Launch Application with Doc using Apple Events

```
/*
	Launch application with doc using Apple Events
	This short application demonstrates how to send an Apple event (Æ) to the
	Finder requesting it to open a document as if it had been double clicked.
	Everything about this code is System 7 dependent, so don't even bother trying
	to run it under System 6. Just add MacTraps and MacTraps2 to the project.
*/

// Assumes inclusion of <MacHeaders>
#include <AppleEvents.h>
#include <Processes.h>
#include <Aliases.h>

// Constants for dealing with FinderEvents. See Chapter 8 of the Apple Event
// Registry for more information.
#define kFinderSig		'FNDR'
#define kAEFinderEvents		'FNDR'
#define kSystemType		'MACS'

#define kAEOpenSelection		'sope'
#define keySelection		'fsel'

// Prototypes
void InitToolbox(void);
OSErr FindAProcess(OSType, OSType, ProcessSerialNumber*);
OSErr OpenSelection(FSSpecPtr theDoc);

// Given a FSSpecPtr to either an application or a document, OpenSelection creates a
// finder Open Selection Apple event for the object described by the FSSpec.
OSErr OpenSelection(FSSpecPtr theDoc)
{
	AppleEvent		aeEvent;	// the event to create;
	AEDesc		myAddressDesc;	// descriptors for the Æ
	AEDesc		aeDirDesc;
	AEDesc		listElem;
	AEDesc		fileList;	// our list
	FSSpec		dirSpec;
	AliasHandle		dirAlias;		// alias to directory with our file
	AliasHandle		fileAlias;		// alias of the file itself
	ProcessSerialNumber process;		// the finder's psn
	OSErr		myErr;		// duh

	// Get the psn of the Finder and create the target address for the Æ.
	if(FindAProcess(kFinderSig,kSystemType,&process))
		return procNotFound;
	myErr = AECreateDesc(typeProcessSerialNumber,(Ptr) &process,
				sizeof(process), &myAddressDesc);
	if(myErr)	return myErr;

	// Create an empty Æ
	myErr = AECreateAppleEvent (kAEFinderEvents, kAEOpenSelection,
			&myAddressDesc, kAutoGenerateReturnID, kAnyTransactionID,
&aeEvent);
```

```
	if(myErr)
		return myErr;

	// Make an FSSpec and alias for the parent folder, and an alias for the file
	FSMakeFSSpec(theDoc->vRefNum,theDoc->parID,nil,&dirSpec);
	NewAlias(nil,&dirSpec,&dirAlias);
	NewAlias(nil,theDoc,&fileAlias);

	// Create the file list.
	if(myErr=AECreateList(nil,0,false,&fileList))
		return myErr;

	/* Create the folder descriptor
	*/
	HLock((Handle)dirAlias);
	AECreateDesc(typeAlias, (Ptr) *dirAlias, GetHandleSize
			((Handle) dirAlias), &aeDirDesc);
	HUnlock((Handle)dirAlias);
	DisposHandle((Handle)dirAlias);

	if((myErr = AEPutParamDesc(&aeEvent,keyDirectObject,&aeDirDesc)) ==
		noErr)
	{
		AEDisposeDesc(&aeDirDesc);
		HLock((Handle)fileAlias);

		AECreateDesc(typeAlias, (Ptr)*fileAlias,
			GetHandleSize((Handle)fileAlias), &listElem);
		HUnlock((Handle)fileAlias);
		DisposHandle((Handle)fileAlias);
		myErr = AEPutDesc(&fileList,0,&listElem);
	}
	if(myErr)
		return myErr;
	AEDisposeDesc(&listElem);

	if(myErr = AEPutParamDesc(&aeEvent,keySelection,&fileList))
		return myErr;

	myErr = AEDisposeDesc(&fileList);

	myErr = AESend(&aeEvent, nil,
		kAENoReply+kAEAlwaysInteract+kAECanSwitchLayer,
		kAENormalPriority, kAEDefaultTimeout, nil, nil);
	AEDisposeDesc(&aeEvent);
}

// Search through the current process list to find the given application. See
// Using the Process Manager for a similar way of doing this.
OSErr FindAProcess(OSType typeToFind, OSType creatorToFind,
		ProcessSerialNumberPtr processSN)
{
	ProcessInfoRec		tempInfo;
	FSSpec		procSpec;
	Str31			processName;
	OSErr			myErr = noErr;
	// start at the beginning of the process list
	processSN->lowLongOfPSN = kNoProcess;
	processSN->highLongOfPSN = kNoProcess;

	// initialize the process information record
	tempInfo.processInfoLength = sizeof(ProcessInfoRec);
	tempInfo.processName = (StringPtr)&processName;
	tempInfo.processAppSpec = &procSpec;

	while((tempInfo.processSignature != creatorToFind ||
		tempInfo.processType != typeToFind) ||
		myErr != noErr)
	{
		myErr = GetNextProcess(processSN);
		if (myErr == noErr)
			GetProcessInformation(processSN, &tempInfo);
	}
	return(myErr);
}

main()
{
	StandardFileReply	mySFR;
	SFTypeList		myTypeList;

	// Initialize the toolbox
	InitGraf((Ptr) &(thePort));
	InitFonts();
	InitWindows();
	InitMenus();
	TEInit();
	InitDialogs(nil);

	// Get a file to open...
	myTypeList[0] = 'TEXT';
	StandardGetFile(nil,1,myTypeList,&mySFR);

	// ...and open it.
	OpenSelection(&(mySFR.sfFile));
}
```