This chapter describes the Package Manager, the part of the system software that loads packages into memory. The packages include one for presenting the standard user interface when a file is to be saved or opened and others for doing more specialized operations such as floating-point arithmetic.

Read the information in this chapter to get a complete list of all packages and to get a description of the Package Manager routines that load the packages into memory.

Ordinarily, you do not need to use the Package Manager routines described in this chapter. The Operating System itself is responsible for installing the packages when an application is launched. While your application probably won't ever need to use these routines, for the sake of completeness they are described in this chapter.

About the Package Manager

The Package Manager lets you load packages into memory. A ***package*** is a set of routines and data types that is stored as a resource of type 'PACK'. In early models of the Macintosh computer, all packages were disk-based and brought into memory only when needed; some packages are now in ROM. The System file contains the standard Macintosh packages and the resources they use or own. Table 10-1 lists the standard Macintosh packages.

Table 10-1 The standard Macintosh packages

Package	Description	Resource ID
List Manager	Provides routines that your application can use to create scrollable lists that allow the user to select one or more of a group of items.	0
Disk Initialization Manager	Provides routines that initialize and name new floppy disks. This package is called by the Standard File Package and applications.	2
Standard File Package	Provides routines that your application can use to display dialog boxes that let the user specify the locations of files to be saved or opened.	3
Floating-Point Arithmetic Package	Provides routines that support extended-precision arithmetic according to IEEE Standard 754.	4

continued

Table 10-1 The standard Macintosh packages (continued)

Package	Description	Resource ID
Transcendental Functions Package	Provides routines that support trigonometric, logarithmic, exponential, and financial functions, and a random number generator.	5
Text Utilities (formerly referred to as the International Utilities Package)	Provides routines that your application can use to specify strings for various purposes, to format numbers and currency, format date and time, search and replace text, and more.	6
Text Utilities (formerly referred to as the Binary-Decimal Conversion Package)	Provides routines that your application can use to specify strings for various purposes, to format numbers and currency, format date and time, search and replace text, and more.	7
Apple Event Manager	Provides routines that your application can use to respond, send, and record Apple events.	8
PPC Browser	Provides routines that your application can use to display the program linking dialog box, which allows a user to select a port to communicate with.	9
Edition Manager	Provides routines that your application can use to allow users to share and automatically update data and numerous documents and applications.	11
Color Picker	Provides routines that your application can use to display a standard dialog box for choosing a color, and converts color specifications from one color model to another.	12

Table 10-1 The standard Macintosh packages (continued)

Package	Description	Resource ID
Data Access Manager	Provides routines that your application can use to gain access to data in another application, and provides templates to be used for data transactions.	13
Help Manager	Provides routines that your application can use to provide Balloon Help online assistance.	14
Picture Utilities	Provides routines that obtain qualitative and quantitative information about pictures and pixel maps.	15

If the Package Manager is not able to load a package, the Package Manager adds the resource ID number of the affected package to 17 to get an error number. The System Error Handler uses this error number to display an error message. Originally this approach worked because there were only 7 packages, and the error number would fall between 17 and 24, which are the error numbers that define the "Can't load package" error. However, now there are more packages and the resulting error messages from packages with resource IDs greater than 7 are misleading.

The error messages that corresponds to packages with resource IDs greater than 7 are as follows:

Resource ID	Package	Error ID	Error
9	Apple Event Manager	25	Out of memory
9	PPC Toolbox	26	Can't launch file
11	Edition Manager	28	Stack overflow
12	Color Picker	29	*
13	Data Access Manager	30	Disk insertion required
14	Help Manager	31	Wrong disk inserted
15	Picture Utilities	32	*

* There is not a defined system error for this error ID.

The system errors are described in detail in the chapter "System Error Handler" in this book.

Using the Package Manager

The Package Manager provides two routines: the `InitPack` procedure and the `InitAllPacks` procedure. The `InitPack` procedure loads one specified package into memory. To specify which package to load, you pass, as a parameter to the `InitPack` procedure, the package's resource ID. You can use the `InitAllPacks` procedure to load all packages into memory. Typically, you do not need to use either of these two procedures because the `InitAllPacks` procedure is automatically called when your application is launched.

The `InitPack` and `InitAllPacks` procedures do not initialize the packages. Consult the description of the specific package to see if it needs to be initialized before your application can utilize all of its routines. For example, to use the Data Access Manager routines, your application must first call the `InitDBPack` function (an initialization routine provided by the Data Access Manager). If a package needs to be initialized, it provides an initialization routine.

