New Language Features Give Programmers More Choices

By Dr. Kevin King

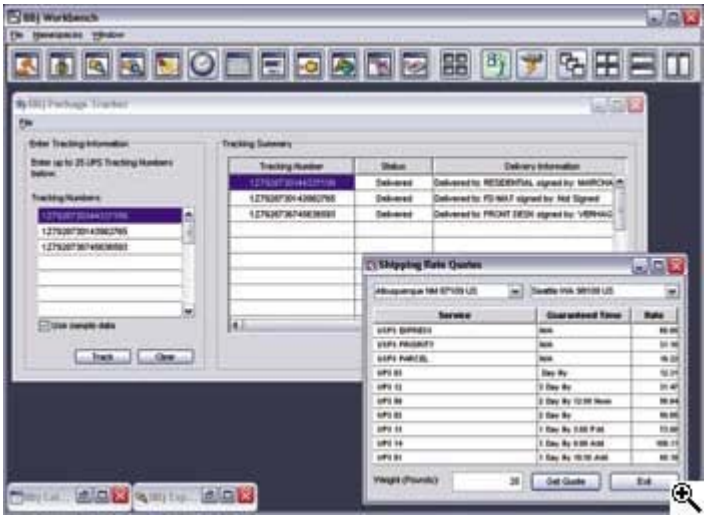
Traditionally, application developers depend on language providers for current and powerful features. BASIS continues its tradition of providing such features with BBj® 3.x, which includes the Multiple Document Interface (MDI), InterProcess Communication (IPC), Progress Bars, Popup Menus, Memory Graphing, and Automatic Licensing.

Multiple Document Interface (MDI)

Many GUI applications use MDI functionality to keep the application modules centrally contained within the master application's window. Microsoft Excel® is a common example of an MDI application. When users launch more than one workbook from Excel, all of the workbooks remain within the boundaries of the Excel master application. This traditional implementation of MDI allows the user to minimize one or more workbooks, without the iconized workbook getting lost among other open applications. In addition, the master application can cascade, tile, and align the workbook modules vertically and horizontally, allowing the user simultaneous access to the different workbooks.

BASIS gives developers this same functionality in their applications. The PRO/5®, Visual PRO/5® and BBj online manuals describe the simple process of retrofitting existing GUI applications to incorporate this new functionality. For example, programmers can now designate their menu program the MDI master and all of the SCALLed programs MDI children. As a result, the SCALLed programs remain constrained within the MDI master window, providing the same MDI cascading, tiling, window docking, and collective termination features that other visual development languages offer.

Figure 1. Multiple Document Interface (MDI).



InterProcess Communication (IPC)

For many years, developers wanted the ability to communicate between programs running in different BBj interpreter sessions. With the release of BBj 3.x, BASIS developers now have this ability. BASIS implemented this feature using shared memory in the JVM that runs BBj Services. When programmers need access to information in more than one application, they store it in BBjNamespace variables and register a callback on the variable. When the variable value changes, all programs with

callbacks registered on that variable receive an event signaling the change of the variable.

This implementation is quicker, cleaner, and more efficient than the traditional method of writing to a shared file on disk and programming loop structures that periodically check for changes in the shared file. Using this new interprocess communication, developers can easily write monitoring programs, instant messengers, and tightly integrated applications showing real-time inventory controls and account balance summaries, just to name a few.

Figure 2. InterProcess Communication (IPC).

Progress Bar and Timer Functionality

Despite the meteoric increase in processor and disk speeds, some processes still take a few seconds or more to complete. The traditional GUI solution for providing user feedback during these lengthy processes is to provide a progress bar displaying the progress of the running process. BBj 3.x contains a new progress bar control for these situations. Now, developers can create this new control programmatically or include it in a resource file with ResBuilder®. When programmers are unable to determine the length of time a process will take, they can use the indeterminate progress bar, which runs a mark back and forth across the progress bar, until the process finishes. When the programmer can determine the process's progress, he or she can use the new timer function and set it to fire at programmable intervals. The programmer can then check the status of the process, and properly paint the progress bar with the appropriate percentage necessary to give accurate feedback to the user. With the availability of the new progress bar and timer function, programmers can now provide the end user with much more intuitive feedback than the hourglass cursor available in previous product releases.

