Display Manager Reference

(Not Recommended)

Carbon > Graphics & Imaging

2007-12-04

Apple Inc.
1 Infinite Loop
Cupertino, CA 95014
408-996-1010

Simultaneously published in the United States and Canada.

Contents

Display Manager Reference (Not Recommended)

Framework: Carbon/Carbon.h

Declared in Displays.h

Overview

Important: The Display Manager is deprecated in Mac OS X version 10.4 and later. The replacement is Quartz Display Services, a modern Mac OS X API that provides similar functionality. For more information, see *Quartz Display Services Reference*.

In Mac OS 9 and earlier, the Display Manager allowed users to dynamically change the arrangement and display modes of the monitors attached to their computers. The Display Manager was included in Carbon to facilitate the porting of legacy applications to Mac OS X. You should not use Display Manager functions in new application development. Instead, you should use Quartz Display Services.

Functions by Task

Adding and Removing Video Devices From the Device List

`DMAddDisplay` (page 49) Deprecated in Mac OS X v10.4

Adds the `GDevice` structure for a video device to the device list. (Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMDisposeDisplay` (page 55) Deprecated in Mac OS X v10.4

Disposes of the `GDevice` structure for a video device. (Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMDisposeList` (page 56) Deprecated in Mac OS X v10.4

Disposes of a display mode list built by `DMNewDisplayModeList`. (Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMGetIndexedDisplayModeFromList` (page 65) Deprecated in Mac OS X v10.4

Obtains a display mode from the display mode list built by `DMNewDisplayModeList`. (Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMNewDisplay` (page 74) Deprecated in Mac OS X v10.4

Adds a video device to the device list and makes the device active. (Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMNewDisplayModeList` (page 75) Deprecated in Mac OS X v10.4

Builds a new display mode list for a specified video device. (Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMRemoveDisplay` (page 78) Deprecated in Mac OS X v10.4

Removes a video device from the device list. (Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

Changing Display Modes and Display Configurations

`DMBeginConfigureDisplays` (page 50) Deprecated in Mac OS X v10.4

Allows your application to configure displays. You should generally never need to use this function. (Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMBlockMirroring` (page 51) Deprecated in Mac OS X v10.4

Disables video mirroring. You should generally never need to use this function. (Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMDisableDisplay` (page 54) Deprecated in Mac OS X v10.4

Makes a video device inactive by removing its display area from the desktop. You should generally never need to use this function. (Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMEnableDisplay` (page 57) Deprecated in Mac OS X v10.4

Reactivates a display made inactive with the function `DMDisableDisplay`. You should generally never need to use this function. (Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMEndConfigureDisplays` (page 58) Deprecated in Mac OS X v10.4

Ends configuration begun by `DMBeginConfigureDisplays`. You should generally never need to use this function. (Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMMirrorDevices` (page 69) Deprecated in Mac OS X v10.4

Turns on video mirroring. (Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMMoveDisplay` (page 70) Deprecated in Mac OS X v10.4

Moves the boundary rectangle for a video device. You should generally never need to use this function. (Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMSetDisplayMode` (page 83) Deprecated in Mac OS X v10.4

Sets the display mode and pixel depth for a video device. (Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMSetMainDisplay` (page 84) Deprecated in Mac OS X v10.4

Sets a display to be the main screen. You should generally never need to use this function. (Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMUnblockMirroring` (page 85) Deprecated in Mac OS X v10.4

Reenables video mirroring disabled by the function `DMUnblockMirroring`. You should generally never need to use this function. (Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMUnmirrorDevice` (page 86) Deprecated in Mac OS X v10.4

Turns off video mirroring. (Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

Determining Display Modes and Display Configurations

`DMCanMirrorNow` (page 52) Deprecated in Mac OS X v10.4

Determines whether video mirroring can be activated on the user's computer system. (Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMCheckDisplayMode` (page 52) Deprecated in Mac OS X v10.4

Determines if a video device supports a particular display mode and pixel depth. (Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMGetAVPowerState` (page 59) Deprecated in Mac OS X v10.4

Obtains the current power state of a display. (Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMGetDisplayMode` (page 62) Deprecated in Mac OS X v10.4

Obtains the current display mode of a specified video display. (Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMGetGraphicInfoByAVID` (page 64) Deprecated in Mac OS X v10.4

Obtains information about the graphic display of a display device. (Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMGetNameByAVID` (page 66) Deprecated in Mac OS X v10.4

Obtains the name of a display device. (Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMIsMirroringOn` (page 69) Deprecated in Mac OS X v10.4

Determines if video mirroring is active. (Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMQDIsMirroringCapable` (page 76) Deprecated in Mac OS X v10.4

Determines if QuickDraw supports video mirroring on the user's system. (Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMSaveScreenPrefs` (page 80) Deprecated in Mac OS X v10.4

Saves the user's screen configuration preferences. (Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMSetAVPowerState` (page 82) Deprecated in Mac OS X v10.4

Sets the power state of a display device. (Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

Getting Video Devices

`DMGetDisplayIDByGDevice` (page 61) Deprecated in Mac OS X v10.4

Obtains the display ID number for a video device. (Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMGetFirstScreenDevice` (page 62) Deprecated in Mac OS X v10.4

Returns a handle for the first video device in the device list. (Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMGetGDeviceByDisplayID` (page 63) Deprecated in Mac OS X v10.4

Obtains a handle for the video device with a specified display ID. (Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMGetNextMirroredDevice` (page 67) Deprecated in Mac OS X v10.4

Obtains a handle for a video device that mirrors another specified video device. (Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMGetNextScreenDevice` (page 67) Deprecated in Mac OS X v10.4

Returns a handle for the next video device in the device list. (Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

Registering and Unregistering Your Program

`DMRegisterExtendedNotifyProc` (page 76) Deprecated in Mac OS X v10.4

Registers a function that responds to a Display Notice event outside of an event loop. (Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMRemoveExtendedNotifyProc` (page 79) Deprecated in Mac OS X v10.4

Removes your Display Notice event-handling function registered by the `DMRegisterExtendedNotifyProc` function. (Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMSendDependentNotification` (page 81) Deprecated in Mac OS X v10.4

Notifies dependent displays of changes in depth mode or configuration. (Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

Working With Universal Procedure Pointers for Display Manager Callbacks

`DisposeDMComponentListIteratorUPP` (page 47) Deprecated in Mac OS X v10.4

(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DisposeDMDisplayListIteratorUPP` (page 47) Deprecated in Mac OS X v10.4

(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DisposeDMDisplayModeListIteratorUPP` (page 47) Deprecated in Mac OS X v10.4

(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DisposeDMExtendedNotificationUPP` (page 48) Deprecated in Mac OS X v10.4

(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DisposeDMNotificationUPP` (page 48) Deprecated in Mac OS X v10.4

(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DisposeDMProfileListIteratorUPP` (page 48) Deprecated in Mac OS X v10.4

(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`InvokeDMComponentListIteratorUPP` (page 87) Deprecated in Mac OS X v10.4

(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`InvokeDMDisplayListIteratorUPP` (page 87) Deprecated in Mac OS X v10.4

(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`InvokeDMDisplayModeListIteratorUPP` (page 87) Deprecated in Mac OS X v10.4

(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`InvokeDMExtendedNotificationUPP` (page 88) Deprecated in Mac OS X v10.4

(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`InvokeDMNotificationUPP` (page 88) Deprecated in Mac OS X v10.4

(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`InvokeDMProfileListIteratorUPP` (page 88) Deprecated in Mac OS X v10.4
(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`NewDMComponentListIteratorUPP` (page 89) Deprecated in Mac OS X v10.4
(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`NewDMDisplayListIteratorUPP` (page 89) Deprecated in Mac OS X v10.4
(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`NewDMDisplayModeListIteratorUPP` (page 89) Deprecated in Mac OS X v10.4
(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`NewDMExtendedNotificationUPP` (page 90) Deprecated in Mac OS X v10.4
(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`NewDMNotificationUPP` (page 90) Deprecated in Mac OS X v10.4
(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`NewDMProfileListIteratorUPP` (page 90) Deprecated in Mac OS X v10.4
(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

Miscellaneous

`DMConfirmConfiguration` (page 53) Deprecated in Mac OS X v10.4
(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMDisposeAVComponent` (page 55) Deprecated in Mac OS X v10.4
(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMDrawDesktopRect` (page 57) Deprecated in Mac OS X v10.4
(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMDrawDesktopRegion` (page 57) Deprecated in Mac OS X v10.4
(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMGetDeskRegion` (page 60) Deprecated in Mac OS X v10.4
(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMGetDeviceAVIDByPortAVID` (page 60) Deprecated in Mac OS X v10.4
(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMGetDeviceComponentByAVID` (page 60) Deprecated in Mac OS X v10.4
(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMGetDisplayComponent` (page 61) Deprecated in Mac OS X v10.4
(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMGetEnableByAVID` (page 62) Deprecated in Mac OS X v10.4
(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMGetIndexedComponentFromList` (page 65) Deprecated in Mac OS X v10.4
(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMGetPortComponentByAVID` (page 68) Deprecated in Mac OS X v10.4
(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMNewAVDeviceList` (page 71) Deprecated in Mac OS X v10.4
(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMNewAVEngineList` (page 72) Deprecated in Mac OS X v10.4
(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMNewAVIDByDeviceComponent` (page 72) Deprecated in Mac OS X v10.4
(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMNewAVIDByPortComponent` (page 72) Deprecated in Mac OS X v10.4
(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMNewAVPanelList` (page 73) Deprecated in Mac OS X v10.4
(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMNewAVPortListByDeviceAVID` (page 73) Deprecated in Mac OS X v10.4
(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMNewAVPortListByPortType` (page 73) Deprecated in Mac OS X v10.4
(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMRegisterNotifyProc` (page 77) Deprecated in Mac OS X v10.4
(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMRemoveNotifyProc` (page 80) Deprecated in Mac OS X v10.4
(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMResolveDisplayComponents` (page 80) Deprecated in Mac OS X v10.4
(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMSetDisplayComponent` (page 83) Deprecated in Mac OS X v10.4
(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

`DMSetEnableByAVID` (page 84) Deprecated in Mac OS X v10.4
(Deprecated. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

Callbacks

DMComponentListIteratorProcPtr

```
typedef void (*DMComponentListIteratorProcPtr)
(
    void * userData,
    DMListIndexType itemIndex,
    DMComponentListEntryPtr componentInfo
);
```

If you name your function `MyDMComponentListIteratorProc`, you would declare it like this:

```
void MyDMComponentListIteratorProc (
    void * userData,
    DMListIndexType itemIndex,
    DMComponentListEntryPtr componentInfo
);
```

Availability
Available in Mac OS X v10.0 and later.

Declared In
`Displays.h`

DMDisplayListIteratorProcPtr

```
typedef void (*DMDisplayListIteratorProcPtr)
(
    void * userData,
    DMListIndexType itemIndex,
    DisplayListEntryPtr displaymodeInfo
);
```

If you name your function `MyDMDisplayListIteratorProc`, you would declare it like this:

```
void MyDMDisplayListIteratorProc (
    void * userData,
    DMListIndexType itemIndex,
    DisplayListEntryPtr displaymodeInfo
);
```

Availability
Available in Mac OS X v10.0 and later.

Declared In
`Displays.h`

DMDisplayModeListIteratorProcPtr

Defines a pointer to a list iterator callback function.

```
typedef void (*DMDisplayModeListIteratorProcPtr)
(
    void * userData,
    DMListIndexType itemIndex,
    DMDisplayModeListEntryPtr displaymodeInfo
);
```

If you name your function `MyDMDisplayModeListIteratorProc`, you would declare it like this:

```
void MyDMDisplayModeListIteratorProc (
    void * userData,
    DMListIndexType itemIndex,
    DMDisplayModeListEntryPtr displaymodeInfo
);
```

Parameters

userData

A pointer to data about mode changes provided by the user.

itemIndex

Specifies the list entry. See `DMListIndexType` (page 25) for more information. This is the index passed into `DMGetIndexedDisplayModeFromList` (page 65).

displaymodeInfo

A pointer to a structure of type `DMDisplayModeListEntryRec` (page 23) that provides display mode information.

Discussion
The function DMGetIndexedDisplayModeFromList (page 65) uses this callback function to retrieve and return information about a display mode to the caller of DMGetIndexedDisplayModeFromList.

