Components of Apple Events

An Apple event consists of attributes (which identify the Apple event and denote its task) and, often, parameters (which contain data to be used by the target application). An application uses the **Apple Event Manager** to create an Apple event. Using arguments you pass to the **AECreateAppleEvent** function and to other **Apple Event Manager** routines, the **Apple Event Manager** constructs the necessary data structures containing attributes and parameters and converts these structures into an Apple event. Applications must use the **AESend** function to transmit the Apple event. After receiving an Apple event, applications must use **Apple Event Manager** routines to extract the attributes and parameters of the event.

Attributes are a fundamental component of Apple events. **Apple event attributes** are records that identify the Event Class, eventID, target application, and other characteristics of an Apple event. Taken together, the attributes of an Apple event denote the task to be performed on any data specified in the Apple event's parameters. You do not have any direct way to access the data stored in these records. You must use **Apple Event Manager** routines to extract or specify the attributes.

An **Apple event parameter** is a record containing data that the target application uses. Unlike Apple event attributes (which contain information that can be used by both the **Apple Event Manager** and the target application), Apple event parameters contain data used only by the target application. For example, an attribute like the eventID is used by the **Apple Event Manager** to call a handler from the server application's dispatch table, and the server application must have a handler to process the event identified by that attribute. By comparison, the list of documents contained in a parameter to an Apple events event is used only by the server application. As with attributes, you do not have any direct way to access the data structure of a parameter. Use **Apple Event Manager** functions to extract data from or put data into parameters.

Note that Apple event parameters are different from the parameters of **Apple Event Manager** functions. Apple event parameters are records private to the **Apple Event Manager**; function parameters are arguments you pass to the function or that the function returns to you. You typically specify the Apple event parameters (as well as the attributes) in parameters to **Apple Event Manager** functions. For example, the **AEGetParamPtr** function uses a buffer to return the data contained in an Apple event parameter. You specify which Apple event parameter in one of the parameters of the **AEGetParamPtr** function.

Apple events are identified by their EventClass and eventID attributes. The **event class** is the attribute that identifies a group of related Apple events. The event class appears in the message field of the EventRecord for an Apple event. For example, the four required Apple Events (in fact, all core Apple events) have the value 'aevt' in the message fields of their EventRecords. The value 'aevt' can also be represented by the kCoreEventClass constant. Several event classes are shown here.

Event class	Description
kCoreEventClass	A core Apple event
kAEFinderEvents	An event that the Finder accepts

THINK Reference © 1991-1992 Symantec Corporation

kSectionEventMsgClass An event sent by the **Edition Manager**

The **eventID** is the attribute that identifies the particular Apple event within its event class. In conjunction with the event class, the eventID uniquely identifies the Apple event and communicates what action the Apple event should perform. (The eventIDs appear in the where field of the EventRecord for an Apple event.) For example, the eventID of an Open Documents event has the value 'odoc' (which can also be represented by the kAEOpenDocuments constant). The kCoreEventClass constant in combination with the kAEOpenDocuments constant identifies the Open Documents event to the **Apple Event Manager**.

Shown here are the event IDs for the four required Apple Events.

Event ID	Description
kAEOpenApplication	Open your application
kAEOpenDocuments	Open documents
kAEPrintDocuments	Print documents
kAEQuitApplication	Quit your application.

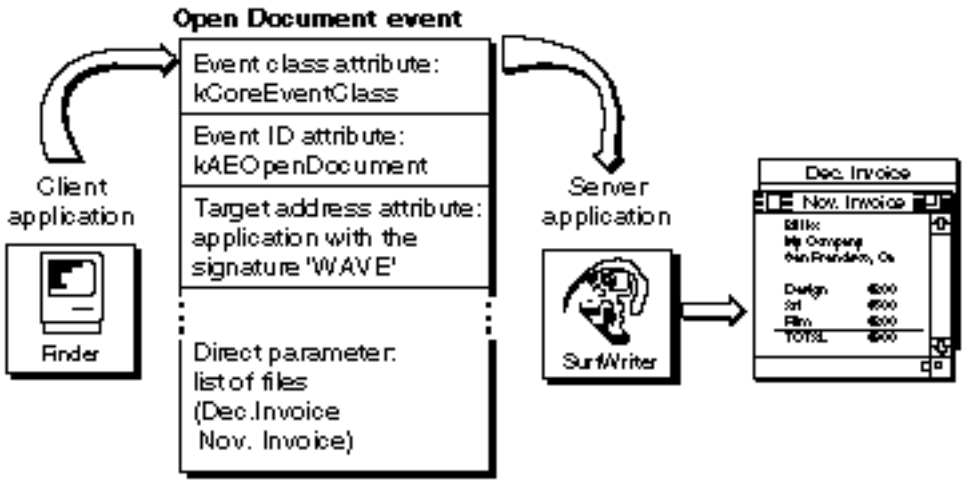
The target application's address is another required attribute. As previously described, the target application is the one addressed to receive the Apple event. Your application can send an Apple event to itself or to another application (on the same computer or on a remote computer connected to the network).

As with attributes, there are various types of Apple events. A direct parameter contains the data to be acted upon by the server application. For example, a list of documents is contained in the direct parameter of the Print Documents event. Direct parameters are usually required parameters-parameters that the server application needs in order to carry out the task denoted by the Apple event. Some Apple events also take additional parameters, which the server application uses in addition to the data specified in the direct parameter. For example, an Apple event for arithmetic operations may include additional parameters that specify operands in an equation. Additional parameters may be required or optional.

An optional parameter is a supplemental parameter that also can be used to specify data to the server application. Optional parameters need not be included in an Apple event; default values for optional parameters are part of the event definition. The server application that handles the event must supply default values if the optional parameters are omitted.

The following figure shows in greater detail the components of the Open Documents event that was introduced in the previous figure.

THINK Reference © 1991-1992 Symantec Corporation



Major components of an Open Documents event

To process the information contained in the Open Documents event, the SurfWriter application uses the **AEProcessAppleEvent** function. The **AEProcessAppleEvent** function provides an easy way for your application to identify the event class and event ID of the Apple event and to direct the **Apple Event Manager** to call the code in your program that handles the Apple event.

THINK Reference © 1991-1992 Symantec Corporation