Receiving Apple Events From the Edition Manager

Applications that use the **Edition Manager** must support Apple events. This requires that your application support the required **Open Documents** event and Apple events sent by the **Edition Manager**. See the description of **Open Documents** in the **Apple Event Manager**.

Apple events sent by the **Edition Manager** arrive as high-level events. The **EventRecord** data type defines the event record.

The **Edition Manager** can send the following Apple events:

- Section Read events ('sect' 'read')
- Section Write events ('sect' **'writ'**)
- Section Cancel events ('sect' 'cncl')
- Section Scroll events ('sect' **'scrl'**)

Each time your application creates a publisher or a subscriber, the **Edition Manager** registers its section. When an edition is updated, the **Edition Manager** scans its list to locate registered subscribers. For each registered subscriber that is set up to receive updated editions automatically, your application receives a Section Read event.

If the **Edition Manager** discovers that an edition file is missing while registering a publisher, it creates a new edition file and sends the publisher a Section Write event.

When you receive a Section Cancel event, you need to cancel the specified section. Note that the current **Edition Manager** does not send you Section Cancel events, but you do need to provide a handler for future expansion.

If the user selects a subscriber within a document and then selects the menu command, **Open Publisher** in the subscriber options dialog box, the publishing application receives the Open Documents event and opens the document containing the publisher. The publishing application also receives a Section Scroll event. Scroll to the location of the publisher, display this section on the user's screen, and turn on its border.

See **Opening and Closing a Document Containing Sections** for detailed information on registering and unregistering a section and writing data to an edition. See **Using Publisher and Subscriber Options** for information on publisher and subscriber options.

After receiving an Apple event sent by the **Edition Manager**, use the **Apple Event Manager** to extract the section handle. In addition, you must also call the **IsRegisteredSection** function to determine whether the section is registered. It is possible (due to a race condition) to receive an event for a section that you recently disposed of or unregistered. One way to ensure that an event corresponds to a valid section is to call the **IsRegisteredSection** function after you receive an event.

err = **IsRegisteredSection** (*sectionH*);

The following Listing illustrates how to use the **Apple Event Manager** and install an event handler to handle Section Read events. You can write similar code for Section Write events, Section Scroll events, and Section Cancel events.

// Accepting Section Read events and verifying if a section is registered

```
// Assuming inclusion of <Macheaders>
#include <AppleEvents.h>
#include <Editions.h>

// Make sure you place the following call in your initialization code

//MyErr = AEInstallEventHandler(sectionEventMsgClass,
//                        sectionReadMsgID , &MyHandleSectionReadEvent,
//                        0,FALSE);

// This the correct prototype for MyHandleSectionReadEvent
pascal OSErr MyHandleSectionReadEvent(AppleEvent,AppleEvent,long);


//This is the routine the Apple Event Manager calls when a Section Read
// event arrives.

pascal OSErr MyHandleSectionReadEvent(AppleEvent theAppleEvent,
    AppleEvent  reply, long refCon)
{
    OSErr getErr;
    SectionHandle sectionH;

    // Prototype user defined functions
    OSErr GetSectionHandleFromEvent(AppleEvent, SectionHandle *);
    OSErr DoSectionRead(SectionHandle);


    //Get section handle out of Apple event message buffer.
    getErr = GetSectionHandleFromEvent(theAppleEvent, &sectionH);


    if (getErr == noErr) {

        //Do nothing if section is not registered.
        if (IsRegisteredSection(sectionH) == noErr)
            return DoSectionRead(sectionH);
    }
    else
        return getErr;
}



// The following routine should read in subscriber data and update its
// display.
```

```
OSErr DoSectionRead(SectionHandle subscriber)
{

     // Your code here.
}


// This is part of your Apple event-handling code.
OSErr  GetSectionHandleFromEvent (AppleEvent theAppleEvent,
        SectionHandle *sectionH)
{
    DescType  ignoreType;
    Size ignoreSize;

    // Parse section handle out of message buffer.

    return  AEGetParamPtr(&theAppleEvent,   // event to parse
                keyDirectObject,  // Look for direct object
                typeSectionH,     // Want a Sectionhandle type
                &ignoreType,      // Ignore type it could get
                (Ptr)&sectionH,    // Put SectionHandle here
                sizeof(sectionH), // size of storage for sectionhandle
                &ignoreSize       // Ignore storage it used
                );
}
```