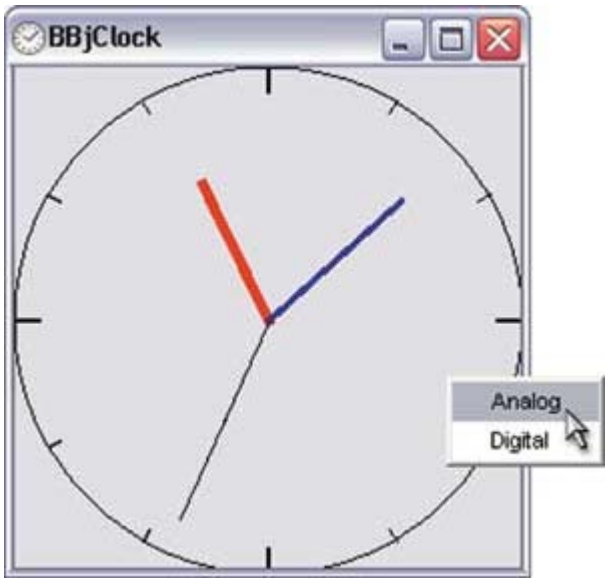
Figure 3. BBj 3.x Progress Bar.

Popup Menu

The Popup Menu is a context-sensitive menu that pops up when a user right-clicks on a GUI control. BASIS created a popup menu available to all appropriate BASIS GUI controls. Like other GUI controls, programmers add a popup menu to their applications, either programmatically or through ResBuilder. The popup menu control is a powerful feature that allows developers to add simple or complex nested menus to any control. Before BASIS added this control to the language, developers had to write a significant amount of code to determine where the cursor was before popping up a custom window to emulate a menu. Now, the language associates the popup

menu with the underlying GUI control, simplifying its use and adding the existing menu control capability to the popup menu control.

Figure 4. Popup Menu example.



Java Memory Management

Java memory management continues to provide new and interesting challenges for BBj deployment. To simplify the memory management process, BASIS has added a BBj Services memory graphing function in the BBj Enterprise Manager. This memory-graphing tool parses the **`BBjServices.out`** log files and plots the amount of used memory recorded in the files every time a BBj interpreter starts. The best way to use this graph is to view it after BBj Services records several hundred connections. As an opportunistic memory consumer, Java eventually uses as much memory as the administrator allows when the java -Xmx command line parameter is set. By default, the JVM uses 64 megabytes of memory. Therefore, if BBj runs out of memory, the administrator must increase the -Xmx parameter until BBj runs without generating "out of memory" errors. As the administrator increases the -Xmx value, it is important that the system have enough unused physical memory to accommodate the -Xmx value of the JVM, preventing the OS from swapping out the Java memory to the swap file on the disk.

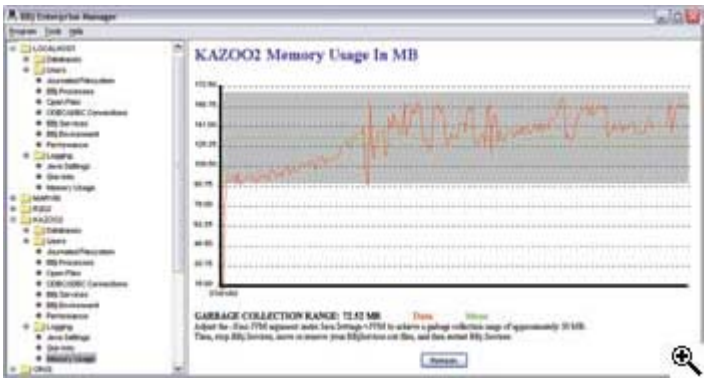
Once BBj Services have enough memory, it is time to start using the memory-graphing tool. Now that BBj has enough memory to run the applications in a production environment, it is possible to begin tuning the amount of memory allocated to the JVM. After hundreds of connections, the memory-graphing tool reveals the garbage collection patterns in the JVM. The pattern is typically saw tooth, which is caused by the JVM's internal garbage collector.

