Yes, Real GUI Programming In UNIX

By Kurt Williams

Kurt brings 26 years of computer experience to his many roles as an engineering consultant for BASIS.

Can it be done? Wouldn't it be great? Since 1985, BBx® has been a write-it-one-time, run-it-on-any-platform product. Applications developed on a Novell network with DOS-based workstations can be moved to an HP-9000 running HP-UX with VT-100 terminals without modifying a single line of code. Write the application once and run it on any platform where a BBx interpreter is available. Portability like this makes a lot of money for developers. It allows them to focus on the application and let the customer decide what operating platform he or she wants.

Then BASIS introduced Visual PRO/5® and made GUI programming in Business Basic a reality. There is a problem, however. An application written in Visual PRO/5 is tied to a specified operating system. Visual PRO/5 applications must run on Windows-based workstations. There are no other options. What about Macintosh? What about X? What about Motif? What about the next great GUI system? This violates the founding tradition of our BBx product line.

BBj™ brings Business Basic GUI programming back to its portable roots. BBj programs run in the Java™ Virtual Machine (JVM). A GUI application developed on a Windows platform with BBj can be moved to any system where a JVM is available and it will run without modification. Once again, the developer can concentrate on the application and leave the platform selection up to the customer.

Visual PRO/5 provides a standard set of graphical controls that the developer uses to create GUI applications.

If the developer asks for a text box control, Visual PRO/5 graciously complies with the request. It does so by calling Windows-specific functions. BBj complies just as graciously, but it does so by using platform-independent classes provided by the Java Abstract Windows Toolkit (AWT) or the Java Swing component set. These classes worry about the low-level implementation of the GUI system. The BBj programmer can forget about the platform and focus on the application, secure in the knowledge that it will look the same wherever it runs.

So GUI programming *can* be done in UNIX but not until BBj is available. True enough, but developers can get started immediately. Any program written with Visual PRO/5 today for Windows will run unmodified tomorrow in the JVM using BBj. No need to wait. The developer can design, write and test the application using Visual PRO/5 and Windows and deploy it any GUI system in a Windows or UNIX environment with the proper JVM.

Even more benefit can be gained by using the GUIBuilder™ to develop the GUI application. GUIBuilder generates the event-dispatching loop on which all GUI programs rely. The developer simply provides the initialization routines, the code to handle each of the pertinent events that might occur and any end-of-job code.

Every component in a GUI application can generate events. Windows, menus, list boxes, text boxes, grid controls and all the others can generate a plethora of events such as focus

gains and losses, mouse clicks, content changes, selections made and many more. Some of these events are very important and require a response. Others are meaningless and can be safely ignored by the program. A Visual PRO/5 program must monitor the GUI event queue and handle each of these events as it occurs. It must look at the event data, decide if the event requires response and then dispatch the program flow to the proper code to handle the event. GUIBuilder constructs the event-monitor and dispatch code for the developer, thereby freeing him or her from routine, repetitive coding.

BBj will support this event-monitor system, but it also provides an easier method. A drawback of the event-queue system is that the program must sort through a large number of events looking for those events that require response. BBj programs can register the specific events that require responses with code that might look like this:

```
CALLBACK ( eventName, subroutineName, contextID {,controlID} )
```

Once all callbacks have been registered, the programmer replaces the entire event loop with the single line:

```
PROCESS_EVENTS
```

Whenever an event occurs on a control for which a CALLBACK has been registered, the registered subroutine will be called. When events occur for which a CALLBACK has not been registered, no processing occurs within BBj. This event model is more efficient and will help make programs (especially programs running in a thin client environment) run more efficiently.

A program written today using GUIBuilder will be generated with the standard, event-monitor type event loop. The resulting Visual PRO/5 program will run with Visual PRO/5 now and unmodified on BBj tomorrow. However, the GUIBuilder project file could be loaded in the BBj GUIBuilder and regenerated with the event-registration model, resulting in a program that was generated specifically for BBj. In either case, a tremendous benefit can be gained by working with GUIBuilder. The amount of code that must be written is greatly reduced and the event model that gets employed is the best event model for the current environment. Conversely, programs developed working directly in Visual PRO/5 will still run on BBj, but reworking the event model will take time and effort. With GUIBuilder, it's a simple recompile step.

While being able to write a program one time and then run it on any platform is not a reality for Visual PRO/5, it will be for BBj. But developers don't have to wait for BBj to get started. They can develop now with Visual PRO/5. Even better, they can develop now with GUIBuilder. The resulting applications will run only on Windows platforms now, but when BBj becomes available, they will run on any platform that has a Java Virtual Machine. And those GUIBuilder applications created today will take advantage of the improved event model tomorrow with nothing more than a simple recompile step.

BBj brings BASIS visual programming products in line with the BBx tradition of platform independence and easy portability. So yes, GUI programming in UNIX can be done and it will be great!