MiscFileFinder

Inherits From: Object

Declared In: misckit/MiscFileFinder.h

Class Description

The MiscFileFinder Class is a service provider. Each MiscFileFinder object defines a type of file and contains all the information on where to look for them. If the MiscFileFinder object is named, it is kept in a hash list and no other object may be created with the same name. An named instance can be found by name.

The MiscFileFinder handles all the messy lookup activities that are associated with files in large file systems. It is often the case that a file might reside in one of many places, or that a file earlier in a search list should be used in preference to a later one. And even if a file is found earlier in the search list than another by the same name, it does little good if the first file cannot be accessed in the way requiredÐread, write, etc.

1) If *filename* is an absolute path name, ie it begins with "/", it is used as is.
2) The paths in your environment variable (*pathVariable*) are searched.
3) The *defaultPath* in this class as set by you. If you have never set it, it will be the same as the contents of the environment variable named by *pathVariable*.

The defaultPath is a compile time colon seperated list of places to look, in the order they are to be searched. The environment variable named in *pathVariable* allows runtime change to this search sequence. By creating and initing this environment variable, you can override the selection that the compiled in search list would make.

The first full path found that allows the required access is the one returned. A typical situation: you have a user copy of a file in a standard placeÐsuch as one of the App directoriesÐand you are debugging the same app in a working directory. A file that otherwise matches will be bypassed if the sender does not have at least the capabilities list that is or'd together in *fileMode*.

The following sample code shows a typical usage:

```
if (![MiscFileFinder findFileTypeNamed: "Cursors"])
    [[MiscFileFinder alloc] initName: "Cursors"
                         defaultPath: "~/Library/Cursors:/Library/Cursors:/LocalLibrary/Cursors"
                        pathVariable: "IMAGEPATH"
                                mode: R_OK];
file = [[MiscFileFinder findFileTypeNamed: "Cursors"] fullPathForFile: "cursor.tiff"];
```

Instance Variables

const char ***typeName**;

const char ***defaultPath**;
const char ***pathVariable**;
int **fileMode**;
char ***pathTmp**;

typeName	Name of this type object
defaultPath	Default path for this type.
pathVariable	Environment PATH variable name.
fileMode	Required attributes of file type: R_OK, X_OK, etc.
pathTmp	Private.

Method Types

Initializing and freeing	- initName:defaultPath:pathVariable:mode: - init - free
FileType Name	+ findFileTypeNamed: - setTypeName: - typeName
File Lookup	- fullPathForFile: - setDefaultPath: - defaultPath - setPathVariable: - pathVariable - setFileMode: - fileMode

Diagnostics	+ setDebug: + isDebug:
Archiving	- read: - write:

Class Methods

debug
+ **(BOOL)debug**

Returns YES if the debug flag is set.

See also: + **setDebug**

findFileTypeNamed:
+ **findFileTypeNamed:**(const char*)*aKey*

If there is a MiscFileFinder object with the *typeName* of *aKey*, return it. Otherwise return nil.

See also: - **typeName,** - **setTypeName:**

setDebug:
+ **setDebug:**(BOOL)*flag*

Turn diagnostic printouts on or off for all objects of this class. YES means on. If on, a localized message is printed on the Workspace console or on the GDB console each time one of the debug trace points is executed. Returns **self**.

See also: + **debug**

Instance Methods

defaultPath
- (const char*)**defaultPath**

Returns the *defaultPath*.

See also: - **fullPathForFile:**, - **pathVariable**, - **fileMode**, - **setDefaultPath:**, - **setPathVariable:**, - **setFileMode:**

fileMode
- (int)**fileMode**

Returns the *fileMode*.

See also: - **defaultPath**, - **pathVariable**, - **fullPathForFile:**, - **setDefaultPath:**, - **setPathVariable:**, - **setFileMode:**

free

- **free**

Free the MiscFileFinder object.

