Shutdown Manager

This chapter describes the Shutdown Manager, the part of the Operating System that manages the final stages of shutting down or restarting a Macintosh computer. The Shutdown Manager allows you to install a custom procedure that is executed during the process of shutting down or restarting. You can also use the Shutdown Manager to restart or shut down the computer directly, although this practice is strongly discouraged.

▲ **WARNING**
For reasons described later, you should avoid shutting down or restarting the computer directly except in an emergency (for instance, when data on the disk might be destroyed). If you need to restart or shut down the system, send a Shutdown or Restart event to the Finder, as described in "Sending a Shutdown or Restart Event" on page 8-7. ▲

Read the information in this chapter if your application or other software component needs to intervene in the standard process of shutting down or restarting the computer. In general, applications do not need to intervene in this process. You are likely to use the Shutdown Manager only if you are designing a device driver or system extension requiring notification that the computer is about to be shut down or restarted.

If you want to install a custom shutdown procedure, you should know how to install a code segment into the system heap, as described in the chapter "Memory Manager" in *Inside Macintosh: Memory*. If you want to shut down or restart the computer and need to familiarize yourself with the process of sending Apple events, see the chapter "Apple Event Manager" in *Inside Macintosh: Interapplication Communication*.

This chapter begins with a description of the Shutdown Manager and of the typical shutdown or restart process. Then it describes how you can

- use Apple events to request that the system be shut down or restarted
- install a custom shutdown procedure to be executed during the shutdown or restart process
- remove a shutdown procedure that you have previously installed

About the Shutdown Manager

The Shutdown Manager gives applications and other software a chance to perform any necessary shutdown processing before the computer is turned off or restarted. It also shuts down or restarts Macintosh computers, providing a consistent human interface for shutting down and restarting different models. Before restarting a computer or turning off the power, the Shutdown Manager checks for open device drivers and desk accessories and allows them to perform any necessary housekeeping.

The Shutdown Manager does not notify open applications that they are about to be shut down. This notification is handled by the Process Manager, as explained in "Using the Shutdown Manager" beginning on page 8-7. The Shutdown Manager provides two procedures that allow you to shut down or restart a Macintosh computer. However, these procedures do not perform the preliminary tasks that the Finder initiates when a user chooses Shut Down or Restart from the Finder's Special menu. Accordingly, your application should not call these procedures directly because all open applications will terminate abruptly without the opportunity to save their current states and exit gracefully. Instead, your application should send a Shutdown or Restart event to the Finder, as described in "Sending a Shutdown or Restart Event" on page 8-7.

The main function of the Shutdown Manager is to execute a custom **shutdown procedure** that lets your application perform some additional tasks before the computer shuts down. The types of software most likely to install shutdown procedures are device drivers and system extensions. For example, drivers of early hard disk drives that use stepper motors usually need to park the drive heads in a safe zone before the power is turned off. A shutdown procedure could notify a driver to park the head. In another case, a user could install a system extension that displays an alert box asking whether to back up the hard disk before the computer shuts down.

The Shutdown Process

When a user chooses Shut Down or Restart from the Finder's Special menu, the tasks performed to shut down or restart a Macintosh computer differ in one important respect from those performed when you call the Shutdown Manager directly. In the former case, the Finder receives notification of a Shutdown or Restart event and calls the Process Manager to notify any open applications to quit. Only after the applications return does the Finder call the appropriate Shutdown Manager procedures to shut down or restart the system. To have your driver or application initiate this process, you can send a Shutdown or Restart event to the Finder, as described in "Sending a Shutdown or Restart Event" on page 8-7.

The Shutdown Manager procedures for shutting down or restarting the system (either `ShutDwnPower` or `ShutDwnStart`) perform an identical five-step process:

1. Checking for and executing custom shutdown procedures installed by calls to `ShutDwnInstall`. (This step occurs three times during the shutdown process.)
2. Checking the Device Manager's unit table to determine whether any drivers or desk accessories are open and, if so, notifying them of the impending shutdown or restart.
3. Saving the desk scrap, if any.
4. Unmounting mounted volumes.
5. Turning off the computer.

