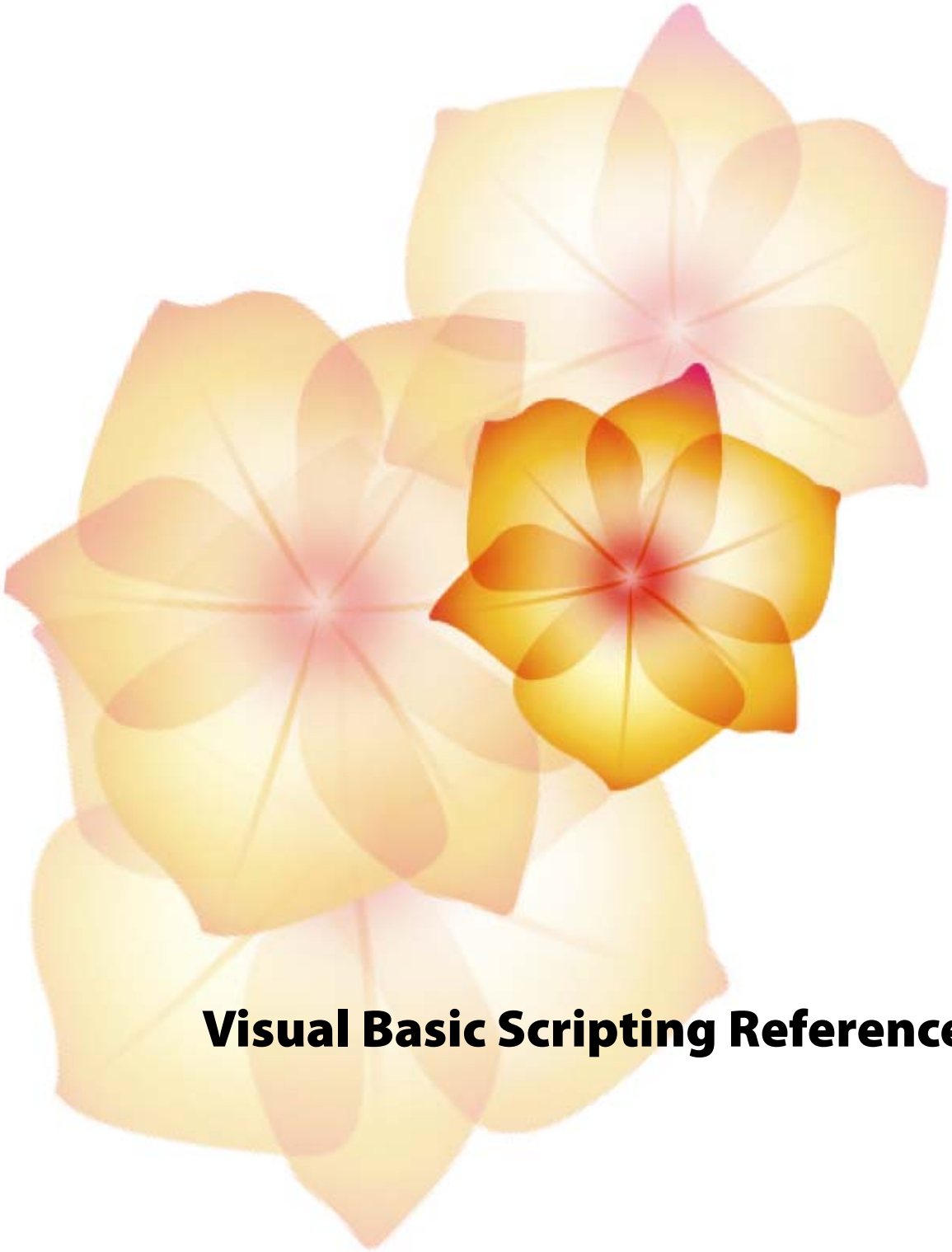


Adobe® Illustrator® CS2

Visual Basic Scripting Reference

Adobe

Adobe® Illustrator® CS2 Visual Basic Scripting Reference for Windows®.

Adobe Systems Incorporated, 345 Park Avenue, San Jose, California 95110, USA.

Contents

1 Introduction

About this Manual

This manual provides an introduction to scripting Adobe® Illustrator® CS2 on Windows® with Visual Basic or VBScript. This document contains the following chapters:

- Introduction —An introduction to scripting.
- Scripting Basics — The basics of scripting with Visual Basic in Windows. If you are new to scripting, be sure to read this chapter.
- Scripting Illustrator — A brief introduction to the specifics of scripting Illustrator. Concepts and approaches specific to the application are covered here, such as measurement units, matrices, and color models.
- Visual Basic Object Reference — Details and examples are provided for every class in Illustrator's Visual Basic type library.

For further information and developments on this and other Adobe products, see the Adobe Solutions Network website:

`http://partners.adobe.com/asn`

What is Scripting?

A script is a series of commands that tells Illustrator to perform a series of actions. These actions can be simple, and affect only a single, selected object in the current document; or complex, and affect all of the objects in all of your Illustrator documents. The actions might involve only Illustrator, or they might involve other applications, such as word processors, spreadsheets, and database management programs. Many of the tasks you can perform with Illustrator's tools, menus, palettes, and dialog boxes can be performed by a script (a notable exception is third-party plug-ins, which cannot be scripted at this time).

We naturally think of scripting as a way to automate repetitive tasks, but it can also be a creative tool. You can use scripts for creative tasks that would be too difficult or time consuming to do manually. For example, you could write a script to systematically create a series of objects, modifying the new objects' position, stroke, and fill properties along the way. You could also write a script that accessed Illustrator's built-in transformation matrix functions to stretch, scale and distort a series of objects. Without scripting, you'll likely miss out on the creative potential of such labor-intensive techniques.

Scripting isn't just for computer programmers—it's for everybody. You don't need a degree in computer science or mathematics to write scripts that can automate a wide variety of common tasks. If you can read this text, you can write scripts.

The language you use to write scripts depends on the operating system of the platform you're using: AppleScript for Mac OS®; Visual Basic for Windows; JavaScript for either platform. Each of these languages is described in a separate manual.

Why use scripting?

Graphic design is a field characterized by creativity, but aspects of the actual work of illustration and page layout are anything but creative. When you think about the work that you do, chances are good you'll find

that you spend most of your time doing the same or similar production tasks, over and over again. In fact, you'll probably notice that the time you spend placing and replacing images, correcting errors in text, and preparing files for printing at an imagesetting service provider often reduce the time you have available for doing creative work.

Wouldn't it be great if you had an assistant—one that wouldn't mind doing some or all of the boring, repetitive tasks for you? With that kind of help, you'd have more time to concentrate on the creative aspects of your work.

With a small investment of time, Illustrator scripting can be the assistant you need. You can start with short, simple scripts that save you a few seconds every day, and move on to scripts that work all night while you're sleeping.

Think about your work—is there a repetitive task that's driving you crazy? If so, you've identified a candidate for a script. What are the steps involved in performing the task? What are the conditions in which you need to do the task? Once you understand the process you go through to perform the task, you'll be ready to turn it into a script.

What about actions?

Illustrator actions are different from scripts. An Illustrator action is a series of tasks you have recorded while using the application—menu choices, tool choices, object selection, and other commands. When you "play" an action, Illustrator performs all of the recorded commands.

You record, play, edit, and delete actions using Illustrator's built-in Actions palette. The "Automating Tasks" chapter in the Adobe Illustrator User Guide covers actions in detail.

With the introduction of scripting for Illustrator, it is important to avoid any confusion about the difference between actions and scripting. Actions and scripts are both ways of automating repetitive tasks, but they work very differently. The following points summarize the key differences.

- Actions use a program's user interface to do their work. As an action runs, menu choices are executed, objects are selected, and recorded paths are created. Scripts do not use a program's user interface to perform tasks, and can execute faster than actions.
- Actions have very limited facilities for getting and responding to information. You cannot add conditional logic to an action. Therefore, actions cannot make decisions based on the current situation. Scripts are capable of getting information and making decisions and calculations based on the information they receive from Illustrator.
- A script can execute an action, but actions cannot execute scripts.

Script Support in Adobe Illustrator CS2

The Scripts menu supports AppleScript and JavaScript scripts for Mac OS, and VBScript, JavaScript, and Visual Basic scripts for Windows.

For a file to be recognized by Adobe Illustrator CS2 as a valid script file it must have the correct file name extension:

Script Type	File Type	Extension	Platform
AppleScript	compiled script OSAS file	.scpt (none)	Mac OS
JavaScript ExendScript	text	.js .jsx	Mac OS & Windows
VBScript	text	.vbs	Windows
Visual Basic	executable	.exe	Windows

Executing scripts

The Adobe Illustrator CS2 interface includes a Scripts menu (**File > Scripts**) which provides quick and easy access to your scripts. Scripts can be listed directly as menu items, that run when you select them, or you can navigate to and run any script in your file system.

If Illustrator CS2 encounters an error during script execution, it displays the error message returned by the script in an error dialog.

Note: It is not possible to execute scripts that contain the `do script` command from the **Scripts** menu. Attempting to do so causes an error.

Installing scripts

To install a script in the Scripts menu, place it in the Scripts folder (**Illustrator CS2 > Presets > Scripts**). The names of the scripts in the Scripts folder, less any file name extension, will be displayed in the Scripts menu. Any number of scripts may be installed in the Scripts menu.

Scripts added to the Scripts folder while Illustrator is running will not appear in the Scripts menu until the next time you launch Illustrator.

If you have a large collection of scripts you wish to use, you may use sub-folders in the Scripts folder to help organize the scripts in the Scripts menu. Each subfolder will be displayed as a separate submenu containing the scripts in that subfolder.

Executing other scripts

The **Other Scripts** item at the end of the **Scripts** menu (**File > Scripts > Other Scripts**) allows you to execute scripts which are not installed in the Scripts folder. Selecting **Other Scripts** displays a file browser dialog which allows you to select a script file for execution. Only files which are of one of the supported file types are displayed in the browse dialog. When you select a script file, it is executed the same way as an installed script.

System requirements for Windows and Visual Basic

Make sure the scripting plug-in is installed on your system before attempting to script Illustrator.

To use Illustrator scripting in Windows, you must have Windows 98®, Windows NT 4.0, Windows 2000, or Windows XP.® You will also need the Microsoft® Visual Basic development environment or one of the applications that contain a Visual Basic editor. Many applications that support the Visual Basic for Applications (VBA) language contain a built-in editor. Applications that contain a built-in editor include: Microsoft Word, Microsoft Excel, and Visio. You can use any Visual Basic editor to create your scripts.

As your scripts become more complex or require a user interface, you will find the need for a complete development environment than one of the built-in editors. The Microsoft Visual Basic development environment comes in a variety of package, all of which provide everything you need to script Illustrator.

In this manual, we use the Microsoft Visual Basic development environment's editor.

Changes to Visual Basic and VBA support in Adobe Illustrator CS2

The following changes have been made to the Visual Basic and VBA object model in this release:

- Updates to `PDFSaveOptions`, `ExportOptionsSVG`, and `FlashExportOptions` to reflect new capabilities in the corresponding dialogs.
- In earlier version, a script used `RasterItems.Add()` to place a raster file format into a document. In this version, use `PlacedItems.Add()` instead. To embed the art in the document, use the `PlacedItem.Embed` function, which converts it to an embedded `RasterItem` or other art items.
- New scripting support for underline and strikethrough font styles.
- New scripting capability for converting raster art into vector art, called *tracing*. The tracing operation reorders the raster art into the source art of a plugin group, and converts it into a group of filled and/or stroked paths that resemble the original image.
    - New methods `PlacedItems.Trace` and `RasterItems.Trace` initiate tracing, creating a new `PluginItem` for the new vector art.
    - `PluginItem.IsTracing` is `true` for the new item, and `PluginItem.Tracing` contains a reference to a new `TracingObject` object.
    - The `TracingObject.TracingOption` property references a `TracingOptions` object that collects the parameters used for the tracing operation. You can save tracing options to a preset file, and load previously saved tracing presets, using the `TracingOptions.LoadFromPreset` and `TracingOptions.StoreToPreset` methods.

2 Scripting Basics

When you use Illustrator, you work with documents and their contents. You create documents, layers, colors, and design elements. You probably think of these things as objects, that you can look at and move around, and they are in fact represented by objects in the Illustrator object model. The Illustrator object model contains documents, layers, colors, and page items—objects that can appear in an Illustrator document.

Automating Illustrator with scripting uses the same object-oriented way of thinking. Each type of object has its own special properties, and the scripting language has ways to look at and change these properties.

This chapter provides a brief introduction to the basic concepts and syntax of Visual Basic and VBA on Windows. A bibliography at the end contains references to more complete language guides.

- For more information on Illustrator's object model and specific Illustrator concepts, see Scripting Illustrator.
- For detailed information on the Visual Basic Illustrator objects and methods, see Visual Basic Object Reference.

Object Model Concepts

In object-oriented programming, *objects* belong to *classes* and have *properties* that describe them. You manipulate the objects and their properties using *methods* in Visual Basic. (In other languages, these can be called *commands* or *functions*.) What do these terms mean in this context?

Here's a way to think about objects and their properties. Imagine that you live in a house that responds to your commands (you can think of this house as technologically advanced, or magical, or both). The house is an object, and its properties might include the number of rooms, the color of the exterior paint, or the date of its construction.

Your house can also contain other objects. Similarly, the objects within the house can also contain smaller objects. Each room, for example, is an object in the house, while each window, door, or appliance is an object inside a room.

Each object can respond to various commands according to its capabilities. Windows and doors, for example, can open or close—but the floor and ceiling cannot. Using scripting, you can talk to each object directly, or you can talk to them as part of the container. You have to be very specific, though—you can't tell your house to open a window without telling it which window you want to open. So windows, like all other objects, need names or at least a numbering system so you can refer to them specifically. For example, you might say "Tell the house to open the north window of the living room."

Objects also have properties that describe specific details about them, like color and size. Imagine that the properties of objects in your house can be changed. You might say, "Door, paint yourself blue." Because your door can respond to the command "paint," you'll soon have a door of a different color.

Now let's apply this object model idea to Illustrator. The Illustrator application is the house, its documents are the rooms, and the objects in your documents are the windows and doors. You can tell Illustrator documents to add and remove objects. You can ask objects to get or change their properties.

Object classes

Objects with the same properties and behaviors are grouped into "classes." In the house example, windows and doors belong to their own classes, since they have unique properties, like number of panes for windows or the door style for doors. In Illustrator, every type of graphic object—paths, text, meshes, etc.—belongs to its own class, each with its own set of properties and behaviors. Properties such as `VisibleBounds`, `Width`, and `Height`, for example, are common to all `PageItems`.

Object inheritance

Since each container can contain smaller objects, there can be hierarchy of classes: the house contains a room, which contains a door. Each bigger class is called a *superclass*, and each smaller one a *subclass*. House and room are both superclasses of door. Superclasses and subclasses are also called parent classes and child classes.

Object classes may *inherit*, or share, the properties of a parent, or superclass. So in our house example, windows and doors are both openings in a house, so they can be subclasses of an openings class, which contains properties that are common to all types of openings, such as open-state. In Illustrator, `PathItems`, for example, inherit geometric properties like `Width` and `Height` from the `PageItem` class.

Classes often have additional properties that are not shared with their superclass. In our house, both a window and door inherit an open-state property from the opening class, but a window has number-of-panes property which the opening class does not. In Illustrator, `PathItems`, for example, have the property `StrokeColor` which is not inherited from the `PageItem` class.

Object elements or collections

Object *collections*, in Visual Basic, are objects contained within other objects. For example, rooms are elements (or collections) of our house, contained within the house object. In Illustrator, documents are elements of the application object, and page items are elements of a document object.

Object references

The objects in your documents are arranged in a hierarchy like the house object—page items are in layers, which are inside a document, which is inside Illustrator. When you send a message to an Illustrator object, you need to make sure you send it to the right object. To do this, you identify objects by their position in the hierarchy. You might, for example, write the following statement.

```
Documents(1).Layers(1).PageItems(1)
```

When you identify an object in this fashion, you're creating an *object reference*, to give the script a way of finding the object you want to work with.

Scripting Concepts

This section discusses various basic programming and scripting features and how they are used in the Visual Basic language.

Comments

Comments are a way to add descriptive text to a script. Comments come in handy when you want to document the operation of a script (for yourself or for someone else). The use of comments is the most important technique for good scripting. Comments are where you should leave important notes about the specific operation of a script that might provide valuable help when the script is modified at a later date. The time you save later trying to figure out what the script does may be your own. Comments are ignored by the scripting system as the script executes and cause no run-time speed penalty.

In Visual Basic, enter ' (a single straight quote) to the left of the comment.

```
' This is a comment
```

Long script lines

In some cases, individual script lines are too long to print on a single line in this guide.

Visual Basic uses the underscore character (_) to show that the line continues to the next line. This continuation character denotes a “soft return” in the script. You can enter this character in the editor by pressing Shift-dash at the end of the line you wish to continue.

Value types

Values are the data your scripts use to do their work. Most of the time, the values used in your scripts will be numbers or text. The following table shows the Visual Basic value types:

Value type	What it is	Example:
`Boolean`	Logical true or false	`true`
`Long`	Whole numbers (no decimal points). Longs can be positive or negative.	`14`
`Double`	A number which may contain a decimal point.	`13.9972`
`String`	A series of text characters. Strings appear inside (straight) quotation marks.	`“I am a string”`
`Array`	A list of values. Arrays contain a single value type unless the type is defined as Variant.	`Array(10.0, 20.0, 30.0, 40.0)`
`Object reference`	A specific reference to an object.	`Application.Documents(1)`
`User-defined`	A collection of elements referenced by a key and stored as a key-value pair.	`Var.name = “you”` `Var.index = 1`

Variables

Variables are containers for data. A variable might contain a number, a string, a list (or array), or an object reference. Variables have names, and you refer to a variable by its name. To put data into a variable, we assign the data to the variable. The file name of the current Illustrator document or the current date are both examples of data that can be assigned to a variable.

Why not simply enter the value directly in the script rather than using a variable? When you use a value directly the flexibility of script is reduced. By using variables the scripts you write will be reusable in a wider variety of situations. As a script executes, it can assign data to the variables that reflect the state of the current document and selection, for example, and then make decisions based on the content of the variables.

Declaring variables

In Visual Basic, it is considered good form to declare all of your variables with the `Dim` statement before using them. Using the `Dim` statement assigns a value type to the variable, which helps us keep our scripts clear and readable. Memory is also used more efficiently if variables are declared before use. If you start your scripts in Visual Basic with the line `Option Explicit`, you will be required to declare all variables before assigning data to them.

```
Option Explicit
Dim thisNumber As Single
thisNumber = 10
```

Assigning variable values

To assign a value that is not an object reference to a variable in Visual Basic, use Visual Basic's assignment operator, the equal sign:

```
thisNumber = 10
thisString = "Hello, World!"
```

Variables can also be used to store references to objects. In Visual Basic, you must use the `Set` command to assign an *object reference* to a variable. For example, use `Set` to assign a variable as you create a layer:

```
Set thisLayer = Illustrator.Documents(1).Layers.Add
```

This code uses `Set` with a reference to an existing layer:

```
Set thisLayer = Illustrator.Documents(1).Layers(1)
```

Variable naming

Try to use descriptive names for your variables—something like `firstPage` or `corporateLogo`, rather than `x` or `c`. While it will take a little more time to type the longer names, using them will make your scripts much easier to read. The length of a variable's name has no effect on the execution speed of your script, so use descriptive names.

You can also give your variable names a standard prefix so that they'll stand out from the objects, commands, and keywords of your scripting system.

Variable names must be a single word, but you can use internal capitalization (such as `myFirstPage`) or underscore characters (`my_first_page`) to create more readable names. Variable names cannot begin with a number, and they can't contain punctuation or quotation marks.

Script properties

Visual Basic allows you to define properties for your scripts. Script properties are much like variables, but with additional features and requirements specific to each language. The meaning and usage of script properties differs greatly between languages; consult the bibliography for appropriate language references.

Operators

Operators perform calculations (addition, subtraction, multiplication, and division) on variables or values and return a result. For example:

```
docWidth/2
```

This returns a value equal to half of the content of the variable `docWidth`. So if `docWidth` contained the number `20.5`, the value returned would be `10.25`.

You can also use operators to perform comparisons (equal to, not equal to, greater than, or less than). For example:

```
docWidth > docHeight
```

This returns the value `true` if `docWidth` is greater than `docHeight`, or `false`, if it is not.

Some operators differ between languages For example, Visual Basic uses the greater and less than symbols juxtaposed, `<>` as the non-equality symbol.

Use the ampersand (`&`) as the concatenation operator to join two strings.

```
"Pride " & "and Prejudice."
```

This returns the string “Pride and Prejudice.”

Methods

If objects are “nouns” and properties are “adjectives” in our scripting systems, then *methods* are the “verbs”—they’re the parts of the script that make things happen. The type of the object you’re working with determines which methods you can use to manipulate it.

In Visual Basic, use the `Add` method to create new objects, the `Set` statement to assign object references to Visual Basic variables or properties, and the assignment operator (`=`) to retrieve and change object properties.

Conditional statements

If you could speak to Illustrator in the course of a work session, you might say, “If the selected object is a path, then set its stroke width to 12 points.” Conditional statements make decisions—they give your scripts a way to evaluate something (the color of the selected object, or the number of color swatches in the document, or the date) and then act according to the result. Conditional statements generally start with the word `if`.

The following example checks the number of currently open documents. If no documents are open, the script displays a messages in a dialog box.

```
Private Sub Command1_Click()
      Dim documentCount as long
```

```
    Dim appRef As New Illustrator.Application
    documentCount = appRef.Documents.Count
    If documentCount = 0 then
       MsgBox "No Illustrator documents are open!"
    End If
End Sub
```

Control structures

If you could talk to Illustrator, you might say, "Repeat the following procedure twenty times." In scripting terms, this sort of direction is called a "control structure." Control structures provide for repetitive processes, or "loops." The idea of a loop is to repeat some action over and over again, with or without changes each time through the loop, until some condition is met.

Visual Basic has a variety of different control structures to choose from. The simplest form of a loop is one that repeats some series of script operations a set number of times.

```
For counter = 1 to 20
  MsgBox counter
Next
```

A more complicated type of control structure includes conditional logic, so that it loops while or until some condition is true or false.

```
Do While flag = false
  flag = (MsgBox ("Quit?", vbOKCancel)) = vbCancel
loop

Do Until flag = true
  flag = (MsgBox ("Quit?", vbOKCancel)) = vbOK
loop
```

Subroutines

In Visual Basic, *subroutines* are scripting modules you can refer to from within your script. These are sometimes called routines or handlers. Subroutines are ways to re-use parts of scripts. Typically, you send one or more values to a subroutine, and it returns one or more values. A subroutine might, for example, perform conversions from one measurement system to another, or calculate the geometric center point of an object from its geometric bounds.

There's nothing special about the code used in subroutines—they are simply conveniences that save you from having to type the same lines of code over and over again in your script. If you find yourself typing or pasting the same lines of code into several different places in a script, you've identified a good candidate for a subroutine.

This example calculates the geometric center of a selected art item. It assumes you have a single art item selected.

```
Private Sub Command1_Click()
Dim appRef As New Illustrator.Application
  Dim selectedObjects As Variant
  Dim objectBounds As Variant
  Dim objectCenter As Variant

  If appRef.Documents.Count > 0 Then
```

```
    selectedObjects = appRef.Documents(1).Selection
       If TypeName(selectedObjects) = "Variant()" Then
          objectBounds = selectedObjects(0).GeometricBounds
          objectCenter = GetItemCenter(objectBounds)
          MsgBox ("Center x:" & objectCenter(0) & ", y:" & objectCenter(1))
       End If
    End If
End Sub
```

The following lines define the function:

```
Function GetItemCenter(sourceBounds As Variant) As Variant
    Dim left As Single
    Dim top As Single
    Dim right As Single
    Dim bottom As Single
    Dim xCenter As Single
    Dim yCenter As Single

    left = sourceBounds(0)
    top = sourceBounds(1)
    right = sourceBounds(2)
    bottom = sourceBounds(3)
    xCenter = (left + right) / 2
    yCenter = (top + bottom) / 2

    GetItemCenter = Array(xCenter, yCenter)
End Function
```

Testing and Troubleshooting

The scripting environment provides tools for monitoring the progress of your script while it is running—which make it easier for you to track down any problems your script might be encountering or causing.

In Visual Basic, you can stop your script at any point, or step through your script one line at a time. To stop your script at a particular line, select that line in your script and choose **Debug > Toggle Breakpoint**. When you run the script, Visual Basic will stop at the breakpoint you have set. Choose **Debug > Step Into** (or press F8) to execute the next line of your script, or choose **Run > Start** (or press F5) to continue normal execution of the script. You can also observe the values of variables defined in your script using the Watch window—a very valuable tool for debugging your scripts. To view a variable in the Watch window, select the variable and choose **Debug > Quick Watch**. Visual Basic displays the Quick Watch dialog box. Click **Add**. Visual Basic displays the Watch window.

If you have closed the Watch window, you can display it again by choosing **View > Watch** Window.

About error handling

Imagine that you've written a script that formats the current text selection. What should the script do if the current selection turns out not to be text at all, but a path item? You can add *error handling* code to your script to respond to conditions other than those you expect it to encounter.

If you have complete control over the situations in which your script will run, there's no need for you to worry about error handling. If not, however, you'll have to add some error handling capabilities to your script.

The following example shows how to how you can stop a script from executing when a specific file cannot be found. This example stores a reference to the fifth path item of a document in a variable. If such an object does not exist in the current document, it displays a message.

```
Private Sub Command1_Click()
     Dim appRef As New Illustrator.Application
     Dim docRef As Illustrator.Document
     Dim aiObject As Illustrator.PathItem
     Dim numObjects As Single
     Dim errorMessage As String

     Set docRef = appRef.ActiveDocument
     numObjects = docRef.PathItems.Count
     On Error GoTo DisplayError
     Set aiObject = docRef.PathItems(5)
Exit Sub

DisplayError:
     errorMessage = "Couldn't locate 5th path object - Only "
     errorMessage = errorMessage & numObjects & " objects."
     MsgBox errorMessage
End Sub
```

Visual Basic Resources

For further information and instruction in using Visual Basic and the VBA scripting language, see these documents and resources:

- "Learn to Program with Visual Basic 6," 1st ed., John Smiley, Active Path, 1998. ISBN 1-902-74500-0.
- "Microsoft Visual Basic 6.0 Professional," 1st ed., Michael Halvorson, Microsoft Press, 1998. ISBN 1-572-31809-0.
- "VB & VBA in a Nutshell," 1st ed., Paul Lomax, O'Reilly, 1998. ISBN 1-56592-358-8.
- Microsoft Developers Network (MSDN) scripting website:

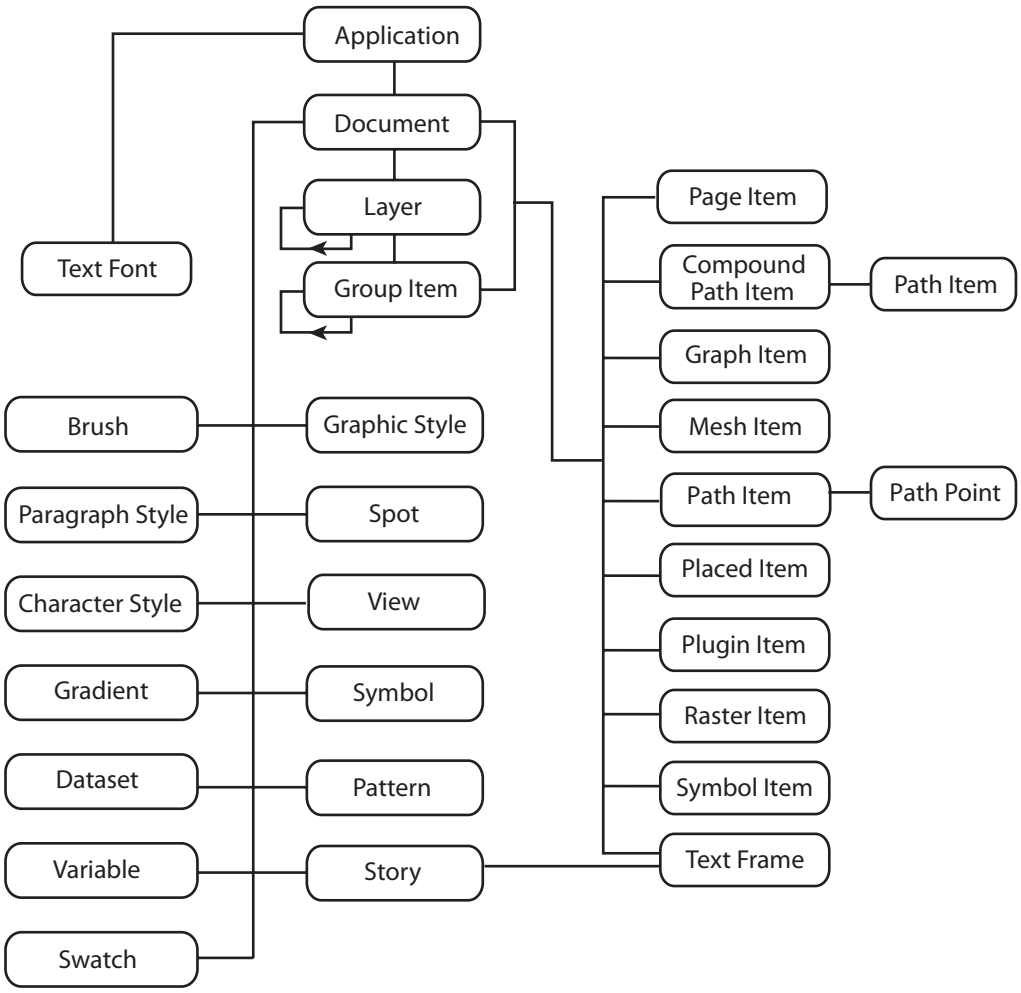
  msdn.microsoft.com/scripting

3 Scripting Illustrator

This chapter discusses the object model and application-specific concepts that you use to script Adobe Illustrator using Visual Basic and VBScript.

The Illustrator Object Model

A good understanding of the Illustrator object model will improve your scripting abilities. The following figure shows the containment hierarchy of the object model, starting with the application object.



Adobe Illustrator Scripting Object Model

Note that the `Layer`, `GroupItem`, and all text classes can contain additional objects of the same class which can in turn contain additional nested objects. For the text object model, see Working with text art.

Looking at Illustrator objects and commands

While the objects and commands available in Illustrator are all documented in this guide, you can also view them from inside your scripting system.

➤ **To view the Illustrator type library:**

1. In any Visual Basic project, choose **Project > References**. Visual Basic displays the References dialog box. If you are using a built-in editor in a VBA application, choose **Tools > References**.
2. Turn on the "Adobe Illustrator CS2 Type Library" option from the list of available references and click **OK**. If the library does not appear in the list of available references, reinstall Illustrator with your scripting plug-ins.
3. Choose **View > Object Browser**. Visual Basic displays the Object Browser window.
4. Choose "Illustrator" from the list of open libraries shown in the top-left pull-down menu. Visual Basic displays the classes and the members of those classes in the Object Browser window.
5. Click an object class or class member. Visual Basic displays more information about the object in the frame at the bottom of the Object Browser window.

Your First Illustrator Script

The traditional first project in any programming language is to display the message "Hello World!" In this example, we'll create a new Illustrator document, then add a text frame item containing this message.

➤ **To create a Visual Basic script follow these steps:**

1. Start Visual Basic and create a new project. Add the "Adobe Illustrator CS2 Type Library" reference to the project, as shown earlier. If you are using a built-in editor in a VBA application, skip to step 4.
2. Add a form to the project.
3. Create a new button on the form. Double-click the button to open the Code window.
4. Enter the following code. The lines which begin with a *single quote* character (') are comments, and will be ignored by the scripting system. They're included to describe the operation of the script. As you look through the script, you'll see how we create, then address each object in turn.

```
Private Sub Command1_Click()
   Dim appRef As New Illustrator.Application
   Dim documentRef As Illustrator.Document
   Dim sampleText As Illustrator.TextFrame
'Create a new document and assign it to a variable
   Set documentRef = appRef.Documents.Add
'Create a new text frame item and assign it to a variable
   Set sampleText = documentRef.TextFrames.Add
'Set the contents and position of the TextFrame
   sampleText.Position = Array(200, 200)
   sampleText.Contents = "Hello World!"
End Sub
```

5. Save the form.
6. Start Illustrator.

7. Return to Visual Basic and run the program. If you created a form, click the button you created earlier.

8. Run the script.

   Illustrator creates a new document, adds a text frame item at the specified position, and sets the text to "Hello World!".

VBScript

You don't need to use Visual Basic to run scripts on Windows. Another way to script Illustrator is to use a VBA editor (such as the one that is included in Microsoft Word) or to use Windows Scripting Host.

Windows Scripting Host is part of Windows2000. If you don't have Windows2000, you can download Windows Scripting Host from: http://msdn.microsoft.com/scripting/

Both VBA and Windows Scripting Host use VBScript as their scripting language. The syntax for VBScript is very similar to the Visual Basic syntax. The three main differences relating to the scripts shown in this guide are:

- VBScript is not as strongly typed as Visual Basic.
  - In Visual Basic you say:

    ```
    Dim aRef as Illustrator.PathItem
    ```

  - For VBScript simply omit the "as X" part:

    ```
    Dim aRef
    ```

- VBScript does not support the `as New Illustrator.Application` form.
  - In Visual Basic you retrieve the Application object as:

    ```
    Dim appRef as New Illustrator.Application
    ```

  - In VBScript you write the following to retrieve the `Application` object:

    ```
    Dim appRef
    Set appRef = CreateObject("Illustrator.Application.3")
    ```

- VBScript does not support enumerations; you must use integer values instead. (See the Enumerations reference section in Visual Basic Object Reference to find the values that correspond to the various enumerations.) Here's an example of how to close the frontmost document without saving.
  - In Visual Basic:

    ```
    Dim appRef As New Illustrator.Application
    appRef.ActiveDocument.Close (aiDoNotSaveChanges)
    ```

  - In VBScript:

    ```
    Dim appRef
    Set appRef = CreateObject("Illustrator.Application.3")
    appRef.ActiveDocument.Close ( 2 )
    ```

Here is an example of Hello World! for VBScript:

```
Dim appRef
Dim documentRef
Dim SampleText
   'Create a new document and assign it to a variable
Set appRef = CreateObject("Illustrator.Application.3")
Set documentRef = appRef.Documents.Add
```

```
   'Create a new text frame item and assign it to a variable
Set SampleText = documentRef.TextFrames.Add
   'Set the contents and position of the TextFrame item
SampleText.Position = Array(200, 200)
SampleText.Contents = "Hello World!"
```

To run this script create a text file and copy the script into it. Save the file with a `.vbs` extension. If you have Windows Scripting Host installed, you can double-click on the file to execute the script.

Another way to execute the script is to choose **File > Scripts > Browse** from the **Scripts** menu in Illustrator and select the file.

Adding features to "Hello World"

Next, let's create a new script that makes changes to the Illustrator document you created with your first script. Don't worry if you've closed the Illustrator document without saving it—just run your script to create a new one.

Our second script will demonstrate how to:

- Get the active document.
- Get the width of the active document.
- Resize the text frame item to match the document's width.

➤ **To create the enhanced script follow these steps:**

1. Open the project you created for the "Hello World" script, if it's not already open.
2. Add a new button to the form.
3. Double-click the button to display the Code window, then enter the following code.

```
Private Sub Command1_Click()
   Dim appRef As New Illustrator.Application
   Dim documentRef As Illustrator.Document
   Dim sampleText As Illustrator.TextFrame
   Dim documentWidth As Single

   ' Get the active document
   Set documentRef = appRef.ActiveDocument
   documentWidth = documentRef.Width
   Set sampleText = documentRef.TextFrames(1)
   ' Resize the TextFrame item to match the document width
   sampleText.Width = documentWidth
   sampleText.Left = 0
End Sub
```

4. Save the form.
5. Open the original document you created using the "Hello World" script, then return to Visual Basic and run the script.
6. Click the button you created in Step 2.

Object References

In Visual Basic, you must use the `Set` command to assign an object reference to a variable, rather than the simple assignment operator (=). For example, the following lines use `Set` to assign a newly created and an existing layer reference to variables:

```
Set thisLayer = Illustrator.Documents(1).Layers.Add()
Set thisLayer = Illustrator.Documents(1).Layers(1)
```

For some types of objects, the `Name` property is writable. The collections are sorted alphabetically by name, so if a script modifies the name of such an object, references to that object by the old name or index can become invalid. These object types include the following:

```
Brush
Gradient
GraphicStyle
Pattern
Swatch
Symbol
Variable
```

Object references in Visual Basic are dynamic and remain valid until disposed. For example, the following creates a star and rectangle and uses object references to select them:

```
Private Sub Command1_Click()
   'Make 2 new objects and select both
   Dim appRef As New Illustrator.Application
   Dim pathItemsRef As Illustrator.PathItems
   Dim rectPath As Illustrator.PathItem
   Dim starPath As Illustrator.PathItem
   Set pathItemsRef = appRef.ActiveDocument.ActiveLayer.PathItems
   Set rectPath = pathItemsRef.Rectangle(50, 70, 100, 200)
   Set starPath = pathItemsRef.Star(40, 70, 200, 110, 5, False)

   Dim pathSelection(1) As Variant
   Set pathSelection(0) = rectPath
   Set pathSelection(1) = starPath
   appRef.Selection = pathSelection
End Sub
```

Creating objects in Visual Basic

As the object model diagram shows, objects are arranged in a hierarchy. To obtain a reference to a specific object you need to navigate the hierarchy. For example, to store a reference to the first `PathItem` in the second layer of the active document in the variable `myPath` you would write:

```
Set myPath = appRef.ActiveDocument.layers(2).PathItems(1)
```

Since most objects in the hierarchy are referenced through the `Application` object, you must have a reference to the Illustrator application before you can start modifying any other objects. There are several ways to obtain a reference to an Illustrator application object.

- Use the `New` command if you have added a reference to the Illustrator type library to the project. For example, the following line creates a new reference to the `Application` object:

  ```
  Dim appRef As New Illustrator.Application
  ```

- `GetObject` also creates a reference to a running instance of Illustrator. If Illustrator is not running an error is returned.

```
Dim appRef As Illustrator.Application
Set appRef = GetObject(, "Illustrator.Application")
```

- `CreateObject` launches Illustrator as an invisible application if it is not already running. Note that if Illustrator was launched as an invisible application you have to manually activate the application to make it visible.

```
Dim appRef As Illustrator.Application
Set appRef = CreateObject("Illustrator.Application")
```

Note: If you have both earlier and later versions of Illustrator installed on the same machine and use the `CreateObject()` or `GetObject()` method to obtain an application reference, use the optional numeric version identifier at the end of the string "`Illustrator.Application`" —when the version identifier is not present, the string refers to the latest installed Illustrator version. To specifically target a version:

- For Illustrator 10, use "`Illustrator.Application.1`"
- For Illustrator CS, use "`Illustrator.Application.2`"
- For Illustrator CS2, use "`Illustrator.Application.3`"

Objects that must be created

There are a number of objects in addition to `Application` that cannot be obtained by using the hierarchy shown in the object model diagram. These objects must created directly using the techniques shown above for the `Application` object. Those objects include:

ExportOptionsFlash	PrintColorManagementOptions	PrintOptions
Ink	PrintColorSeparationOptions	PrintPageMarksOptions
InkInfo	PrintCoordinateOptions	PrintPaperOptions
NoColor	Printer	PrintPostScriptOptions
OpenOptions	PrinterInfo	Screen
PaperInfo	PrintFlattenerOptions	ScreenSpotFunction
PPDFile	PrintFontOptions	
PPDFileInfo	PrintJobOptions	

The following example demonstrates how to create new objects such as `EPSSaveOptions`.

```
' Create a reference to the Illustrator Application
Dim appRef As New Illustrator.Application
' Create an EPS-save option object
Dim myEPSSaveOptions As New Illustrator.EPSSaveOptions
' Set the options according to how you want the save to occur
myEPSSaveOptions.EmbedAllFonts = True
myEPSSaveOptions.Compatibility = aiIllustrator8
myEPSSaveOptions.Preview = aiColorTIFF
' Save the active document
appRef.ActiveDocument.SaveAs "C:\Temp\AI_TestDocument.eps",
myEPSSaveOptions
```

Objects that cannot be created by a script

Following are some objects that cannot be created from a script:

- Graphic styles

- Brushes
- Graphs
- Mesh art
- Plugin art
- Spirals

Object containment: document vs. layer

In Illustrator, all artwork objects are contained in layers, groups, or compound paths that are themselves contained in a document. The index of an object in a layer or group indicates the object's position in the stacking order of the layer or group. This means that `Layers(1).PageItems(1)` is the frontmost object in a document, while `Layers(1).PageItems(2)` lies directly behind in the stacking order.

Note that if you delete all the layers in a document, the document is left with the default empty layer called `Layer 1`.

When you refer to an object in your document, you can reference it directly as part of the document or by its complete containment hierarchy, including layers and any group or compound path if valid. When you refer to objects contained by the document directly, you can access the entire flattened contents of the document, without regard to the containment of objects within layers, groups, or compound paths. All objects, whether or not they are contained in groups or compound paths, are returned as individual objects contained by the document. The following script demonstrates how to reference an object as part of a document.

```
Private Sub Command1_Click()
   'Get reference for first page item of document 1
   Dim appRef As New Illustrator.Application
   Dim documentRef As Illustrator.Document
   Dim pageItemRef As Object

   Set documentRef = appRef.ActiveDocument
   Set pageItemRef = documentRef.PageItems(1)
End Sub
```

In the script below, the variable `pageItemRef` will not necessarily refer to the same object as the above script since this script includes a reference to a layer:

```
Private Sub Command1_Click()
   'Get reference for first page item of document 1
   Dim appRef As New Illustrator.Application
   Dim documentRef As Illustrator.Document
   Dim pageItemRef As Object

   Set documentRef = appRef.ActiveDocument
   Set pageItemRef = documentRef.Layers(1).PageItems(1)
End Sub
```

Syntax Differences Between Sub and Function Methods

Visual Basic supports different types of methods. When scripting Illustrator, your scripts will call both Sub and Function methods. The difference between the two methods is that a Function returns a value while a

Sub, or method call, does not. For example, the `Add` method is a function because it returns a reference to the newly added object.

When calling a Function, Visual Basic expects you to put parentheses around the function's arguments. With a Sub, however, Visual Basic disallows the use of parentheses around the arguments unless you use an alternative syntax. The following lines demonstrate the difference between the syntax for calling a Function, `Add`, and a Sub, `ApplyTo`, in Visual Basic.

```
Set newDoc = appRef.Open("C:\myfile.eps")
appRef.ActiveDocument.GraphicStyles(2).ApplyTo artItem
```

Working with Document Contents

The following sections provide details of how to work with various kinds of document contents:

- "Working with selections"
- "Working with paths"
- "Working with color"
- "Working with symbols and symbol items"
- "Working with text art"

Working with selections

There are instances where you will want to write scripts that act upon the currently selected object or objects. For example, you might want to have a script that applies formatting to selected text, or changes a selected path's shape. To do this, you need to know the number of selected objects and the type of each object.

The following script works with the current selection.

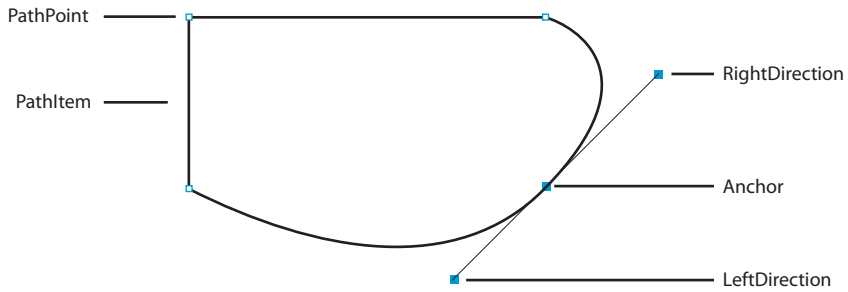
```
'Selection sorter
Private Sub Command1_Click()
Dim appRef As New Illustrator.Application
Dim documentRef As Illustrator.Document
Set documentRef = appRef.ActiveDocument
selectedObjects = documentRef.Selection
If TypeName(selectedObjects) = "String" Then
   'text is selected
Else
'Is anything selected?
   If IsEmpty(selectedObjects) Then GoTo noSelection
      For Each artObject In selectedObjects
         selectedObjectClass = TypeName(artObject)
         Select Case selectedObjectClass
   'Something is selected, let's find out what it is.
            Case "PathItem"
      'Object is a path item
            Case "CompoundPathItem"
      'Object is a compound path
            Case "RasterItem"
      'Object is a raster image
            Case "PlacedItem"
      'Object is a placed image
```

```
            Case "MeshItem"
        'Object is a mesh
            Case "TextFrame"
        'Object is a text frame
            Case "PluginItem"
        'Object is a plugin art item
            Case "PathPoint"
        'Object is a point of a path
            Case "GroupItem"
        'Object is a group of objects
          End Select
        Next
    End If
Exit Sub

noSelection:
    MsgBox "Select an object and try again."
End Sub
```

Working with paths

Path items include all artwork that contain paths, including rectangles, ellipses, polygons, as well as freeform paths. In Illustrator, every path consists of a series of points; see the following figure.



Path items, as well as path points, can be created and manipulated from a script. Every aspect of a path point can be accessed from scripting, including the `AnchorPoint` and both control points, known as the `LeftDirection` and `RightDirection` properties.

For more information on working with paths, Bézier curves, and path points, refer to the *Illustrator Plug-in Software Development Kit Function Reference*. This document is available as part of the Illustrator Software Development Kit (SDK), which can be downloaded from the Adobe Solutions Network (ASN) web site:

http://partners.adobe.com/asn/developer/sdks.html

Working with color

Swatches can be created and manipulated from your scripts. You can also create new patterns, gradients, and spot colors from scripts. Just as in the user interface, percentages (0.0 through 100.0) are used to specify grayscale, individual CMYK values and spot tints. The range 0.0 to 255.0 is used for the individual RGB color values. Special attention should be paid to working with CMYK and RGB color values. Illustrator CS2 supports only a single color model within each document, either CMYK or RGB. When you specify a CMYK color value in a document that uses the RGB color model, Illustrator will convert the values to RGB and return an RGB color, and vice-versa when specifying RGB colors in a CMYK document.

However, there is some data loss during this conversion. Refer to the "Applying Color" chapter in the Adobe Illustrator User Guide for more information on working with color.

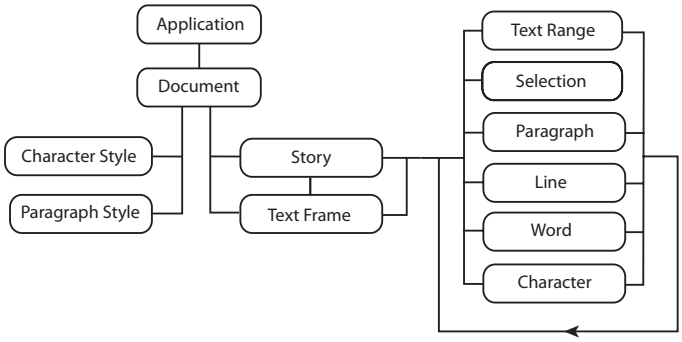
Working with symbols and symbol items

Symbols are art items that are stored in the Symbols palette and applied to documents. You can create, delete and duplicate symbols. When you create symbols, Illustrator adds them to the Symbols Palette for the target document. When you save the document, Illustrator also saves the symbols you created and used in the document.

Symbol items refer to instances of symbols in a document. You can create, delete, and duplicate symbol items. They are "linked" to the symbol definition such that changing the definition of a symbol causes all of the instances of the symbol to change as well. Symbol items are Illustrator art items and therefore can be treated as other art items or page items. In other words, you can rotate, resize, select, lock, hide, and perform other operations on them.

Working with text art

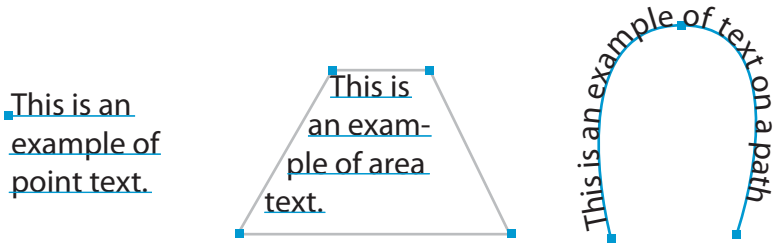
In Illustrator CS2, the object model for text changed significantly. The following figure shows the CS2 object model for text.



Illustrator Object Model for Text

With the new text model, the text in an Illustrator document is contained inside a *story*. A document can have multiple stories, and each story has a *text range*. A story can contain one or more *text frames*. In this case, multiple text frames are linked together to form a single story. There are special sets of text ranges within a text range that have semantic meanings such as paragraphs, lines, words, characters.

There are three types of text frame items in Adobe Illustrator: point text, path text, and area text. The `Kind` property of a text frame item is used to determine the type of the text frame item. While all three kinds of text art have some common characteristics, such as an orientation, each kind of text art also has unique characteristics.



All three kinds of text frames have least one text path associated with them. A `TextPath` is not the same as a path art item, but defines the text frame's position on the artboard and its orientation (horizontal or vertical). Point text is defined completely by the properties of its text frame and associated text path.

For path and area text, text paths are associated with normal path art items. These path art items can be accessed and manipulated to modify the appearance of the associated text frame. If the text frame is path text, it will have a `TextPathOffset` property, which indicates where on the path object the text begins.

All text art items also have at least one line of text depending on the object's geometry. A `TextLine` is all of the characters that fit on a single line in the text frame item. Text art will have multiple text lines if it contains hard line breaks or its characters flow to a new line because they do not fit in the width of the text art. Unlike characters, paragraphs and words, lines can only be created by the Illustrator application.

Refer to the "Using Type" chapter in the Adobe Illustrator User Guide for more information on working with text art.

Content of a text range

You can set the content of a text range by passing in a Unicode string known to that particular scripting language. If you know the Unicode value of a character, you can enter it using a C language style backslash escape sequence. If the string starts with a single or double quote, treat the quote as a delimiter. The exact escape characters it recognizes depends on the scripting language being used.

Character style, character attributes, local character attributes

You can change the display properties of a text range by applying an appropriate character style and/or providing local overrides of character attributes at the text and/or paragraph levels.

The character styles are hierarchical, namely you can derive a new character style from another base (parent) character style, although only one level of inheritance is allowed. The root character style is named 'Normal', with all character attributes defined and set to default values.

If you query for the character attributes of a text range, Illustrator will derive a fully defined/flattened set of character attributes (because at the root level, 'Normal' character style is always fully defined).

This principle also applies the paragraph style, paragraph attributes and local paragraph attributes.

Measurement Units

Illustrator uses points as the unit of measurement for almost all distances, where one inch is equal to 72 points. The one exception is that for values for properties such as kerning, tracking, and the *aki* properties (used for Japanese text composition), em units are used; see Em Space Units.

Even if you change the current document ruler's units of measurement, Illustrator will still use points when communicating with your scripts. Your scripts will need to perform any unit conversions needed to represent your measurements as points. For example, to move the current selection to a position 2 inches to the right of, and 6 inches above, its current position, you'd use the following script for Visual Basic:

```
Private Sub Command1_Click()
   Dim appRef As New Illustrator.Application
   Dim documentRef As Illustrator.Document
   Dim selectedObjects As Variant
   Dim objectRef As Variant
```

```
    Set documentRef = appRef.ActiveDocument
    selectedObjects = documentRef.Selection

    If TypeName(selectedObjects) = "Variant()" Then
       For Each objectRef In selectedObjects
             'There are 72 points per inch, so this moves
             'the object 2" to the right and 6" up
          objectRef.Translate 144, 226
       Next
    End If
End Sub
```

If your script depends on adding, subtracting, multiplying, or dividing specific measurement values for units other than points, the script will need to convert between the units numerically. For example, to use English measurements such as inch values for coordinates or measurement units, your script will need to multiply all inch values by 72 to convert to points, since there are 72 points in an inch. To use metric measurements such as centimeters, you will need to multiply all centimeter values by 28.346, since there are 28.346 points in a centimeter.

Unit conversion to points

This table shows the conversion formulae for various units of measurement:

Unit	Conversion formula
centimeters	28.346 points = 1 centimeter
inches	72 points = 1 inch
millimeters	2.834645 points = 1 millimeter
picas	12 points = 1 pica
Qs	0.709 point = 1 Q (1 Q equals 0.23 millimeter)

Em Space Units

One exception to the rule of points being used for all measurements is the use of *em* units (a traditional typesetting measure) for a few properties such as for kerning and tracking. Values for these properties are measured in thousandths of an em space.

Em units are proportional to the current font size. For example, in a 6-point font, 1 em equals 6 points; in a 10-point font, 1 em equals 10 points. Similarly, a kerning value of 20 em units for a 10-point font would be equivalent to:

```
(20 units x  10 points) / 1000 units/em  =  0.2 points
```

Coordinates

Illustrator uses simple two-dimensional geometry to record the position of objects in a document. The coordinates used in Illustrator are the same as the “traditional” geometric coordinate system you learned about in school. The horizontal component of a coordinate pair (or “point”) is referred to as “x” and the vertical position is denoted by “y”. You can see these coordinates in the Info palette when you select or create an object in Illustrator.

Illustrator scripting uses a special class called `fixed point` to receive and return coordinate data. The fixed point is represented as a variant array of two elements in Visual Basic. The first item is the horizontal or "x" coordinate, while the second item is the vertical or "y" coordinate. The `Position` of objects on a document are described with a fixed point.

Fixed points

In Visual Basic, a fixed point with an x coordinate of 5.0 and a y coordinate of 10.2 is represented as a variant array that looks like this:

```
Array (5.0, 10.2)
```

Note that if you declare an array to hold the values of a point, you should pass 1 as the dimension, since Visual Basic uses index position 0 for the first item in an array.

```
Dim aPoint(1) As Single
aPoint(0) = 5.0
aPoint(1) = 10.2
```

Zero point

The zero point (0, 0) for coordinate numbering in Illustrator is in the lower left corner of the document. On the horizontal axis, coordinates to the right of the ruler's zero point are positive numbers, and on the vertical axis, coordinates above the zero point are positive. The `PageOrigin` of a document defines the lower left corner of the printable region of the document as a fixed point.

Fixed rectangle

To work with rectangular coordinates where there are a pair of x and y values, Illustrator uses the special class called a `FixedRectangle`. This class consists of a variant array with four elements in Visual Basic. The coordinates of a fixed rectangle in order are: left, top, right, bottom.

In Visual Basic, a fixed rectangle with a left-top corner of (5.0, 200.0) and a right-bottom corner of (100.0, 20.0) is represented by a variant array that looks like this:

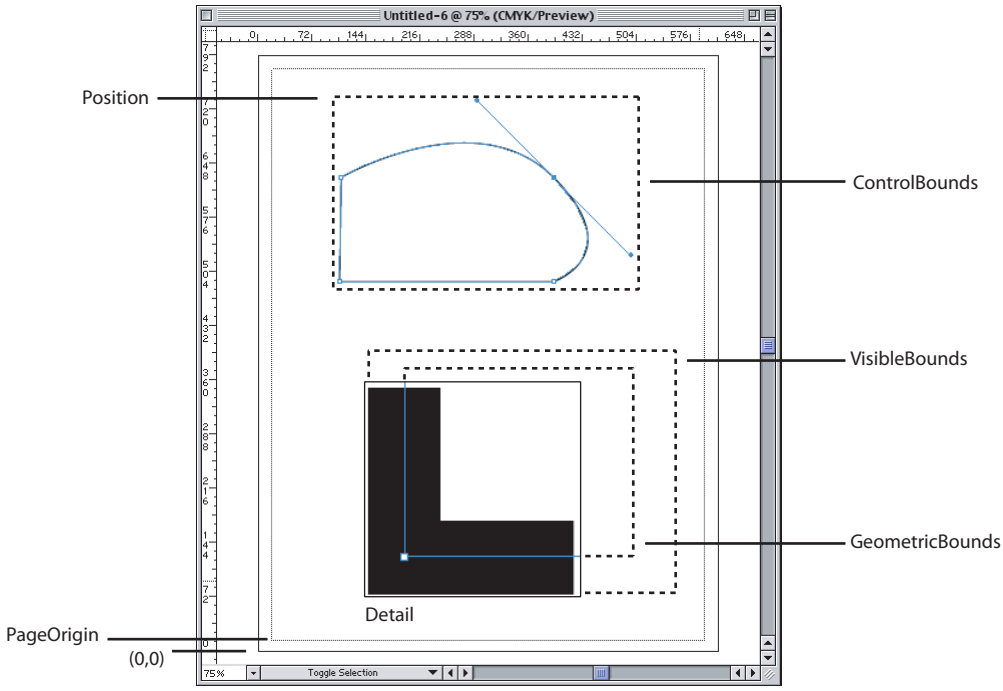
```
Array (5.0, 200.0, 100.0, 20.0)
```

Alternatively, you can define your array and assign values to its elements like this:

```
Dim boundsRect(3) As Single
boundsRect(0) = 5.0
boundsRect(1) = 20.0
boundsRect(2) = 100.0
boundsRect(3) = 200.0
```

Page item positioning and dimensions

Every object, or page item, in a document has a position described by a fixed point and a width and height. The maximum value allowed for the width or height of a page item is 16348 points.



Every page item also has three properties that describe the object's overall extent using fixed rectangles. The `GeometricBounds` of a page item are the rectangular dimensions of the object excluding stroke width. The `VisibleBounds` of a page item are the dimensions of the object including any stroke widths. Finally, the `ControlBounds` define the rectangular dimensions of the object including in- and out-control points.

Printing Illustrator Documents

Prior to Illustrator CS, scripts could use the print command, but they could not control any of the print settings. With the new printing model, you can use the full range of Illustrator print capabilities from a script. In many instances, the print features available via scripting are greater than those available through the Illustrator CS2 print user interface.

Using the print scripting feature, scripters can capture and automate parts of their print workflow, which allows them to focus on other more creative work. Scripting exposes the full capabilities of Illustrator printing, some of which may not be accessible through the normal print user interface.

Illustrator supports at most one print session at any give time because of limitations in the current printing architecture.

When printing, you may provide an options parameter to more fully control the printing process. Illustrator CS2 supports an extensive list of new printing options, all of which have default values. As a scripter, you can override any one of these printing options. If you do not override those printing options or if you override those printing options with illegal values (such as specifying a printer or paper that does not exist), an error is returned.

The following lists the categories of printing options that you can specify. Each one of these categories is optional. Within each category, default values have been provided for all properties.

Print Option	Description
Printer name	Name of the selected printer
PPD file name	File name of the selected PPD file
Print style name	Specifies the printing style
Paper options	Specifies the paper name and custom paper sizes
Print job options	Options which control things such as number of print copies
Color separation options	Specifies the separation mode, ink list, etc.
Page marks options	Controls the printing of page marks
Coordinate options	Specifies the positioning and scaling of artwork on the media
Font options	Controls the fonts used for printing
PostScript® options	Controls parameters such as the PostScript LanguageLevel
Color management	Sets color profiles
Flattener options	Controls the transparency flattening

The print settings are determined through the following precedence order:

- The print style settings, if any are specified, override the default print settings.
- The specific printing options, if any are specified, override the print style settings.

Transformation Matrices

Thanks to the `matrix` class and the many commands that support matrices, you have access to the power of geometric transformation matrices. Transformation matrices are mathematical concepts originating in the field of linear algebra. Geometric manipulations like scaling, rotating, and moving can all be described using transformation matrices.

Matrices are the basis of how Illustrator internally performs a user's request to scale, rotate, or move an object. Using the command set available to create, concatenate, and apply matrices, you can transform objects in documents with programmatic precision and control. By concatenating a series of rotation, translation and scaling matrices together and applying the resulting matrix, you can perform a large series of geometric transformations in record speed. The following examples demonstrate how to combine multiple modifications in a single matrix and then apply the matrix to every object in a document.

Refer to the Illustrator Plug-in Software Development Kit Function Reference for more information on working with transformation matrices.

This example shows how to apply 2 transformations to all art in a document using the matrix command. This is more efficient than performing these transformations one at a time.

```
Private Sub ApplyMatrix_Click()
   Dim appRef As New Illustrator.Application
   Dim moveMatrix As Illustrator.Matrix
   Dim totalMatrix As Illustrator.Matrix
      'Move art half an inch to the right and 1.5 inch up on the page
   Set moveMatrix = appRef.GetTranslationMatrix(72# * 0.5, 72# * 1.5)
```

```
        'Add a rotation to the translation
        'to rotate 10 degrees counter clockwise
    Set totalMatrix = appRef.ConcatenateRotationMatrix(moveMatrix,10)
        'Apply the transformation to all art in the document
    Dim frontDocument As Illustrator.Document
    Dim artItem As Illustrator.PageItem

    Set frontDocument = appRef.ActiveDocument
    For Each artItem In frontDocument.PageItems
        artItem.Transform totalMatrix
    Next
End Sub
```

A matrix object in Illustrator consists of six properties. In Visual Basic, these properties are:

```
MValueA
MValueB
MValueC
MValueD
MValueTX
MValueTY
```

By experimenting with the matrix concatenation commands, you can discover how to construct matrices that can applied to perform movement (also called translation), rotation, scaling, skewing, and other transformations. See the script examples for the matrix commands for working samples.

Working with Variables and Datasets

By creating dynamic objects, you can create data-driven graphics. You can define dynamic objects by using variables. In scripting, the `Variable` class corresponds to these variables. Variables are document-level objects; therefore, you create them in the document object. You can add and delete variables to/from a script by using the `Add` and `Remove`/ `RemoveAll` methods.

Datasets are closely related to variables in that a dataset collects variables and their associated dynamic data into a single object. The `DataSet` class is the object that corresponds to an `AI DataSet`. The `DataSet` collection in the `Document` class provides methods so you can create, update and delete datasets.

Launching and Quitting Illustrator from a Script

Your scripts can control the activation and quitting of the Illustrator application. In Visual Basic, the `Activate` method brings the Illustrator application to the front if it is not already frontmost. The `Quit` method quits the application.

Note that if the clipboard contains data at the time of quitting, Illustrator may show a dialog prompting the user to save the data on the clipboard for other applications. You can avoid this dialog by first clearing the clipboard with the command:

```
Clipboard.Clear
```

User Interaction Levels

An application will usually present a dialog when it needs to provide feedback or request information. This is called user interaction, and is useful and expected when you are directly interacting with the application. On the other hand, when a script is interacting with an application, an unexpected dialog will bring the execution of the script to a halt until the dialog is dismissed. This can be a serious problem in an automation environment where there is typically no one present to deal with dialogs.

The Illustrator CS2 application class contains a user interaction level property. By setting this property a script can control the level of interaction allowed during script execution. All interaction is normally suppressed in an automation environment, and some interaction might be useful where scripts are being used in a more interactive fashion.

There are two possible values for the `UserInteractionLevel` property in Visual Basic:

Property Value	Result
`aiDontDisplayAlerts`	No interaction is allowed
`aiDisplayAlerts`	Interaction is allowed

4 Visual Basic Object Reference

This reference section describes the objects and methods in the Illustrator Visual Basic type library. All of the classes in the type library are presented alphabetically. The chapter concludes with an enumerations reference which lists all of the enumerations in the Illustrator type library.

Each class listing includes the following:

- Properties of the class, including value type, read-only status, and an explanation.
- Methods for the class. Value types needed by the method are shown in bold face. Enumerated values are linked to the Enumerations reference. Required terms are shown in plain face. All items surrounded by brackets [ ] are optional.
- Script examples.

  These example are intended to illustrate concepts, and do not necessarily represent the best or most efficient way to construct a Visual Basic script. They contain little error checking, and assume that the proper context exists for the scripts to execute in (for instance, that there is a document open or items selected).

  Each script contains a single subroutine that can be pasted into any event in a Visual Basic form if you are using the Visual Basic development environment. A standard button click event is used for all examples. If you are using a built-in Visual Basic editor in a VBA application, you can paste the script into a macro routine. In either case, modify the `Sub` statement in the example to work with your situation.

See The Illustrator Object Model for an overview of how the Illustrator object model is structured.

Application

The Adobe Illustrator application object, which contains all other Illustrator objects.

Note: If you have both earlier and later versions of Illustrator installed on the same machine and use the `CreateObject()` or `GetObject()` method to obtain an application reference, use the optional numeric version identifier at the end of the string "`Illustrator.Application`" —when the version identifier is not present, the string refers to the latest installed Illustrator version. To specifically target a version:

- For Illustrator 10, use "`Illustrator.Application.1`"
- For Illustrator CS, use "`Illustrator.Application.2`"
- For Illustrator CS2, use "`Illustrator.Application.3`"

Application properties

Property	Value type	What it is
ActionIsRunning	Boolean	Read-only. If `true`, an action is still running.
ActiveDocument	Document object	The active (frontmost) document in Illustrator.
Application	Application object	Read-only. The Illustrator `Application` object.
BrowserAvailable	Boolean	Read-only. If `true`, a Web browser is available.
Documents	Documents collection object	Read-only. The documents in the application.
FlattenerPresetsList	Variant	Read-only. The list of flattener style names currently available for use.
FreeMemory	Long	Read-only. The amount of unused memory (in bytes) within the Adobe Illustrator partition.
Name	String	Read-only. The application's name (not related to the filename of the application file).
Path	String	Read-only. The file path to the application.
PDFPresetsList	Variant Array of Strings	Read-only. The list of preset PDF-options names available for use.
PPDFileList	Variant Array of Strings	Read-only. The list of PPD files currently available for use.
Preferences	Preference Object	Read-only. The application preferences.
PrinterList	Variant	Read-only. A list of available printers.
PrintPresetsList	Variant	Read-only. The list of preset printing-options names currently available for use.
ScriptingVersion	String	Read-only. The version of the Scripting plug-in.
Selection	Variant Array (of objects)	All of the currently selected objects in the active (frontmost) document; see note for more information.

Property	Value type	What it is
TextFonts	TextFonts collection	Read-only. The installed fonts.
TracingPresetList	Variant Array of Strings	Read-only. The list of preset tracing-options names available for use.
UserInteractionLevel	AiUserInteractionLevel	Whether or not to interact with users by displaying dialogs during the running of a script.
Version	String	Read-only. The version of the Adobe Illustrator application.
Visible	Boolean	Read-only. If true, the application is visible.

Application methods

Method	Returns	What it does
ConcatenateMatrix (matrix as **Matrix**, secondMatrix as **Matrix**)	Matrix object	Concatenates two matrices together.
ConcatenateRotationMatrix (matrix as **Matrix**, angle as **Double**)	Matrix object	Concatenates a rotation translation to a transformation matrix.
ConcatenateScaleMatrix (matrix as **Matrix**, [, scaleX as **Double**] [, scaleY as **Double**])	Matrix object	Concatenates a scale translation to a transformation matrix.
ConcatenateTranslationMatrix (matrix as **Matrix**, [, deltaX as **Double**] [, deltaY as **Double**])	Matrix object	Concatenates a translation to a transformation matrix.
DoJavaScript (action as **String**, from as **String** [, dialogs as **Boolean**])	String	Executes a JavaScript script contained in the action argument, and returns the value of the last executed statement.
DoJavaScriptFile (action as **String**, from as **String** [, dialogs as **Boolean**])	String	Executes a JavaScript script from a file specified by the action argument, and returns the value of the last executed statement.
DoScript (action as **String**, from as **String** [, dialogs as **Boolean**])	Nothing	Plays an action from the Actions palette.
GetIdentityMatrix ()	Matrix object	Returns an identity matrix.
GetRotationMatrix ([angle as **Double**])	Matrix object	Returns a transformation matrix containing a single rotation.

Method	Returns	What it does
GetScaleMatrix ([scaleX as **Double**] [, scaleY as **Double**])	Matrix object	Returns a transformation matrix containing a single scale.
GetTranslationMatrix ([deltaX as **Double**] [, deltaY as **Double**])	Matrix object	Returns a transformation matrix containing a single translation.
InvertMatrix (matrix as **Matrix**)	Matrix object	Inverts a matrix.
IsEqualMatrix (matrix as **Matrix**, secondMatrix as **Matrix**)	Boolean	Checks whether two matrices are equal.
IsSingularMatrix (matrix as **Matrix**)	Boolean	Checks whether a matrix is singular and cannot be inverted.
Open (files as **String**, [documentColorSpace as AiDocumentColorSpace], [options as **Object**])	Document object	Opens the file or files specified by the string or array of strings containing file paths. If called to open a pre-Illustrator 9 document that contains both RGB and CMYK colors, supply DocumentColorSpace in order to convert all colors to the specified color space. If not supplied, Illustrator displays a dialog to the user.
Quit ()	Nothing	Quits Illustrator. If the clipboard contains data, a dialog prompts the user to save the data for other applications. To avoid this, empty the clipboard first with Clipboard.Clear.
Redraw ()	Nothing	Forces Illustrator to redraw all its windows.
ShowPresets (fileSpec as **String**)	Nothing	Gets the presets from the file.
TranslatePlaceholderText (text as **String**)	Nothing	Translates the placeholder text to regular text. A way to enter unicode points in hex values. Member of Illustrator.Application.

➤ Opening a document

To open a document and obtain a reference to the document that was opened use this code:

```
' Open a document and get the reference to it
```

```
Dim appRef as New Illustrator.Application
Dim docRef as Illustrator.Document
Set docRef = appRef.Open("C:\temp\aFile.ai")
```

➤ Accessing the selection

In Illustrator, the application's `Selection` can be accessed as well as modified. The selection will contain `Empty` when there are no selected objects. To deselect all objects in the current document, simply set the selection to `Empty`, as the following example shows.

```
Private Sub DeselectAll_Click()
   Dim appRef as New Illustrator.Application
   appRef.activeDocument.Selection = Empty
End Sub
```

A reference to a text range is returned when there is an active insertion point in the contents of a `TextFrame`. Similarly, a reference to a range of text is returned when characters are selected in the contents of a `TextFrame`.

➤ Executing an action

You can run an action from the Action Palette from a script by using the `DoScript` method. When you do this, the control returns to your script before the action has completed. Use the `ActionIsRunning` property to test for when the action has completed before executing any other VB command (see the `Windows.DoAction` example). If you are using VBScript, you can use the `Sleep` method defined on the WScript object to insert a pause to test this property, as in the following example:

```
Dim appRef
Set appRef = CreateObject("Illustrator.Application")
appRef.DoScript "Opacity 60 (selection)", "Default Actions"
WHILE(appRef.ActionIsRunning)
   WScript.sleep 1000
WEND
msgbox "Done"
```

➤ Invoking JavaScript

`DoJavaScript` and `DoJavaScriptFile` can be used to invoke scripts written in JavaScript for Illustrator. See the JavaScript documentation for more information on how to write JavaScripts for Illustrator.

The first argument to `DoJavaScript()` is a string that must evaluate to a valid JavaScript expression. For example, the following script displays an alert using the JavaScript `alert` method.

```
appRef.DoJavaScript "alert( 'Number of open documents: ' + _
   documents.length);"
```

This example returns the number of open documents to Visual Basic.

```
Dim myNumberOfDocuments
myNumberOfDocuments = appRef.DoJavaScript("documents.length;")
MsgBox myNumberOfDocuments
```

➤ Copy and paste between documents

This script uses the application property `ActiveDocument` to copy the current document's selection to the clipboard before pasting it into our new document. This script also demonstrates how to create a new document with a specific color space and dimensions.

```
Dim appRef As New Illustrator.Application
Dim newDocument As Illustrator.Document

If appRef.Documents.Count > 0 Then
  If Not IsEmpty(appRef.ActiveDocument.selection) Then
    appRef.ActiveDocument.Copy
    Set newDocument = appRef.Documents.Add(aiDocumentCMYKColor, _
      250#, 400#)
    newDocument.Paste
  End If
End If
```

Brush

A brush in an Illustrator document. Brushes are contained in documents. Illustrator brushes can be accessed from a script, but not created or deleted.

Brush properties

Property	Value type	What it is
Application	Application	Read-only. The Illustrator Application object.
Name	String	The Brush name.
Parent	Object	Read-only. The container object.

Brush methods

Method	Returns	What it does
ApplyTo (artItem as **PageItem**)	Nothing	Applies the brush to a specific art item.

➤ Applying a Brush

This example duplicates and groups the current selection, applying the second Brush in the document to the items in the group.

```
Dim appRef As New Illustrator.Application
Dim newGroup As Illustrator.GroupItem
Dim dupItem As Object
Dim artItem As Object
Dim i As Integer
Dim endIndex As Integer

If appRef.Documents.Count > 0 Then
  If Not IsEmpty(appRef.ActiveDocument.selection) Then
    endIndex = UBound(appRef.ActiveDocument.selection)
    Set newGroup = appRef.ActiveDocument.GroupItems.Add
    For i = 0 To endIndex
      Set artItem = appRef.ActiveDocument.selection(i)
      Set dupItem = artItem.Duplicate(newGroup, aiPlaceAtEnd)
      appRef.ActiveDocument.Brushes(2).ApplyTo artItem
    Next
  End If
End If
```

Brushes

A collection of brushes in a document.

Brushes properties

Property	Value type	What it is
Application	Application	Read-only. The Illustrator Application object.
Count	Long	Read-only. Long. The number of objects in the collection.
Parent	Object	Read-only. The document that contains this object.

Brushes methods

Method	Returns	What it does
Index (item as **Brush**)	Long	Returns the index position of the object within the collection.
Item (itemKey)	Brush object	Returns an object reference to the object identified by itemKey (name or index).

➤ Get all brushes in a document

This script displays the total number of available brushes in the current document.

```
Dim appRef As New Illustrator.Application
Dim numBrushes As Long

If appRef.Documents.Count > 0 Then
  numBrushes = appRef.ActiveDocument.Brushes.Count
  MsgBox ("There are " & numBrushes & " brushes in the document.")
End If
```

CharacterAttributes

Specifies the properties of a character contained in a text frame. A `CharacterStyle` object associates attributes with a text range.

Note: Character attributes do not have default values, and are undefined until explicitly set.

CharacterAttributes properties

Property	Value Type	What it is
AkiLeft	Double	The amount of inter-glyph space added to the left side of the glyph in Japanese text (in thousandths of an em).
AkiRight	Double	The amount of inter-glyph spacing added to the right side of the glyph in Japanese text (in thousandths of an em).
Alignment	AiStyleRunAlignmentType	The character alignment type.
AlternateGlyphs	AiAlternateGlyphsForm	Specifies which kind of alternate glyphs should be used.
Application	Application object	Read-only. The Illustrator `Application` object.
AutoLeading	Boolean	If `true`, automatic leading must be used.
BaselineDirection	AiBaselineDirectionType	The Japanese text baseline direction.
BaselinePosition	AiFontBaselineOption	The baseline position of text.
BaselineShift	Double	The amount of shift (in points) of the text baseline.
Capitalization	AiFontCapsOption	Specifies whether the text is normal, all uppercase, all small caps, or a mix of small caps and lowercase.
ConnectionForms	Boolean	If `true`, the OpenType® connection forms should be used.
ContextualLigature	Boolean	If `true`, the contextual ligature should be used.
DiscretionaryLigature	Boolean	If `true`, the discretionary ligature should be used.
FigureStyle	AiFigureStyleType	Specifies the figure style to use in an OpenType font.
FillColor	Object	The color of the text fill.

Property	Value Type	What it is
Fractions	Boolean	Specifies whether OpenType fractions should be used.
HorizontalScale	Double	The horizontal scaling factor for the character.
Italics	Boolean	If true, the Japanese font supports italics.
KerningMethod	AiAutoKernType	Specifies the kerning method to be used.
Language	AiLanguageType	The language of the text.
Leading	Double	The amount of space between two lines of text, in points.
Ligature	Boolean	If true, the ligature should be used.
NoBreak	Boolean	If true, no break is allowed.
OpenTypePosition	AiFontOpenTypePositionOption	The OpenType baseline position.
Ordinals	Boolean	If true, the OpenType ordinals should be used.
Ornaments	Boolean	If true, the OpenType ornaments should be used.
OverprintFill	Boolean	If true, overprint the fill of the text.
OverprintStroke	Boolean	If true, overprinting of the stroke of the text is allowed.
Parent	Object	Read-only. The object's container.
ProportionalMetrics	Boolean	If true, proportional metrics in Japanese OpenType is to be used.
Rotation	Double	The character rotation angle.
Size	Double	The font size in points.
StrikeThrough	Boolean	If true, characters use strike-through style.
StrokeColor	Object	The color of the text stroke.
StrokeWeight	Double	Line width of stroke.
StylisticAlternates	Boolean	If true, OpenType stylistic alternates should be used.
Swash	Boolean	If true, the OpenType swash character should be used.
TateChuYokoHorizontal	Long	The Tate-Chu-Yoko horizontal adjustment in points.

Property	Value Type	What it is
TateChuYokoVertical	Long	The Tate-Chu-Yoko vertical adjustment in points.
TextFont	TextFont	The font used for the character.
Titling	Boolean	If true, the OpenType titling alternates should be used.
Tracking	Long	The tracking or range kerning amount in thousandths of an em.
Tsume	Double	The percentage of space reduction around a Japanese character.
Underline	Boolean	If true, characters are underlined.
VerticalScale	Double	Character vertical scaling factor.
WariChuCharactersAfterBreak	Long	Specifies how the characters in Wari-Chu text (an inset note in Japanese text) are divided into two or more lines.
WariChuCharactersBeforeBreak	Long	Specifies how the characters in Wari-Chu text (an inset note in Japanese text) are divided into two or more lines.
WariChuEnabled	Boolean	If true, Wari-Chu is enabled.
WariChuJustification	AiWariChuJustificationType	The Wari-Chu justification.
WariChuLineGap	Long	The Wari-Chu line gap.
WariChuLines	Long	The number of Wari-Chu (multiple text lines fit into a space meant for one) lines.
WariChuScale	Double	The Wari-Chu scale.

➤ Setting character attributes

```
' Character Attributes
'
' Open a document and add a simple text frame
' Use the CharacterAttributes object to increment
' the size of each character in the text frame.

Dim appRef As New Illustrator.Application
Dim docRef As Illustrator.Document
Dim textRef As Illustrator.TextFrame
Dim iCount As Integer
Dim i As Integer
Dim dSize As Double

Set docRef = appRef.Documents.Add()
```

```
Set textRef = docRef.TextFrames.Add()
textRef.Contents = "I'd rather be scripting!"
textRef.Top = 500
textRef.Left = 40

dSize = 100
iCount = textRef.Characters.Count
i = 1
Dim charRef
Do While (i < (iCount + 1))
  dSize = dSize * 1.1
  textRef.TextRange.Characters(i).CharacterAttributes.HorizontalScale =
dSize
  textRef.TextRange.Characters(i).CharacterAttributes.VerticalScale = dSize
  i = i + 1
Loop
```

Characters

A collection of `TextRange` objects in which each represents a single character.

Characters properties

Property	Value type	What it is
Application	Application	Read-only. Application that the collection belongs to.
Count	Long	Read-only. Number of elements in the collection.
Parent	Object	Read-only. The object's container.

Characters methods

Method	Returns	What it does
Add (contents as **String** [, relativeObject as **TextFrame**] [, insertionLocation as AiElementPlacement])	TextRange	Adds a new character with specified text contents at the specified location in the current document. If location is not specified, adds the new character to the containing text frame after the current text selection or insertion point.
AddBefore (contents as **String**)	TextRange	Adds a character before the current text selection or insertion point.
Index (itemPtr as **TextRange**)	Long	Returns the index position of the object within the collection.
RemoveAll ()	Nothing	Deletes every element in the collection.

➤ **Counting characters**

```
Dim appRef As New Illustrator.Application
Dim numChars As Integer
Dim textArt As Illustrator.TextFrame
Dim textArtRange As Illustrator.TextRange

If appRef.Documents.Count > 0 Then
  numChars = 0
  For Each textArt In appRef.ActiveDocument.TextFrames
    Set textArtRange = textArt.TextRange
    numChars = numChars + textArtRange.length
  Next
  MsgBox ("There are " & numChars & " characters in the document.")
End If
```

CharacterStyle

A named style that specifies character attributes. For an example, see CharacterStyles.

CharacterStyle properties

Property	Value type	What it is
Application	Application	Read-only. Application to which the collection belongs.
CharacterAttributes	CharacterAttributes	Read-only. The character properties for the text range.
Name	String	The character style's name.
Parent	Object	Read-only. The object's container.

CharacterStyle methods

Method	Returns	What it does
ApplyTo (textItem as **Object** [, clearingOverrides as **Boolean**])	Nothing	Applies the character style to the text object or objects.
Delete ()	Nothing	Deletes the object.

CharacterStyles

A collection of character styles.

CharacterStyles properties

Property	Value type	What it is
Application	Application	Read-only. Application to which the collection belongs.
Count	Long	Read-only. Number of elements in the collection.
Parent	Object	Read-only. The object's container.

CharacterStyles methods

Method	Returns	What it does
Add (Name as **String**)	CharacterStyle	Creates a named character style.
Index (itemPtr as **CharacterStyle**)	Long	Returns the index position of the object within the collection.
Item (itemKey)	CharacterStyle	Returns an object reference to the object identified by itemKey (name or index).
RemoveAll ()	Nothing	Deletes all objects in this collection.

