaseVariables Collection

Properties

Application
Count
Parent

Methods

Add
Item (Default)
Remove

Object Properties

► DatabaseVariables.Count

Returns the number of items in the specified collection.

Access Read-Only.

Type VT_I2 or Integer.

► DatabaseVariables.Application

This property returns the Application object that represents the root object of the hierarchy.

Access Read-Only.

Type VT_DISPATCH or Application object.

► DatabaseVariables.Parent

Returns the parent object of the specified object.

Access Read-Only.

Type VT_DISPATCH.

Object Methods

► DatabaseVariables.Add

VTS_DISPATCH or Free **Add**(*strFreeName*).

Adds a new Free object to the collection with database attribute.

Return value: Returns a Free object.

Parameters:

strFreeName Optional VT_BSTR or String. The name of the object to add.

► DatabaseVariables.Item

VTS_DISPATCH or Free **Item**(*varIndex*).

Returns a member of a collection, either by position or by name.

Note

If the value provided as Index does not match any existing member of the collection, no object is returned.

Parameters:

varIndex Required VT_VARIANT or Variant. The name or index number of a member of the collection.

The index can be a numeric expression (a number from 1 to the value of the collection's Count property), a constant, or a string.

Note

If the value provided as Index doesn't match any existing member of the collection, an error occurs. The Item method is the default method for collections. Therefore, the following two lines of code are equivalent.

```
Object.DatabaseVariables(1)
Object.DatabaseVariables.Item(1)
```

► DatabaseVariables.Remove

VTS_NONE **Remove** (*varIndex*).

Removes a member from the collection.

Parameters:

varIndex Required VT_VARIANT or Variant. An expression that specifies the position of a member of the collection. If a numeric expression, index must be a number from 1 to the value of the collection's Count property. If a string expression, index must correspond to the key argument specified when the member referred to was added to the collection.

FormVariables Collection

Properties	Methods
Application Count Parent	Add Item (Default) Remove

Object Properties

► **FormVariables.Count**

Returns the number of items in the specified collection.

Access Read-Only.

Type VT_I2 or Integer.

► **FormVariables.Application**

This property returns the Application object that represents the root object of the hierarchy.

Access Read-Only.

Type VT_DISPATCH or Application object.

► **FormVariables.Parent**

Returns the parent object of the specified object.

Access Read-Only.

Type VT_DISPATCH.

Object Methods

► **FormVariables.Add**

VTS_DISPATCH or Free **Add**(*strFreeName*).

Adds a new Free object to the collection with form attribute.

Return value: Returns a Free object.

Parameters:

strFreeName Optional VT_BSTR or String. The name of the object to add.

► **FormVariables.Item**

VTS_DISPATCH or Free **Item**(*varIndex*).

Returns a member of a collection, either by position or by name.

Note

If the value provided as Index does not match any existing member of the collection, no object is returned.

Parameters:

varIndex Required VT_VARIANT or Variant. The name or index number of a member of the collection.

The index can be a numeric expression (a number from 1 to the value of the collection's Count property), a constant, or a string.

Note

If the value provided as Index doesn't match any existing member of the collection, an error occurs. The Item method is the default method for collections. Therefore, the following two lines of code are equivalent.

```
Object.FormVariables(1)
Object.FormVariables.Item(1)
```

► **FormVariables.Remove**

VTS_NONE **Remove** (*varIndex*).

Removes a member from the collection.

Index Required VT_VARIANT or Variant. An expression that specifies the position of a member of the collection. If a numeric expression, index must be a number from 1 to the value of the collection's Count property. If a string expression, index must correspond to the key argument specified when the member referred to was added to the collection.

Strings Collection

Properties

Application
Count
Parent

Methods

Add
Item (Default)
Remove

Object Properties

► **Strings.Application**

This property returns the [Application](#) object that represents the root object of the hierarchy.

Access Read-Only.

Type VT_DISPATCH or [Application](#) object.

