About Process Management Information on sharing resources

The **Process Manager** schedules the processing of all applications and desk accessories, and allows multiple applications to share the CPU and other resources. Applications share the available memory and access to the CPU. Several applications can be open (loaded into memory) at once, but only one uses the CPU at any one time.

The **Process Manager** manages the scheduling of processes. A **process** is an open application or, in some cases, an open desk accessory. (Desk accessories that are opened in the context of an application are not considered processes.) In version 7.0, the number of processes is limited only by available memory.

The **Process Manager** maintains information about each process-for example, the current state of the process, the address and size of its partition, its type, its creator, a copy of its low-memory globals, information about its 'SIZE' resource, and a process serial number. This process information is referred to as the **context** of a process. The **Process Manager** assigns a **process serial number** to identify each process. A process serial number identifies a particular instance of an application; this number is unique during a single boot of the local machine.

The **foreground process** is the one currently interacting with the user; it appears to the user as the active application. The foreground process displays its menu bar, and its windows are in front of the windows of all other applications.

A background process is a process that is not currently interacting with the user. At any given time a process is either in the foreground or the background; a process can switch between the two states at well-defined times.

The foreground process has first priority for accessing the CPU. Other processes can access the CPU only when the foreground process yields time to them. There is only one foreground process at any one time. However, multiple processes can exist in the background.

An application that is in the background can get CPU time but can not interact with the user while it is in the background. (However, the user can choose to bring the application to the foreground-for example, by clicking in one of the application's windows.) Any application that has the canBackground flag set in its 'SIZE' resource is eligible to obtain access to the CPU when it is in the background.

Applications can be designed without a user interface. A background-only application is an application that does not have a user interface. A background-only application does not call the InitWindows routine and is identified by having the onlyBackground flag set in its 'SIZE' resource. Background-only applications do not display windows or a menu bar and are not listed in the Application menu. Background-only applications and applications that can run in the background should be designed to relinquish the CPU often enough so that the foreground process can perform its work and respond to the user.

Once an application is executing, in either the foreground or the background, the CPU is available only to that application. The application can be interrupted only by hardware interrupts, which are transparent to the

scheduling of the application. However, to give processing time to background applications and to allow the user to interact with the foreground application or switch to another application, you must periodically relinquish the CPU using the **WaitNextEvent** or **EventAvail** functions. (You can also use the **GetNextEvent** function; however, you should use **WaitNextEvent** to provide greater support for multitasking.) Use these **Event Manager** functions to let the user interact with your application and also with other applications.