➤ Create and apply character styles

```
' Open a document, and add 3 simple text frames
' Create a new character style and apply it to
' each of the text frames

Dim appRef As New Illustrator.Application
Dim docRef As Illustrator.Document
Dim textRef1 As Illustrator.TextFrame
Dim textRef2 As Illustrator.TextFrame
Dim textRef3 As Illustrator.TextFrame

' Create a new document add a 3 simple text items
Set docRef = appRef.Documents.Add()
Set textRef1 = docRef.TextFrames.Add()
textRef1.Contents = "Scripting is fun!"
textRef1.Top = 700
textRef1.Left = 50

Set textRef2 = docRef.TextFrames.Add()
textRef2.Contents = "Scripting is easy!"
textRef2.Top = 625
textRef2.Left = 100

Set textRef3 = docRef.TextFrames.Add()
```

```
textRef3.Contents = "Everyone should script!"
textRef3.Top = 550
textRef3.Left = 150
appRef.Redraw

' Create a new character style
MsgBox "Create and apply a character style 'BigRed'."

Dim charStyle As Illustrator.CharacterStyle
Set charStyle = docRef.CharacterStyles.Add("BigRed")

' Create a red color
Dim colorRed As New Illustrator.RGBColor
colorRed.Red = 255

' Set character attributes of the new style
With charStyle.CharacterAttributes
    .Size = 40
    .Tracking = -50
    .Capitalization = aiNormalCaps
    .FillColor = colorRed
End With

' apply style to each textFrame in the document
charStyle.ApplyTo textRef1.TextRange
charStyle.ApplyTo textRef2.TextRange
charStyle.ApplyTo textRef3.TextRange
```

CMYKColor

A CMYK color specification, used to apply a CMYK color to a layer or art item.

If the color space of a document is RGB and you specify the color value for a page item in that document using CMYK, Illustrator will translate the CMYK color specification into an RGB color specification. The same thing happens if the document's color space is CMYK and you specify colors using RGB. Since this translation can lose information, you should specify colors using the class that matches the document's actual color space.

CMYKColor properties

Property	Value type	What it is
Application	Application object	Read-only. The Illustrator Application object.
Black	Double	The black color value. Range 0.0 to 100.0
Cyan	Double	The cyan color value. Range 0.0 to 100.0
Magenta	Double	The magenta color value. Range 0.0 to 100.0
Yellow	Double	The yellow color value. Range 0.0 to 100.0

➤ Setting CMYK colors

```
Dim appRef As New Illustrator.Application
Dim frontPath As Illustrator.PathItem
Dim newCMYKColor As New Illustrator.CMYKColor

' Get a reference to the frontmost path in the document
Set frontPath = appRef.ActiveDocument.PathItems(1)

' Set color values for the CMYK objects
' then wrap the color in a standard color object
newCMYKColor.Black = 0
newCMYKColor.Cyan = 30.4
newCMYKColor.Magenta = 32
newCMYKColor.Yellow = 0

frontPath.Filled = True
frontPath.fillColor = newCMYKColor
```

CompoundPathItem

Compound paths are objects composed of multiple intersecting paths, resulting in transparent interior spaces where the original paths overlapped. The PathItems property provides access to the paths that make up the compound path.

All paths in a compound path share property values. Therefore, if you set the value of a property of any one of the paths in the compound path, all other path's matching property will be updated to the new value.

Paths contained within a compound path or group in a document are returned as individual paths when a script asks for the paths contained in the document. However, paths contained in a compound path or group are not returned when a script asks for the paths in a layer which contains the compound path or group.

CompoundPathItem properties

Property	Value type	What it is
Application	Application object	Read-only. The Illustrator Application object.
ArtworkKnockout	AiKnockoutState	Is this object used to create a knockout? If so, what kind of knockout?
BlendingMode	AiBlendModes	The mode used when compositing an object.
ControlBounds	Variant Array of 4 Doubles	Read-only. The bounds of the object including stroke width and controls.
Editable	Boolean	Read-only. If true, the object is editable.
GeometricBounds	Variant Array of 4 Doubles	Read-only. The bounds of the object excluding stroke width.
Height	Double	The height of the object, excluding stroke width, calculated from the GeometricBounds.
Hidden	Boolean	If true, this object is hidden.
IsIsolated	Boolean	If true, this object is isolated.
Layer	Layer object	Read-only. The layer to which this object belongs.
Left	Double	The left position of the object.
Locked	Boolean	If true, the object is locked.
Name	String	The name of this object .
Opacity	Double	The opacity of the object. The value is between 0.0 and 100.0.
Parent	Document object	Read-only. The document that contains this object.
PathItems	PathItems collection object	Read-only. The path art items in this compound path.

Property	Value type	What it is
Position	Variant Array of 2 Doubles	The position of the top left corner of the CompoundPathItem excluding stroke width.
Selected	Boolean	If true, the object is selected.
Sliced	Boolean	If true, the CompoundPathItem is sliced. Default: false
Tags	Tags collection object	Read-only. The tags contained in this object.
Top	Double	The top position of the object.
URL	String	The value of the Adobe URL tag assigned to this object.
VisibilityVariable	Variable	The VisibilityVariable bound to this object.
VisibleBounds	Variant Array of 4 Doubles	Read-only. The visible bounds of the object including stroke width.
Width	Double	The width of the object, excluding stroke width, calculated from the GeometricBounds.
WrapInside	Boolean	If true, the text frame object should be wrapped inside this object.
WrapOffset	Double	The offset to use when wrapping text around this object.
Wrapped	Boolean	If true, wrap text frame objects around this object (text frame must be above the object).
ZOrderPosition	Long	Read-only. The position of this art item within the stacking order of the group or layer (Parent) that contains the art item.

CompoundPathItem methods

Method	Returns	What it does
Copy ()	Nothing	Copies the compound path to the clipboard. The associated document must be the frontmost document.
Cut ()	Nothing	Cuts the compound path onto the clipboard. The associated document must be the frontmost document.
Delete ()	Nothing	Deletes the object.
Duplicate ([relativeObject as **Object**], [insertionLocation as AiElementPlacement])	Object	Duplicates the art item, optionally specifying the location and position for the copy.
Move (relativeObject as **Object**, insertionLocation as AiElementPlacement)	Nothing	Moves the art item, specifying the new location and position.

Method	Returns	What it does
Resize (scaleX as **Double**, scaleY as **Double**, [, changePositions as **Boolean**] [, changeFillPatterns as **Boolean**] [, changeFillGradients as **Boolean**] [, changeStrokePattern as **Boolean**] [, changeLineWidths as **Double**] [, scaleAbout as AiTransformation])	Nothing	Scales the compound path where scaleX is the horizontal scaling factor and scaleY is the vertical scaling factor; 100.0 = 100%
Rotate (angle as **Double**, [, changePositions as **Boolean**] [, changeFillPatterns as **Boolean**] [, changeFillGradients as **Boolean**] [, changeStrokePattern as **Boolean**] [, rotateAbout as AiTransformation])	Nothing	Rotates the art item relative to the current rotation. The object is rotated counter-clockwise if the Angle value is positive, clockwise if the value is negative.
Transform (transformationMatrix as **Matrix**, [, changePositions as **Boolean**] [, changeFillPatterns as **Boolean**] [, changeFillGradients as **Boolean**] [, changeStrokePattern as **Boolean**] [, changeLineWidths as **Double**] [, transformAbout as AiTransformation])	Nothing	Transforms the compound path by applying a transformation matrix.
Translate ([deltaX as **Double**] [, deltaY as **Double**] [, transformObjects as **Boolean**] [, transformFillPatterns as **Boolean**] [, transformFillGradients as **Boolean**] [, transformStrokePatterns as **Boolean**])	Nothing	Repositions the compound path relative to the current position, where deltaX is the horizontal offset and deltaY is the vertical offset.
ZOrder (zOrderCmd as AiZOrderMethod)	Nothing	Arranges the compound path's position in the stacking order of the group or layer (Parent) of this object.

➤ Selecting paths in a document

This example demonstrates how to select all of the paths in a document that are not part of a compound path or a group by testing the type of the Parent property with a TypeName function.

```
Dim appRef As New Illustrator.Application
Dim pathArt As Illustrator.PathItem

If appRef.Documents.Count > 0 Then
  If appRef.ActiveDocument.PathItems.Count > 0 Then
    For Each pathArt In appRef.ActiveDocument.PathItems
      If (Not TypeName(pathArt.Parent) = "CompoundPathItem") Then _
          pathArt.Selected = True
    Next
  End If
End If
```

➤ Creating a compound path

This example demonstrates how to create a new compound path containing three `PathItems`. The example then modifies the stroke of the paths in the compound path. Note that when you modify the properties of a `PathItem` inside a compound path you affect all paths contained in the compound path. The example also shows how to access swatches in a document by name.

```
Dim appRef As New Illustrator.Application
Dim frontDocument As Illustrator.Document
Dim activeLayer As Illustrator.Layer
Dim newCompoundPath As Illustrator.CompoundPathItem
Dim newPath As Illustrator.PathItem

Set frontDocument = appRef.ActiveDocument
Set activeLayer = frontDocument.activeLayer
Set newCompoundPath = activeLayer.CompoundPathItems.Add

' Create the path items
Set newPath = newCompoundPath.PathItems.Add
newPath.SetEntirePath Array(Array(30, 50), Array(30, 100))

Set newPath = newCompoundPath.PathItems.Add
newPath.SetEntirePath Array(Array(40, 100), Array(100, 100))

Set newPath = newCompoundPath.PathItems.Add
newPath.SetEntirePath Array(Array(100, 110), Array(100, 300))

' Set the gradient of the compound path
newPath.Stroked = True
newPath.StrokeWidth = 3.5
newPath.StrokeColor = frontDocument.Swatches(8).Color
```

CompoundPathItems

A collection of compound paths.

CompoundPathItems properties

Property	Value type	What it is
Application	Application object	Read-only. The Illustrator Application object.
Count	Long	Read-only. The number of objects in the collection.
Parent	Document object	Read-only. The document that contains this object.

CompoundPathItem methods

Method	Returns	What it does
Add ()	CompountPathItem object	Creates a new CompoundPathItem object.
Index (item as **CompoundPathItem**)	Long	Returns the index position of the object within the collection.
Item (itemKey)	CompoundPathItem object	Returns an object reference to the object identified by itemKey (name or index).
RemoveAll ()	Nothing	Deletes all objects in this collection.

➤ Counting compound paths

This example displays the total number of compound paths contained in the first layer of the current document.

```
Dim appRef As New Illustrator.Application
Dim numCPaths As Integer

If appRef.Documents.Count > 0 Then
  numCPaths = appRef.ActiveDocument.Layers(1).CompoundPathItems.Count
  MsgBox ("There are " & numCPaths & " compound paths in document.")
End If
```

DataSet

A set of data used for dynamic publishing. A dataset allows you to collect a number of variables and their dynamic data into one object. You must have at least one variable bound to an art item in order to create a `DataSet` object.

- See the `Variable` class, and Working with Variables and Datasets for more information.
- See DataSets for a sample script that uses a `DataSet` object.

DataSet properties

Property	Value type	What it is
`Application`	`Application` object	Read-only. The Illustrator `Application` object.
`Name`	`String`	The name of the `DataSet`.
`Parent`	`Document` object	Read-only. The name of the object that is this `DataSet` object's parent.

DataSet methods

Method	Returns	What it does
`Delete` `()`	`Nothing`	Deletes the object.
`Display` `()`	`Nothing`	Displays the `DataSet`.
`Update` `()`	`Nothing`	Updates the `DataSet`.

DataSets

A collection of `DataSet` objects.

DataSets properties

Property	Value type	What it is
Application	`Application` object	Read-only. The Illustrator `Application` object.
Count	`Long`	Read-only. The number of datasets in the collection.
Parent	`Document` object	Read-only. The name of the object that is this `DataSet` object's parent.

DataSets methods

Method	Returns	What it does
Add ()	`DataSet` object	Creates a new `DataSet` object.
Index (item as DataSets)	`Long`	Returns the index position of the object within the collection.
Item (itemKey)	`DataSet` object	Returns an object reference to the object identified by `itemKey` (name or index).
RemoveAll ()	`Nothing`	Deletes all objects in this collection.

➤ Creating datasets

```
' Variables and Datasets
'
' Create two variables (visibility and text)
' Create two datasets with different values
' Display both datasets

Dim appRef As New Illustrator.Application
Dim docRef As Illustrator.Document
Set docRef = appRef.Documents.Add()

' Create a visibility variable
Dim itemRef As Illustrator.pathItem
Set itemRef = docRef.PathItems.Rectangle(600, 200, 150, 150)
Dim colorRef As New RGBColor
colorRef.Red = 255
itemRef.FillColor = colorRef
Dim visibilityVar As Illustrator.Variable
Set visibilityVar = docRef.Variables.Add()
visibilityVar.Kind = aiVisibility
itemRef.VisibilityVariable = visibilityVar

' Create a text variable
```

```
Dim textRef As Illustrator.TextFrame
Set textRef = docRef.TextFrames.Add
textRef.Contents = "Text Variable, dataset 1"
textRef.Top = 400
textRef.Left = 200
Dim textVar As Illustrator.Variable
Set textVar = docRef.Variables.Add
textVar.Kind = aiTextual
textRef.ContentVariable = textVar

MsgBox "There are " & CStr(docRef.Variables.Count) & " variables"
MsgBox "Creating new datasets."

' Create a dataset
Dim ds1 As Illustrator.DataSet
Set ds1 = docRef.DataSets.Add()

' Change data and create a 2nd dataset
itemRef.Hidden = True
textRef.Contents = "Text Variable, dataset 2"
Dim ds2 As Illustrator.DataSet
Set ds2 = docRef.DataSets.Add()

' Display datasets
MsgBox "Switching to dataset 1"
ds1.Display
MsgBox "Switching to dataset 2"
ds2.Display
```

Document

An Illustrator document. Documents are contained in the `Application` object.

The default document settings—those properties starting with the word "Default"—are global settings that affect the current document. Be sure to modify these default properties only when a document is open. Note that if you set default properties to desired values before creating new objects, you can streamline your scripts, eliminating the need to specify properties such as `FillColor` and `Stroked` that have analogous default properties.

A document's `DocumentColorSpace`, `Height`, and `Width` can only be set when the document is created. Once a document is created, these properties cannot be changed.

The frontmost document can be referred to as either `AppRef.ActiveDocument` or `AppRef.Documents(1)`.

Document properties

Property	Value type	What it is
ActiveDataSet	DataSet object	The active DataSet object in the document.
ActiveLayer	Layer object	The active Layer in the document.
ActiveView	View object	Read-only. The document's current View.
Application	Application object	Read-only. The Illustrator Application object.
Brushes	Brushes collection object	Read-only. The Brushes contained in the document.
CharacterStyles	CharacterStyles	Read-only. The list of character styles in this document.
CompoundPathItems	CompoundPathItems collection object	Read-only. The CompoundPathItems contained in the document.
CropBox	Variant Array of 4 Doubles	The boundary of the document's cropping box for output. A document does not have a default CropBox. In order to read this property you have to set the CropBox first.
CropStyle	AiCropOptions	The style of the document's cropping box.
DataSets	DataSet object	Read-only. A DataSets collection in the document.
DefaultFillColor	Color object	The Color to fill new paths if default filled is true.
DefaultFilled	Boolean	If true, a new path should be filled.
DefaultFillOverprint	Boolean	If true, the art beneath a filled object should be overprinted by default.
DefaultStrokeCap	AiStrokeCap	Default type of line capping for paths created.

Property	Value type	What it is
DefaultStrokeColor	Color object	The stroke color for new paths if default stroked is true.
DefaultStroked	Boolean	If true, the new path should be stroked.
DefaultStrokeDashes	Variant Array of Doubles	Default lengths for dashes and gaps in dashed lines, starting with the first dash length, followed by the first gap length, and so on. Set to an empty variant array for solid line.
DefaultStrokeDashOffset	Double	The default distance into the dash pattern at which the pattern should be started for new paths.
DefaultStrokeJoin	AiStrokeJoin	Default type of joints in new paths.
DefaultStrokeMiterLimit	Double	When a default stroke join is set to mitered, this property specifies when the join will be converted to beveled (squared-off) by default. The default miter limit of 4 means that when the length of the point reaches four times the stroke weight, the join switches from a miter join to a bevel join. Range: 1 to 500; a value of 1 specifies a bevel join.
DefaultStrokeOverprint	Boolean	If true, the art beneath a stroked object should be overprinted by default.
DefaultStrokeWidth	Double	Default width of stroke for new paths.
DocumentColorSpace	AiDocumentColorSpace	Read-only. The color specification system to use for this document's color space.
FullName	String	Read-only. The file associated with the document, which includes the complete path to the file.
GeometricBounds	Variant Array of 4 Doubles	Read-only. The bounds of the illustration excluding the stroke width of any objects in the document.
Gradients	Gradients collection object	Read-only. The Gradients collection contained in the document.
GraphicStyles	GraphicStyles collection object	Read-only. The GraphicStyles contained in the document.
GraphItems	collection object	Read-only. The GraphItems collection contained in the document.
GroupItems	GroupItems collection object	Read-only. The GroupItems collection contained in the document.
Height	Double	Read-only. The height of the document, calculated from the GeometricBounds.

Property	Value type	What it is
InkList	Variant Array	Read-only. The list of inks in this document.
KinsokuSet	Object	Read-only. The Kinsoku set of characters that cannot begin or end a line of Japanese text.
Layers	Layers collection object	Read-only. The layers contained in the document.
LegacyTextItems	LegacyTextItems	Read-only. The text frame items in this story.
MeshItems	MeshItems collection object	Read-only. The mesh art items contained in the document.
MojikumiSet	Object	Read-only. A list of names of predefined Mojikumi sets which specify the spacing for the layout and composition of Japanese text.
Name	String	Read-only. The document's name (not the complete file path to the document).
OutputResolution	Double	The current output resolution for the document in dots per inch (dpi).
PageItems	PageItems collection object	Read-only. The PageItems (contains all art item classes) contained in the document.
PageOrigin	Variant Array of 2 Doubles	The zero-point of the page in the document without margins, relative to the overall height and width.
ParagraphStyles	ParagraphStyles	Read-only. The list of paragraph styles in this document.
Parent	Application object	Read-only. The application that contains this document.
Path	String	Read-only. The file associated with the document, which includes the complete path to the file.
PathItems	PathItems collection object	Read-only. The PathItems contained in this document.
Patterns	Patterns collection object	Read-only. The patterns contained in this document.
PlacedItems	PlacedItems collection object	Read-only. The PlacedItems contained in this document.
PluginItems	PluginItems collection object	Read-only. The PluginItems contained in this document.
PrintTiles	Boolean	Read-only. If true, this document should be tiled when printed.
RasterItems	RasterItems collection object	Read-only. The raster items contained in this document.

Property	Value type	What it is
RulerOrigin	Variant Array of 2 Doubles	The zero-point of the rulers in the document relative to the bottom left of the document.
RulerUnits	AiRulerUnits	Read-only. The default measurement units for the rulers in the document.
Saved	Boolean	If true, the document has been saved, or the document has not been changed since the last time it was saved.
Selection	Variant Array of objects	The array of references to the objects in this document's current selection.
ShowPlacedImages	Boolean	Read-only. If true, placed images should be displayed in the document.
SplitLongPaths	Boolean	Read-only. If true, long paths should be split when printing.
Spots	Spots collection object	Read-only. The SpotColors contained in this document.
Stationary	Boolean	Read-only. If true, the file is a stationery file.
Stories	Stories	Read-only. The story items in this document.
Swatches	Swatches collection object	Read-only. The Swatches contained in this document.
SymbolItems	SymbolItems collection object	Read-only. The SymbolItems collection contained in this document.
Symbols	Symbols collection object	Read-only. The Symbols collection contained in this document.
Tags	Tags collection object	Read-only. The tags contained in this document.
TextFrames	TextFrames collection object	Read-only. The TextFrames contained in this document.
TileFullPages	Boolean	Read-only. If true, full pages should be tiled when printing this document.
UseDefaultScreen	Boolean	Read-only. If true, the printer's default screen should be used when printing this document.
Variables	Variables collection object	Read-only. The Variables collection contained in this document.
VariablesLocked	Boolean	If true, the Variables in this document are locked.
Views	Views collection object	Read-only. The views contained in this document.

Property	Value type	What it is
VisibleBounds	Variant Array of 4 Doubles	Read-only. The visible bounds of the document, including stroke width of any objects in the illustration.
Width	Double	Read-only. The width of this document, calculated from the GeometricBounds.

Document methods

Method	Returns	What it does
Activate ()	Nothing	Bring the first window associated with the document to the front
Close ([saving as AiSaveOptions])	Nothing	Closes a document
Copy ()	Nothing	Copies the current selection in the document to the clipboard; the associated document must be the frontmost document
Cut ()	Nothing	Cuts the current selection in the document to the clipboard; the associated document must be the frontmost document
ExportPDFPreset (File as a **String**)	Nothing	Save all PDF presets to a file
Export (exportFile as **String**, exportFormat as AiExportType, [options As **ExportOptionsFlash**/ **ExportOptionsGIF**/ **ExportOptionsJPEG**/ **ExportOptionsPhotoshop**/ **ExportOptionsPNG24**/ **ExportOptionsPNG8**/ **ExportOptionsSVG**])	Nothing	Exports the document to the specified file using one of the export file formats
ExportPDFPreset ()	String	Exports the current PDF preset values to the file
ExportPrintPreset ()	String	Exports the current print preset values to the file
ExportVariables (Filename as **String**)	Nothing	Exports Variables from this document to a specified file
ImportCharacterStyles (FileSpec as **String**)	Nothing	Load the character styles from the Illustrator file
ImportParagraphStyles (FileSpec as **String**)	Nothing	Load the paragraph styles from the Illustrator file

Method	Returns	What it does
ImportPDFPreset (FileSpec as **String** [, ReplacingPreset as **Boolean**])	Nothing	Load all PDF presets from a file
ImportPrintPreset (printPreset as **String** FileSpec as **String**)	Nothing	Load the named print preset from the file
ImportVariables(filename as **String**)	Nothing	Import Variables from specified file into this document
Paste ()	Nothing	Pastes the contents of the clipboard into the current layer of the document; if the document is the frontmost then all pasted objects remain selected after the paste
PrintOut (options as **PrintOptions**)	Nothing	Prints the document
Save ()	Nothing	Saves the document in it current location
SaveAs ([saveIn as **String**] [, options As: **EPSSaveOptions**/ **IllustratorSaveOptions**/ **PDFSaveOptions**])	Nothing	Saves the document in the specified file as an Illustrator, EPS, or PDF file

➤ Closing documents

If you close the document, you should set your document reference to Nothing to prevent your script accidentally trying to access closed documents.

```
doc.Close aiDoNotSaveChanges
Set doc = Nothing
```

➤ Creating a document with defaults

This example demonstrates how to create a new document with specific default properties.

```
Dim appRef As New Illustrator.Application
Dim frontDocument As Illustrator.Document

If (appRef.Documents.Count = 0) Then
  Set frontDocument = appRef.Documents.Add
Else
  Set frontDocument = appRef.Documents(1)
End If
frontDocument.DefaultFilled = True
frontDocument.DefaultStroked = True
```

Documents

A collection of documents.

Documents properties

Property	Value type	What it is
Application	Application object	Read-only. The Illustrator Application object.
Count	Long	Read-only. The number of objects in the collection.
Parent	Object	Read-only. The parent of this object.

Documents methods

Method	Returns	What it does
Add ([documentColorSpace As AiDocumentColorSpace] [, width as **Double**] [, height as **Double**])	Document object	Creates a new document using optional parameters and returns a reference to the new document
Index (item as **Document**)	Long	Returns the index position of the object within the collection
Item (itemKey)	Document object	Returns an object reference to the object identified by itemKey (name or index).

➤ Creating a document with a color space

This examples demonstrates how to create a new document with a specific color space.

```
Dim appRef As New Illustrator.Application
appRef.Documents.Add aiDocumentRGBColor
```

EPSSaveOptions

Options which may be supplied when saving a document as an Illustrator EPS file. `EPSSaveOptions` can only be supplied in conjunction with the `document SaveAs` method. All properties are optional.

EPSSaveOptions properties

Property	Value type	What it is
Application	`Application object`	Read-only. The Illustrator `Application` object.
CMYKPostScript	`Boolean`	If `true`, CMYK PostScript® should be used.
Compatibility	`AiCompatibility`	Illustrator version compatibility for the EPS file format.
CompatibleGradientPrinting	`Boolean`	If `true`, a raster item of the gradient or gradient mesh should be created so the object can be printed on PostScript level 2 printers. Default: `false`
EmbedAllFonts	`Boolean`	If `true`, fonts should be included in the EPS file.
EmbedLinkedFiles	`Boolean`	If `true`, linked image files should be included in the saved document.
FlattenOuput	`AiOutputFlattening`	How should transparency be flattened for file formats older than Illustrator 9 or greater.
IncludeDocumentThumbnails	`Boolean`	If `true`, the thumbnail image of the EPS artwork should be included.
Overprint	`AiPDFOverprint`	Style of overprinting. Default: `PreservePDFOverprint`
PostScript	`AiEPSPostScriptLevelEnum`	Specifies the PostScript level to use when saving the file.
Preview	`AiEPSPreviewX`	Format for the EPS preview image.

➤ Saving to EPS

This example demonstrates how to save the current document as an Illustrator 8-compatible EPS file using CMYK PostScript with all fonts embedded.

```
Dim appRef As New Illustrator.Application
Dim newSaveOptions As New Illustrator.EPSSaveOptions
Dim frontDocument As Illustrator.Document

If appRef.Documents.Count > 0 Then
  newSaveOptions.CMYKPostScript = True
  newSaveOptions.Compatibility = aiIllustrator8
```

```
  newSaveOptions.EmbedAllFonts = True
  Set frontDocument = appRef.ActiveDocument
  frontDocument.SaveAs "C:\temp\SaveAs.eps", newSaveOptions
End If
```

ExportOptionsFlash

Specifies options that you can supply when exporting a document as Macromedia® Flash™ (SWF) with the Document `Export` method.

When you export a document, the appropriate file extension is appended automatically. You should not include any file extension in the file specification.

ExportOptionsFlash properties

Property	Value type	What it is
Application	`Application object`	Read-only. The Illustrator `Application` object.
ArtBoardClipping	`Boolean`	If `true`, the exported image should be clipped to the art board. Default: `false`
BackgroundColor	`RGBColor object`	The background color of the exported Flash frames.
BackgroundLayers	`Variant Array of Layer objects`	A list of layers to be included as the static background of the exported Flash frames.
BlendAnimation	`AiBlendAnimationType`	How the blend art objects are animated when exported to Flash frames. Default: `AiNoBlendAnimation`
Compressed	`boolean`	If `true`, the exported file should be exported compressed. Default: `false`
ConvertTextToOutlines	`boolean`	If `true`, all text should be converted to outlines. Default: `false`
CurveQuality	`Long`	The amount of curve information that should be preserved. Range: 0 to 10, Default: 7
ExportStyle	`AiFlashExportStyle`	The style in which the exported data should be created in Flash. Default: `aiAsFlashFile`.
FlattenOutput	`AiOutputFlattening`	How transparency should be flattened to preserve appearance or editability on export. Default: `aiPreserveAppearance`
FrameRate	`Double`	The display rate in frames per second; Range: 0.01 to 12.0, Default: 12.0
GenerateHTML	`Boolean`	If `true`, the image should be exported as an HTML file. Default: `true`
ImageFormat	`AiFlashImageFormat`	How the image in the exported Flash file should be compressed. Default: `aiLossless`
JPEGMethod	`AiFlashJPEGMethod`	The JPEG method to use. Default: either `aiJPEGStandard` or `aiJPEGOptimized`
JPEGQuality	`Long`	Level of compression to use. Range: 0 to 10. Default: 3

Property	Value type	What it is
LayerOrder	AILayerOrderType	The order in which layers are exported to Flash frames. Default: AiBottomUp
Looping	Boolean	If true, the Flash file should be set to loop when run. Default: false
ReadOnly	Boolean	If true, the file should be exported as read-only. Default: false
Replacing	AiSaveOptions	If a file with the same name already exists, should it be replaced. Default: AiPromptToSaveChanges
Resolution	Double	The resolution in pixels per inch. Range: 72 to 2400. Default: 72

ExportOptionsGIF

Options which may be supplied when exporting a document as a GIF file. See the Document `Export` method for additional details.

When you export a document, the appropriate file extension is appended automatically. You should not include any file extension in the file specification.

ExportOptionsGIF properties

Property	Value type	What it is
AntiAliasing	Boolean	If `true`, the exported image should be anti-aliased. Default: `true`
Application	Application object	Read-only. The Illustrator `Application` object.
ArtBoardClipping	Boolean	If `true`, the exported image should be clipped to the art board. Default: `false`
ColorCount	Long	The number of colors in the exported image's color table. Range: 2 to 256. Default: 128
ColorDither	AiColorDitherMethod	The method used to dither colors in the exported image. Default: `aiDiffusionDither`
ColorReduction	AiColorReductionMethod	The method used to reduce the number of colors in the exported image. Default: `aiSelective`
DitherPercent	Long	How much should the colors of the exported image be dithered, where 100.0 is 100%
HorizontalScale	Double	The horizontal scaling factor to apply to the exported image, where 100.0 is 100%. Default: 100.0
InfoLossPercent	Long	The level of information loss allowed during compression, where 100.0 is 100%. Default: `0%`
Interlaced	Boolean	If `true`, the exported image should be interlaced. Default: `false`
Matte	Boolean	If `true`, the art board should be matted with a color. Default: `true`
MatteColor	RGBColor object	The color to use when matting the art board. Default: `white`
SaveAsHTML	Boolean	If `true`, the exported image should be saved with an accompanying HTML file. Default: `false`
Transparency	Boolean	If `true`, the exported image should use transparency. Default: `true`

Property	Value type	What it is
VerticalScale	Double	The vertical scaling factor to apply to the exported image, where 100.0 is 100%. Default: 100.0
WebSnap	Long	How much should the color table be changed to match the web palette, where 100 is maximum. Default: 0

➤ Exporting as GIF

This example demonstrates how to export the current document as a GIF.

```
Dim appRef As New Illustrator.Application
Dim gifExportOptions As New Illustrator.ExportOptionsGIF
Dim docRef As Illustrator.Document
If appRef.Documents.Count > 0 Then
  gifExportOptions.AntiAliasing = False
  gifExportOptions.ColorCount = 64
  gifExportOptions.ColorDither = aiDiffusion
  Set docRef = appRef.ActiveDocument
  docRef.Export "C:\temp\AiExport.gif", aiGIF, gifExportOptions
End If
```

ExportOptionsJPEG

Options which may be supplied when exporting a document as a JPEG file. See the Document `Export` method for additional details.

When you export a document, the appropriate file extension is appended automatically. You should not include any file extension in the file specification.

ExportOptionsJPEG

Property	Value type	What it is
AntiAliasing	`Boolean`	If `true`, the exported image should be anti-aliased. Default: `true`
Application	`Application object`	Read-only. The Illustrator `Application` object.
ArtBoardClipping	`Boolean`	If `true`, the exported image should be clipped to the art board.
BlurAmount	`Double`	The amount of blur to apply to the exported image. Range: 0.0 to 2.0. Default: 0.0
HorizontalScale	`Double`	The horizontal scaling factor to apply to the exported image, where 100.0 is 100%. Default: 100.0
Matte	`Boolean`	If `true`, the art board should be matted with a color. Default: `true`
MatteColor	`RGBColor object`	The color to use when matting the art board. Default: `white`
Optimization	`Boolean`	If `true`, the exported image should be optimized for web viewing. Default: `true`
QualitySetting	`Long`	The quality of the exported image. Range: 0 to 100. Default: 30
SaveAsHTML	`Boolean`	If `true`, the exported image should be saved with an accompanying HTML file. Default: `false`
VerticalScale	`Double`	The vertical scaling factor to apply to the exported image, where 100.0 is 100%. Default: 100.0

➤ Exporting to JPEG

This example demonstrates how to export the current document as a JPEG with specific options.

```
Dim appRef As New Illustrator.Application
Dim jpegExportOptions As New Illustrator.ExportOptionsJPEG
Dim docRef As Illustrator.Document
If appRef.Documents.Count > 0 Then
  jpegExportOptions.AntiAliasing = False
  jpegExportOptions.QualitySetting = 70
  Set docRef = appRef.ActiveDocument
  docRef.Export "C:\temp\AiExport.jpg", aiJPEG, jpegExportOptions
End If
```

ExportOptionsPhotoshop

Options which may be supplied when exporting a document as an Adobe Photoshop® file. See the Document `Export` method for additional details.

When you export a document, the appropriate file extension is appended automatically. You should not include any file extension in the file specification.

ExportOptionsPhotoshop properties

Property	Value type	What it is
AntiAliasing	`Boolean`	If `true`, the exported image should be anti-aliased. Default: `true`
Application	`Application object`	Read-only. The Illustrator `Application` object.
Compatibility	`AiPhotoshopCompatibility`	Which Photoshop file format to create. Default: `Photoshop8`
EditableText	`Boolean`	If `true`, text objects should be exported as editable text layers. Default: `true`
EmbedICCProfile	`Boolean`	If `true`, an ICC profile should be embedded in the exported file. Default: `false`
ImageColorSpace	`AiImageColorSpace`	The color space of the exported file
MaximumEditability	`Boolean`	If `true`, preserve as much of the original document's structure as possible when exporting. Default: `true`
Resolution	`Double`	The resolution of the exported file in dots per inch. Range: 72 to 2400. Default: `150`
Warnings	`Boolean`	If `true`, a warning dialog should be displayed because of conflicts in the export settings. Default: `true`
WriteLayers	`Boolean`	If `true`, the document layers should be preserved in the exported file. Default: `true`

➤ Exporting to Photoshop

This example exports the current document as a Photoshop 5 file with layers.

```
Dim appRef As New Illustrator.Application
Dim psdExportOptions As New Illustrator.ExportOptionsPhotoshop
Dim docRef As Illustrator.Document
If appRef.Documents.Count > 0 Then
  psdExportOptions.Resolution = 150
  Set docRef = appRef.ActiveDocument
  docRef.Export "C:\temp\AiExport.psd", aiPhotoshop, psdExportOptions
End If
```

ExportOptionsPNG8

Options which may be supplied when exporting a document as an 8-bit PNG file. See the Document `Export` method for additional details.

When you export a document, the appropriate file extension is appended automatically. You should not include any file extension in the file specification.

ExportOptionsPNG8 properties

Property	Value type	What it is
AntiAliasing	`Boolean`	If `true`, the exported image should be anti-aliased. Default: `true`
Application	`Application object`	Read-only. The Illustrator `Application` object.
ArtBoardClipping	`Boolean`	If `true`, the exported image should be clipped to the art board. Default: `false`
ColorCount	`Long`	The number of colors in the exported image's color table. Range: 2 to 256. Default: 128
ColorDither	`AiColorDitherMethod`	The method used to dither colors in the exported image. Default: `aiDiffusionDither`
ColorReduction	`AiColorReductionMethod`	The method used to reduce the number of colors in the exported image. Default: `aiSelective`
DitherPercent	`Long`	Specifies how much the colors of the exported image should be dithered, where 100.0 is 100%
HorizontalScale	`Double`	The horizontal scaling factor to apply to the exported image, where 100.0 is 100%. Default: 100.0
Interlaced	`Boolean`	If `true`, the exported image should be interlaced. Default: `false`
Matte	`Boolean`	If `true`, the art board should be matted with a color. Default: `true`
MatteColor	`RGBColor object`	The color to use when matting the art board. Default: `white`
SaveAsHTML	`Boolean`	If `true`, the exported image should be saved with an accompanying HTML file. Default: `false`
Transparency	`Boolean`	If `true`, the exported image should use transparency. Default: `true`
VerticalScale	`Double`	The vertical scaling factor to apply to the exported image, where 100.0 is 100%. Default: 100.0
WebSnap	`Long`	Specifies how much the color table should be changed to match the Web palette, where 100 is maximum. Default: 0

➤ Exporting to PNG8

This example exports the current document as a PNG8 file.

```
Dim appRef As New Illustrator.Application
Dim png8ExportOptions As New Illustrator.ExportOptionsPNG8
Dim docRef As Illustrator.Document
If appRef.Documents.Count > 0 Then
  png8ExportOptions.AntiAliasing = False
  png8ExportOptions.Interlaced = True
  Set docRef = appRef.ActiveDocument
  docRef.Export "C:\temp\AiExport.png", aiPNG8, png8ExportOptions
End If
```

ExportOptionsPNG24

Options which may be supplied when exporting a document as a 24-bit PNG file. See the Document `Export` method for additional details.

When you export a document, the appropriate file extension is appended automatically. You should not include any file extension in the file specification.

ExportOptionsPNG24 properties

Property	Value type	What it is
`AntiAliasing`	`Boolean`	If `true`, the exported image should be anti-aliased. Default: `true`
`Application`	`Application object`	Read-only. The Illustrator `Application` object.
`ArtBoardClipping`	`Boolean`	If `true`, the exported image should be clipped to the art board. Default: `false`
`HorizontalScale`	`Double`	The horizontal scaling factor to apply to the exported image, where 100.0 is 100%. Default: 100.0
`Matte`	`Boolean`	If `true`, the art board should be matted with a color. Default: `true`
`MatteColor`	`RGBColor object`	The color to use when matting the art board. Default: `white`
`SaveAsHTML`	`Boolean`	If `true`, the exported image should be saved with an accompanying HTML file. Default: `false`
`Transparency`	`Boolean`	If `true`, the exported image should use transparency. Default: `true`
`VerticalScale`	`Double`	The vertical scaling factor to apply to the exported image, where 100.0 is 100%. Default: 100.0

➤ Exporting to PNG24

This example exports the current document as a PNG24 file with specific options.

```
Dim appRef As New Illustrator.Application
Dim png24ExportOptions As New Illustrator.ExportOptionsPNG24
Dim docRef As Illustrator.Document
If appRef.Documents.Count > 0 Then
  png24ExportOptions.AntiAliasing = False
  png24ExportOptions.Transparency = False
  Set docRef = appRef.ActiveDocument
  docRef.Export "C:\temp\AiExport.png", aiPNG24, png24ExportOptions
End If
```

ExportOptionsSVG

Options which may be supplied when exporting a document as a SVG file. See the Document Export method for additional details.

When you export a document, the appropriate file extension is appended automatically. You should not include any file extension in the file specification.

ExportOptionsSVG properties

Property	Value type	What it is
Application	Application object	Read-only. The Illustrator Application object.
Compressed	Boolean	If true, the exported file should be compressed. Default is false
CoordinatePrecision	Long	The decimal precision for element coordinate values. Range: 1 to 7. Default: 3
CSSProperties	AiSVGCSSPropertyLocation	Specifies how the CSS properties of the document should be included in the exported file
DocumentEncoding	AiSVGDocumentEncoding	Specifies how the text in the document should be encoded
DTD	AiSVGDTDVersion	The SVG version to which the file should conform. Default: aiSVG1_1
EmbedRasterImages	Boolean	If true, all raster images contained in the document should be embedded in the saved file
FontSubsetting	AiSVGFontSubsetting	Specifies which font glyphs should be included in the export file
FontType	AiSVGFontType	The type of font to included in the exported file. Default: aiCEFFont
IncludeFileInfo	Boolean	If true, the XMP (Extensible Metadata Platform) metadata should be included in the output file. Default: false
IncludeVariablesAndDatasets	Boolean	If true, Variables and Datasets should be included. Default: false
OptimizeForSVGViewer	Boolean	If true, the Adobe namespace should be included. Default: false
PreserveEditability	Boolean	If true, preserve Illustrator editing capability when exporting the document. Default: false

Property	Value type	What it is
Slices	Boolean	If true, preserve slice data in exported document. Default: false
SVGAutoKerning	Boolean	If true, SVG automatic kerning is allowed in the file. Default: false
SVGTextOnPath	Boolean	If true, the SVG text-on-path construct is allowed in the file. Default: false

➤ Exporting to SVG

This example exports the current document as a SVG file.

```
Dim appRef As New Illustrator.Application
Dim svgExportOptions As New Illustrator.ExportOptionsSVG
Dim docRef As Illustrator.Document
If appRef.Documents.Count > 0 Then
  svgExportOptions.EmbedRasterImages = True
  svgExportOptions.FontSubsettings = aiAllGlyphs
  Set docRef = appRef.ActiveDocument
  docRef.Export "C:\temp\AiExport.svg", aiSVG, svgExportOptions
End If
```

Gradient

A gradient definition contained in a document. A script can create new gradients.

Gradient properties

Property	Value type	What it is
Application	Application object	Read-only. The Illustrator Application object.
GradientStops	GradientStops collection object	Read-only. The gradient stops contained in this gradient.
Name	String	The gradient's name.
Parent	Document object	Read-only. The document that contains this gradient.
Type	AiGradientType	The kind of the gradient, either radial or linear.

