

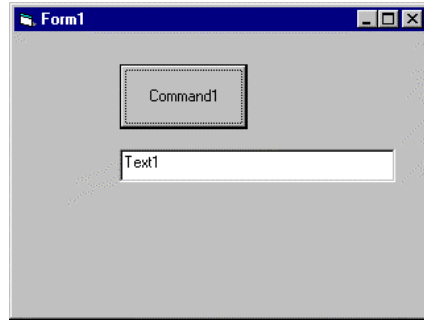
# Unraveling the Mystery of Visual Basic, Objects, and ActiveX

The terms Visual Basic, Objects, and ActiveX are becoming more and more prevalent in the press and in vendor's literature and presentations in the automation industry. To many users, it is just another onslaught of terms and new technologies to learn, with conflicting and confusing stories about what these things are. We seek to unravel the mysteries here and put some clarity behind the questions "What is Visual Basic, what are objects and what are ActiveX controls?"

## What is Visual Basic?

Visual Basic (VB) is a product developed by Microsoft for the writing of software applications. You can go down and purchase a copy of Visual Basic at your local electronics store or call your favorite mail-order house and order it. Visual Basic is also a programming language used to create software applications ranging from small to large and complex. The product has grown out of the original Basic computer programming language that has been around for 20 years. However, VB shares little in common with the Basic of 20 years ago. Today's VB can accomplish in one command what took 500 lines of Basic just 10 years ago. Today's VB is fast and may have been used to create many of the software programs you have loaded on your PC. Today's VB looks more like a scripting language than a cryptic programming language.

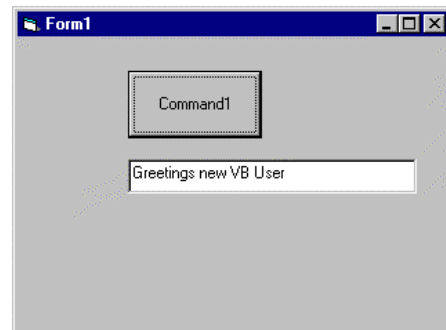
Are you skeptical? Take a look at this simple application. When the user clicks on the button, the message "Greetings New VB User" is displayed in the box on the screen. This VB application took just six mouse clicks to create and we wrote one line of Visual Basic. If we're really honest with ourselves, we should quickly realize this is



**The running program - before we click on the Command1 button**

The sum total of the Visual Basic program - The VB development environment wrote the first and last lines for us! -- We just typed the 2<sup>nd</sup> line and we were done! Click on the Command1 Button and see the results below!

```
Private Sub Command1_Click()  
Text1 = "Greetings new VB User"  
End Sub
```



**The program after we clicked on the Command1 button**





no more work than we do to type a formula in an Excel spreadsheet!

If you start playing with VB, you'll find that the Visual Basic program that you use to create your programs does a lot of the work for you. It even checks spelling and gives you hints along the way as you are typing program statements in! Notice that of the three lines of program, the VB development

program wrote 2 of them - we only wrote one line ourselves! That's called making your computer work for you!

### What is an object?

Forget the "object speak" here. Let's get down to simple terms. Take a look at the screen shown in our sample application. It has 2 objects on the screen. The command button and the text box where we displayed our "Greetings New VB User" message. Where did these objects come from? How did they get on the page? Once again, the program does most of the work for us. Take a look at the Visual Basic toolbar shown here. See all the little pictures or icons on the toolbar? Each of those represents an object that you can click on with your mouse and place on your screen. This is like playing a game -- pickup the object, put it on your screen! We placed the Command Button object by clicking on the command button object on the toolbar and then clicking on our screen. We placed the text box by doing the same thing with the text box object button.

Item	Button on Toolbar	Result on Form
Command Button Object		
Text Box Object		

That is all there is to objects! Now you can get pretty fancy with your objects if you want. There are objects that display pictures, can read/write data from your database, add a web browser window to your program, and more. Basically, you get objects with Visual Basic that handle all the user interface functionality of Windows. You don't have to worry about what makes

the object work. You just know what it can do and put it to work.

### What is ActiveX?

So what is ActiveX? ActiveX is just a name for a type of object. It is not a programming language. The actual correct term is ActiveX control. You can go out and buy ActiveX controls to add capability to Visual Basic just like you buy parts and accessories for your car or PC.



An ActiveX control is a small piece of software that someone has written and made available to perform a specific special function. Now when you buy the ActiveX control, you are actually buying a license to use the object in your applications. The person who developed it still owns the source code. You can buy ActiveX object accessories for prices ranging from \$50 to thousands of dollars, depending upon the power and capabilities found in the object. Like the promise of Java, ActiveX controls facilitate the reuse of program code between applications. ActiveX controls generally will work only on Windows 9x and NT operating systems, where Java will allow the careful programmer to reuse code across operating systems.