► **Strings.Count**

Returns the number of items in the specified collection.

Access Read-Only.

Type VT_I2 or [Integer](#).

► **Strings.Parent**

Returns the parent object of the specified object.

Access Read-Only.

Type VT_DISPATCH.

Object Methods

► **Strings.Add**

VTS_NONE **Add** (*strStringValue*).

Adds a new string to the collection.

Parameters:

strStringValue Required VT_BSTR or [String](#). Specifies the string to add.

► Strings.Item

VTS_BSTR **Item**(*longIndex*).

Returns a member of a collection, either by position or by name.

Note

If the value provided as Index does not match any existing member of the collection, no object is returned.

Parameters:

longIndex Required VT_14 or Variant. The index number of a member of the collection.

The index can be a numeric expression (a number from 1 to the value of the collection's Count property), a constant, or a string.

Note

If the value provided as Index doesn't match any existing member of the collection, an error occurs. The Item method is the default method for collections. Therefore, the following two lines of code are equivalent.

```
Object.Strings(1)  
Object.Strings.Item(1)
```

► Strings.Remove

VTS_NONE **Remove** (*longIndex*).

Removes a member from the collection.

Parameters:

longIndex Required VT_I4 or Long. The index number of a member of the collection.

Must be a numeric expression (a number from 1 to the value of the collection's Count property).

Document Events

Properties

(None)

Methods

Change
BeginPrinting
ProgressPrinting
EndPrinting
PausedPrinting

Object Methods

► Document.BeginPrinting

VTS_NONE **BeginPrinting** (*strDocName*).

Informs the user when the printing process begins for document *strDocName*.

Parameters:

strDocName VT_BSTR or String Name of the document starting to print.

► Document.Change

VTS_NONE **Change** ().

Informs the user of a change in the document.

User is invited to save or save as the current document.

► Document.ProgressPrinting

VTS_NONE **ProgressPrinting** (*LabelPercent*, *refCancel*).

Informs the user of the printing progress for the current document.

Parameters:

LabelPercent VT_I2 or Integer Percent value of the printing of the current label.

refCancel VT_I2 or Integer. User must assign this parameter to 1 to abort process.

► **Document.EndPrinting**

VTS_NONE **EndPrinting** (*Reason*).

Infoms the user of the end of the printing process for current document with anotification code.

Reason VT_I2 or Integer or enumEndPrinting type. Code for the reason for the end of the process.

The value can be one of the following:

lppxEndOfJob	= 1
lppxCancelled	= 2
lppxSystemFailure	= 3

► **Document.PausedPrinting**

VTS_NONE **PausedPrinting** (*Reason, refCancel*).

Infoms the user of a problem during printing process for the current document.

Reason VT_I2 or Integer or enumPausedReasonPrinting type. Reason for the end of the process.

The value can be one of the following:

lppxGenericError	= 0
lppxNoPaper	= 2
lppxNoRibbon	= 3
lppxPortNotAvailable	= 4
lppxPrinterNotReady	= 5
lppxCommunicationError	= 6
lppxHeadLifted	= 7
lppxPrinterMemory	= 8
lppxPrinterSettings	= 9
lppxSetupCommunication	= 10

Parameters:

refCancel VT_I2 or Integer. User must assign this parameter to 1 to abort process.

Application Events

Properties

(None)

Methods

Close
Quit
DocumentClosed

Object Methods

► ApplicationEvent. Close

VTS_NONE **Close** ().

Informs that a user has closed the application.

► ApplicationEvent. Quit

VTS_NONE **Quit** ().

Informs users when someone quits the application with **Application.Quit** method or if the user has manually closed the application.

► ApplicationEvent. DocumentClosed

VTS_NONE **DocumentClosed** (*strDocTitle*).

Informs that the document with title *strDocTitle* has been closed.

Parameters:

strDocTitle VT_BSTR or String Title of the document closed.