See also: - **initFrame:**, - **init**

fullPathForFile:

- (char *)**fullPathForFile:**(const char *)*filename*

Return a full pathname if we successfully find a file with the specified name. The following places are searched, in order:

1) If *filename* is an absolute path name, ie it begins with "/", it is used as is.
2) The paths in your environment variable (*pathVariable*) are searched.
3) The *defaultPath* in this Class as set by you. If you have never set it, it will be the same as the contents of your
the environment variable named by *pathVariable*.

The first path found is the one returned. A typical situation: you have a user copy of a file in a standard placeÐsuch as one of the App directoriesÐand you are debugging the same app in a working directory. A file that otherwise matches will be bypassed if the sender does not have at least the capabilities listed in *fileMode*.

Returns a pathname string allocated from the default zone. Freeing it is the senders responsibility.

See also: - **defaultPath,** - **pathVariable,** - **fileMode,** - **setDefaultPath:,** - **setPathVariable:,** - **setFileMode:**

init

- **init**

Create a MiscFileFinder with a NULL *typeName*. and a NULL *defaultPath*. Uses the PATH environment variable and requires files allow at least R_OK mode by the sender. Returns **self**.

See also: - **free**, - **initFrame:**

initName:defaultPath:pathVariable:mode:

- **initName:**(const char *)*name* **defaultPath:**(const char *)*path* **pathVariable:**(const char *)*var* **mode:** (int)*fmode*

Designated initializer. Create a MiscFileFinder with a *typeName* of *name*; a default search path named *path;* an environment path name variable *var* and *fileMode* of *fmode*. Mode can be any combination of R_OK, W_OK, and so forth (see ctypes.h). All strings are copied into local storage and managed by the MiscFileFinder object thereafter. If *name* already exists a nil is returned (a special allowance is made for reinitializing an existing object). Once it has been set, the name of a file type is unchangeable. Returns **self**.

See also: - **free**, - **initFrame:**

pathVariable

- (const char*)**pathVariable**

Returns the *pathVariable*.

See also: - **defaultPath,** - **fullPathForFile:,** - **fileMode,** - **setDefaultPath:,** - **setPathVariable:,** - **setFileMode:**

read:
- **read:**(NXTypedStream *)*stream*

Reads the object from the typed stream *stream*. Returns **self**.

See also: - **write:**

typeName
- (const char*)**typeName**

Returns the *typeName*.

See also: - **setTypeName:**, + **findFileTypeNamed:**

setDefaultPath:
- **setDefaultPath:**(const char*)*path*

Set the *defaultPath* by copying *path*. The *defaultPath* is a colon seperated list of directory path names. If an old *defaultPath* exists, it is freed. Returns **self**

See also: - **defaultPath**, - **pathVariable**, - **fileMode**, - **fullPathForFile:**, - **setPathVariable:**, - **setFileMode:**

setFileMode:
- **setFileMode:**(int)*mode*

Set the *fileMode* requirements. Any file found must allow the sender at least the capabilities in *fileMode* for it to

be acceptable and ªfoundº. Returns **self**

See also: - **defaultPath,** - **pathVariable,** - **fileMode,** - **setDefaultPath:,** - **setPathVariable:,** -**fullPathForFile:**

setPathVariable:

- **setPathVariable:**(const char*)*var*

Set the *pathVariable* by copying *var*. The *pathVariable* is the name of an envirnoment variable that contains a colon seperated list of directory path names. If an old *pathVariable* exists, it is freed. Returns **self**

See also: - **defaultPath,** - **pathVariable,** - **fileMode,** - **setDefaultPath:,** - **fullPathForFile:,** - **setFileMode:**

setTypeName:

- (BOOL)**setTypeName:**(const char*)*name*

Set the *typeName* by copying *name*. The *typeName* allows lookup of MiscFileFinder instances by name. If an old *typeName* exists, it is freed. Once it has been set, the name of a file type is unchangeable. Thus setTypeName: would only be of use if an object was initially created with a NULL typeName. Returns YES if the name was set.

See also: - **typeName,** + **findFileTypeNamed:**

write:

- **write:**(NXTypedStream *)*stream*

Writes the object to the typed stream *stream*. Returns **self**.

See also: - read: