Folder Manager Reference

Carbon > File Management

2006-07-12

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

Simultaneously published in the United States and Canada.

Contents

Folder Manager Reference

Framework: CoreServices/CoreServices.h

Declared in Folders.h

Overview

The Folder Manager allows you to find and search folders, create new folders, and control how files are routed between folders. Because you can use the Folder Manager to manipulate standard Mac OS folders without relying on their names, your program is tolerant of changes to folder names and easier to localize.

Carbon supports the Folder Manager, although some functions have been deprecated in Mac OS X. You should always check the value of `gestaltFindFolderAttr` in Mac OS X to determine what functionality is available.

Functions by Task

Describing Folders

`GetFolderTypes` (page 13)
Obtains the folder types contained in the global descriptor list.

`IdentifyFolder` (page 13)
Obtains the folder type for the specified folder.

`InvalidateFolderDescriptorCache` (page 14)
Invalidates any prior `FindFolder` results for the specified folder.

`GetFolderName` (page 52) Deprecated in Mac OS X v10.5
Obtains the name of the specified folder.

Manipulating Folders

`FSFindFolder` (page 12)
Obtains location information for system-related directories.

`FindFolder` (page 9)
Obtains location information for system-related directories.

`FindFolderExtended` (page 43) Deprecated in Mac OS X v10.3
Obtains location information for system-related directories. (Deprecated. Use `FindFolder` (page 9) instead.)

`FSFindFolderExtended` (page 45) Deprecated in Mac OS X v10.3
: Locates a system-related folder and returns a reference to the folder. (Deprecated. Use `FSFindFolder` (page 12) instead.)

`ReleaseFolder` (page 47) Deprecated in Mac OS X v10.3
: Releases the Trash folder in preparation for unmounting a server volume. (Deprecated. This function is not needed in Mac OS X.)

Routing Files

`AddFolderRouting` (page 48) Deprecated in Mac OS X v10.4
: Adds a folder routing structure to the global routing list. (Deprecated. There is no replacement function.)

`FindFolderRouting` (page 49) Deprecated in Mac OS X v10.4
: Finds the destination folder from a matching folder routing structure for the specified file. (Deprecated. There is no replacement function.)

`GetFolderRoutings` (page 49) Deprecated in Mac OS X v10.4
: Obtains folder routing information from the global routing list. (Deprecated. There is no replacement function.)

`RemoveFolderRouting` (page 50) Deprecated in Mac OS X v10.4
: Deletes a folder routing structure from the global routing list. (Deprecated. There is no replacement function.)

Working With Folder Manager Notification Functions

`NewFolderManagerNotificationUPP` (page 15)
: Creates a new universal procedure pointer (UPP) to a notification function.

`DisposeFolderManagerNotificationUPP` (page 9)
: Disposes of the universal procedure pointer (UPP) to a notification function.

`InvokeFolderManagerNotificationUPP` (page 15)
: Calls your notification function.

`FolderManagerRegisterCallNotificationProcs` (page 43) Deprecated in Mac OS X v10.3
: Calls the registered Folder Manager notification procs. (Deprecated. There is no replacement function.)

`FolderManagerRegisterNotificationProc` (page 44) Deprecated in Mac OS X v10.3
: Registers your notification function with the Folder Manager. (Deprecated. There is no replacement function.)

`FolderManagerUnregisterNotificationProc` (page 45) Deprecated in Mac OS X v10.3
: Removes your notification function from the Folder Manager's queue. (Deprecated. There is no replacement function.)

Working With Folder Descriptors

`AddFolderDescriptor` (page 7)
: Copies the supplied information into a new folder descriptor entry in the system folder list.

`RemoveFolderDescriptor` (page 15)
: Deletes the specified folder descriptor entry from the system folder list.

`GetFolderDescriptor` (page 46) Deprecated in Mac OS X v10.3
: Obtains the folder descriptor information for the specified folder type from the global descriptor list. (Deprecated. There is no replacement function.)

Finding Files in Special Folders

`FSDetermineIfRefIsEnclosedByFolder` (page 11)
: Determines whether a file of type `FSRef` is enclosed inside a special folder type for the given domain.

`DetermineIfPathIsEnclosedByFolder` (page 8)
: Determines whether a file path is enclosed inside a special folder type for the given domain.

`FSpDetermineIfSpecIsEnclosedByFolder` (page 51) Deprecated in Mac OS X v10.5
: Determines whether a file of type `FSSpec` is enclosed inside a special folder type for the given domain.

Functions

AddFolderDescriptor

Copies the supplied information into a new folder descriptor entry in the system folder list.

```
OSErr AddFolderDescriptor (
   FolderType foldType,
   FolderDescFlags flags,
   FolderClass foldClass,
   FolderLocation foldLocation,
   OSType badgeSignature,
   OSType badgeType,
   ConstStrFileNameParam name,
   Boolean replaceFlag
);
```

Parameters

foldType
: Pass a constant identifying the type of the folder you wish the Folder Manager to be able to find. See Folder Type Constants (page 26).

flags
: Set these flags to indicate whether a folder is created during startup, if the folder name is locked, and if the folder is created invisible; see Folder Descriptor Flags (page 23).

foldClass
: Pass the class of the folder which you wish the Folder Manager to be able to find. The folder class determines how the `foldLocation` parameter is interpreted. See Folder Descriptor Classes (page 22) for a discussion of relative and special folder classes.

foldLocation
: For a relative folder, specify the folder type of the parent folder of the target. For a special folder, specify the location of the folder; see Folder Descriptor Locations (page 25).

badgeSignature

Reserved. Pass 0.

badgeType

Reserved. Pass 0.

name

A string specifying the name of the desired folder. For relative folders, this is the exact name of the desired folder. For special folders, the actual target folder may have a different name than the name specified in the folder descriptor. For example, the System Folder is often given a different name, but it can still be located with `FindFolder` (page 9).

replaceFlag

Pass a `Boolean` value indicating whether you wish to replace a folder descriptor that already exists for the specified folder type. If `true`, it replaces the folder descriptor for the specified folder type. If `false`, it does not replace the folder descriptor for the specified folder type.

Return Value

A result code. See "Folder Manager Result Codes" (page 42). The result code `duplicateFolderDescErr` indicates that a folder descriptor is already installed with the specified folder type and `replaceFlag` is false.

Discussion

The `AddFolderDescriptor` function copies the supplied information into a new descriptor entry in the system folder list. You need to provide folder descriptors for each folder you wish the Folder Manager to be able to find via the function `FindFolder` (page 9). For example, a child folder located in a parent folder needs to have a descriptor created both for it and its parent folder, so that the child can be found. This function is supported under Mac OS 8 and later.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`Folders.h`

DetermineIfPathIsEnclosedByFolder

Determines whether a file path is enclosed inside a special folder type for the given domain.

```
OSErr DetermineIfPathIsEnclosedByFolder (
   FSVolumeRefNum domainOrVRefNum,
   OSType folderType,
   const UInt8 *utf8Path,
   Boolean pathIsRealPath,
   Boolean *outResult
);
```

Parameters

domainOrVRefNum

The domain or volume reference number to check. For information about the possible domains, see Disk and Domain Constants (page 38). You can also pass `0` to check all domains and volumes, or you can pass `kOnAppropriateDisk` to check the appropriate volume for the specified file.

folderType

The special folder type to check. For information about the possible folder types, see Folder Type Constants (page 26).

utf8Path

A UTF-8 encoded path to the file for which to search.

pathIsRealPath

A Boolean value that indicates whether the `utf8Path` parameter is guaranteed to be a full and complete path, as opposed to a path containing a symbolic link, an alias, or a relative path.

outResult

A pointer to a Boolean variable. On return, indicates whether or not the file is enclosed inside the special folder type for the given domain.

Discussion

This function provides an efficient way to check to see if a file (or folder) is inside a special folder for a given domain. A typical use for this function is to determine if a given file is inside the trash on a volume:

```
err = DetermineIfPathIsEnclosedByFolder (kOnAppropriateDisk, kTrashFolderType,
 path, false, &result);
```

Availability

Available in Mac OS X v10.4 and later.

Declared In

`Folders.h`

DisposeFolderManagerNotificationUPP

Disposes of the universal procedure pointer (UPP) to a notification function.

```
void DisposeFolderManagerNotificationUPP (
   FolderManagerNotificationUPP userUPP
);
```

Parameters

userUPP

The UPP to dispose of.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`Folders.h`

FindFolder

Obtains location information for system-related directories.

```
OSErr FindFolder (
   FSVolumeRefNum vRefNum,
   OSType folderType,
   Boolean createFolder,
   FSVolumeRefNum *foundVRefNum,
   SInt32 *foundDirID
);
```

Parameters

vRefNum

Pass the volume reference number of the volume on which you want to locate a directory, or a constant specifying a disk or domain. The constants which you can use in this parameter are described in Disk and Domain Constants (page 38).