This section describes the shutdown process in detail, beginning with the preliminary step, mediated by the Finder, of closing applications.

Closing Open Applications

When a user or application notifies the Finder to shut down or restart the computer, the Finder calls the Process Manager. The Process Manager performs the important task of notifying all open applications to quit. It checks its list of open applications and sends a Quit Application event to those applications that can process Apple events. For applications that cannot process Apple events, the Process Manager sends a mouse-down event indicating that Quit was chosen from the File menu. This technique works for applications that display Quit in the File menu. Applications that display Quit in a different menu or that display a different form of Quit (such as Quit Document or Quit…) must specify a resource of type `'mstr'` or `'mst#'` with a resource ID of 100 or 101, respectively. The Process Manager reads these resources to locate the menu containing the Quit item or to find the exact Quit string to send to the application.

Once notified, open applications have the opportunity to save data and execute other exit procedures before they quit. Note that `ShutDwnPower` and `ShutDwnStart` do not notify open applications to quit. For this reason, you should not call these routines directly.

Checking for Custom Shutdown Procedures

After all open applications have quit, the Finder calls either `ShutDwnPower` or `ShutDwnStart`, respectively, depending on whether the user chose Shut Down or Restart from the Finder's Special menu. Because these two procedures perform the same set of tasks, the ensuing explanation applies to both routines.

The `ShutDwnPower` routine first checks for custom shutdown procedures installed by calls to `ShutDwnInstall`. The Shutdown Manager maintains a queue that contains the address of each custom procedure and a constant indicating when during the shutdown process to execute each procedure. The `ShutDwnPower` routine reads this queue three times during the shutdown process: before notifying drivers to shut down, before unmounting volumes, and before turning off the power. (See the description of `ShutDwnInstall` on page 8-13 for an explanation of the shutdown constants.) At this point, `ShutDwnPower` executes any custom procedures that specify the `sdOnDrivers` constant. Then it begins the next step of the shutdown process: checking for open device drivers.

Checking for Open Device Drivers

After locating any custom shutdown procedures, `ShutDwnPower` checks the Device Manager's unit table to determine whether any device drivers or desk accessories are open. It also inspects the `dNeedGoodBye` bit in the `drvrFlags` word for each driver. This bit, if set, indicates that the driver requests notification when the application heap is reinitialized or the system shuts down. Accordingly, `ShutDwnPower` calls the driver's `Control` function with the `csCode` field set to –1 (the `goodBye` global constant). This notification of impending termination is called a **good-bye message.**

A driver in an application heap also receives a good-bye message every time an application quits. For this reason, the driver cannot always determine whether a good-bye message means that the system is about to shut down. If making this distinction is important, you can call `ShutDwnInstall` to install a simple procedure that informs your driver when the computer is about to shut down. For more information about the `drvrFlags` word and the `Control` function, see the chapter "Device Manager" in *Inside Macintosh: Devices.*

The `ShutDwnPower` procedure does not actually close the drivers. They stay open until the power is switched off.

Saving the Desk Scrap

Having sent a good-bye message to any open driver that requested one, `ShutDwnPower` next calls the Scrap Manager function `UnloadScrap` to write the desk scrap, if any, from the Clipboard to the disk. Later, when the user restarts the computer, your application can retrieve the desk scrap by calling the `LoadScrap` function, as described in the chapter "Scrap Manager" of *Inside Macintosh: More Macintosh Toolbox*.

