

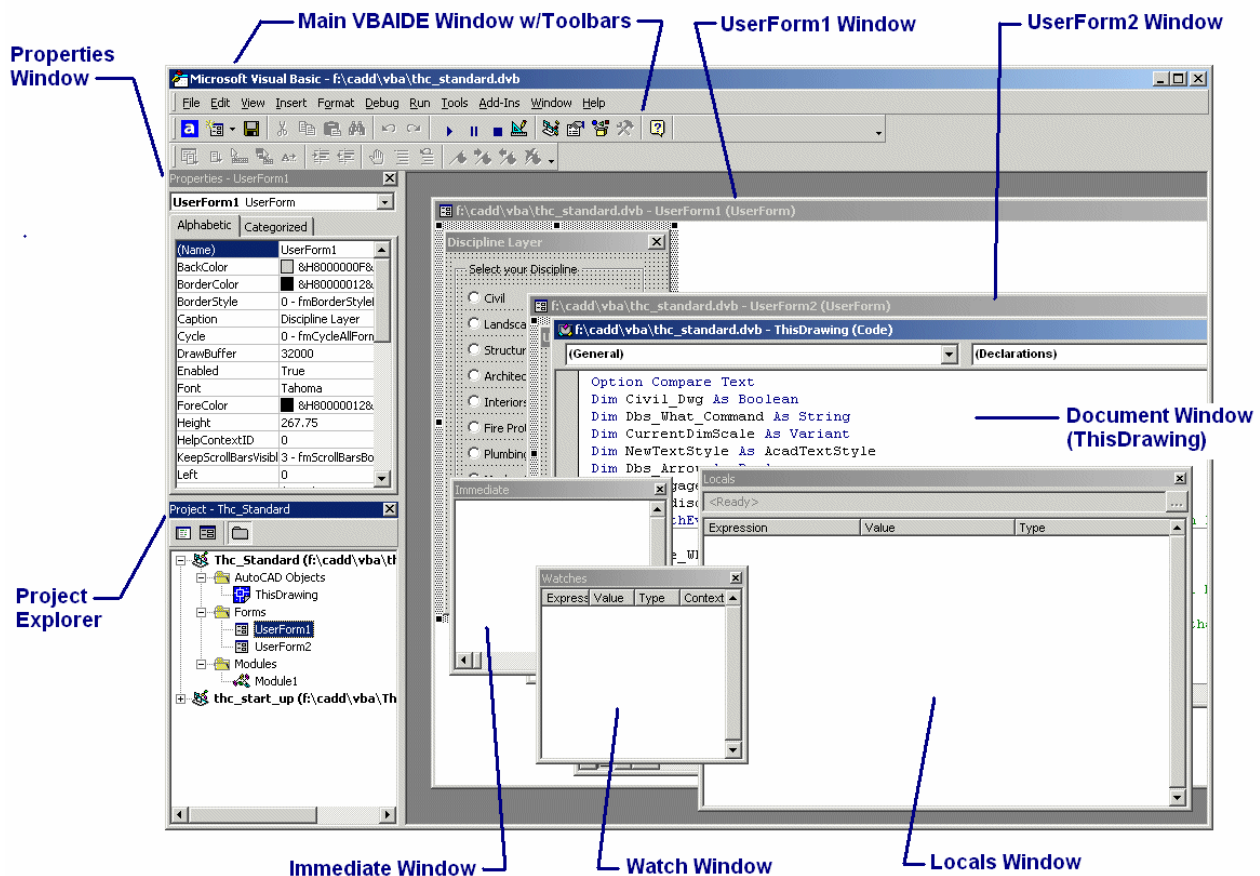
VBA Foundations, part 2

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

Richard L. Binning / rbinning@mediaone.net

Looking back at Bill Gate’s vision for the future (from circa 1991), he mentioned in his guest editorial, “... 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”. Describing his ideal interface further he stated, “Imagine designing the visual components of an application graphically—merely by placing controls on a form.” (Bill Gates, Chief Software Architect, MicroSoft, Inc.) Is it really that simple? In two words? “Yes and No”.

“NO”, because nothing is ever as easy as it may seem. Think for a moment about the recently completed Olympic games. Watching the gold medal winners perform their routines, or participate in their respective events often leaves you, the spectator, marveling at how simple and graceful these athletes make it seem. Yet behind the scenes these same athletes have spent countless hours of hard grueling practice and exercise in order to “perfect” the performance for which they were awarded the “gold.”