When you implement this function, the pointer you pass to the DMGetIndexedDisplayModeFromList function should be a universal procedure pointer with the following type definition:

```
typedef (DMDisplayModeListIteratorProcPtr)
DMDisplayModeListIteratorUPP;
```

To create a universal procedure pointer for your application-defined function, you should use the NewDMDisplayModeListIteratorUPP function as follows:

```
DMDisplayModeListIteratorUPP MyDMDisplayModeListIteratorUPP;
MyDMDisplayModeListIteratorUPP = NewDMDisplayModeListIteratorUPP
(&MyDMDisplayModeListIteratorCallback)
```

You can then pass MyDMDisplayModeListIteratorUPP in the listIterator parameter of the DMGetIndexedDisplayModeFromList (page 65) function. When you no longer need the list iterator, you should dispose of the UPP using the DisposeDMDisplayModeListIteratorUPP function:

```
DisposeDMDisplayModeListIteratorUPP (
    MyDMDisplayModeListIteratorUPP);
```

Using this call ensures that the call is made through a universal procedure pointer.

Special Considerations

Because this function may move or purge memory blocks or access handles, you cannot call it at interrupt time.

Availability
Available in Mac OS X v10.0 and later.

Declared In
Displays.h

DMExtendedNotificationProcPtr

Defines a pointer to an extended notification callback function.

```
typedef void (*DMExtendedNotificationProcPtr)
(
    void * userData,
    short theMessage,
    void * notifyData
);
```

If you name your function MyDMExtendedNotificationProc, you would declare it like this:

```
void MyDMExtendedNotificationProc (
    void * userData,
    short theMessage,
    void * notifyData
);
```

Parameters

userData

A pointer you passed into DMRegisterExtendedNotifyProc (page 76).

theMessage

A message selector. See "Notification Messages" (page 39) for information on specific message selectors.

notifyData

A pointer to message-specific information data provided by the the Display Manager, described in "Notification Messages" (page 39).

Discussion

Display Manager notification functions use this callback function when your application needs to know when certain events have occurred. The system software may implement these events or follow a user action. When these events occur, the Display Manager will send notification messages to registrants.

When you call the function DMRegisterExtendedNotifyProc (page 76) you designate an application-defined function to handle the extended notification procedure.

When you implement this function, the pointer you pass to the DMRegisterExtendedNotifyProc function should be a universal procedure pointer with the following type definition:

```
typedef (DMExtendedNotificationProcPtr)             DMExtendedNotificationUPP;
```

To create a universal procedure pointer for your application-defined function, you should use the NewDMExtendedNotificationProc macro as follows:

```
DMExtendedNotificationUPP  MyExtendedNotificationUPP;
MyExtendedNotificationUPP = NewDMExtendedNotificationProc
(MyExtendedNotificationCallback);
```

You can then pass MyExtendedNotificationUPP in the notifyProc parameter of the DMRegisterExtendedNotifyProc (page 76) function. When you no longer need notifications, you should remove it using the DMRemoveExtendedNotifyProc (page 79) function. You sould also dispose of the UPP using the DisposeDMExtendedNotificationUPP function:

```
DisposeDMExtendedNotificationUPP(MyExtendedNotificationUPP);
```

Using this call ensures that the call is made through a universal procedure pointer.

Special Considerations

Because this function may move or purge memory blocks or access handles, you cannot call it at interrupt time.

Availability

Available in Mac OS X v10.0 and later.

Declared In

Displays.h

DMNotificationProcPtr

```
typedef void (*DMNotificationProcPtr)
(
    AppleEvent * theEvent
);
```

If you name your function `MyDMNotificationProc`, you would declare it like this:

```
void MyDMNotificationProc (
    AppleEvent * theEvent
);
```

Availability
Available in Mac OS X v10.0 and later.

Declared In
`Displays.h`

DMProfileListIteratorProcPtr

```
typedef void (*DMProfileListIteratorProcPtr)
(
    void * userData,
    DMListIndexType itemIndex,
    DMProfileListEntryPtr profileInfo
);
```

If you name your function `MyDMProfileListIteratorProc`, you would declare it like this:

```
void MyDMProfileListIteratorProc (
    void * userData,
    DMListIndexType itemIndex,
    DMProfileListEntryPtr profileInfo
);
```

Availability
Available in Mac OS X v10.0 and later.

Declared In
`Displays.h`

Data Types

AVLocationRec

```
struct AVLocationRec {
    unsigned long locationConstant;
};
typedef struct AVLocationRec AVLocationRec;
typedef AVLocationRec * AVLocationPtr;
```

Fields

`locationConstant`

Reserved for future expansion. Set this field to zero.

Discussion

The function `DMGetGraphicInfoByAVID` (page 64) uses the `AVLocationRec` structure to get information about graphic displays.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`Displays.h`

AVPowerStatePtr

```
typedef VDPowerStateRec * AVPowerStatePtr;
```

Availability

Available in Mac OS X v10.0 and later.

Declared In

`Displays.h`

AVPowerStateRec

```
typedef VDPowerStateRec AVPowerStateRec;
```

Discussion

The functions `DMGetAVPowerState` (page 59) and `DMSetAVPowerState` (page 82) contain a parameter of type `AVPowerStatePtr`, which is a pointer to the `AVPowerStateRec` data type.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`Displays.h`

DependentNotifyRec

```
struct DependentNotifyRec {
    ResType notifyType;
    ResType notifyClass;
    DisplayIDType notifyPortID;
    ComponentInstance notifyComponent;
    unsigned long notifyVersion;
    unsigned long notifyFlags;
    unsigned long notifyReserved;
    unsigned long notifyFuture;
};
typedef struct DependentNotifyRec DependentNotifyRec;
typedef DependentNotifyRec * DependentNotifyPtr;
```

Fields

`notifyType`

A value that specifies the type of engine, if any, that made the change. The Display Manager may set this field to zero.

`notifyClass`

A value specifying the class of change that occurred: for instance, color or screen size. This field uses a value supplied by the constant described under "Dependent Notification Constants" (page 33) to specify the class of change that has occurred in a dependent display.

`notifyPortID`

Specifies which device was touched (`kInvalidDisplayID` specifies all or none).

`notifyComponent`

A value that indentifies the engine that made the change. The Display Manager may set this field to zero.

`notifyVersion`

Reserved for future expansion. The Display Manager sets this field to zero.

`notifyFlags`

Reserved for future expansion. The Display Manager sets this field to zero.

`notifyReserved`

Reserved for future expansion. The Display Manager sets this field to zero.

`notifyFuture`

Reserved for future expansion. The Display Manager sets this field to zero.

Discussion

The function `DMSendDependentNotification` (page 81) uses the `notifyType` and `notifyClass` fields of the `DependentNotifyRec` structure.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`Displays.h`

DisplayListEntryRec

```
struct DisplayListEntryRec {
    GDHandle displayListEntryGDevice;
    DisplayIDType displayListEntryDisplayID;
    UInt32 displayListEntryIncludeFlags;
    UInt32 displayListEntryReserved1;
    UInt32 displayListEntryReserved2;
    UInt32 displayListEntryReserved3;
    UInt32 displayListEntryReserved4;
    UInt32 displayListEntryReserved5;
};
typedef struct DisplayListEntryRec DisplayListEntryRec;
typedef DisplayListEntryRec * DisplayListEntryPtr;
```

Fields

`displayListEntryGDevice`
: A value of type GDHandle.

`displayListEntryDisplayID`
: A value of type `DisplayIDType` that specifies the display ID.

`displayListEntryIncludeFlags`
: A value of type `UInt32` that specifies the reason this entry was included.

`displayListEntryReserved1`
: Reserved for future expansion. Set this field to zero.

`displayListEntryReserved2`
: Reserved for future expansion. Set this field to zero.

`displayListEntryReserved3`
: Reserved for future expansion. Set this field to zero.

`displayListEntryReserved4`
: Reserved for future expansion. Set this field to zero.

`displayListEntryReserved5`
: Reserved for future expansion. Set this field to zero.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`Displays.h`

DMComponentListEntryRec

```
struct DMComponentListEntryRec {
    DisplayIDType itemID;
    Component itemComponent;
    ComponentDescription itemDescription;
    ResType itemClass;
    DMFidelityType itemFidelity;
    ResType itemSubClass;
    Point itemSort;
    unsigned long itemFlags;
    ResType itemReserved;
    unsigned long itemFuture1;
    unsigned long itemFuture2;
    unsigned long itemFuture3;
    unsigned long itemFuture4;
};
typedef struct DMComponentListEntryRec DMComponentListEntryRec;
typedef DMComponentListEntryRec * DMComponentListEntryPtr;
```

Fields

`itemID`

`itemComponent`

`itemDescription`

`itemClass`

`itemFidelity`

`itemSubClass`

`itemSort`

Reserved for future expansion. Set this field to zero.

`itemFlags`

Reserved for future expansion. Set this field to zero.

`itemReserved`

`itemFuture1`

Reserved for future expansion. Set this field to zero.

`itemFuture2`

Reserved for future expansion. Set this field to zero.

`itemFuture3`

Reserved for future expansion. Set this field to zero.

`itemFuture4`

Reserved for future expansion. Set this field to zero.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`Displays.h`

DMComponentListIteratorUPP

```
typedef DMComponentListIteratorProcPtr DMComponentListIteratorUPP;
```

Discussion
For more information, see the description of the `DMComponentListIteratorProcPtr` (page 12) callback function.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`Displays.h`

DMDepthInfoBlockRec

```
struct DMDepthInfoBlockRec {
    unsigned long depthBlockCount;
    DMDepthInfoPtr depthVPBlock;
    unsigned long depthBlockFlags;
    unsigned long depthBlockReserved1;
    unsigned long depthBlockReserved2;
};
typedef struct DMDepthInfoBlockRec DMDepthInfoBlockRec;
typedef DMDepthInfoBlockRec * DMDepthInfoBlockPtr;
```

Fields

`depthBlockCount`
Specifies the number of mode depths available.

`depthVPBlock`
Array of `DMDepthInfoRec` (page 22).

`depthBlockFlags`
Reserved for future expansion.

`depthBlockReserved1`
Reserved for future expansion.

`depthBlockReserved2`
Reserved for future expansion.

Discussion
When you call the function `DMGetIndexedDisplayModeFromList` (page 65) , the Display Manager passes a pointer to a `DMDisplayModeListEntryRec` (page 23) structure to your application. Its field `displayModeDepthBlockInfo` is a pointer to a `DMDepthInfoBlockRec` structure.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`Displays.h`

DMDepthInfoRec

```
struct DMDepthInfoRec {
    VDSwitchInfoPtr depthSwitchInfo;
    VPBlockPtr depthVPBlock;
    UInt32 depthFlags;
    UInt32 depthReserved1;
    UInt32 depthReserved2;
};
typedef struct DMDepthInfoRec DMDepthInfoRec;
typedef DMDepthInfoRec * DMDepthInfoPtr;
```

Fields

`depthSwitchInfo`

A pointer to the structure `VDSwitchInfoRec`, which contains values that specify information on video switch modes and data.

`depthVPBlock`

A pointer to the structure `VPBlock`, which supplies information about size, depth and format.

`depthFlags`

Values from the video structure `VDVideoParametersInfoRec`, which specify color, size, and depth.

`depthReserved1`

Reserved for future expansion.

`depthReserved2`

Reserved for future expansion.

Discussion

This structure provides information that the structure `DMDepthInfoBlockRec` (page 21) supplies to the function `DMGetIndexedDisplayModeFromList` (page 65).

Availability

Available in Mac OS X v10.0 and later.

Declared In

`Displays.h`

DMDisplayListIteratorUPP

```
typedef DMDisplayListIteratorProcPtr DMDisplayListIteratorUPP;
```

Discussion

For more information, see the description of the `DMDisplayListIteratorProcPtr` (page 13) callback function.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`Displays.h`

DMDisplayModeListEntryRec

```
struct DMDisplayModeListEntryRec {
    UInt32 displayModeFlags;
    VDSwitchInfoPtr displayModeSwitchInfo;
    VDResolutionInfoPtr displayModeResolutionInfo;
    VDTimingInfoPtr displayModeTimingInfo;
    DMDepthInfoBlockPtr displayModeDepthBlockInfo;
    UInt32 displayModeVersion;
    StringPtr displayModeName;
    DMDisplayTimingInfoPtr displayModeDisplayInfo;
};
typedef struct DMDisplayModeListEntryRec DMDisplayModeListEntryRec;
typedef DMDisplayModeListEntryRec * DMDisplayModeListEntryPtr;
```

Fields

`displayModeFlags`

A pointer to a video structure, `VDSwitchInfoRec`, which provides information you need to tell the driver how to switch into different configurations, bit depths, or resolutions. See the function `DMSetDisplayMode` (page 83) for more information.

`displayModeSwitchInfo`

A pointer to a `VDSwitchInfoRec` video structure, which provides information you need to tell the driver how to switch into different configurations, bit depths, or resolutions. See the function `DMSetDisplayMode` (page 83) for more information.