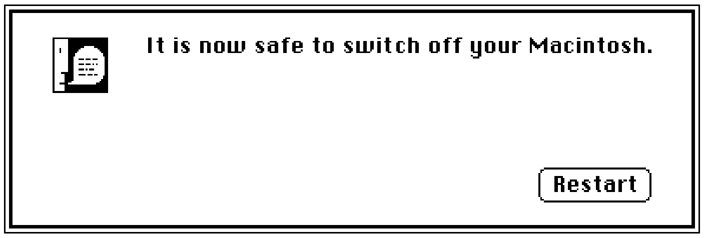
Unmounting Volumes

After saving the desk scrap, the `ShutDwnPower` procedure reads the Shutdown Manager's queue and executes any shutdown procedures that specify the `sdOnUnmount` constant. Next, `ShutDwnPower` searches the volume control block queue for mounted volumes. It unmounts each one by calling the File Manager functions `Eject` and `UnmountVol`. The `ShutDwnPower` procedure then reads the Shutdown Manager's queue and executes any shutdown procedures that specify the `sdOnRestart` constant, the `sdOnPowerOff` constant, or both.

Turning Off the Computer

Currently, there are two methods of turning off the various Macintosh models: one is software-controlled; the other, manual. With the software-controlled method, the Shutdown Manager actually turns off the power. With the manual method, by contrast, the Shutdown Manager darkens the screen and displays an alert box (Figure 8-1) stating that it is safe to turn off the computer.

Figure 8-1 A shutdown alert box


It is now safe to switch off your Macintosh.

Currently, the product lines that employ the software-controlled method are the Macintosh II models, the Macintosh Quadra models, the Macintosh Portable computers, and the PowerBook computers. Those that employ the manual method are the Macintosh LC computers, the Macintosh SE computers, the Macintosh Classic computers, the Macintosh Plus models, and all earlier models.

All Macintosh models restart the same way when a user chooses Restart from the Special menu or when the Finder or other software calls the `ShutDwnStart` procedure. Remember not to call `ShutDwnPower` and `ShutDwnStart` directly because these procedures abruptly terminate other applications that are currently running, possibly resulting in a loss of data.

Using the Shutdown Manager

The Shutdown Manager provides four procedures. The procedures `ShutDwnPower` and `ShutDwnStart` perform the same set of shutdown tasks, except that `ShutDwnPower` turns off a Macintosh computer, whereas `ShutDwnStart` restarts it. The `ShutDwnInstall` routine installs a custom shutdown procedure to perform a certain task before the computer shuts down or restarts. The `ShutDwnPower` or `ShutDwnRestart` routine calls your shutdown procedure at a predetermined point during the shutdown or restart process. The last procedure, `ShutDwnRemove`, removes custom shutdown procedures installed by `ShutDwnInstall`.

▲ **WARNING**
Usually, only the Finder or other system software should call `ShutDwnPower` and `ShutDwnInstall`. An application calling these procedures will cause other open applications to terminate abruptly, potentially destroying their data. ▲

Sending a Shutdown or Restart Event

Applications that support high-level events can send a Shutdown or Restart event to the Finder to request the system to shut down or restart. Once notified, the Finder calls the Process Manager, which gives open applications the opportunity to exit gracefully before the computer shuts down or restarts. The Process Manager checks its list of open applications and sends a Quit Application event to applications that can process Apple events. For applications that can't, the Process Manager sends a mouse-down event indicating that Quit was chosen from the File menu. Applications that display the Quit item in a different menu or that use a different wording must specify a resource of type `'mstr'` or `'mst#'` with a resource ID of 100 or 101, respectively. Once notified, the open applications then have time to perform cleanup operations (such as displaying a Save Changes alert box) before quitting.

The Shutdown and Restart events have the event class defined by the `kAEFinderEvents` constant.

```
CONST
   kAEFinderEvents = 'FNDR';     {event class for Finder}
```

The Restart event has the event ID defined by the `kAERestart` constant, and the Shutdown event has the event ID defined by the `kAEShutDown` constant:

```
CONST
   kAERestart = 'rest';    {event ID for Restart event}
   kAEShutDown = 'shut';   {event ID for Shutdown event}
```

Listing 8-1 defines a function that sends a Shutdown event to the Finder.