But don’t get discouraged. I haven’t told you about the “Yes” part of the answer. Again I ask, Is it really that simple? In some ways, I can happily report that it is, “YES” it is! Remember I mentioned how much code and effort has already been expended providing this graphical interface for you. Consider that there are

(Continued on page 2)

VBA Foundations, part 2

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

Richard L. Binning / rbinning@mediaone.net

more people writing Visual Basic programs than in any other programming language in the world. “*More than that, VB's ease of use and accessibility has brought the world of programming to millions of people who otherwise were too intimidated to become deeply involved with computers. In the sense that it has touched and in some cases profoundly transformed people's lives, VB has been a truly revolutionary product*”, (Ron Petrusa, O'Reilly Windows Programming and .NET editor, June 2001) When utilizing Visual Basic for Applications (VBA), you have a rich tradition of programming history to draw from and because the language uses everyday common language, pre-built visual objects, and a fully developed programming environment before long you will hear yourself say, “Is it really that easy? Did I really create that whole program?” At which point I will be happy to step up and say, “YES”! Especially after watching you do it.

Lets get started, we'll start by exploring the environment we'll be creating our programs in, through this exploration we will discover just what the Integrated Development Environment (IDE) has to offer. Take a look at the picture, on the previous page, of AutoCAD's VBAIDE.

The VBAIDE is made up of different windows and can be grouped according to their purpose or use. The “Design Group” includes the Code, the UserForm, and the Properties windows. The “Debugging Group” includes the Immediate, Locals, and Watch windows. Other windows include the toolbox, the object browser, and the project explorer. The parts and pieces that make up this IDE include the following windows grouped according to their usage:

Design Windows

- **Code Window** - This is where we will put the logic and decision making portions of our projects. In addition to the main code area are two combo boxes (drop down boxes) located at the top of the window.
 - **Object Box** – Located on the left side, this displays the main divisions of your code. These divisions change based on what this code refers to such as: UserForms, Modules, or This-Drawing. We'll discuss these in a later article.
 - **Procedures Box** – Located on the right side displays the individual functions, procedures, or pre-defined methods available to the object displayed in the Object Box.
- **UserForm Window** – This is our “Visual” playground, where we arrange the buttons, boxes, text, etc. that are displayed when we run our programs. We add these objects to our form(s) by dragging them from the “toolbox”. We modify these objects on our playground through the use of the properties window.
- **Toolbox** – This is where most of the “Visual” parts of our programs come from. By right-clicking on an empty portion of the “toolbox” we can add additional controls.
- **Properties Window** – Like the properties window inside of AutoCAD, when we select something such as a button or form we can adjust all the properties of that selection such as color, font, enabled, etc.

Debugging Windows

- **Immediate Window** – This is where we can test and explore commands and “code” or logic. This is where all your “debug.print” commands will be displayed. You'll learn more about

(Continued on page 3)

VBA Foundations, part 2

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

Richard L. Binning / rbinning@mediaone.net

“debug.print” in future articles.

- **Locals Window** – This is where we can see the values of parts of our program while it is running. As we verify our program and check the values of variables, we can expand and display the value of virtually any object in our AutoCAD drawing, or other applications if we are accessing them. More on this later.
- **Watch Window** – You can define certain variables, methods, etc. to be included in the watch window, so you can keep track of how the values change over time during your programs execution.

Project Windows

- **Project Explorer Window** – Like windows explorer, this window enables us to manage, add, and delete all the parts and pieces of the project we are creating. It works in a similar fashion and each category can be expanded or minimized by double-clicking the name or clicking the plus sign to the left of the category. Categories include the different types of objects available in your VBA project. It always shows all projects that are currently loaded, even if they are not currently running.
 - **AutoCAD Object** – ThisDrawing is a built in object that accesses the Document object of the current “Active” drawing.
 - **Module** – Modules contain code, functions, and subroutines (subs), declarations, etc. which can be made available to all the other objects in your project.
 - **Class** – Classes are the heart of Object Enabled programming, but are also generally difficult to understand for the beginner. We’ll explore classes as we get a little further in our understanding of VBA.
- **Object Browser Window** – Using the Object Browser Window allows us to explore what is made available to us by AutoCAD and other VBA enabled programs. This is the plumbing I mentioned in the first article. This is how we discover the properties, methods, and events of the ActiveX object we are interested in. In this browser we are able to search, navigate, and display help for any object, property, or event listed.

We will explore and utilize this development editor in greater depth in future articles, but feel free to explore it on your own as you wait for the next article. For those of you “old salt” AutoCAD veterans who are looking to extend your prodigious talents with a little more than just “fancy scripting” *have no fear*. You may be thinking that we have been getting further and further from the “command line” driven interface you have come to know and love. You’ll be happy to know that there are short cuts and hot key options for getting around this interface as well. Check out this spreadsheet for a comprehensive listing of “hot function keys” that will have you navigating the VBAIDE like a pro in no time.

In Summary, the VBAIDE provided “free” inside of AutoCAD is a graphically rich and easy to learn development environment that can help you debug your code in real-time. You won’t find an easier tool to step through your code in and the fact that the code is written in basically (pun intended) the same language (English) you use everyday just makes it that much better. In addition, the fact that there are over 8 million plus “Visual Basic” programmers using this interface means that there is plenty of code to learn from, borrow from, and expand upon. Please stay tuned to further explore, customize, and begin using this editor in the next issue of this series.