

MiscFile (Searching)

Inherits From:

Declared In: <misckit/MiscFile.h>

Category Description

Implements slow and inefficient (but working) searching through a hierarchy of MiscFile instances. You can either search the hierarchy for a specific filename (using the searchFor* methods), or use the very flexible searchFiles* methods. The searchFiles* methods use the class delegate to ask if a specific file should be added to the list of returned matches. You can therefore create a list of all files that have a certain extension, or have a given set of permissions, etc. The class delegate implements the method addFile: (MiscFile *)file. If the file should be added to the returned list, the delegate method should return YES.

Method Types

- | | |
|------------------------------|--------------------------------------|
| Searching via class delegate | - searchFiles |
| | - searchFilesAndRecurse: |
| | - searchFilesAndRecurse:followLinks: |

Searching for a file

- searchFor:
- searchFor:recursive:
- searchFor:recursive:followLinks:

Instance Methods

searchFiles

- (List *)**searchFiles**

Equivalent to **searchFilesAndRecurse:** NO **followLinks:** NO.

See also: \pm **searchFilesAndRecurse:**, \pm **searchFilesAndRecurse:followLinks:**

searchFilesAndRecurse:

- (List *)**searchFilesAndRecurse:**(BOOL)*recursive*

Equivalent to **searchFilesAndRecurse:** *recursive* **followLinks:** NO.

See also: \pm **searchFiles**, \pm **searchFilesAndRecurse:followLinks:**

searchFilesAndRecurse:followLinks:

- (List *)**searchFilesAndRecurse:**(BOOL)*recursive*
followLinks:(BOOL)*followLinks*

Starts the search at the receiver and sends the class delegate a message for each file in the directory tree to see if the file should be added to the list returned. The class delegate needs only to implement the method (BOOL)**addFile:** (MiscFile *)*aFile*. The delegate method then returns YES if the file should be returned to the list. Also, if *followLinks* is YES, then any symbolic links to directories will be followed. You are responsible for freeing the list and it's contents (MiscFiles) when you are done with it.

See also: \pm **searchFiles**, \pm **searchFilesAndRecurse:**

searchFor:

- (MiscFile *)**searchFor:**(const char *)*filename*

Equivalent to **searchFor:** *filename* **recursive:** NO **followLinks:** NO

See also: \pm **searchFor:recursive:**, \pm **searchFor:recursive:followLinks:**

searchFor:recursive:

- (MiscFile *)**searchFor:**(const char *)*filename*
recursive:(BOOL)*recursive*

Equivalent to **searchFor:** *filename* **recursive:** *recursive* **followLinks:** NO.

See also: \pm **searchFor:**, \pm **searchFor:recursive:followLinks:**

searchFor:recursive:followLinks:

- (MiscFile *)**searchFor:**(const char *)*filename*
recursive:(BOOL)*recursive*
followLinks:(BOOL)*followLinks*

Searches for *filename*, starting at the receiver (which has to represent a directory). If *recursive* is YES, then the search for *filename* will recursively check directories until the file is found. If *followLinks* is YES, then any symbolic link will also be followed. If it is found, a MiscFile representing it is returned. If no match is found, nil is returned. You are responsible for freeing the object returned.

See also: \pm **searchFor:**, \pm **searchFor:recursive:**