Listing 8-1 Sending a Shutdown event

```
FUNCTION ShutDownSafely: OSErr;
CONST
   kFinderSig = 'FNDR';
VAR
   myErr:         OSErr;
   finderAddr:    AEDesc;
   myShutDown:    AppleEvent;
   nilReply:      AppleEvent;
BEGIN
   myErr := AECreateDesc(typeApplSignature, kFinderSig,
                        SizeOf(OSType), finderAddr);
   IF myErr = noErr THEN
      myErr := AECreateAppleEvent(kAEFinderEvents, kAEShutDown,
                              finderAddr, kAutoGenerateReturnID,
                              kAnyTransactionID, myShutDown);
   IF myErr = noErr THEN
      myErr := AESend(myShutDown, nilReply, kAENoReply +
                  kAECanSwitchLayer + kAEAlwaysInteract,
                  kAENormalPriority, kAEDefaultTimeout, NIL, NIL);
   ShutDownSafely := myErr;
END;
```

To send a Shutdown or Restart event, you must call three Apple Event Manager functions. First, use the `AECreateDesc` function to create an address descriptor record that specifies the address of the Finder. You can specify the address of the Finder by its signature, `'FNDR'`. Next, call `AECreateAppleEvent` to create the Apple event you want to send. Finally, call the `AESend` function. Use the Apple event returned in the `myShutDown` variable of the `AECreateAppleEvent` function as the Apple event to send in `AESend`.

After sending the event, remember to dispose of the descriptor record and Apple event at some point, by calling the `AEDisposeDesc` function. For complete details about this function and the ones used in Listing 8-1, see the chapter "Apple Event Manager" in *Inside Macintosh: Interapplication Communication*.

Note
Applications running under system software version 6.0.x cannot send Apple events to MultiFinder because it cannot process them. As a result, your application cannot request that open applications be notified to quit before it calls `ShutDwnPower` and `ShutDwnStart`. Therefore, you should avoid calling these procedures unless absolutely necessary. ◆

Installing a Custom Shutdown Procedure

If you write a shutdown procedure, you can install a pointer to it in the Shutdown Manager's queue by calling the `ShutDwnInstall` procedure. You're most likely to need to use a custom shutdown procedure if you are writing a device driver or a system extension. For example, drivers for early hard disk drives that use stepper motors usually need to park the drive heads in a safe zone before the power is turned off. Similarly, drivers for floppy disks and CD-ROM discs use a shutdown procedure that ejects the disks so that they don't remain in the drives when the computer shuts down.

If you are developing an application, you can also make use of shutdown procedures. For example, a remote backup application might install a shutdown procedure that reminds a user about scheduled backups. If the user attempts to shut down the computer before the application has backed up the disk, the shutdown procedure could display an alert box asking whether the user wants to back up the disk before the computer shuts down.

Remember that the Process Manager frees all application heaps before the Finder calls `ShutDwnPower`. For this reason, you can't rely on your heap being intact. You should load your shutdown procedure into the system heap and specify the constants `sdOnDrivers` or `sdOnUnmount` to ensure that the procedure is executed while the system heap and any necessary system software components are still available.

The `ShutDwnInstall` procedure accepts a number of constants that specify when during the shutdown process `ShutDwnPower` or `ShutDwnStart` should execute your custom procedure. You can specify more than one constant to have your procedure executed at different phases of the process. The points indicated by these constants are: before the drivers receive good-bye messages, before volumes are unmounted, or before the computer is restarted or the power supply is switched off. However, Apple Computer, Inc., cannot guarantee the state of the computer after volumes are unmounted. Accordingly, if you plan to use the system heap or call Toolbox or Operating System routines to open a file, display a dialog box, play a sound, and so forth, be sure to specify the `sdOnDrivers` or `sdOnUnmount` constants. For more information about these constants, see the description of the `ShutDwnInstall` routine on page 8-13.