Gradient methods

Method	Returns	What it does
Delete ()	Nothing	Deletes the object.

➤ Creating and applying a gradient

This example shows how you can create a new gradient and apply it as a fill pattern to the frontmost PathItem.

```
Dim appRef As New Illustrator.Application
Dim frontDocument As Illustrator.Document
Dim newGradient As Illustrator.Gradient
Dim locationSpecification As Illustrator.GradientStop
Set frontDocument = appRef.ActiveDocument

'Create a color for both ends of the gradient
Dim startColor As New Illustrator.RGBColor
Dim endColor As New Illustrator.RGBColor

startColor.Red = 0
startColor.Green = 100
startColor.Blue = 255
endColor.Red = 220
endColor.Green = 0
endColor.Blue = 100

'Create a new gradient
'A new gradient always have 2 stops
Set newGradient = frontDocument.Gradients.Add
newGradient.Name = "Gradient created from script"
newGradient.Type = aiLinearGradient

'Modify the first gradient stop.
```

```
Set locationSpecification = newGradient.GradientStops(1)
locationSpecification.RampPoint = 30
locationSpecification.MidPoint = 60
locationSpecification.Color = startColor

'Modify the last gradient stop.
'The MidPoint for the last gradient stop is ignored
Set locationSpecification = newGradient.GradientStops(2)
locationSpecification.RampPoint = 80
locationSpecification.Color = endColor

'construct an Illustrator.GradientColor object referring to the
'newly created gradient
Dim ColorOfGradient As New Illustrator.GradientColor
ColorOfGradient.Gradient = newGradient

'now get the frontmost path item and
'apply the new gradient as its fill
Dim topPath As Illustrator.PathItem
Set topPath = frontDocument.PathItems(1)
topPath.Filled = True
topPath.fillColor = ColorOfGradient
```

GradientColor

A gradient color specification, used in conjunction with the `Gradient` property of the `Color` specification.

A script can create a `GradientColor` using a reference to an existing gradient in the application. If no existing gradient object is referenced, a default gradient is supplied. An origin is used to specify the center point of the gradient in this specific gradient color. Single values are used to specify the gradient vector angles and lengths. A matrix may be specified to further transform the gradient color.

GradientColor properties

Property	Value type	What it is
`Angle`	`Double`	The gradient vector angle in degrees.
`Application`	`Application object`	Read-only. The Illustrator `Application` object.
`Gradient`	`Gradient object`	Reference to the object defining the gradient.
`HiliteAngle`	`Double`	The gradient hilite vector angle in degrees.
`HiliteLength`	`Double`	The gradient hilite vector length.
`Length`	`Double`	The gradient vector length.
`Matrix`	`Matrix object`	An additional transformation matrix to manipulate the gradient path.
`Origin`	`Variant Array of 2 Doubles`	The gradient vector origin.

➤ Changing a gradient color

The following script obtains the gradient called “Black, White Radial” from the current document and changes the color of the first gradient stop. The Gradient “Black, White Radial” is one of the default gradients that appear when you create a new Illustrator document.

```
Dim appRef As New Illustrator.Application
Dim frontDocument As Illustrator.Document
Dim firstGradient As Illustrator.Gradient
Set frontDocument = appRef.ActiveDocument

'Get a reference to the gradient that you want to delete
Set firstGradient = frontDocument.Gradients("White, Black Radial")

'Create the new color
Dim startColor As New Illustrator.RGBColor
startColor.Red = 255
startColor.Green = 238
startColor.Blue = 98

firstGradient.GradientStops(1).Color = startColor
```

Gradients

A collection of `Gradient` objects in a document.

Gradients properties

Property	Value type	What it is
`Application`	`Application object`	Read-only. The Illustrator `Application` object.
`Count`	`Long`	Read-only. The number of objects in the collection.
`Parent`	`Document object`	Read-only. The parent document of this object.

Gradients methods

Method	Returns	What it does
`Add` `()`	`Gradient object`	Creates a new object
`Index` `(item as` **`Gradient`**`)`	`Long`	Returns the index position of the object within the collection.
`Item` `(itemKey)`	`Gradient object`	Returns an object reference to the object identified by `itemKey` (name or index).
`RemoveAll` `()`	`Nothing`	Deletes all objects in this collection.

➤ Removing a gradient

This example illustrates how you can remove a gradient from a document.

```
Dim appRef As New Illustrator.Application
Dim frontDocument As Illustrator.Document
Dim gradientToDelete As Illustrator.Gradient
Set frontDocument = appRef.ActiveDocument

'Get a reference to the gradient that you want to delete
Set gradientToDelete = frontDocument.Gradients(1)

'Now delete the gradient using the collection
frontDocument.Gradients.Remove gradientToDelete
```

GradientStop

A gradient stop definition contained in a specific gradient. Represents a point on a gradient that specifies a color change. For an example, see GradientColor.

GradientStop properties

Property	Value type	What it is
Application	Application object	Read-only. The Illustrator Application object.
Color	Color object	The color linked to this gradient stop.
Midpoint	Double	The distance between two GradientStops, in percentage. Range: 13.0 to 87.0
Parent	Document object	Read-only. The document that contains this gradient stop.
RampPoint	Double	The location of the color in the blend. Range: 0.0 to 100.0, where 100.0 is 100%

GradientStop methods

Method	Returns	What it does
Delete ()	Nothing	Deletes the object.

GradientStops

A collection of `GradientStop` objects in a specific gradient.

GradientStops properties

Property	Value type	What it is
Application	`Application object`	Read-only. The Illustrator `Application` object.
Count	`Long`	Read-only. The number of objects in the collection.
Parent	`Document object`	Read-only. The document that contains this object.

GradientStops methods

Method	Returns	What it does
Add ()	`GradientStop object`	Creates a new object
Index (item as **GradientStop**)	`Long`	Returns the index position of the object within the collection.
Item (itemKey)	`GradientStops object`	Returns an object reference to the object identified by `itemKey` (name or index).
RemoveAll ()	`Nothing`	Deletes all objects in this collection.

➤ Adding a new gradient stop

This example illustrates how to add a new gradient stop to an existing gradient.

```
Dim appRef As New Illustrator.Application
Dim frontDocument As Illustrator.Document
Dim changeGradient As Illustrator.Gradient
Dim lastStop As Illustrator.GradientStop
Dim newStop As Illustrator.GradientStop
Set frontDocument = appRef.ActiveDocument

'Get a reference to the gradient that you want to change
Set changeGradient = frontDocument.Gradients(1)

'Get a reference to the gradient stop that is the last one
'before you add a new gradient stop
Dim originalCount As Integer
originalCount = changeGradient.GradientStops.Count
Set lastStop = changeGradient.GradientStops(originalCount)

'add the new gradient stop
Set newStop = changeGradient.GradientStops.Add

'Set the values of the new gradient stop. move the original
'last gradient stop a bit to the left, and
```

```
'insert the new gradient stop at the old gradient stop's position
newStop.RampPoint = lastStop.RampPoint
lastStop.RampPoint = lastStop.RampPoint - 10

'Create a new color to apply to the newly created gradient stop
'choose a Gray tint value of 70%
Dim newStopColor As New Illustrator.GrayColor
newStopColor.Gray = 70
newStop.Color = newStopColor
```

GraphicStyle

Each graphic style defines a set of appearance attributes that you can apply to `PageItem` objects. Graphic styles are contained in documents. Scripts cannot create new graphic styles.

GraphicStyle properties

Property	Value type	What it is
`Application`	`Application object`	Read-only. The Illustrator `Application` object.
`Name`	`String`	The `GraphicStyle` name.
`Parent`	`Document object`	Read-only. The document that contains this object.

GraphicStyle methods

Method	Returns	What it does
`ApplyTo` (`artItem as` **`PageItem`**)	`Nothing`	Applies the `GraphicStyle` to a specific art item.
`Delete` ()	`Nothing`	Deletes the object.

➤ Applying a graphic style

This example duplicates and groups the current selection, applying the second graphic style in the document to the items in the group.

```
Dim appRef As New Illustrator.Application
Dim newGroup As Illustrator.GroupItem
Dim dupItem As Object
Dim artItem As Object
Dim i As Integer
Dim endIndex As Integer

If appRef.Documents.Count > 0 Then
  If Not IsEmpty(appRef.ActiveDocument.selection) Then
    endIndex = UBound(appRef.ActiveDocument.selection)
    Set newGroup = appRef.ActiveDocument.GroupItems.Add
    For i = 0 To endIndex
      Set artItem = appRef.ActiveDocument.selection(i)
      Set dupItem = artItem.Duplicate(newGroup, aiPlaceAtEnd)
      appRef.ActiveDocument.GraphicStyles(2).ApplyTo dupItem
    Next
  End If
End If
```

GraphicStyles

A collection of graphic styles in a document.

GraphicStyles properties

Property	Value type	What it is
Application	Application object	Read-only. The Illustrator Application object.
Count	Long	Read-only. The number of objects in the collection.
Parent	Document object	Read-only. The document that contains this object.

GraphicStyles methods

Method	Returns	What it does
Index (item as **GraphicStyle**)	Long	Returns the index position of the object within the collection.
Item (itemKey)	GraphicStyle object	Returns an object reference to the object identified by itemKey (name or index).
RemoveAll ()	Nothing	Deletes all objects in this collection.

➤ Counting graphic styles

This script displays the total number of available graphic styles in the current document.

```
Dim appRef As New Illustrator.Application
Dim numStyles As Integer

If appRef.Documents.Count > 0 Then
  numStyles = appRef.ActiveDocument.GraphicStyles.Count
  MsgBox ("There are " & numStyles & " graphic styles in document.")
End If
```

GraphItem

A graph artwork item. Scripts cannot create new graph items.

GraphItem properties

Property	Value type	What it is
Application	Application object	Read-only. The Illustrator Application object.
ArtworkKnockout	AiKnockoutState	Is this GraphItem used to create a knockout, and if so, what kind.
BlendingMode	AiBlendModes	The mode used when compositing an object.
ContentVariable	Variable object	The content variable bound to this GraphItem. It is not necessary to set the type of the ContentVariable before binding. Illustrator automatically sets the type to AiGraph.
ControlBounds	Variant Array of 4 Doubles	Read-only. The bounds of the object including stroke width and controls.
Editable	Boolean	Read-only. If true, the GraphItem is editable.
GeometricBounds	Variant Array of 4 Doubles	Read-only. The bounds of the GraphItem excluding stroke width.
Height	Double	The height of the GraphItem, calculated on the GeometricBounds.
Hidden	Boolean	If true, this GraphItem is hidden.
IsIsolated	Boolean	If true, this GraphItem is isolated.
Layer	Layer object	Read-only. The Layer to which this GraphItem belongs.
Left	Double	The left position of the GraphItem.
Locked	Boolean	If true, this GraphItem is locked.
Name	String	The name of this GraphItem.
Opacity	Double	The opacity of the GraphItem. Range: 0.0 to 100.0
Parent	Layer object or GroupItem object	Read-only. The parent of this GraphItem.
Position	Variant Array of 2 Doubles	The position of the top left corner of the GraphItem.
Selected	Boolean	If true, this GraphItem is selected.
Sliced	Boolean	If true, this GraphItem is sliced. Default: false

Property	Value type	What it is
Tags	Tags object	Read-only. The collection of Tags contained in this GraphItem.
Top	Double	The top position of the GraphItem.
URL	String	The value of the Adobe URL tag assigned to this GraphItem.
VisibilityVariable	Variable	The visibility variable bound to this GraphItem. It is not necessary to set the type of the VisibilityVariable before binding. Illustrator automatically sets the type to AiVisibility.
VisibleBounds	Variant Array of 4 Doubles	Read-only. The visible bounds of the GraphItem including stroke width.
Width	Double	The width of the GraphItem, based on the GeometricBounds.
WrapInside	Boolean	If true, the text frame object should be wrapped inside this object.
WrapOffset	Double	The offset to use when wrapping text around this object.
Wrapped	Boolean	If true, wrap text frame objects around this object (text frame must be above the object).
ZOrderPosition	Long	Read-only. The position of this GraphItem within the stacking order of the GroupItem or Layer (Parent) that contains the GraphItem.

GraphItem methods

Method	Returns	What it does
Copy ()	Nothing	Copies the GraphItem to the clipboard; the associated document must be the frontmost document.
Cut ()	Nothing	Cuts the GraphItem to the clipboard; the associated document must be the frontmost document.
Delete ()	Nothing	Deletes the object.
Duplicate ([relativeObject as **Object**], [insertionLocation as AiElementPlacement])	Object	Duplicates the art item, optionally specifying the location and position for the copy.
Move (relativeObject as **Object**, insertionLocation as AiElementPlacement)	Nothing	Moves the art item, specifying the new location and position.

Method	Returns	What it does
Resize (scaleX as **Double**, scaleY as **Double**, [, changePositions as **Boolean**] [, changeFillPatterns as **Boolean**] [, changeFillGradients as **Boolean**] [, changeStrokePattern as **Boolean**] [, changeLineWidths as **Double**] [,scaleAbout as AiTransformation])	Nothing	Scales the GraphItem where scaleX is the horizontal scaling factor and scaleY is the vertical scaling factor. 100.0 = 100%
Rotate (angle as **Double**, [, changePositions as **Boolean**] [, changeFillPatterns as **Boolean**] [, changeFillGradients as **Boolean**] [, changeStrokePattern as **Boolean**] [, rotateAbout as AiTransformation])	Nothing	Rotates the GraphItem relative to the current rotation; counter-clockwise if the Angle value is positive, clockwise if the value is negative.
Transform (transformationMatrix as **Matrix**, [, changePositions as **Boolean**] [, changeFillPatterns as **Boolean**] [, changeFillGradients as **Boolean**] [, changeStrokePattern as **Boolean**] [, changeLineWidths as **Double**] [, transformAbout as AiTransformation])	Nothing	Transforms the GraphItem by applying a transformation matrix.
Translate ([deltaX as **Double**] [, deltaY as **Double**] [, transformObjects as **Boolean**] [, transformFillPatterns as **Boolean**] [, transformFillGradients as **Boolean**] [, transformStrokePatterns as **Boolean**])	Nothing	Repositions the GraphItem relative to the current position, where deltaX is the horizontal offset and deltaY is the vertical offset.
ZOrder (zOrderCmd as AiZOrderMethod)	Nothing	Arranges the GraphItem's position in the stacking order of the group or layer (Parent) of this object.

➤ Rotating graph items

```
' Rotate each GraphItem in the current document 90 degrees.

Dim appRef As New Illustrator.Application
Dim ok As Boolean
ok = True

' verify a document with graph item is open
If (appRef.Documents.Count < 1) Then
  ok = False
ElseIf (appRef.ActiveDocument.GraphItems.Count < 1) Then
  ok = False
End If
If (ok = False) Then
  MsgBox "Open a document with graph items before running this sample."
  Exit Sub
End If
```

```
' Rotate each graph item 90 degrees
Dim graphRef As Illustrator.graphItem
For Each graphRef In appRef.ActiveDocument.GraphItems
  graphRef.Rotate 90
Next graphRef
appRef.Redraw
```

GraphItems

A collection of `GraphItem` objects.

GraphItems properties

Property	Value type	What it is
Application	`Application object`	Read-only. The Illustrator `Application` object.
Count	`Long`	Read-only. The number of objects in the collection.
Parent	`Document object`	Read-only. The document that contains this object.

GraphItems methods

Method	Returns	What it does
Index (item as **GraphItem**)	`Long`	Returns the index position of the object within the collection.
Item (itemKey)	`GraphItem object`	Returns an object reference to the object identified by `itemKey` (name or index).
RemoveAll ()	`Nothing`	Deletes all objects in this collection.

GrayColor

A grayscale color specification used to apply a gray color to a layer or art item.

GrayColor properties

Property	Value type	What it is
Application	Application object	Read-only. The Illustrator Application object.
Gray	Double	The tint of the gray. Range: 0.0 to 100.0, where 0.0 is black and 100.0 is white.

➤ Changing word color to gray

This example illustrates how to change the color of the first word in the active document to a shade of gray.

```
Dim appRef As New Illustrator.Application
Dim text As Illustrator.TextRange
Dim firstWord As Illustrator.TextRange
Dim textColor As New Illustrator.GrayColor

' Get a reference to the first word in the active document
Set text = appRef.ActiveDocument.TextFrames(1).TextRange
Set firstWord = text.Words(1)

' Create the new color
textColor.Gray = 45

firstWord.CharacterAttributes.fillColor = textColor
```

GroupItem

A grouped set of art items. Group items can contain all of the same page items that a layer can contain, including other nested groups.

Paths contained within a group or compound path in a document are returned as individual paths when a script asks for the paths contained in the document. However, paths contained in a group or compound path are not returned when a script asks for the paths in a layer which contains the group or compound path.

GroupItem properties

Property	Value type	What it is
Application	Application object	Read-only. The Illustrator Application object.
ArtworkKnockout	AiKnockoutState	Is this object used to create a knockout, and if so, what kind.
BlendingMode	AiBlendModes	The mode used when compositing an object.
Clipped	Boolean	If true, the GroupItem is clipped to its first PathItem.
CompoundPathItems	CompoundPathItems collection object	Read-only. The CompoundPathItems contained in this GroupItem.
ControlBounds	Variant Array of 4 Doubles	Read-only. The bounds of the object including stroke width and controls.
Editable	Boolean	Read-only. If true, this GroupItem is editable.
GeometricBounds	Variant Array of 4 Doubles	Read-only. The bounds of the object excluding stroke width.
GraphItems	GraphItems collection object	Read-only. The raster items contained in this GroupItem.
GroupItems	GroupItems collection object	Read-only. The GroupItems contained in this GroupItem.
Height	Double	The height of the GroupItem, based on the GeometricBounds.
Hidden	Boolean	If true, this GroupItem is hidden.
IsIsolated	Boolean	If true, this object is isolated.
Layer	Layer object	Read-only. The layer to which this GroupItem belongs.
Left	Double	The left position of the GroupItem.
LegacyTextItems	LegacyTextItems	Read-only. The text frame items in this story.
Locked	Boolean	If true, this GroupItem is locked.

Property	Value type	What it is
MeshItems	MeshItems collection object	Read-only. The MeshItems contained in this GroupItem.
Name	String	The name of this GroupItem.
Opacity	Double	The opacity of the object. Range: 0.0 to 100.0
PageItems	PageItems collection object	Read-only. The PageItems contained in this GroupItem.
Parent	Document object	Read-only. The document that contains this GroupItem.
PathItems	PathItems collection object	Read-only. The PathItems contained in this GroupItem.
PlacedItems	PlacedItems collection object	Read-only. The PlacedItems contained in this GroupItem.
PluginItems	PluginItems collection object	Read-only. The PluginItems contained in this GroupItem.
Position	Variant Array of 2 Doubles	The position of the top left corner of the GroupItem.
RasterItems	RasterItem	Read-only. The RasterItems contained in this GroupItem.
Selected	Boolean	If true, this GroupItem is selected.
Sliced	Boolean	If true, this GroupItem is sliced. Default: false
SymbolItems	SymbolItems object	Read-only. The SymbolItems contained in this GroupItem.
Tags	Tags collection object	Read-only. The tags contained in this GroupItem.
TextFrames	TextFrames collection object	Read-only. The TextFrame objects contained in this GroupItem.
Top	Double	The top position of the GroupItem.
URL	String	The value of the Adobe URL tag assigned to this GroupItem.
VisibilityVariable	Variable	The Variable bound to this GroupItem.
VisibleBounds	Variant Array of 4 Doubles	Read-only. The visible bounds of the GroupItem including stroke width.
Width	Double	The width of the GroupItem, based on the GeometricBounds.
WrapInside	Boolean	If true, the text frame object should be wrapped inside this object.

Property	Value type	What it is
WrapOffset	Double	The offset to use when wrapping text around this object.
Wrapped	Boolean	If true, wrap text frame objects around this object (text frame must be above the object).
ZOrderPosition	Long	Read-only. The position of this group item within the stacking order of the group or layer (Parent) that contains this group item.

GroupItem methods

Method	Returns	What it does
Copy ()	Nothing	Copies the GroupItem to the clipboard; the associated document must be the frontmost document.
Cut ()	Nothing	Cuts the GroupItem to the clipboard; the associated document must be the frontmost document.
Delete ()	Nothing	Deletes the object.
Duplicate ([relativeObject as **Object**], [insertionLocation as AiElementPlacement])	Object	Duplicates the art item, optionally specifying the location and position for the copy.
Move (relativeObject as **Object**, insertionLocation as AiElementPlacement)	Nothing	Moves the art item, specifying the new location and position.
Paste ()	Nothing	Inserts the contents of the clipboard at the beginning of the GroupItem. You may only paste into a group that is contained in the active document.
Resize (scaleX as **Double**, scaleY as **Double**, [, changePositions as **Boolean**] [, changeFillPatterns as **Boolean**] [, changeFillGradients as **Boolean**] [, changeStrokePattern as **Boolean**] [, changeLineWidths as **Double**] [, scaleAbout as AiTransformation])	Nothing	Scales the art item where scaleX is the horizontal scaling factor and scaleY is the vertical scaling factor; 100.0 = 100%
Rotate (angle as **Double**, [, changePositions as **Boolean**] [, changeFillPatterns as **Boolean**] [, changeFillGradients as **Boolean**] [, changeStrokePattern as **Boolean**] [, rotateAbout as AiTransformation])	Nothing	Rotates the art item relative to the current rotation. The object is rotated counter-clockwise if the Angle value is positive, clockwise if the value is negative.

Method	Returns	What it does
Transform (transformationMatrix as **Matrix** [, changePositions as **Boolean**] [, changeFillPatterns as **Boolean**] [, changeFillGradients as **Boolean**] [, changeStrokePattern as **Boolean**] [, changeLineWidths as **Double**] [, transformAbout as AiTransformation])	Nothing	Transforms the art item by applying a transformation matrix.
Translate ([deltaX as **Double**] [, deltaY as **Double**] [, transformObjects as **Boolean**] [, transformFillPatterns as **Boolean**] [, transformFillGradients as **Boolean**] [, transformStrokePatterns as **Boolean**])	Nothing	Repositions the art item relative to the current position, where deltaX is the horizontal offset and deltaY is the vertical offset.
ZOrder (zOrderCmd as AiZOrderMethod)	Nothing	Arranges the art item's position in the stacking order of the group or layer (Parent) of this object.

➤ Creating a group

It is easy to modify all of the objects contained in a group. This example demonstrates how to simplify your operations on multiple objects by creating a group to contain them.

```
Dim appRef As New Illustrator.Application
Dim triangleGroup As Illustrator.GroupItem

'Create a new group in the active document.
'This will be the group that holds the new triangle art
Set triangleGroup = appRef.ActiveDocument.GroupItems.Add

'Create a triangle and add text inside the group
Dim trianglePath As Illustrator.PathItem
Dim captionText As Illustrator.TextFrame

Set trianglePath = triangleGroup.PathItems.Add
trianglePath.SetEntirePath Array(Array(100, 100), Array(300, 100), _
    Array(200, Math.Tan(1.0471975) * 100 + 100))

Set captionText = triangleGroup.TextFrames.Add
captionText.Position = Array(100, 100)
captionText.Contents = "A triangle"
```

GroupItems

A collection of `GroupItem` objects.

GroupItems properties

Property	Value type	What it is
Application	`Application object`	Read-only. The Illustrator `Application` object.
Count	`Long`	Read-only. The number of objects in the collection.
Parent	`Document Object`	Read-only. The document that contains this object.

GroupItems methods

Method	Returns	What it does
Add ()	`GroupItem object`	Creates a new object
CreateFromFile (imagefile as **String**)	`GroupItem object`	Creates a `GroupItem` from a vector graphics file.
Index (item as **GroupItem**)	`Long`	Returns the index position of the object within the collection.
Item (itemkey)	`GroupItem`	Returns an object reference to the object identified by `itemkey` (name or index).
RemoveAll ()	`Nothing`	Deletes all objects in this collection.

➤ Importing a PDF using GroupItems

The following script shows how you can import an Adobe PDF document using the `CreateFromFile` function. Before running this script you have to create a one-page PDF file and put it in the location `C:\testPDF.pdf`.

```
Dim appRef As New Illustrator.Application
Dim importedGroup As Illustrator.GroupItem

Set importedGroup = appRef.ActiveDocument.GroupItems.CreateFromFile _
    ("C:\temp\Sample.jpg")
```

IllustratorSaveOptions

Options that can be supplied when saving a document as an Illustrator file with the `document SaveAs` method.

IllustratorSaveOptions properties

Property	Value type	What it is
Application	`Application object`	Read-only. The Illustrator `Application` object.
Compatibility	`AiCompatibility`	Version of the Illustrator file format to create.
Compressed	`Boolean`	If `true`, the saved file should be compressed. Default: `true` (version 10 or later).
EmbedICCProfile	`Boolean`	If `true`, an ICC profile should be embedded in the saved file.
EmbedLinkedFiles	`Boolean`	If `true`, linked image files should be included in the saved document (only valid for `SaveOptions` that specify an Illustrator compatibility of version 7 or later).
FlattenOutput	`AiOutputFlattening`	Specifies how transparency should be flattened for file formats older than Illustrator 9.
FontSubsetThreshold	`Double`	Include a subset of fonts when less than this percentage of characters is used in the document (valid for Illustrator 9 or newer file format).
Overprint	`AiPDFOverprint`	The overprinting style. Default: `PreservePDFOverprint`
PDFCompatible	`Boolean`	If `true`, save as a PDF compatible file (version 10 or later). Default: `true`

➤ Saving to Illustrator 8 format

This example illustrates how to save the frontmost document as Illustrator 8 format. Because the document is saved as a version earlier than 9, the example specifies to convert opacity by breaking paths up in to sub-path to preserve the appearance of the illustration.

```
Dim appRef As New Illustrator.Application
Dim saveOptions As New Illustrator.IllustratorSaveOptions
saveOptions.Compatibility = aiIllustrator8
saveOptions.FlattenOutput = aiPreserveAppearance
appRef.Documents(1).SaveAs "C:\temp\Ai8SaveAs.ai", saveOptions
```

Ink

Provides information about the ink name and related information.

Ink properties

Property	Value type	What it is
Application	Application object	Read-only. The Illustrator Application object.
InkInfo	InkInfo object	The ink information.
Name	String	The ink's name.

InkInfo

Specifies ink properties.

InkInfo properties

Property	Value type	What it is
Angle	Double	The ink's screen angle in degrees.
Application	Application	Read-only. The Illustrator Application object.
CustomColor	Object	The color of the custom ink.
Density	Double	The neutral density. Minimum: 0.0
DotShape	String	The dot shape name.
Frequency	Double	The ink's frequency. Minimum: 0.0
Kind	AiInkType	The ink type.
PrintingStatus	AiInkPrintStatus	The ink printing status.
Trapping	AiTrappingType	The trapping type.
TrappingOrder	Long	The order of trapping for the ink. Minimum: 1

➤ List inks in a document

```
' Assemble and display a string of the inks in this document

Dim appRef As New Illustrator.Application
Dim docRef As Illustrator.Document

' create a new document and simple text frame
Set docRef = appRef.Documents.Add()
Dim textRef As Illustrator.TextFrame
Set textRef = docRef.TextFrames.Add()
textRef.Top = docRef.Height - 100
textRef.Left = 100
```

```
' Collect data from each Ink and InkInfo object
Dim sInks As String
Dim inkRef
For Each inkRef In appRef.ActiveDocument.InkList
  sInks = sInks & inkRef.Name
  sInks = sInks & vbCrLf & vbTab
  sInks = sInks & "Frequency = " & inkRef.InkInfo.Frequency
  sInks = sInks & vbCrLf & vbTab
  sInks = sInks & "Density = " & inkRef.InkInfo.Density
  sInks = sInks & vbCrLf
Next inkRef
textRef.Contents = sInks
appRef.Redraw
```

InsertionPoint

A location between characters, used to insert new text objects. This is a `TextRange` object in which `characterOffset` indicates the location of the insertion point and `length` is 0. This subclass does not define any additional properties.

InsertionPoints

A collection of `InsertionPoint` objects.

InsertionPoints properties

Property	Value type	What it is
`Application`	`Application`	Read-only. The Illustrator `Application` object.
`Count`	`Long`	Read-only. Number of elements in the collection.
`Parent`	`TextFrame object`	Read-only. The object's container.

InsertionPoints methods

Method	Returns	What it does
`Index` `(itemPtr as` **`TextRange`**`)`	`Long`	Returns the index position of the object within the collection.
`Item` `(itemKey)`	`InsertionPoint`	Gets an element from the collection by `itemKey` (name or index).

➤ **Using insertion point to add spaces**

```
' Create a new document and add a simple text frame
' Use insertion points to add spaces between all characters

Dim appRef As New Illustrator.Application
Dim docRef As Illustrator.Document

' create a new document and simple text frame
Set docRef = appRef.Documents.Add()
Dim textRef As Illustrator.TextFrame
Set textRef = docRef.TextFrames.Add()
textRef.Contents = "Wouldn't you rather be scripting?"
textRef.Top = 400
textRef.Left = 100
textRef.TextRange.CharacterAttributes.Size = 20
appRef.Redraw

MsgBox "Use insertion points to add a space between all letters."
Dim i As Integer
i = 2
Do While (i < textRef.InsertionPoints.Count)
  textRef.InsertionPoints(i).Characters.Add (" ")
```

```
  i = i + 2
Loop
```

LabColor

A color specification in the CIE Lab color space, used where a `color` object is required.

LabColor properties

Property	Value type	What it is
A	`double`	The a (red-green) color value. Range -128.0–128.0. Default: 0.0
B	`double`	The b (yellow-blue) color value. Range -128.0–128.0. Default: 0.0
L	`double`	The l (lightness) color value. Range -128.0–128.0. Default: 0.0
Name	`string`	Read-only. The class name of the referenced object.

Layer

A layer in an Illustrator document. Layers may contain nested layers, which are called sublayers in the user interface.

`Layer` object contains all of the page items in the layer as elements. Your script can access `PageItems` as elements of either the `Layer` object or as elements of the `Document` object. Only those page items in that layer can be accessed through the `Layer` object. To see page items in different layers, access them through the document.

Layer properties

Property	Value type	What it is
Application	Application	Read-only. The Illustrator `Application` object.
ArtworkKnockout	AiKnockoutState	Is this `Layer` used to create a knockout, and if so, what kind.
BlendingMode	AiBlendModes	The mode used when compositing an object.
Color	RGBColor object	The `Layer`'s selection mark color.
CompoundPathItems	CompoundPathItems collection object	Read-only. The `CompoundPathItems` contained in this layer.
DimPlacedImages	Boolean	If `true`, placed images are to be rendered as dimmed in this layer.
GraphItems	GraphItems collection object	Read-only. The `GraphItems` collection contained in this layer.
GroupItems	GroupItems collection object	Read-only. The `GroupItems` contained in this layer.
HasSelectedArtwork	Boolean	If `true`, an object in this `Layer` has been selected; setting this property to `false` deselects all objects in the layer.
IsIsolated	Boolean	If `true`, this `Layer` is isolated.
Layers	Layers collection object	Read-only. The `Layers` contained in this layer.
LegacyTextItems	LegacyTextItems	Read-only. `LegacyTextItems` contained in this layer.
Locked	Boolean	If `true`, this layer is editable.
MeshItems	MeshItems collection object	Read-only. The `MeshItems` contained in this layer.
Name	String	The name of this layer.
Opacity	Double	The opacity of the layer. Range: 0.0 to 100.0
PageItems	PageItems collection object	Read-only. The `PageItems` contained in this layer.
Parent	Document object or Layer Object	Read-only. The `document` or `Layer` that contains this layer.

Property	Value type	What it is
PathItems	PathItems collection object	Read-only. The PathItems contained in this layer.
PlacedItems	PlacedItems collection object	Read-only. The PlacedItems contained in this layer.
PluginItems	PluginItems collection object	Read-only. The PluginItems contained in this layer.
Preview	Boolean	If true, this layer should be displayed using preview mode.
Printable	Boolean	If true, this layer should be printed when printing the document.
RasterItems	RasterItems collection object	Read-only. The RasterItems contained in this layer.
Sliced	Boolean	If true, this layer is sliced.
SymbolItems	SymbolItems collection object	Read-only. The SymbolItems contained in this layer.
TextFrames	TextFrameItems collection object	Read-only. The TextFrames contained in this layer.
Visible	Boolean	If true, this layer is visible.
ZOrderPosition	Long	Read-only. The position of this layer within the stacking order of Layers in the document

Layer methods

Method	Returns	What it does
Delete ()	Nothing	Deletes the object.
Move (relativeObject as **Object**, insertionLocation as AiElementPlacement)	Nothing	Moves the art item, specifying the new location and position. Does not change the position of the object on the art board, but changes the order in which Illustrator draws the objects, and the containment hierarchy. Places the object in the specified container, behind all other such objects.
Paste ()	Nothing	Pastes the contents of the clipboard into the Layer; if the associated document is the frontmost then all pasted objects remain selected after the paste.
ZOrder (zOrderCmd as AiZOrderMethod)	Nothing	Arranges the Layer's position in the stacking order of Layers in this document

➤ Count layers in a document

```
Dim appRef As New Illustrator.Application
Dim frontDocument As Illustrator.Document
Dim bottomLayer As Illustrator.Layer
Dim countOfLayers As Integer

' Get a reference to the layers, and obtain the total number
Set frontDocument = appRef.ActiveDocument
countOfLayers = frontDocument.Layers.Count

If (frontDocument.Layers.Count < countOfLayers) Then
  MsgBox "The frontmost application only has 1 layer"
  Exit Sub
End If
```

Layers

A collection of `Layer` objects.

Layers properties

Property	Value type	What it is
`Application`	`Application`	Read-only. The Illustrator `Application` object.
`Count`	`Long`	Read-only. The number of objects in the collection.
`Parent`	`object`	Read-only. The object that contains this `Layer` (can be another `Layer`).

Layers methods

Method	Returns	What it does
`Add` `()`	`Layer object`	Creates a new object.
`Index` `(item as` **`Layer`**`)`	`Long`	Returns the index position of the object within the collection.
`RemoveAll` `()`	`Nothing`	Deletes all objects in this collection.

➤ Deleting layers

This example illustrates how to delete all layers whose name starts with the word “Temporary” in all open documents.

```
' Remove temporary layers

Dim appRef As New Illustrator.Application
Dim targetDocument As Illustrator.Document
Dim targetLayer As Illustrator.Layer
Dim countOfLayers As Integer
Dim layerIndex As Integer
Dim layerName As String
Dim searchString As String
searchString = "Temporary"

' Loop through all open documents
For Each targetDocument In appRef.Documents
  countOfLayers = targetDocument.Layers.Count
' (Go through layers from the back to avoid changing the
'  index of unvisited layers when we remove a layer)
  For layerIndex = countOfLayers To 1 Step -1
    Set targetLayer = targetDocument.Layers(layerIndex)
    layerName = targetLayer.Name
    If (InStr(layerName, searchString) = 1) Then
      targetDocument.Layers.Remove targetLayer
    End If
  Next layerIndex
```

```
Next targetDocument
```

LegacyTextItem

Unconverted legacy text items in documents from versions prior to Illustrator CS2.

LegacyTextItem properties

Property	Value type	What it is
Application	Application object	Read-only. The Illustrator Application object.
ArtworkKnockout	AiKnockoutState	Is this object used to create a knockout, and if so, what kind.
BlendingMode	AiBlendModes	The mode used when compositing an object.
ControlBounds	Variant Array of 4 Doubles	Read-only. The bounds of the object including stroke width and controls.
Converted	Boolean	Read-only. If true, the legacy text has been updated to a native text frame item.
Editable	Boolean	Read-only. If true, this item is editable.
GeometricBounds	Variant Array of 4 Doubles	Read-only. The bounds of the object excluding stroke width.
Height	Double	The height of the LegacyTextItem excluding stroke width, based on the GeometricBounds.
Hidden	Boolean	If true, this object is hidden.
IsIsolated	Boolean	If true, this object is isolated.
Layer	Layer object	Read-only. The layer to which this LegacyTextItem belongs.
Left	Double	The left position of the LegacyTextItem.
Locked	Boolean	If true, this LegacyTextItem is locked.
Name	String	The name of this LegacyTextItem.
Opacity	Double	The opacity of the object. Range: 0.0 to 100.0
Parent	object	Read-only. The object that contains this LegacyTextItem.
Position	Variant Array of 2 Doubles	The position of the top left corner of the LegacyTextItem excluding stroke width.
Selected	Boolean	If true, this object is selected.
Sliced	Boolean	If true, this LegacyTextItem is sliced.
Tags	Tags collection object	Read-only. The tags contained in this LegacyTextItem.
Top	Double	The top position of this LegacyTextItem.

Property	Value type	What it is
URL	String	The value of the Adobe URL tag assigned to this LegacyTextItem.
VisibilityVariable	Variable	The Variable bound to this LegacyTextItem.
VisibleBounds	Variant Array of 4 Doubles	Read-only. The visible bounds of the LegacyTextItem including stroke width.
Width	Double	The width of the LegacyTextItem excluding stroke width, based on the GeometricBounds.
WrapInside	Boolean	If true, the text frame object should be wrapped inside this object.
WrapOffset	Double	The offset to use when wrapping text around this object.
Wrapped	Boolean	If true, wrap text frame objects around this object (text frame must be above the object).
ZOrderPosition	Long	Read-only. The position of this art item within the stacking order of the group or layer (Parent) that contains the art item.

LegacyTextItem methods

Method	Returns	What it does
ConvertToNative ()	Boolean	If true, create text frames from all legacy text items; the original legacy text items are deleted.
Copy ()	Nothing	Copies the legacy text item to the clipboard; the associated document must be the frontmost document.
Cut ()	Nothing	Cuts the legacy text item to the clipboard; the associated document must be the frontmost document.
Duplicate ([relativeObject as **Object**], [insertionLocation as AiElementPlacement])	Object	Duplicates the art item, optionally specifying the location and position for the copy.
Delete ()	Nothing	Deletes the object.
Move (relativeObject as **Object**, insertionLocation as AiElementPlacement)	Nothing	Moves the art item, specifying the new location and position.

Method	Returns	What it does
Resize (scaleX as **Double**, scaleY as **Double** [, changePositions as **Boolean**] [, changeFillPatterns as **Boolean**] [, changeFillGradients as **Boolean**] [, changeStrokePattern as **Boolean**] [, changeLineWidths as **Double**] [, scaleAbout as AiTransformation])	Nothing	Scales the art item where scaleX is the horizontal scaling factor and scaleY is the vertical scaling factor; 100.0 = 100%
Rotate (angle as **Double**, [, changePositions as **Boolean**] [, changeFillPatterns as **Boolean**] [, changeFillGradients as **Boolean**] [, changeStrokePattern as **Boolean**] [, rotateAbout as AiTransformation])	Nothing	Rotates the art item relative to the current rotation. The object is rotated counter-clockwise if the Angle value is positive, clockwise if the value is negative.
Transform (transformationMatrix as **Matrix**, [, changePositions as **Boolean**] [, changeFillPatterns as **Boolean**] [, changeFillGradients as **Boolean**] [, changeStrokePattern as **Boolean**] [, changeLineWidths as **Double**] [, transformAbout as AiTransformation])	Nothing	Transforms the art item by applying a transformation matrix.
Translate ([deltaX as **Double**] [, deltaY as **Double**] [, transformObjects as **Boolean**] [, transformFillPatterns as **Boolean**] [, transformFillGradients as **Boolean**] [, transformStrokePatterns as **Boolean**])	Nothing	Repositions the art item relative to the current position, where deltaX is the horizontal offset and deltaY is the vertical offset.
ZOrder (zOrderCmd as AiZOrderMethod)	Nothing	Arranges the art item's position in the stacking order of the group or layer (Parent) of this object.

LegacyTextItems

A collection of `LegacyText` items.

LegacyTextItems properties

Property	Value type	What it is
Application	Application	Read-only. The Illustrator `Application` object.
Count	Long	Read-only. Number of elements in the collection.
Parent	object	Read-only. The object's container.

LegacyTextItems methods

Method	Returns	What it does
ConvertToNative ()	Boolean	Create text frames from all legacy text items; the original legacy text items are deleted. Returns `true` if successful.
Index (itemPtr as **LegacyTextItem**)	Long	Returns the index position of the object within the collection.
Item (itemKey)	LegacyTextItem	Returns an object reference to the object identified by `itemKey` (name or index).
RemoveAll ()	Nothing	Deletes all objects in this collection.

Lines

A collection of `TextRange` objects corresponding to a line of text.

Lines properties

Property	Value type	What it is
`Application`	`Application`	Read-only. The Illustrator `Application` object.
`Count`	`Long`	Read-only. Number of elements in the collection.
`Parent`	`object`	Read-only. The object's container.

Lines methods

Method	Returns	What it does
`Index` (`itemPtr as` **`TextRange`**)	`Long`	Returns the index position of the object within the collection.
`Item` (`itemKey`)	`TextRange`	Returns an object reference to the object identified by `itemKey` (name or index).
`RemoveAll` ()	`Nothing`	Deletes all objects in this collection.