During the normal running of BBj Services, the JVM continuously consumes more memory than the "incremental" garbage collector can free. Once the JVM approaches the upper bound of the -Xmx value, it suspends the running processes, does a full garbage collection, and frees all of the memory not consumed by the running processes. As long as the "full" garbage collection frees 40 to 50 megabytes of memory, users usually do not notice when a full garbage collection occurs. However, if the full garbage collection frees more memory than this, the end users experience application pauses. To provide better performance for the end users, reduce the -Xmx value until the full garbage collections free only 40 to 50 megabytes of memory.

Remember that the changes to the -Xmx value take effect only after restarting BBj Services. The graphing function in the Enterprise Manager automatically applies a

standard deviation function that bands the average full garbage collection delta to simplify the tuning procedures. In the past, many releases of the JVM included different garbage collection algorithms, so it is important to monitor the used memory graph after introducing a new JVM in production.

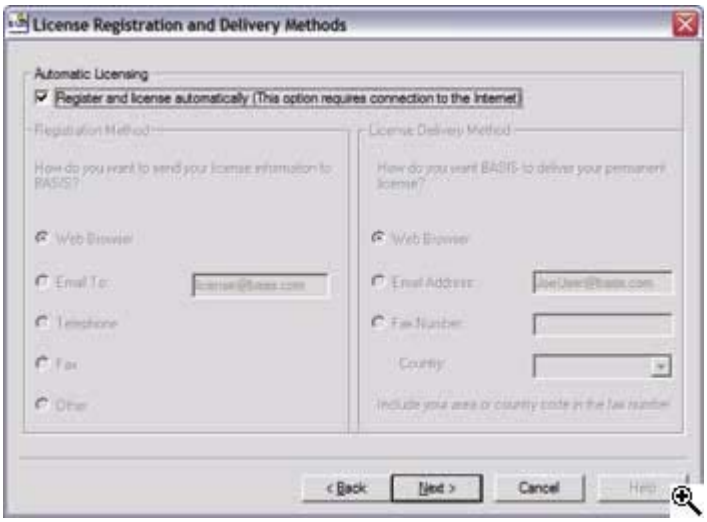
Figure 5. BBj Enterprise Manager's memory graphing capabilities.



Automatic Licensing

As of BBj 3.01 and Visual PRO/5 4.1, both products feature a new automatic licensing option on Win32 platforms. This automation reduces many of the licensing challenges encountered over the past few years. During the registration and licensing process, customers with an Internet connection can register and license their software with a single click of their mouse after entering their serial and authorization numbers. The BASIS registration and licensing wizards utilize the Internet connectivity setting in MS Internet Explorer to circumnavigate any proxies and firewalls and to register and install the necessary license.

Figure 6. Automatic license registration.



The automatic license feature and the customized Win32 license executable are delivered when the user chooses Web browser or email options. The license delivery mechanism:

- Ensures that the license was not previously installed
- Ensures that the downloaded license file is the correct file for that machine (validates hostid)
- Checks for conflicting licenses by scanning the BLM directory, reading all license files for same feature(s), and renaming conflicting files to have an extension of **`.bak`**

- Installs the license file into the appropriate directory

Additionally, the automatic license delivery mechanism uses the following licensing algorithms:

- If the license is a multi-user license and a BLM is running, the BLM will perform a re-read function in order to obtain a new license file
- If the license is a multi-user license and the BLM is not running, the program starts the BLM
- If the license file is a multi-user license and selection of a BLM has not been previously performed, then upon user notification, BBj Services will be stopped, branded (the license file location will be written), and then restarted
- If the BBj License Wizard and/or the Visual PRO/5 installation program are running, the program terminates them

BASIS continues to enhance its products based on customer feedback. Please e-mail any ideas for additional features and enhancements that can benefit the BASIS user community to info@basis.com.

Click HERE for the "New Language Features Give Programmers More Choices" source code.