New Technical Notes

Macintosh

Developer Support

Background-Only Applications

Processes **M.PS.BackgroundOnlyApp**

Written by: Greg Robbins and C. K. Haun December 1992

Background-only applications (BOAs) are Macintosh applications that run only in the background under MultiFinder and under System 7. BOAs are the preferred alternative to INITs, drivers, and stand-alone code for most startup-time and background "daemon" services. This note discusses various issues that affect development of BOAs, as well as useful implementation strategies for BOAs.

Introduction

Background-only applications (BOAs), also called faceless background applications (FBAs) or application extensions, are the easiest way to provide either startup-time or continuous background services. Although BOAs work under System 6 and MultiFinder, the process management and interprocess communication facilities of System 7 allow BOAs to provide far more sophisticated services.

This note covers some of the uses and pitfalls of developing background-only applications. For an introduction to writing BOAs, see "Be Our Guest" in issue 9 of *develop.* Sample Pascal and C code for background-only applications is available in the Snippets collection on the *Developer Series CD* as pSmallDaemon.p and cSmallDaemon.c.

How to Be a BOA

Background-only applications tend to look like reduced Macintosh applications. A BOA should call InitGraf to initialize QuickDraw globals (which is necessary to use the Apple Event Manager), but a BOA must not call InitWindows, InitMenus, or any toolbox or operating system routine that might draw on the screen. This rules out making any call from a BOA that might raise a dialog either intentionally (like PPCBrowser) or unintentionally (like ResolveAlias). There are typically alternative routines that avoid the need for user interaction (such as IPCListPorts and MatchAlias) that can be used from within a BOA. One-way communication from a BOA to the user can be done via the Notification Manager (see "Posting Notifications" later in this note).

Like all other Macintosh applications, a BOA should handle the four required Apple events.

It is particularly important that the QuitApplication Apple event be supported, as System 7 will refuse to shut down if a BOA does not quit as requested.

Be sure to set the canBackground and backgroundOnly bits in the application's SIZE -1 resource to indicate to the system that the application is a background-only process.

Stack Issues

The stack space allocated to BOAs by default is only 2K. This is much smaller than the usual default stack size of 8K (on Macintosh computers without Color QuickDraw) or 24K (on Macintosh models with Color QuickDraw). Any recursive or stack-intensive operation can easily cause the BOA's stack to crash into its heap. This can typically be avoided by using standard looping constructs rather than recursion and by using heap allocations (NewHandle, NewPointer, or for transitory needs, TempNewHandle) instead of large parameters or local variables. Be careful to avoid declaring Str255s as local variables, as those will quickly use up stack space.

If it is necessary to increase a BOA's stack, it can be done by calling GetApplLimit and SetApplLimit to reduce the BOA's heap space:

```
// Increase the space allocated for the application stack.
//
// Warning: SetApplLimit always sets the stack to at least as large as the
//    default stack for the machine (8K on machines with original QuickDraw,
//    24K on machines with Color QuickDraw), so the application partition
//    must be large enough to accommodate an appropriate stack and heap.
//
//    Call this only once, at the beginning of the application.

OSErr IncreaseApplicationStack(Size incrementSize)
{
  OSErr retCode;

  // Increase the stack size by lowering the heap limit.
  SetApplLimit((Ptr) ((unsigned long) GetApplLimit() - incrementSize));
  retCode = MemError();
  if (retCode == noErr) MaxApplZone();

  return retCode;
}
```

Posting Notifications

When an exceptional condition needs to be made known to the user, a BOA can use the Notification Manager to display a string or play a sound. Since a BOA is not listed in the application menu and cannot be brought to the front like other applications, the user may not even know that it is running, so notifications can be startling. Use the Notification Manager only to report a serious problem or to request that the user take some needed action.

The Finder and BOA Launching

There are numerous ways to launch a BOA. If a BOA has the file type `APPL`, it can be opened by the user in the Finder just like other applications. It can also be placed in the Startup Items folder under System 7 for launching when the system starts up.

