

# Course Overview

This course is about “Scripting” — how another application can control your application reliably to automate complex or repetitive tasks. In the past, the only way to control a Macintosh application was through its user interface, either by pointing and clicking or by pressing keys on the keyboard. If one application wanted to control another, it had to emulate these actions, often by posting mouse-down and key-down events. Since the controlling application cannot see the screen, and cannot take advantage of the visual feedback provided there, such an emulation can be quite unreliable.

Scripting is much more reliable, but it requires the application’s cooperation. The application must accept commands from the outside world. These commands should be able to do anything the user can do, and allow the controlling application to get information about the controlled application’s internal state. For example, an outside application might want to “clean up” a window full of icons in the Finder. The application could send a “clean up” command to the Finder, but what if the user wanted a different organization for the icons? In that case, the controlling application could ask the Finder for the location of each icon in the window and then send individual commands to place each icon. In other words, the controlling application would extract some information about the Finder’s internal state (i.e., the number and location of each of the icons) and would send some events to change that state.

All of this control goes through the application’s Scripting interface instead of its User interface. Just as Macintosh applications need to present a well-designed User interface, they also must provide an organized and well-designed Scripting interface. This alternate interface must be powerful and flexible, yet easy for the scripting user to understand.

Apple has created Human Interface guidelines that help developers follow the model set by the rest of the Macintosh graphical interface, and Apple has created an “Apple Events Object Model” to help developers fit their applications into Apple’s scripting model. The Apple Events Object Model is the framework around which you build your application’s scripting interface.

This course will help you add Apple Events Object Model support to your application. We begin with the Object Support Library — a library which Apple provides to help process Object Model Apple events. The OSL modules explain the structure of Object Model Apple events and show you how to parse “Object Specifiers.” (Object specifiers are a key component of the Object Model.)

Once your application can receive Object Model Apple events, and extract and process the Object Specifiers contained within, you can add support for the standard “Core” Apple events. The Apple Event Registry documents these events, and the Apple Event Registry is the subject (and title) of the second part of this course. This section also discusses the major differences between the Apple Events Object Model’s

use of the word “object” and traditional Object-Oriented Programming’s use of the same term. You will implement 4 of the Core events — Create Element, Delete, Get Data, and Set Data — by the end of this module.

Adding the Core events makes your application “Scriptable.” If you want to add extra functionality, including script recording, you need to “factor” your application. We discuss this topic, and some future directions for Macintosh scripting, in the section entitled “Application Architecture.”

Once you have completed these three sections, you will have an application that is scriptable and recordable. The Developer University classroom course goes beyond this point, into “script embedding” and using scripts to implement user-modifiable functionality in your application.

Scripting — and AppleScript — is an important extension to the Macintosh interface. By taking this course, and implementing the Apple Events Object Model, you’ll be ready for the future of Macintosh computing.

The following article, taken from develop magazine, Issue 10 (© 1992 by Apple Computer, Inc.) provides a further introduction to the Apple Events Object Model, and a preview of the “Object-first” implementation strategy used in this course.