Note that, on Mac OS X, passing a volume reference number in this parameter does not make sense for most of the folder type selectors which you can specify in the `folderType` parameter. On Mac OS X, folders are "domain-oriented"; because there may be more than one domain on any given physical volume, asking for these folders on a per-volume basis yields undefined results. For example, if you were to request the Fonts folder (represented by the selector `kFontsFolderType`) on volume -100, are you requesting the folder /System/Library/Fonts, /Library/Fonts, or ~/Fonts? On Mac OS X you should pass a disk or domain constant in this parameter.

folderType

Pass a four-character folder type, or a constant that represents the type, for the folder you want to find; see Folder Type Constants (page 26).

createFolder

A value of type `Boolean`, as defined in Create Folder Flags (page 22). Pass the constant `kCreateFolder` to create a directory if it does not already exist; otherwise, pass the constant `kDontCreateFolder`. Directories inside the System Folder are created only if the System Folder directory exists. The `FindFolder` function will not create a System Folder directory even if you specify the `kCreateFolder` constant in the `createFolder` parameter. Passing `kCreateFolder` will also not create a parent folder; if the parent of the target folder does not already exist, attempting to create the target will fail.

foundVRefNum

A pointer to a value of type `short`. On return, the value specifies the volume reference number for the volume containing the directory specified in the `folderType` parameter.

foundDirID

A pointer to a value of type `long`. On return, the value specifies the directory ID number for the directory specified in the `folderType` parameter.

Return Value

A result code. See "Folder Manager Result Codes" (page 42). The result code `fnfErr` indicates that the type has not been found in the `'fld#'` resource, or the disk doesn't have System Folder support, or the disk does not have desktop database support for Desktop Folder—in all cases, the folder has not been found. The result code `dupFNErr` indicates that a file has been found instead of a folder.

Discussion

As of Mac OS 8 and later, your application can add folders to the System Folder—or nest folders within other folders—and locate the folders via the `FindFolder` function. Prior to Mac OS 8, your application could only use `FindFolder` to find folders that were immediately inside of the System Folder, and a few other special folders such as the Trash folder and the System Folder itself. Now, once a folder (and any folders that it is nested within) is described in a folder descriptor—that is, registered using the function `AddFolderDescriptor` (page 7) —your application can use `FindFolder` to find the folder no matter where it is located.

Those folders you're most likely to want to access are Preferences and Trash. For example, you might wish to check for the existence of a user's configuration file in Preferences or, if your application runs out of disk storage when trying to save a file, check how much disk storage is taken by items in the Trash directory and report this to the user.

The specified folder used for a given volume might be located on a different volume; therefore, do not assume the volume that you specify in `vRefNum` and the volume returned through `foundVRefNum` will be the same.

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

Related Sample Code
Simple DrawSprocket

Declared In
`Folders.h`

FSDetermineIfRefIsEnclosedByFolder

Determines whether a file of type `FSRef` is enclosed inside a special folder type for the given domain.

```
OSErr FSDetermineIfRefIsEnclosedByFolder (
   FSVolumeRefNum domainOrVRefNum,
   OSType folderType,
   const FSRef *inRef,
   Boolean *outResult
);
```

Parameters

`domainOrVRefNum`

The domain or volume reference number to check. For information about the possible domains, see Disk and Domain Constants (page 38). You can also pass `0` to check all domains and volumes, or you can pass `kOnAppropriateDisk` to check the appropriate volume for the specified file.

`folderType`

The special folder type to check. For information about the possible folder types, see Folder Type Constants (page 26).

`inRef`

The file for which to search.

`outResult`

A pointer to a Boolean variable. On return, indicates whether or not the file is enclosed inside the special folder type for the given domain.

Discussion
This function provides an efficient way to check to see if a file (or folder) is inside a special folder for a given domain. A typical use for this function is to determine if a given file is inside the trash on a volume:

```
err = FSDetermineIfRefIsEnclosedByFolder (kOnAppropriateDisk, kTrashFolderType,
 &ref, &result);
```

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

Declared In
Folders.h

FSFindFolder

Obtains location information for system-related directories.

```
OSErr FSFindFolder (
   FSVolumeRefNum vRefNum,
   OSType folderType,
   Boolean createFolder,
   FSRef *foundRef
);
```

Parameters

vRefNum

Pass the volume reference number of the volume on which you want to locate a directory, or a constant specifying a disk or domain. The constants which you can use in this parameter are described in Disk and Domain Constants (page 38).

Note that, on Mac OS X, passing a volume reference number in this parameter does not make sense for most of the folder type selectors which you can specify in the `folderType` parameter. On Mac OS X, folders are "domain-oriented"; because there may be more than one domain on any given physical volume, asking for these folders on a per-volume basis yields undefined results. For example, if you were to request the Fonts folder (represented by the selector `kFontsFolderType`)on volume -100, are you requesting the folder /System/Library/Fonts, /Library/Fonts, or ~/Fonts? On Mac OS X you should pass a disk or domain constant in this parameter.

folderType

Pass a four-character folder type, or a constant that represents the type, for the folder you want to find; see Folder Type Constants (page 26).

createFolder

A value of type `Boolean`, as defined in Create Folder Flags (page 22). Pass the constant `kCreateFolder` to create a directory if it does not already exist; otherwise, pass the constant `kDontCreateFolder`. Passing `kCreateFolder` will not create a parent folder; if the parent of the target folder does not already exist, attempting to create the target will fail.

foundRef

A pointer to a file system reference. On return, the `FSRef` refers to the directory specified by the `vRefNum` and `folderType` parameters.

Return Value
A result code. See "Folder Manager Result Codes" (page 42).

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

Related Sample Code
BSDLLCTest

Declared In
Folders.h

GetFolderTypes

Obtains the folder types contained in the global descriptor list.

```
OSErr GetFolderTypes (
   UInt32 requestedTypeCount,
   UInt32 *totalTypeCount,
   FolderType *theTypes
);
```

Parameters

requestedTypeCount

Pass the number of `FolderType` values that can fit in the buffer pointed to by the `theTypes` parameter; see Folder Type Constants (page 26).

totalTypeCount

Pass a pointer to an unsigned 32-bit integer value. On return, the value is set to the total number of `FolderType` values in the list. The `totalTypeCount` parameter may produce a value that is larger or smaller than that of the `requestedTypeCount` parameter. If `totalTypeCount` is equal to or smaller than the value passed in for `requestedTypeCount` and the value produced by the `theTypes` parameter is non-null, then all folder types were returned to the caller.

theTypes

Pass a pointer to an array of `FolderType` values; see Folder Type Constants (page 26). On return, the array contains the folder types for the installed descriptors. You can step through the array and call `GetFolderDescriptor` for each folder type. Pass `null` if you only want to know the number of descriptors installed in the system's global list, rather than the actual folder types of those descriptors.

Return Value

A result code. See "Folder Manager Result Codes" (page 42).

Availability

Available in Mac OS X v10.0 and later.

Declared In

`Folders.h`

IdentifyFolder

Obtains the folder type for the specified folder.

```
OSErr IdentifyFolder (
   FSVolumeRefNum vRefNum,
   SInt32 dirID,
   FolderType *foldType
);
```

Parameters

vRefNum

Pass the volume reference number (or the constant `kOnSystemDisk` for the startup disk) of the volume containing the folder whose type you wish to identify.

dirID

Pass the directory ID number for the folder whose type you wish to identify.

foldType

Pass a pointer to a value of type `FolderType`. On return, the value is set to the folder type of the folder with the specified `vRefNum` and `dirID` parameters; see Folder Type Constants (page 26) for descriptions of possible values.

Return Value
A result code. See "Folder Manager Result Codes" (page 42).

Discussion
The folder type is identified for the folder specified by the `vRefNum` and `dirID` parameters, if such a folder exists. Note that if there are multiple folder descriptors that map to an individual folder, `IdentifyFolder` returns the folder type of only the first matching descriptor that it finds.

Carbon Porting Notes

This function is not useful on Mac OS X.

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

Declared In
`Folders.h`

InvalidateFolderDescriptorCache

Invalidates any prior `FindFolder` results for the specified folder.

```
OSErr InvalidateFolderDescriptorCache (
   FSVolumeRefNum vRefNum,
   SInt32 dirID
);
```

Parameters

vRefNum

Pass the volume reference number (or the constant `kOnSystemDisk` for the startup disk) of the volume containing the folder for which you wish the descriptor cache to be invalidated. Pass 0 to completely invalidate all folder cache information.

dirID

Pass the directory ID number for the folder for which you wish the descriptor cache to be invalidated. Pass 0 to invalidate the cache for all folders on the specified disk.