If a BOA has the file type `appe`, it is an *application extension*. If the user drags an application extension file to the System folder, the extension will be routed to the Extensions folder and then launched when the system starts up.

If an INIT resource is in an application extension file and the file is in the Extensions folder, the system will execute the INIT code along with the INITs of other extensions. This can be useful for displaying an icon during booting. (Displaying an icon from an INIT at startup is typically done by using the ShowInit code, available on the *Developer Series CD*.) Since the launch of the BOA will not actually occur until after all INIT resources have been run, a BOA's startup icon cannot be informative. For example, the BOA can't display an X over its icon to indicate that loading has failed, because the BOA has not yet been launched at INIT time. However, showing an icon during booting is still useful to remind users that the application extension may be installed and eating up memory.

A BOA can be launched with the Process Manager routine LaunchApplication. LaunchApplication ignores the type of files it is asked to launch, so the BOA can be launched with LaunchApplication whether its file type is `APPL`, `appe`, or something else.

Under System 7, application extensions are launched before the Finder, but they actually start running after the Finder is up. To understand this, remember that a call to LaunchApplication simply queues a file for later launching. The actual launch is done by the Process Manager at the next call to WaitNextEvent. The system queues application extensions for launching before the Finder is started. Then the Finder starts up, and when the Finder calls WaitNextEvent, the BOAs in the launch queue start running. This is why application extensions have lower process serial numbers than the Finder, even though the Finder is loaded first.

The System 7 Finder is unable to determine when a BOA stops running. As a result, a BOA icon that was dimmed when the user launched the application by opening it in the Finder will not be undimmed if the BOA quits. This is just a cosmetic problem, as the user can relaunch the BOA by double-clicking the still-dimmed icon.

BOA Tasks

Faceless background applications can do most of the jobs formerly handled by stand-alone code and by periodically executed drivers. The primary reason to use an INIT instead of a BOA is to make global trap patches. Because a BOA is an application, a trap address set from within the BOA will apply only to the BOA, not globally.

A BOA can not ever explicitly draw on the screen. While safely creating and initializing an appropriate world for drawing from stand-alone code (typically from an INIT) is quite difficult, it can be done. If any drawing will be necessary, do not use a BOA.

It is safe and appropriate to store interrupt routines within the heap of a BOA. For example, a Time Manager task's code or a completion routine may actually be a procedure of a BOA.

Since all of its heap space is released once a BOA quits, a BOA must not quit while any interrupt or completion routines are pending. A BOA may be put to sleep (by passing a large sleep parameter to WaitNextEvent) and, under System 7, later woken up at interrupt time with a call to WakeUpProcess.

Any application that is not performing a periodic task should use a large sleep value in its calls to WaitNextEvent. Since an application is woken up whenever any events are pending for it, using large sleep values will help avoid slowing down the execution of other processes without

hurting the sleeping process's responsiveness. A background-only application can run periodically (as a daemon) by passing an appropriately small sleep value to WaitNextEvent.

Background Apple Events

Although Apple events are an excellent medium for communication between a BOA and other processes, be careful to avoid situations that may require user interaction. For example, attempting to target a process on another machine for which there is no current session would raise the program linking dialog. A BOA can usually avoid those situations by handling Apple events and responding with the reply event, or by saving the address of an event sender (by calling AEGetAttributePtr to extract the keyAddressAttr attribute of an Apple event sent to the BOA) and using it later to target a response.

Further Reference:

- *Inside Macintosh*, Volume VI, Event Manager, Process Management, Finder Interface, and Apple Events chapters
- "Be Our Guest: Background-Only Applications in System 7" in issue 9 of *develop* introduces BOAs.
- Macintosh Technical Note M.PT.StandAloneCode "Stand-alone code, *ad nauseam*" (formerly #256) discusses requirements for INITs.