➤ Counting lines

```
' Create an area text item and add some text
' Display the line count, then resize it and display count again

' Create an area text
Dim appRef As New Illustrator.Application
Dim docRef As Illustrator.Document
Set docRef = appRef.Documents.Add()
Dim itemRef As Illustrator.pathItem
Set itemRef = docRef.PathItems.Rectangle(500, 200, 50, 300)
Dim textRef As Illustrator.TextFrame
Set textRef = docRef.TextFrames.AreaText(itemRef)
textRef.Contents = "Scripting with Illustrator is fun and easy."
appRef.Redraw
MsgBox "There are " & CStr(textRef.Lines.Count) & " lines."
' Change the path's width and display line count
itemRef.Width = 300
appRef.Redraw
MsgBox "Now there are " & CStr(textRef.Lines.Count) & " lines."
```

Matrix

A transformation matrix specification, used to transform the geometry of objects.

This class is used to define a record which contains the component values of an Illustrator transformation matrix. It is used for specifying and retrieving matrix information from an Illustrator document or from PageItems in a document.

Matrices are used in conjunction with the `Transform` method and as a property of a number of objects. A matrix specifies how to transform the geometry of an object. You can generate an original matrix using the application methods `getIdentityMatrix`, `getTranslationMatrix`, `getScaleMatrix`, or `getRotationMatrix`.

A `Matrix` is a record containing the matrix values, not a reference to a matrix object. The matrix commands listed above operate on the values of a matrix record. If a command modifies a matrix, a modified matrix record is returned as the result of the command. The original matrix record pass to the command is not modified.

Matrix properties

Property	Value type	What it is
Application	`Application object`	Read-only. The Illustrator `Application` object.
MValueA	`Double`	Matrix property `a`.
MValueB	`Double`	Matrix property `b`.
MValueC	`Double`	Matrix property `c`.
MValueD	`Double`	Matrix property `d`.
MValueTX	`Double`	Matrix property `tx`.
MValueTY	`Double`	Matrix property `ty`.

➤ Applying transformations with a matrix

If you need to apply multiple transformations to objects it is more efficient to use the matrix suite than to apply the transformations one at a time. The following script demonstrates how to combine multiple matrices together.

```
Dim appRef As New Illustrator.Application
Dim moveMatrix As Illustrator.Matrix
Dim totalMatrix As Illustrator.Matrix

' Move art half an inch to the right and 1.5 inch up on the page
Set moveMatrix = appRef.GetTranslationMatrix(72# * 0.5, 72# * 1.5)
' Add a rotation to the translation -- 10 degrees counterclockwise
Set totalMatrix = appRef.ConcatenateRotationMatrix(moveMatrix, 10)

' Apply the transformation to all art in the document
Dim artItem As Object
For Each artItem In appRef.ActiveDocument.PageItems
  artItem.Transform totalMatrix
Next
```

MeshItem

A gradient mesh art item. Mesh items cannot be created from a script, but can be copied and pasted.

MeshItem properties

Property	Value type	What it is
Application	Application object	Read-only. The Illustrator Application object.
ArtworkKnockout	AiKnockoutState	Is this object used to create a knockout, and if so, what kind.
BlendingMode	AiBlendModes	The mode used when compositing an object.
ControlBounds	Variant Array of 4 Doubles	Read-only. The bounds of the object including stroke width and controls.
Editable	Boolean	Read-only. If true, this MeshItem is editable.
GeometricBounds	Variant Array of 4 Doubles	Read-only. The bounds of the object excluding stroke width.
Height	Double	The height of the MeshItem, based on the GeometricBounds.
Hidden	Boolean	If true, this MeshItem is hidden.
IsIsolated	Boolean	If true, this object is isolated.
Layer	Layer object	Read-only. The Layer to which this MeshItem belongs.
Left	Double	The left position of the MeshItem.
Locked	Boolean	If true, this MeshItem is locked.
Name	String	The name of this MeshItem.
Opacity	Double	The opacity of the object. Range 0.0 to 100.0
Parent	object	Read-only. The object that contains this MeshItem.
Position	Variant Array of 2 Doubles	The position of the top left corner of the MeshItem.
Selected	Boolean	If true, this MeshItem is selected.
Sliced	Boolean	If true, this MeshItem is sliced.
Tags	Tags collection object	Read-only. The tags contained in this MeshItem.
Top	Double	The top position of this MeshItem.
URL	String	The value of the Adobe URL tag assigned to this MeshItem.
VisibilityVariable	Variant	The Variable bound to this MeshItem.
VisibleBounds	Variant Array of 4 Doubles	Read-only. The visible bounds of the MeshItem including stroke width.

Property	Value type	What it is
Width	Double	The width of the MeshItem, based on the GeometricBounds.
WrapInside	Boolean	If true, the text frame object should be wrapped inside this object.
WrapOffset	Double	The offset to use when wrapping text around this object.
Wrapped	Boolean	If true, wrap text frame objects around this object (text frame must be above the object).
ZOrderPosition	Long	Read-only. The position of this art item within the stacking order of the group or layer (Parent) that contains the art item.

MeshItem methods

Method	Returns	What it does
Copy ()	Nothing	Copies the mesh item to the clipboard; the associated document must be the frontmost document.
Cut ()	Nothing	Cuts the mesh item to the clipboard; the associated document must be the frontmost document.
Delete ()	Nothing	Deletes the object.
Duplicate ([relativeObject as **Object**], [insertionLocation as AiElementPlacement])	Object	Duplicates the art item, optionally specifying the location and position for the copy.
Move (relativeObject as **Object**, insertionLocation as AiElementPlacement)	Nothing	Moves the art item, specifying the new location and position.
Resize (scaleX as **Double**, scaleY as **Double**, [, changePositions as **Boolean**] [, changeFillPatterns as **Boolean**] [, changeFillGradients as **Boolean**] [, changeStrokePattern as **Boolean**] [, changeLineWidths as **Double**] [, scaleAbout as AiTransformation])	Nothing	Scales the art item where scaleX is the horizontal scaling factor and scaleY is the vertical scaling factor; 100.0 = 100%
Rotate (angle as **Double**, [, changePositions as **Boolean**] [, changeFillPatterns as **Boolean**] [, changeFillGradients as **Boolean**] [, changeStrokePattern as **Boolean**] [, rotateAbout as AiTransformation])	Nothing	Rotates the art item relative to the current rotation. The object is rotated counter-clockwise if the Angle value is positive, clockwise if the value is negative.

Method	Returns	What it does
Transform (transformationMatrix as **Matrix**, [, changePositions as **Boolean**] [, changeFillPatterns as **Boolean**] [, changeFillGradients as **Boolean**] [, changeStrokePattern as **Boolean**] [, changeLineWidths as **Double**] [, transformAbout as AiTransformation])	Nothing	Transforms the art item by applying a transformation matrix.
Translate ([deltaX as **Double]** [, deltaY as **Double]** [, transformObjects as **Boolean**] [, transformFillPatterns as **Boolean**] [, transformFillGradients as **Boolean**] [, transformStrokePatterns as **Boolean**])	Nothing	Repositions the art item relative to the current position, where deltaX is the horizontal offset and deltaY is the vertical offset.
ZOrder (zOrderCmd as AiZOrderMethod)	Nothing	Arranges the art item's position in the stacking order of the group or layer (Parent) of this object.

➤ Locking mesh items

This script illustrates how to lock all MeshItems in the active document.

```
Dim appRef As New Illustrator.Application
Dim meshItem As Illustrator.meshItem

For Each meshItem In appRef.ActiveDocument.MeshItems
  meshItem.Locked = True
Next
```

MeshItems

A collection of gradient mesh art items.

MeshItems properties

Property	Value type	What it is
Application	Application	Read-only. The Illustrator Application object.
Count	Long	Read-only. Number of elements in the collection.
Parent	object	Read-only. The object's container.

MeshItems methods

Method	Returns	What it does
Index (itemPtr as **MeshItem**)	Long	Returns the index position of the object within the collection.
Item (itemKey)	MeshItem	Returns an object reference to the object identified by itemKey (name or index).
RemoveAll ()	Nothing	Deletes all objects in this collection.

➤ Copying mesh items between documents

The following script illustrates how to copy MeshItems from one document to another. To run this script you need to have two open documents. One document should contain at least one MeshItem, the other document can be empty. Make the empty document the frontmost before running the script.

```
Dim appRef As New Illustrator.Application
Dim sourceDocument As Illustrator.Document
Dim targetDocument As Illustrator.Document
Dim meshItem As Illustrator.meshItem
Dim newMeshItem  As Illustrator.meshItem
Dim targetSelection As Variant
Dim locationOffset As Single

Set targetDocument = appRef.Documents(1)
Set sourceDocument = appRef.Documents(2)
locationOffset = 0
For Each meshItem In sourceDocument.MeshItems
  sourceDocument.Activate
  meshItem.Copy
  targetDocument.Activate
  targetDocument.Paste

'Get a reference to the item that was just copied into the document
  targetSelection = appRef.selection
  If (IsEmpty(targetSelection)) Then
    MsgBox "Copy/Paste failed"
    Exit For
```

```
  End If
  Set newMeshItem = targetSelection(0)

  newMeshItem.Position = Array(100, 40 + locationOffset)
  locationOffset = locationOffset + 50
Next
```

NoColor

Represents the "none" color. Assignment of a reference to a NoColor object to the document's default fill or stroke color, or those of an art item, is equivalent to setting their "Filled" or "Stroked" property to False.

NoColor properties

Property	Value type	What it is
Application	Application	Read-only. The Illustrator Application object.

➤ Removing a fill color

```
' Create 2 overlapping objects with different fill colors.
' Assign top object a fill color of "NoColor"
' to make the bottom object visible.

Dim appRef As New Illustrator.Application
Dim docRef As Illustrator.Document
Set docRef = appRef.Documents.Add()
Dim itemRef1 As Illustrator.PathItem
Dim itemRef2 As Illustrator.PathItem
Set itemRef1 = docRef.PathItems.Rectangle(500, 200, 200, 100)
Set itemRef2 = docRef.PathItems.Rectangle(550, 150, 200, 200)

Dim colorRef As New Illustrator.RGBColor
colorRef.Blue = 255
itemRef1.FillColor = colorRef
colorRef.Red = 255
colorRef.Blue = 0
itemRef2.FillColor = colorRef
appRef.Redraw

' Create a nocolor and assign it to the top object
MsgBox "Change red object to NoColor."
Dim noColorRef As New Illustrator.NoColor
itemRef2.FillColor = noColorRef
appRef.Redraw
```

OpenOptions

Options to use when opening files in Illustrator.

OpenOptions properties

Property	Value type	What it is
Application	Application	Read-only. The Illustrator Application object.
UpdateLegacyText	Boolean	If true, update all text objects for documents saved with legacy text by Illustrator versions previous to CS. Default: false

➤ Opening a file and updating legacy text

This example opens a file from Illustrator 10 or older, and uses the OpenOptions object to automatically update any legacy text in the file. This script assumes the file C:\temp\LegacyText.ai exists.

```
Dim appRef As New Illustrator.Application
Dim docRef As Illustrator.Document

' Open a file with legacy text
Dim optionsRef As New Illustrator.OpenOptions
optionsRef.UpdateLegacyText = True
Set docRef = appRef.Open("C:\temp\LegacyText.ai",_
   aiDocumentRGBColor, optionsRef)
```

PageItems

A collection of `PageItem` objects. Provides complete access to all the art items in an Illustrator document in the following classes:

```
CompoundPathItem
GraphItem
GroupItem
MeshItem
PathItem
PlacedItem
RasterItem
SymbolItem
TextFrame
```

You can reference page items through the `PageItems` property in a `Document`, `Layer`, or `Group`. When you access an individual item in one of these collections, the reference a page item of one of a particular type. For example, if you use `PageItems` to reference a graph item, the `typename` value of that object is `GraphItem`.

PageItems properties

Property	Value type	What it is
`Application`	`Application`	Read-only. The Illustrator `Application` object.
`Count`	`Long`	Read-only. Number of elements in the collection.
`Parent`	`Object`	Read-only. The object's container.

PageItems methods

Method	Returns	What it does
`Index` (`itemPtr as` **`PageItem`**)	`Long`	Returns the index position of the object within the collection.
`Item` (`itemKey`)	`Object`	Returns an object reference to the object identified by `itemKey` (name or index).
`RemoveAll` ()	`Nothing`	Deletes all objects in this collection.

➤ Getting references to external files

This example illustrates how to obtain all references to external files in the current document. The result is presented in a new Illustrator document. Before running this, you must open a document that contains one or more linked images.

```
Dim appRef As New Illustrator.Application
Dim sourceDocument As Illustrator.Document
Dim artItem As Object
Dim rasterArt As Illustrator.RasterItem
Dim placedArt As Illustrator.PlacedItem
Dim fileReferences(9) As String
```

```
Dim index As Integer

Set sourceDocument = appRef.ActiveDocument
index = 0
For Each artItem In sourceDocument.PageItems
  Select Case TypeName(artItem)
    Case Is = "PlacedItem"
      fileReferences(index) = artItem.File
      index = index + 1
    Case Is = "RasterItem"
      If (Not artItem.Embedded) Then
        fileReferences(index) = artItem.File
        index = index + 1
      End If
  End Select
  If index = 10 Then
    Exit For
  End If
Next

'Write the file references to a new document
Dim reportDocument As Illustrator.Document
Dim fileNameText As Illustrator.TextFrame
Set reportDocument = appRef.Documents.Add

Set fileNameText = reportDocument.TextFrames.Add
fileNameText.Position = Array(50, 520)
fileNameText.Contents = "File references in " &_
   sourceDocument.Name & ":"
Dim counter As Integer
For counter = 0 To (index - 1)
  Set fileNameText = reportDocument.TextFrames.Add
  fileNameText.Position = Array(65, 500 - 20 * counter)
  fileNameText.Contents = fileReferences(counter)
Next
```

➤ Getting page items of particular types

```
Dim appRef As New Illustrator.Application
Dim artItem As Object

For Each artItem In appRef.ActiveDocument.PageItems
  If (TypeName(artItem) = "PlacedItem" Or _
       TypeName(artItem) = "RasterItem") Then
    artItem.Hidden = True
  End If
```

Paper

A container for information objects about the paper to be used for printing.

Paper properties

Property	Value type	What it is
Application	Application	Read-only. The Illustrator Application object.
Name	String	The paper name.
PaperInfo	PaperInfo object	The paper information.

PaperInfo

Contains information about the dimensions and imageable area of the paper to be used for printing.

PaperInfo properties

Property	Value type	What it is
Application	Application	Read-only. The Illustrator Application object.
CustomPaper	Boolean	If true, a custom paper is being used.
Height	Double	The paper's height in points.
ImageableArea	Array of 4 Doubles	The imageable area.
Width	Double	The paper's width in points.

➤ Using Paper and PaperInfo objects

```
' Paper and PaperInfo

Dim appRef As New Illustrator.Application
Dim docRef As Illustrator.Document
Dim itemRef As Illustrator.pathItem
Dim textRef As Illustrator.TextFrame
Dim printerRef
Dim sText As String

' Add a new document with some simple artwork
' (Must be a printable artwork for use with Printer object.)
Set docRef = appRef.Documents.Add()
Set itemRef = docRef.PathItems.Rectangle(600, 300, 200, 100)
Set textRef = docRef.TextFrames.Add()
textRef.Top = 600
textRef.Left = 50

' Get paper objects for first printer
Set printerRef = appRef.PrinterList(1)
sText = printerRef.Name
```

```
sText = sText & " paper list:" & vbCrLf

' Display data about available paper types
Dim paperRef
For Each paperRef In printerRef.PrinterInfo.PaperSizes
  sText = sText & paperRef.Name & vbCrLf
  sText = sText & vbTab & paperRef.PaperInfo.Width
  sText = sText & " x " & paperRef.PaperInfo.Height & vbCrLf
Next paperRef

textRef.Contents = sText
appRef.Redraw
```

ParagraphAttributes

Specifies the properties and attributes of a paragraph contained in a text frame.

Note: Paragraph attributes do not have default values, and are undefined until explicitly set.

ParagraphAttributes properties

Property	Value type	What it is
Application	Application	Read-only. The Illustrator Application object.
AutoLeadingAmount	Double	Auto leading amount as a percentage.
BunriKinshi	Boolean	If true, BunriKinshi is enabled.
BurasagariType	AiBurasagariTypeEnum	The Burasagari type which specifies whether punctuation is allowed to fall outside of the paragraph bounding box (not available when Kinsoku Shori is set to None).
DesiredGlyphScaling	Double	Desired glyph scaling expressed as a percentage.
DesiredLetterSpacing	Double	Desired letter spacing expressed as a percentage.
DesiredWordSpacing	Double	Desired word spacing expressed as a percentage.
EveryLineComposer	Boolean	If true, the *Every Line Composer* is enabled.
FirstLineIndent	Double	First line left indent expressed in points.
HyphenateCapitalizedWords	Boolean	If true, hyphenation is enabled for capitalized words.
Hyphenation	Boolean	If true, hyphenation is enabled for the paragraph.
HyphenationPreference	Double	Hyphenation preference scale for better spacing (0) or fewer hyphens (1) Range: 0.0 to 1.0
HyphenationZone	Double	Size of the hyphenation zone.
Justification	AiJustification	Paragraph justification.
Kinsoku	String	The name of a Kinsoku Shori set (a set of characters which cannot be used to begin or end a line of Japanese text).
KinsokuOrder	AiKinsokuOrderEnum	The preferred Kinsoku order.

Property	Value type	What it is
KurikaeshiMojiShori	Boolean	If true, Kurikaeshi Moji Shori is enabled (controls how repeated characters are handled in Japanese text).
LeadingType	AiAutoLeadingType	Specifies the type of auto leading.
LeftIndent	Double	Left indent of margin expressed in points.
MaximumConsecutiveHyphens	Long	Maximum number of consecutive hyphenated lines.
MaximumGlyphScaling	Double	Maximum glyph scaling expressed as a percentage.
MaximumLetterSpacing	Double	Maximum letter spacing expressed as a percentage.
MaximumWordSpacing	Double	Maximum word spacing expressed as a percentage.
MinimumAfterHyphen	Long	Minimum number of characters after a hyphen.
MinimumBeforeHyphen	Long	Minimum number of characters before a hyphen.
MinimumGlyphScaling	Double	Minimum glyph scaling expressed as a percentage.
MinimumHyphenatedWordSize	Long	Minimum hyphenated word size.
MinimumLetterSpacing	Double	Minimum letter spacing expressed as a percentage.
MinimumWordSpacing	Double	Minimum word spacing expressed as a percentage.
Mojikumi	String	The name of a predefined Mojikumi set for Japanese text composition.
Parent	Object	Read-only. The object's container.
RightIndent	Double	Right indent of margin expressed in points.
RomanHanging	Boolean	If true, Roman hanging punctuation is enabled.
SingleWordJustification	AiJustification	Single word justification.
SpaceAfter	Double	Spacing after paragraph in points.
SpaceBefore	Double	Spacing before paragraph in points.
TabStops	Array of TabStopInfo	Tab stop settings, as TabStopInfo objects.

➤ Modifying paragraph attributes

```
' Add a new document with an area text item
```

```
' containing 3 paragraphs.
' Use paragraph attributes to adjust the
' justification of each paragraph

Dim appRef As New Illustrator.Application
Dim docRef As Illustrator.Document
Dim pathRef As Illustrator.PathItem
Dim textRef As Illustrator.TextFrame

' Create 3 new pararagraphs
Set docRef = appRef.Documents.Add()
Set pathRef = docRef.PathItems.Rectangle(600, 200, 200, 400)
Set textRef = docRef.TextFrames.AreaText(pathRef)
textRef.Paragraphs.Add ("Left justified paragraph.")
textRef.Paragraphs.Add ("Center justified paragraph.")
textRef.Paragraphs.Add ("Right justified paragraph.")
textRef.TextRange.CharacterAttributes.Size = 28

' Change the justification of each paragraph
' using the paragraph attributes object
textRef.Paragraphs(1).ParagraphAttributes.Justification = aiRight
textRef.Paragraphs(2).ParagraphAttributes.Justification = aiCenter
textRef.Paragraphs(3).ParagraphAttributes.Justification = aiLeft
```

Paragraphs

A collection of `Paragraph` objects.

Paragraphs properties

Property	Value type	What it is
Application	`Application`	Read-only. The Illustrator `Application` object.
Count	`Long`	Read-only. Number of elements in the collection.
Parent	`Object`	Read-only. The object's container.

Paragraphs methods

Method	Returns	What it does
Add (contents as **String** [, relativeObject as **TextFrame**] [, insertionLocation as AiElementPlacement])	`TextRange` object	Adds a new paragraph with specified text contents at the specified location in the current document. If location is not specified, adds the new paragraph to the containing text frame after the current text selection or insertion point.
AddBefore (contents as **String**)	`TextRange` object	Adds a new paragraph with specified text contents before the current text selection or insertion point.
Index (itemPtr as **PageItem**)	`Long`	Returns the index position of the object within the collection.
Item (itemKey)	`Object`	Returns an object reference to the object identified by `itemKey` (name or index).
RemoveAll ()	`Nothing`	Deletes all objects in this collection.

➤ Counting paragraphs in a document

This script displays the total number of paragraphs contained in all of the `TextFrameItems` in the current document.

```
Dim appRef As New Illustrator.Application
Dim curTextArt As Illustrator.TextFrame
Dim curTextRange As Illustrator.TextRange
Dim numParagraphs As Integer

If appRef.Documents.Count > 0 Then
  numParagraphs = 0
  For Each curTextArt In appRef.ActiveDocument.TextFrames
```

```
    Set curTextRange = curTextArt.TextRange
    numParagraphs = numParagraphs + curTextRange.Paragraphs.Count
  Next
  If (numParagraphs > 1) Then
    MsgBox ("There are " & numParagraphs & " paragraphs in the doc.")
  Else
    MsgBox ("There is only one paragraph in the document.")
  End If
End If
```

ParagraphStyle

Associates character and paragraph attributes with a style name. The style object can be used to apply those attributes to the text in a `TextFrame` object.

ParagraphStyle properties

Property	Value type	What it is
Application	Application	Read-only. The Illustrator Application object.
CharacterAttributes	CharacterAttributes	Read-only. The character properties for the text range.
Name	String	The ParagraphStyle's name.
ParagraphAttributes	ParagraphAttributes	Read-only. The paragraph properties for the text range.
Parent	Object	Read-only. The object's container.

ParagraphStyle methods

Method	Returns	What it does
ApplyTo (textFrame as **TextFrame** [, clearingOverrides as **Boolean**])	Nothing	Applies the paragraph style to the text object.
Delete ()	Nothing	Deletes the object.

➤ Creating and applying a paragraph style

```
' Create 3 simple paragraphs with different attributes
' Create a ParagraphStyle and apply it to each paragraph

Dim appRef As New Illustrator.Application
Dim docRef As Illustrator.Document
Dim pathRef As Illustrator.PathItem
Dim textRef As Illustrator.TextFrame

' Create 3 new pararagraphs
Set docRef = appRef.Documents.Add()
Set pathRef = docRef.PathItems.Rectangle(600, 200, 200, 400)
Set textRef = docRef.TextFrames.AreaText(pathRef)
textRef.Paragraphs.Add ("Left justified paragraph.")
textRef.Paragraphs.Add ("Center justified paragraph.")
textRef.Paragraphs.Add ("Right justified paragraph.")
textRef.TextRange.CharacterAttributes.Size = 28

' Change the justification of each paragraph
' using the paragraph attributes object
textRef.Paragraphs(1).ParagraphAttributes.Justification = aiRight
textRef.Paragraphs(2).ParagraphAttributes.Justification = aiCenter
```

```
textRef.Paragraphs(3).ParagraphAttributes.Justification = aiLeft

' Create a new paragraph style
MsgBox "Creating and applying a paragraph style"
Dim paraStyle As Illustrator.ParagraphStyle
Set paraStyle = docRef.ParagraphStyles.Add("LeftIndent")

' Add some paragraph attributes
paraStyle.ParagraphAttributes.Justification = aiLeft
paraStyle.ParagraphAttributes.FirstLineIndent = 10

Dim pg
For Each pg In textRef.Paragraphs
  paraStyle.ApplyTo pg, True
Next pg
appRef.Redraw
```

ParagraphStyles

A collection of `ParagraphStyle` objects. See ParagraphStyle for an example.

ParagraphStyles properties

Property	Value type	What it is
Application	Application	Read-only. The Illustrator `Application` object.
Count	Long	Read-only. Number of elements in the collection.
Parent	Object	Read-only. The object's container.

ParagraphStyles methods

Method	Returns	What it does
Add (name as **String**)	ParagraphStyle object	Creates a new `ParagraphStyle` object.
Index (itemPtr as **ParagraphStyle**)	Long	Returns the index position of the object within the collection.
Item (itemKey)	ParagraphStyle object	Returns an object reference to the object identified by `itemKey` (name or index).
RemoveAll ()	Nothing	Deletes all objects in this collection.

PathItem

Art items such as those created using the Line, Rectangle, or Pen Tools. A path consists of path points that define its geometry. The `PathItem` class give you complete access to paths in Illustrator.

PathItem properties

Property	Value type	What it is
Application	`Application object`	Read-only. The Illustrator `Application` object.
Area	`Double`	Read-only. The area of this path in square points. If the area is negative, the path is wound counterclockwise. Self-intersecting paths can contain sub-areas that cancel each other out, which makes this value zero even though the path has apparent area.
ArtworkKnockout	`AiKnockoutState`	Is this object used to create a knockout, and if so, what kind.
BlendingMode	`AiBlendModes`	The mode used when compositing an object.
Clipping	`Boolean`	If `true`, this path is to be used as a clipping path
Closed	`Boolean`	If `true`, this path is closed
ControlBounds	`Variant Array of 4 Doubles`	Read-only. The bounds of the object including stroke width and controls.
Editable	`Boolean`	Read-only. If `true`, this path item is editable.
Evenodd	`Boolean`	If `true`, the even-odd rule is used to determine insideness.
FillColor	`Color object`	The fill color of the path.
Filled	`Boolean`	If `true`, the path should be filled.
FillOverprint	`Boolean`	If `true`, the art beneath a filled object should be overprinted.
GeometricBounds	`Variant Array of 4 Doubles`	Read-only. The bounds of the object excluding stroke width.
Guides	`Boolean`	If `true`, this path is a guide object.
Height	`Double`	The height of the path item, based on the `GeometricBounds`.
Hidden	`Boolean`	If `true`, this path item is hidden.
IsIsolated	`Boolean`	If `true`, this object is isolated.
Layer	`Layer object`	Read-only. The `Layer` to which this path item belongs.
Left	`Double`	The left position of the path item.
Locked	`Boolean`	If `true`, this path item is locked.

Property	Value type	What it is
Name	String	The name of this path item.
Note	String	The note text assigned to the path
Opacity	Double	The opacity of the object. Range 0.0 to 100.0
Parent	object	Read-only. The object that contains this path item.
PathPoints	PathPoints collection object	Read-only. The path points contained in this path item.
Polarity	AiPolarityValues	The polarity of the path.
Position	Variant Array of 2 Doubles	The position of the top left corner of the path item.
Resolution	Double	The resolution of the path in dots per inch.
Selected	Boolean	If true, this path item is selected.
SelectedPathPoints	PathPoints collection object	Read-only. All of the selected path points in the path.
Sliced	Boolean	If true, this path item is sliced.
StrokeCap	AiStrokeCap	The type of line capping.
StrokeColor	Color object	The stroke color for the path.
Stroked	Boolean	If true, the path should be stroked.
StrokeDashes	Variant Array of Doubles	Dash lengths (set to an empty array for a solid line).
StrokeDashOffset	Double	The default distance into the dash pattern at which the pattern should be started.
StrokeJoin	AiStrokeJoin	Type of joints for the path.
StrokeMiterLimit	Double	When a default stroke join is set to mitered, this property specifies when the join will be converted to beveled (squared-off) by default. The default miter limit of 4 means that when the length of the point reaches four times the stroke weight, the join switches from a miter join to a bevel join. A value of 1 specifies a bevel join. Range: 1 to 500 Default: 4
StrokeOverprint	Boolean	If true, the art beneath a stroked object should be overprinted.
StrokeWidth	Double	Width of stroke.
Tags	Tags collection object	Read-only. The tags contained in this path item.
Top	Double	The top position of this path item.
URL	String	The value of the Adobe URL tag assigned to this path item.

Property	Value type	What it is
VisibilityVariable	Variant	The Variable bound to this path item.
VisibleBounds	Variant Array of 4 Doubles	Read-only. The visible bounds of the path item including stroke width.
Width	Double	The width of the path item, based on the GeometricBounds.
WrapInside	Boolean	If true, the text frame object should be wrapped inside this object.
WrapOffset	Double	The offset to use when wrapping text around this object.
Wrapped	Boolean	If true, wrap text frame objects around this object (text frame must be above the object).
ZOrderPosition	Long	Read-only. The position of this art item within the stacking order of the group or layer (Parent) that contains the art item.

PathItem methods

Method	Returns	What it does
Copy ()	Nothing	Copies the path item to the clipboard; the associated document must be the frontmost document.
Cut ()	Nothing	Cuts the path item to the clipboard; the associated document must be the frontmost document.
Delete ()	Nothing	Deletes the object.
Duplicate ([relativeObject as **Object**], [insertionLocation as AiElementPlacement])	Object	Duplicates the art item, optionally with the location and position for the copy.
Move (relativeObject as **Object**, insertionLocation as AiElementPlacement)	Nothing	Moves the art item to a new location and position.
Resize (scaleX as **Double**, scaleY as **Double**, [, changePositions as **Boolean**] [, changeFillPatterns as **Boolean**] [, changeFillGradients as **Boolean**] [, changeStrokePattern as **Boolean**] [, changeLineWidths as **Double**] [, scaleAbout as AiTransformation])	Nothing	Scales the art item where scaleX is the horizontal scaling factor and scaleY is the vertical scaling factor; 100.0 = 100%

Method	Returns	What it does
Rotate (angle as **Double**, [, changePositions as **Boolean**] [, changeFillPatterns as **Boolean**] [, changeFillGradients as **Boolean**] [, changeStrokePattern as **Boolean**] [, rotateAbout as AiTransformation])	Nothing	Rotates the art item relative to the current rotation. The object is rotated counter-clockwise if the Angle value is positive, clockwise if the value is negative.
SetEntirePath (pathSpecification as **PathPoints**)	Nothing	Sets the path using the specified anchor points.
Transform (transformationMatrix as **Matrix**, [, changePositions as **Boolean**] [, changeFillPatterns as **Boolean**] [, changeFillGradients as **Boolean**] [, changeStrokePattern as **Boolean**] [, changeLineWidths as **Double**] [, transformAbout as AiTransformation])	Nothing	Transforms the art item by applying a transformation matrix.
Translate ([deltaX as **Double**] [, deltaY as **Double**] [, transformObjects as **Boolean**] [, transformFillPatterns as **Boolean**] [, transformFillGradients as **Boolean**] [, transformStrokePatterns as **Boolean**])	Nothing	Repositions the art item relative to the current position, where deltaX is the horizontal offset and deltaY is the vertical offset.
ZOrder (zOrderCmd as AiZOrderMethod)	Nothing	Arranges the art item's position in the stacking order of the group or layer (Parent) of this object.

➤ Setting attributes of a path

This script sets the stroke color and the fill color of the first path in the frontmost document.

```
Dim appRef As New Illustrator.Application
Dim frontDocument As Illustrator.Document
Dim firstPath As Illustrator.PathItem

Set frontDocument = appRef.ActiveDocument
Set firstPath = frontDocument.PathItems(1)
firstPath.Filled = True
firstPath.fillColor = frontDocument.Swatches(10).Color
firstPath.Stroked = True
firstPath.StrokeWidth = 5
```

➤ Setting a path consisting of straight lines

The SetEntirePath method provides an extremely efficient way to create paths that consist of straight lines. This script illustrates the use of the method.

```
Dim appRef As New Illustrator.Application
Dim frontDocument As Illustrator.Document
Dim newPath As Illustrator.PathItem
Dim lineList(10) As Variant
Dim index As Integer
```

```
For index = 0 To 10
 lineList(index) = Array(index * 10 + 50, (index - 5) ^ 2 * 5 + 50)
Next

Set frontDocument = appRef.ActiveDocument
Set newPath = frontDocument.PathItems.Add
newPath.SetEntirePath(lineList)
```

PathItems

A collection of `PathItem` objects. The methods `Ellipse`, `Polygon`, `Rectangle`, `RoundedRectangle`, and `Star` allow you to create complex `PathItems` using straightforward parameters. Note that `Rectangle`, `RoundedRectangle`, and `Ellipse` take the Y axis first (Top), then the X axis (Left). If you do not provide any parameters when calling these methods, default values are used.

PathItems properties

Property	Value type	What it is
Application	Application	Read-only. The Illustrator `Application` object.
Count	Long	Read-only. Number of elements in the collection.
Parent	Object	Read-only. The object's container.

PathItems methods

Method	Returns	What it does
Add ()	PathItem object	Creates a new `PathItem` object.
Ellipse ([top as **Double**] [, left as **Double**] [, width as **Double**] [, height as **Double**] [, reversed as **Boolean**] [, inscribed as **Boolean**])	PathItem object	Creates a new `PathItem` in the shape of an ellipse using the supplied parameters.
Index (item as **PathItem**)	Long	Returns the index position of the object within the collection.
Item (itemKey)	PathItem object	Returns an object reference to the object identified by `itemKey` (name or index).
Polygon ([centerX as **Double**] [, centerY as **Double**] [, radius as **Double**] [, sides as **Long**] [, reversed as **Boolean**])	PathItem object	Creates a new `PathItem` in the shape of a polygon using the supplied parameters.
Rectangle ([top as **Double**] [, left as **Double**] [, width as **Double**] [, height as **Double**] [, reversed as **Boolean**])	PathItem object	Creates a new `PathItem` in the shape of a rectangle using the supplied parameters.
RemoveAll ()	Nothing	Deletes all objects in this collection.

Method	Returns	What it does
RoundedRectangle ([top as **Double**] [, left as **Double**] [, width as **Double**] [, height as **Double**] [, horizontalRadius as **Double**] [, verticalRadius as **Double**] [, reversed as **Boolean**])	PathItem object	Creates a new PathItem in the shape of a rectangle with rounded corners using the supplied parameters.
Star ([centerX as **Double**] [, centerY as **Double**] [, radius as **Double**] [, innerRadius as **Double**] [, points as **Long**] [, reversed as **Boolean**])	PathItem object	Creates a new PathItem in the shape of a star using the supplied parameters.

➤ Creating a rectangle

This script illustrates how to create a new rectangle in the first layer of the frontmost document.

```
Dim appRef As New Illustrator.Application
Dim frontDocument As Illustrator.Document
Dim newRectangle As Illustrator.PathItem

Set frontDocument = appRef.ActiveDocument

' Create a new rectangle with
' top = 400, left side = 50, witdth = 150 and height = 100
Set newRectangle = frontDocument.PathItems.Rectangle(400,50,150,100)
```

PathPoint

A point on a specific path. Each path point is made up of an anchor point and a pair of handles, or control points.

PathPoint properties

Property	Value type	What it is
Anchor	Variant Array of 2 Doubles	The position of the anchor point.
Application	Application object	Read-only. The Illustrator Application object.
LeftDirection	Variant Array of 2 Doubles	The position of this path point's inward control point.
Parent	object	Read-only. The object that contains this PathPoint object.
PointType	AiPointType	The type of path point, either a curve or a corner. Any point can considered a corner point. Setting the type to a corner forces the left and right direction points to be on a straight line when the user attempts to modify them in the user interface.
RightDirection	Variant Array of 2 Doubles	The position of this path point's outward control point.
Selected	AiPathPointSelection	Are points of this path point selected, and if so, which ones.

PathPoint methods

Method	Returns	What it does
Delete ()	Nothing	Deletes the object.

➤ Shaping a path item by modifying a path point

This script illustrates how to change the shape of a PathItem by modifying the left direction and the right direction of the PathPoint.

```
Dim appRef As New Illustrator.Application
Dim firstPath As Illustrator.PathItem
Dim currentPoint As Illustrator.PathPoint
Dim nextPoint As Illustrator.PathPoint
Dim countOfPoints As Integer
Dim index As Integer
Dim deltax, deltay, length As Double

Set firstPath = appRef.ActiveDocument.PathItems(1)
countOfPoints = firstPath.PathPoints.Count

' Go through all PathPoints except the last one and set the
' left/right direction according to where the next point is
```

```
For index = 1 To (countOfPoints - 1)
  Set currentPoint = firstPath.PathPoints(index)
  Set nextPoint = firstPath.PathPoints(index + 1)

  deltax = nextPoint.Anchor(0) - currentPoint.Anchor(0)
  deltay = currentPoint.Anchor(1) - nextPoint.Anchor(1)
  length = Math.Sqr(deltax ^ 2 + deltay ^ 2)

  currentPoint.LeftDirection = _
     Array(currentPoint.Anchor(0) - (50 * deltax / length), _
          currentPoint.Anchor(1) - (50 * deltay / length))
  currentPoint.RightDirection = _
     Array(currentPoint.Anchor(0) + (50 * deltax / length), _
          currentPoint.Anchor(1) + (50 * deltay / length))
Next index
```

PathPoints

A collection of `PathPoint` objects in a path.

PathPoints properties

Property	Value type	What it is
`Application`	`Application`	Read-only. The Illustrator `Application` object.
`Count`	`Long`	Read-only. Number of elements in the collection.
`Parent`	`Object`	Read-only. The object's container.

PathPoints methods

Method	Returns	What it does
`Add` `()`	`PathPoint object`	Creates a new `PathPoint` object.
`Index` `(item as` **`PathPoint`**`)`	`Long`	Returns the index position of the object within the collection.
`Item` `(itemKey)`	`PathPoint object`	Returns an object reference to the object identified by `itemKey` (name or index).
`RemoveAll` `()`	`Nothing`	Deletes all objects in this collection.

➤ Adding a new path point

This script illustrates how to add a new path point to an existing path.

```
Dim appRef As New Illustrator.Application
Dim firstPath As Illustrator.PathItem
Dim newPoint As Illustrator.PathPoint

Set firstPath = appRef.ActiveDocument.PathItems(1)
Set newPoint = firstPath.PathPoints.Add

newPoint.Anchor = Array(75, 300)
newPoint.LeftDirection = Array(10, 280)
newPoint.RightDirection = Array(165, 330)
```

Pattern

A pattern definition contained in a document.

Pattern properties

Property	Value type	What it is
Application	Application object	Read-only. The Illustrator Application object.
Name	String	The pattern name.
Parent	Document object	Read-only. The document that contains this pattern.

Pattern methods

Method	Returns	What it does
Delete ()	Nothing	Deletes the object.

➤ Setting a fill color to a pattern

This script illustrates how to set the default fill color of document 1 to pattern 1.

```
Dim appRef As New Illustrator.Application
Dim frontDocument As Illustrator.Document
Dim PatternColor As Illustrator.PatternColor

Set frontDocument = appRef.Documents(1)
Set PatternColor = New Illustrator.PatternColor
PatternColor.Pattern = frontDocument.Patterns(1)
frontDocument.DefaultFilled = True
```

PatternColor

A pattern color specification, used in conjunction with the `Pattern` property of the `Color` specification. Pattern colors are created using a reference to an existing pattern in the document. A matrix may be specified to further transform the pattern color.

`PatternColor` objects can be used in any property that takes a color object, such as `fillColor` or `strokeColor`.

PatternColor properties

Property	Value type	What it is
Application	`Application object`	Read-only. The Illustrator `Application` object.
Matrix	`Matrix object`	An additional transformation matrix to manipulate the prototype pattern.
Pattern	`Pattern object`	A reference to the pattern object to use in this color definition.
Reflect	`Boolean`	If `true`, the prototype should be reflected before filling.
ReflectAngle	`Double`	The axis in degrees around which to reflect.
Rotation	`Double`	The angle in degrees to rotate the prototype pattern before filling.
ScaleFactor	`Variant Array of 2 Doubles`	The fraction to which to scale the prototype pattern before filling, represented as a point containing horizontal and vertical scaling percentages
ShearAngle	`Double`	The angle in degrees by which to slant the shear.
ShearAxis	`Double`	The axis in degrees to shear relative to.
ShiftAngle	`Double`	The angle in degrees to translate the unscaled prototype pattern before filling.
ShiftDistance	`Double`	The distance in points to which to translate the unscaled prototype pattern before filling.