Return Value
A result code. See "Folder Manager Result Codes" (page 42).

Discussion
The `InvalidateFolderDescriptorCache` function searches to see if there is currently a cache of results from `FindFolder` calls on the specified folder. If so, it invalidates the cache from the previous calls to the `FindFolder` function in order to force the Folder Manager to reexamine the disk when `FindFolder` is called again on the specified directory ID or volume reference number.

If you remove a directory on disk which you know is a Folder Manager folder, you should call this function to update the Folder Manager.

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

Declared In
`Folders.h`

InvokeFolderManagerNotificationUPP

Calls your notification function.

```
OSStatus InvokeFolderManagerNotificationUPP (
   OSType message,
   void *arg,
   void *userRefCon,
   FolderManagerNotificationUPP userUPP
);
```

Discussion
You should not need to use the `InvokeFolderManagerNotificationUPP` function, as the system calls your notification function for you.

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

Declared In
`Folders.h`

NewFolderManagerNotificationUPP

Creates a new universal procedure pointer (UPP) to a notification function.

```
FolderManagerNotificationUPP NewFolderManagerNotificationUPP (
   FolderManagerNotificationProcPtr userRoutine
);
```

Parameters
`userRoutine`
A pointer to your notification function.

Return Value
The UPP to the notification function.

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

Declared In
`Folders.h`

RemoveFolderDescriptor

Deletes the specified folder descriptor entry from the system folder list.

```
OSErr RemoveFolderDescriptor (
   FolderType foldType
);
```

Parameters

foldType

Pass a constant identifying the type of the folder for which you wish to remove a descriptor. See Folder Type Constants (page 26).

Return Value

A result code. See "Folder Manager Result Codes" (page 42).

Discussion

Once a folder descriptor has been removed, the function `FindFolder` (page 9) will no longer be able to locate the folder type.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`Folders.h`

Callbacks

FolderManagerNotificationProcPtr

Defines a pointer to a notification function, called for all Folder Manager notifications.

```
typedef OSStatus (*FolderManagerNotificationProcPtr)
(
                OSType message,
                void * arg,
                void * userRefCon);
```

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

```
OSStatus MyFolderManagerNotificationProc
(
            OSType message,
            void * arg,
            void * userRefCon);
```

Parameters

message

The type of notification (user login, user logout, etc.). See "Notification Messages" (page 40).

arg

A pointer to additional information, if any. For most messages, this is a pointer to a `FindFolderUserRedirectionGlobals` structure. If the message is `kFolderManagerNotificationDiscardCachedData`, `arg` is undefined.

userRefCon

A pointer to a value for your own use; this may be any value you want, such as a pointer to your globals or other state information.

Return Value
A result code. See "Folder Manager Result Codes" (page 42).

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

Declared In
`Folders.h`

Data Types

FindFolderUserRedirectionGlobals

Used in the `arg` parameter of a notification function.

```
struct FindFolderUserRedirectionGlobals {
    UInt32 version;
    UInt32 flags;
    Str31 userName;
    short userNameScript;
    short currentUserFolderVRefNum;
    long currentUserFolderDirID;
    short remoteUserFolderVRefNum;
    long remoteUserFolderDirID;
};
typedef struct FindFolderUserRedirectionGlobals FindFolderUserRedirectionGlobals;
typedef FindFolderUserRedirectionGlobals * FindFolderUserRedirectionGlobalsPtr;
```

Availability
Available in Mac OS X v10.0 through Mac OS X v10.4.

Declared In
`Folders.h`

FolderDesc

Used to find existing folder descriptors and create new ones.

```
struct FolderDesc {
    Size descSize;
    FolderType foldType;
    FolderDescFlags flags;
    FolderClass foldClass;
    FolderType foldLocation;
    OSType badgeSignature;
    OSType badgeType;
    UInt32 reserved;
    StrFileName name;
};
typedef struct FolderDesc FolderDesc;
typedef FolderDesc * FolderDescPtr;
```

Fields

`descSize`

The size (in bytes) of this structure.

`foldType`

A constant of type `FolderType` that identifies the kind of target folder. See "Folder Type Constants" (page 26) for a list of possible folder types.

`flags`

Flags indicating whether a folder is created during startup, if the folder name is locked, and if the folder created is invisible; see "Folder Descriptor Flags" (page 23).

`foldClass`

The class indicating whether the folder is relative to the parent folder or special; see "Folder Descriptor Classes" (page 22).

`foldLocation`

For a relative folder, the `foldLocation` field specifies the `FolderType` of the parent folder of the target. For special folders, the location of the folder. See "Folder Descriptor Locations" (page 25).

`badgeSignature`

Reserved. Set this field to 0.

`badgeType`

Reserved. Set this field to 0.

`reserved`

Reserved. Set this field to 0.

`name`

A string specifying the name of the desired folder. For relative folders, this will be the exact name of the desired folder. For special folders, the actual target folder may have a different name than the name specified in the folder descriptor. For example, the System Folder is often given a different name, but it can still be located with `FindFolder` (page 9).

Discussion

The `FolderDesc` structure is supported under Mac OS 8 and later.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`Folders.h`

FolderManagerNotificationUPP

Defines a universal procedure pointer (UPP) to a notification function.

```
typedef FolderManagerNotificationProcPtr FolderManagerNotificationUPP;
```

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

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

Declared In
`Folders.h`

FolderRouting

Specifies the folder that files are routed to, based on the folder they are routed from.

```
struct FolderRouting {
    Size descSize;
    OSType fileType;
    FolderType routeFromFolder;
    FolderType routeToFolder;
    RoutingFlags flags;
};
typedef struct FolderRouting FolderRouting;
typedef FolderRouting * FolderRoutingPtr;
```

Fields

`descSize`
The size (in bytes) of this structure.

`fileType`
A constant of type `OSType` that describes the file type of the item to be routed.

`routeFromFolder`
The folder type identifying the folder from which an item will be routed. If an item is dropped on the folder specified in the `routeFromFolder` field, it will be routed to the folder described in the `routeToFolder` field. See "Folder Type Constants" (page 26) for a list of possible values.

`routeToFolder`
The folder type identifying the folder to which an item will be routed; see "Folder Type Constants" (page 26) for a list of possible values.

`flags`
Reserved. Set this field to 0.

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

Declared In
`Folders.h`

MultiUserGestalt

```
struct MultiUserGestalt {
    short giVersion;
    short giReserved0;
    short giReserved1;
    short giReserved2;
    short giReserved3;
    FSSpec giReserved4;
    short giDocsVRefNum;
    long giDocsDirID;
    short giForceSaves;
    short giForceOpens;
    Str31 giSetupName;
    Str31 giUserName;
    Str31 giFrontAppName;
    short giReserved5;
    short giIsOn;
    short giUserLoggedInType;
    char giUserEncryptPwd[16];
    short giUserEnvironment;
    long giReserved6;
    long giReserved7;
    Boolean giDisableScrnShots;
    Boolean giSupportsAsyncFSCalls;
    short giPrefsVRefNum;
    long giPrefsDirID;
    unsigned long giUserLogInTime;
    Boolean giUsingPrintQuotas;
    Boolean giUsingDiskQuotas;
    Boolean giInSystemAccess;
    Boolean giUserFolderEnabled;
    short giReserved8;
    long giReserved9;
    Boolean giInLoginScreen;
};
typedef struct MultiUserGestalt MultiUserGestalt;
typedef MultiUserGestalt * MultiUserGestaltPtr;
```

Fields

`giVersion`
: The structure version. A structure version of 0 is invalid.

`giReserved0`
: Obsolete with structure version 3.

`giReserved1`
: Obsolete.

`giReserved2`
: Obsolete with structure version 6.]

`giReserved3`
: Obsolete.

`giReserved4`
: Obsolete with structure version 6.

`giDocsVRefNum`
: The volume reference number associated with the user's documents location.

`giDocsDirID`
: The directory ID of the user's documents folder.

`giForceSaves`
: `True` if the user is forced to save to their documents folder.

`giForceOpens`
: `True` if the user is forced to open from their documents folder.

`giSetupName`
: The name of the current setup.

`giUserName`
: The name of the current user.

`giFrontAppName`
: The name of the frontmost application.

`giReserved5`
: Obsolete with structure version 6.

`giIsOn`
: `True` if Multiple Users or Macintosh Manager is currently on.

`giUserLoggedInType`
: The logged in user type. Zero indicates a normal user, 1 indicates a workgroup administrator, and 2 indicates a global administrator.

`giUserEncryptPwd`
: The encrypted user password.

`giUserEnvironment`
: The environment that the user has logged into.

