FileXtra v2.0.5 Docs

26-May-97

FileXtra v2 is a free extension for use with MacroMedia Director v5 or later that performs file, directory and disk operations such as copy, delete and rename.

Little Planet Publishing
5045 Hillsboro Road
Nashville, TN 37215

Program and documentation were written by Kent Kersten, Little Planet Publishing (kent@littleplanet.com).

Please be aware that the Xtras, sample movies and documentation are all provided “as-is”, without any kind of warranty or suitability statements whatsoever. You use them at your own risk and we are in no way responsible for any consequences.

If you find this Xtra useful and are feeling generous, I would personally be very grateful if you would make a contribution to the following non-profit organization:

> Nashville Children’s Theatre
> 724 Second Avenue South
> Nashville, TN 37210
> (615) 254-9103

This is an organization with a long history of serving the children of Nashville with arts entertainment and education. I would suggest a donation of $20, but any amount would be appreciated. Please note that making a contribution is NOT a requirement to use this Xtra.

Target platforms:

> Macintosh 68K
> Macintosh PowerPC
> Windows 3.1
> Windows 95/NT

The Macintosh versions are combined in a single “FAT” binary Xtra named FileXtra. The Windows 3.1 version is named FileXtra.x16 (for 16-bit Windows platforms) and the Windows 95/NT version is named FileXtra.x32 (for 32-bit Windows platforms).

How to use it:

Simply drop it into the Xtras folder or directory that your copy of Director uses.

For authoring on the Macintosh, place the Xtra into the folder named “Xtras” that resides in the folder where the Director application is installed. For use with Macintosh projectors, create a folder named “Xtras” that resides in the folder where the projector is installed and place FileXtra there.

For authoring on the PC, place FileXtra.x16 into the directory named “Xtras” that resides in the directory where the 16-bit Director application is installed. For use with 16-bit PC projectors, create a directory named “Xtras” that resides in the directory where the projector is installed and place FileXtra.x16 there.

For the 32-bit PC version of Director, place FileXtra.x32 into the directory named “Xtras” that resides in the directory where the 32-bit Director application is installed. For use with 32-bit PC projectors, create a directory named “Xtras” that resides in the directory where the projector is installed and place FileXtra.x32 there.

To get a quick overview of the available functions, type the following into Director’s Message window:

put mMessageList(xtra “FileXtra”)

Initialization:

There are no “initialization” or “mNew” calls to make if you place the Xtra into the proper directory (see above). The new handlers simply exist when Director is launched. Likewise, there is no cleanup or “mDispose” calling to do.

General notes:

1) On the Macintosh it sometimes is necessary when installing new Xtras to delete the file “Director 5 Xtra Cache” that lives in the System Folder’s “Preferences” folder. On occasion Director will fail to recognize new Xtras until you exit, delete the cache file, then restart Director.

2) FileXtra was formerly named “FileUtil”. Delete the older, less adequate version when installing.

3) On a PC, both the .x16 and .x32 files can be together. Depending upon whether you are running 16- or 32-bit Director, the correct file will be used automatically.

4) The old version of the Xtra had a Macintosh-only call, *VolumesToList*, that is now present in both PC and Mac versions and has been renamed to *DrivesToList*.

5) Please see the FileXtra.dir movie for more examples.

Supported calls:

Drive Functions:

DriveExists(driveName)

Pass the name of a drive to see if it exists. If it does, a 0 is returned.

Windows example:

retval = DriveExists("D:")

Mac example:

retval = DriveExists("Macintosh HD:")

DriveFreeSpace(driveName)

Returns the number of bytes free on the specified drive.

DriveIsCDROM(driveName)

Returns 0 if the specified drive is a CD-ROM drive, a negative error code if not.

***Windows 3.1 note**: This call is sometimes confused by network drives and will report those as CD-ROM drives. Windows 95/NT does not share this weakness.*

DrivesToList()

Returns a Director list of the available drives on the system. This includes drives that are removable and have no media currently mounted.

***Macintosh note:** This call was originally named VolumesToList.*

Example:

set dlist to DrivesToList()

File Functions

FileExists(filename)

Pass a filename as a character string to this function and it will return 0 if it exists.

***Windows note**: You can pass wildcards in the filename string, such as "config.*" and if any matches are found, it will return a 0 (success code).*

RenameFile(oldFilename, newFilename)

Pass the names of two files as character strings. No wildcards are allowed in this function.

DeleteFile(filename)

Pass a character string for the filename to delete.

***Windows note**: You can pass wildcards to delete multiple files, such as "*.bmp".*

CopyFile(sourceFilename, destFilename)

Pass the names of the source and destination files as character strings.

***Windows note**: You can pass wildcards in the sourceFilename argument to copy multiple files, such as "c:*.bmp".*

GetFileModDate(filename)

This method returns the last modified time & date for the specified file as a 25-character string as follows: "Wed Jan 02 02:03:55 1980\n". The '\n' is a newline character. Note that the time is in 24-hour format and numbers are zero-padded.