➤ Modifying a pattern

This script illustrates how to modify the first pattern in a document.

```
Dim appRef As New Illustrator.Application
Dim colorOfPattern As New Illustrator.PatternColor
Dim swatchRef As Illustrator.Swatch
Dim swatchColor As Object
Dim firstPath As Illustrator.PathItem

For Each swatchRef In appRef.ActiveDocument.Swatches
   ' Get the color object of the swatch
   Set swatchColor = swatchRef.Color
   ' Only operate on patterns
   If (TypeName(swatchRef) = "PatternColor") Then
```

```
        ' Obtain the PatternColor from generic color object
        colorOfPattern = swatchColor.Pattern
        ' Change the pattern properties
        colorOfPattern.Rotation = 10
        ' Set the PatternColor of the original Color object
        swatchColor.Pattern = colorOfPattern
        ' Apply the color to the frontmost path
        Set firstPath = appRef.ActiveDocument.PathItems(1)
        firstPath.Filled = True
        firstPath.fillColor = swatchColor
        ' Change the definition of the pattern in the palette
        swatchRef.Color = swatchColor
    End If
Next
```

Patterns

A collection of `Pattern` objects in a document.

Patterns properties

Property	Value type	What it is
Application	`Application`	Read-only. The Illustrator `Application` object.
Count	`Long`	Read-only. Number of elements in the collection.
Parent	`Object`	Read-only. The object's container.

Patterns methods

Method	Returns	What it does
Add ()	`Pattern object`	Creates a new `Pattern` object.
Index (item as **Pattern**)	`Long`	Returns the index position of the object within the collection.
Item (itemKey)	`Pattern object`	Returns an object reference to the object identified by `itemKey` (name or index).
RemoveAll ()	`Nothing`	Deletes all objects in this collection.

➤ Removing a pattern

This script illustrates how to remove a pattern. Note after removing Illustrator objects you should set the variable that referenced the object you just removed to `Nothing`.

```
Dim appRef As New Illustrator.Application
Dim frontDocument As Illustrator.Document
Dim patternToRemove As Illustrator.Pattern

'Remove the second pattern. Then set the patternToRemove reference to
'nothing because it no longer references an existing Illustrator pattern
Set frontDocument = appRef.Documents(1)
Set patternToRemove = frontDocument.Patterns(2)
frontDocument.Patterns.Remove patternToRemove
```

PDFFileOptions

Options for opening Adobe PDF documents. This object is found in the `PDFFileOptions` property of the `Preferences` class.

PDFFileOptions properties

Property	Value type	What it is
`Application`	`Application`	Read-only. The Illustrator `Application` object.
`PageToOpen`	`Long`	Specifies which page should be used when opening a multipage document. Default: 1
`Parent`	`Object`	Read-only. The object's container.
`PDFCropToBox`	`AiPDFBoxType`	Specifies which box should be used when placing a multipage document. Default: `AiPDFMediaBox`

➤ Opening a PDF file

```
' PDFFileOptions

' Open a multi-page PDF file to a specific
' page using the PDFFileOptions object
' This sample assumes the "C:\temp\Multipage.pdf" exists

Dim appRef As New Illustrator.Application
Dim docRef As Illustrator.Document

appRef.UserInteractionLevel = aiDontDisplayAlerts

With appRef.Preferences.PDFFileOptions
   .PageToOpen = 2
   .PDFCropToBox = aiPDFBoundingBox
End With

Set docRef = appRef.Open("C:\temp\MultiPage.pdf", aiDocumentRGBColor)
```

PDFSaveOptions

Options that can be supplied when saving a document as an Adobe PDF file with the `document SaveAs` method.

PDFSaveOptions properties

Property	Value type	What it is
AcrobatLayers	`Boolean`	If `true`, create PDF layers from top-level layers (Acrobat 6 only option). Default: `false`
Application	`Application object`	Read-only. The Illustrator `Application` object.
BleedLink	`Boolean`	Link 4 bleed values. Default: `true`
BleedOffsetRect	`Variant Array of 4 Doubles`	The bleed offset rectangle.
ColorBars	`Boolean`	Draw color bars. Default: `false`
ColorCompression	AiCompressionQuality	Compression method for color bitmap images. Default: `aiNoCompression`
ColorConversionID	AiColorConversion	The PDF color conversion policy. Default: `aiColorConversionNone`
ColorDestinationID	AiColorDestination	The conversion target for color conversion. Default: `aiColorDestinationNone`
ColorDownsampling	`Double`	if zero, no downsampling, otherwise, the resolution to downsample color bitmap images to. Default: 150.0
ColorDownsamplingImage-Threshold	`Double`	Downsample if the image's resolution is above this value. Default: 225.0
ColorDownsamplingMethod	AiDownsampleMethod	Specifies how color bitmap images should be resampled. Default: `NoDownsample`
ColorProfileID	AiColorProfile	The color profile to include. Default: `aiColorProfileNone`
ColorTileSize	`Long`	Tile size when compressing with JPEG2000. Default: 256
Compatibility	AiPDFCompatibility	The version of the Acrobat® file format to create. Default: `Acrobat6`
CompressArt	`Boolean`	If `true`, line art and text should be compressed. Default: `true`
DocumentPassword	`String`	A password string to open the document. Default: no string

Property	Value type	What it is
EnableAccess	Boolean	If true, enable accessing 128-bit. Default: true
EnableCopy	Boolean	If true, enable copying of text 128-bit. Default: true
EnableCopyAccess	Boolean	If true, enable copying and accessing 40-bit. Default: true
EnablePlainText	Boolean	Enable plaintext metadata 128-bit; available only for Acrobat 6. Default: false
FlattenerOptions	PrintFlattenerOptions object	The printing flattener options
FlattenerPreset	String	Transparency flattener style name. Default: Custom
FontSubsetThreshold	Double	Include a subset of fonts when less than this percentage of characters are used. Range: 0.0 to 100.0. Default: 100.0
GenerateThumbnails	Boolean	If true, generate thumbnails for the saved document. Default: true
GrayscaleCompression	AiCompressionQuality	Specifies how grayscale bitmap images should be compressed. Default: NoCompression
GrayscaleDownsampling	Double	If zero, no downsampling, otherwise, the resolution to downsample grayscale images to. Default: 150.0
GrayscaleDownsampling-ImageThreshold	Double	Downsample if the image's resolution is above this value. Default: 225.0
GrayscaleDownsampling-Method	AiDownsampleMethod	How should grayscale bitmap images be resampled. Default: NoDownsample
GrayscaleTileSize	Long	Tile size when compressing with JPEG2000. Default: 256
MonochromeCompression	AiMonochromeCompression	How should monochrome bitmap images be compressed. Default: NoMonoCompression
MonochromeDownsampling	Double	The resolution to downsample images to. If 0, no downsampling. Default: 300.0
MonochromeDownsampling-ImageThreshold	Double	Downsample if the image's resolution is above this value. Default: 450
MonochromeDownsampling-Method	AiDownsampleMethod	How should monochrome bitmap images be resampled. Default: NoDownsample

Property	Value type	What it is
Offset	Double	Offset from artwork to draw printer marks. Default: 0
Optimization	Boolean	If true, saved PDF should be optimized for fast web viewing. Default: false
OutputCondition	String	An optional comment to add to the PDF file, describing the intended printing condition. Default: not included
OutputConditionID	String	The name of a registered printing condition. Default: not included
PageInformation	Boolean	If true, draw page information. Default: false
PageMarksType	AiPageMarksStyle	The page marks style. Default: PageMarksStandard
PDFAllowPrinting	AiPDFPrintAllowedEnum	PDF security printing permission. Default: Print128HighResolution
PDFChangesAllowed	AiPDFChangesAllowedEnum	PDF security changes allowed. Default: Change128AnyChanges
PDFPreset	String	PDF preset name
PDFXStandard	AiPDFChangesAllowedEnum	The PDF standard with which this document complies. Default: aiPDFXNone
PDFXStandardDescription	string	A description of the PDF standard from the selected preset.
PermissionPassword	String	A password string to restrict editing security settings. Default: no string
PreserveEditability	Boolean	If true, preserve Illustrator editing capabilities when saving the document. Default: true
PrinterResolution	Double	Flattening printer resolution. Default: 800.0
RegistrationMarks	Boolean	Draw registration marks. Default: false
RequireDocumentPassword	Boolean	If true, a password should be required to open the document. Default: false
RequirePermissionPassword	Boolean	If true, a password should be used to restrict editing security settings. Default: false
Trapped	Boolean	If true, manual trapping has been prepared for the document. Default: false

Property	Value type	What it is
TrimMarks	Boolean	If true, draw trim marks. Default: false
TrimMarkWeight	AiPDFTrimMarkWeight	Trim mark weight. Default: TrimMarkWeight0125
ViewAfterSaving	Boolean	If true, view PDF after saving. Default: false

➤ Saving to PDF

This script illustrates how to save the frontmost document as PDF.

```
Dim appRef As New Illustrator.Application
Dim saveOptions As New Illustrator.PDFSaveOptions
saveOptions.ColorCompression = aiJPEGHigh
saveOptions.Compatibility = aiAcrobat5
```

PhotoshopFileOptions

Options for opening a Photoshop file with the `Document open` method.

PhotoshopFileOptions properties

Property	Value type	What it is
Application	`Application object`	Read-only. The Illustrator `Application` object.
Parent	`Object`	Read only. This object's parent object.
PixelAspectRatioCorrection	`Boolean`	If `true`, imported images having non-square pixel aspect ratios should be corrected.
PreserveImageMaps	`Boolean`	If `true`, the image maps should be preserved when the document is converted. Default: `true`
PreserveLayers	`Boolean`	If `true`, layers should be preserved when the document is converted. Default: `true`
PreserveSlices	`Boolean`	If `true`, slices should be preserved when the document is converted. Default: `true`

➤ Opening a Photoshop document

```
' PhotoshopFileOptions

' Open a PSD file and retain layers
' using the PhotoshopFileOptions object
' This sample assumes the "C:\temp\MultiLayer.psd" exists

Dim appRef As New Illustrator.Application
Dim docRef As Illustrator.Document

With appRef.Preferences.PhotoshopFileOptions
   PreserveLayers = True
   PixelAspectRatioCorrection = False
End With

Set docRef = appRef.Open("C:\temp\MultiLayers.psd", aiDocumentRGBColor)
```

PlacedItem

An artwork item (optionally stored in an external file) placed in a document. A `PlacedItem` must correspond to a file containing vector-graphic data, such as a PICT, EPS, or PDF file.

Placed items work only with embedded vector files: EPS, PDF, SVG, and embedded AI. Users can place vector art files, such as `EPS` and `PDF` files, with the **File > Place...** command in Illustrator. However, placed image files such as JPG files are represented in the scripting object model as `RasterItem` objects (see RasterItem/RasterItems).

When you create a `PlacedItem`, Illustrator may display a dialog. To avoid this, check the box to turn the warning off the first time the dialog is displayed.

PlacedItem properties

Property	Value type	What it is
Application	`Application object`	Read-only. The Illustrator `Application` object.
ArtworkKnockout	`AiKnockoutState`	Is this object used to create a knockout, and if so, what kind.
BlendingMode	`AiBlendModes`	The mode used when compositing an object.
BoundingBox	`Variant Array of 4 Doubles`	Read-only. Dimensions of `PlacedItem` regardless of transformations.
ContentVariable	`Variable`	The `Variable` bound to this `PlacedItem`. It is not necessary to set the type of the `ContentVariable` before binding. Illustrator automatically sets the type to be the same as the `PageItem` to which it is bound.
ControlBounds	`Variant Array of 4 Doubles`	Read-only. The bounds of the object including stroke width and controls.
Editable	`Boolean`	Read-only. If `true`, this `PlacedItem` is editable.
File	`String`	The file containing the placed object.
GeometricBounds	`Variant Array of 4 Doubles`	Read-only. The bounds of the object excluding stroke width.
Height	`Double`	The height of the `PlacedItem`, based on `GeometricBounds`.
Hidden	`Boolean`	If `true`, this `PlacedItem` is hidden.
IsIsolated	`Boolean`	If `true`, this object is isolated.
Layer	`Layer object`	Read-only. The layer this `PlacedItem` belongs to.
Left	`Double`	The left position of the `PlacedItem`.
Locked	`Boolean`	If `true`, this `PlacedItem` is locked
Matrix	`Matrix object`	The transformation matrix applied to the `PlacedItem`.

Property	Value type	What it is
Name	String	The name of this PlacedItem.
Opacity	Double	The opacity of the object. Range: 0.0 to 100.0
Parent	object	Read-only. The object that contains this PlacedItem.
Position	Variant Array of 2 Doubles	The position of the top left corner of the PlacedItem.
Selected	Boolean	If true, this PlacedItem is selected.
Sliced	Boolean	If true, this PlacedItem is sliced.
Tags	Tags collection object	Read-only. The tags contained in this PlacedItem.
Top	Double	The top position of the PlacedItem.
URL	String	The value of the Adobe URL tag assigned to this PlacedItem.
VisibilityVariable	Variable	The Variable bound to this PlacedItem.
VisibleBounds	Variant Array of 4 Doubles	Read-only. The visible bounds of the PlacedItem including stroke width.
Width	Double	The width of the PlacedItem, based on GeometricBounds.
WrapInside	Boolean	If true, the text frame object should be wrapped inside this object.
WrapOffset	Double	The offset to use when wrapping text around this object.
Wrapped	Boolean	If true, the text frame objects should be wrapped around this object (text frame must be above the object).
ZOrderPosition	Long	Read-only. The position of this art item within the stacking order of the group or layer (Parent) that contains the art item.

PlacedItem methods

Method	Returns	What it does
Copy ()	Nothing	Copies the item to the clipboard; the associated document must be the frontmost document.
Cut ()	Nothing	Cuts the item to the clipboard; the associated document must be the frontmost document.
Delete ()	Nothing	Deletes the object.

Method	Returns	What it does
Duplicate ([relativeObject as **Object**], [insertionLocation as AiElementPlacement])	Object	Duplicates the art item, optionally with the location and position for the copy.
Embed ()	Nothing	Embeds this art in the document. Converts the art to art item objects as needed and deletes this object.
Resize (scaleX as **Double**, scaleY as **Double**, [, changePositions as **Boolean**] [, changeFillPatterns as **Boolean**] [, changeFillGradients as **Boolean**] [, changeStrokePattern as **Boolean**] [, changeLineWidths as **Double**] [, scaleAbout as AiTransformation])	Nothing	Scales the art item where scaleX is the horizontal scaling factor and scaleY is the vertical scaling factor; 100.0 = 100%
Rotate (angle as **Double**, [, changePositions as **Boolean**] [, changeFillPatterns as **Boolean**] [, changeFillGradients as **Boolean**] [, changeStrokePattern as **Boolean**] [, rotateAbout as AiTransformation])	Nothing	Rotates the art item relative to the current rotation. The object is rotated counter-clockwise if the Angle value is positive, clockwise if the value is negative.
Transform (transformationMatrix as **Matrix**, [, changePositions as **Boolean**] [, changeFillPatterns as **Boolean**] [, changeFillGradients as **Boolean**] [, changeStrokePattern as **Boolean**] [, changeLineWidths as **Double**] [, transformAbout as AiTransformation])	Nothing	Transforms the art item by applying a transformation matrix.
Translate ([deltaX as **Double**] [, deltaY as **Double**] [, transformObjects as **Boolean**] [, transformFillPatterns as **Boolean**] [, transformFillGradients as **Boolean**] [, transformStrokePatterns as **Boolean**])	Nothing	Repositions the art item relative to the current position, where deltaX is the horizontal offset and deltaY is the vertical offset.
ZOrder (zOrderCmd as AiZOrderMethod)	Nothing	Arranges the art item's position in the stacking order of the group or layer (Parent) of this object.

➤ Changing selection in placed items

This script illustrates how to change the selection of PlacedItems.

```
Dim appRef As New Illustrator.Application
Dim placedArt As Illustrator.PlacedItem

For Each placedArt In appRef.ActiveDocument.PlacedItems
  placedArt.Selected = Not (placedArt.Selected)
Next
```

PlacedItems

A collection of placed art items. See PlacedItem for an example of use.

PlacedItems properties

Property	Value type	What it is
Application	Application	Read-only. The Illustrator Application object.
Count	Long	Read-only. Number of elements in the collection.
Parent	Document	Read-only. The object's document container.

PlacedItems methods

Method	Returns	What it does
Add ()	PlacedItem object	Creates a new PlacedItem object. Use to place new art in a document, and use the resulting PlacedItem object's embed method to convert that art to embedded art items.
Index (item as **PlacedItem**)	Long	Returns the index position of the object within the collection.
Item (itemKey)	PlacedItem object	Returns an object reference to the object identified by itemKey (name or index).
RemoveAll ()	Nothing	Deletes all objects in this collection.

PluginItem

An art item created by an Illustrator plug-in such as the blend tool. Scripts can create a plugin item using `PlacedItem.trace` or `RasterItem.trace`, and can copy existing plugin items using the `duplicate` method, or copy and paste them, but cannot create `PluginItem` objects directly.

PluginItem properties

Property	Value type	What it is
Application	`Application object`	Read-only. The Illustrator `Application` object.
ArtworkKnockout	`AiKnockoutState`	Is this object used to create a knockout, and if so, what kind.
BlendingMode	`AiBlendModes`	The mode used when compositing an object.
ControlBounds	`Variant Array of 4 Doubles`	Read-only. The bounds of the object including stroke width and controls.
Editable	`Boolean`	Read-only. If `true`, this item is editable.
GeometricBounds	`Variant Array of 4 Doubles`	Read-only. The bounds of the object excluding stroke width.
Height	`Double`	The height of the item, based on `GeometricBounds`.
Hidden	`Boolean`	If `true`, this item is hidden.
IsIsolated	`Boolean`	If `true`, this object is isolated.
IsTracing	`Boolean`	If `true`, this plugin group represents a vector art item created by tracing a raster art item. The `tracing` property contains the tracing object associated with the options used to create it.
Layer	`Layer object`	Read-only. The layer this item belongs to.
Left	`Double`	The left position of the item.
Locked	`Boolean`	If `true`, this item is locked
Name	`String`	The name of this item.
Opacity	`Double`	The opacity of the object. Range: 0.0 to 100.0
Parent	`object`	Read-only. The object that contains this item.
Position	`Variant Array of 2 Doubles`	The position of the top left corner of the item.
Selected	`Boolean`	If `true`, this item is selected.
Sliced	`Boolean`	If `true`, this item is sliced.
Tags	`Tags collection object`	Read-only. The tags contained in this item.
Top	`Double`	The top position of the item.

Property	Value type	What it is
Tracing	TracingObject	When this plugin group was created by tracing (IsTracing is true), the tracing object associated with the options used to create it.
URL	String	The value of the Adobe URL tag assigned to this item.
VisibilityVariable	Variable	The Variable bound to this item.
VisibleBounds	Variant Array of 4 Doubles	Read-only. The visible bounds of the item including stroke width.
Width	Double	The width of the item, based on GeometricBounds.
WrapInside	Boolean	If true, the text frame object should be wrapped inside this object.
WrapOffset	Double	The offset to use when wrapping text around this object.
Wrapped	Boolean	If true, the text frame objects should be wrapped around this object (text frame must be above the object).
ZOrderPosition	Long	Read-only. The position of this art item within the stacking order of the group or layer (Parent) that contains the art item.

PluginItem methods

Method	Returns	What it does
Copy ()	Nothing	Copies the item to the clipboard; the associated document must be the frontmost document.
Cut ()	Nothing	Cuts the item to the clipboard; the associated document must be the frontmost document.
Delete ()	Nothing	Deletes the object.
Duplicate ([relativeObject as **Object**], [insertionLocation as AiElementPlacement])	Object	Duplicates the art item, optionally with the location and position for the copy.
Move (relativeObject as **Object**, insertionLocation as AiElementPlacement)	Nothing	Moves the art item to a new location and position.

Method	Returns	What it does
Resize (scaleX as **Double**, scaleY as **Double**, [, changePositions as **Boolean**] [, changeFillPatterns as **Boolean**] [, changeFillGradients as **Boolean**] [, changeStrokePattern as **Boolean**] [, changeLineWidths as **Double**] [, scaleAbout as AiTransformation])	Nothing	Scales the art item where scaleX is the horizontal scaling factor and scaleY is the vertical scaling factor; 100.0 = 100%
Rotate (angle as **Double**, [, changePositions as **Boolean**] [, changeFillPatterns as **Boolean**] [, changeFillGradients as **Boolean**] [, changeStrokePattern as **Boolean**] [, rotateAbout as AiTransformation])	Nothing	Rotates the art item relative to the current rotation. The object is rotated counter-clockwise if the Angle value is positive, clockwise if the value is negative.
Trace ()	PluginItem object	Converts the raster art for this object to vector art, using default options. Reorders the raster art into the source art of a plugin group, and converts it into a group of filled and/or stroked paths that resemble the original image. Creates and returns a PluginItem object that references a TraceObject object.
Transform (transformationMatrix as **Matrix**, [, changePositions as **Boolean**] [, changeFillPatterns as **Boolean**] [, changeFillGradients as **Boolean**] [, changeStrokePattern as **Boolean**] [, changeLineWidths as **Double**] [, transformAbout as AiTransformation])	Nothing	Transforms the art item by applying a transformation matrix.
Translate ([deltaX as **Double**] [, deltaY as **Double**] [, transformObjects as **Boolean**] [, transformFillPatterns as **Boolean**] [, transformFillGradients as **Boolean**] [, transformStrokePatterns as **Boolean**])	Nothing	Repositions the art item relative to the current position, where deltaX is the horizontal offset and deltaY is the vertical offset.
ZOrder (zOrderCmd as AiZOrderMethod)	Nothing	Arranges the art item's position in the stacking order of the group or layer (Parent) of this object.

➤ Copying a plug-in item

This example demonstrates how to create a new PluginItem by copying an existing PluginItem.

```
Dim appRef As New Illustrator.Application
Dim frontDocument As Illustrator.Document
Dim pluginArt As Illustrator.PluginItem
```

```
Set frontDocument = appRef.ActiveDocument
If (frontDocument.PluginItems.Count > 0) Then
   Set pluginArt = frontDocument.PluginItems(1)
   pluginArt.Copy
   frontDocument.Paste
Else
   MsgBox "There is no plug-in art in the front document"
End If
```

PluginItems

A collection of `PluginItem` objects in a document.

PluginItems properties

Property	Value type	What it is
`Application`	`Application`	Read-only. The Illustrator `Application` object.
`Count`	`Long`	Read-only. Number of elements in the collection.
`Parent`	`Layer or GroupItem object`	Read-only. The object's container.

PluginItems methods

Method	Returns	What it does
`Index` (`item as` **`PluginItem`**)	`Long`	Returns the index position of the object within the collection.
`Item` (`itemKey`)	`Plugin object`	Returns an object reference to the object identified by `itemKey` (name or index).
`RemoveAll` ()	`Nothing`	Deletes all objects in this collection.

PPDFile

Associates file information with a PostScript Printer Description (PPD) file.

PPDFile properties

Property	Value type	What it is
Application	Application object	Read-only. The Illustrator Application object.
Name	String	The PPD name.
PPDInfo	PPDFileInfo object	The PPD file information.

PPDFileInfo

Information about a PostScript Printer Description (PPD) file.

PPDFileInfo properties

Property	Value type	What it is
Application	Application object	Read-only. The Illustrator Application object.
LanguageLevel	String	The PostScript language level.
PPDFilePath	String	Path specification for the PPD file.
ScreenList	Variant	List of color separation screens.
ScreenSpotFunctionList	Variant	List of color separation screen spot functions.

➤ Using a PPD file

```
' PPD
' Create a document and add a TextFrame for each available PPD.
' Use the textFrame to display PPD info

Dim appRef As New Illustrator.Application
Dim docRef As Illustrator.Document
Dim sPPD As String
Dim x As Integer
Dim y As Integer
Set docRef = appRef.Documents.Add()
x = 30
y = docRef.Height - 30

Dim ppd
For Each ppd In appRef.PPDFileList
   ' get data for each PPD
   sPPD = ppd.Name & vbCrLf
   sPPD = sPPD & vbTab & "PS Level "
   sPPD = sPPD & ppd.PPDInfo.LanguageLevel & vbCrLf
```

```
    sPPD = sPPD & vbTab & "Path: "
    sPPD = sPPD & ppd.PPDInfo.File & vbCrLf

    ' display data with a textFrame
    Dim textRef As Illustrator.TextFrame
    Set textRef = docRef.TextFrames.Add()
    textRef.TextRange.CharacterAttributes.Size = 8
    textRef.Contents = sPPD
    textRef.Top = y
    textRef.Left = x
    appRef.Redraw
    y = y - textRef.Height
Next ppd
```

Preferences

Specifies the preferred options for PDF and Photoshop files.

Preferences properties

Property	Value type	What it is
Application	Application object	Read-only. The Illustrator Application object.
Parent	Object	Read-only. The object's container.
PDFFileOptions	PDFFileOptions	Read-only. Options to use when opening or placing a PDF file.
PhotoshopFileOptions	PhotoshopFileOptions	Read-only. Options to use when opening or placing a Photoshop file.

PrintColorManagementOptions

Contains information used for color management of the document.

PrintColorManagementOptions properties

Property	Value type	What it is
Application	Application object	Read-only. The Illustrator Application object.
ColorProfileMode	AiPrintColorProfile	The color management profile mode. Default: aiSourceProfile
Intent	AiPrintColorIntent	The color management intent type. Default: aiRelativeColorimetric
Name	String	The color management profile name.

➤ Managing colors for printing

```
' PrintColorManagementOptions
' Create a PrintColorManagementOptions object and assign it
' to a PrintOptions object, then print with each color intent

' create a simple path item and apply a graphic style to it
Dim appRef As New Illustrator.Application
Dim docRef As Illustrator.Document
Dim pathItem As Illustrator.pathItem
Set docRef = appRef.Documents.Add
Set pathItem = docRef.PathItems.Rectangle(600, 200, 200, 200)
docRef.GraphicStyles(2).ApplyTo pathItem

Dim colorOptions As New Illustrator.PrintColorManagementOptions
Dim printOptionsRef As New Illustrator.PrintOptions
printOptionsRef.ColorManagementOptions = colorOptions
colorOptions.Name = "ColorMatch RGB"

' Print the current document once for each color intent.
colorOptions.Intent = aiAbsoluteColorimetric
docRef.PrintOut
colorOptions.Intent = aiPerceptualIntent
docRef.PrintOut
colorOptions.Intent = aiRelativeColorimetric
docRef.PrintOut
colorOptions.Intent = aiSaturationIntent
docRef.PrintOut
```

PrintColorSeparationOptions

Information about the color separations to be used in printing the document.

PrintColorSeparationOptions properties

Property	Value type	What it is
Application	Application object	Read-only. The Illustrator Application object.
ColorSeparationMode	AiPrintColorSeparationMode	The color separation type. Default: Composite
ConvertSpotColors	Boolean	If true, spot colors should be converted to process colors. Default: false
InkList	Variant Array of Ink objects	The list of inks for color separation.
OverPrintBlack	Boolean	If true, overprint in black. Default: false

➤ Managing print color separations

```
' PrintColorSeparationOptions
' Create a new document and add some symbol items
' Create a PrintColorSeparationOptions object and
' print with various separation settings

' Create a new document and add some artwork
Dim appRef As New Illustrator.Application
Dim docRef As Illustrator.Document
Set docRef = appRef.Documents.Add()
Dim symbolRef As Illustrator.Symbol
Dim itemRef As Illustrator.SymbolItem
Dim y As Integer
Dim i as Integer

y = docRef.Height - 30
i = 1
Do While (i < 11)
   Set symbolRef = docRef.Symbols(i)
   Set itemRef = docRef.SymbolItems.Add(symbolRef)
   itemRef.Top = y
   itemRef.Left = 100
   y = (y - (itemRef.Height + 10))
   i = i + 1
Loop
appRef.Redraw

' Create a separations object and assign it a
' PrintOptions object
Dim printOpts As New Illustrator.printOptions
Dim separationOpts As New Illustrator.PrintColorSeparationOptions
printOpts.ColorSeparationOptions = separationOpts
```

```
' Print with various separation options
separationOpts.ConvertSpotColors = True
separationOpts.OverPrintBlack = True
separationOpts.ColorSeparationMode = aiComposite
docRef.PrintOut printOpts

separationOpts.ColorSeparationMode = aiInRIPSeparation
docRef.PrintOut printOpts

separationOpts.ConvertSpotColors = False
separationOpts.OverPrintBlack = False
separationOpts.ColorSeparationMode = aiHostBasedSeparation
docRef.PrintOut printOpts
```

PrintCoordinateOptions

Information about the media and associated printing parameters.

PrintCoordinateOptions properties

Property	Value type	What it is
Application	Application object	Read-only. The Illustrator Application object.
Emulsion	Boolean	If true, flip artwork will be flipped horizontally. Default: false
FitToPage	Boolean	Whether to proportionally scale the artwork to fit on media. Default: false
HorizontalScale	Double	The horizontal scaling factor expressed as a percentage (100 = 100%) Range: 1.0 to 10000.0. Default: 100.0
Orientation	AiPrintOrientation	The artwork orientation. Default: Portrait
Position	AiPrintPosition	The artwork position on media. Default: TranslateCenter
Tiling	AiPrintTiling	The page tiling mode. Default: TileNone
VerticalScale	Double	The vertical scaling factor expressed as a percentage (100 = 100%) Range: 1.0 to 10000.0. Default: 100.0

➤ Managing print coordinates

```
' PrintCoordinateOptions
' Create some simple artwork (that extends off the page)
' and print it with various Coordinate Options

' Create a TextFrame that extends off the page
Dim appRef As New Illustrator.Application
Dim docRef As Illustrator.Document
Set docRef = appRef.Documents.Add()
Dim textRef As Illustrator.TextFrame
Set textRef = docRef.TextFrames.Add()
textRef.Contents = "This should extend off the page"
textRef.Left = -50
textRef.Top = 600
textRef.Width = docRef.Width + 100
textRef.Height = 150
appRef.Redraw

' Print the item using various settings of the
' PrintCoordinateOptions object
Dim coordinateOptions As New Illustrator.PrintCoordinateOptions
Dim printOptions As New Illustrator.printOptions
printOptions.coordinateOptions = coordinateOptions

coordinateOptions.Emulsion = True       ' reverse from right to left
coordinateOptions.FitToPage = True      ' fit artwork to page size
coordinateOptions.Orientation = aiLandscape
```

```
docRef.PrintOut printOptions

coordinateOptions.Emulsion = False
coordinateOptions.Orientation = aiPortrait
coordinateOptions.HorizontalScale = 50
coordinateOptions.VerticalScale = 50
docRef.PrintOut printOptions
```

Printer

Associates an available printer with printer information. To request a list of printers, you must first have a document open or an error is returned.

Printer properties

Property	Value type	What it is
Application	Application object	Read-only. The Illustrator Application object.
Name	String	The printer name
PrinterInfo	PrinterInfo object	The printer information

PrinterInfo

Contains all configuration information about a printer.

PrinterInfo properties

Property	Value type	What it is
Application	Application object	Read-only. The Illustrator Application object.
BinaryPrintingSupport	Boolean	If true, the printer supports binary printing.
ColorSupport	AiPrinterColorMode	The printer color capability.
CustomPaperSupport	Boolean	If true, the printer supports custom paper sizes.
CustomPaperTransverse-Support	Boolean	If true, the printer supports custom paper transverse.
DeviceResolution	Double	The printer default resolution.
InRIPSeparationSupport	Boolean	If true, the printer supports InRIP color separation.
MaxDeviceResolution	Double	The printer maximum device resolution.
MaxPaperHeight	Double	Custom paper's maximum height.
MaxPaperHeightOffset	Double	Custom paper's maximum height offset.
MaxPaperWidth	Double	Custom paper's maximum width.
MaxPaperWidthOffset	Double	Custom paper's maximum width offset.
MinPaperHeight	Double	Custom paper's minimum height.
MinPaperHeightOffset	Double	Custom paper's minimum height offset.
MinPaperWidth	Double	Custom paper's minimum width.
MinPaperWidthOffset	Double	Custom paper's minimum width offset.

Property	Value type	What it is
PaperSizes	Variant Array of Paper objects	The list of supported paper sizes.
PostScriptLevel	AiPrinterPostScriptLevelEnum	The PostScript language level.
PrinterType	AiPrinterTypeEnum	The printer type.

➤ Finding printers

This script accesses and displays the list of printer names.

```
' Printer
' Use the PrinterList to obtain the name
' of each printer and display it on the page

Dim appRef As New Illustrator.Application
Dim docRef As Illustrator.Document
Dim textRef As Illustrator.TextFrame
Dim sData As String

Set docRef = appRef.Documents.Add()
Set textRef = docRef.TextFrames.Add()
textRef.Top = docRef.Height - 50
textRef.Left = 50
sData = "Printers:"

Dim printerRef
For Each printerRef In appRef.PrinterList
   sData = sData & vbCrLf & printerRef.Name
Next PrinterRef

textRef.Contents = sData
appRef.Redraw
```

➤ Finding printer information

```
' PrintInfo
' Add a document and TextFrame
' Loop through printer list, show info on each printer
' using the PrintInfo attribute

' create a new document
Dim appRef As New Illustrator.Application
Dim docRef As Illustrator.Document
Set docRef = Documents.Add()

' add title text frame
Dim textRef1 As Illustrator.TextFrame
Set textRef1 = docRef.TextFrames.Add()
textRef1.Contents = "Checking Printers..."
textRef1.Top = 600
textRef1.Left = 200
appRef.Redraw
```

```
' for each printer, list the PS support and InHostRip support
Dim sPrintInfo As String
Dim infoRef As Illustrator.PrinterInfo
Dim printerRef
For Each printerRef In appRef.PrinterList
   sPrintInfo = sPrintInfo & printerRef.Name & vbCrLf
   sPrintInfo = sPrintInfo & vbTab & "PS Level = "
   sPrintInfo = sPrintInfo & CStr(printerRef.PrinterInfo.PostScriptLevel)_
      & vbCrLf
   sPrintInfo = sPrintInfo & vbTab & "Device Resolution = "
   sPrintInfo = sPrintInfo & CStr(printerRef.PrinterInfo.DeviceResolution)_
      & vbCrLf
   sPrintInfo = sPrintInfo & vbTab & "InRIPSeparation Support = "
   sPrintInfo = sPrintInfo &
CStr(printerRef.PrinterInfo.InRIPSeparationSupport) & vbCrLf
Next printerRef

textRef1.Contents = sPrintInfo
appRef.Redraw
```

PrintFlattenerOptions

Contains flattening options for use when Illustrator outputs artwork that contains transparency into a non-native format.

PrintFlattenerOptions properties

Property	Value type	What it is
Application	Application	Read-only. The Illustrator Application object.
ClipComplexRegions	Boolean	If true, clip complex regions. Default: false
ConvertStrokesToOutlines	Boolean	If true, convert all strokes to outlines. Default: false
ConvertTextToOutlines	Boolean	If true, convert all text to outlines. Default: false
FlatteningBalance	Double	The flattening balance. Range: 0.0 to 100.0. Default: 100.0
GradientResolution	Long	The gradient resolution in dots per inch (dpi). Range: 1.0 to 9600.0. Default: 300.0
Overprint	AiPDFOverprint	Whether to preserve, discard, or simulate overprinting. Default: aiPreservePDFOverprint
RasterizationResolution	Double	The rasterization resolution in dots per inch (dpi). Range: 1.0 to 9600.0. Default: 300.0

➤ Setting print flattening

```
' PrintFlattenerOptions
' Open a document, and add a simple text frame
' with a graphic style applied.
' Print the document with "low" and "high"
' flattener settings

Dim appRef As New Illustrator.Application
Dim docRef As Illustrator.Document

Set docRef = appRef.Documents.Add()
Dim itemRef As Illustrator.PathItem
Set itemRef = docRef.PathItems.Rectangle(600, 200, 200, 200)
docRef.GraphicStyles(2).ApplyTo itemRef

' Create a PrintFlattenerOptions object and
' assign it to a PrintOptions object.
Dim flatOpts As New Illustrator.PrintFlattenerOptions
Dim printOpts As New Illustrator.printOptions
printOpts.FlattenerOptions = flatOpts

' print faster with low resolution
flatOpts.ClipComplexRegions = True
flatOpts.GradientResolution = 30
flatOpts.RasterizationResolution = 30
docRef.PrintOut printOpts
```

```
' print slower with higher resolution
flatOpts.ClipComplexRegions = False
flatOpts.GradientResolution = 300
flatOpts.RasterizationResolution = 300
docRef.PrintOut printOpts
```

PrintFontOptions

Contains information about font downloading and substitution for the fonts used for printing a document.

PrintFontOptions properties

Property	Value type	What it is
Application	Application object	Read-only. The Illustrator Application object.
DownloadFonts	AiPrintFontDownloadMode	The font download mode. Default: DownloadSubset
FontSubstitution	AiFontSubstitutionPolicy	The font substitution policy. Default: SubstituteOblique

➤ Setting print font options

```
' PrintFontOptions
' Create a new print font options object,
' change some attributes and print with it.

Dim appRef As New Illustrator.Application
Dim docRef As Illustrator.Document
Dim textRef As Illustrator.TextFrame
Set docRef = appRef.Documents.Add()
Set textRef = docRef.TextFrames.Add()
textRef.Top = 600
textRef.Left = 50
textRef.Contents = "PrintFontOptions object"

' Create a PrintFontOptions object and
' assign it to a PrintOptions object.
Dim fontOpts As New Illustrator.PrintFontOptions
Dim printOpts As New Illustrator.printOptions
printOpts.FontOptions = fontOpts

fontOpts.DownloadFonts = aiDownloadComplete
fontOpts.FontSubstitution = aiSubstituteOblique

' print it
docRef.PrintOut printOpts
```

PrintJobOptions

Contains information about how a job is to be printed.

PrintJobOptions properties

Property	Value type	What it is
Application	Application object	Read-only. The Illustrator Application object.
BitmapResolution	Double	The bitmap resolution. Minimum 0.0. Default: 0.0
Collate	Boolean	If true, collate print pages. Default: false
Copies	Long	The number of copies to print. Minimum: 1 Default: 1
Designation	AiPrintArtworkDesignation	The layers/objects to be printed. Default: VisiblePrintableLayers
File	String	The file to which to print.
Name	String	The print job name.
PrintArea	AiPrintingBounds	The printing bounds. Default: ArtboardBounds
PrintAsBitmap	Boolean	If true, print as bitmap. Default: false
ReversePages	Boolean	If true, print pages in reverse order. Default: false

➤ Printing with job options

```
' PrintJobOptions
' Create simple text items on a variety of
' Print/No-Print and Non-Visible layers.
' Print the document with various printJobOptions

Dim appRef As New Illustrator.Application
Dim docRef As Illustrator.Document
Set docRef = appRef.Documents.Add()

Dim textRef_1 As Illustrator.TextFrame
Dim textRef_2 As Illustrator.TextFrame
Dim textRef_3 As Illustrator.TextFrame
Dim layerRef_2 As Illustrator.Layer
Dim layerRef_3 As Illustrator.Layer

' create a visible, printable item
Set textRef_1 = docRef.Layers(1).TextFrames.Add()
textRef_1.Contents = "Visible and Printable"
textRef_1.Top = 600
textRef_1.Left = 200

' create a visible, non-printable item
Set layerRef_2 = docRef.Layers.Add()
Set textRef_2 = layerRef_2.TextFrames.Add()
textRef_2.Contents = "Visible and Non-Printable"
```

```
textRef_2.Top = 500
textRef_2.Left = 250
layerRef_2.Printable = False

' create a non-visible item
Set layerRef_3 = docRef.Layers.Add()
Set textRef_3 = layerRef_3.TextFrames.Add()
textRef_3.Contents = "Non-Visible"
textRef_3.Top = 400
textRef_3.Left = 300
layerRef_3.Visible = False
appRef.Redraw

' Print with various jobOptions
Dim jobOptionsRef As New Illustrator.PrintJobOptions
Dim printOpts As New Illustrator.printOptions
printOpts.JobOptions = jobOptionsRef

jobOptionsRef.Designation = aiAllLayers
jobOptionsRef.ReversePages = True
docRef.PrintOut printOpts

jobOptionsRef.Designation = aiVisibleLayers
jobOptionsRef.ReversePages = False
jobOptionsRef.PrintAsBitmap = True
docRef.PrintOut printOpts

jobOptionsRef.Designation = aiVisiblePrintableLayers
jobOptionsRef.File = "C:\temp\printJobTest1.ps"
docRef.PrintOut printOpts
```

PrintOptions

Collects all information about all printing options including flattening, color management, coordinates, fonts, and paper. Used as an argument to the PrintOut method.