`displayModeResolutionInfo`

A pointer to a pointer to a `VDResolutionInfoRec` video structure, which provides information about horizontal pixels, maximum depth modes, and the vertical line of the specified display mode.

`displayModeTimingInfo`

A pointer to a pointer to a `VDTimingInfoRec` video structure, which provides information about timing, format of the specified display mode.

`displayModeDepthBlockInfo`

A pointer to a `DMDepthInfoBlockRec` (page 21) structure, which provides information about available pixel formats and the `VPBlock`, including size and depth.

`displayModeVersion`

The version of this structure. Currently it is version `kDisplayTimingInfoVersionOne`. See "Display Version Values" (page 35) for more information.

`displayModeName`

A string pointer giving the display mode name.

`displayModeDisplayInfo`

A pointer to the `DMDisplayTimingInfoRec` (page 24) data type. This data type supplies information about the quality and default values of the timing.

Discussion

The `DMDisplayModeListEntryRec` structure contains information about a display mode in a display mode list built by the function `DMNewDisplayModeList` (page 75).

Availability

Available in Mac OS X v10.0 and later.

Declared In

`Displays.h`

DMDisplayModeListIteratorUPP

```
typedef DMDisplayModeListIteratorProcPtr DMDisplayModeListIteratorUPP;
```

Discussion
For more information, see the description of the `DMDisplayModeListIteratorProcPtr` (page 13) callback function.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`Displays.h`

DMDisplayTimingInfoRec

```
struct DMDisplayTimingInfoRec {
    UInt32 timingInfoVersion;
    UInt32 timingInfoAttributes;
    SInt32 timingInfoRelativeQuality;
    SInt32 timingInfoRelativeDefault;
    UInt32 timingInfoReserved[16];
};
typedef struct DMDisplayTimingInfoRec DMDisplayTimingInfoRec;
typedef DMDisplayTimingInfoRec * DMDisplayTimingInfoPtr;
```

Fields

`timingInfoVersion`
: An unsigned 32 bit integer that shows the timing version. See "Display Version Values" (page 35) for timing version values.

`timingInfoAttributes`
: An unsigned 32 bit integer that the Display Manager sets to show timing attributes.

`timingInfoRelativeQuality`
: A signed 32 bit integer whose flags the Display Manager sets to provide information on the quality of the timing.

`timingInfoRelativeDefault`
: A signed 32 bit integer the Display Manager sets that specifies the relative default value of the timing.

`timingInfoReserved`
: Reserved for future expansion.

Discussion
This structure supplies information about timing attributes, defaults and values to the structure `DMDisplayModeListEntryRec` (page 23).

Availability
Available in Mac OS X v10.0 and later.

Declared In
`Displays.h`

DMExtendedNotificationUPP

```
typedef DMExtendedNotificationProcPtr DMExtendedNotificationUPP;
```

Discussion
For more information, see the description of the `DMExtendedNotificationProcPtr` (page 14) callback function.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`Displays.h`

DMFidelityType

```
typedef UInt32 DMFidelityType;
```

Availability
Available in Mac OS X v10.0 and later.

Declared In
`Displays.h`

DMListIndexType

```
typedef unsigned long DMListIndexType;
```

Discussion
The function `DMGetIndexedDisplayModeFromList` (page 65) uses this data type to supply a list of display modes from which you can obtain information about a specified display mode.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`Displays.h`

DMListType

```
typedef void * DMListType;
```

Availability
Available in Mac OS X v10.0 and later.

Declared In
`Displays.h`

DMMakeAndModelRec

```
struct DMMakeAndModelRec {
    ResType manufacturer;
    UInt32 model;
    UInt32 serialNumber;
    UInt32 manufactureDate;
    UInt32 makeReserved[4];
};
typedef struct DMMakeAndModelRec DMMakeAndModelRec;
typedef DMMakeAndModelRec * DMMakeAndModelPtr;
```

Fields

`manufacturer`
Represents the manufacturer of the specified display.

`model`
Represents the model name of the specified display.

`serialNumber`
Represents the serial number of the specified display.

`manufactureDate`
Represents the date of manufacture of the specified display.

`makeReserved`
Reserved for future expansion.

Discussion

This structure stores information about a specified monitor or display. If you need to keep track of configurations and user preferences, you can store that information in this structure.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`Displays.h`

DMModalFilterUPP

```
typedef void * DMModalFilterUPP;
```

Availability

Available in Mac OS X v10.0 and later.

Declared In

`Displays.h`

DMNotificationUPP

```
typedef DMNotificationProcPtr DMNotificationUPP;
```

Discussion

For more information, see the description of the `DMNotificationProcPtr` (page 16) callback function.

Availability

Available in Mac OS X v10.0 and later.

Declared In
Displays.h

DMProcessInfoPtr

```
typedef void * DMProcessInfoPtr;
```

Availability
Available in Mac OS X v10.0 and later.

Declared In
Displays.h

DMProfileListEntryRec

```
struct DMProfileListEntryRec {
    CMProfileRef profileRef;
    Ptr profileReserved1;
    Ptr profileReserved2;
    Ptr profileReserved3;
};
typedef struct DMProfileListEntryRec DMProfileListEntryRec;
typedef DMProfileListEntryRec * DMProfileListEntryPtr;
```

Availability
Available in Mac OS X v10.0 and later.

Declared In
Displays.h

DMProfileListIteratorUPP

```
typedef DMProfileListIteratorProcPtr DMProfileListIteratorUPP;
```

Discussion
For more information, see the description of the DMProfileListIteratorProcPtr (page 16) callback function.

Availability
Available in Mac OS X v10.0 and later.

Declared In
Displays.h

Constants

Active Device Only Values

```
enum {
    dmOnlyActiveDisplays = true,
    dmAllDisplays = false
};
```

Constants

`dmOnlyActiveDisplays`

Returns a handle to the `GDevice` structure for an active device only.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`dmAllDisplays`

Returns a handle to the `GDevice` structure for a device, active or not.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

Discussion

The functions `DMGetFirstScreenDevice` (page 62) and `DMGetNextScreenDevice` (page 67) contain the parameter `activeOnly` which you can specify with an Active Device Constant.

Apple Event Notification Keywords

```
enum {
    kAESystemConfigNotice = 'cnfg',
    kAEDisplayNotice = 'dspl',
    kAEDisplaySummary = 'dsum',
    keyDMConfigVersion = 'dmcv',
    keyDMConfigFlags = 'dmcf',
    keyDMConfigReserved = 'dmcr',
    keyDisplayID = 'dmid',
    keyDisplayComponent = 'dmdc',
    keyDisplayDevice = 'dmdd',
    keyDisplayFlags = 'dmdf',
    keyDisplayMode = 'dmdm',
    keyDisplayModeReserved = 'dmmr',
    keyDisplayReserved = 'dmdr',
    keyDisplayMirroredId = 'dmmi',
    keyDeviceFlags = 'dddf',
    keyDeviceDepthMode = 'dddm',
    keyDeviceRect = 'dddr',
    keyPixMapRect = 'dpdr',
    keyPixMapHResolution = 'dphr',
    keyPixMapVResolution = 'dpvr',
    keyPixMapPixelType = 'dppt',
    keyPixMapPixelSize = 'dpps',
    keyPixMapCmpCount = 'dpcc',
    keyPixMapCmpSize = 'dpcs',
    keyPixMapAlignment = 'dppa',
    keyPixMapResReserved = 'dprr',
    keyPixMapReserved = 'dppr',
    keyPixMapColorTableSeed = 'dpct',
    keySummaryMenubar = 'dsmb',
    keySummaryChanges = 'dsch',
    keyDisplayOldConfig = 'dold',
    keyDisplayNewConfig = 'dnew'
};
```

Constants

`kAESystemConfigNotice`

Keyword for the Event ID for a Display Notice event.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`kAEDisplayNotice`

Keyword for a required parameter to a Display Notice event.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`kAEDisplaySummary`

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`keyDMConfigVersion`

Keyword for the descriptor structure describing the version number for this Display Notice event.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`keyDMConfigFlags`
Reserved for future expansion. Internal use only.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`keyDMConfigReserved`
Reserved for future expansion. Internal use only.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`keyDisplayID`
Keyword for the descriptor structure describing the display ID for the video device.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`keyDisplayComponent`
Unless you are disconnecting display components, this is for internal use only.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`keyDisplayDevice`
Keyword for the descriptor structure containing a handle to the `GDevice` structure for the video device.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`keyDisplayFlags`
Reserved for future expansion. Internal use only.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`keyDisplayMode`
Keyword for the descriptor structure containing the `sResource` number from the video device for this display mode.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`keyDisplayModeReserved`
Reserved for future expansion. Internal use only.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`keyDisplayReserved`
Reserved for future expansion. Internal use only.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`keyDisplayMirroredId`
Keyword for the display this device is mirrored to.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`keyDeviceFlags`
: Keyword for the descriptor structure describing the attributes for the video device as maintained in the `gdFlags` field of the `GDevice` structure for the device.

  Available in Mac OS X v10.0 and later.

  Declared in `Displays.h`.

`keyDeviceDepthMode`
: Keyword for the descriptor structure describing the depth mode for the video device; that is, the value of the `gdMode` field in the `GDevice` structure for the device.

  Available in Mac OS X v10.0 and later.

  Declared in `Displays.h`.

`keyDeviceRect`
: Keyword for the descriptor structure describing the boundary rectangle of the video device; that is, the value of the `gdRect` field in the `GDevice` structure for the device.

  Available in Mac OS X v10.0 and later.

  Declared in `Displays.h`.

`keyPixMapRect`
: Keyword for the descriptor structure describing the boundary rectangle into which QuickDraw can draw; that is, the bounds field in the `PixMap` structure for the `GDevice` structure for the video device.

  Available in Mac OS X v10.0 and later.

  Declared in `Displays.h`.

`keyPixMapHResolution`
: Keyword for the descriptor structure describing the horizontal resolution of the pixel image in the `PixMap` structure for the `GDevice` structure for the video device.

  Available in Mac OS X v10.0 and later.

  Declared in `Displays.h`.

`keyPixMapVResolution`
: Keyword for the descriptor structure describing the vertical resolution of the pixel image in the `PixMap` structure for the `GDevice` structure for the video device.

  Available in Mac OS X v10.0 and later.

  Declared in `Displays.h`.

`keyPixMapPixelType`
: Keyword for the descriptor structure describing the storage format for the pixel image on the device; that is, the value of the `pixelType` field in the `PixMap` structure for the `GDevice` structure for the video device.

  Available in Mac OS X v10.0 and later.

  Declared in `Displays.h`.

`keyPixMapPixelSize`
: Keyword for the descriptor structure describing the pixel depth for the device; that is, the value of the `pixelSize` field in the `PixMap` structure for the `GDevice` structure for the video device.

  Available in Mac OS X v10.0 and later.

  Declared in `Displays.h`.

`keyPixMapCmpCount`
: Keyword for the descriptor structure containing the number of components used to represent a color for a pixel; that is, the value of the `cmpCount` field in the `PixMap` structure for the `GDevice` structure for the device.

  Available in Mac OS X v10.0 and later.

  Declared in `Displays.h`.

`keyPixMapCmpSize`
: Keyword for the descriptor structure describing the size in bits of each component for a pixel; that is, the value of the `cmpSize` field in the `PixMap` structure for the `GDevice` structure for the device.

  Available in Mac OS X v10.0 and later.

  Declared in `Displays.h`.

`keyPixMapAlignment`
: Reserved for future expansion. Internal use only.

  Available in Mac OS X v10.0 and later.

  Declared in `Displays.h`.

`keyPixMapResReserved`
: Reserved for future expansion. Internal use only.

  Available in Mac OS X v10.0 and later.

  Declared in `Displays.h`.

`keyPixMapReserved`
: Reserved for future expansion. Internal use only.

  Available in Mac OS X v10.0 and later.

  Declared in `Displays.h`.

`keyPixMapColorTableSeed`
: Keyword for the descriptor structure containing the value of the `ctSeed` field of the `ColorTable` structure for the `PixMap` structure for the `GDevice` structure for the video device.

  Available in Mac OS X v10.0 and later.

  Declared in `Displays.h`.

`keySummaryMenubar`
: Reserved for future expansion. Internal use only.

  Available in Mac OS X v10.0 and later.

  Declared in `Displays.h`.

`keySummaryChanges`
: Reserved for future expansion. Internal use only.

  Available in Mac OS X v10.0 and later.

  Declared in `Displays.h`.

`keyDisplayOldConfig`
: Keyword for the descriptor structure describing the video device's previous state.

  Available in Mac OS X v10.0 and later.

  Declared in `Displays.h`.