`giReserved6`
: Obsolete.

`giReserved7`
: Obsolete.

`giDisableScrnShots`
: `True` if screen shots are not allowed.

`giSupportsAsyncFSCalls`
: The Finder uses this to tell if our patches support asynchronous trap patches.

`giPrefsVRefNum`
: The volume reference number of preferences.

`giPrefsDirID`
: The directory ID of the At Ease Items folder on the preferences volume.

`giUserLogInTime`
: The time in seconds the user has been logged in.

`giUsingPrintQuotas`
: `True` if the logged in user is using printer quotas.

`giUsingDiskQuotas`
: `True` if the logged in user has disk quotas active.

`giInSystemAccess`
: `True` if the system is in System Access, that is the owner is logged in.

`giUserFolderEnabled`
`True` if `FindFolder` is redirecting folders.

`giReserved8`

`giReserved9`

`giInLoginScreen`
`True` if no user has logged in, including the owner.

Availability
Available in Mac OS X v10.0 through Mac OS X v10.4.

Declared In
`Folders.h`

Constants

Create Folder Flags

Indicate whether a folder should be created, if it is not found.

```
enum {
    kCreateFolder = true,
    kDontCreateFolder = false
};
```

Constants

`kCreateFolder`
Specifies that the folder should be created, if it is not found.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kDontCreateFolder`
Specifies that the folder should not be created, if it is not found.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

Discussion
You can pass these flag constants in the `createFolder` parameter of the function `FSFindFolder` (page 12).

Folder Descriptor Classes

Specify how folder location information should be interpreted.

```
enum {
    kRelativeFolder = 'relf',
    kSpecialFolder = 'spcf'
};
typedef OSType FolderClass;
```

Constants

`kRelativeFolder`

Relative folders are located in terms of the folders in which they are nested, that is, their parent folders. This constant indicates that the folder location specified is the folder type of the parent folder, and the name specified is the name of the folder. Most folder descriptors are for relative folders.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kSpecialFolder`

Special folders—such as the System Folder and the disk's root directory—are in set locations that are not determined relative to any other folder. This constant indicates that the folder is located algorithmically, according to the constant supplied for the folder location (`kBlessedFolder` or `kRootFolder`). Developers cannot create new folder descriptors of the `kSpecialFolder` class.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

Discussion

Constants of type `FolderClass` are used to specify how folder location information should be interpreted in the function `AddFolderDescriptor` (page 7) and the structure `FolderDesc` (page 17). The `FolderClass` constants are supported under Mac OS 8 and later.

Developers can only create new folder descriptors with a class of `kRelativeFolder`.

Folder Descriptor Flags

Specify various attributes of a folder.

```
enum {
    kCreateFolderAtBoot = 0x00000002,
    kCreateFolderAtBootBit = 1,
    kFolderCreatedInvisible = 0x00000004,
    kFolderCreatedInvisibleBit = 2,
    kFolderCreatedNameLocked = 0x00000008,
    kFolderCreatedNameLockedBit = 3,
    kFolderCreatedAdminPrivs = 0x00000010,
    kFolderCreatedAdminPrivsBit = 4
};enum {
    kFolderInUserFolder = 0x00000020,
    kFolderInUserFolderBit = 5,
    kFolderTrackedByAlias = 0x00000040,
    kFolderTrackedByAliasBit = 6,
    kFolderInRemoteUserFolderIfAvailable = 0x00000080,
    kFolderInRemoteUserFolderIfAvailableBit = 7,
    kFolderNeverMatchedInIdentifyFolder = 0x00000100,
    kFolderNeverMatchedInIdentifyFolderBit = 8,
    kFolderMustStayOnSameVolume = 0x00000200,
    kFolderMustStayOnSameVolumeBit = 9,
    kFolderManagerFolderInMacOS9FolderIfMacOSXIsInstalledMask =
0x00000400,
    kFolderManagerFolderInMacOS9FolderIfMacOSXIsInstalledBit = 10,
    kFolderInLocalOrRemoteUserFolder = kFolderInUserFolder |
kFolderInRemoteUserFolderIfAvailable
};
typedef UInt32 FolderDescFlags;
```

Constants

`kCreateFolderAtBoot`

If the bit specified by this mask is set, the folder is created during startup if needed.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kCreateFolderAtBootBit`

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kFolderCreatedInvisible`

If the bit specified by this mask is set, the folder created is invisible.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kFolderCreatedInvisibleBit`

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kFolderCreatedNameLocked`

If the bit specified by this mask is set, the name of the folder is locked when the folder is created.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kFolderCreatedNameLockedBit`

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

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

Declared in `Folders.h`.

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

Declared in `Folders.h`.

Discussion
The `FolderDescFlags` enumeration defines masks your application can use in the `AddFolderDescriptor` (page 7) function and the `FolderDesc` (page 17) structure to specify various attributes of a folder. All other flag bits are reserved for future use. The `FolderDescFlags` constants are supported under Mac OS 8 and later.

Folder Descriptor Locations

Identify special folder locations.

```
enum {
    kBlessedFolder = 'blsf',
    kRootFolder = 'rotf'
};typedef OSType FolderLocation;
```

Constants
`kBlessedFolder`
Indicates that the folder location is the System Folder on the volume.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kRootFolder`
Indicates that the folder location is the root directory of the volume.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

Discussion
There are two special folder locations that you can specify in folder descriptors via the `FolderDesc` (page 17) structure and the `AddFolderDescriptor` (page 7) function. For folders whose class is `kSpecialFolder`, you can use the following constants to specify the location of the folder algorithmically. The `FolderLocation` constants are supported under Mac OS 8 and later.

kCurrentUserFolderLocation

```
enum {
    kCurrentUserFolderLocation = 'cusf'
};
```

Constants
`kCurrentUserFolderLocation`
Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

Folder Type Constants

Specify a type of folder on a particular volume.

```
enum {
    kSystemFolderType = 'macs',
    kDesktopFolderType = 'desk',
    kSystemDesktopFolderType = 'sdsk',
    kTrashFolderType = 'trsh',
    kSystemTrashFolderType = 'strs',
    kWhereToEmptyTrashFolderType = 'empt',
    kPrintMonitorDocsFolderType = 'prnt',
    kStartupFolderType = 'strt',
    kShutdownFolderType = 'shdf',
    kAppleMenuFolderType = 'amnu',
    kControlPanelFolderType = 'ctrl',
    kSystemControlPanelFolderType = 'sctl',
    kExtensionFolderType = 'extn',
    kFontsFolderType = 'font',
    kPreferencesFolderType = 'pref',
    kSystemPreferencesFolderType = 'sprf',
    kTemporaryFolderType = 'temp'
};
enum {
    kExtensionDisabledFolderType = 'extD',
    kControlPanelDisabledFolderType = 'ctrD',
    kSystemExtensionDisabledFolderType = 'macD',
    kStartupItemsDisabledFolderType = 'strD',
    kShutdownItemsDisabledFolderType = 'shdD',
    kApplicationsFolderType = 'apps',
    kDocumentsFolderType = 'docs'
};
enum {
    kVolumeRootFolderType = 'root',
    kChewableItemsFolderType = 'flnt',
    kApplicationSupportFolderType = 'asup',
    kTextEncodingsFolderType = 'ftex',
    kStationeryFolderType = 'odst',
    kOpenDocFolderType = 'odod',
    kOpenDocShellPlugInsFolderType = 'odsp',
    kEditorsFolderType = 'oded',
    kOpenDocEditorsFolderType = 'fodf',
    kOpenDocLibrariesFolderType = 'odlb',
    kGenEditorsFolderType = 'fedi',
    kHelpFolderType = 'fhlp',
    kInternetPlugInFolderType = 'fnet',
    kModemScriptsFolderType = 'fmod',
    kPrinterDescriptionFolderType = 'ppdf',
    kPrinterDriverFolderType = 'fprd',
    kScriptingAdditionsFolderType = 'fscr',
    kSharedLibrariesFolderType = 'flib',
    kVoicesFolderType = 'fvoc',
    kControlStripModulesFolderType = 'sdev',
    kAssistantsFolderType = 'astf',
    kUtilitiesFolderType = 'utif',
    kAppleExtrasFolderType = 'aexf',
    kContextualMenuItemsFolderType = 'cmnu',
    kMacOSReadMesFolderType = 'morf',
    kALMModulesFolderType = 'walk',
    kALMPreferencesFolderType = 'trip',
    kALMLocationsFolderType = 'fall',
    kColorSyncProfilesFolderType = 'prof',
```

```
    kThemesFolderType = 'thme',
    kFavoritesFolderType = 'favs',
    kInternetFolderType = 'intf',
    kAppearanceFolderType = 'appr',
    kSoundSetsFolderType = 'snds',
    kDesktopPicturesFolderType = 'dtpf',
    kInternetSearchSitesFolderType = 'issf',
    kFindSupportFolderType = 'fnds',
    kFindByContentFolderType = 'fbcf',
    kInstallerLogsFolderType = 'ilgf',
    kScriptsFolderType = 'scrf',
    kFolderActionsFolderType = 'fasf',
    kLauncherItemsFolderType = 'laun',
    kRecentApplicationsFolderType = 'rapp',
    kRecentDocumentsFolderType = 'rdoc',
    kRecentServersFolderType = 'rsvr',
    kSpeakableItemsFolderType = 'spki',
    kKeychainFolderType = 'kchn',
    kQuickTimeExtensionsFolderType = 'qtex',
    kDisplayExtensionsFolderType = 'dspl',
    kMultiprocessingFolderType = 'mpxf',
    kPrintingPlugInsFolderType = 'pplg'
};
typedef OSType FolderType;
```