PrintOptions properties

Property	Value type	What it is
Application	Application object	Read-only. The Illustrator Application object.
ColorManagementOptions	PrintColorManagementOptions object	The printing color management options.
ColorSeparationOptions	PrintColorSeparationOptions object	The printing color separation options.
CoordinateOptions	PrintCoordinateOptions object	The printing coordinate options.
FlattenerOptions	PrintFlattenerOptions object	The printing flattener options.
FlattenerPreset	String	The transparency flattener preset name.
FontOptions	PrintFontOptions object	The printing font options.
JobOptions	PrintJobOptions object	The printing job options.
PageMarksOptions	PrintPageMarksOptions object	The printing page marks options.
PaperOptions	PrintPaperOptions object	The paper options.
PostScriptOptions	PrintPostScriptOptions object	The PostScript options for printing.
PPDName	String	The PPD name.
PrinterName	String	The printer name.
PrintPreset	String	The print style.

➤ Setting print options

```
' PrintOptions
' Create a new document and add some symbol items
' Create a variety of print options and assign each
' to a PrintOptions object, then print the documennt
' with the combined PrintOptions object.

Dim appRef As New Illustrator.Application
Dim docRef As Illustrator.Document
Set docRef = appRef.Documents.Add()
Dim symbolRef As Illustrator.Symbol
Dim itemRef As Illustrator.SymbolItem
Dim y As Integer
dim i as Integer
```

```
y = docRef.Height - 30
i = 1
Do While (i < 11)
   Set symbolRef = docRef.Symbols(i)
   Set itemRef = docRef.SymbolItems.Add(symbolRef)
   itemRef.Top = y
   itemRef.Left = 100
   y = (y - (itemRef.Height + 10))
   i = i + 1
Loop
appRef.Redraw

' create multiple options and assign to PrintOptions object
Dim printOpts As New Illustrator.printOptions
Dim colorOpts As New Illustrator.PrintColorManagementOptions
colorOpts.Name = "ColorMatch RGB"
colorOpts.Intent = aiSaturationIntent
printOpts.ColorManagementOptions = colorOpts

Dim jobOpts As New Illustrator.PrintJobOptions
jobOpts.Designation = aiAllLayers
jobOpts.ReversePages = True
printOpts.JobOptions = jobOpts

Dim coordinateOpts As New Illustrator.PrintCoordinateOptions
coordinateOpts.FitToPage = True
printOpts.coordinateOptions = coordinateOpts

Dim flattenerOpts As New Illustrator.PrintFlattenerOptions
flattenerOpts.ClipComplexRegions = True
flattenerOpts.GradientResolution = 60
flattenerOpts.RasterizationResolution = 60
printOpts.FlattenerOptions = flattenerOpts

docRef.PrintOut printerOpts
```

PrintPageMarksOptions

The options for printing page marks.

PrintPageMarksOptions properties

Property	Value type	What it is
Application	Application object	Read-only. The Illustrator Application object.
BleedOffsetRect	Variant Array of 4 Doubles	The bleed offset rectangle.
ColorBars	Boolean	If true, enable color bars printing. Default: false
MarksOffsetRect	Variant Array of 4 Doubles	The page marks offset rectangle.
PageInfoMarks	Boolean	If true, enable page info marks printing. Default: false
PageMarksType	AiPageMarksStyle	The predefined page marks style name. Default: PageMarksStandard
RegistrationMarks	Boolean	If true, enable registration marks printing. Default: false
TrimMarks	Boolean	If true, enable trim marks printing. Default: false
TrimMarksWeight	Double	Stroke weight of trim marks. Minimum: 0.0 Default: 0.125

➤ Setting page mark options

```
' PrintPageMarksOptions
' Create a PrintPageMarksOptions object and assign it
' to a PrintOptions object.
' Set some attributes and print it.

' create a simple path item for printing
Dim appRef As New Illustrator.Application
Dim docRef As Illustrator.Document
Dim pathItem As Illustrator.pathItem
Set docRef = appRef.Documents.Add
Set pathItem = docRef.PathItems.Rectangle(600, 200, 200, 200)
docRef.GraphicStyles(2).ApplyTo pathItem

' Create a PrintCoordinateOptions object and assign it
' to a PrintOptions object
Dim pageMarksOpts As New Illustrator.PrintPageMarksOptions
pageMarksOpts.ColorBars = True
pageMarksOpts.PageInfoMarks = True
pageMarksOpts.RegistrationMarks = True
pageMarksOpts.TrimMarks = True

Dim printerOpts As New Illustrator.printOptions
printerOpts.PageMarksOptions = pageMarksOpts
docRef.PrintOut printerOpts
```

```
' print it
docRef.PrintOut printOpts
```

PrintPaperOptions

Contains information about the paper to be used in a print job.

PrintPaperOptions properties

Property	Value type	What it is
Application	Application	Read-only. The Illustrator Application object.
Height	Double	The custom height (in points) for using the custom paper. Default: 0.0
Name	String	The paper's name.
Offset	Double	Custom offset (in points) for using the custom paper. Default: 0.0
Transverse	Boolean	If true, transverse the artwork (rotate 90 degrees) on the custom paper. Default: false
Width	Double	The custom width in points, for using the custom paper. Default: 0.0

PrintPostScriptOptions

Specifies the options to be used when printing to a PostScript printer.

PrintPostScriptOptions properties

Property	Value type	What it is
Application	Application object	Read-only. The Illustrator Application object.
BinaryPrinting	Boolean	If true, print in binary mode. Default: false
CompatibleShading	Boolean	If true, use PostScript level 1 compatible gradient and gradient mesh printing. Default: false
ForceContinuousTone	Boolean	If true, force continuous tone. Default: false
ImageCompression	AiPostScriptImageCompressionType	The image compression type. Default: ImageCompressionNone
NegativePrinting	Boolean	If true, print in negative mode. Default: false
PostScriptLevel	AiPrinterPostScriptLevelEnum	The PostScript language level. Default: aiLevel2
ShadingResolution	Double	The shading resolution Range: 1.0 to 9600.0. Default: 300.0

➤ Setting PostScript options

```
' PrintPostScriptOptions
' Create a new PrintPostScriptOptions object,
' print with several PS levels

' Create a new document and add a TextFrame
Dim appRef As New Illustrator.Application
Dim docRef As Illustrator.Document
Dim textRef As Illustrator.TextFrame
Set docRef = appRef.Documents.Add()
Set textRef = docRef.TextFrames.Add()
textRef.Top = 600
textRef.Left = 50
textRef.Contents = "PrintPostScriptOptions object"

' Create a PrintPostScriptOptions object and
' assign it to a PrintOptions object.
Dim psOpts As New Illustrator.PrintPostScriptOptions
Dim printOpts As New Illustrator.printOptions
printOpts.PostScriptOptions = psOpts

' print with different PS levels
```

```
psOpts.PostScriptLevel = aiPSLevel2
docRef.PrintOut printOpts

psOpts.PostScriptLevel = aiPSLevel3
docRef.PrintOut printOpts
```

RasterItem

A bitmap art item in a document. A script can create raster items from an external file, and can create new raster items by copying and pasting or duplicating existing raster items.

RasterItem properties

Property	Value type	What it is
Application	Application object	Read-only. The Illustrator Application object.
ArtworkKnockout	AiKnockoutState	Is this object used to create a knockout, and if so, what kind.
BlendingMode	AiBlendModes	The mode used when compositing an object.
BoundingBox	Variant Array of 4 Doubles	Read-only. Dimensions of item regardless of transformations.
ContentVariable	Variable	The Variable bound to this item. It is not necessary to set the type before binding.
ControlBounds	Variant Array of 4 Doubles	Read-only. The bounds of the object including stroke width and controls.
Editable	Boolean	Read-only. If true, this item is editable.
Embedded	Boolean	If true, the Raster art can be embedded within the illustration.
File	String	The file containing the art, if stored externally.
GeometricBounds	Variant Array of 4 Doubles	Read-only. The bounds of the object excluding stroke width.
Height	Double	The height of the item, based on GeometricBounds.
Hidden	Boolean	If true, this item is hidden.
ImageColorSpace	AiImageColorSpace	The color space of the Raster image.
IsIsolated	Boolean	If true, this object is isolated.
Layer	Layer object	Read-only. The layer this item belongs to.
Left	Double	The left position of the item.
Locked	Boolean	If true, this item is locked
Matrix	Matrix object	The transformation matrix applied to the item.
Name	String	The name of this item.
Opacity	Double	The opacity of the object. Range: 0.0 to 100.0
Parent	object	Read-only. The object that contains this item.
Position	Variant Array of 2 Doubles	The position of the top left corner of the item.

Property	Value type	What it is
Selected	Boolean	If true, this item is selected.
Sliced	Boolean	If true, this item is sliced.
Status	AiRasterLinkState	The status of the linked image, if the image is stored externally.
Tags	Tags collection object	Read-only. The tags contained in this item.
Top	Double	The top position of the item.
URL	String	The value of the Adobe URL tag assigned to this item.
VisibilityVariable	Variable	The Variable bound to this item.
VisibleBounds	Variant Array of 4 Doubles	Read-only. The visible bounds of the item including stroke width.
Width	Double	The width of the item, based on GeometricBounds.
WrapInside	Boolean	If true, the text frame object should be wrapped inside this object.
WrapOffset	Double	The offset to use when wrapping text around this object.
Wrapped	Boolean	If true, the text frame objects should be wrapped around this object (text frame must be above the object).
ZOrderPosition	Long	Read-only. The position of this art item within the stacking order of the group or layer (Parent) that contains the art item.

RasterItem methods

Method	Returns	What it does
Colorize (rasterColor as **object**)	Nothing	Colorizes the RasterItem with a CMYK or RGB color.
Copy ()	Nothing	Copies the item to the clipboard; the associated document must be the frontmost document.
Cut ()	Nothing	Cuts the item to the clipboard; the associated document must be the frontmost document.
Delete ()	Nothing	Deletes the object.
Duplicate ([relativeObject as **Object**], [insertionLocation as AiElementPlacement])	Object	Duplicates the art item, optionally with the location and position for the copy.

Method	Returns	What it does
Move (relativeObject as **Object**, insertionLocation as AiElementPlacement)	Nothing	Moves the art item to a new location and position.
Resize (scaleX as **Double**, scaleY as **Double**, [, changePositions as **Boolean**] [, changeFillPatterns as **Boolean**] [, changeFillGradients as **Boolean**] [, changeStrokePattern as **Boolean**] [, changeLineWidths as **Double**] [, scaleAbout as AiTransformation])	Nothing	Scales the art item where scaleX is the horizontal scaling factor and scaleY is the vertical scaling factor; 100.0 = 100%
Rotate (angle as **Double**, [, changePositions as **Boolean**] [, changeFillPatterns as **Boolean**] [, changeFillGradients as **Boolean**] [, changeStrokePattern as **Boolean**] [, rotateAbout as AiTransformation])	Nothing	Rotates the art item relative to the current rotation. The object is rotated counter-clockwise if the Angle value is positive, clockwise if the value is negative.
Trace ()	PluginItem object	Converts the raster art for this object to vector art, using default options. Reorders the raster art into the source art of a plugin group, and converts it into a group of filled and/or stroked paths that resemble the original image. Creates and returns a PluginItem object that references a TraceObject object.
Transform (transformationMatrix as **Matrix**, [, changePositions as **Boolean**] [, changeFillPatterns as **Boolean**] [, changeFillGradients as **Boolean**] [, changeStrokePattern as **Boolean**] [, changeLineWidths as **Double**] [, transformAbout as AiTransformation])	Nothing	Transforms the art item by applying a transformation matrix.
Translate ([deltaX as **Double**] [, deltaY as **Double**] [, transformObjects as **Boolean**] [, transformFillPatterns as **Boolean**] [, transformFillGradients as **Boolean**] [, transformStrokePatterns as **Boolean**])	Nothing	Repositions the art item relative to the current position, where deltaX is the horizontal offset and deltaY is the vertical offset.
ZOrder (zOrderCmd as AiZOrderMethod)	Nothing	Arranges the art item's position in the stacking order of the group or layer (Parent) of this object.

RasterItems

A collection of `RasterItem` objects.

RasterItems properties

Property	Value type	What it is
Application	Application	Read-only. The Illustrator Application object.
Count	Long	Read-only. Number of elements in the collection.
Parent	Object	Read-only. The object's container.

RasterItems methods

Method	Returns	What it does
Add ()	RasterItem object	Creates a new RasterItem object.
Index (item as **RasterItem**)	Long	Returns the index position of the object within the collection.
Item (itemKey)	RasterItem object	Returns an object reference to the object identified by itemKey (name or index).
RemoveAll ()	Nothing	Deletes all objects in this collection.

➤ Getting properties of raster items

This script illustrates how to obtain the color space of a raster item.

```
Dim appRef As New Illustrator.Application
Dim rasterArt As Illustrator.RasterItem
Set rasterArt = appRef.ActiveDocument.RasterItems(1)
Select Case (rasterArt.ImageColorSpace)
   Case Is = aiImageCMYK
      MsgBox "The first raster item is a CMYK raster item"
   Case Is = aiImageRGB
      MsgBox "The first raster item is an RGB raster item"
   Case Is = aiImageGrayScale
      MsgBox "The first raster item is a Grayscale raster item"
```

RGBColor

An RGB color specification, used to apply an RGB color to a layer or art item.

If the color space of a document is `RGB` and you specify the color value for a page item in that document using CMYK, Illustrator will translate the CMYK color specification into an RGB color specification. The same thing happens if the document's color space is CMYK and you specify colors using RGB. Since this translation can lose information, you should specify colors using the class that matches the document's actual color space.

RGBColor properties

Property	Value type	What it is
`Application`	`Application`	Read-only. The Illustrator `Application` object.
`Blue`	`Double`	The blue color value. Range: 0.0 to 255.0
`Green`	`Double`	The green color value. Range: 0.0 to 255.0
`Red`	`Double`	The red color value. Range: 0.0 to 255.0

➤ Setting RGB colors

This script sets the default fill color of the frontmost document to yellow using an RGB object. If the color space of the frontmost document is CMYK, then Illustrator will regard the RGB fill color as a CMYK color although it is specified using RGB.

```
Dim appRef As New Illustrator.Application
Dim newFillColor As New Illustrator.RGBColor

'Define the new color
newFillColor.Red = 255#
newFillColor.Green = 255#
newFillColor.Blue = 0
```

Screen

Associates a color separation screen with information to be used for printing.

Screen properties

Property	Value type	What it is
Application	Application object	Read-only. The Illustrator Application object.
Name	String	The color separation screen.
ScreenInfo	ScreenInfo object	The color separation screen information.

ScreenInfo

Contains information about the angle and frequency of the screen to be used for printing.

ScreenInfo properties

Property	Value type	What it is
Angle	Double	The screen's angle in degrees.
Application	Application object	Read-only. The Illustrator Application object.
DefaultScreen	Boolean	If true, it is the default screen.
Frequency	Double	The screen's frequency.

➤ Getting screen information

```
' Screen
' Display screen info available for first PPD ojbect

' Create a new document and add a TextFrame
Dim appRef As New Illustrator.Application
Dim docRef As Illustrator.Document
Dim textRef As Illustrator.TextFrame
Set docRef = appRef.Documents.Add()
Set textRef = docRef.TextFrames.Add()
textRef.Top = 600
textRef.Left = 50
textRef.Contents = "Screen Objects for 1st PPD File: " & vbCrLf

' Get the first PPD
Dim ppdRef As Illustrator.PPDFile
Set ppdRef = appRef.PPDFileList(1)
Dim sInfo As String
sInfo = ppdRef.Name & vbCrLf

' Get info on the all screens from the 1st PPD
Dim screenRef
```

```
For Each screenRef In ppdRef.PPDInfo.ScreenList
   sInfo = sInfo & screenRef.Name & vbCrLf
   sInfo = sInfo & vbTab & "Angle = "
   sInfo = sInfo & CStr(screenRef.ScreenInfo.Angle) & vbCrLf
   sInfo = sInfo & vbTab & "Frequency = "
   sInfo = sInfo & screenRef.ScreenInfo.Frequency & vbCrLf
Next screenRef

textRef.Contents = textRef.Contents & sInfo
appRef.Redraw
```

ScreenSpotFunction

Contains information about the screen spot function including its definition in PostScript language code.

ScreenSpotFunction properties

Property	Value type	What it is
Application	Application object	Read-only. The Illustrator Application object.
Name	String	The color separation screen spot function name.
SpotFunction	String	The spot function expressed in PostScript language commands.

➤ Getting screen spot information

```
' ScreenSpotFunction
' Display ScreenSpot info info available for first PPD ojbect

' Create a new document and add a TextFrame
Dim appRef As New Illustrator.Application
Dim docRef As Illustrator.Document
Dim textRef As Illustrator.TextFrame
Set docRef = appRef.Documents.Add()
Set textRef = docRef.TextFrames.Add()
textRef.Top = 600
textRef.Left = 50
textRef.Contents = "ScreenSpotFunctions for 1st PPD:" & vbCrLf

' Get the first PPD
Dim ppdRef As Illustrator.PPDFile
Set ppdRef = appRef.PPDFileList(1)
Dim sInfo As String
sInfo = ppdRef.Name & vbCrLf

' Get info on the all ScreenSpot info from the 1st PPD
Dim screenSpot
For Each screenSpot In ppdRef.PPDInfo.ScreenSpotFunctionList
   sInfo = sInfo + screenSpot.Name & ": "
   sInfo = sInfo + screenSpot.SpotFunction & vbCrLf
   sInfo = sInfo & vbCrLf
Next screenSpot

textRef.Contents = textRef.Contents + sInfo
appRef.Redraw
```

Spot

A custom color definition contained in a `SpotColor` object. All Illustrator documents contain the spot color `[Registration]` which can be used to print to all plates of a separation.

If no properties are specified when creating a new spot, default values are provided. However, if specifying the color, you must use the same color space as the document, either CMYK or RGB. Otherwise, an error results. When created, the spot is inserted into the swatch palette at the end.

Spot properties

Property	Value type	What it is
`Application`	`Application object`	Read-only. The Illustrator `Application` object.
`Color`	`Color object`	The color information for this spot color.
`ColorType`	`AiColorModel`	The color model for this spot color.
`Name`	`String`	The spot color's name.
`Parent`	`Document object`	Read-only. The document that contains this spot color.

Spot methods

Method	Returns	What it does
`Delete` `()`	`Nothing`	Deletes the object.

➤ Creating a new spot

This script illustrates how to create a new spot in the frontmost document.

```
Dim appRef As New Illustrator.Application
Dim frontDocument As Illustrator.Document
Dim newSpot As Illustrator.Spot
Dim newSpotColor As Illustrator.SpotColor
'Create the new spot
Set frontDocument = appRef.ActiveDocument
Set newSpot = frontDocument.Spots.Add
'Define the new color value
Dim newColor As New Illustrator.CMYKColor
newColor.Cyan = 35
newColor.Magenta = 0
newColor.Yellow = 50
newColor.Black = 0
'Define a new SpotColor with an 80% tint
'of the new Spot's color. The spot color can then
'be applied to an art item like any other color.
newSpot.Name = "Pea-Green"
newSpot.ColorType = aiSpot
newSpot.Color = newColor
Set newSpotColor = New Illustrator.SpotColor
newSpotColor.Spot = newSpot
```

SpotColor

Color class used to apply the color value of a spot at a specified tint value.

SpotColor properties

Property	Value type	What it is
Application	Application object	Read-only. The Illustrator Application object.
Spot	Spot object	A reference to the Spot object that defines the color.
Tint	Double	The tint of the color. Range: 0.0 to 100.0

➤ Applying a tint to a spot

This script shows how to create a new spot, and apply a tint of that spot to the fill of a PathItem.Your frontmost document must already have a PathItem before you can run this script.

```
Dim appRef As New Illustrator.Application
Dim frontDocument As Illustrator.Document
Dim newColor As New Illustrator.RGBColor
Dim newSpot As Illustrator.Spot
Dim frontPath As Illustrator.PathItem

'Define the new color value
newColor.Red = 255#
newColor.Green = 0
newColor.Blue = 0

'Create the new spot
Set frontDocument = appRef.ActiveDocument
Set newSpot = frontDocument.Spots.Add

'Define the new SpotColor as 80% of the specified RGB color
newSpot.Name = "Red spot color"
newSpot.Color = newColor

'Apply 50% of the spot color just created to the frontmost path item,
'by creating a SpotColor object, setting its properties,
'then applying that to a path item's fill color.
Dim newSpotColor As New Illustrator.SpotColor
newSpotColor.Spot = newSpot
newSpotColor.Tint = 50#

Set frontPath = frontDocument.PathItems(1)
frontPath.Filled = True
```

Spots

A collection of spot colors in a document.

Spots properties

Property	Value type	What it is
Application	Application	Read-only. The Illustrator Application object.
Count	Long	Read-only. Number of elements in the collection.
Parent	Document	Read-only. The object's document container.

Spots methods

Method	Returns	What it does
Add ()	Spot object	Creates a new Spot object.
Index (item as **Spot**)	Long	Returns the index position of the object within the collection.
Item (itemKey)	Spot object	Returns an object reference to the object identified by itemKey (name or index).
RemoveAll ()	Nothing	Deletes all objects in this collection.

➤ Removing spots from a document

This script illustrates how to remove all spots defined in the frontmost document.

```
Dim appRef As New Illustrator.Application
Dim documentSpots As Illustrator.Spots
Set documentSpots = appRef.ActiveDocument.Spots
documentSpots.RemoveAll
```

Story

A contiguous block of text as specified by a text range. A story can contain one or more text frames; if there is more than one, the multiple text frames are linked together to form a single story.

Story properties

Property	Value Type	What it is
Application	Application object	Read-only. The Illustrator Application object.
Characters	Characters	Read-only. All of the characters in the story.
InsertionPoints	InsertionPoints	Read-only. All of the insertion points in this story.
Length	Long	Read-only. The number of characters in the story.
Lines	Lines	Read-only. All of the lines in this story.
Paragraphs	Paragraphs	Read-only. All of the paragraphs in this story.
Parent	Object	Read-only. The object's container.
TextFrames	TextFrames	Read-only. The text frame items in this story.
TextRange	TextRange	Read-only. The text range of the story.
TextRanges	TextRanges	Read-only. All of the text ranges in the story.
TextSelection	Variant Array of TextRange objects	Read-only. The selected text ranges in the story.
Words	Words	Read-only. All of the words in the story.

➤ Linking text frames into a story

```
' Story
' Create two TextFrames and link
' the 2nd to the first to create a single story.

Dim appRef As New Illustrator.Application
Dim docRef As Illustrator.Document
Set docRef = appRef.Documents.Add()
Dim storyRef1 As Illustrator.Story
Dim storyRef2 As Illustrator.Story
Dim textRef1 As Illustrator.TextFrame
Dim textRef2 As Illustrator.TextFrame
Dim itemRef1 As Illustrator.pathItem
Dim itemRef2 As Illustrator.pathItem

' Create the first text frame
Set itemRef1 = docRef.PathItems.Rectangle(600, 200, 50, 30)
Set textRef1 = docRef.TextFrames.AreaText(itemRef1)
textRef1.Selected = True

' create the 2nd text frame and
```

```
' link it to the first
Set itemRef2 = docRef.PathItems.Rectangle(550, 300, 50, 200)
Set textRef2 = docRef.TextFrames.AreaText(itemRef2, aiHorizontal, textRef1)
textRef1.Contents = "This is two text frames linked together as one story"
textRef2.Selected = True
appRef.Redraw
MsgBox "There is " & CStr(docRef.Stories.Count) & " story"

' Create a 3rd text frame and count the stories
Dim textRef3 As Illustrator.TextFrame
Set textRef3 = docRef.TextFrames.Add
textRef3.Contents = "Each unlinked textFrame adds a story."
textRef3.Top = 650
textRef3.Left = 200
appRef.Redraw
MsgBox "Now there are " & CStr(docRef.Stories.Count) & " stories"
```

Stories

A collection of `Story` objects. See Story for an example.

Stories properties

Property	Value type	What it is
`Application`	`Application`	Read-only. The Illustrator `Application` object.
`Count`	`Long`	Read-only. Number of elements in the collection.
`Parent`	`Document`	Read-only. The object's document container.

Stories methods

Method	Returns	What it does
`Index` (`item as` **`Story`**)	`Long`	Returns the index position of the object within the collection.
`Item` (`itemKey`)	`Story object`	Returns an object reference to the object identified by `itemKey` (name or index).

Swatch

A color swatch definition contained in a document. The swatches correspond to the swatch palette in the Illustrator user interface. Additional swatches can be created either manually by a user or by a script. The swatch can hold all types of color data (that is, pattern, gradient, CMYK, RGB, gray, and spot).

Swatch properties

Property	Value type	What it is
Application	Application object	Read-only. The Illustrator Application object.
Color	Color object	The color information for this swatch.
Name	String	The swatch's name.
Parent	Document object	Read-only. The document that contains this swatch.

Swatch methods

Method	Returns	What it does
Delete ()	Nothing	Deletes the object.

➤ Changing the name of a swatch

This script illustrates how to change the name of the fifth swatch.

```
Dim appRef As New Illustrator.Application
Dim swatch5 As Illustrator.Swatch
Set swatch5 = appRef.ActiveDocument.Swatches(5)
swatch5.Name = "myColor"
```

Swatches

A collection of `swatch` objects in a document.

Swatches properties

Property	Value type	What it is
Application	Application	Read-only. The Illustrator Application object.
Count	Long	Read-only. Number of elements in the collection.
Parent	Document	Read-only. The object's document container.

Swatches methods

Method	Returns	What it does
Add ()	Swatch object	Creates a new Swatch object.
Index (item as **Swatch**)	Long	Returns the index position of the object within the collection.
Item (itemKey)	Swatch object	Returns an object reference to the object identified by itemKey (name or index).
RemoveAll ()	Nothing	Deletes all objects in this collection.

➤ Deleting a swatch

This script illustrates how to first obtain a swatch by index and then how to delete that swatch.

```
Dim appRef As New Illustrator.Application
Dim swatchToDelete As Illustrator.Swatch
Dim documentSwatches As Illustrator.Swatches

Set documentSwatches = appRef.ActiveDocument.Swatches
Set swatchToDelete = documentSwatches("Orange")
swatchToDelete.delete
```

Symbol

An art item that is stored in the Symbols Palette, and can be reused one or more times in the document without duplicating the art data. Symbols are contained in documents. Instances of `Symbol` in a document are associated with `SymbolItem` objects, which store the art-object properties. See Symbols and SymbolItem.

Symbol properties

Property	Value type	What it is
`Application`	`Application object`	Read-only. The Illustrator `Application` object.
`Name`	`String`	The symbol's name.
`Parent`	`Document object`	Read-only. The document that contains this symbol.

Symbol methods

Method	Returns	What it does
`Delete` `()`	`Nothing`	Deletes the object.
`Duplicate` `()`	`object`	Creates a duplicate of the object.

Symbols

A collection of `symbol` objects in a document.

Symbols properties

Property	Value type	What it is
Application	Application	Read-only. The Illustrator Application object.
Count	Long	Read-only. Number of elements in the collection.
Parent	Document	Read-only. The object's document container.

Symbols methods

Method	Returns	What it does
Add (sourceArt as **Object**)	Symbol object	Creates a new Symbol object in the document based on the art item.
Index (item as **Symbol**)	Long	Returns the index position of the object within the collection.
Item (itemKey)	Symbol object	Returns an object reference to the object identified by itemKey (name or index).
RemoveAll ()	Nothing	Deletes all objects in this collection.

➤ Creating symbols

```
' Symbols
' Create a simple path item from each graphic style
' and then add the item as a new symbol

Dim appRef As New Illustrator.Application
Dim docRef As Illustrator.Document
Set docRef = appRef.Documents.Add()
Dim y As Integer
y = docRef.Height - 30

Dim iCount As Integer
iCount = docRef.GraphicStyles.Count
MsgBox "There are " + CStr(iCount) + " symbols."

Dim styleRef As Illustrator.GraphicStyle
Dim itemRef As Illustrator.pathItem
Dim symbolRef As Illustrator.Symbol

Dim i As Integer
i = 2
Do While (i < (iCount + 1))
   ' create a PathItem and apply the style to it
   Set styleRef = docRef.GraphicStyles(i)
```

```
    Set itemRef = docRef.PathItems.Rectangle(y, 100, 20, 20)
    styleRef.ApplyTo itemRef

    ' create a new symbol from the graphic style
    docRef.Symbols.Add itemRef

    y = (y - (itemRef.Height + 40)) ' reduce y for next item
    i = i + 1
Loop

appRef.Redraw
MsgBox "There are now " + CStr(docRef.Symbols.Count) + " symbols."
```

SymbolItem

An art item made reuseable by adding it to the Symbols palette. A `SymbolItem` is linked to the `Symbol` from which it was created and changes if you modify the associated `Symbol` object.

SymbolItem properties

Property	Value type	What it is
Application	`Application object`	Read-only. The Illustrator `Application` object.
ArtworkKnockout	`AiKnockoutState`	Is this object used to create a knockout, and if so, what kind.
BlendingMode	`AiBlendModes`	The mode used when compositing an object.
ControlBounds	`Variant Array of 4 Doubles`	Read-only. The bounds of the object including stroke width and controls.
Editable	`Boolean`	Read-only. If `true`, this item is editable.
GeometricBounds	`Variant Array of 4 Doubles`	Read-only. The bounds of the object excluding stroke width.
Height	`Double`	The height of the item, based on `GeometricBounds`.
Hidden	`Boolean`	If `true`, this item is hidden.
IsIsolated	`Boolean`	If `true`, this object is isolated.
Layer	`Layer object`	Read-only. The layer this item belongs to.
Left	`Double`	The left position of the item.
Locked	`Boolean`	If `true`, this item is locked
Name	`String`	The name of this item.
Opacity	`Double`	The opacity of the object. Range: 0.0 to 100.0
Parent	`object`	Read-only. The object that contains this item.
Position	`Variant Array of 2 Doubles`	The position of the top left corner of the item.
Selected	`Boolean`	If `true`, this item is selected.
Sliced	`Boolean`	If `true`, this item is sliced.
Symbol	`Symbol object`	The symbol object to which this item is linked. You can set this property to link it to a different symbol object.
Tags	`Tags collection object`	Read-only. The tags contained in this item.
Top	`Double`	The top position of the item.
URL	`String`	The value of the Adobe URL tag assigned to this item.
VisibilityVariable	`Variable`	The Variable bound to this item.

Property	Value type	What it is
VisibleBounds	Variant Array of 4 Doubles	Read-only. The visible bounds of the item including stroke width.
Width	Double	The width of the item, based on GeometricBounds.
WrapInside	Boolean	If true, the text frame object should be wrapped inside this object.
WrapOffset	Double	The offset to use when wrapping text around this object.
Wrapped	Boolean	If true, the text frame objects should be wrapped around this object (text frame must be above the object).
ZOrderPosition	Long	Read-only. The position of this art item within the stacking order of the group or layer (Parent) that contains the art item.

SymbolItem methods

Method	Returns	What it does
Copy ()	Nothing	Copies the item to the clipboard; the associated document must be the frontmost document.
Cut ()	Nothing	Cuts the item to the clipboard; the associated document must be the frontmost document.
Delete ()	Nothing	Deletes the object.
Duplicate ([relativeObject as **Object**], [insertionLocation as AiElementPlacement])	Object	Duplicates the art item, optionally with the location and position for the copy.
Move (relativeObject as **Object**, insertionLocation as AiElementPlacement)	Nothing	Moves the art item to a new location and position.
Resize (scaleX as **Double**, scaleY as **Double**, [, changePositions as **Boolean**] [, changeFillPatterns as **Boolean**] [, changeFillGradients as **Boolean**] [, changeStrokePattern as **Boolean**] [, changeLineWidths as **Double**] [, scaleAbout as AiTransformation])	Nothing	Scales the art item where scaleX is the horizontal scaling factor and scaleY is the vertical scaling factor; 100.0 = 100%

Method	Returns	What it does
Rotate (angle as **Double**, [, changePositions as **Boolean**] [, changeFillPatterns as **Boolean**] [, changeFillGradients as **Boolean**] [, changeStrokePattern as **Boolean**] [, rotateAbout as AiTransformation])	Nothing	Rotates the art item relative to the current rotation. The object is rotated counter-clockwise if the Angle value is positive, clockwise if the value is negative.
Transform (transformationMatrix as **Matrix**, [, changePositions as **Boolean**] [, changeFillPatterns as **Boolean**] [, changeFillGradients as **Boolean**] [, changeStrokePattern as **Boolean**] [, changeLineWidths as **Double**] [, transformAbout as AiTransformation])	Nothing	Transforms the art item by applying a transformation matrix.
Translate ([deltaX as **Double**] [, deltaY as **Double**] [, transformObjects as **Boolean**] [, transformFillPatterns as **Boolean**] [, transformFillGradients as **Boolean**] [, transformStrokePatterns as **Boolean**])	Nothing	Repositions the art item relative to the current position, where deltaX is the horizontal offset and deltaY is the vertical offset.
ZOrder (zOrderCmd as AiZOrderMethod)	Nothing	Arranges the art item's position in the stacking order of the group or layer (Parent) of this object.

➤ Getting symbol items for symbols

```
' SymbolItems
' Add a symbol item to the document for every available.symbol

Dim appRef As New Illustrator.Application
Dim docRef As Illustrator.Document
Set docRef = appRef.Documents.Add()

Dim x As Integer
Dim y As Integer
y = docRef.Height - 30
x = 50

Dim symbolRef As Illustrator.Symbol
Dim itemRef As Illustrator.SymbolItem
Dim i As Integer
i = 1
Do While (i < (docRef.Symbols.Count + 1))
   Set symbolRef = docRef.Symbols(i)
   Set itemRef = docRef.SymbolItems.Add(symbolRef)
   itemRef.Top = y
   itemRef.Left = x
   y = y - (itemRef.Height + 20)
   If (y < 60) Then
      y = docRef.Height - 30
      x = x + 200
```

```
    End If
    i = i + 1
Loop
```

SymbolItems

A collection of `SymbolItem` objects in a document. See SymbolItem for example.

SymbolItems properties

Property	Value type	What it is
Application	`Application`	Read-only. The Illustrator `Application` object.
Count	`Long`	Read-only. Number of elements in the collection.
Parent	`object`	Read-only. The object's container.

SymbolItems methods

Method	Returns	What it does
Add (`symbol as` **Symbol**)	`SymbolItem object`	Creates a new `SymbolItem` object in the document linked to the given symbol.
Index (`item as` **SymbolItem**)	`Long`	Returns the index position of the object within the collection.
Item (`itemKey`)	`SymbolItem object`	Returns an object reference to the object identified by `itemKey` (name or index).
RemoveAll ()	`Nothing`	Deletes all objects in this collection.

TabStopInfo

Information about the alignment, position, and other details for a tab stop in a `ParagraphAttributes` object.

TabStopInfo properties

Property	Value type	What it is
Alignment	AiTabStopAlignment	The alignment of the tab stop. Default: `LeftTab`
Application	`Application object`	Read-only. The Illustrator `Application` object.
DecimalCharacter	`String`	The character used for decimal tab stops. Default: "."
Leader	`String`	The leader dot.
Position	`Double`	The position of the tab stop expressed in points. Default: 0.0

➤ Getting tab stop information

```
' TabStopInfo
' Check the current document for textFrames.
' Display information about each TabStop found.

' verify a document is open
Dim appRef As New Illustrator.Application
If (appRef.Documents.Count < 1) Then
   MsgBox "open a document that has paragraphs that contains TabStops."
Else
   Dim docRef As Illustrator.Document
   Set docRef = appRef.ActiveDocument
   If (docRef.TextFrames.Count < 1) Then
      MsgBox "open a document that has paragraphs that contains TabStops."
   Else
      Dim sData As String
      Dim textRef As Illustrator.TextFrame
      Dim paraRef As Illustrator.TextRange
      Dim tabRef
      For Each textRef In docRef.TextFrames
         ' check each TextFrame for paragraphs
         For Each paraRef In textRef.Paragraphs
            ' check each paragraph for TabStops
            For Each tabRef In paraRef.ParagraphAttributes.TabStops
               sData = "TabStop Leader = " & tabRef.Leader & vbCrLf
               sData = sData & "TabStop Position = " & CStr(tabRef.Position)
               MsgBox sData, vbApplicationModal, "TabStop Found"
            Next tabRef
         Next paraRef
      Next textRef
   End If
End If
```

Tag

A tag associated with a specific art item. Tags allow you to assign an unlimited number of key-value pairs to any item in a document.

Tag properties

Property	Value type	What it is
Application	Application object	Read-only. The Illustrator Application object.
Name	String	The Tag's name.
Parent	Object	Read-only. The object that contains this Tag.
Value	String	The data stored in this Tag.

Tag methods

Method	Returns	What it does
Delete ()	Nothing	Deletes the object.

➤ Using tags

This example illustrates how to list the tags associated with the first selected item. The name and value of the tags are listed in a new document.

```
Dim appRef As New Illustrator.Application
Dim reportDocument As Illustrator.Document
Dim selection As Variant
Dim selectedArt As Object
Dim tagList As Illustrator.Tags
Dim tagItem As Illustrator.Tag
Dim top_offset As Single

selection = appRef.selection
If (Not IsEmpty(selection)) Then
   'Get the first selected item
   Set selectedArt = selection(0)
   Set tagList = selectedArt.Tags
   If (tagList.Count = 0) Then
      MsgBox "The selected art has no tags"
      Exit Sub
   End If

   'Create a document and add a line of text per tag
   Set reportDocument = appRef.Documents.Add
   top_offset = 400
   For Each tagItem In tagList
      Dim tagText As Illustrator.TextFrame
      Set tagText = reportDocument.TextFrames.Add
      tagText.Contents = "Tag: (" & tagItem.Name & " _
         , " & tagItem.Value & ")"
```

```
        tagText.Position = Array(100, top_offset)
        top_offset = top_offset - 20
    Next
End If
```

Tags

A collection of `Tag` objects.

Tags properties

Property	Value type	What it is
`Application`	`Application`	Read-only. The Illustrator `Application` object.
`Count`	`Long`	Read-only. Number of elements in the collection.
`Parent`	`object`	Read-only. The object's container.

Tags methods

Method	Returns	What it does
`Add` `()`	`Tag object`	Creates a new `Tag` object.
`Index` `(item as` **`Tag`**`)`	`Long`	Returns the index position of the object within the collection.
`Item` `(itemKey)`	`Tag object`	Returns an object reference to the object identified by `itemKey` (name or index).
`RemoveAll` `()`	`Nothing`	Deletes all objects in this collection.

➤ Creating tags to mark images

This example illustrates how to mark all images in a document with a specific tag. If your script creates temporary `PageItems`, you can then later look at the "MyInfo" tag. If it exist for a particular `PageItem` and it's value is "OriginalItem" you know not to delete it.

```
Dim appRef As New Illustrator.Application
Dim frontDocument As Illustrator.Document
Dim imageArt As Object
Dim tagRef As Illustrator.Tag

Set frontDocument = appRef.ActiveDocument
For Each imageArt In frontDocument.PageItems
   If ((imageArt.PageItemType = aiPlacedItem) Or _
      (imageArt.PageItemType = aiRasterItem)) Then

      'Create a new Tag with the name myInfo and the value originalItem
      Set tagRef = imageArt.Tags.Add
      tagRef.Name = "MyInfo"
      tagRef.Value = "OriginalItem"
   End If
```

TextFont

Information about a font in the document, found in a CharacterAttributes object.

TextFont properties

Property	Value type	What it is
Application	Application	Read-only. The Illustrator Application object.
Family	String	Read-only. The font's family name.
Name	String	Read-only. The font's full name.
Parent	Object	Read-only. The object's container.
Style	String	Read-only. The font's style name.

➤ Showing fonts in a document

```
' TextFont, TextFonts
' Create a new document, list all available fonts.

Dim appRef As New Illustrator.Application
Dim docRef As Illustrator.Document
Set docRef = appRef.Documents.Add()

Dim x As Integer
Dim y As Integer
x = 30
y = docRef.Height - 30

Dim fontRef
Dim textRef As Illustrator.TextFrame
For Each fontRef In appRef.TextFonts
   Set textRef = docRef.TextFrames.Add()
   textRef.TextRange.CharacterAttributes.Size = 8
   textRef.Contents = fontRef.Name + " " + fontRef.Style
   textRef.Top = y
   textRef.Left = x
   appRef.Redraw

   y = y - textRef.Height
   If (y < 30) Then    'reset y,x at bottom of page
      y = docRef.Height - 30
      x = x + 150
   End If
Next fontRef
```

TextFonts

A collection of `TextFont` objects.

TextFonts properties

Property	Value type	What it is
Application	Application	Read-only. The Illustrator `Application` object.
Count	Long	Read-only. Number of elements in the collection.
Parent	object	Read-only. The object's container.

TextFonts methods

Method	Returns	What it does
Index (item as **TextFont**)	Long	Returns the index position of the object within the collection.
Item (itemKey)	TextFont object	Returns an object reference to the object identified by `itemKey` (name or index).