`keyDisplayNewConfig`

Keyword for the descriptor structure describing the video device's new state.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

Discussion

The Display Manager sends an Apple event—the Display Notice event—to notify applications that it has changed the display environment. The keywords that specify the Display Notice event and its descriptor structures are described here.

Confirm Flags

```
enum {
    kForceConfirmBit = 0,
    kForceConfirmMask = (1 << kForceConfirmBit)
};
```

Constants

`kForceConfirmBit`

Indicates to force a confirm dialog.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`kForceConfirmMask`

Use to set or test for a forced confirm dialog.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

Dependent Notification Constants

```
enum {
    kDependentNotifyClassShowCursor = 'shcr',
    kDependentNotifyClassDriverOverride = 'ndrv',
    kDependentNotifyClassDisplayMgrOverride = 'dmgr',
    kDependentNotifyClassProfileChanged = 'prof'
};
```

Constants

`kDependentNotifyClassShowCursor`

The Display Manager sends an extended notification when a hidden cursor shows during a display unmirror.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`kDependentNotifyClassDriverOverride`

The Display Manager sends notification that a video driver has been overridden with a newer revision.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`kDependentNotifyClassDisplayMgrOverride`
The Display Manager sends notification that it has been upgraded with a newer revision.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`kDependentNotifyClassProfileChanged`
The Display Manager sends notification when the profile associated with a display changes.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

Discussion
The function `DMSendDependentNotification` (page 81) contains the parameter `notifyClass` which you can specify with a Dependent Notification Constant.

Display/Device ID Constants

The Display Manager uses these values to help with the configuration of the display.

```
enum {
    kDummyDeviceID = 0x00FF,
    kInvalidDisplayID = 0x0000,
    kFirstDisplayID = 0x0100
};
```

Constants
`kDummyDeviceID`
This is the ID of the dummy display, used when the last “real” display is removed.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`kInvalidDisplayID`
This is the ID of the invalid display, which has been removed from the active display list.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`kFirstDisplayID`
When your application sets this bit it asks the Display Manager to return the ID of the first display device on the active display list.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

Display Gestalt Constants

```
enum {
    kDisplayGestaltDisplayCommunicationAttr = 'comm',
    kDisplayGestaltForbidI2CMask = (1 << 0),
    kDisplayGestaltUseI2CPowerMask = (1 << 1),
    kDisplayGestaltCalibratorAttr = 'cali',
    kDisplayGestaltBrightnessAffectsGammaMask = (1 << 0),
    kDisplayGestaltViewAngleAffectsGammaMask = (1 << 1)
};
```

Display Mode Flags

The structure `DMDisplayModeListEntryRec` uses these values for its `displayModeFlags` field.

```
enum {
    kDisplayModeListNotPreferredBit = 0,
    kDisplayModeListNotPreferredMask = (1 << kDisplayModeListNotPreferredBit)
};
```

Constants

`kDisplayModeListNotPreferredBit`

Indicates there is a better timing available and that this timing should be shown only if the user wants to see all options.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`kDisplayModeListNotPreferredMask`

(1 `kDisplayModeListNotPreferredBit`)

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

Display Version Values

```
enum {
    kDisplayTimingInfoVersionZero = 1,
    kDisplayTimingInfoReservedCountVersionZero = 16,
    kDisplayModeEntryVersionZero = 0,
    kDisplayModeEntryVersionOne = 1
};
```

Constants

`kDisplayTimingInfoVersionZero`

This relative information is always `NULL` in this version.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`kDisplayTimingInfoReservedCountVersionZero`

This relative information is always `NULL` in this version.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`kDisplayModeEntryVersionZero`
This relative information is always `NULL` in this version.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`kDisplayModeEntryVersionOne`
This relative information is always `NULL` in this version.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

Discussion
These values supply information to the structure `DMDisplayModeListEntryRec` (page 23).

Fidelity Check Constants

```
enum {
    kNoFidelity = 0,
    kMinimumFidelity = 1,
    kDefaultFidelity = 500,
    kDefaultManufacturerFidelity = 1000
};
```

Get Name By AVID Mask

```
enum {
    kDMSupressNumbersMask = (1 << 0),
    kDMForceNumbersMask = (1 << 1),
    kDMSupressNameMask = (1 << 2)
};
```

Constants
`kDMSupressNumbersMask`
If the bit specified by this mask is set, the numbers are suppressed and only names are returned.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`kDMForceNumbersMask`
If the bit specified by this mask is set, the numbers are forced to always be shown–even on single display configs.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`kDMSupressNameMask`
If the bit specified by this mask is set, the names are suppressed and only numbers are returned.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

Include Masks

```
enum {
    kIncludeOnlineActiveDisplaysMask = (1 << 0),
    kIncludeOnlineDisabledDisplaysMask = (1 << 1),
    kIncludeOfflineDisplaysMask = (1 << 2),
    kIncludeOfflineDummyDisplaysMask = (1 << 3),
    kIncludeHardwareMirroredDisplaysMask = (1 << 4)
};
```

Item Flags

```
enum {
    kComponentListNotPreferredBit = 0,
    kComponentListNotPreferredMask = (1 << kComponentListNotPreferredBit)
};
```

Mode List Masks

```
enum {
    kDMModeListIncludeAllModesMask = (1 << 0),
    kDMModeListIncludeOfflineModesMask = (1 << 1),
    kDMModeListExcludeDriverModesMask = (1 << 2),
    kDMModeListExcludeDisplayModesMask = (1 << 3),
    kDMModeListExcludeCustomModesMask = (1 << 4),
    kDMModeListPreferStretchedModesMask = (1 << 5),
    kDMModeListPreferSafeModesMask = (1 << 6)
};
```

Constants

`kDMModeListIncludeAllModesMask`
Available in Mac OS X v10.0 and later.
Declared in `Displays.h`.

`kDMModeListIncludeOfflineModesMask`
Available in Mac OS X v10.0 and later.
Declared in `Displays.h`.

`kDMModeListExcludeDriverModesMask`
Available in Mac OS X v10.0 and later.
Declared in `Displays.h`.

`kDMModeListExcludeDisplayModesMask`
Available in Mac OS X v10.0 and later.
Declared in `Displays.h`.

`kDMModeListExcludeCustomModesMask`
Available in Mac OS X v10.0 and later.
Declared in `Displays.h`.

`kDMModeListPreferStretchedModesMask`

Prefer modes that are stretched over modes that are letterboxed when setting kDisplayModeListNotPreferredBit

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`kDMModeListPreferSafeModesMask`

Prefer modes that are safe over modes that are not when setting kDisplayModeListNotPreferredBit

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

Name Flags

```
enum {
    kSuppressNumberBit = 0,
    kSuppressNumberMask = 1,
    kForceNumberBit = 1,
    kForceNumberMask = 2,
    kSuppressNameBit = 2,
    kSuppressNameMask = 4
};
```

New Engine List Constants

```
enum {
    kAnyPanelType = 0,
    kAnyEngineType = 0,
    kAnyDeviceType = 0,
    kAnyPortType = 0
};
```

Notification Messages

```
enum {
    kDMNotifyRequestConnectionProbe = 0,
    kDMNotifyInstalled = 1,
    kDMNotifyEvent = 2,
    kDMNotifyRemoved = 3,
    kDMNotifyPrep = 4,
    kDMNotifyExtendEvent = 5,
    kDMNotifyDependents = 6,
    kDMNotifySuspendConfigure = 7,
    kDMNotifyResumeConfigure = 8,
    kDMNotifyRequestDisplayProbe = 9,
    kDMNotifyDisplayWillSleep = 10,
    kDMNotifyDisplayDidWake = 11,
    kExtendedNotificationProc = (1L << 16)
};
```

Constants

`kDMNotifyRequestConnectionProbe`

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`kDMNotifyInstalled`

The Display Manager provides this message during a callback function to if your application has installed an extended notification procedure pointer for the first time. The Display Manager provides this message in the `notifyData` parameter of `DMSendDependentNotification` (page 81).

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`kDMNotifyEvent`
The Display Manager provides this message when an Apple event update occurs, after a display configuration change is made. This is the only time non-extended notifications are called.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`kDMNotifyRemoved`
The Display Manager provides this message when the function `DMSendDependentNotification` (page 81) is called on your function.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`kDMNotifyPrep`
Before passing `kDMSNotifyRemoved`, the Display Manager provides this message to indicate that it is about to begin to configure. Calling `DMSendDependentNotification` (page 81) tells the Display Manager to send this message.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`kDMNotifyExtendEvent`
Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`kDMNotifyDependents`
The Display Manager provides this message to `DMSendDependentNotification` (page 81).

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`kDMNotifySuspendConfigure`
The Display Manager passes this selector to notify your UPP that configuration is temporarily suspended. For instance, if a video game makes a temporary change to the display configuration, the game is expected to resume configuration and restore video before allowing other applications to access the screen.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`kDMNotifyResumeConfigure`
The Display Manager passes this selector to notify your application when previously suspended configuration is resumed. Your application can then replace windows and icons, and change depth mode if necessary.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`kDMNotifyRequestDisplayProbe`
Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`kDMNotifyDisplayWillSleep`
This selector is only available in Mac OS X.

Available in Mac OS X v10.2 and later.

Declared in `Displays.h`.

`kDMNotifyDisplayDidWake`
This selector is only available in Mac OS X.

Available in Mac OS X v10.2 and later.

Declared in `Displays.h`.

`kExtendedNotificationProc`
Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

Discussion
Display Manager functions needed for dependency notification and event processing use the notification message selectors in extended application-defined functions. `DMRegisterExtendedNotifyProc` (page 76) gets all these messages. Applications should update all information about the display configurations at this point.

Notification Types

```
enum {
    kFullNotify = 0,
    kFullDependencyNotify = 1
};
```

Constants
`kFullNotify`
The Display Manager sets this bit to provide the major Apple notification event.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`kFullDependencyNotify`
The Display Manager sets this bit to provide notification only to those applications that need to know about interrelated functionality. It is used for updating the user interface.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

Discussion
The function `DMSendDependentNotification` (page 81) uses these values in the `notifyType` parameter.

Panel List Flags

```
enum {
    kAllowDuplicatesBit = 0
};
```

Port List Flags

```
enum {
    kPLIncludeOfflineDevicesBit = 0
};
```

Constants

`kPLIncludeOfflineDevicesBit`

Should offline devices be put into the port list (such as dummy display)

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

Reserved Count Constants

```
enum {
    kMakeAndModelReservedCount = 4
};
```

Constants

`kMakeAndModelReservedCount`

Indicates the number of reserved fields.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

Summary Change Flags

```
enum {
    kBeginEndConfigureBit = 0,
    kMovedDisplayBit = 1,
    kSetMainDisplayBit = 2,
    kSetDisplayModeBit = 3,
    kAddDisplayBit = 4,
    kRemoveDisplayBit = 5,
    kNewDisplayBit = 6,
    kDisposeDisplayBit = 7,
    kEnabledDisplayBit = 8,
    kDisabledDisplayBit = 9,
    kMirrorDisplayBit = 10,
    kUnMirrorDisplayBit = 11
};
```

Switch Flags

```
enum {
    kNoSwitchConfirmBit = 0,
    kDepthNotAvailableBit = 1,
    kShowModeBit = 3,
    kModeNotResizeBit = 4,
    kNeverShowModeBit = 5
};
```

Constants

`kNoSwitchConfirmBit`

If the Display Manager sets this bit the display mode is required to function correctly. Your application does not need to provide confirmation if the user switches to this mode.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`kDepthNotAvailableBit`

If the Display Manager sets this bit the pixel depth of the specified device is not available for the specified display mode.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`kShowModeBit`

If the Display Manager sets this bit your application should display this mode to the user, even though it may require confirmation.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`kModeNotResizeBit`

If the Display Manager sets this bit you should not use this mode to resize a display; this mode drives a different connector in cards than in a built-in display.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

`kNeverShowModeBit`

If the Display Manager sets this bit you should not show the mode in the user interface.

Available in Mac OS X v10.0 and later.

Declared in `Displays.h`.

Discussion

In its `switchFlags` parameter, the function `DMCheckDisplayMode` (page 52) returns a pointer to a long integer that specifies flags in two of its bits. The constants represent bits that are set to 1. These bits are set by the Display Manager, not your application

Result Codes

The table below lists the result codes that are specific to the Display Manager.