Constants

`kSystemFolderType`

Specifies the System Folder.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kDesktopFolderType`

Specifies the Desktop Folder.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kSystemDesktopFolderType`

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kTrashFolderType`

Specifies the single-user Trash folder.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kSystemTrashFolderType`

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kWhereToEmptyTrashFolderType`

Specifies the shared Trash folder on a file server, this indicates the parent directory of all logged-on users' Trash subdirectories.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kPrintMonitorDocsFolderType`
Specifies the PrintMonitor Documents folder in the System Folder.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kStartupFolderType`
Specifies the Startup Items folder in the System Folder.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kShutdownFolderType`
Specifies the Shutdown Items folder in the System Folder.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kAppleMenuFolderType`
Specifies the Apple Menu Items folder in the System Folder.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kControlPanelFolderType`
Specifies the Control Panels folder in the System Folder.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

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

Declared in `Folders.h`.

`kExtensionFolderType`
Specifies the Extensions folder in the System Folder.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kFontsFolderType`
Specifies the Fonts folder in the System Folder.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kPreferencesFolderType`
Specifies the Preferences folder in the System Folder.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

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

Declared in `Folders.h`.

`kTemporaryFolderType`
Specifies the Temporary folder. This folder exists as an invisible folder at the volume root.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kExtensionDisabledFolderType`
Specifies the Extensions (Disabled) folder in the System Folder. Supported with Mac OS 8 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kControlPanelDisabledFolderType`
Specifies the Control Panels (Disabled) folder in the System Folder. Supported with Mac OS 8 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kSystemExtensionDisabledFolderType`
Specifies the System Extensions (Disabled) folder in the System Folder. Supported with Mac OS 8 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kStartupItemsDisabledFolderType`
Specifies the Startup Items (Disabled) folder in the System Folder. Supported with Mac OS 8 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kShutdownItemsDisabledFolderType`
Specifies the Shutdown Items (Disabled) folder in the System Folder. Supported with Mac OS 8 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kApplicationsFolderType`
Specifies the Applications folder installed at the root level of the volume. Supported with Mac OS 8 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kDocumentsFolderType`
Specifies the Documents folder. This folder is created at the volume root. Supported with Mac OS 8 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kVolumeRootFolderType`
Specifies the root folder of a volume. Supported with Mac OS 8 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kChewableItemsFolderType`
Specifies the invisible folder on the system disk called "Cleanup at Startup" whose contents are deleted when the system is restarted, instead of merely being moved to the Trash. When the `FindFolder` function indicates this folder is available (by returning `noErr`), developers should usually use this folder for their temporary items, in preference to the Temporary Folder. Supported with Mac OS 8 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kApplicationSupportFolderType`
Specifies the Application Support folder in the System Folder. This folder contains code and data files needed by third-party applications. These files should usually not be written to after they are installed. In general, files deleted from this folder remove functionality from an application, unlike files in the Preferences folder, which should be non-essential. One type of file that could be placed here would be plug-ins that the user might want to maintain separately from any application, such as for an image-processing application that has many "fourth-party" plug-ins that the user might want to upgrade separately from the host application. Another type of file that might belong in this folder would be application-specific data files that are not preferences, such as for a scanner application that needs to read description files for specific scanner models according to which are currently available on the SCSI bus or network. Supported with Mac OS 8 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kTextEncodingsFolderType`
Specifies the Text Encodings folder in the System Folder. Supported with Mac OS 8 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kStationeryFolderType`
Specifies the OpenDoc stationery folder. Supported with Mac OS 8 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kOpenDocFolderType`
Specifies the OpenDoc root folder. Supported with Mac OS 8 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kOpenDocShellPlugInsFolderType`
Specifies the OpenDoc shell plug-ins folder in the OpenDoc folder. Supported with Mac OS 8 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kEditorsFolderType`
Specifies the OpenDoc editors folder in the Mac OS folder. Supported with Mac OS 8 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kOpenDocEditorsFolderType`
Specifies the OpenDoc subfolder in the Editors folder. Supported with Mac OS 8 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kOpenDocLibrariesFolderType`
Specifies the OpenDoc libraries folder. Supported with Mac OS 8 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kGenEditorsFolderType`
Specifies a general editors folder. Supported with Mac OS 8 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kHelpFolderType`
Specifies the Help folder in the System Folder. Supported with Mac OS 8 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kInternetPlugInFolderType`
Specifies the Browser Plug-ins folder in the System Folder. Supported with Mac OS 8 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kModemScriptsFolderType`
Specifies the Modem Scripts folder in the Extensions folder. Supported with Mac OS 8 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kPrinterDescriptionFolderType`
Specifies the Printer Descriptions folder in the Extensions folder. Supported with Mac OS 8 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kPrinterDriverFolderType`
Specifies the printer drivers folder. This constant is not currently supported.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kScriptingAdditionsFolderType`
Specifies the Scripting Additions folder in the System Folder. Supported with Mac OS 8 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kSharedLibrariesFolderType`
Specifies the general shared libraries folder. This constant is not currently supported.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kVoicesFolderType`
Specifies the Voices folder in the Extensions folder. Supported with Mac OS 8 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kControlStripModulesFolderType`
Specifies the Control Strip Modules folder in the System Folder. Supported with Mac OS 8 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kAssistantsFolderType`
Specifies the Assistants folder installed at the root level of the volume. Supported with Mac OS 8 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kUtilitiesFolderType`
Specifies the Utilities folder installed at the root level of the volume. Supported with Mac OS 8 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kAppleExtrasFolderType`
Specifies the Apple Extras folder installed at the root level of the volume. Supported with Mac OS 8 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kContextualMenuItemsFolderType`
Specifies the Contextual Menu Items folder in the System Folder. Supported with Mac OS 8 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kMacOSReadMesFolderType`
Specifies the Mac OS Read Me Files folder installed at the root level of the volume. Supported with Mac OS 8 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kALMModulesFolderType`
Specifies the Location Manager Modules folder in the Extensions Folder. Supported with Mac OS 8.1 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kALMPreferencesFolderType`
Specifies the Location Manager Prefs folder in the Preferences folder. Supported with Mac OS 8.1 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kALMLocationsFolderType`
Specifies the Locations folder in the Location Manager Prefs folder. Files containing configuration information for different locations are stored here. Supported with Mac OS 8.1 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kColorSyncProfilesFolderType`
Specifies the ColorSync Profiles folder in the System Folder. Supported with Mac OS 8.1 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kThemesFolderType`
Specifies the Theme Files folder in the Appearance folder. Supported with Mac OS 8.1 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kFavoritesFolderType`
Specifies the Favorites folder in the System Folder. This folder is for storing Internet location files, aliases, and aliases to other frequently used items. Facilities for adding items into this folder are found in Contextual Menus, the Finder, Navigation Services, and others. Supported with Mac OS 8.1 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kInternetFolderType`
Specifies the Internet folder installed at the root level of the volume. This folder is a location for saving Internet-related applications, resources, and tools. Supported with Mac OS 8.5 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kAppearanceFolderType`
Specifies the Appearance folder in the System Folder. Supported with Mac OS 8.5 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kSoundSetsFolderType`
Specifies the Sound Sets folder in the Appearance folder. Supported with Mac OS 8.5 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kDesktopPicturesFolderType`
Specifies the Desktop Pictures folder in the Appearance folder. This folder is used for storing desktop picture files. Files of type `'JPEG'` are auto-routed into this folder when dropped into the System Folder. Supported with Mac OS 8.5 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kInternetSearchSitesFolderType`
Specifies the Internet Search Sites folder in the System Folder. This folder contains Internet search site specification files used by the Find application when it accesses Internet search sites. Files of type `'issp'` are auto-routed to this folder. Supported with Mac OS 8.5 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kFindSupportFolderType`
Specifies the Find folder in the Extensions folder. This folder contains files used by the Find application. Supported with Mac OS 8.5 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kFindByContentFolderType`
Specifies the Find By Content folder installed at the root level of the volume. This folder is invisible and its use is private to Find By Content. Supported with Mac OS 8.5 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kInstallerLogsFolderType`
Specifies the Installer Logs folder installed at the root level of the volume. You can use this folder to save installer log files. Supported with Mac OS 8.5 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kScriptsFolderType`
Specifies the Scripts folder in the System Folder. This folder is for saving AppleScript scripts. Supported with Mac OS 8.5 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kFolderActionsFolderType`
Specifies the Folder Action Scripts folder in the Scripts folder. Supported with Mac OS 8.5 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kLauncherItemsFolderType`
Specifies the Launcher Items folder in the System Folder. Items in this folder appear in the Launcher control panel. Items included in folders with names beginning with a bullet (Option-8) character will appear as a separate panel in the Launcher window. Supported with Mac OS 8.5 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kRecentApplicationsFolderType`
Specifies the Recent Applications folder in the Apple Menu Items folder. Apple Menu Items saves aliases to recent applications here. Supported with Mac OS 8.5 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kRecentDocumentsFolderType`