Listing 8-2 illustrates a sample custom shutdown procedure that ejects a CD-ROM disc just before the Macintosh computer shuts down or restarts.

Listing 8-2 A sample custom shutdown procedure

```
PROCEDURE MyShutDownProc;
CONST
   kMaxScsiID = 7;
VAR
   MyDCEHandle:      DCtlHandle;
   CDRefNum:         Integer;       {driver reference number}
   myID:             Integer;       {SCSI ID}
   MyDevStatHandle:  DevStatHandle; {handle to driver’s array}
BEGIN
   {Read driver reference number from the unit table.}
   CDRefNum := GetMyRefNum;
   IF CDRefNum = 0 THEN
      Exit(MyShutDownProc);

   {Get handle to driver’s device control entry.}
   MyDCEHandle := GetDCtlEntry(CDRefNum);

   {If handle is NIL, couldn’t get device control entry.}
   IF MyDCEHandle = NIL THEN
      Exit(MyShutDownProc);

   {Eject all mounted CD-ROM discs.}
   MyDevStatHandle := DevStatHandle(MyDCEHandle^^.dCtlStorage);
   FOR myID := 0 to kMaxScsiID DO
   IF (MyDevStatHandle^^[myID].isMyCDDrive = TRUE) AND
      (MyDevStatHandle^^[myID].mounted = TRUE) THEN
      MyEjectCDProc(myID);          {your routine to eject CDs}
END;
```

In Listing 8-2, `GetMyRefNum` returns the reference number for a CD device driver from the Device Manager’s unit table. A value of 0 indicates that `GetMyRefNum` did not find a CD device driver. Next, `MyShutDownProc` calls `GetDCtlEntry` to retrieve the handle to the CD driver’s device control entry record. Both the handle (of type `DCtlHandle`) and the device control entry record (of type `DCE`) are described in the chapter “Device Manager” in *Inside Macintosh: Devices*. If the value returned is `NIL`, then `GetDCtlEntry` did not find the device control entry, and `MyShutDownProc` returns. The `MyShutDownProc` procedure next loops through the driver’s device status array looking for and ejecting mounted compact discs. The `MyDevStatHandle` handle points to the driver's array. After checking all elements in the array, `MyShutDownProc` returns control to the Shutdown Manager.

Note
Listing 8-2 does not define the device driver's array or the `GetMyRefNum` and `MyEjectCDProc` routines. These items are internal to the CD-ROM driver. ◆

Once you have finished writing your shutdown procedure, you can install it by calling `ShutDwnInstall`. Your call to `ShutDwnInstall` should specify a pointer to `MyShutDownProc`, as follows.

```
ShutDwnInstall(@MyShutDwnProc, sdOnDrivers);
```

Assembly-Language Note
When the Shutdown Manager calls a shutdown procedure, it sets a bit in the D0 register indicating the current phase of the shutdown process. The values 0, 1, 2, and 3 represent the constants `sdOnPowerOff`, `sdOnRestart`, `sdOnUnmount`, and `sdOnDrivers`, respectively. You can have your shutdown procedure read D0 if it needs to keep track of the shutdown process. ◆

The Shutdown Manager follows the standard conventions for saving registers specified in *Inside Macintosh: Overview.*

You can remove your shutdown procedure at any time by calling `ShutDwnRemove`.

Shutdown Manager Reference

This section describes the routines and constants that are specific to the Shutdown Manager. For a description of the constants provided by the Shutdown Manager, see the description of the `ShutDwnInstall` procedure on page 8-13. The section "Summary of the Shutdown Manager" lists these routines and constants for your reference.

Shutdown Manager Routines

This section first describes the routines for shutting down or restarting a Macintosh computer. It then describes the routines for installing or removing a custom shutdown procedure.

Shutting Down or Restarting a Macintosh Computer

The Shutdown Manager provides the routines `ShutDwnPower` and `ShutDwnStart` to shut down or restart the computer.