Result Code	Value	Description
`kDMGenErr`	-6220	An indeterminate error occurred. Available in Mac OS X v10.0 and later.
`kDMMirroringOnAlready`	-6221	Video mirroring is already enabled. Available in Mac OS X v10.0 and later.
`kDMWrongNumberOfDisplays`	-6222	Wrong number of displays. Available in Mac OS X v10.0 and later.
`kDMMirroringBlocked`	-6223	Video is blocked. Available in Mac OS X v10.0 and later.
`kDMCantBlock`	-6224	Video mirroring is already enabled and can't be blocked; use `DMUnMirrorDevice`, then call `DMBlockMirroring` again. Available in Mac OS X v10.0 and later.
`kDMMirroringNotOn`	-6225	Video mirroring is not currently enabled. Available in Mac OS X v10.0 and later.
`kSysSWTooOld`	-6226	Some piece of system software is too old for the Display Manager to operate. Available in Mac OS X v10.0 and later.
`kDMSWNotInitializedErr`	-6227	The required pieces of system software are not initialized. Available in Mac OS X v10.0 and later.
`kDMDriverNotDisplayMgrAwareErr`	-6228	The video driver for the display does not support the Display Manager. Available in Mac OS X v10.0 and later.

Result Code	Value	Description
`kDMNotFoundErr`	-6229	Available in Mac OS X v10.0 and later.
`kDMDisplayNotFoundErr`	-6229	There are no `GDevice` structures for displays in the device list. Available in Mac OS X v10.0 and later.
`kDMDisplayAlreadyInstalledErr`	-6230	The display is already in the device list and can't be added. Available in Mac OS X v10.0 and later.
`kDMNoDeviceTableclothErr`	-6231	Available in Mac OS X v10.0 and later.
`kDMMainDisplayCannotMoveErr`	-6231	Available in Mac OS X v10.0 and later.
`kDMFoundErr`	-6232	Item found Available in Mac OS X v10.0 and later.

Gestalt Constants

You can check for version and feature availability information by using the Display Manager Version selectors defined in the Gestalt Manager. For more information, see *Gestalt Manager Reference*.

Deprecated Display Manager Reference (Not Recommended) Functions

A function identified as deprecated has been superseded and may become unsupported in the future.

Deprecated in Mac OS X v10.4

DisposeDMComponentListIteratorUPP

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
void DisposeDMComponentListIteratorUPP (
   DMComponentListIteratorUPP userUPP
);
```

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.

Declared In
`Displays.h`

DisposeDMDisplayListIteratorUPP

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
void DisposeDMDisplayListIteratorUPP (
   DMDisplayListIteratorUPP userUPP
);
```

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.

Declared In
`Displays.h`

DisposeDMDisplayModeListIteratorUPP

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
void DisposeDMDisplayModeListIteratorUPP (
   DMDisplayModeListIteratorUPP userUPP
);
```

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.

Declared In
`Displays.h`

DisposeDMExtendedNotificationUPP

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
void DisposeDMExtendedNotificationUPP (
   DMExtendedNotificationUPP userUPP
);
```

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.

Declared In
`Displays.h`

DisposeDMNotificationUPP

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
void DisposeDMNotificationUPP (
   DMNotificationUPP userUPP
);
```

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.

Declared In
`Displays.h`

DisposeDMProfileListIteratorUPP

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
void DisposeDMProfileListIteratorUPP (
   DMProfileListIteratorUPP userUPP
);
```

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.

Declared In
Displays.h

DMAddDisplay

Adds the GDevice structure for a video device to the device list. (Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMAddDisplay (
   GDHandle newDevice,
   short driver,
   UInt32 mode,
   UInt32 reserved,
   UInt32 displayID,
   Component displayComponent,
   Handle displayState
);
```

Parameters

newDevice

A handle to the GDevice structure for the video device you want to add to the device list. The function DMNewDisplay (page 74) usually initializes this structure.

driver

The reference number of the graphics device which you are adding to the device list. For most video devices, this information is set at system startup. The function DMAddDisplay passes the number supplied in this parameter to the InitGDevice function in its gdRefNum parameter.

mode

The depth mode. Used by the video device driver, this value sets the pixel depth and specifies color. The function DMAddDisplay passes the value supplied here to the function InitGDevice in its mode parameter.

reserved

Reserved for future expansion. Pass NULL in this parameter.

displayID

A unique identification for the display. For new displays, supply this parameter with the value 0, which causes the Display Manager to generate a unique display ID for this device. If this display was removed, then pass the display ID number of the current display in this parameter.

displayComponent

Reserved for future expansion. Pass NULL in this parameter.

displayState

If your application called DMNewDisplay (page 74), you must pass the displayState handle obtained. Otherwise pass NULL in this parameter.

Return Value
A result code. See "Display Manager Result Codes" (page 44).

Discussion
The DMAddDisplay function adds the display specified by the newDevice parameter as inactive. However, if the specified display is the only display, the Display Manager automatically makes it active. Otherwise, you must call the function DMEnableDisplay (page 57) to make the specified display active.

The function DMNewDisplay (page 74) automatically calls DMAddDisplay and DMEnableDisplay. The only time you ned to call DMAddDisplay directly is after the device has been removed by DMRemoveDisplay (page 78) but not yet disposed of by DMDisposeDisplay (page 55).

Special Considerations

Because this function may move or purge memory blocks or access handles, you cannot call it at interrupt time.

Generally, your application should not use this function, but should instead allow system software to maintain the device list. This function is described here for completeness only.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

Displays.h

DMBeginConfigureDisplays

Allows your application to configure displays. You should generally never need to use this function. (Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMBeginConfigureDisplays (
   Handle *displayState
);
```

Parameters

displayState

On return, a pointer to a handle to internal Display Manager information about the current display state. The DMEndConfigureDisplays (page 58) function and many other functions require this parameter.

Return Value

A result code. See "Display Manager Result Codes" (page 44).

Discussion

The DMBeginConfigureDisplays function tells the Display Manager to postpone Display Manager configuration checking, the rebuilding of desktop regions, and Apple event notification of Display Manager changes until your application uses the DMEndConfigureDisplays function.

You should call the function DMBeginConfigureDisplays before calling other Display Manager functions that configure the user's display. When calling functions that configure displays, you should pass the handle obtained by the DMBeginConfigureDisplays function. DMBeginConfigureDisplays causes system software to wait for your application to complete display changes before managing additional Display Manager events. When your application completes configuring the display environment, call the function DMEndConfigureDisplays.

Special Considerations

Because this function may move or purge memory blocks or access handles, you cannot call it at interrupt time.

Applications generally never need to use this function. In case you find a compelling need to change the user's display configuration, this function is described here for completeness. Note that if your application uses Display Manager functions to change the display configuration of the user's video devices, your application should make these changes only with the consent of the user. If your application must have a specific pixel depth, for example, it should display a dialog box that offers the user a choice between changing to that depth or canceling display of the image.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Displays.h`

DMBlockMirroring

Disables video mirroring. You should generally never need to use this function. (Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMBlockMirroring (
   void
);
```

Parameters

Return Value
A result code. See "Display Manager Result Codes" (page 44).

Discussion
The function `DMBlockMirroring` disables video mirroring until the user restarts the computer or until an application calls the function `DMUnblockMirroring` (page 85).

Special Considerations

Because this function may move or purge memory blocks or access handles, you cannot call it at interrupt time.

Applications generally never need to use this function. In case you find a compelling need to change the user's display configuration, this function is described here for completeness. Note that if your application uses Display Manager functions to change the display configuration of the user's video devices, your application should make these changes only with the consent of the user. If your application must have a specific pixel depth, for example, it should display a dialog box that offers the user a choice between changing to that depth or canceling display of the image.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Displays.h`

DMCanMirrorNow

Determines whether video mirroring can be activated on the user's computer system. (Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMCanMirrorNow (
   Boolean *canMirrorNow
);
```

Parameters

canMirrorNow

A pointer to a Boolean value; true indicates that mirroring can be activated; false indicates it cannot.

Return Value

A result code. See "Display Manager Result Codes" (page 44).

Discussion

In the value pointed to by the `canMirrorNow` parameter, the `DMCanMirrorNow` function reports whether video mirroring can be activated. When the `canMirrorNow` parameter points to a value of true, then the computer uses a version of QuickDraw that supports video mirroring, has exactly two displays attached, and does not have mirror blocking in effect.

You can use the `DMQDIsMirroringCapable` (page 76) function to determine whether the computer uses a version of QuickDraw that supports video mirroring. You can use the `DMBlockMirroring` (page 51) function and the `DMUnblockMirroring` (page 85) function to block and unblock video mirroring. To determine whether the user's computer system currently uses video mirroring, use the `DMIsMirroringOn` (page 69) function.

Special Considerations

The `DMCanMirrorNow` function may move or purge memory blocks in the application heap. Your application should not call this function at interrupt time.

Version Notes

As of System Software version 7.5, only PowerBook computers support video mirroring.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

`Displays.h`

DMCheckDisplayMode

Determines if a video device supports a particular display mode and pixel depth. (Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMCheckDisplayMode (
   GDHandle theDevice,
   UInt32 mode,
   UInt32 depthMode,
   UInt32 *switchFlags,
   UInt32 reserved,
   Boolean *modeOk
);
```

Parameters

theDevice

A handle to the `GDevice` structure for the video device whose display mode and pixel depth you wish to check.

mode

The display mode you wish to check. You get a list of display modes by calling `DMGetDisplayMode` (page 62).

depthMode

The pixel depth you wish to check. See "Video Depth Mode Values" for list of possible values.

switchFlags

On return, a pointer to a long integer that indicates if a video device will support the mode specified by the `mode` parameter and the pixel depth specified by the `depthMode` parameter. See "Switch Flags" (page 43) for a description.

reserved

Reserved for future expansion. Pass `NULL` in this parameter.

modeOk

On return, a pointer to a `Boolean`. If `modeOk` points to a value of `true`, the user or your application can switch the display mode for the video device to the one specified by `mode`.

Return Value
A result code. See "Display Manager Result Codes" (page 44).

Discussion
Usually, your application only needs to know if a video device supports a specific pixel depth. Thus your application can use the Color QuickDraw function `HasDepth`. The function `DMCheckDisplayMode` is essentially obsolete, and is here for completeness.

Special Considerations

Because this function may move or purge memory blocks or access handles, you cannot call it at interrupt time.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Displays.h`

DMConfirmConfiguration

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMConfirmConfiguration (
   DMModalFilterUPP filterProc,
   UInt32 confirmFlags,
   UInt32 reserved,
   Handle displayState
);
```

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Displays.h`

DMDisableDisplay

Makes a video device inactive by removing its display area from the desktop. You should generally never need to use this function. (Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMDisableDisplay (
   GDHandle disableDevice,
   Handle displayState
);
```

Parameters

`disableDevice`

A handle to the `GDevice` structure for the video device whose display you wish to disable.

`displayState`

If your application called `DMBeginConfigureDisplays` (page 50), you must pass the `displayState` handle obtained. Otherwise pass `NULL` in this parameter.

Return Value
A result code. See "Display Manager Result Codes" (page 44).

Discussion
You are not allowed to disable the last remaining display. Doing so will simply re-enable it. If you want to remove the last remaining display, thereby enabling the `GDevice` structure not associated with any video device, call the function `DMRemoveDisplay` (page 78).

If you specify the device for the main screen in the `disableDevice` parameter, then `DMDisableDisplay` picks another device and makes it the new main screen.

If `DMDisableDisplay` results in setting a new main screen, the handle you pass in the `disableDevice` parameter does not point to the same `GDevice` structure after `DMDisableDisplay` completes; instead, it points to the `GDevice` structure for the new main screen. If you need to recover the `GDevice` structure for the device you disabled, determine its display ID by using the function `DMGetDisplayIDByGDevice` (page 61) before calling `DMDisableDisplay`. Then use the function `DMGetGDeviceByDisplayID` (page 63) to obtain its structure.

Special Considerations

Because this function may move or purge memory blocks or access handles, you cannot call it at interrupt time.

Applications generally never need to use this function. In case you find a compelling need to change the user's display configuration, this function is described here for completeness. Note that if your application uses Display Manager functions to change the display configuration of the user's video devices, your application should make these changes only with the consent of the user. If your application must have a specific pixel depth, for example, it should display a dialog box that offers the user a choice between changing to that depth or canceling display of the image.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Displays.h`

DMDisposeAVComponent

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMDisposeAVComponent (
   Component theAVComponent
);
```

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Displays.h`

DMDisposeDisplay

Disposes of the `GDevice` structure for a video device. (Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMDisposeDisplay (
   GDHandle disposeDevice,
   Handle displayState
);
```

Parameters

`disposeDevice`
A handle to the `GDevice` structure for a video device you want to delete.

`displayState`
If your application called `DMBeginConfigureDisplays` (page 50), you must pass the `displayState` handle obtained. Otherwise pass `NULL` in this parameter.

Return Value
A result code. See "Display Manager Result Codes" (page 44).

Discussion
The `DMDisposeDisplay` function disposes of a `GDevice` structure, releases the space allocated for it, and disposes of all the data structures allocated for it. The Display Manager calls this function when appropriate.

Special Considerations

Because this function may move or purge memory blocks or access handles, you cannot call it at interrupt time.

Generally, your application should not use this function, but should instead allow system software to maintain the device list. This function is described here for completeness only.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Displays.h`

DMDisposeList

Disposes of a display mode list built by `DMNewDisplayModeList`. (Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMDisposeList (
   DMListType panelList
);
```

Parameters

`panelList`

A value that specifies the display mode list you want to delete.