(Windows)
FileOpenDialog(initialDir, filterStr, dlogTitle, createPrompt, fileMustExist)

(Mac)
FileOpenDialog(initialDir, filterStr)

This method displays a system "File Open" dialog box which allows the user to select a directory and file to open. The filename selected is returned to the caller as a string INCLUDING THE FULL PATH. If no filename was selected or Cancel was pressed, the empty string ("") is returned.

'initialDir' is the path of the directory where the dialog should be opened.

'filterStr' is a string that tells what kind of files to show in the dialog. On Windows, these consist of descriptor/extension pairs separated by '/', such as "Text Files/*.TXT/All Files/*.*". On the Macintosh, the filters are file 'types' separated by '/', such as "TEXT/WORD". There is no limit to the number of filters on Windows, but you can specify a maximum of four (4) filters on the Macintosh.

'dlogTitle' (Windows only) is the title of the Windows dialog. If you pass "", the title defaults to "Open."

'createPrompt' (Windows only) is a boolean that tells the Windows dialog to prompt the user about creating the file if the file does not already exist. For instance, the user can type the name of a file into the text box; if that file does not exist when they press the "Open" button, a dialog will appear asking them if they wish to create it. If they answer "No", the Open File dialog stays on the screen. If they answer "Yes", the filename is returned to the caller. Remember that you can use the FileExists() method to see if the file actually exists or not.

'fileMustExist' (Windows only) is a boolean that tells the Windows dialog that a user must either select a file from the list or type the name of an existing file. If they do not, a message will appear asking them to try again.

(Windows)

FileSaveAsDialog(initialDir, filename, dlogTitle, overwritePrompt)

(Mac)

FileSaveAsDialog(initialDir, filename, prompt)

This method displays a system "File Save As" dialog box that allows the user to select a directory and type in a filename to save a file under. The full path and the filename are returned to the caller if the Save button was pressed. Otherwise the empty string ("") is returned if Cancel was pressed.

'initialDir' is the path of the directory where the dialog should be opened.

'filename' is the name to show initially in the dialog box. The user can change this by typing over it.

'prompt' (Macintosh only) is the text that appears above the name of the file. If you leave this blank, it defaults automatically to "Save As:".

'dlogTitle' (Windows only) will be used as the title bar text of the Windows dialog.

'overwritePrompt' (Windows only) is a boolean that, when True, brings up a warning dialog if the user types in the name of an existing file and presses the Save button. If this flag is False then no warning is given.

Directory functions

Note: You don't have to worry about passing '\' or ':' trailing directory separator characters to these functions. They don't care if they are present or not.

DirectoryExists(dirName)

Pass a character string and a 0 will be returned if the directory exists.

Example:

```
set retVal to DirectoryExists("C:\MYDIR")
if retVal = 0 then
        alert "Just where you left it!"
else
        alert "You've lost it!"
end if
```

CreateDirectory(dirName)

This function creates a directory. If a directory or file already exists with the given path and name, the function returns an error code (see table below). A 0 is returned if successful.

DeleteDirectory(dirName)

This function will delete an empty directory only. An error code is returned if files or directories are found in the directory to be deleted.

XDeleteDirectory(dirName)

This function is powerful and dangerous. It will perform a "tree walk" to delete not only the target directory given in the dirName argument, but it will also delete all subdirectories and their files.

***Caution**! It is possible to erase an entire hard drive with the careless use of this command! Issuing a command such as XDeleteDirectory("C:\") will in fact erase drive C:! Be warned!*

CopyDirectory(sourceDirName, destDirName)

This command will copy all of the files from the source directory to the destination directory. It will not copy any subdirectories or their files found in the source directory.

XCopyDirectory(sourceDirName, destDirName)

This command does a "tree walk" and copies all files, subdirectories and their files from the source directory to the destination directory. If you need to duplicate an entire directory tree, this command is a real time saver.

***Macintosh note**: Invisible files are not copied by this or other file or directory commands.*

DirectoryToList(dirName)

This function will create a Director list and place an item in the list for each file and directory found in the given directory name.

***Macintosh note**: Macintosh folders found in the given folder have a ':' character appended.*

***Windows note**: Subdirectories found in the given directory have a '\' character appended.*

***Example**:*

```
set filelist to [ ]
set filelist to DirectoryToList("Macintosh HD:Applications")
```

Error messages:

The Xtra returns status values from each of its callable functions to let you know if the operation was successful, and if not, why. The following table lists each error code returned by the Xtra.

The calls that return a string instead of an integer status code are:

FileOpenDialog
FileSaveAsDialog
GetFileModDate

The calls that return a list instead of an integer status code are:

DrivesToList
DirectoryToList

return value	error explanation
0	Successful completion
-1	General error of unknown origin
-5	File deletion failure
-6	File rename failure
-7	File not found
-8	Specified file is actually a directory
-9	File creation failure
-10	File open failure
-11	File write failure
-12	File close failure
-13	File read failure
-14	Destination disk full
-15	Directory not found
-16	Specified directory is actually a file
-17	Directory creation failure
-18	Could not delete specified directory
-19	Could not retrieve directory ID number
-40	Could not allocate memory for file copy
-51	Specified drive does not exist
-52	Specified drive exists but is not mounted
-61	Specified drive is not a CD-ROM
-210	New filename already exists or two paths are different