▲ **WARNING**
The `ShutDwnPower` and `ShutDwnStart` procedures are used by the Finder and other system software. You usually do not need to call these two routines. ▲

ShutDwnPower

The system software calls the `ShutDwnPower` procedure to shut down a Macintosh computer.

```
PROCEDURE ShutDwnPower;
```

DESCRIPTION

The `ShutDwnPower` procedure initiates the final stage of the system shutdown process. It performs system housekeeping, executes any custom shutdown procedures installed by calls to `ShutDwnInstall`, and, if possible, turns the computer off. (The Shutdown Manager displays the Shutdown alert box if the user has to turn the computer off manually.) The system housekeeping functions consist of a five-step process, described in full in “The Shutdown Process” on page 8-4.

You should always call `ShutDwnPower` indirectly, through the Finder, to give any other applications running at the time a chance to exit gracefully. “Sending a Shutdown or Restart Event” on page 8-7 describes the correct way to shut down a Macintosh computer.

ASSEMBLY-LANGUAGE INFORMATION

The trap macro and routine selector for the `ShutDwnPower` procedure are

Trap macro	Selector
`_Shutdown`	$0001

ShutDwnStart

The system software calls the `ShutDwnStart` procedure to restart a Macintosh computer.

```
PROCEDURE ShutDwnStart;
```

DESCRIPTION

The `ShutDwnStart` procedure initiates the final stage of restarting the system. It performs system housekeeping, executes any custom shutdown procedures installed with `ShutDwnInstall`, and restarts the computer. The system housekeeping functions consist of a five-step process, described in full in "The Shutdown Process" on page 8-4.

You should always call `ShutDwnStart` indirectly, through the Finder, to give any other applications running at the time a chance to exit gracefully. "Sending a Shutdown or Restart Event" on page 8-7 describes the correct way to restart a Macintosh computer.

ASSEMBLY-LANGUAGE INFORMATION

The trap macro and routine selector for the `ShutDwnStart` procedure are

Trap macro	Selector
`_Shutdown`	$0002

Installing or Removing a Shutdown Procedure

The Shutdown Manager provides the routines `ShutDwnInstall` and `ShutDwnRemove` to install and remove custom shutdown procedures.

ShutDwnInstall

You can use the `ShutDwnInstall` procedure to install a custom shutdown procedure that performs a certain task before the computer shuts down or restarts.

```
PROCEDURE ShutDwnInstall (shutDownProc: ProcPtr; flags: Integer);
```

`shutDownProc`	A pointer to your shutdown procedure.
`flags`	An integer that indicates when during the shutdown process to execute your shutdown procedure.

DESCRIPTION

The `ShutDwnInstall` procedure installs the custom shutdown procedure pointed to by the `shutDownProc` parameter. You can install more than one custom procedure; simply call `ShutDwnInstall` for each one. For complete information on using a shutdown procedure, see "Installing a Custom Shutdown Procedure" on page 8-9.

The `flags` parameter indicates when during the shutdown process `ShutDwnPower` or `ShutDwnStart` executes your shutdown procedure. The following constants serve as masks for setting the bits in the `flags` parameter. Set the appropriate bits to have your procedure executed at different points during shutdown.

```
CONST
   sdOnPowerOff      = 1;  {call procedure before power off}
   sdOnRestart       = 2;  {call procedure before restart}
   sdRestartOrPower  = 3;  {call procedure before power off }
                           { or restart}
   sdOnUnmount       = 4;  {call procedure before unmounting }
                           { volumes}
   sdOnDrivers       = 8;  {call procedure before checking for }
                           { open drivers}
```

The following list indicates when `ShutDwnPower` or `ShutDwnStart` executes your procedure and summarizes the known state of the computer at the point specified by each constant:

Constant	**Description**
`sdOnDrivers`	The Shutdown Manager executes your procedure before checking the Device Manager's unit table for open drivers. All Toolbox and Operating System managers are available. The system heap is available. It is safe to open files, display dialog boxes, play sounds, or perform similar tasks.
`sdOnUnmount`	The Shutdown Manager executes your procedure before unmounting volumes. All Toolbox and Operating System managers are available. The system heap is available. It is safe to open files, display dialog boxes, play sounds, or perform similar tasks.
`sdOnRestart`	The Shutdown Manager executes your procedure before restarting the computer. The system heap is still available. However, in other respects, the state of the computer is indeterminate.
`sdOnPowerOff`	The Shutdown Manager executes your procedure before switching off the power supply or displaying the shutdown alert box. The system heap is still available. However, in other respects, the state of the computer is indeterminate.
`sdRestartOrPower`	The Shutdown Manager executes your procedure before restarting the computer or before switching off the power supply or displaying the shutdown alert box. The system heap is still available. However, in other respects, the state of the computer is indeterminate.

You can also combine these constants in the following ways:

Expression	Description
`sdOnPowerOff + sdOnDrivers`	When the computer is shutting down, `ShutDwnPower` calls your shutdown procedure before checking for open drivers.
`sdOnPowerOff + sdOnUnmount`	When the computer is shutting down, `ShutDwnPower` calls your shutdown procedure before unmounting volumes.
`sdOnRestart + sdOnDrivers`	When the computer is to be restarted, `ShutDwnStart` calls your shutdown procedure before checking for open drivers.
`sdOnRestart + sdOnUnmount`	When the computer is to be restarted, `ShutDwnStart` calls your shutdown procedure before unmounting volumes.

Note
These combinations of constants are recognized by the Shutdown Manager only in system software versions 7.0 and later. ◆

The Shutdown Manager executes a custom shutdown procedure just once. As soon as a custom procedure returns, the Shutdown Manager removes the address and flag entries for that procedure from its shutdown queue. As a result, the combination `sdOnDrivers + sdOnUnmount` does not work.

If your driver or system extension remains resident in memory after the boot process, be sure to load your shutdown procedure into the system heap because the Process Manager frees all application and other temporary heaps before calling the Shutdown Manager.

ASSEMBLY-LANGUAGE INFORMATION

The trap macro and routine selector for the `ShutDwnInstall` procedure are

Trap macro	Selector
`_Shutdown`	$0003

ShutDwnRemove

The `ShutDwnRemove` procedure removes a shutdown procedure that you have previously installed by calling `ShutDwnInstall`.

```
PROCEDURE ShutDwnRemove (shutDownProc: ProcPtr);
```

`shutDownProc`
A pointer to your shutdown procedure.

DESCRIPTION

The `ShutDwnRemove` procedure removes the shutdown procedure pointed to by the `shutDownProc` parameter. The `ShutDwnRemove` procedure is useful for removing a shutdown procedure when the device controlled by the driver that installed the shutdown procedure is not operating.

If you have specified that your procedure should be executed at several points during the shutdown process (for instance, before unmounting at restart and before unmounting at power off), `ShutDwnRemove` removes it at all points.

ASSEMBLY-LANGUAGE INFORMATION

The trap macro and routine selector for the `ShutDwnRemove` procedure are

Trap macro	Selector
`_Shutdown`	$0004

Application-Defined Routine

While the computer is restarting or shutting down, you can provide a custom shutdown procedure to perform any housekeeping tasks that your device driver or system extension requires. For example, your device driver might have to park the drive head or your system extension might have to write some statistics to a log. However, under normal circumstances, applications don't need to use a custom shutdown procedure.

Shutdown Procedures

A shutdown procedure performs any last-minute tasks required to put a device driver (or, in rare cases, an application) in a stable state before the computer restarts or shuts down.