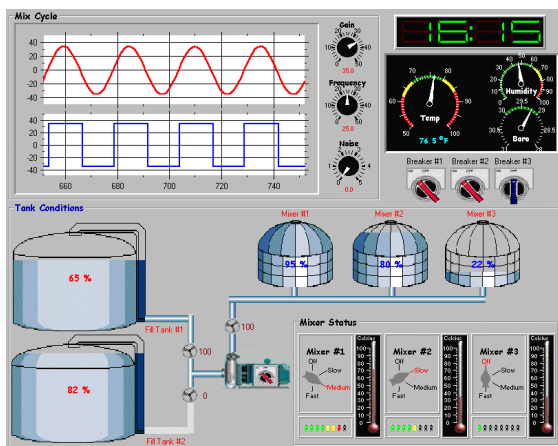
## Why more objects?

A good example of why an automation user would by an ActiveX object would be the need to talk to your PLC using a Visual Basic application. You won't find any objects on the out-of-the box VB5 toolbar that will let you connect to a PLC.

However, you can go out and buy an ActiveX control to do the job and load it on your PC. It will come delivered on a disk with some documentation explaining what it looks like and how it is used.

When you load the ActiveX control, it appears on your VB toolbar just like our command button and text box did. You can click on it and add it to your screen. Then you can write some simple lines of Visual Basic language to make the control read some PLC data and put it in our text box. In the box shown here, we have modified our sample program to talk a PLC. We installed an ActiveX control, added it to the form, and wrote four lines of Visual Basic. The VB editor wrote the other 4 lines for us again!

There are dozens of other objects you can purchase to add to your program. You can obtain objects for trend charts, gauges, knobs, and much more. The screen shown here contains several different types of objects:

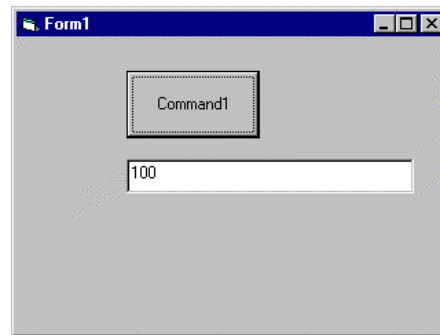


AB  
OCK

## The PLC ActiveX Control

```
Private Sub ABCTL1_OnReadDone()  
    Text1 = ABCTL1.WordVal(0)  
End Sub
```

```
Private Command1_Click()  
    ABCTL1.FileAddr = "N7:0"  
    ABCTL1.Node = 1  
    ABCTL1.Trigger  
End Sub
```



## The sample program, now reading data from a PLC

- Trend Chart
- Rotary Gauges
- LED display
- Knobs
- Selector Switches
- Sliders (for fills)
- Segmented bar graphs

Each of these objects was added to this VB screen just like we did for the command button and text box objects.

## Conclusions

To review, Visual Basic is a programming language and a software development environment. You use the Visual Basic development environment to write programs using the Visual Basic language. Those

programs consist of functional pieces such as command buttons, text boxes, PLC drivers, and graphics displays that are known as objects. You connect all the objects together using the Visual Basic programming language in the Visual Basic development environment. When you need to add functionality to Visual Basic that isn't in the box, you purchase ActiveX controls as accessories to add the functionality needed. A wide variety of ActiveX controls are available on the open market for adding PLC communications functionality, graphics capabilities, enhanced database connectivity and data visualization, reporting, and more.



*About the author: John Weber, founder of The Software Toolbox (Charlotte, NC) as well as the vice president of technology has programmed PCs for 16 years and has spent the last 10 years in the automation industry. Like many of the readers of this paper, he learned to program using procedural programming where you had to plan, manage and tell the program to do everything. The object-oriented and event-driven nature of Visual Basic required a mind-shift in programming style for him to be able to grasp the power and ease-of-use of VB. He hopes that this article will help its readers make that same mind shift and that they too can enjoy the benefits VB brings to the developer. John can be reached by email at [jweber@softwaretoolbox.com](mailto:jweber@softwaretoolbox.com)*

This technical paper provided to you compliments of The Software Toolbox<sup>®</sup>. To learn more about The Software Toolbox's products or about Visual Basic technologies and their use for HMI, visit the following Internet web site links or call us at the numbers below.

Software Toolbox Web Site: [www.softwaretoolbox.com](http://www.softwaretoolbox.com)

Email: [tools@softwaretoolbox.com](mailto:tools@softwaretoolbox.com)

Microsoft VB Web Site: <http://msdn.microsoft.com/vba>

Contacting Software Toolbox:

Phone: Office hours are 8 AM to 5 PM EST (GMT-5) Monday through Friday

1-888-665-3678 - Toll Free US

1-704-849-2773- International

Fax: 704-849-6388