Specifies the Recent Documents folder in the Apple Menu Items folder. Apple Menu Items saves aliases to recently opened documents here. Supported with Mac OS 8.5 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kRecentServersFolderType`

Specifies the Recent Servers folder in the Apple Menu Items folder. Apple Menu Items saves aliases to recently mounted servers here. Supported with Mac OS 8.5 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kSpeakableItemsFolderType`

Specifies the Speakable Items folder. This folder is for storing scripts and items recognized by speech recognition. Supported with Mac OS 8.5 and later.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

kDomainTopLevelFolderType

```
enum {
    kDomainTopLevelFolderType = 'dtop',
    kDomainLibraryFolderType = 'dlib',
    kColorSyncFolderType = 'sync',
    kColorSyncCMMFolderType = 'ccmm',
    kColorSyncScriptingFolderType = 'cscr',
    kPrintersFolderType = 'impr',
    kSpeechFolderType = 'spch',
    kCarbonLibraryFolderType = 'carb',
    kDocumentationFolderType = 'info',
    kDeveloperDocsFolderType = 'ddoc',
    kDeveloperHelpFolderType = 'devh',
    kISSDownloadsFolderType = 'issd',
    kUserSpecificTmpFolderType = 'utmp',
    kCachedDataFolderType = 'cach',
    kFrameworksFolderType = 'fram',
    kPrivateFrameworksFolderType = 'pfrm',
    kClassicDesktopFolderType = 'sdsk',
    kDeveloperFolderType = 'devf',
    kSystemSoundsFolderType = 'ssnd',
    kComponentsFolderType = 'cmpd',
    kQuickTimeComponentsFolderType = 'wcmp',
    kCoreServicesFolderType = 'csrv',
    kPictureDocumentsFolderType = 'pdoc',
    kMovieDocumentsFolderType = 'mdoc',
    kMusicDocumentsFolderType = 'doc',
    kInternetSitesFolderType = 'site',
    kPublicFolderType = 'pubb',
    kAudioSupportFolderType = 'adio',
    kAudioSoundsFolderType = 'asnd',
    kAudioSoundBanksFolderType = 'bank',
    kAudioAlertSoundsFolderType = 'alrt',
    kAudioPlugInsFolderType = 'aplg',
    kAudioComponentsFolderType = 'acmp',
    kKernelExtensionsFolderType = 'kext',
    kDirectoryServicesFolderType = 'dsrv',
    kDirectoryServicesPlugInsFolderType = 'dplg',
    kInstallerReceiptsFolderType = 'rcpt',
    kFileSystemSupportFolderType = 'fsys',
    kAppleShareSupportFolderType = 'shar',
    kAppleShareAuthenticationFolderType = 'auth',
    kMIDIDriversFolderType = 'midi',
    kKeyboardLayoutsFolderType = 'klay',
    kIndexFilesFolderType = 'indx',
    kFindByContentIndexesFolderType = 'fbcx',
    kManagedItemsFolderType = 'mang',
    kBootTimeStartupItemsFolderType = 'empz'
};
```

kAppleshareAutomountServerAliasesFolderType

```
enum {
    kAppleshareAutomountServerAliasesFolderType = 'srvƒ',
    kPreMacOS91ApplicationsFolderType = 'åpps',
```

```
    kPreMacOS91InstallerLogsFolderType = 'îlgf',
    kPreMacOS91AssistantsFolderType = 'åstƒ',
    kPreMacOS91UtilitiesFolderType = 'ütiƒ',
    kPreMacOS91AppleExtrasFolderType = 'åexƒ',
    kPreMacOS91MacOSReadMesFolderType = 'orƒ',
    kPreMacOS91InternetFolderType = 'întƒ',
    kPreMacOS91AutomountedServersFolderType = 'ßrvƒ',
    kPreMacOS91StationeryFolderType = 'ødst'
};
```

kUsersFolderType

```
enum {
    kUsersFolderType = 'usrs',
    kCurrentUserFolderType = 'cusr',
    kCurrentUserRemoteFolderLocation = 'rusf',
    kCurrentUserRemoteFolderType = 'rusr',
    kSharedUserDataFolderType = 'sdat',
    kVolumeSettingsFolderType = 'vsfd'
};
```

kLocalesFolderType

```
enum {
    kLocalesFolderType = 'ƒloc',
    kFindByContentPluginsFolderType = 'fbcp'
};
```

Disk and Domain Constants

Identify the disk or domain in which to locate a folder.

```
enum {
    kOnSystemDisk = -32768L,
    kOnAppropriateDisk = -32767,
    kSystemDomain = -32766,
    kLocalDomain = -32765,
    kNetworkDomain = -32764,
    kUserDomain = -32763,
    kClassicDomain = -32762
};
enum {
    kLastDomainConstant = kUserDomain
};
```

Constants

`kOnSystemDisk`

Specifies the system disk.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

`kOnAppropriateDisk`
In most cases, the equivalent of `kOnSystemDisk`. On Mac OS X, use this constant instead of the constant `kOnSytemDisk` to indicate any disk.

Available in Mac OS X v10.0 and later.

Declared in `Folders.h`.

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

Declared in `Folders.h`.

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

Declared in `Folders.h`.

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

Declared in `Folders.h`.

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

Declared in `Folders.h`.

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

Declared in `Folders.h`.

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

Declared in `Folders.h`.

Discussion
You can pass this constant in the `vRefNum` parameter of `FSFindFolder` (page 12) to locate a folder on the startup disk.

Notification Options

Specify options for the `FolderManagerRegisterNotificationProc` function.

```
enum {
    kDoNotRemoveWhenCurrentApplicationQuitsBit = 0,
    kDoNotRemoveWheCurrentApplicationQuitsBit =
kDoNotRemoveWhenCurrentApplicationQuitsBit
};
```

Constants
`kDoNotRemoveWhenCurrentApplicationQuitsBit`
Tells the Folder Manager to not remove your notification function when the current application quits. Otherwise, a notification function registered within an application's context will be automatically removed when that application quits. Programs that register notifications at system startup should set this bit.

Available in Mac OS X v10.0 through Mac OS X v10.4.

Declared in `Folders.h`.

`kDoNotRemoveWheCurrentApplicationQuitsBit`
Use `kDoNotRemoveWhenCurrentApplicationQuitsBit` instead.
Available in Mac OS X v10.0 through Mac OS X v10.4.
Declared in `Folders.h`.

FSFindFolderExtended Flags

Specify additional options for folder searches performed with the `FSFindFolderExtended` function.

```
enum {
    kFindFolderExtendedFlagsDoNotFollowAliasesBit = 0,
    kFindFolderExtendedFlagsDoNotUseUserFolderBit = 1,
    kFindFolderExtendedFlagsUseOtherUserRecord = 0x01000000
};
```

Discussion
These are passed to `FSFindFolderExtended` (page 45) and `FindFolderExtended` (page 43) in the `flags` field.