Note
Applications can usually perform housekeeping tasks before they quit. If your application requires that some action be taken after it quits, such as having the system display a dialog box, you should use a system extension whenever possible. ◆

MyShutDownProc

A typical shutdown procedure has the following form:

```
PROCEDURE MyShutDownProc;
```

DESCRIPTION

You can install the address of your shutdown procedure in the Shutdown Manager's queue by calling the `ShutDwnInstall` procedure. When the computer restarts or shuts down, the Shutdown Manager searches its queue, reads the addresses it finds there, and executes the procedures at the points specified by the shutdown flag or flags that you passed with `ShutDwnInstall`. See the description of `ShutDwnInstall` on page 8-13 for details.

To remove your shutdown procedure, call the `ShutDwnRemove` routine, passing it a pointer to your procedure.

Be sure to install your shutdown procedure in the system heap. Because the Process Manager deallocates memory for all application heaps before the Finder calls `ShutDwnPower`, you can't rely on the application heap being intact. In addition, if you need to use Toolbox managers, specify the constants `sdOnDrivers` or `sdOnUnmount` to ensure that your procedure is executed while these managers are still available.

ASSEMBLY-LANGUAGE INFORMATION

The Shutdown Manager conforms with the standard assembly-language conventions for saving registers. The Shutdown Manager does not preserve the contents of these registers. Therefore, be sure to save and restore the data and address registers before issuing a jump instruction to your shutdown procedure.

Summary of the Shutdown Manager

Pascal Summary

Constants

```
CONST
   {masks for ShutDwnInstall flags}
   sdOnPowerOff      = 1;  {call procedure before power off}
   sdOnRestart       = 2;  {call procedure before restart}
   sdRestartOrPower  = 3;  {call procedure before power off or restart}
   sdOnUnmount       = 4;  {call procedure before unmounting volumes}
   sdOnDrivers       = 8;  {call procedure before checking for open drivers}
```

Shutdown Manager Routines

Shutting Down or Restarting the Computer

```
PROCEDURE ShutDwnPower;
PROCEDURE ShutDwnStart;
```

Installing or Removing a Shutdown Procedure

```
PROCEDURE ShutDwnInstall    (shutDownProc: ProcPtr; flags: Integer);
PROCEDURE ShutDwnRemove     (shutDownProc: ProcPtr);
```

Application-Defined Routine

Shutdown Procedures

```
PROCEDURE MyShutDownProc;
```

C Summary

Constants

```
/*masks for ShutDwnInstall flags*/
enum {
   sdOnPowerOff      = 1,  /*call procedure before power off*/
   sdOnRestart       = 2,  /*call procedure before restart*/
   sdRestartOrPower  = 3,  /*call procedure before power off or restart*/
   sdOnUnmount       = 4,  /*call procedure before unmounting volumes*/
   sdOnDrivers       = 8   /*call procedure before checking for open */
};                         /* drivers*/
```

Data Types

```
typedef pascal void (*ShutDwnProcPtr)(void)
```

Shutdown Manager Routines

Shutting Down or Restarting the Computer

```
pascal void ShutDwnPower    (void);
pascal void ShutDwnStart    (void);
```

Installing or Removing a Shutdown Procedure

```
pascal void ShutDwnInstall  (ShutDwnProcPtr shutDownProc, short flags);
pascal void ShutDwnRemove   (ShutDwnProcPtr shutDownProc);
```

Application-Defined Routine

Shutdown Procedures

```
pascal void MyShutDownProc  (void);
```

Assembly-Language Summary

Constants

```
sdPowerOff        EQU      1        ;selector for ShutDwnPower
sdRestart         EQU      2        ;selector for ShutDwnStart
sdInstall         EQU      3        ;selector for ShutDwnInstall
sdRemove          EQU      4        ;selector for ShutDwnRemove
```

Trap Macros Requiring Routine Selectors

`_Shutdown`

Selector	Routine
$0001	`ShutDwnPower`
$0002	`ShutDwnStart`
$0003	`ShutDwnInstall`
$0004	`ShutDwnRemove`