Return Value
A result code. See "Display Manager Result Codes" (page 44).

Discussion
You should call the `DMDisposeList` function after you have iterated the mode list.

Special Considerations

Because this function may move or purge memory blocks or access handles, you cannot call it at interrupt time.

Generally, your application should not use this function, but should instead allow system software to maintain the device list. This function is described here for completeness only.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Displays.h`

DMDrawDesktopRect

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
void DMDrawDesktopRect (
   Rect *globalRect
);
```

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Displays.h`

DMDrawDesktopRegion

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
void DMDrawDesktopRegion (
   RgnHandle globalRgn
);
```

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Displays.h`

DMEnableDisplay

Reactivates a display made inactive with the function `DMDisableDisplay`. You should generally never need to use this function. (Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMEnableDisplay (
   GDHandle enableDevice,
   Handle displayState
);
```

Parameters

`enableDevice`
A handle to the `GDevice` structure for the video device whose display you wish to make active.

`displayState`
If your application called `DMBeginConfigureDisplays` (page 50), you must pass the `displayState` handle obtained. Otherwise pass `NULL` in this parameter.

Return Value
A result code. See "Display Manager Result Codes" (page 44).

Discussion
The function `DMEnableDisplay` reactivates the specified video device by adding its display area to the desktop.

If you add a display with the function `DMAddDisplay` (page 49) and there are no active displays, the Display Manager will enable the added display.

Special Considerations

Because this function may move or purge memory blocks or access handles, you cannot call it at interrupt time.

Applications generally never need to use this function. In case you find a compelling need to change the user's display configuration, this function is described here for completeness. Note that if your application uses Display Manager functions to change the display configuration of the user's video devices, your application should make these changes only with the consent of the user. If your application must have a specific pixel depth, for example, it should display a dialog box that offers the user a choice between changing to that depth or canceling display of the image.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Displays.h`

DMEndConfigureDisplays

Ends configuration begun by `DMBeginConfigureDisplays`. You should generally never need to use this function. (Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMEndConfigureDisplays (
   Handle displayState
);
```

Parameters

displayState
Supply this parameter with the handle obtained by the `DMBeginConfigureDisplays` (page 50) function.

Return Value
A result code. See "Display Manager Result Codes" (page 44).

Discussion
The function `DMEndConfigureDisplays` resumes Display Manager configuration checking, the rebuilding of desktop regions, and Apple event notification of Display Manager changes, all of which are postponed when you use the function `DMBeginConfigureDisplays` (page 50). Your application will then receive a single Display Notice event notifying your application of Display Manager changes, and your application can manage its windows accordingly.

Special Considerations

Because this function may move or purge memory blocks or access handles, you cannot call it at interrupt time.

Applications generally never need to use this function. In case you find a compelling need to change the user's display configuration, this function is described here for completeness. Note that if your application uses Display Manager functions to change the display configuration of the user's video devices, your application should make these changes only with the consent of the user. If your application must have a specific pixel depth, for example, it should display a dialog box that offers the user a choice between changing to that depth or canceling display of the image.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Displays.h`

DMGetAVPowerState

Obtains the current power state of a display. (Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMGetAVPowerState (
   AVIDType theID,
   AVPowerStatePtr getPowerState,
   UInt32 reserved1
);
```

Parameters

theID
The ID number of the display device whose power state you want to obtain.

getPowerState
A pointer to a structure of type `AVPowerStateRec` (page 17). On return, this parameter points to a value specifying the current power state of display device.

reserved1
Reserved for future expansion. Pass NULL in this parameter.

Return Value
A result code. See "Display Manager Result Codes" (page 44).

Special Considerations

Because this function may move or purge memory blocks or access handles, you cannot call it at interrupt time.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Displays.h`

DMGetDeskRegion

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMGetDeskRegion (
   RgnHandle *desktopRegion
);
```

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Displays.h`

DMGetDeviceAVIDByPortAVID

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMGetDeviceAVIDByPortAVID (
   AVIDType portAVID,
   AVIDType *deviceAVID
);
```

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Displays.h`

DMGetDeviceComponentByAVID

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMGetDeviceComponentByAVID (
   AVIDType theDeviceID,
   Component *theDeviceComponent,
   ComponentDescription *theDesciption,
   ResType *theDeviceKind
);
```

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Displays.h`

DMGetDisplayComponent

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMGetDisplayComponent (
   GDHandle theDevice,
   Component *displayComponent
);
```

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

`Displays.h`

DMGetDisplayIDByGDevice

Obtains the display ID number for a video device. (Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMGetDisplayIDByGDevice (
   GDHandle displayDevice,
   DisplayIDType *displayID,
   Boolean failToMain
);
```

Parameters

`displayDevice`

A handle to the `GDevice` structure for the video device whose display ID you wish to obtain.

`displayID`

On return, a pointer to the display ID for the video device specified by the `displayDevice` parameter.

`failToMain`

If `true` and the specified video device does not have a display ID, on return the function sets the `displayID` parameter to a pointer to the display ID of the video device for the main screen. If `false` and the specified video device does not have a display ID, the function returns the `kDMDisplayNotFoundErr` result code.

Return Value

A result code. See "Display Manager Result Codes" (page 44).

Special Considerations

Because this function may move or purge memory blocks or access handles, you cannot call it at interrupt time.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

`Displays.h`

DMGetDisplayMode

Obtains the current display mode of a specified video display. (Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMGetDisplayMode (
   GDHandle theDevice,
   VDSwitchInfoPtr switchInfo
);
```

Parameters

theDevice

A handle to the `GDevice` structure for the video device whose display mode you wish to obtain.

switchInfo

On return, a pointer to an internal Display Manager structure containing display mode information.

Return Value

A result code. See "Display Manager Result Codes" (page 44).

Special Considerations

Because this function may move or purge memory blocks or access handles, you cannot call it at interrupt time.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

`Displays.h`

DMGetEnableByAVID

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMGetEnableByAVID (
   AVIDType theAVID,
   Boolean *isAVIDEnabledNow,
   Boolean *canChangeEnableNow
);
```

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

`Displays.h`

DMGetFirstScreenDevice

Returns a handle for the first video device in the device list. (Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
GDHandle DMGetFirstScreenDevice (
   Boolean activeOnly
);
```

Parameters

activeOnly

If `true`, the `DMGetFirstScreenDevice` function returns a handle to the first of all active video devices. If `false`, the function returns a handle to the first of all video devices, active or not. You may use the Active Device Constants in this parameter. See "Active Device Only Values" (page 28).

Return Value

If `activeOnly` is `true`, a handle to the `GDevice` structure for the first active video device. If `activeOnly` is `false`, a handle to the `GDevice` structure for the first video device. See the QuickDraw Manager documentation for a description of the `GDHandle` data type.

Discussion

The `DMGetFirstScreenDevice` function is useful if you want to find out more about the current mode.

You can use the function `DMGetNextScreenDevice` (page 67) to loop through all of the video devices in the device list.

The `DMGetFirstScreenDevice` function is similar to the QuickDraw function `GetDeviceList`, except that when returning `GDevice` structures, `GetDeviceList` does not distinguish between inactive and active video devices or between the `GDevice` structures for video devices and the `GDevice` structures associated with no video devices.

Special Considerations

Because this function may move or purge memory blocks or access handles, you cannot call it at interrupt time.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

`Displays.h`

DMGetGDeviceByDisplayID

Obtains a handle for the video device with a specified display ID. (Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMGetGDeviceByDisplayID (
   DisplayIDType displayID,
   GDHandle *displayDevice,
   Boolean failToMain
);
```

Parameters

displayID

The display ID for the video device whose handle you wish to obtain.

displayDevice

On return, a pointer to the handle to the `GDevice` structure for the video device specified by the `displayID` parameter.

failToMain

If `true` and there is no video device associated with the `displayID` parameter, on return the function sets `displayDevice` to a pointer to the handle for the video device for the main screen. If `false` and there is no video device associated with the `displayID` parameter, the function returns the `kDMDisplayNotFoundErr` result code.

Return Value
A result code. See "Display Manager Result Codes" (page 44).

Special Considerations

Because this function may move or purge memory blocks or access handles, you cannot call it at interrupt time.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Related Sample Code
Simple DrawSprocket

Declared In
`Displays.h`

DMGetGraphicInfoByAVID

Obtains information about the graphic display of a display device. (Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMGetGraphicInfoByAVID (
   AVIDType theID,
   PicHandle *theAVPcit,
   Handle *theAVIconSuite,
   AVLocationRec *theAVLocation
);
```

Parameters

theID

The ID number of the display device whose information you want to obtain.

theAVPcit

On return, a pointer to the handle for the picture structure you want to get.

theAVIconSuite

On return, a pointer to a handle whose structure reports the icon suite for a display device.

theAVLocation

On return, a pointer to the location structure for the device you want information about.

Return Value
A result code. See "Display Manager Result Codes" (page 44).

Special Considerations

Because this function may move or purge memory blocks or access handles, you cannot call it at interrupt time.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

`Displays.h`

DMGetIndexedComponentFromList

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMGetIndexedComponentFromList (
   DMListType panelList,
   DMListIndexType itemIndex,
   UInt32 reserved,
   DMComponentListIteratorUPP listIterator,
   void *userData
);
```

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

`Displays.h`

DMGetIndexedDisplayModeFromList

Obtains a display mode from the display mode list built by `DMNewDisplayModeList`. (Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMGetIndexedDisplayModeFromList (
   DMListType panelList,
   DMListIndexType itemIndex,
   UInt32 reserved,
   DMDisplayModeListIteratorUPP listIterator,
   void *userData
);
```

Parameters

panelList

A value that specifies the list from which to obtain information about the display modes created by the function `DMNewDisplayModeList` (page 75).

itemIndex

A value that specifies the index of the display mode you wish to obtain.

reserved
Reserved for future expansion. Pass `NULL` in this parameter.

listIterator
A universal procedure pointer. The iterator this pointer specifies supplies the function to be called with the information about the display mode specified by `theListCount`.

userData
A pointer you pass for `listIterator` usually used to obtain information about the display mode from the UPP and return it to the caller of `DMGetIndexedDisplayModeFromList`.

Return Value
A result code. See "Display Manager Result Codes" (page 44).

Special Considerations

Because this function may move or purge memory blocks or access handles, you cannot call it at interrupt time.

Generally, your application should not use this function, but should instead allow system software to maintain the device list. This function is described here for completeness only.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Displays.h`

DMGetNameByAVID

Obtains the name of a display device. (Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMGetNameByAVID (
   AVIDType theID,
   UInt32 nameFlags,
   Str255 name
);
```

Parameters

theID
The ID number of the display device whose name you want to obtain.

nameFlags
Reserved for future expansion. Pass `NULL` in this parameter.

name
On return, a string containing the name of the display device specified by the parameter `theID`.

Return Value
A result code. See "Display Manager Result Codes" (page 44).

Discussion
An AVID is really a display ID as an AVID references a video display just like a display ID. Developers planned to use AVIDs for an extended set of devices, however, they never did this.

Special Considerations

Because this function may move or purge memory blocks or access handles, you cannot call it at interrupt time.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

`Displays.h`

DMGetNextMirroredDevice

Obtains a handle for a video device that mirrors another specified video device. (Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMGetNextMirroredDevice (
   GDHandle gDevice,
   GDHandle *mirroredDevice
);
```

Parameters

gDevice

A handle to the `GDevice` structure for the video device that another video device mirrors.

mirroredDevice

On return, a pointer to the handle for the video device that displays a mirror image of the device specified in the `gDevice` parameter.

Return Value

A result code. See "Display Manager Result Codes" (page 44).

Special Considerations

Because this function may move or purge memory blocks or access handles, you cannot call it at interrupt time.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

`Displays.h`

DMGetNextScreenDevice

Returns a handle for the next video device in the device list. (Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
GDHandle DMGetNextScreenDevice (
   GDHandle theDevice,
   Boolean activeOnly
);
```

Parameters

theDevice

A handle to the GDevice structure at which you want the function to begin. You can supply the handle returned by the function DMGetFirstScreenDevice or DMGetNextScreenDevice.

activeOnly

If true, the DMGetNextScreenDevice function returns a handle for the next active video device. If false, DMGetNextScreenDevice returns a handle for the next video device, active or not. You may use the Active Device Constants in this parameter. See "Active Device Only Values" (page 28).

