

VBA Foundations, part 1

A Tutorial in VBA for Beginners—The First of a Twelve Part Series

Richard L. Binning / rbinning@mediaone.net

“My vision for the future is that PC systems will evolve to the point where the user is not even aware that different applications are being invoked to produce a document. One essential element in this vision is a common macro language. A common macro language will have several advantages for users. First, it will be easy to use. Second, it will be the same in a variety of applications. Finally, by using agents—a graphical interface “operative” that can cross applications boundaries—users can work in one application and call parts of other applications into play as needed, or they can start from outside any application and tie them together in various ways.

Thus, just as applications are becoming increasingly programmable, programming languages are becoming generally more accessible. Soon we will have tools that make the creation of Windows applications easy and enjoyable.

Imagine designing the visual components of an application graphically—merely by placing controls on a form. All programs would be designed, created and run within the Windows environment. Taking the ease of use that is common to both BASIC and Windows, the result would be a visual, productive, and interoperable tool for creating applications in the environment that is currently the most popular for personal computers.”
—Bill Gates, Chief Software Architect, Microsoft, Inc., Guest Editorial, BasicPro Magazine, 1991

I hope, over this series of articles on VBA Foundations for AutoCAD, to demonstrate how descriptive this quote is with regards to Visual Basic for Applications. Please take a moment to re-read the above quote and see if it answers some of the questions you have been asking as you embark on this new field of study.

Now that you’ve read it for the second time did you mutter to yourself “Yes! That is precisely why I wanted to learn VBA, so get on with it.” Good! Before we dive in to the subject at hand, consider that this quote was written in 1991. Intrigued? Great! Lets get started.

In this issue we are going to explore the following topics:

- The History of VBA in AutoCAD
- What is VBA?
- What is ActiveX?
- Why should we use VBA?
- Can I mix VBA in with AutoLISP®, Visual LISP™, etc.?

The History of VBA in AutoCAD

Although VBA is a relative newcomer in the AutoCAD programming ring, don’t be mistaken and think that VBA is some kind of lightweight or rookie. VBA, (Visual Basic® for Applications), comes from a long line of BASIC language development environments and in its latest incarnation, VBA6, is by far the most powerful yet. VBA first appeared in Microsoft Excel and Microsoft Project in 1994. It was introduced to AutoCAD in Release 14.0 as a “preview edition”. In Release 14.01, VBA became a permanent addition to the core AutoCAD application. The integration of this programming language used by more programmers than any other language in the history of computing provides AutoCAD with another well-developed and robust interface for customization and automation. Visual Basic® for Applications utilizes the same Visual Basic language syntax as its stand-alone brother VB. With its new forms package, full support for ActiveX Controls, and complete integration into AutoCAD, VBA is both complete and powerful. Internal to AutoCAD it acts as an in-process controller, providing outstanding performance. True to Bill Gates’ vision of the future, it also allows integration with and automated control of other applications that have a similar VBA interface.

(Continued on page 2)

VBA Foundations, part 1

A Tutorial in VBA for Beginners—The First of a Twelve Part Series

Richard L. Binning / rbinning@mediaone.net

What is ActiveX?

ActiveX Automation is the umbrella term for the component framework developed by Microsoft to provide an “object” based interface to applications. Basically, no pun intended, ActiveX is the plumbing that allows VBA to talk to AutoCAD directly and in a relatively easy to understand method. This frees you the “programmer” from learning all the intricacies of the programming game. Its as simple as this, when you walk into a room, you may not understand how electricity really works, but you know if the room is wired odds are that when you flip the switch on the wall, the overhead lights will come on. Autodesk has done us a great favor by providing the software plumbing (ActiveX) allowing us to flip the switch to automation and do some really powerful customization in an easy to understand environment. Using Automation, you can create and manipulate AutoCAD objects inside of AutoCAD or from any other VBA enabled application. AutoCAD’s ActiveX interface operates as both a client and controller. Thus it can be controlled and it can do the controlling. VBA and ActiveX together enable programming control across applications, a capability that does not exist in AutoLISP. With this level of automation the features of many applications can be combined or put to use in a single application.

Why Use VBA?

Beginners can create useful applications by learning just a few of the keywords, yet the power of the language allows seasoned professionals to accomplish virtually anything that can be accomplished using any of the high level programming languages. Writing “Visual” applications used to mean writing huge amounts of hard to decipher code. This was necessary just to describe to the application what the application was supposed to look like when it was running. This was in addition to the huge amounts of code written to actually perform the functions or logic that was the engine of the application itself. Because of these monumental tasks, many would-be developers shied away from or avoided like the plague doing what we are about to begin. Because VBA’s powerful visual interface takes care of most of the background programming tasks, the act of creating highly functional and useful programs is virtually nothing more than dragging pre-built objects into place on screen and tying them together with a little bit of easy to understand logic. In addition, being a user of AutoCAD means that you already have most of the skills necessary to create an effective user interface. The benefits of VBA include the following:

- Run time Speed – faster than AutoLISP applications
- Ease of Use – built in and easy to use.
- Debugging – finding and correcting errors has never been easier.
- Interoperability – built into Windows, VBA allows communication and control of other applications.
- Rapid Prototyping – build your first form and you’ll wonder why you ever spent time learning to write DCL.

Can I mix VBA in with AutoLISP, Visual LISP, etc.?

VBA is at its best when it is used in conjunction with the other programming interfaces already available inside of AutoCAD. Those of you who have history with AutoCAD are no doubt married to your keyboard every bit as much as you are now married to your mouse. Visual LISP provides the functions that simplify creating shortcut macros to load, run, and unload your VBA macros. VBA still has need of the AutoCAD command line for performing certain commands or functions and is fully capable of calling and running both AutoLISP and Visual LISP routines. As has always been the case with AutoCAD, the best method of automation or customized control of AutoCAD is the method that is the easiest and most direct at the moment. In this series of Articles we will explore and learn VBA, but will not shy away from other interfaces when necessary.

(Continued on page 3)

VBA Foundations, part 1

A Tutorial in VBA for Beginners—The First of a Twelve Part Series

Richard L. Binning / rbinning@mediaone.net

In Summary, Autodesk has provided a built-in, easy to learn, development environment in VBA. They have also taken care of the plumbing and provided the hookups to connect VBA with AutoCAD. VBA can already talk to most of your Windows applications, so now you can control them from inside of AutoCAD. This ability travels in two directions, because now AutoCAD can be controlled by your other applications as well. Stay tuned for an in-depth exploration of VBA's editor (your programming toolbox) in the next issue of this series.

Richard Binning is the CADD Coordinator for The Haskell Company in Jacksonville, Florida. This position acts as a liaison between the Information Technology Department, user groups and functional departments, and Upper Management. "Basically, I translate 'Geek' to English and English to 'Geek'," he says. "I also coordinate and conduct in-house training and coordinate large-scale out of house training for CADD users. I oversee the in-house automation and customization of AutoCAD for 120+ CADD users. I am also an adjunct faculty member at the local ATC (AUTODESK TRAINING CENTER), TIS (Technology Institute of the South)."

We deeply appreciate Richard's commitment to our readers with this year long series.—ed.

