About the Event Manager

The **Event Manager** sends events to other applications and receives events from other applications. The **Event Manager** in System 7.0 provides routines for sending and receiving a new type of event, called
high-level events. In addition, Apple has defined a protocol for high-level events.The protocol is called the Apple Event Interprocess Messaging Protocol. High-level events that adhere to this protocol are called Apple events and these are handled with the **Apple Event Manager**. Your application can also define other types of high-level events and send them to applications, either locally or across a network.

The **Event Manager** includes the following topics:

- Low-Level Events
- Operating-System Events
- High-Level Events
- Sending High-Level Events
- Receiving High-Level Events
- Requesting Return Receipts
- Responding to Events
- Searching for a Specific High-Level Event
- Identifying Senders and Receivers
- Event Loops
- Event Mask
- Context Switching
- 'SIZE' Resource
- Creating a 'SIZE' Resource

This description also provides some information about Apple events, Apple's protocol governing a class of high-level events. Additional information about Apple events, including descriptions of how to process the required Apple events, is provided in the **Apple Event Manager** description.

Operation of the multitasking environment, formerly known as **MultiFinder** which is now an integral part of the Macintosh Operating System 7.0 is described. In this environment, numerous applications can be open simultaneously, cooperatively sharing the available system resources. The Macintosh Operating System coordinates the execution of multiple applications by sending another type of event (an operating-system event) to applications whenever their execution status changes or whenever processor time is available for background processing. Your application takes advantage of this multi-tasking capability primarily by receiving operating-system events that guide its execution. The operating-system events are the suspend event and the resume event.

In System 7.0+, the cooperative multitasking capabilities previously available through **MultiFinder** are an integral part of the Operating System. As a result, applications running under version 7.0 or greater must process events and reserve memory in ways that contribute to the smooth operation of all applications that are open. In practice, this means that you should retrieve events from the **Event Manager** by using the **WaitNextEvent** function and that you should include a 'SIZE' resource that specifies a reasonable memory partition size. The **Event Manager** allows your application to retrieve events, to mask out unwanted events, and to specify memory and scheduling options for your application.

The **Event Manager** routines that let your application communicate with other applications depend on the services of the Program-to-Program Communications **PPC Toolbox** and are available in System 7.0+. Before using any of the routines that handle high-level events, you should first use the **Gestalt** function to determine that the **PPC Toolbox** is present. You can also use **Gestalt** to determine which multitasking features of the Operating System are present.

The ability to have multiple applications open at once is available when running System 7.0+ or when running **MultiFinder** in System versions 5.0 and 6.0. Any significant differences between the multitasking environment of System 7.0 and that provided by **MultiFinder** in earlier system versions are noted where appropriate. In system software earlier than version 7.0, there is no recommended way to determine whether **MultiFinder** is running or whether other applications are open if it is running. When running in System 7.0, applications that need to know what other applications are open (for example, to send high-level events to them) can get that information by calling one of three functions: the **PPCBrowser** function or the **IPCListPorts** function or the **GetNextProcess** function (documented in the **Process Management** description).

The sections on cooperative multitasking supersede the information in the *Programmer's Guide to MultiFinder*.

To use the **Event Manager**, you should first be familiar with the way in which the Macintosh Operating System manages processes. See "About Process Management" in the **Process Management** description for details about how the Operating System schedules processes, performs context switches, and launches applications. If you want to communicate with applications across a network, then you should be familiar with the discussion of **authentication** in the **PPC Toolbox** description.

You can use the **Event Manager** to:

- receive key presses and mouse clicks as input for your application
- receive indication that your application's windows need to be activated or updated
- allow other applications to use the available system resources when no events are pending for your application
- send events to other applications
- receive events from other applications
- respond to events received from other applications
- search for a specific event from another application

Most Macintosh programs are event-driven: they decide what to do from moment to moment by asking the **Event Manager** for events and responding to them one by one in whatever way is appropriate. The **Event Manager** is your application's primary link to the user, to other applications that are running at the same time as your application, to the various managers that are controlling operations in the Macintosh, and to the

Operating System itself. Events sent to your application from these various sources can communicate important information to it and help ensure its smooth operation.

Using the Event Manager

You can use the **Event Manager** to receive information about hardware-related events, about changes in the appearance of your application's windows, or about changes in the operating status of your application. You can also use the **Event Manager** to communicate directly with other applications. This communication can include sending events to other applications, receiving events from other applications, and searching for specific events from other applications.

The events that your application can send to and receive from other applications are called high-level events. Your application can both send and receive high-level events, but it generally only receives **low-level events** and should not send them. Your application receives both low-level and high-level events in the same way, which is by asking the **Event Manager** for the next available event. If the Event your application receives is a high-level event, your application might need to use another **Event Manager** routine to retrieve an optional data buffer accompanying that event.

The four general types of events are:

- low-level events
- operating-system events
- high-level events
- Apple events

An Apple event is a particular kind of high-level event. Apple events are used in communications between applications. The low-level operating system events are the suspend event ,the resume event, the mouse-moved event, and the application-died event.