FindFolderUserRedirectionGlobals Flags

Used in the `flags` field of the `FindFolderUserRedirectionGlobals` structure

```
enum {
    kFindFolderRedirectionFlagUseDistinctUserFoldersBit = 0,
    kFindFolderRedirectionFlagUseGivenVRefAndDirIDAsUserFolderBit
= 1,
    kFindFolderRedirectionFlagsUseGivenVRefNumAndDirIDAsRemoteUserFolderBit
= 2
};
typedef UInt32 RoutingFlags;
```

FindFolderUserRedirectionGlobals Structure Version

Represents the current version of the `FindFolderUserRedirectionGlobals` structure.

```
enum {
    kFolderManagerUserRedirectionGlobalsCurrentVersion = 1
};
```

Notification Messages

Define messages sent to your notification function.

```
enum {
    kFolderManagerNotificationMessageUserLogIn = 'log+',
    kFolderManagerNotificationMessagePreUserLogIn = 'logj',
    kFolderManagerNotificationMessageUserLogOut = 'log-',
    kFolderManagerNotificationMessagePostUserLogOut = 'logp',
    kFolderManagerNotificationDiscardCachedData = 'dche',
    kFolderManagerNotificationMessageLoginStartup = 'stup'
};
```

Constants

`kFolderManagerNotificationMessageUserLogIn`
Sent when a user has logged in. When you receive this message `FindFolder` will return the `vRefNum` and `dirID` of the user's redirected folders until the user logs out. This message can be used to load the new user's preferences.

Available in Mac OS X v10.0 through Mac OS X v10.4.

Declared in `Folders.h`.

`kFolderManagerNotificationMessagePreUserLogIn`
Sent just prior to redirecting `FindFolder` to the user's folders. Calling `FindFolder` when receiving this notification will return the `vRefNum` and `dirID` of the system folders. This message can be used to update the owner's preference files prior to `FindFolder` being redirected.

Available in Mac OS X v10.0 through Mac OS X v10.4.

Declared in `Folders.h`.

`kFolderManagerNotificationMessageUserLogOut`
Sent when a user has logged out. This is the last time `FindFolder` will return the user's folders; after this notification `FindFolder` will return the `vRefNum` and `dirID` of system folders. This message can be used to update a user's preference files during logout.

Available in Mac OS X v10.0 through Mac OS X v10.4.

Declared in `Folders.h`.

`kFolderManagerNotificationMessagePostUserLogOut`
Sent just after `FindFolder` has been restored to return the `vRefNum` and `dirID` of system folders. This message can be used to load the owner's preferences.

Available in Mac OS X v10.0 through Mac OS X v10.4.

Declared in `Folders.h`.

`kFolderManagerNotificationDiscardCachedData`
Sent by third-party software when the entire Folder Manager cache should be flushed.

Available in Mac OS X v10.0 through Mac OS X v10.4.

Declared in `Folders.h`.

`kFolderManagerNotificationMessageLoginStartup`
Available in Mac OS X v10.0 through Mac OS X v10.4.

Declared in `Folders.h`.

FolderManagerCallNotificationProcs Options

Used in the `options` parameter of `FolderManagerCallNotificationProcs`.

```
enum {
    kStopIfAnyNotificationProcReturnsErrorBit = 31
};
```

Result Codes

The most common result codes returned by Folder Manager are listed below.

Result Code	Value	Description
`badFolderDescErr`	-4270	Invalid folder Available in Mac OS X v10.0 and later.
`duplicateFolderDescErr`	-4271	Duplicate folders for a particular routing Available in Mac OS X v10.0 and later.
`noMoreFolderDescErr`	-4272	Available in Mac OS X v10.0 and later.
`invalidFolderTypeErr`	-4273	Invalid folder name Available in Mac OS X v10.0 and later.
`duplicateRoutingErr`	-4274	Same routing for two folders Available in Mac OS X v10.0 and later.
`routingNotFoundErr`	-4275	No routing set up for the folder passed in Available in Mac OS X v10.0 and later.
`badRoutingSizeErr`	-4276	Incorrect `descSize` field of the folder routing structure Available in Mac OS X v10.0 and later.

Gestalt Constants

You can check for version and feature availability information by using the Folder Manager selectors defined in the Gestalt Manager. For more information see *Inside Mac OS X: Gestalt Manager Reference*.

Deprecated Folder Manager Functions

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

Deprecated in Mac OS X v10.3

FindFolderExtended

Obtains location information for system-related directories. (Deprecated in Mac OS X v10.3. Use `FindFolder` (page 9) instead.)

```
OSErr FindFolderExtended (
   FSVolumeRefNum vRefNum,
   OSType folderType,
   Boolean createFolder,
   UInt32 flags,
   void *data,
   FSVolumeRefNum *foundVRefNum,
   SInt32 *foundDirID
);
```

Parameters

foldType

createFolder

flags

Return Value

A result code. See "Folder Manager Result Codes" (page 42).

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.3.

Not available to 64-bit applications.

Declared In

`Folders.h`

FolderManagerRegisterCallNotificationProcs

Calls the registered Folder Manager notification procs. (Deprecated in Mac OS X v10.3. There is no replacement function.)

```
OSStatus FolderManagerRegisterCallNotificationProcs (
   OSType message,
   void *arg,
   UInt32 options
);
```

Parameters

message

options

Return Value
A result code. See "Folder Manager Result Codes" (page 42).

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.3.
Not available to 64-bit applications.

Declared In
`Folders.h`

FolderManagerRegisterNotificationProc

Registers your notification function with the Folder Manager. (Deprecated in Mac OS X v10.3. There is no replacement function.)

```
OSErr FolderManagerRegisterNotificationProc (
   FolderManagerNotificationUPP notificationProc,
   void *refCon,
   UInt32 options
);
```

Parameters

notificationProc
A UPP to your notification function.

refCon
A pointer to client-defined data. This value is passed to your notification function each time it is called.

options
A value specifying registration options. See FolderManagerCallNotificationProcs Options (page 41).

Return Value
A result code. See "Folder Manager Result Codes" (page 42).

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.3.
Not available to 64-bit applications.

Declared In
`Folders.h`

FolderManagerUnregisterNotificationProc

Removes your notification function from the Folder Manager's queue. (Deprecated in Mac OS X v10.3. There is no replacement function.)

```
OSErr FolderManagerUnregisterNotificationProc (
   FolderManagerNotificationUPP notificationProc,
   void *refCon
);
```

Parameters

notificationProc

The UPP to your notification function that you passed to the `FolderManagerRegisterNotificationProc` function.

refCon

A pointer to the same value that you passed to the `FolderManagerRegisterNotificationProc` function in the *refCon* parameter.

Return Value

A result code. See "Folder Manager Result Codes" (page 42).

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.3.

Not available to 64-bit applications.

Declared In

`Folders.h`

FSFindFolderExtended

Locates a system-related folder and returns a reference to the folder. (Deprecated in Mac OS X v10.3. Use `FSFindFolder` (page 12) instead.)

```
OSErr FSFindFolderExtended (
   FSVolumeRefNum vRefNum,
   OSType folderType,
   Boolean createFolder,
   UInt32 flags,
   void *data,
   FSRef *foundRef
);
```

Parameters

vRefNum

The volume reference number or domain in which you want to locate a folder. To specify the startup disk, use the constant `kOnSystemDisk`. To specify a domain, use a domain constant such as `kUserDomain`. See Disk and Domain Constants (page 38).

foldType

The type of folder you want to find. See Folder Type Constants (page 26).

createFolder

A value of type `Boolean`, as defined in Create Folder Flags (page 22). Pass the constant `kCreateFolder` to create a folder if it does not already exist; otherwise, pass the constant `kDontCreateFolder`.

flags

An extended behavior constant. See FSFindFolderExtended Flags (page 40).

data

User data which is interpreted differently depending on the constant specified in the `flags` parameter.

foundRef

A pointer to a `FSRef` variable. On return, the variable contains a file system reference to the specified folder.

Return Value
A result code. See "Folder Manager Result Codes" (page 42).

Discussion
The specified folder might be relocated in future versions of system software; therefore, do not assume the volume that you specify in the `vRefNum` constant and the volume returned in the file system reference will be the same.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.3.
Not available to 64-bit applications.

Declared In
`Folders.h`

GetFolderDescriptor

Obtains the folder descriptor information for the specified folder type from the global descriptor list. (Deprecated in Mac OS X v10.3. There is no replacement function.)

```
OSErr GetFolderDescriptor (
   FolderType foldType,
   Size descSize,
   FolderDesc *foldDesc
);
```

Parameters

foldType

Pass a constant identifying the type of the folder for which you wish to get descriptor information. See Folder Type Constants (page 26).

descSize

Pass the size (in bytes) of the folder descriptor structure for which a pointer is passed in the `foldDesc` parameter. This value is needed in order to determine the version of the structure being used.

foldDesc

Pass a pointer to a folder descriptor structure. On return, the folder descriptor structure contains information from the global descriptor list for the specified folder type.

Return Value
A result code. See "Folder Manager Result Codes" (page 42).

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.3.
Not available to 64-bit applications.

