Getting Files Selected from the Finder

```
// Getting Files Selected from the Finder
// A code example demonstrating how to get the files
// that the user selected in the Finder before opening
// your application.  You must have the High-Level
// Event Aware flag in the application's 'SIZE' resource
// set in order for the Apple Event code to work.
// The code that does not use apple events will work
// properly under both System 6 & 7.



// Assumes inclusion of <MacHeaders>

#include <stdio.h>
#include <GestaltEqu.h>
#include <AppleEvents.h>

void InitToolbox(void);
Boolean AppleEventsInstalled (void);
pascal OSErr  MyHandleODoc (AppleEvent *theAppleEvent, AppleEvent* reply, long
                                                    handlerRefCon);
pascal OSErr  MyHandlePDoc (AppleEvent *theAppleEvent, AppleEvent *reply, long
                                                    handlerRefCon);
pascal OSErr  MyHandleOApp (AppleEvent *theAppleEvent, AppleEvent *reply, long
                                                    handlerRefCon);
OSErr MyGotRequiredParams (AppleEvent *theAppleEvent);


Boolean gDone = FALSE;

void InitToolbox()
{
    InitGraf(&thePort);                 // Standard initialization calls
    InitFonts();
    InitWindows();
    InitMenus();
    TEInit();
    InitDialogs(nil);
    InitCursor();
}

Boolean AppleEventsInstalled ()
{
    OSErr err;
    long  result;

    // THINK C's MacTraps library provides glue for Gestalt, so
    // it can be called safely under System 6. If an error is
    // returned, then Gestalt for the AppleEvents Selector is
    // not available (this also means that Apple Events are
    // not available)
    err = Gestalt (gestaltAppleEventsAttr, &result);
    return (!err && ((result >> gestaltAppleEventsPresent) & 0x0001));
                                        // return TRUE if there is no
```

```
                                                // error and the proper bit of
                                                // result is set
}

pascal OSErr  MyHandleODoc (AppleEvent *theAppleEvent, AppleEvent* reply, long
                                                              handlerRefCon)
{
    FSSpec myFSS;
    AEDescList      docList;
    OSErr           err;
    long            index,
                    itemsInList;
    Size            actualSize;
    AEKeyword       keywd;
    DescType        returnedType;

    // get the direct parameter--a descriptor list--and put it into a docList
    err = AEGetParamDesc (theAppleEvent, keyDirectObject, typeAEList,
            &docList);
    if (err)
            return err;

    // check for missing parameters
    err = MyGotRequiredParams (theAppleEvent);
    if (err)
            return err;

    // count the number of descriptor records in the list
    err = AECountItems (&docList, &itemsInList);

    // now get each descriptor record from the list, coerce the returned
    // data to an FSSpec record, and open the associated file
    printf ("Files to open:\n");
    for (index = 1; index <= itemsInList; index++) {

            err = AEGetNthPtr (&docList, index, typeFSS, &keywd,
                            &returnedType, (Ptr) &myFSS, sizeof(myFSS),
&actualSize);
            if (err)
                    return err;

            printf ("%s\n", PtoCstr(myFSS.name));
    }

    err = AEDisposeDesc (&docList);
    return noErr;
}

pascal OSErr  MyHandlePDoc (AppleEvent *theAppleEvent, AppleEvent *reply, long
                                                              handlerRefCon)
{
    FSSpec myFSS;
    AEDescList      docList;
    OSErr           err;
    long            index,
                    itemsInList;
```

```
	Size		actualSize;
	AEKeyword	keywd;
	DescType	returnedType;

	// get the direct parameter--a descriptor list--and put it into a docList
	err = AEGetParamDesc (theAppleEvent, keyDirectObject, typeAEList,
					&docList);
	if (err)
		return err;

	// check for missing parameters
	err = MyGotRequiredParams (theAppleEvent);
	if (err)
		return err;

	// count the number of descriptor records in the list
	err = AECountItems (&docList, &itemsInList);

	// now get each descriptor record from the list, coerce the returned
	// data to an FSSpec record, and open the associated file
	printf ("Files to print:\n");
	for (index = 1; index <= itemsInList; index++) {

		err = AEGetNthPtr (&docList, index, typeFSS, &keywd,
				&returnedType, (Ptr) &myFSS, sizeof(myFSS),
&actualSize);
		if (err)
			return err;

		printf ("%s\n", PtoCstr(myFSS.name));
	}

	err = AEDisposeDesc (&docList);
	return noErr;
}

pascal OSErr  MyHandleOApp (AppleEvent *theAppleEvent, AppleEvent *reply, long
							handlerRefCon)
{
	printf ("No files to print or open\n");
}

OSErr MyGotRequiredParams (AppleEvent *theAppleEvent)
{
	DescType	returnedType;
	Size		actualSize;
	OSErr		err;

	err = AEGetAttributePtr (theAppleEvent, keyMissedKeywordAttr,
					typeWildCard, &returnedType, nil, 0,
					&actualSize);
	if (err == errAEDescNotFound)	// you got all the required parameters
		return noErr;
	else if (!err)			// you missed a required parameter
		return errAEEventNotHandled;
	else				// the call to AEGetAttributePtr failed
```

```
          return err;
}


main ()
{
    Boolean             aEvents;
    short               doWhat;
    short               fileCnt;
    short               i;
    AppFile             fileStuff;
    EventRecord         theEvent;
    OSErr               err;

    InitToolbox ();
    aEvents = AppleEventsInstalled();

    if (aEvents) {

          err = AEInstallEventHandler (kCoreEventClass, kAEOpenDocuments,
                                    MyHandleODoc,0, FALSE);
          err = AEInstallEventHandler (kCoreEventClass, kAEOpenApplication,
                                    MyHandleOApp,0, FALSE);
          err = AEInstallEventHandler (kCoreEventClass, kAEPrintDocuments,
                                    MyHandlePDoc,0, FALSE);

          while (!gDone) {
                if ( WaitNextEvent ( everyEvent, &theEvent, 0, nil ) ) {
                      switch (theEvent.what) {
                      case mouseDown:
                            gDone = TRUE;
                      case kHighLevelEvent:
                            err = AEProcessAppleEvent (&theEvent);
                            break;
                      }
                }
          }
    }
    else {
          //Get number of files double-clicked on by the user
          CountAppFiles ( &doWhat, &fileCnt );
          if (fileCnt > 0) {            // if the user selected one or more files
                if (doWhat == appOpen)      {
                      printf ("Files to open:\n");
                      for (i = 1; i <= fileCnt; i++) {
                            GetAppFiles ( i, &fileStuff );
                            printf ("%s\n", PtoCstr (fileStuff.fName));
                      }
                }
                else if (doWhat == appPrint) {
                      printf ("Files to print:\n");
                      for (i = 1; i <= fileCnt; i++) {
                            GetAppFiles ( i, &fileStuff );
                            printf ("%s\n", PtoCstr (fileStuff.fName));
                      }
                }
                for (i = 0; i < fileCnt; i++)
                        ClrAppFiles(i);
                }
        else
                printf ("No files to print or open\n");
        while (!Button());
    }
}
```