Note
You can access a routine in a package through a trap macro and a routine selector. The name of the trap macro includes the word "`Pack`" and the resource ID of the specific package. For example, the trap macro for the routines in the Edition Manager is `_Pack11`. Most system software routines that are accessed through a trap macro and a routine selector also have a corresponding macro that expands to call the original trap macro and automatically puts the correct routine selector on the stack. For example, to access the Standard File Package routine `StandardGetFile`, you can call the `_StandardGetFile` macro. The `_StandardGetFile` macro then expands to call the `_Pack3` trap macro and places the routine selector on the stack (in this example the routine selector is $0006). See the chapter "Trap Manager" in this book for more information about trap macros and routine selectors. ◆

Package Manager Reference

This section describes routines that are specific to the Package Manager.

Routines

This section describes the two routines in the Package Manager. One routine lets you load a specified package into memory, and one routine lets you load all packages into memory.

Initialization of Packages

You use the routines in this section to load one specified package or all packages into memory.

InitPack

You can use the `InitPack` procedure to load a specified package into memory.

```
PROCEDURE  InitPack (packID: Integer);
```

`packID` A package resource ID.

DESCRIPTION

The `InitPack` procedure loads the package specified by the `packID` parameter into memory. The `packID` parameter is the package's resource ID. To initialize a specific package or manager, consult the documentation of the specific package or manager.

InitAllPacks

You can use the `InitAllPacks` procedure to load all packages into memory.

```
PROCEDURE InitAllPack;
```

DESCRIPTION

The `InitAllPacks` procedure loads all the packages into memory. The `InitAllPacks` procedure is automatically called when your application is launched.

Summary of the Package Manager

Pascal Summary

Constants

```
CONST
   listMgr     = 0;     {List Manager}
   dskInit     = 2;     {Disk Initialization Manager}
   stdFile     = 3;     {Standard File Package}
   flPoint     = 4;     {Floating-Point Arithmetic Package}
   trFunc      = 5;     {Transcendental Functions Package}
   textUtil1   = 6;     {Text Utilities}
   textUtil2   = 7;     {Text Utilities}
   aevtMgr     = 8;     {Apple Event Manager}
   ppcBrowser  = 9;     {PPC Browser}
   editionMgr  = 11;    {Edition Manager}
   colorPicker = 12;    {Color Picker}
   dataAccess  = 13;    {Data Access Manager}
   helpMgr     = 14;    {Help Manager}
   pictUtil    = 15;    {Picture Utilities}
   intUtil     = 6;     {Text Utilities}
   bdConv      = 7;     {Text Utilities}
```

Routines

Initializing Packages

```
PROCEDURE InitPack (packID: Integer);
PROCEDURE InitAllPacks;
```

C Summary

Constants

```
enum {
   listMgr     = 0,     /*List Manager*/
   dskInit     = 2,     /*Disk Initialization Manager*/
   stdFile     = 3,     /*Standard File Package*/
   flPoint     = 4,     /*Floating-Point Arithmetic Package*/
   trFunc      = 5,     /*Transcendental Functions Package*/
   textUtil1   = 6,     /*Text Utilities*/
   textUtil2   = 7,     /*Text Utilities*/
   aevtMgr     = 8,     /*Apple Event Manager*/
   ppcBrowser  = 9,     /*PPC Browser*/
   editionMgr  = 11,    /*Edition Manager*/
   colorPicker = 12,    /*Color Picker*/
   dataAccess  = 13,    /*Data Access Manager*/
   helpMgr     = 14,    /*Help Manager*/
   pictUtil    = 15,    /*Picture Utilities*/
   intUtil     = 6,     /*Text Utilities*/
   bdConv      = 7      /*Text Utilities*/
};
```

Routines

Initializing Packages

```
pascal void InitPack        (short packID);
pascal void InitAllPacks    (void);
```

Assembly-Language Summary

Trap Macros

Trap Macros Requiring Routine Selectors

```
_Pack0      ;List Manager
_Pack2      ;Disk Initialization Manager
_Pack3      ;Standard File Package
_Pack6      ;Text Utilities
_Pack7      ;Text Utilities
_Pack8      ;Apple Event Manager
_Pack9      ;PPC Browser
_Pack11     ;Edition Manager
_Pack12     ;Color Picker
_Pack13     ;Data Access Manager
_Pack14     ;Help Manager
_Pack15     ;Picture Utilities
```