Return Value

If activeOnly is true, a handle to the next GDevice structure for an active video device. If activeOnly is false, a handle to the next GDevice structure for a video device. If there are no more GDevice structures in the list, DMGetNextScreenDevice returns NULL. See the QuickDraw Manager documentation for a description of the GDHandle data type.

Discussion

The DMGetNextScreenDevice function is similar to the QuickDraw function GetNextDevice, except that when returning GDevice structures, GetNextDevice does not distinguish between inactive and active video devices or between the GDevice structures for video devices and the GDevice structures associated with no video devices.

Special Considerations

Because this function may move or purge memory blocks or access handles, you cannot call it at interrupt time.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

Displays.h

DMGetPortComponentByAVID

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMGetPortComponentByAVID (
   DisplayIDType thePortID,
   Component *thePortComponent,
   ComponentDescription *theDesciption,
   ResType *thePortKind
);
```

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In
`Displays.h`

DMIsMirroringOn

Determines if video mirroring is active. (Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMIsMirroringOn (
   Boolean *isMirroringOn
);
```

Parameters

isMirroringOn

On return, a pointer to a Boolean value; `true` indicates that mirroring is on; `false` indicates it is not.

Return Value
A result code. See "Display Manager Result Codes" (page 44).

Special Considerations

Because this function may move or purge memory blocks or access handles, you cannot call it at interrupt time.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Displays.h`

DMMirrorDevices

Turns on video mirroring. (Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMMirrorDevices (
   GDHandle gD1,
   GDHandle gD2,
   Handle displayState
);
```

Parameters

gD1

A handle to the `GDevice` structure for the video device whose pixel image you want duplicated on another device.

gD2

A handle to the `GDevice` structure for the video device on which you want to duplicate the pixel image specified in the `gD1` parameter.

displayState

If your application called `DMBeginConfigureDisplays` (page 50), you must pass the `displayState` handle obtained. Otherwise pass `NULL` in this parameter.

Return Value
A result code. See "Display Manager Result Codes" (page 44).

Discussion
Your application should leave control of video mirroring to the user. However, if video mirroring is useful for your application (for example, if your application displays on-screen presentations), you might provide a control so that the user can switch to video mirroring directly from your application. In this case, `DMMirrorDevices` is useful to your application. Your control should also allow the user to turn video mirroring off; the function `DMUnmirrorDevice` (page 86) supports this.

Special Considerations

Because this function may move or purge memory blocks or access handles, you cannot call it at interrupt time.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In
`Displays.h`

DMMoveDisplay

Moves the boundary rectangle for a video device. You should generally never need to use this function. (Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMMoveDisplay (
   GDHandle moveDevice,
   short x,
   short y,
   Handle displayState
);
```

Parameters

moveDevice

A handle to the `GDevice` structure for the video device whose boundary rectangle you wish to move.

x

The horizontal coordinate on the QuickDraw global coordinate plane for the point to which you want to move the upper-left corner of the boundary rectangle.

y

The vertical coordinate on the QuickDraw global coordinate plane for the point to which you want to move the upper-left corner of the boundary rectangle.

displayState

If your application called `DMBeginConfigureDisplays` (page 50), you must pass the `displayState` handle obtained. Otherwise pass `NULL` in this parameter.

Return Value
A result code. See "Display Manager Result Codes" (page 44).

Discussion
The `DMMoveDisplay` function moves the boundary rectangle for the specified video device to the point (`x`,`y`) in the QuickDraw global coordinate plane. If the video device controls the main screen, which always has the global coordinates (`0`,`0`), then all other video devices are offset by horizontal distance `x` and vertical distance `y`.

A boundary rectangle is the rectangle that links the local coordinate system of a graphics port to QuickDraw's global coordinate system and defines the area of the pixel image or bit image into which QuickDraw can draw. The boundary rectangle is stored in either the pixel map or the bitmap contained in a `GDevice` structure.

The Display Manager will reposition overlapped or discontiguous boundary rects to create a non-overlapping contiguous desktop space.

Special Considerations

Because this function may move or purge memory blocks or access handles, you cannot call it at interrupt time.

Applications generally never need to use this function. In case you find a compelling need to change the user's display configuration, this function is described here for completeness. Note that if your application uses Display Manager functions to change the display configuration of the user's video devices, your application should make these changes only with the consent of the user. If your application must have a specific pixel depth, for example, it should display a dialog box that offers the user a choice between changing to that depth or canceling display of the image.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Displays.h`

DMNewAVDeviceList

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMNewAVDeviceList (
   ResType deviceType,
   UInt32 deviceListFlags,
   UInt32 reserved,
   DMListIndexType *deviceCount,
   DMListType *deviceList
);
```

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Displays.h`

DMNewAVEngineList

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMNewAVEngineList (
   DisplayIDType displayID,
   ResType engineType,
   DMFidelityType minimumFidelity,
   UInt32 engineListFlags,
   UInt32 reserved,
   DMListIndexType *engineCount,
   DMListType *engineList
);
```

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Displays.h`

DMNewAVIDByDeviceComponent

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMNewAVIDByDeviceComponent (
   Component theDeviceComponent,
   ResType portKind,
   UInt32 reserved,
   DisplayIDType *newID
);
```

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Displays.h`

DMNewAVIDByPortComponent

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMNewAVIDByPortComponent (
   Component thePortComponent,
   ResType portKind,
   UInt32 reserved,
   AVIDType *newID
);
```

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Displays.h`

DMNewAVPanelList

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMNewAVPanelList (
   DisplayIDType displayID,
   ResType panelType,
   DMFidelityType minimumFidelity,
   UInt32 panelListFlags,
   UInt32 reserved,
   DMListIndexType *thePanelCount,
   DMListType *thePanelList
);
```

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Displays.h`

DMNewAVPortListByDeviceAVID

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMNewAVPortListByDeviceAVID (
   AVIDType theID,
   DMFidelityType minimumFidelity,
   UInt32 portListFlags,
   UInt32 reserved,
   DMListIndexType *devicePortCount,
   DMListType *theDevicePortList
);
```

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Displays.h`

DMNewAVPortListByPortType

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMNewAVPortListByPortType (
   ResType subType,
   UInt32 portListFlags,
   UInt32 reserved,
   DMListIndexType *devicePortCount,
   DMListType *theDevicePortList
);
```

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Displays.h`

DMNewDisplay

Adds a video device to the device list and makes the device active. (Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMNewDisplay (
   GDHandle *newDevice,
   short driverRefNum,
   UInt32 mode,
   UInt32 reserved,
   DisplayIDType displayID,
   Component displayComponent,
   Handle displayState
);
```

Parameters

newDevice
A pointer to a handle to a `GDevice` structure for the video device that you want to add to the device list.

driverRefNum
The reference number of the video device which you are adding to the device list. This information is usually set at system startup. The function `DMAddDisplay` (page 49) passes the value supplied here to the `InitGDevice` function in its `gdRefNum` parameter.

mode
The depth mode. Used by the video device driver, this value sets the pixel depth and specifies color. The function `DMAddDisplay` (page 49) passes the value supplied here to the function `InitGDevice` in its `mode` parameter.

reserved
Reserved for future expansion. Pass `NULL` in this parameter.

displayID
A unique identification for the display. For new displays, supply this parameter with the value 0, which causes the Display Manager to generate a unique display ID for this device. If this display was removed, then pass the display ID of the current display in this parameter.

displayComponent
Reserved for future expansion. Pass `NULL` in this parameter.

displayState
If your application called DMAddDisplay (page 49), you must pass the displayState handle obtained. Otherwise pass NULL in this parameter.

Return Value
A result code. See "Display Manager Result Codes" (page 44).

Special Considerations

Because this function may move or purge memory blocks or access handles, you cannot call it at interrupt time.

Generally, your application should not use this function, but should instead allow system software to maintain the device list. This function is described here for completeness only.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
Displays.h

DMNewDisplayModeList

Builds a new display mode list for a specified video device. (Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMNewDisplayModeList (
   DisplayIDType displayID,
   UInt32 modeListFlags,
   UInt32 reserved,
   DMListIndexType *thePanelCount,
   DMListType *thePanelList
);
```

Parameters

displayID
The display ID for the video device that will have a new display mode list.

modeListFlags
Reserved for future expansion. Pass NULL in this parameter.

reserved
Reserved for future expansion. Pass NULL in this parameter.

thePanelCount
The number of entries in the display mode list specified by the theList parameter.

thePanelList
The display mode list for the specified video device.

Return Value
A result code. See "Display Manager Result Codes" (page 44).

Special Considerations

Because this function may move or purge memory blocks or access handles, you cannot call it at interrupt time.

Generally, your application should not use this function, but should instead allow system software to maintain the device list. This function is described here for completeness only.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Displays.h`

DMQDIsMirroringCapable

Determines if QuickDraw supports video mirroring on the user's system. (Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMQDIsMirroringCapable (
   Boolean *qdIsMirroringCapable
);
```

Parameters

qdIsMirroringCapable
On return, a pointer to the value `true` if QuickDraw supports video mirroring; otherwise, a pointer to the value `false`.

Return Value
A result code. See "Display Manager Result Codes" (page 44).

Special Considerations

Because this function may move or purge memory blocks or access handles, you cannot call it at interrupt time.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Displays.h`

DMRegisterExtendedNotifyProc

Registers a function that responds to a Display Notice event outside of an event loop. (Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMRegisterExtendedNotifyProc (
   DMExtendedNotificationUPP notifyProc,
   void *notifyUserData,
   unsigned short nofifyOnFlags,
   DMProcessInfoPtr whichPSN
);
```

Parameters

notifyProc

A pointer to your function that handles a Display Notice event.

notifyUserData

A pointer to caller-specific information which the Display Manager will return to your application when you request it.

notifyOnFlags

Reserved for future expansion. You should pass `kNilOptions` in this parameter.

whichPSN

A pointer to the Process Serial Number associated with your Display Notice event-handling function. If this process terminates, the Display Notice event-handling function is automatically removed. For example, the Monitors control panel supplies the Finder's process number when registering its Display Notice event-handling function.

Return Value

A result code. See "Display Manager Result Codes" (page 44).

Discussion

When the Display Manager sends your function the Display Notice event, your application or utility should respond by moving or resizing its windows and updating any internally-maintained video device information as appropriate.

When you are finished with your notification function, remove it by calling `DMRemoveExtendedNotifyProc` (page 79).

Special Considerations

Because this function may move or purge memory blocks or access handles, you cannot call it at interrupt time.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In

`Displays.h`

DMRegisterNotifyProc

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMRegisterNotifyProc (
   DMNotificationUPP notificationProc,
   DMProcessInfoPtr whichPSN
);
```

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Displays.h`

DMRemoveDisplay

Removes a video device from the device list. (Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMRemoveDisplay (
   GDHandle removeDevice,
   Handle displayState
);
```

Parameters

removeDevice

A handle to the `GDevice` structure for the video device you want to remove from the device list. The function `DMRemoveDisplay` does not actually dispose of this structure, but instead removes it from the device list.

displayState

If your application called `DMBeginConfigureDisplays` (page 50), you must pass the `displayState` handle obtained. Otherwise pass `NULL` in this parameter.

Return Value
A result code. See "Display Manager Result Codes" (page 44).

Discussion
The function `DMRemoveDisplay` may call the function `DMSetMainDisplay` (page 84), which causes the `removeDevice` parameter to contain a handle to the `GDevice` structure for the new main screen, not the video device whose handle was passed to `DMRemoveDisplay`. To recover the `GDevice` structure for the disabled device, determine its display ID by using the function `DMGetDisplayIDByGDevice` (page 61) before calling `DMRemoveDisplay`. Then use the function `DMGetGDeviceByDisplayID` (page 63) to obtain the `GDevice` structure for the specified device.

You are not allowed to disable the last remaining display using the `DMDisableDisplay` (page 54) function. Doing so will simply re-enable it. If you want to remove the last remaining display, thereby enabling the `GDevice` structure not associated with any video device, you must call `DMRemoveDisplay`.

Special Considerations

Because this function may move or purge memory blocks or access handles, you cannot call it at interrupt time.

Generally, your application should not use this function, but should instead allow system software to maintain the device list. This function is described here for completeness only.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Displays.h`

DMRemoveExtendedNotifyProc

Removes your Display Notice event-handling function registered by the `DMRegisterExtendedNotifyProc` function. (Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMRemoveExtendedNotifyProc (
   DMExtendedNotificationUPP notifyProc,
   void *notifyUserData,
   DMProcessInfoPtr whichPSN,
   unsigned short removeFlags
);
```

Parameters

notifyProc
A pointer to your function you want to remove that handles a Display Notice event.

notifyUserData
A pointer to caller-specific information which the Display Manager will return to your application when you request it.

whichPSN
A pointer to the Process Serial Number associated with your Display Notice event-handling function. If this process terminates, the Display Notice event-handling function is automatically removed. For example, the Monitors control panel supplies the Finder's process number when registering its Display Notice event-handling function.

removeFlags
Reserved for future expansion. You should pass `kNilOptions` in this parameter.

