Mac OS 8 OpenDoc Components

This paper describes guidelines for developing OpenDoc components on Mac OS 8. It covers information which is specific to OpenDoc components, including part editors, container applications, and shell plug-ins. In general, the guidelines for Mac OS 8 applications also apply to OpenDoc components. The application guidelines are described in three associated documents: **Mac OS 8 Compatible Application**, **Mac OS 8 Transitional Application**, and **Mac OS 8 Only Application**. This paper assumes familiarity with these documents and will point out exceptions and provide supplemental information specific to OpenDoc components.

1.0 Mac OS 8 Compatible Part Editor

This section defines and describes guidelines for developing a Mac OS 8 Compatible Part Editor.

1.1 Definition

A Mac OS 8 Compatible Part Editor is an OpenDoc part editor (or viewer) that can run equally well on System 7.x and Mac OS 8. It only uses features documented in Inside Macintosh and OpenDoc documentation. This type of part editor is nothing more than an editor written for System 7.x in such a way that it is compatible with Mac OS 8.

1.2 Guidelines

1. All of the guidelines described for a Mac OS 8 Compatible Application also apply to a Compatible Part Editor (see **Mac OS 8 Compatible Application**, Section 1.1). The guidelines that follow apply specifically to Compatible Part Editors.
2. Don't patch session-level objects. Session-level objects (ODDispatcher, ODArbitrator, etc.) can only be patched by container applications (described later). This guideline also applies to parts running on System 7.x, but deserves special emphasis for Mac OS 8 due to compatibility issues. Note that the **OpenDoc Programmer's Guide** states that the

root part of a document may install a patch. This is a documentation error.

3. Don't assume the document process has an event loop. Calling `WaitNextEvent()` is still supported (when a part has the modal focus), but it is discouraged because doing so introduces extra event translation overhead when a part is running in an Apple Event-based document process (see **Mac OS 8 Only Application**, Section 1.1.1). When a part editor calls `WaitNextEvent()` followed by `ODDispatcher::Dispatch()` the event is translated from an AppleEvent to an EventRecord and back again before being handled by the target.
4. Use the OpenDoc memory API. By allocating memory through OpenDoc's memory manager, a Compatible Part Editor both ensures compatibility with and enjoys the benefits of Mac OS 8 memory management.
5. In rare circumstances, a Compatible Part Editor may be unable to avoid calling the traditional System 7.x Memory API directly because System 7.x requires that certain allocations be located in the application heap. In this situation, a Compatible Part Editor running on Mac OS 8 is subject to the guidelines and limitations of the Mac OS 8 Transitional Memory API. OpenDoc for Mac OS 8 uses the Transitional Memory API to take advantage of its improved performance and heap management. For a detailed description of the Transitional Memory API guidelines, see **Mac OS 8 Transitional Application**, Section 1.3.5.

1.3 How to Build a Mac OS 8 Compatible Part Editor

A Mac OS 8 Compatible Part Editor is built using the System 7.x Universal Interfaces and the OpenDoc 1.0.x interfaces that are available today. Compile and link your part editor just as you would for System 7.x. Refer to the **OpenDoc Programmer's Guide** for further details.

TEMPORARY SOLUTION FOR <u>Mac OS 8 Developers Release</u>: *<u>Compatibility Edition:</u>*

To run your part editor on Mac OS 8, first install it on a System 7.x system and create a stationery file for the editor. Then, after installing Mac OS 8, copy the editor shared library into the Editors folder inside the Mac OS 8 Folder. Copy your stationery into the Stationery folder, and open it just as you would on System 7.x. Your part editor will become the root part of an untitled OpenDoc document.

2.0 Mac OS 8 Transitional Part Editor

This section defines and describes guidelines for developing a Mac OS 8 Transitional Part Editor.

2.1 Definition

A Mac OS 8 Transitional Part Editor is an OpenDoc part editor (or viewer) that adopts selected Mac OS 8 technologies but continues to use APIs that are deprecated in Mac OS 8. Like a Transitional Application, a Transitional Part Editor may be either binary compatible with System 7.x or source compatible with System 7.x.