Declared In
`Folders.h`

ReleaseFolder

Releases the Trash folder in preparation for unmounting a server volume. (Deprecated in Mac OS X v10.3. This function is not needed in Mac OS X.)

```
OSErr ReleaseFolder (
   FSVolumeRefNum vRefNum,
   OSType folderType
);
```

Parameters

vRefNum

Pass the volume reference number of the server volume on which you want to release the Trash folder.

folderType

Always pass the `kTrashFolderType` constant. Other folder types are currently ignored.

Return Value
A result code. See "Folder Manager Result Codes" (page 42).

Discussion
When you call `FindFolder` (page 9) with the `kTrashFolderType` constant, it opens a file on a server volume that ensures each server volume user gets a unique Trash folder. Because a server volume's Trash folder may contain files or folders put there by the user, applications should delete the contents of the server volume's Trash folder. To do this, before your application unmounts a server volume, your application should call `ReleaseFolder`, or the `UnmountVol` request could fail with a `fBsyErr` result code. `ReleaseFolder` closes the file `FindFolder` may have opened and releases the Trash folder on that volume.

Your application should not use this function unless you want to unmount one or more server volumes. Normally, applications should not unmount servers; they should let users use the Finder to unmount volumes. In particular, applications should have no need to release the Trash folder explicitly; rather, unmounting volumes should be left to users to do with the Finder or by restarting.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.3.

Declared In
`Folders.h`

Deprecated in Mac OS X v10.4

AddFolderRouting

Adds a folder routing structure to the global routing list. (Deprecated in Mac OS X v10.4. There is no replacement function.)

```
OSErr AddFolderRouting (
   OSType fileType,
   FolderType routeFromFolder,
   FolderType routeToFolder,
   RoutingFlags flags,
   Boolean replaceFlag
);
```

Parameters

fileType

Pass the `OSType` of the file to be routed.

routeFromFolder

Pass the folder type of the "from" folder see Folder Type Constants (page 26) for descriptions of possible values. An item dropped on the folder specified in this parameter will be routed to the folder specified in the `routeToFolder` parameter.

routeToFolder

The folder type of the "to" folder see Folder Type Constants (page 26) for descriptions of possible values.

flags

Reserved for future use; pass 0.

replaceFlag

Pass a `Boolean` value indicating whether you wish to replace a folder routing that already exists. If `true`, it replaces the folder to which the item is being routed. If `false`, it leaves the folder to which the item is being routed.

Return Value

A result code. See "Folder Manager Result Codes" (page 42). The result code `duplicateRoutingErr` indicates that a folder routing is already installed with the specified folder type and `replaceFlag` is false.

Discussion

Your application can use the `AddFolderRouting` function to specify how the Finder routes a given file type.

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

`Folders.h`

FindFolderRouting

Finds the destination folder from a matching folder routing structure for the specified file. (Deprecated in Mac OS X v10.4. There is no replacement function.)

```
OSErr FindFolderRouting (
   OSType fileType,
   FolderType routeFromFolder,
   FolderType *routeToFolder,
   RoutingFlags *flags
);
```

Parameters

fileType

Pass the file type specified in the appropriate folder routing structure for the file for which you wish to find a destination folder.

routeFromFolder

Pass the folder type of the "from" folder for which you wish to find a "to" folder see Folder Type Constants (page 26) for descriptions of possible values. An item dropped on the folder specified in this parameter will be routed to the folder specified in the `routeToFolder` parameter.

routeToFolder

A pointer to a value of type `FolderType`. On return, the value is set to the folder type of the destination folder.

flags

Reserved; pass 0.

Return Value

A result code. See "Folder Manager Result Codes" (page 42).

Discussion

Both the file type and the folder type specified must match those of a folder routing structure in the global routing list for the `FindFolderRouting` function to succeed.

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

`Folders.h`

GetFolderRoutings

Obtains folder routing information from the global routing list. (Deprecated in Mac OS X v10.4. There is no replacement function.)

```
OSErr GetFolderRoutings (
   UInt32 requestedRoutingCount,
   UInt32 *totalRoutingCount,
   Size routingSize,
   FolderRouting *theRoutings
);
```

Parameters

requestedRoutingCount

An unsigned 32-bit value. Pass the number of folder routing structures that can fit in the buffer pointed to by the `theRoutings` parameter.

totalRoutingCount

A pointer to an unsigned 32-bit value. On return, the value is set to the number of folder routing structures in the global list. If this value is less than or equal to `requestedRoutingCount`, all folder routing structures were returned to the caller.

routingSize

Pass the size (in bytes) of the `FolderRouting` structure.

theRoutings

Pass a pointer to an array of `FolderRouting` (page 19) structures. On return the structure(s) contain the requested routing information. You may pass `null` if you do not wish this information.

Return Value
A result code. See "Folder Manager Result Codes" (page 42).

Discussion
The folder routing information in the global routing list determines how the Finder routes files.

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
`Folders.h`

RemoveFolderRouting

Deletes a folder routing structure from the global routing list. (Deprecated in Mac OS X v10.4. There is no replacement function.)

```
OSErr RemoveFolderRouting (
   OSType fileType,
   FolderType routeFromFolder
);
```

Parameters

fileType

Pass the file type value contained in the folder routing structure to be removed.

routeFromFolder

Pass the folder type of the "from" folder see Folder Type Constants (page 26) for descriptions of possible values.

Return Value
A result code. See "Folder Manager Result Codes" (page 42).

Discussion
Both the file type and the folder type specified must match those of an existing folder routing structure for the *RemoveFolderRouting* function to succeed.

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
Folders.h

Deprecated in Mac OS X v10.5

FSpDetermineIfSpecIsEnclosedByFolder

Determines whether a file of type FSSpec is enclosed inside a special folder type for the given domain. (Deprecated in Mac OS X v10.5.)

```
OSErr FSpDetermineIfSpecIsEnclosedByFolder (
   FSVolumeRefNum domainOrVRefNum,
   OSType folderType,
   const FSSpec *inSpec,
   Boolean *outResult
);
```

Parameters

domainOrVRefNum
The domain or volume reference number to check. For information about the possible domains, see Disk and Domain Constants (page 38). You can also pass 0 to check all domains and volumes, or you can pass kOnAppropriateDisk to check the appropriate volume for the specified file.

folderType
The special folder type to check. For information about the possible folder types, see Folder Type Constants (page 26).

inSpec
The file for which to search.

outResult
A pointer to a Boolean variable. On return, indicates whether or not the file is enclosed inside the special folder type for the given domain.

Discussion
This function provides an efficient way to check to see if a file (or folder) is inside a special folder for a given domain. A typical use for this function is to determine if a given file is inside the trash on a volume:

```
err = FSpDetermineIfSpecIsEnclosedByFolder (kOnAppropriateDisk, kTrashFolderType,
 &spec, &result);
```

Availability
Available in Mac OS X v10.4 and later.
Deprecated in Mac OS X v10.5.
Not available to 64-bit applications.

Declared In
`Folders.h`

GetFolderName

Obtains the name of the specified folder. (Deprecated in Mac OS X v10.5.)

```
OSErr GetFolderName (
   FSVolumeRefNum vRefNum,
   OSType foldType,
   FSVolumeRefNum *foundVRefNum,
   StrFileName name
);
```

Parameters

vRefNum
Pass the volume reference number (or the constant `kOnSystemDisk` for the startup disk) of the volume containing the folder for which you wish the name to be identified.

foldType
Pass a constant identifying the type of the folder for which you wish the name to be identified. See Folder Type Constants (page 26).

foundVRefNum
On return, a pointer to the volume reference number for the volume containing the folder specified in the `foldType` parameter.

name
On return, a string containing the title of the folder specified in the `foldType` and `vRefNum` parameters.

Return Value
A result code. See "Folder Manager Result Codes" (page 42).

Discussion
The `GetFolderName` function obtains the name of the folder in the folder descriptor, not the name of the folder on the disk. The names may differ for a few special folders such as the System Folder. For relative folders, however, the actual name is always returned. You typically do not need to call this function.

Availability
Available in Mac OS X v10.0 and later.
Deprecated in Mac OS X v10.5.
Not available to 64-bit applications.

Declared In
`Folders.h`

Document Revision History

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

Date	Notes
2006-07-12	Made minor formatting changes.
2006-07-24	Updated availability information.
2006-04-04	Added deprecation information.
	Added descriptions of three functions: `FSpDetermineIfSpecIsEnclosedByFolder` (page 51), `FSDetermineIfRefIsEnclosedByFolder` (page 11), and `DetermineIfPathIsEnclosedByFolder` (page 8).
2005-08-11	Added information about the function FSFindFolderExtended.
2003-02-18	Incorporated documentation for `FolderManagerRegisterNotificationProc`, `FolderManagerRegisterNotificationProc`, and `FolderManagerNotificationProcPtr`.

Document Revision History

Index

K

M

N

R