➤ **Finding a font**

```
Dim appRef As New Illustrator.Application
Dim fontToTest As Illustrator.TextFont
Dim foundTextFace As Boolean
Dim fontName As String

fontName = "Symbol"
foundTextFace = False
For Each fontToTest In appRef.TextFonts
   If (fontToTest.Name = fontName) Then
      foundTextFace = True
      Exit For
   End If
Next

If (foundTextFace) Then
   MsgBox fontName & " is installed on this machine"
Else
   MsgBox fontName & " is not installed on this machine"
```

TextFrame

A text frame contained in a `Story`; there may be more than one `TextFrame` if the text is area text.

TextFrame properties

Property	Value type	What it is
Anchor	`VariantArray of 2 Doubles`	The position of the anchor point along the path text.
Application	`Application object`	Read-only. The Illustrator `Application` object.
ArtworkKnockout	`AiKnockoutState`	Is this object used to create a knockout, and if so, what kind.
BlendingMode	`AiBlendModes`	The mode used when compositing an object.
Characters	`Characters`	Read-only. All the characters in this text range.
ColumnCount	`Long`	The column count in the text frame (area text only).
ColumnGutter	`Double`	The column gutter in the text frame (area text only).
Contents	`String`	The text contents of this text frame.
ContentVariable	`Variable`	The content variable bound to this text frame.
ControlBounds	`Variant Array of 4 Doubles`	Read-only. The bounds of the object including stroke width and controls.
Editable	`Boolean`	Read-only. If `true`, this item is editable.
EndTValue	`Double`	The end position of text along a path, as a value relative to the path's segments (path text only).
FlowsLinkHorizontally	`Boolean`	If `true`, text is first flowed horizontally between linked frames.
GeometricBounds	`Variant Array of 4 Doubles`	Read-only. The bounds of the object excluding stroke width.
Height	`Double`	The height of the item, based on `GeometricBounds`.
Hidden	`Boolean`	If `true`, this item is hidden.
InsertionPoints	`InsertionPoints`	Read-only. All the insertion points in this text range.
IsIsolated	`Boolean`	If `true`, this object is isolated.
Kind	`AiTextType`	Read-only. The type of a text frame item.
Layer	`Layer object`	Read-only. The layer this item belongs to.
Left	`Double`	The left position of the item.
Lines	`Lines`	Read-only. All the lines in this text range.

Property	Value type	What it is
Locked	Boolean	If true, this item is locked.
Matrix	Matrix	Read-only. The transformation matrix for this text frame.
Name	String	The name of this item.
NextFrame	TextFrame	The linked frame following this one.
Opacity	Double	The opacity of the object. Range: 0.0 to 100.0
OpticalAlignment	Boolean	If true, the optical alignment is active.
Orientation	AiTextOrientation	The orientation of the text in the frame.
Paragraphs	Paragraphs	Read-only. All the paragraphs in this text range.
Parent	object	Read-only. The object that contains this item.
Position	Variant Array of 2 Doubles	The position of the top left corner of the item.
PreviousFrame	TextFrame	The linked text frame preceding this one.
RowCount	Long	The row count in the text frame (area text only).
RowGutter	Double	The row gutter in the text frame (area text only).
Selected	Boolean	If true, this item is selected.
Sliced	Boolean	If true, this item is sliced.
Spacing	Double	The amount of spacing.
StartTValue	Double	The start position of text along a path, as a value relative to the path's segments (path text only).
Story	Story	Read-only. The story of the text frame.
Tags	Tags collection object	Read-only. The tags contained in this item.
TextPath	TextPath	The path points defining the path for the text frame (area and path text).
TextRange	TextRange	Read-only. The text range of the text frame.
TextRanges	TextRanges	Read-only. All the text in this text range.
TextSelection	Variant Array of TextRange	The selected text (ranges) in the story.
Top	Double	The top position of the item.
URL	String	The value of the Adobe URL tag assigned to this item.
VisibilityVariable	Variable	The Variable bound to this item.
VisibleBounds	Variant Array of 4 Doubles	Read-only. The visible bounds of the item including stroke width.

Property	Value type	What it is
Width	Double	The width of the item, based on GeometricBounds.
Words	Words	Read-only. All the words in this text range.
WrapInside	Boolean	If true, the text frame object should be wrapped inside this object.
WrapOffset	Double	The offset to use when wrapping text around this object.
Wrapped	Boolean	If true, the text frame objects should be wrapped around this object (text frame must be above the object).
ZOrderPosition	Long	Read-only. The position of this art item within the stacking order of the group or layer (Parent) that contains the art item.

TextFrame methods

Method	Returns	What it does
CreateOutline ()	GroupItem object	Creates an outline for the frame text.
Duplicate ([relativeObject as **Object**] [, insertionLocation as AiElementPlacement])	Object	Duplicates the art item, optionally specifying the location and position for the copy.
Move (relativeObject as **Object**, insertionLocation as AiElementPlacement)	Nothing	Moves the art item, specifying the new location and position.
Remove ()	Nothing	Deletes this object.
Resize (scaleX as **Double**, scaleY as **Double**, [, changePositions as **Boolean**] [, changeFillPatterns as **Boolean**] [, changeFillGradients as **Boolean**] [, changeStrokePattern as **Boolean**] [, changeLineWidths as **Double**] [, scaleAbout as AiTransformation])	Nothing	Scales the art item where scaleX is the horizontal scaling factor and scaleY is the vertical scaling factor; 100.0 = 100%
Rotate (angle as **Double**, [, changePositions as **Boolean**] [, changeFillPatterns as **Boolean**] [, changeFillGradients as **Boolean**] [, changeStrokePattern as **Boolean**] [, rotateAbout as AiTransformation])	Nothing	Rotates the art item relative to the current rotation. The object is rotated counter-clockwise if the Angle value is positive, clockwise if the value is negative.

Transform (transformationMatrix as **Matrix**, [, changePositions as **Boolean**] [, changeFillPatterns as **Boolean**] [, changeFillGradients as **Boolean**] [, changeStrokePattern as **Boolean**] [, changeLineWidths as **Double**] [, transformAbout as AiTransformation])	Nothing	Transforms the art item by applying a transformation matrix.
Translate ([deltaX as **Double**] [, deltaY as **Double**] [, transformObjects as **Boolean**] [, transformFillPatterns as **Boolean**] [, transformFillGradients as **Boolean**] [, transformStrokePatterns as **Boolean**])	Nothing	Repositions the art item relative to the current position, where deltaX is the horizontal offset and deltaY is the vertical offset.
ZOrder (zOrderCmd as AiZOrderMethod)	Nothing	Arranges the art item's position in the stacking order of the group or layer (Parent) of this object.

➤ Using text frames

```
Dim appRef As New Illustrator.Application
Dim selection As Variant
Dim frontDocument As Illustrator.Document
Dim sourceTextArt As Illustrator.TextFrame
Dim newTextArt As Illustrator.TextFrame
Dim textArtGroup As Illustrator.TextFrames
' First check the selection of the application
' It has to be a text art item in order for this script to run
selection = appRef.selection
If (IsEmpty(selection)) Then
   MsgBox "Select a text item before running this script"
   Exit Sub
End If

If (TypeName(selection(0)) <> "TextFrame") Then
   MsgBox "Select a text item before running this script"
   Exit Sub
End If

Set frontDocument = appRef.ActiveDocument
Set sourceTextArt = selection(0)

'Get the parent of the text art so new text art items can be inserted
'in the same group or layer as the selected text art is in
Set textArtGroup = sourceTextArt.Parent.TextFrames

'Create 5 new versions of the text art each rotated a bit
Dim i As Integer
For i = 1 To 5
   Set newTextArt = textArtGroup.Add
   newTextArt.Position = sourceTextArt.Position
   newTextArt.Contents = sourceTextArt.Contents
   newTextArt.Rotate 180 * i / 6
Next
```

TextFrames

A collection of `TextFrame` objects.

TextFrames properties

Property	Value type	What it is
Application	`Application`	Read-only. The Illustrator `Application` object.
Count	`Long`	Read-only. Number of elements in the collection.
Parent	`object`	Read-only. The object's container.

TextFrames methods

Method	Returns	What it does
Add `()`	`TextFrame object`	Creates a new point `TextFrame` object.

AreaText
`(textPath as` **PathItem**
`[, orientation as AiTextOrientation]`
`[, baseFrame as` **TextFrame**`]`
`[, postFix as` **Boolean**`]) )`
	`TextFrame object`	Creates a new area `TextFrame` object.

Index `(item as` **TextFrame**`)`	`Long`	Returns the index position of the object within the collection.
Item `(itemKey)`	`TextFrame object`	Returns an object reference to the object identified by `itemKey` (name or index).

PathText
`(textPath as` **PathItem**
`[, startTValue as` **Double**`]`
`[, endTValue as` **Double**`]`
`[, orientation as AiTextOrientation]`
`[, baseFrame as` **TextFrame**`]`
`[, postFix as` **Boolean**`])`
	`TextFrame object`	Creates an on-path text frame item.

PointText `(anchor as` **Array of 2 Doubles** `[, orientation as AiTextOrientation])`	`TextFrame object`	Creates a point text frame item.
RemoveAll `()`	`Nothing`	Deletes all objects in this collection.

➤ Adding 3 types of text frames

The following example illustrates how to add a document and create three different types of text frames.

```
' TextFrames
' Add a document and create 3 TextFrames (area, path, point).
' Change the content of each TextFrame.
' Remove one TextFrame and display the count.
```

```
Dim appRef As New Illustrator.Application
Dim docRef As Illustrator.Document
Set docRef = appRef.Documents.Add()

' create 3 new textFrames (are, line, point)
' Area Text
Dim rectRef As Illustrator.pathItem
Set rectRef = docRef.PathItems.Rectangle(700, 50, 100, 100)
Dim areaTextRef As Illustrator.TextFrame
Set areaTextRef = docRef.TextFrames.AreaText(rectRef)
areaTextRef.Contents = "TextFrame #1"
areaTextRef.Selected = True

' Line Text
Dim lineRef As Illustrator.pathItem
Set lineRef = docRef.PathItems.Add()
lineRef.SetEntirePath (Array(Array(200, 700), Array(300, 550)))
Dim pathTextRef As Illustrator.TextFrame
Set pathTextRef = docRef.TextFrames.PathText(lineRef)
pathTextRef.Contents = "TextFrame #2"
pathTextRef.Selected = True

' Point Text
Dim pointTextRef As Illustrator.TextFrame
Set pointTextRef = docRef.TextFrames.Add()
pointTextRef.Contents = "TextFrame #3"
pointTextRef.Top = 700
pointTextRef.Left = 400
pointTextRef.Selected = True
appRef.Redraw

Dim sText As String
sText = "There are " & CStr(docRef.TextFrames.Count) & " TextFrames."
sText = sText & vbCrLf & "Changing contents of each TextFrame."
MsgBox sText

' change the content of each
areaTextRef.Contents = "Area TextFrame."
pathTextRef.Contents = "Path TextFrame."
pointTextRef.Contents = "Point TextFrame."
appRef.Redraw

MsgBox "Removing 1 TextFrame."
docRef.TextFrames(2).Delete
appRef.Redraw

' count again, count should reduce by 1.
MsgBox "Now there are " & CStr(docRef.TextFrames.Count) & " TextFrames."
```

TextPath

A path or list of paths for area or path text. A path consists of path points that define its geometry.

TextPath properties

Property	Value type	What it is
Application	Application object	Read-only. The Illustrator Application object.
Area	Double	Read-only. The area of this path in square points. If the area is negative, the path is wound counterclockwise. Self-intersecting paths can contain sub-areas that cancel each other out, which makes this value zero even though the path has apparent area.
BlendingMode	AiBlendModes	The blend mode used when compositing an object.
Clipping	Boolean	If true, this path should be used as a clipping path.
Editable	Boolean	Read-only. If true, this item is editable.
Evenodd	Boolean	If true, the even-odd rule should be used to determine insideness.
FillColor	Color Object	The fill color of the path.
Filled	Boolean	If true, the path be filled.
FillOverprint	Boolean	If true, the art beneath a filled object should be overprinted.
Guides	Boolean	If true, this path is a guide object.
Height	Double	The height of the group item.
Left	Double	The position of the left side of the item.
Note	String	The note text assigned to the path.
Opacity	Double	The opacity of the object. Range: 0.0 to 100.0
Parent	Layer or GroupItem	Read-only. The parent of this object.
PathPoints	PathPoints	Read-only. The path points contained in this path item.
Polarity	AiPolarityValues	The polarity of the path.
Position	Array of 2 Doubles	The position of the top left corner of the item.
Resolution	Double	The resolution of the path in dots per inch (dpi).
SelectedPathPoints	PathPoints	Read-only. All of the selected path points in the path.
StrokeCap	AiStrokeCap	The type of line capping.
StrokeColor	Color Object	The stroke color for the path.
Stroked	Boolean	If true, the path should be stroked.

Property	Value type	What it is
StrokeDashes	Object	Dash lengths. Set to an empty object, {}, for a solid line.
StrokeDashOffset	Double	The default distance into the dash pattern at which the pattern should be started.
StrokeJoin	AiStrokeJoin	Type of joints for the path.
StrokeMiterLimit	Double	When a default stroke join is set to mitered, this property specifies when the join will be converted to beveled (squared-off) by default. The default miter limit of 4 means that when the length ofþthe point reaches four times the stroke weight, the join switches from a miter join to a bevel join. A value of 1 specifies a bevel join. Range: 1 to 500 Default: 4
StrokeOverprint	Boolean	If true, the art beneath a stroked object should be overprinted.
StrokeWidth	Double	Width of the stroke.
Top	Double	The position of the top of the item.
Width	Double	The width of the item.

TextPath methods

Method	Returns	What it does
SetEntirePath (pathPoints as **PathPoints**)	Nothing	Sets the path using the specified anchor points.

TextRange

A range of characters from a text item (story, text frame, character, word, line, paragraph, or another text range).

TextRange properties

Property	Value type	What it is
Application	Application object	Read-only. The Illustrator Application object.
CharacterAttributes	CharacterAttributes	Read-only. The character properties for the text range.
CharacterOffset	Long	Offset of the first character.
Characters	Characters	Read-only. All of the characters in the text range.
CharacterStyles	CharacterStyles	Read-only. List of referenced character styles in the text range.
Contents	String	The text string.
InsertionPoints	InsertionPoints	Read-only. All of the insertion points in this text range.
Kerning	Long	The spacing between two characters, in thousandths of an em.
Length	Long	Length of text range. Minimum: 0
Lines	Lines	Read-only. All the lines in this text range.
ParagraphAttributes	ParagraphAttributes	Read-only. The paragraph properties.
Paragraphs	Paragraphs	Read-only. All of the paragraphs in this text range.
ParagraphStyles	ParagraphStyles	Read-only. A list of referenced paragraph styles in the text range.
Parent	Object	Read-only. The object's container.
Story	Story	Read-only. The story of the text range.
TextRanges	TextRanges	Read-only. All of the text in this text range.
TextSelection	Array of TextRange objects	Read-only. The selected text (ranges) in the text range.
Words	Words	Read-only. All of the words in this text range.

TextRange methods

Method	Returns	What it does
ChangeCaseTo (type as AiCaseChangeType)	Nothing	Changes the capitalization of text.
Delete ()	Nothing	Deletes the object.
DeSelect ()	Nothing	Deselects the text range.
Duplicate ([relativeObject as **Object**], [insertionLocation as AiElementPlacement])	TextRange	Duplicates the art item, optionally specifying the location and position for the copy.
Move (relativeObject as **Object**, insertionLocation as AiElementPlacement)	Nothing	Moves the art item to the new location and position.
Select ([addToDocument as **Boolean**])	Nothing	Selects the text range. If addToDocument is true, adds this to the current selection; otherwise replaces the current selection.

➤ Using a text range

```
Dim appRef As New Illustrator.Application
Dim textArt As Illustrator.TextFrame
Dim textArtRange As Illustrator.TextRange
Dim textWord As Illustrator.TextRange
Dim wordLen As Integer
Dim charsToChange As Integer
Dim firstChars As Illustrator.TextRange

For Each textArt In appRef.ActiveDocument.TextFrames
   Set textArtRange = textArt.TextRange

   For Each textWord In textArtRange.Words
      'If word is longer than 2 characters
      'resize just the first 2 characters
      'otherwise, resize the whole word
      wordLen = Len(textWord.Contents)
      If (wordLen < 2) Then
         charsToChange = wordLen
      Else
         charsToChange = 2
      End If
      'resize the word
      If (charsToChange > 0) Then
         'Omit 1st arg to go from the beginning to
         'character number charsToChange-1 (First character is index 0)
         Dim i
         For i = 1 To charsToChange
            Set firstChars = textWord.Characters(i)
```

```
            firstChars.CharacterAttributes.Size =_
               firstChars.CharacterAttributes.Size * 1.5
         Next
      End If
   Next
Next
```

TextRanges

A collection of `TextRange` objects.

TextRanges properties

Property	Value type	What it is
`Application`	`Application`	Read-only. The Illustrator `Application` object.
`Count`	`Long`	Read-only. Number of elements in the collection.
`Parent`	`Document`	Read-only. The object's document container.

TextRanges methods

Method	Returns	What it does
`Index` (`item as` **`TextRange`**)	`Long`	Returns the index position of the object within the collection.
`Item` (`itemKey`)	`TextRange object`	Returns an object reference to the object identified by `itemKey` (name or index).
`RemoveAll` ()	`Nothing`	Deletes all objects in this collection.

TracingObject

A tracing object, which associates source raster art item with a vector-art plugin group created by tracing. Scripts can initiate tracing using `PlacedItem.Trace` or `RasterItem.Trace`. The resulting `PluginItem` object represents the vector art group, and has this object in its `Tracing` property.

A script can force the tracing operation by calling the document's `Redraw` method. The operation is asynchronous, so a script should call `Redraw` after creating the tracing object, but before accessing its properties or expanding the tracing to convert it to an art item group.

The read-only properties that describe the tracing result have valid values only after the first tracing operation completes. A value of 0 indicates that the operation has not yet been completed.

TracingObject properties

Property	Value type	What it is
AnchorCount	Long	Read-only. The number of anchors in the tracing result.
Application	Application	Read-only. The Illustrator `Application` object.
AreaCount	Long	Read-only. The number of areas in the tracing result.
ImageResolution	Real	Read-only. The resolution of the source image in pixels per inch.
Parent	Object	Read-only. The object's container.
PathCount	Long	Read-only. The number of paths in the tracing result.
SourceArt	PlacedItem or RasterItem	The raster art used to create the associated vector-art plugin group.
TracingOptions	TracingOptions	The options used to convert the raster artwork to vector art.
UsedColorCount	Long	Read-only. The number of colors used in the tracing result.

TracingObject methods

Method	Returns	What it does
ExpandTracing ([viewed As Boolean])	GroupItem object	Converts the vector art into a new group item. The new GroupItem object replaces the PluginItem object in the document. By default, viewed is false, and the new group contains only the tracing result (the filled or stroked paths). If viewed is true, the new group retains additional information that was specified for the viewing mode, such as outlines and overlays. Deletes this object and its associated PluginItem object. Any group-level attributes that were applied to the plugin item are applied to the top level of the new group item.
ReleaseTracing ()	PlacedItem or RasterItem object	Reverts the artwork in the document to the original source raster art and removes the traced vector art. Returns the original object used to create the tracing, and deletes this object and its associated PluginItem object.

TracingOptions

A set of options used in converting raster art to vector art by tracing.

TracingOptions properties

Property	Value type	What it is
Application	Application	Read-only. The Illustrator Application object.
CornerAngle	Double	The sharpness, in degrees of a turn in the original image that is considered a corner in the tracing result path. Range: 0 to 180
Fills	Boolean	If true, trace with fills. At least one of Fills or Strokes must be true.
LivePaintOutput	boolean	If true, result is LivePaint art. If false, it is classic art. **Note:** A script should only set this value in preparation for a subsequent expand operation. Leaving a tracing on the artboard when this property is true can lead to unexpected application behavior.
MaxColors	Long	The maximum number of colors allowed for automatic palette generation. Used only if TracingMode is color or grayscale. Range: 2 to 256
MaxStrokeWeight	Double	The maximum stroke weight in points, when Strokes is true. Range: 0.01 to 100.0
MinArea	Long	The smallest feature, in square pixels, that is traced. For example, if it is 4, a feature of 2 pixels wide by 2 pixels high is traced.
MinStrokeLength	Double	The minimum length in pixels of features in the original image that can be stroked, when Strokes is true. Smaller features are omitted. Range: 0.0 to 200.0 Default: 20.0
OutputToSwatches	Boolean	If true, named colors (swatches) are generated for each new color created by the tracing result. Used only if TracingMode is color or grayscale.
Palette	String	The name of a color palette to use for tracing. If the empty string, use the automatic palette. Used only if TracingMode is color or grayscale.
Parent	Object	Read-only. The object's container.
PathFitting	Double	The distance between the traced shape and the original pixel shape. Lower values create a tighter path fitting. Higher values create a looser path fitting. Range: 0.0 to 10.0

Property	Value type	What it is
PreprocessBlur	Double	The amount of blur used during preprocessing, in pixels. Blurring helps reduce small artifacts and smooth jagged edges in the tracing result. Range: 0.0 to 2.0
Preset	String	Read-only. The name of a preset file containing these options.
Resample	Boolean	If true, resample when tracing. (This setting is not captured in a preset file.) Always true when the raster source art is placed or linked.
ResampleResolution	Double	The resolution to use when resampling in pixels per inch (ppi). Lower resolution increases the speed of the tracing operation. (This setting is not captured in a preset file.)
Strokes	Boolean	If true, trace with strokes. At least one of Fills or Strokes must be true. Used only if TracingMode is black-and-white.
Threshold	Long	The threshold value of black-and-white tracing. All pixels with a grayscale value greater than this are converted to black. Used only if TracingMode is black-and-white. Range: 0 to 255
TracingMode	AiTracingModeType	The color mode for tracing.
ViewRaster	AiViewRasterType	The view for previews of the raster image. (This setting is not captured in a preset file.)
ViewVector	AiViewVectorType	The view for previews of the vector result. (This setting is not captured in a preset file.)

TracingOptions methods

Method	Returns	What it does
LoadFromPreset (presetName as **String**)	Boolean	Loads a set of options from the specified preset, as found in the Application.TracingPresetList array.
StoreToPreset (presetName as **String**)	Boolean	Saves this set of options in the specified preset. Use a name found in the Application.TracingPresetList array, or a new name to create a new preset. For an existing preset, overwrites an unlocked preset and returns true. Returns false if the preset is locked.

Variable

A dynamic object used to create data-driven graphics. For an example, see DataSets.

Variable properties

Property	Value type	What it is
Application	Application	Read-only. The Illustrator Application object.
Kind	AiVariableKind	The variable's type.
Name	String	The name of the variable.
PageItems	PageItems	Read-only. All of the artwork in this document.
Parent	Document	Read-only. The document that contains this object.

Variable methods

Method	Returns	What it does
Delete ()	Nothing	Deletes the object.

Variables

A collection of `Variable` objects in a document.

Variables properties

Property	Value type	What it is
Application	`Application`	Read-only. The Illustrator `Application` object.
Count	`Long`	Read-only. Number of elements in the collection.
Parent	`Document`	Read-only. The object's document container.

Variables methods

Method	Returns	What it does
Add ( )	`Variable object`	Creates a new `Variable` object.
Index (item as **Variable**)	`Long`	Returns the index position of the object within the collection.
Item (itemKey)	`Variable object`	Returns an object reference to the object identified by `itemKey` (name or index).
RemoveAll ( )	`Nothing`	Deletes all objects in this collection.

View

A document view in an Illustrator document, which represents a window view onto a document. Scripts cannot create new views, but can modify some properties of existing views, including the center point, screen mode, and zoom.

View properties

Property	Value type	What it is
Application	Application object	Read-only. The Illustrator Application object.
Bounds	Variant Array of 4 Doubles	Read-only. The bounding rectangle of this View relative to the current document's bounds.
CenterPoint	Variant Array of 2 Doubles	The center point of this View relative to the current document's bounds.
Parent	Document object	Read-only. The document that contains this object.
ScreenMode	AiScreenMode	The mode of display for this view.
Zoom	Double	The zoom factor of this view, where 100.0 is 100%

➤ Setting a view

This example illustrates how to set the first view of the frontmost document to full screen mode.

```
Dim appRef As New Illustrator.Application
Dim frontDoc As Illustrator.Document
Set frontDoc = appRef.activeDocument
Dim viewsRef As Illustrator.Views
Set viewsRef = frontDoc.Views
Dim firstView As Illustrator.View
Set firstView = viewsRef(1)
firstView.ScreenMode = aiFullScreen
```

Views

A collection of `View` objects in a document.

Views properties

Property	Value type	What it is
`Application`	`Application`	Read-only. The Illustrator `Application` object.
`Count`	`Long`	Read-only. Number of elements in the collection.
`Parent`	`Document`	Read-only. The object's document container.

Views methods

Method	Returns	What it does
`Index` (`item as` **`View`**)	`Long`	Returns the index position of the object within the collection.
`Item` (`itemKey`)	`View object`	Returns an object reference to the object identified by `itemKey` (name or index).

Words

A collection of words in a text item, where each word is a `TextRange` object.

Words properties

Property	Value type	What it is
Application	Application	Read-only. The Illustrator `Application` object.
Count	Long	Read-only. Number of elements in the collection.
Parent	Object	Read-only. The object's container.

Words methods

Method	Returns	What it does
Add (contents as **String** [, relativeObject as **TextFrame**] [, insertionLocation as AiElementPlacement])	TextRange	Adds a new word with specified text contents at the specified location in the current document. If location is not specified, adds the new word to the containing text frame after the current text selection or insertion point.
AddBefore (contents as **String**)	TextRange	Inserts text at the beginning of the range.
Index (itemPtr as **TextRange**)	Long	Returns the index position of the object within the collection.
Item (itemKey)	TextRange	Returns an object reference to the object identified by `itemKey` (name or index).
RemoveAll ()	Nothing	Deletes all objects in this collection.

➤ Counting words in a document

Displays the total number of words contained in all `TextFrameItems` in the current document.

```
Dim appRef As New Illustrator.Application
Dim numWords As Integer
Dim textArt As Illustrator.TextFrame
Dim textArtRange As Illustrator.TextRange

If appRef.Documents.Count > 0 Then
   numWords = 0
   For Each textArt In appRef.ActiveDocument.TextFrames
      numWords = numWords + textArt.TextRange.Words.Count
   Next
   MsgBox ("There are " & numWords & " words in the document.")
End If
```

Enumerations reference

These enumeration values are defined for object properties in the Visual Basic dictionary.

Enumeration type	Values	What it means
AiAlternateGlyphsForm	aiDefaultForm aiTraditional aiExpert aiJIS78Form aiJIS83Form aiHalfWidth aiThirdWidth aiQuarterWidth aiFullWidth aiProportionalWidth	
AiAutoKernType	aiNoAutoKern aiAuto aiOptical	
AiAutoLeadingType	aiBottomToBottom aiTopToTop	
AiBaselineDirectionType	aiStandardBaseline aiTateChuYokoBaseline aiVerticalRotatedBaseline	
AiBlendAnimationType	aiInBuild aiInSequence aiNoBlendAnimation	
AiBlendModes	aiColorBlend aiColorBurn aiColorDodge aiDarken aiDifference aiExclusion aiHardLight aiHue aiLighten aiLuminosity aiMultiply aiNormalBlend aiOverlay aiSaturation aiScreen aiSoftLight	The blend mode used when compositing an object
AiBurasagariTypeEnum	aiBurasagariNone aiBurasagariStandard aiBurasagariForced	
AiCaseChangeType	aiUpperCase aiLowerCase aiTitleCase aiSentenceCase	
AiColor	aiColorCMYK =1 aiColorGradient =6 aiColorGray = 2 aiColorNone = 0 aiColorPattern = 5 aiColorRGB =3 aiColorSpot =4	A color space.

Enumeration type	Values	What it means
AiColorConversion	aiColorConversionRepurpose aiColorConversionToDest aiColorConversionNone	
AiColorDestination	aiColorDestinationDocCMYK aiColorDestinationDocRGB aiColorDestinationProfile aiColorDestinationWorkingCMYK aiColorDestinationWorkingRGB aiColorDestinationNone	
AiColorDitherMethod	aiDiffusion aiNoise aiNoReduction aiPatternDither	The method used to dither colors in exported GIF and PNG8 images
AiColorModel	aiRegistration aiProcess aiSpot	
AiColorProfile	aiIncludeAllProfile aiIncludedeStProfile aiIncludeRGBProfile aiLeaveProfileUnchanged aiColorProfileNone	
AiColorReductionMethod	aiAdaptive aiPerceptual aiSelective aiWeb	The method used to reduce the number of colors in exported GIF and PNG8 images
AiCompatibility	aiIllustrator3 aiJapaneseVersion3 aiIllustrator8 aiIllustrator9 aiIllustrator10 aiIllustrator11 aiIllustrator12	The version of the Illustrator file format to create when saving an EPS or Illustrator file

Enumeration type	Values	What it means
AiCompressionQuality	aiJPEGHigh aiJPEGLow aiJPEGMaximum aiJPEGMedium aiJPEGMinimum aiNoCompression aiZIP4Bit aiZIP8Bit aiAutomaticJPEGMinimum aiAutomaticJPEGLow aiAutomaticJPEGMedium aiAutomaticJPEGHigh aiAutomaticJPEGMaximum aiAutomaticJPEG2000Minimum aiAutomaticJPEG2000Low aiAutomaticJPEG2000Medium aiAutomaticJPEG2000High aiAutomaticJPEG2000Maximum aiAutomaticJPEG2000Lossless aiJPEG2000Minimum aiJPEG2000Low aiJPEG2000Medium aiJPEG2000High aiJPEG2000Maximum aiJPEG2000Lossless	The quality of bitmap compression used when saving a PDF file
AiCropOptions	aiCropJapanese aiCropStandard	The style of a document's cropping box
AiDocumentColorSpace	aiDocumentCMYKColor aiDocumentRGBColor	The color space of a document
AiDocumentType	aiEPS aiIllustrator aiPDF	The file format used to save a file
AiDownsampleMethod	aiNoDownsample aiAverageDownsample aiSubsample aiBicubicDownsample	
AiElementPlacement	aiPlaceInside aiPlaceAtBeginning aiPlaceAtEnd aiPlaceBefore aiPlaceAfter	
AiEPSPostScriptLevelEnum	aiLevel2 aiLevel3	Specifies the PostScript level to use when saving an EPS file
AiEPSPreview	aiBWMacintosh aiBWTIFF aiColorMacintosh aiColorTIFF aiNoPreview aiTransparentColorTIFF	The preview image format used when saving an EPS file

Enumeration type	Values	What it means
AiExportType	aiGIF aiJPEG aiPhotoshop aiPNG24 aiPNG8 aiSVG aiFlash	The file format used to export a file
AiFigureStyleType	aiDefaultFigureStyle aiTabular aiTabularOldStyle aiProportional aiProportionalOldStyle	
AiFlashExportStyle	aiAsFlashFile aiLayersAsFrames aiLayersAsFiles	
AiFlashImageFormat	aiLossless aiLossy	
AiFlashJPEGMethod	aiJPEGStandard aiJPEGOptimized	
AiFontBaselineOption	aiNormalBaseline aiSuperScript aiSubScript	
AiFontCapsOption	aiNormalCaps aiSmallCaps aiAllCaps aiAllSmallCaps	
AiFontOpenTypePositionOption	aiOpenTypeDefault aiOpenTypeSuperScript aiOpenTypeSubScript aiNumerator aiDenominator	
AiFontSubstitutionPolicy	aiSubstituteOblique aiSubstituteTint aiSubstituteDevice	
AiGradientType	aiLinearGradient aiRadialGradient	The type of the gradient, radial or linear
AiImageColorSpace	aiImageCMYK aiImageGrayScale aiImageRGB	The color space of a raster item or an exported Photoshop 5 file
AiInkPrintStatus	aiDisableInk aiEnableInk aiConvertInk	
AiInkType	aiCyanInk aiMagentaInk aiYellowInk aiBlackInk aiCustomInk	
AiJavaScriptExecutionMode	aiNeverShowDebugger aiDebuggerOnError aiBeforeRunning	

Enumeration type	Values	What it means
AiJustification	aiCenter aiLeft aiRight aiFullJustifyLastLineLeft aiFullJustifyLastLineRight aiFullJustifyLastLineCenter aiFullJustify	The alignment or justification for a paragraph of text
AiKinsokuOrderEnum	aiPushIn aiPushOutFirst aiPushOutOnly	
AiKnockoutState	aiDisabled aiEnabled aiInherited aiKnockoutUnknown	The type of knockout to use on a PageItem
AiLanguageType	aiBokmalNorwegian aiBrazillian Portuguese aiBulgarian aiCanadianFrench aiCatalan aiChinese aiCzech aiDanish aiDutch aiEnglish aiFinnish aiGreek aiHungarian aiIcelandic aiItalian aiJapanese aiNynorskNorwegian aiOldGerman aiPolish aiRomanian aiRussian aiSpanish aiSerbian aiStandardFrench aiStandardGerman aiStandardPortuguese aiSwedish aiSwissGerman aiTurkish aiUKEnglish aiUkranian	
AILayerOrderType	aiTopDown aiBottomUp	
AiMonochromeCompression	aiCCIT3 aiCCIT4 aiMonoZIP aiNoMonoCompression aiRunLength	The type of monochrome bitmap compression to use when saving a PDF

Enumeration type	Values	What it means
AiOutputFlattening	aiPreserveAppearance aiPreservePaths	Specifies how transparency should be flattened when saving EPS and Illustrator file formats with compatibility set to versions of Illustrator less than 9
AiPageItemType	aiCompoundPathItem = 1 aiGraphItem = 2 aiGroupItem =3 aiLegacyTextItem = 11 aiMeshItem = 4 aiPathItem = 5 aiPlacedItem = 6 aiPluginItem =7 aiRasterItem =8 aiSymbolItem = 9 aiTextFrame = 10	The type (class) of art item that is represented by a particular PageItem
AiPageMarksStyle	aiPageMarksRoman aiPageMarksJapanese	
AiPathPointSelection	aiAnchorPoint aiLeftDirection aiLeftRightPoint aiNoSelection aiRightDirection	Specifies which points, if any, of a path point are selected
AiPDFBoxType	aiPDFArtBox aiPDFCropBox aiPDFTrimBox aiPDFBleedBox aiPDFMediaBox aiPDFBoundingBox	
AiPDFChangesAllowedEnum	aiChange128None aiChange128EditPage aiChange128FillForm aiChange128Commenting aiChange128AnyChanges aiChange40None aiChange40PageLayout aiChange40Commenting aiChange40AnyChanges	
AiPDFCompatibility	aiAcrobat4 aiAcrobat5 aiAcrobat6 aiAcrobat7	The version of the Acrobat file format to create when saving a PDF file
AiPDFOverprint	aiPreservePDFOverprint aiDiscardPDFOverprint	
AiPDFPrintAllowedEnum	aiPrint128None aiPrint128LowResolution aiPrint128HighResolution aiPrint40None aiPrint40HighResolution	

Enumeration type	Values	What it means
AiPDFTrimMarkWeight	aiTrimMarkWeight0125 aiTrimMarkWeight025 aiTrimMarkWeight05	
AiPDFXStandard	aiPDFX1A2001 aiPDFX1A2003 aiPDFX32001 aiPDFX32003 aiPDFXNone	
AiPhotoshopCompatibility	aiPhotoshop5 aiPhotoshop8	
AiPointType	aiCorner aiSmooth	The type of path point, either a curve or a corner
AiPolarityValues	aiPositive = 1 aiNegative = –1	
AiPostScriptImageCompressionType	aiImageCompressionNone aiImageCompressionRLE aiImageCompressionJPEG	
AiPrintArtworkDesignation	aiVisiblePrintableLayers aiVisibleLayers aiAllLayers	
AiPrintColorIntent	aiPerceptualIntent aiSaturationIntent aiRelativeColorimetric aiAbsoluteColorimetric	
AiPrintColorProfile	aiOldstyleProfile aiSourceProfile aiPrinterProfile aiCustomProfile	
AiPrintColorSeparationMode	aiComposite aiOnHostSeparation aiInRIPSeparation	
AiPrinterColorMode	aiColorPrinter aiGrayscalePrinter aiBlackAndWhitePrinter	
AiPrinterPostScriptLevelEnum	aiPSLevel1 aiPSLevel2 aiPSLevel3	
AiPrinterTypeEnum	aiUnknownPrinterType aiPostScriptPrinter aiNonPostScriptPrinter	
AiPrintFontDownloadMode	aiDownloadNone aiDownloadSubset aiDownloadComplete	
AiPrintingBounds	aiArtboardBounds aiArtworkBounds aiCropBounds	

Enumeration type	Values	What it means
AiPrintOrientation	aiPortrait aiLandscape aiReversePortrait aiReverseLandscape	
AiPrintPosition	aiTranslateTopLeft aiTranslateTop aiTranslateTopRight aiTranslateLeft aiTranslateCenter aiTranslateRight aiTranslateBottomLeft aiTranslateBottom aiTranslateBottomRight	
AiPrintTiling	aiTileSingleFullPage aiTileFullPages aiTileImageableAreas	
AiRasterLinkState	aiDataFromFile aiDataModified aiNoData	The status of a raster item's linked image, if the image is stored externally
AiRulerUnits	aiUnitsCM aiUnitsInches aiUnitsMM aiUnitsPicas aiUnitsPoints aiUnitsQ aiUnitsUnknown	The default measurement units for the rulers in a document
AiSaveOptions	aiDoNotSaveChanges aiPromptToSaveChanges aiSaveChanges	Save options provided when closing a document
AiScreenMode	aiDesktop aiFullScreen aiMultiWindow	The mode of display for a view
AiStrokeCap	aiButtEndCap aiProjectingEndCap aiRoundEndCap	The type of line capping for a path stroke
AiStrokeJoin	aiBevelEndJoin aiMiterEndJoin aiRoundEndJoin	The type of joints for a path stroke
AiStyleRunAlignmentType	aiAlignBottom aiICFBottom aiRomanBaseline aiAlignCenter aiICFTop aiAlignTop	
AiSVGCSSPropertyLocation	aiEntities aiStyleAttributes aiStyleElements aiPresentationAttributes	How should the CSS properties of the document be included in an exported SVG file?
AiSVGDocumentEncoding	aiASCII aiUTF16 aiUTF8	How text is encoded when exporting an SVG file

Enumeration type	Values	What it means
AiSVGDTDVersion	aiSVG1_0 aiSVG1_0 aiSVG1_1 aiSVGBasic1_1 aiSVGTiny1_1 aiSVGTiny1_1Plus	SVG version compatibility for exported file
AiSVGFontSubsetting	aiAllGlyphs aiCommonEnglish aiCommonRoman aiGlyphsUsed aiGlyphsUsedPlusEnglish aiGlyphsUsedPlusRoman aiNoFonts	What font glyphs should be included in an exported SVG file?
AiSVGFontType	aiCEFFont aiOutlineFont aiSVGFont	Types for fonts included in exported SVG files
AiTabStopAlignment	aiCenterTab aiDecimalTab aiLeftTab aiRightTab	The alignment of a tab stop
AiTextOrientation	aiHorizontal aiVertical	The orientation of text in a `TextFrameItem`
AiTextType	aiAreaText aiPathText aiPointText	The type of text art displayed by this object
AiTracingModeType	aiTracingModeBlackAndWhite aiTracingModeColor aiTracingModeGray	
AiTransformation	aiTransformBottom aiTransformBottomLeft aiTransformBottomRight aiTransformCenter aiTransformDocumentOrigin aiTransformLeft aiTransformRight aiTransformTop aiTransformTopLeft aiTransformTopRight	The point to use as the anchor point about which an object is rotated, resized or transformed
AiTrappingType	aiNormalTrapping aiTransparent aiOpaque aiIgnoreOpaque	
AiUserInteractionLevel	aiDontDisplayAlerts = -1 aiDisplayAlerts = 2	
AiVariableKind	aiGraph aiImage aiTextual aiUnknownKind aiVisibility	The enumerated type of the kind of variable

Enumeration type	Values	What it means
AiViewRasterType	aiTracingViewRasterAdjustedImage aiTracingViewRasterNoImage aiTracingViewRasterOriginalImage aiTracingViewRasterTransparentImage	The raster viewing mode for tracing.
AiViewVectorType	aiTracingViewVectorNoTracingResult aiTracingViewVectorOutlines aiTracingViewVectorOutlinesWithTracing aiTracingViewVectorTracingresult	The vector viewing mode for tracing.
AiWariChuJustificationType	aiWariChuLeft aiWariChuRight aiWariChuCenter aiWariChuFullJustifyLastLineLeft aiWariChuFullJustifyLastLineRight aiWariChuFullJustifyLastLineCenter aiWariChuAutoJustify	
AiZOrderMethod	aiBringForward aiBringToFront aiSendBackward aiSendToBack	The method used to arrange an art item's position in the stacking order of its parent group or layer, as specified with the ZOrder method

Index

E

F

G

H

I

Q

R

S

T

U

V

W

X

Y

Z