Refer to **Mac OS 8 Transitional Application**, Section 1.0 for definitions of binary compatible and source compatible, as well as techniques to use for dynamic feature determination.

If you are not concerned with binary or source compatibility with System 7.x, or you wish to adopt technologies which are not available to Transitional Part Editors, see Section 3, which describes a Mac OS 8 Only Part Editor.

2.2 Guidelines

1. All of the guidelines described for a Mac OS 8 Transitional Application also apply to a Transitional Part Editor (see **Mac OS 8 Transitional Application**).
2. All of the OpenDoc-specific guidelines described for a Mac OS 8 Compatible Part Editor also apply to a Transitional Part Editor (see Section 1.2).
3. A Transitional Part Editor may adopt only those technologies available to a Transitional Application, as described in **Mac OS 8 Transitional Application**, Section 1.3. OpenDoc-specific issues related to technology adoption are described in the next section.

2.3 Transitional Technologies

This section describes transitional technology issues specific to OpenDoc.

Tasking & Synchronization

The `WakeupProcess()`communication technique described for Transitional Applications is not appropriate for Transitional Part Editors because part editors do not normally receive null events. Although null events are sent periodically when requested by a part editor, use of this technique could impair system performance due to excessive polling.

The recommended technique for communication between tasks and Transitional Part Editors is to use Apple Events. This technique requires that a Transitional Part Editor have a Semantic Interface available to handle Apple Events (i.e. the part editor must minimally implement scripting).

After a Transitional Part Editor creates a task, it should send an Apple Event to the task with a reply event that is addressed to the part instance that created the task. This will provide the task with the event addressing information it needs to communicate with the part editor. For details on how to address Apple Events to part instances, refer to chapter 9, "Semantic Events and Scripting", of the **OpenDoc Programmer's Guide**.

Memory

A Transitional Part Editor should generally use the OpenDoc memory API rather than the Transitional Memory API for all memory allocation. On Mac OS 8, both APIs end up calling the same underlying implementation, but using the OpenDoc memory API helps ensure future compatibility. The only exception to this rule is when a part editor needs to allocate/deallocate memory that is deallocated/allocated by non-OpenDoc code that does not use the OpenDoc memory API. In this case, the part editor has no choice but to use the Transitional Memory API directly.

Since part editors do not have SIZE resources, a Transitional Part Editor does not have to do anything to "activate" the Transitional Memory API. OpenDoc on Mac OS 8 assumes that all part editors adhere to the Transitional Memory API guidelines described in **Mac OS 8 Transitional Application**, Section 1.3.5.

2.4 How to Build a Mac OS 8 Transitional Part Editor

A Mac OS 8 Only Part Editor requires both the Mac OS 8 and the OpenDoc for Mac OS 8 interfaces and libraries. It will not be possible to build a Mac OS 8 Transitional Part Editor using the interfaces on the **Mac OS 8 Developer Release: *Compatibility Edition*** CD because the Mac OS 8 version of the OpenDoc interfaces and libraries are not yet available for release. Subsequent developer releases of Mac OS 8 will contain all of the tools and documentation necessary for building a Mac OS 8 Transitional Part Editor.

3.0 Mac OS 8 Only Part Editor

This section defines and describes guidelines for developing a Mac OS 8 Only Part Editor.

3.1 Definition

A Mac OS 8 Only Part Editor adopts key Mac OS 8 technologies and uses only preferred Mac OS 8 APIs. As the name suggests, an 8-Only Part Editor will not run on System 7.x. If you are concerned with System 7.x compatibility, refer to earlier sections on Mac OS 8 Compatible Part Editors and Mac OS 8 Transitional Part Editors.

3.2 Guidelines

1. Don’t call any APIs that are deprecated in Mac OS 8.
2. Use Apple Events as the pervasive event mechanism. On Mac OS 8, the EventRecord structure and the ODEventData structure (which mirrors the EventRecord) are deprecated. OpenDoc provides new methods that process ODAppleEvents instead (see below for more details).
3. Use HIObjects for all windows, menus, controls, lists, and dialogs. In cases where the OpenDoc API exposes the deprecated data types (such as WindowPtr or MenuHandle), OpenDoc provides new methods that process HIObjects instead (see below for more details).
4. Use Quickdraw GX for imaging and printing. Quickdraw GX is always available on Mac OS 8, and OpenDoc’s imaging model is derived from Quickdraw GX, so the APIs work well together. This eases development and positions a part editor to take advantage of Quickdraw GX’s resolution-independent imaging model.