Appendix



Information on Visual C++ Data Type

This section provides information on the data type used with Visual C++.

For more information, refer to the Microsoft Visual C++ 6.0 documentation.

VARENUM usage key,

[V] - may appear in a VARIANT

[T] - may appear in a TYPEDESC

[P] - may appear in an OLE property set

[S] - may appear in a Safe Array

Item	[V]	[T]	[P]	[S]	Value
VT_EMPTY	*		*		nothing
VT_NULL	*		*		SQL style Null
VT_I2	*	*	*	*	2 byte signed int
VT_I4	*	*	*	*	4 byte signed int
VT_R4	*	*	*	*	4 byte real
VT_R8	*	*	*	*	8 byte real
VT_CY	*	*	*	*	currency
VT_DATE	*	*	*	*	date
VT_BSTR	*	*	*	*	OLE Automation string
VT_DISPATCH	*	*	*	*	IDispatch
VT_ERROR	*	*	*	*	SCODE

Item	[V]	[T]	[P]	[S]	Value
VT_BOOL	*	*	*	*	True=-1, False=0
VT_VARIANT	*	*	*	*	VARIANT
VT_UNKNOWN	*	*		*	IUnknown
VT_DECIMAL	*	*		*	16 byte fixed point
VT_RECORD	*		*	*	user defined type
VT_I1	*	*	*	*	signed char
VT_UI1	*	*	*	*	unsigned char
VT_UI2	*	*	*	*	unsigned short
VT_UI4	*	*	*	*	unsigned short
VT_I8		*	*		signed 64-bit int
VT_UI8		*	*		unsigned 64-bit int
VT_INT	*	*	*	*	signed machine int
VT_UINT	*	*		*	unsigned machine int
VT_VOID		*			C style void
VT_HRESULT		*			Standard return type
VT_PTR		*			pointer type
VT_SAFEARRAY		*			(use VT_ARRAY in VARIANT)
VT_CARRAY		*			C style array
VT_USERDEFINED		*			user defined type
VT_LPSTR		*	*		null terminated string
VT_LPWSTR		*	*		wide null terminated string
VT_FILETIME			*		FILETIME
VT_BLOB			*		Length prefixed bytes
VT_STREAM			*		Name of the stream follows
VT_STORAGE			*		Name of the storage follows
VT_STREAMED_OBJECT			*		Stream contains an object
VT_STORED_OBJECT			*		Storage contains an object

Item	[V]	[T]	[P]	[S]	Value
VT_BLOB_OBJECT			*		Blob contains an object
VT_CF			*		Clipboard format
VT_CLSID			*		A Class ID
VT_VECTOR			*		simple counted array
VT_ARRAY	*				SAFEARRAY*
VT_BYREF	*				void* for local use
VT_BSTR_BLOB					Reserved for system use

Note

VT: data type of variable or function parameters.
VTS: data type of the function return value

Index



A

ActiveX , Chapter 1 - 1
ActiveX object, Chapter 1 - 2
Application, Chapter 1 - 2
Application , Chapter 1 - 4
Application Events, Chapter 2 - 151
Application Object, Chapter 1 - 8, Chapter 2 - 17

B

Barcode Object , Chapter 2 - 86
Barcodes Collection, Chapter 2 - 84

C

Code2D Object, Chapter 2 - 93
Collection , Chapter 1 - 2
Collection methods, Chapter 1 - 10
Collection Object, Chapter 1 - 9
COM, Chapter 1 - 2
Compatibility with the previous version, Chapter 1 - 14
Connecting a WithEvents variable to an object, Chapter 1 - 13
Count Property, Chapter 1 - 10
Counter Object, Chapter 2 - 135
Counters Collection, Chapter 2 - 133
Create Object function, Chapter 1 - 4