Return Value
A result code. See "Display Manager Result Codes" (page 44).

Special Considerations

Because this function may move or purge memory blocks or access handles, you cannot call it at interrupt time.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Displays.h`

DMRemoveNotifyProc

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMRemoveNotifyProc (
   DMNotificationUPP notificationProc,
   DMProcessInfoPtr whichPSN
);
```

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Displays.h`

DMResolveDisplayComponents

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMResolveDisplayComponents (
   void
);
```

Parameters

Return Value
A result code. See "Display Manager Result Codes" (page 44).

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Displays.h`

DMSaveScreenPrefs

Saves the user's screen configuration preferences. (Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMSaveScreenPrefs (
   UInt32 reserved1,
   UInt32 saveFlags,
   UInt32 reserved2
);
```

Parameters

reserved1
Reserved for future expansion. Pass `NULL` in this parameter.

saveFlags
Reserved for future expansion. Pass `NULL` in this parameter.

reserved2

Reserved for future expansion. Pass `NULL` in this parameter.

Return Value
A result code. See "Display Manager Result Codes" (page 44).

Discussion
Usually when you change screen properties such as pixel depth, the changes will only be temporary and will usually reset after restarting. However, the function `DMSaveScreenPrefs` makes the current screen properties permanent.

Special Considerations

Because this function may move or purge memory blocks or access handles, you cannot call it at interrupt time.

Availability
Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In
`Displays.h`

DMSendDependentNotification

Notifies dependent displays of changes in depth mode or configuration. (Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMSendDependentNotification (
   ResType notifyType,
   ResType notifyClass,
   AVIDType displayID,
   ComponentInstance notifyComponent
);
```

Parameters

notifyType

The resource type that identifies the engine that made the change. Examples might be component engines that control brightness, contrast, or screen size. You may pass zero in this parameter. See `DependentNotifyRec` (page 18) for more information.

notifyClass

The resource type that identifies the class of change the user or engine has made, such as color depth, pixel size, or screen size. See `DependentNotifyRec` (page 18) for more information.

displayID

The ID number of the dependent display which you want to notify of Display Manager events. On return, the Display Manager sets the `notifyPortID` constant of the `DependentNotifyRec` (page 18) structure. See DependentNotifyRec (page 18) for more information.

notifyComponent

A value that notifies the display component what engine, if any, caused a change in a dependent display. You may pass 0 in this parameter.

Return Value
A result code. See "Display Manager Result Codes" (page 44).

Discussion
The Display Manager uses the `DMSendDependentNotification` function to send notifications to registered Display Notice event-handling functions. This function uses all its parameters to supply values for the `DependentNotifyRec` (page 18) structure which is sent out to registrants. Generally, your application does not need to use this function.

Special Considerations

Because this function may move or purge memory blocks or access handles, you cannot call it at interrupt time.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Displays.h`

DMSetAVPowerState

Sets the power state of a display device. (Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMSetAVPowerState (
   AVIDType theID,
   AVPowerStatePtr setPowerState,
   UInt32 powerFlags,
   Handle displayState
);
```

Parameters

theID
The ID number of the display device whose power state you want to change.

setPowerState
On return, this parameter points to a value that your application can use to set the power state of a display device.

powerFlags
A value that specifies the power state to which a display device can be set.

displayState
A handle to internal Display Manager information about the current display state.

Return Value
A result code. See "Display Manager Result Codes" (page 44).

Special Considerations

Because this function may move or purge memory blocks or access handles, you cannot call it at interrupt time.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Displays.h`

DMSetDisplayComponent

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMSetDisplayComponent (
   GDHandle theDevice,
   Component displayComponent
);
```

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Displays.h`

DMSetDisplayMode

Sets the display mode and pixel depth for a video device. (Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMSetDisplayMode (
   GDHandle theDevice,
   UInt32 mode,
   UInt32 *depthMode,
   long reserved,
   Handle displayState
);
```

Parameters

theDevice

A handle to the `GDevice` structure for the video device whose display mode and pixel depth you wish to set.

mode

The number used by a video device to identify its display mode. If you supply the value 0 in this parameter, `DMSetDisplayMode` uses the current display mode. To specify another display mode, use the function `DMNewDisplayModeList` (page 75).

depthMode

A pointer to the desired pixel depth for the video device specified by `theDevice`. If you pass a pointer to 0, `DMSetDisplayMode` attempts to keep the current depth. If you pass a pointer to 1, 2, 4, 8, 16, or 32, `DMSetDisplayMode` attempts to set the device to use your specified pixel depth. If you supply a pointer to a value of 128 or greater, then `DMSetDisplayMode` sets the depth to the depth mode represented by the Video Depth Mode values. See "Video Depth Mode Values" for more information.

On return, this parameter contains a pointer to the new pixel depth. This value represents the depth mode closest to the one you requested when calling `DMSetDisplayMode`.

reserved

Reserved for future expansion. Pass `NULL` in this parameter.

displayState
If your application called DMBeginConfigureDisplays (page 50), you must pass the displayState handle obtained. Otherwise pass NULL in this parameter.

Return Value
A result code. See "Display Manager Result Codes" (page 44).

Special Considerations

Because this function may move or purge memory blocks or access handles, you cannot call it at interrupt time.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
Displays.h

DMSetEnableByAVID

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMSetEnableByAVID (
   AVIDType theAVID,
   Boolean doEnable,
   Handle displayState
);
```

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
Displays.h

DMSetMainDisplay

Sets a display to be the main screen. You should generally never need to use this function. (Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMSetMainDisplay (
   GDHandle newMainDevice,
   Handle displayState
);
```

Parameters

newMainDevice
A handle to the GDevice structure for the video device whose display you wish to make the main screen.

displayState

If your application called DMBeginConfigureDisplays (page 50), you must pass the displayState handle obtained. Otherwise pass NULL in this parameter.

Return Value
A result code. See "Display Manager Result Codes" (page 44).

Discussion
After a call to the function DMSetMainDisplay, the handle specified by the parameter newMainDevice will point to the GDevice structure for the video device whose display, before calling DMSetMainDisplay, was the main screen. To obtain a handle to the main screen, you can use the Color QuickDraw function GetMainDevice.

DMSetMainDisplay moves the menu bar to the display for the video device specified by newMainDevice. QuickDraw maps the (0,0) origin point of the global coordinate system to the main screen's upper-left corner, and other screens are positioned adjacent to it.

Special Considerations

Because this function may move or purge memory blocks or access handles, you cannot call it at interrupt time.

Applications generally never need to use this function. In case you find a compelling need to change the user's display configuration, this function is described here for completeness. Note that if your application uses Display Manager functions to change the display configuration of the user's video devices, your application should make these changes only with the consent of the user. If your application must have a specific pixel depth, for example, it should display a dialog box that offers the user a choice between changing to that depth or canceling display of the image.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
Displays.h

DMUnblockMirroring

Reenables video mirroring disabled by the function DMUnblockMirroring. You should generally never need to use this function. (Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMUnblockMirroring (
   void
);
```

Parameters

Return Value
A result code. See "Display Manager Result Codes" (page 44).

Special Considerations

Because this function may move or purge memory blocks or access handles, you cannot call it at interrupt time.

Applications generally never need to use this function. In case you find a compelling need to change the user's display configuration, this function is described here for completeness. Note that if your application uses Display Manager functions to change the display configuration of the user's video devices, your application should make these changes only with the consent of the user. If your application must have a specific pixel depth, for example, it should display a dialog box that offers the user a choice between changing to that depth or canceling display of the image.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.
Not available to 64-bit applications.

Declared In
`Displays.h`

DMUnmirrorDevice

Turns off video mirroring. (Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
OSErr DMUnmirrorDevice (
   GDHandle gDevice,
   Handle displayState
);
```

Parameters

gDevice
A handle to the `GDevice` structure for the video device on which you no longer wish to mirror the pixel image of another device.

displayState
If your application called `DMBeginConfigureDisplays` (page 50), you must pass the `displayState` handle obtained. Otherwise pass `NULL` in this parameter.

Return Value
A result code. See "Display Manager Result Codes" (page 44).

Discussion
When the function `DMUnmirrorDevice` completes, the display controlled by the video device specified in the `gDevice` parameter no longer contains the mirror image of another display.

Your application should leave control of video mirroring to the user. However, if video mirroring is useful for your application (for example, if your application displays on-screen presentations), you might provide a control so that the user can switch to video mirroring directly from your application. In this case, the function `DMMirrorDevices` (page 69) is useful for switching video mirroring on, and `DMUnmirrorDevice` function is useful for switching it off again.

Special Considerations

Because this function may move or purge memory blocks or access handles, you cannot call it at interrupt time.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

Declared In
`Displays.h`

InvokeDMComponentListIteratorUPP

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
void InvokeDMComponentListIteratorUPP (
   void *userData,
   DMListIndexType itemIndex,
   DMComponentListEntryPtr componentInfo,
   DMComponentListIteratorUPP userUPP
);
```

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.

Declared In
`Displays.h`

InvokeDMDisplayListIteratorUPP

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
void InvokeDMDisplayListIteratorUPP (
   void *userData,
   DMListIndexType itemIndex,
   DisplayListEntryPtr displaymodeInfo,
   DMDisplayListIteratorUPP userUPP
);
```

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.

Declared In
`Displays.h`

InvokeDMDisplayModeListIteratorUPP

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
void InvokeDMDisplayModeListIteratorUPP (
   void *userData,
   DMListIndexType itemIndex,
   DMDisplayModeListEntryPtr displaymodeInfo,
   DMDisplayModeListIteratorUPP userUPP
);
```

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.

Declared In
`Displays.h`

InvokeDMExtendedNotificationUPP

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
void InvokeDMExtendedNotificationUPP (
   void *userData,
   short theMessage,
   void *notifyData,
   DMExtendedNotificationUPP userUPP
);
```

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.

Declared In
`Displays.h`

InvokeDMNotificationUPP

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
void InvokeDMNotificationUPP (
   AppleEvent *theEvent,
   DMNotificationUPP userUPP
);
```

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.

Declared In
`Displays.h`

InvokeDMProfileListIteratorUPP

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
void InvokeDMProfileListIteratorUPP (
   void *userData,
   DMListIndexType itemIndex,
   DMProfileListEntryPtr profileInfo,
   DMProfileListIteratorUPP userUPP
);
```

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.

Declared In
`Displays.h`

NewDMComponentListIteratorUPP

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
DMComponentListIteratorUPP NewDMComponentListIteratorUPP (
   DMComponentListIteratorProcPtr userRoutine
);
```

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.

Declared In
`Displays.h`

NewDMDisplayListIteratorUPP

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
DMDisplayListIteratorUPP NewDMDisplayListIteratorUPP (
   DMDisplayListIteratorProcPtr userRoutine
);
```

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.

Declared In
`Displays.h`

NewDMDisplayModeListIteratorUPP

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
DMDisplayModeListIteratorUPP NewDMDisplayModeListIteratorUPP (
   DMDisplayModeListIteratorProcPtr userRoutine
);
```

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.

Declared In
`Displays.h`

NewDMExtendedNotificationUPP

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
DMExtendedNotificationUPP NewDMExtendedNotificationUPP (
   DMExtendedNotificationProcPtr userRoutine
);
```

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.

Declared In
`Displays.h`

NewDMNotificationUPP

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
DMNotificationUPP NewDMNotificationUPP (
   DMNotificationProcPtr userRoutine
);
```

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.

Declared In
`Displays.h`

NewDMProfileListIteratorUPP

(Deprecated in Mac OS X v10.4. Use Quartz Display Services instead; see *Quartz Display Services Reference*.)

```
DMProfileListIteratorUPP NewDMProfileListIteratorUPP (
   DMProfileListIteratorProcPtr userRoutine
);
```

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.4.

Declared In
`Displays.h`

Document Revision History

This table describes the changes to *Display Manager Reference*.

Date	Notes
2007-12-04	Updated legacy document information.
2006-07-24	Added information about deprecated functions.
2005-08-11	Made minor changes for GCC 4.0 compliance.
2003-02-01	Updated document format and structure.

Document Revision History

Index

F

G

I

K

M

N

P

R

S