3.3 Events

OpenDoc on Mac OS 8 uses Apple Events (encapsulated by the ODAppleEvent class) to deliver all user and system events to Mac OS 8 Only Part Editors. It does this through a new method, `ODPart::HandleAppleEvent()` which replaces `ODPart::HandleEvent()` It is up to the part developer to implement the `HandleAppleEvent()`method, but the recommended technique is to create an Apple Event dispatcher (AEDispatcher) for each part instance and set up one or more handler tables (AEHandlerTables) populated with Apple Event handlers for each event the part editor wants to receive. When an event is delivered to the part instance via the `HandleAppleEvent()`

method, the part editor simply calls `AESendEventToSelf()`with a reference to the part instance's Apple Event dispatcher. In effect, this model mimics at the part editor level what happens in a Mac OS 8 Only Application. The main difference is that a part editor never calls `AEReceive()`. Instead, OpenDoc calls `AEReceive()` and forwards events on to document parts just as it does today on System 7.x. For more details about Apple Event dispatchers and handler tables, refer to **Mac OS 8 Only Application**, Section 1.1.1.

3.4 HIObjects

OpenDoc allows Mac OS 8 Only Part Editors to embed HIObjects (HIPanels, specifically) inside document parts. In fact, the HIObject class library completely replaces the System 7.x Window, Control, Dialog, Menu and List Managers. In addition, the OpenDoc API provides a number of new methods as needed to support HIObjects.

A part editor embeds HIPanels in a manner very similar to how an application would embed HIPanels in a window. Refer to **Mac OS 8 Only Application**, Section 1.1.2 for a description of how applications use HIPanels. The key difference is in how panels are installed into the panel hierarchy. In an application, each window has a root panel into which all other panels are placed. In OpenDoc, each display facet has a root panel, and a part editor places panels inside the facet's root panel instead of the window's root panel. Once a panel is properly installed in a facet's panel hierarchy, events get automatically routed to panels just as they would from a window's root panel, and all other aspects of a panel's behavior are as documented.

3.5 How to Build a Mac OS 8 Only Part Editor

A Mac OS 8 Only Part Editor requires both the Mac OS 8 and the OpenDoc for Mac OS 8 interfaces and libraries. It will not be possible to build an 8-Only Part Editor using the interfaces on the **Mac OS 8 Developer Release: *Compatibility Edition*** CD because the Mac OS 8 version of the OpenDoc interfaces and libraries are not yet available for release. Subsequent developer releases of Mac OS 8 will contain all of the tools and documentation necessary for building a Mac OS 8 Only Part Editor.

4.0 Container Application

This section defines and describes guidelines for developing an OpenDoc container application on Mac OS 8.

4.1 Definition

Container applications are applications which handle the job of being an OpenDoc shell process as well as managing their own internal content. Any of the three Mac OS 8 application types (Compatible, Transitional, and 8-Only) may be container applications.

A new library will be provided to facilitate container application development, the OpenDoc Internet Adaptor (ODIA). As the name suggests, ODIA is targeted at applications that want to enable embedding of Cyberdog components. ODIA is designed to make it as simple as possible to retrofit existing applications with OpenDoc container capability.

4.2 Guidelines

Mac OS 8 Compatible Application: All guidelines for Mac OS 8 Compatible Applications still apply. Container support can be provided via ODIA or through direct access to the OpenDoc API.

Mac OS 8 Transitional Application: All guidelines for Mac OS 8 Transitional Applications still apply. Container support can be provided via ODIA or through direct access to the OpenDoc API.

Mac OS 8 Only Application: All guidelines for Mac OS 8 Only Applications still apply. Currently, container support can only be provided through direct access to the OpenDoc API.

5.0 Shell Plug-in

The guidelines for developing an OpenDoc Shell Plug-in are unchanged for Mac OS 8. Refer to chapter 10 of the **OpenDoc Programmer's Guide** for more information about Shell Plug-ins.