D

Data Type, Chapter 1 - 8
Database Object , Chapter 2 - 54
DatabaseVariables Collection, Chapter 2 - 143
Date Object, Chapter 2 - 131
Dates Collection, Chapter 2 - 129
Dialog Object, Chapter 2 - 36
Dialogs Collection, Chapter 2 - 34
DocObject Object , Chapter 2 - 76
DocObjects Collection , Chapter 2 - 72
Document , Chapter 1 - 2, Chapter 1 - 4
Document Events, Chapter 2 - 149
Document Object, Chapter 1 - 9, Chapter 2 - 46
DocumentProperties Collection , Chapter 2 - 68
DocumentProperty Object, Chapter 2 - 70
Documents Collection , Chapter 2 - 43
DoEvent, Chapter 1 - 13

E

Early bind, Chapter 1 - 4
EnableEvents property, Chapter 1 - 11
Event management, Chapter 1 - 11
event source, Chapter 1 - 11
Events, Chapter 1 - 2

F

Format Object , Chapter 2 - 63
 Formula Object, Chapter 2 - 127
 Formulas Collection, Chapter 2 - 125
 FormVariables Collection, Chapter 2 - 145
 Free Object, Chapter 2 - 141
 FreeVariables Collection, Chapter 2 - 139

G

GetObject function, Chapter 1 - 5

H

Handling an Object's Events, Chapter 1 - 11
 Hierarchy diagram, Chapter 2 - 15

I

Image Object, Chapter 2 - 82
 Images Collection , Chapter 2 - 80
 Information on Visual C++ Data Type, Chapter 3 - 152

L

LabelManager2, Chapter 1 - 3
 LabelManager2.Application, Chapter 1 - 13
 Late bind, Chapter 1 - 4
 Lppx2.tlb, Chapter 1 - 3

M

Mechanisms, Chapter 1 - 4
 Methods, Chapter 1 - 2

N

New function, Chapter 1 - 6

O

OLEObject Object, Chapter 2 - 106
 OLEObjects Collection, Chapter 2 - 104
 Options Object , Chapter 2 - 29

P

Printer Object, Chapter 2 - 58
 PrinterSystem Object, Chapter 2 - 26
 Properties, Chapter 1 - 2

Q

Quit method, Chapter 1 - 4, Chapter 1 - 8

R

RecentFile Object, Chapter 2 - 41
 RecentFiles Collection, Chapter 2 - 38

S

Server Activation, Chapter 1 - 4
 Server Deactivation, Chapter 1 - 7
 Shape Object, Chapter 2 - 113
 Shapes Collection, Chapter 2 - 108
 Strings Collection, Chapter 2 - 147

T

TableLookup Object, Chapter 2 - 122
 TableLookups Collection, Chapter 2 - 120
 Text Object, Chapter 2 - 97
 Texts Collection, Chapter 2 - 95
 TextSelection Object, Chapter 2 - 102
 TK Labeling ActiveX 6.0, Chapter 1 - 3
 Type library, Chapter 1 - 3

V

- Variable Object, Chapter 2 - 118
- Variables Collection, Chapter 2 - 114

W

- WithEvents keyword, Chapter 1 - 11
- WithEvents variables, Chapter 1 - 11

www.zebra.com

Zebra Technologies Corporation

International Headquarters

333 Corporate Woods Parkway

Vernon Hills, IL 60061 U.S.A.

Phone: +1 847.634.6700

Fax: +1 847.913.8766

Zebra Technologies Europe Limited

European Headquarters

Zebra House

The Valley Centre, Gordon Road

High Wycombe

Buckinghamshire HP13 6EQ, UK

Phone: +44.1494.472872

Fax: +44.1494.450103

Zebra Technologies Singapore

Regional Sales Office-Asia/Pacific

1 Sims Lane, #06-11

Singapore 387355

Phone: +65.842.0322

Fax: +65.842.0366

Other Locations

United States: California, Florida,

Rhode Island, Wisconsin

Europe: France, Germany, Italy

Asia: Hong Kong, Japan, Seoul



Zebra Bar Code Products